Errata

Title & Document Type: 16515A/16A Programming Reference

Manual Part Number: 16515-90907

Revision Date: September 1988

HP References in this Manual

This manual may contain references to HP or Hewlett-Packard. Please note that Hewlett-Packard's former test and measurement, semiconductor products and chemical analysis businesses are now part of Agilent Technologies. We have made no changes to this manual copy. The HP XXXX referred to in this document is now the Agilent XXXX. For example, model number HP8648A is now model number Agilent 8648A.

About this Manual

We've added this manual to the Agilent website in an effort to help you support your product. This manual provides the best information we could find. It may be incomplete or contain dated information, and the scan quality may not be ideal. If we find a better copy in the future, we will add it to the Agilent website.

Support for Your Product

Agilent no longer sells or supports this product. You will find any other available product information on the Agilent Test & Measurement website:

www.tm.agilent.com

Search for the model number of this product, and the resulting product page will guide you to any available information. Our service centers may be able to perform calibration if no repair parts are needed, but no other support from Agilent is available.

Agilent Technologies

Programming Reference

HP 16515A/16516A
1 GHz Timing Analyzer Module

for the HP 16500A Logic Analysis System

© Copyright Hewlett-Packard Company 1988

Manual Part Number 16515-90907

Printed in U.S.A. September 1988

Product Warranty

This Hewlett-Packard product has a warranty against defects in material and workmanship for a period of three years from date of shipment. During warranty period, Hewlett-Packard Company will, at its option, either repair or replace products that prove to be defective.

For warranty service or repair, this product must be returned to a service facility designated by Hewlett-Packard. However, warranty service for products installed by Hewlett-Packard and certain other products designated by Hewlett-Packard will be performed at the Buyer's facility at no charge within the Hewlett-Packard service travel area. Outside Hewlett-Packard service travel areas, warranty service will be performed at the Buyer's facility only upon Hewlett-Packard's prior agreement and the Buyer shall pay Hewlett-Packard's round trip travel expenses.

For products returned to Hewlett-Packard for warranty service, the Buyer shall prepay shipping charges to Hewlett-Packard and Hewlett-Packard shall pay shipping charges to return the product to the Buyer. However, the Buyer shall pay all shipping charges, duties, and taxes for products returned to Hewlett-Packard from another country.

Hewlett-Packard warrants that its software and firmware designated by Hewlett-Packard for use with an instrument will execute its programming instructions when properly installed on that instrument. Hewlett-Packard does not warrant that the operation of the instrument software, or firmware will be uninterrupted or error free.

Limitation of Warranty

The foregoing warranty shall not apply to defects resulting from improper or inadequate maintenance by the Buyer, Buyer-supplied software or interfacing, unauthorized modification or misuse, operation outside of the environmental specifications for the product, or improper site preparation or maintenance.

NO OTHER WARRANTY IS EXPRESSED OR IMPLIED. HEWLETT-PACKARD SPECIFICALLY DISCLAIMS THE IMPLIED WARRANTIES OR MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Exclusive Remedies THE REMEDIES PROVIDED HEREIN ARE THE BUYER'S SOLE AND EXCLUSIVE REMEDIES. HEWLETT-PACKARD SHALL NOT BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL INCIDENTAL, OR CONSEQUENTIAL DAMAGES, WHETHER BASED ON CONTRACT, TORT, OR ANY OTHER LEGAL THEORY.

Assistance Product maintenance agreements and other customer assistance agreements are available for Hewlett-Packard products.

For any assistance, contact your nearest Hewlett-Packard Sales and Service Office.

Certification Hewlett-Packard Company certifies that this product met its published specifications at the time of shipment from the factory. Hewlett-Packard further certifies that its calibration measurements are traceable to the United States National Bureau of Standards, to the extent allowed by the Bureau's calibration facility, and to the calibration facilities of other International Standards Organization members.

Safety This product has been designed and tested according to International Safety Requirements. To ensure safe operation and to keep the product safe, the information, cautions, and warnings in this manual must be heeded.

Table of Contents

Chapter 1: **Programming the HP 16515A/16516A**

1-1 Introduction
1-1 About This Book
1-2 Programming the 1 GHz Timing Analyzer
1-2 Selecting the Module
1-2 Programming the Analyzer
1-5 Mainframe Commands
1-5 CARDcage? Query
1-6 MENU Command/query
1-6 RMODe Command/query
1-6 SELect Command/query
1-6 STARt Command
1-6 STOP Command
1-7 SYSTem:ERRor? Query
1-7 SYSTem:PRINt Command/query
1-7 MMEMory Subsystem
1-7 INTermodule Subsystem
1-8 Command Set Organization
1-11 Module Status Reporting
1-12 MESE
1-14 MESR

Chapter 2: **FORMat Subsystem**

2-1 Introduction
2-3 LABel
2-5 REMove
2-6 THReshold

Chapter 3:

TRACe Subsystem

3-1 Introduction
3-3 DURation
3-5 EDGE
3-7 MATCh
3-8 PATTern

Chapter 4:

WAVeform Subsystem

4-1 Introduction
4-6 ACCumulate
4-7 DELay
4-8 INSert
4-10 MINus
4-11 MMODe
4-12 OCONdition
4-13 OPATtern
4-15 OSEarch
4-16 OTIMe
4-17 OVERlay
4-18 PLUS
4-19 RANGe
4-20 REMove
4-21 RUNTil
4-23 SPERiod
4-24 TAVerage
4-25 TMAXimum
4-26 TMINimum
4-27 VRUNs
4-28 XCONdition
4-29 XOTime
4-30 XPATtern
4-32 XSEarch
4-33 XTIMe

Chapter 5: **SYMBol Subsystem**
5-1 Introduction
5-3 BASE
5-4 PATTern
5-5 RANGe
5-6 REMove
5-7 WIDTh

Appendix A: **DATA and SETup Commands**
A-1 Introduction
A-2 SYSTem:DATA
A-3 Definition of Block Data
A-4 Data Command Configuration
A-7 SYSTem:SETup
A-7 Definition of Block Data

Index

Programming the HP 16515A/16516A

1

Introduction

This manual, combined with the *HP 16500A Programming Reference* manual, provides you with the information needed to program the HP 16515A/16516A 1 GHz Timing Analyzer module. Each module has its own programming manual to supplement the mainframe manual since not all mainframes will be configured with the same modules.

About This Book

This manual is organized into five chapters. The first chapter contains:

- General information and instructions to help you get started
- Mainframe system commands that are frequently used with the 1 GHz timing analyzer module
- The HP 16515A/16516A 1 GHz Timing Analyzer command tree
- Alphabetic command-to-subsystem directory

Chapters two through five contain all the subsystem commands for the HP 16515A/16516A 1 GHz Timing Analyzer.

Appendix A contains information on the SYSTem:DATA and SYSTem:SETup commands for this module.

Error messages for the HP 16515A/16516A 1 GHz Timing Analyzer are included in the system error messages in Appendix C of the *HP 16500A Programming Reference* manual.

Programming the 1 GHz Timing Analyzer

This section introduces you to the basic command structure used to program the 1 GHz timing analyzer. Also included is an example program that sets up the timing analyzer for a measurement.

Selecting the Module

Before you can program the 1 GHz timing analyzer, you must first "select" it. This directs your commands to the appropriate module.

To select the module, use the mainframe command :SELect followed by the numeric reference for the slot location of the master card for the 1 GHz timing analyzer (1...5 corresponds to slot A...E respectively). For example, if the master card for your 1 GHz timing analyzer is in slot B, then the command:

```
:SELECT 2
```

would select this module. For more information on the select command, refer to the *HP 16500A Programming Reference* manual.

Programming the Analyzer

A typical 1 GHz timing analyzer program would:

- select the appropriate module
- assign labels
- set pod thresholds
- specify a trigger condition
- set up the display
- specify the acquisition type
- start the acquisition

The following example program sets up the timing analyzer to make a simple measurement.

Example Program:

```
10 OUTPUT XXX;":SELECT 4"
20 OUTPUT XXX;":FORMAT:LABEL 'A',POS,1"
30 OUTPUT XXX;":FORMAT:LABEL 'B',POS,2"
40 OUTPUT XXX;":FORMAT:THRESHOLD2 TTL"
50 OUTPUT XXX;":TRACE:MATCH EQUAL"
60 OUTPUT XXX;":TRACE:PATTERN 'A','#B0"
70 OUTPUT XXX;":TRACE:PATTERN 'B','#B1"
80 OUTPUT XXX;":TRACE:DURATION GT, 35ns"
90 OUTPUT XXX;":TRACE:EDGE 'A','X"
100 OUTPUT XXX;":TRACE:EDGE 'B','F"
110 OUTPUT XXX;":RMODE SINGLE"
120 OUTPUT XXX;":START"
130 OUTPUT XXX;":MENU 4,2"
140 END
```

Note

The three Xs (XXX) after the "OUTPUT" statements in the previous examples refer to the device address required for programming over either HP-IB or RS-232C. Refer to your controller manual and programming language reference manual for information on initializing the interface.

Program Comments

Line 10 selects the 1 GHz timing analyzer in slot D.
Line 20 assigns channel 0 of pod D2 (master card pod 2) to the label "A" and assigns a positive polarity to this label.
Line 30 assigns channel 1 of pod D2 to the label "B" and assigns a positive polarity to this label.
Line 40 sets the threshold level for pod D2 to TTL.
Lines 50 to 100 configures the analyzer to find the pattern where both the signal for label "A" is low and the signal for label "B" is high for greater-than 35 ns. When it finds this pattern, the analyzer will trigger on the falling edge of the signal on label "B."
Lines 110 and 120 tell the analyzer to run the measurement one time.
Line 130 displays the 1 GHz timing analyzer's Waveforms menu so you can view the results of your measurement.

For more information on the specific 1 GHz timing analyzer commands, refer to chapters 2 through 5 of this manual.

Mainframe Commands

These commands are part of the HP 16500A mainframe system and are mentioned here only for reference. For more information on these commands refer to the *HP 16500A Programming Reference* manual.

CARDcage? Query

The CARDcage query returns a series of integers which identifies the modules that are installed in the mainframe. The returned string is in two parts. The first five numbers returned identify the card type . The identification number for the HP 16515A master card is 1 and the identification number for the HP 16516A expansion card is 2. A "-1" in the first part of the string indicates that no card is installed in the slot.

The five numbers in the second part of the string indicate which slots have cards installed, which card has the controller software for the module, and where the master card is located.

Example: `-1,-1,2,1,31,0,0,4,4,5`

The first five numbers in the string shown in the example indicate that no cards are installed in slot A and B. An HP 16516A expansion card (ID number 2) is loaded in slot C and an HP 16515A master card (ID number 1) is loaded into slot D. Slot E contains an HP 16510A logic analyzer (ID number 31).

The last five numbers (0,0,4,4,5) indicate that a two card module is located in slots C and D with the master card in slot D. Also, a single module card is located in slot E. The "0" indicates that no card is installed in slot A or B or the module software is not recognized or not loaded. For more information on the CARDcage query, refer to the *HP 16500A Programming Reference* manual.

MENU
Command/query

The MENU command selects which menu will be displayed. The first parameter specifies the desired module. The optional second parameter specifies the desired menu in the module (defaults to 0 if not specified). The query returns the currently displayed menu.

For the HP 16515A/16516A 1 GHz Timing Analyzer:

- MENU X,0 - Format menu
- MENU X,1 - Trace menu
- MENU X,2 - Waveforms menu

Where X refers to the slot where the master card for this module is installed.

RMODe
Command/query

The RMODe command specifies the run mode (single or repetitive) for a module or intermodule. If the selected module is in the intermodule configuration, the intermodule run mode will be set by this command. The RMODe query returns the current setting.

SELect
Command/query

The SELect command selects which module (or intermodule) will have parser control. SELECT 0 selects intermodule, SELECT 1 through 5 selects modules A through E respectively. The values -1 and -2 select software options 1 and 2. The SELect query returns the value for the currently selected module.

STARt
Command

The STARt command starts the selected module or intermodule running. If the specified module is in the intermodule configuration, the STARt command will start all of the modules in the intermodule configuration running.

STOP
Command

The STOP command stops the selected module or intermodule. If the specified module is in the intermodule configuration, the STOP command will stop all of the modules in the intermodule configuration.

Note

*The STARt and STOP commands are Overlapped Commands. An Overlapped Command is a command that allows execution of subsequent commands while the device operations initiated by the Overlapped Command are still in progress. For more information, refer to the *OPC and *WAI commands in the HP 16500A Programming Reference manual.*

SYSTem:ERRor? Query

The SYSTem:ERRor query returns the oldest error in the error queue. In order to return all of the errors in the error queue, a simple FOR/NEXT loop can be written to query the queue until all of the errors are returned. When all of the errors are returned, the query returns zeros.

For a complete list of error messages, refer to the *HP 16500A Programming Reference* manual.

SYSTem:PRINt Command/query

The SYSTem:PRINt command initiates a print of the screen or listing buffer over the current printer communication interface. The SYSTem:PRINt query sends the screen or listing buffer data over the current controller communication interface.

MMEMory Subsystem

The MMEMory subsystem provides access to both internal disc drives for loading and storing configurations.

INTermodule Subsystem

The INTermodule subsystem commands are used to specify intermodule arming between multiple modules.

Command Set Organization

The command set for the HP 16515A/16516A is divided into four separate groups. Each of these groups of commands is described in the following chapters.

Each of the chapters contain a brief description of the subsystem, syntax diagrams, and the commands for that subsystem in alphabetical order. The commands are shown in longform and shortform using upper and lowercase letters. For example, LABel indicates that the longform of the command is LABEL and the shortform of the command is LAB. Each of the commands contain a description of the command and its arguments, the command syntax, and a programming example.

Figure 1-1 is a command tree that summarizes of all of the commands for the HP 16515A/16516A 1 GHz Timing Analyzer module. The (x) in the command tree figure represents the slot number in which the master card for the 1 GHz timing analyzer module is located.

where: SELect 0 = System
SELect 1 = Slot A
SELect 2 = Slot B
SELect 3 = Slot C
SELect 4 = Slot D
SELect 5 = Slot E

Table 1-1 is an alphabetical command to subsystem directory.

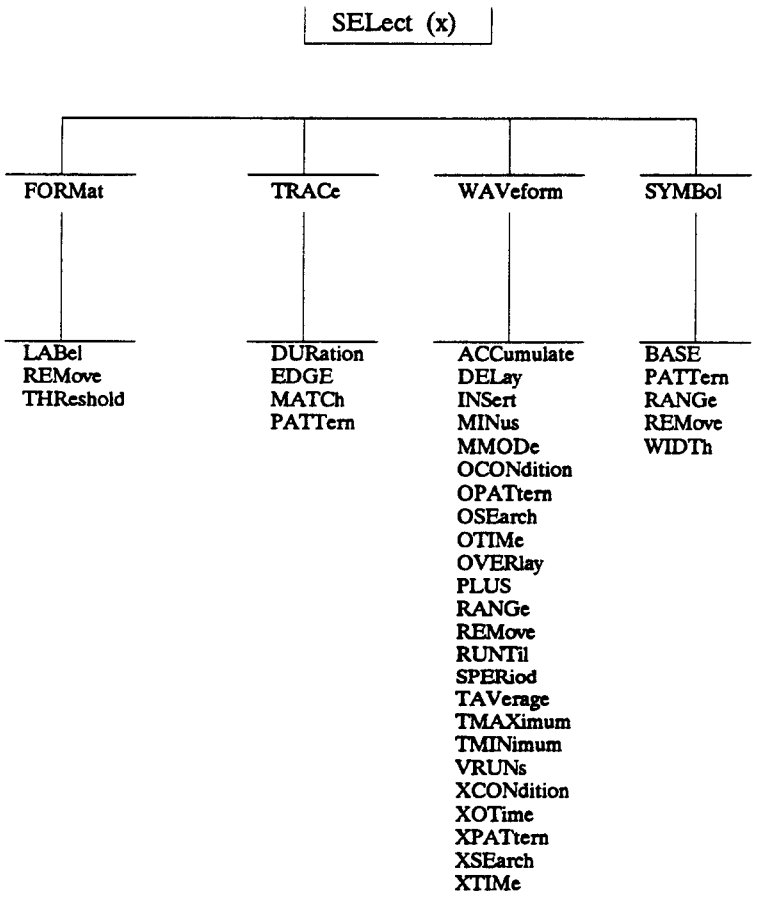


Figure 1-1. HP 16515A/16516A Command Tree

Table 1-1. Alphabetic Command Cross-Reference

Command	Where Used	Command	Where Used
ACCumulate	Waveform Subsystem	RANGe	Waveform Subsystem
BASE	Symbol Subsystem	REMove	Format Subsystem
DELay	Waveform Subsystem	REMove	Symbol Subsystem
DURation	Trace Subsystem	REMove	Waveform Subsystem
EDGE	Trace Subsystem	RUNTil	Waveform Subsystem
FORMat	Subsystem Selector	SPERiod	Waveform Subsystem
INSert	Waveform Subsystem	SYMBol	Subsystem Selector
LABel	Format Subsystem	TAVerage	Waveform Subsystem
MATCh	Trace Subsystem	THReshold	Format Subsystem
MINus	Waveform Subsystem	TMAXimum	Waveform Subsystem
MMODe	Waveform Subsystem	TMINimum	Waveform Subsystem
OCONdition	Waveform Subsystem	TRACe	Subsystem Selector
OPATtern	Waveform Subsystem	VRUNs	Waveform Subsystem
OSEarch	Waveform Subsystem	WAVeform	Subsystem Selector
OTIMe	Waveform Subsystem	WIDTh	Symbol Subsystem
OVERlay	Waveform Subsystem	XCONdition	Waveform Subsystem
PATTern	Symbol Subsystem	XOTime	Waveform Subsystem
PATTern	Trace Subsystem	XPATtern	Waveform Subsystem
PLUS	Waveform Subsystem	XSEarch	Waveform Subsystem
RANGe	Symbol Subsystem	XTIMe	Waveform Subsystem

Module Status Reporting

Each module reports its status to the Module Event Status Register (MESR) which in turn reports to the Combined Event Status Register (CESR) in the HP 16500A mainframe (see chapter 6 of the *HP 16500A Programming Reference* manual). The Module Event Status Register is enabled by the Module Event Status Enable register (MESE).

The following descriptions of the MESE and MESR commands provide the module specific information required to enable and interpret the contents of the registers.

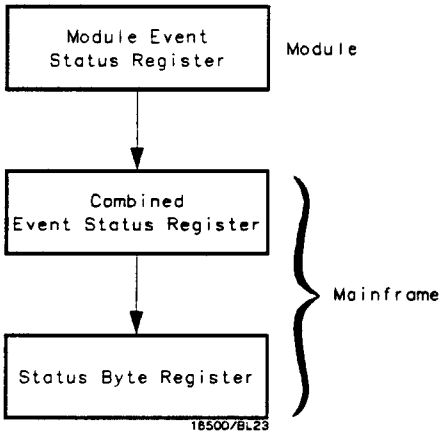


Figure 1-2. Module Status Reporting

MESE command/query

The MESE command sets the Module Event Status Enable register bits. The MESE register contains a mask value for the bits enabled in the MESR register. A one in the MESE register will enable the corresponding bit in the MESR. A zero will disable the bit.

The first parameter specifies the module, and the second parameter specifies the enable value. 1...5 corresponds to the module in slot A...E.

The MESE query returns the current setting.

Refer to table 1-2 for information about the Module Event Status Enable register bits, bit weights, and what each bit masks for this module. For more information on status reporting refer to the *HP 16500A Programming Reference* manual.

Command Syntax: :MESE<N> <enable_mask>

where:

<N> ::= {1|2|3|4|5} number of slot in which the module resides
<enable_mask> ::= 0 to 65535 (integer)

Example: OUTPUT XXX;":MESE4 2"

MESE

Query Syntax: :MESE<N>?

Returned Format: [:MESE<N>] <enable_mask> <NL>

Example:
```
10 DIM Mese$[100]
20 OUTPUT XXX;":MESE4?"
30 ENTER XXX;Mese$
40 PRINT Mese$
50 END
```

Table 1-2. HP 16515A/16516A Module Event Status Enable Register

Bit	Weight	Enables
7	128	not used
6	64	not used
5	32	not used
4	16	not used
3	8	not used
2	4	TRG - Trigger Received
1	2	RNT - Run Until Satisfied
0	1	MC - Measurement Complete

The Module Event Status Enable Register contains a mask value for the bits to be enabled in the Module Event Status Register (MESR). A one in the MESE enables the corresponding bit in the MESR, a zero disables the bit.

MESR query

The MESR query returns the current contents of the Module Event Status register.

Note

Reading the register clears the bits of the Module Event Status Register.

Table 1-3 shows each bit in the Module Event Status Register and their bit weights for this module. When you read the MESR, the value returned is the total bit weights of all bits that are high at the time the register is read.

The parameter specifies the module. 1...5 corresponds to the module in slot A...E respectively.

Query Syntax: :MESR<N>?

Returned Format: [:MESR<N>] <status> <NL>

where:

<N> ::= {1|2|3|4|5} number of slot in which the module resides
<status> ::= 0 to 65535 (integer)

Example:

```
10 DIM Mesr$[100]
20 OUTPUT XXX;":MESR5?"
30 ENTER XXX;Mesr$
40 PRINT Mesr$
50 END
```

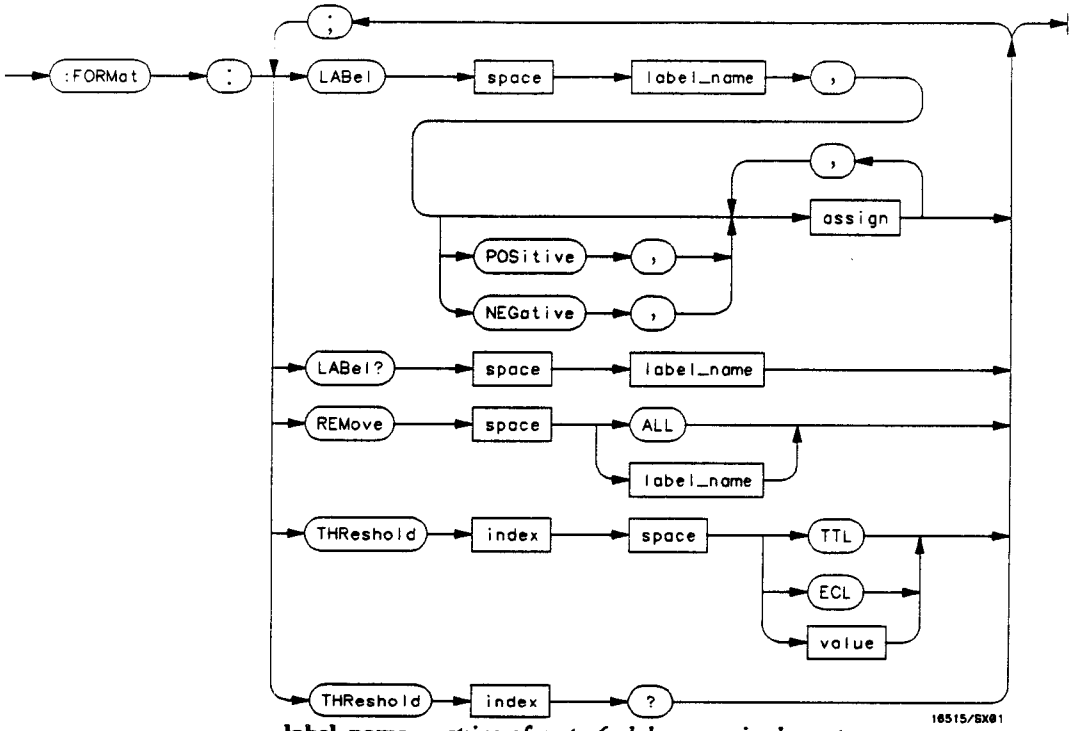
Table 1-3. HP 16515A/16516A Module Event Status Register

Bit	Bit Weight	Bit Name	Condition
7	128		0 = not used
6	64		0 = not used
5	32		0 = not used
4	16		0 = not used
3	8		0 = not used
2	4	TRG	0 = Trigger Not Received 1 = Trigger Received
1	2	RNT	0 = Run Until Not Satisfied 1 = Run Until Satisfied
0	1	MC	0 = Measurement Not Complete 1 = Measurement Complete

FORMat Subsystem 2

Introduction

The FORMat subsystem commands allow you to configure the Format menu of the 1 GHz timing analyzer. These commands specify how the analyzer is connected to the system under test. This includes renaming labels, assigning polarity, grouping channels into different labels, and setting the sampling threshold voltages for each pod. Figure 2-1 shows the syntax diagram for these commands.



label_name = *string of up to 6 alphanumeric characters.*
assign = *integer, 0 to 255.*
index = *integer, 1 to 4.*
value = *real number, -3.5 to +5.0.*

Figure 2-1. Format Subsystem Commands Syntax Diagram

LABel command/query

The LABel command allows you to specify polarity, create labels, and assign channels to new or existing labels. If the specified label name does not match an existing label name, a new label will be created.

The pod order and assignment (number) is significant since each assignment parameter refers to a separate pod. The first two assignment parameters are for pods 2 and 1 (respectively) of the HP 16515A master card. The last two assignment parameters are for pods 2 and 1 (respectively) of the HP 16516A expansion card. When the assignment value is expressed in binary (base 2), each bit of the value represents one of the channels of the associated pod. Bit 0 represents channel 0, bit 1 represents channel 1, bit 2 represents channel 2, etc. A "1" in a bit position means the associated channel in the pod is assigned to the label. A "0" in the bit position means the associated channel in the pod is excluded from the label. If an assignment value is not specified, no channels are assigned to the label.

The LABel query returns the current settings for the specified label. If the label does not exist, no value is returned.

Command Syntax: :FORMat:LABel <label_name>[,{POSitive|NEGative}],<assign>[,<assign>,...]

where:

<label_name> ::= string of up to 6 alphanumeric characters
<assign> ::= 0 through 255 (integer)

Examples: The following example assigns bits 1 and 0 of pod 2 and bit 1 of pod 1 to the label A with a positive polarity..

```
OUTPUT XXX;":FORMAT:LABEL 'A',POSITIVE, 3,2"
```

The following example assigns bits 1 and 0 of pod 1 to the label CAS.

```
OUTPUT XXX;":FORM:LAB 'CAS',0,3"
```

Query Syntax: :FORMat:LABel? <label_name>

Returned Format:
```
[:FORMat:LABel]
<label_name>,{POSitive|NEGative},<assign>[,<assign>...]<NL>
```

Example:
```
10 DIM Setting$[100]
20 OUTPUT XXX;":FORM:LABEL? 'A'"
30 ENTER XXX;Setting$
40 PRINT Setting$
50 END
```

REMove command

The REMove command either deletes all the labels or a specified label from the Format menu.

Command Syntax: :FORMat:REMove {ALL | <label_name>}

Where:

<label_name> :: = string of up to 6 alphanumeric characters

Examples:
```
OUTPUT XXX;":FORMAT:REMOVE ALL"
OUTPUT XXX;":FORM:REM 'A'"
```

THReshold command/query

The THReshold command specifies a threshold level for a particular 1 GHz timing analyzer pod. The <N> parameter is an index which specifies which pod you want to set the threshold on. A 1 or 2 refers to the HP 16515A master card pods 1 and 2 respectively. A 3 or 4 refers to the HP 16516A expansion card pods 1 and 2 respectively. The threshold can be set to standard TTL or ECL levels, or a specific voltage between +5.0 and -3.5 volts in 0.1 volt increments.

The THReshold query returns the specified pod's current threshold setting.

Command Syntax: :FORMat:THReshold <N> {TTL|ECL|<value>}

where:

<N> ::= 1 through 4 (integer)
<value> ::= voltage -3.5 to +5.0 (real number)
TTL ::= default value of +1.5 V
ECL ::= default value of -1.3 V

Examples:
```
OUTPUT XXX;":FORMAT:THRESHOLD2 TTL"
OUTPUT XXX;":FORMAT:THRESHOLD2 -3.5V"
```

Query Syntax: :FORMat:THReshold < N > ?

Returned Format: [:FORMat:THReshold < N >] {TTL | ECL | < value > } < NL >

Example:

```
10 DIM Setting$[100]
20 OUTPUT XXX;":FORMAT:THR2?"
30 ENTER XXX;Setting$
40 PRINT Setting$
50 END
```

TRACe Subsystem 3

Introduction

The TRACe subsystem contains the commands available for configuring the Trace menu of the 1 GHz timing analyzer. These commands are used to specify the trigger point for the analyzer. This includes specifying patterns, edges, and pattern durations. Figure 3-1 shows the syntax diagram for these commands.

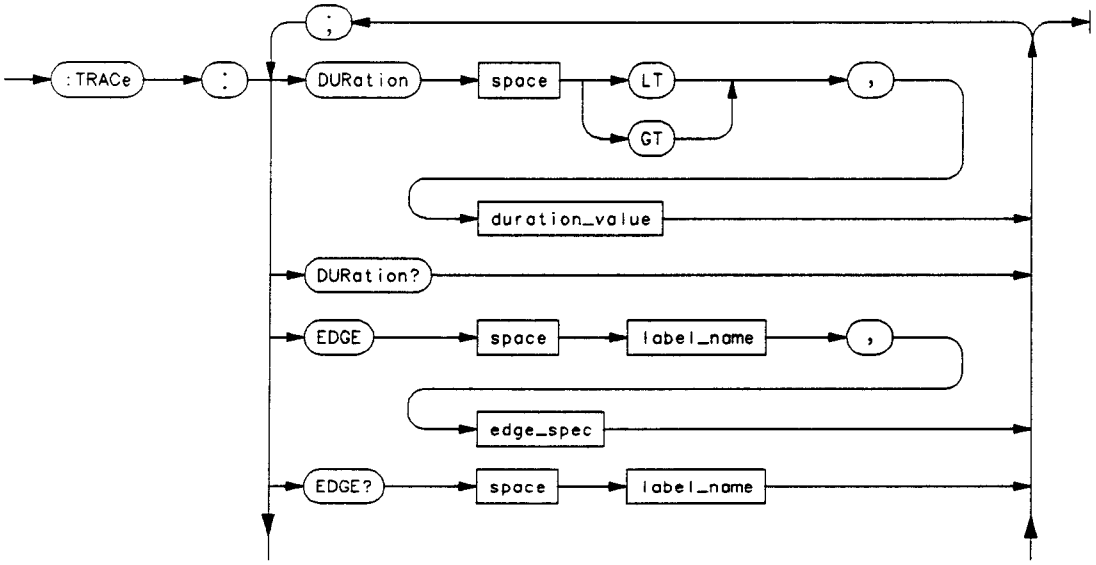
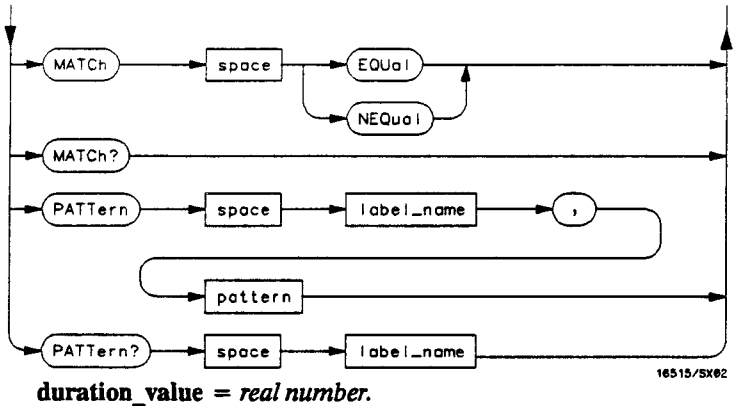


Figure 3-1. Trace Subsystem Commands Syntax Diagram



duration_value = *real number.*
label_name = *string of up to 6 alphanumeric characters.*
edge_spec = {*R*|*F*|*X*}.
pattern = *string in one of the following forms:*
"#B01X" for binary
"#Q01234567X" for octal
"#H0123456789ABCDEFX" for hexadecimal
"0123456789" for decimal
(don't cares "X" not allowed in decimal)

Figure 3-1. Trace Subsystem Commands Syntax Diagram (Continued)

DURation command/query

The DURation command specifies the amount of time the specified pattern must be valid. The first parameter specifies whether the pattern must be valid for Greater-Than (GT) or Less-Than (LT) the specified duration. The second parameter specifies the duration value. The DURation query returns the current pattern duration setting.

Note

The values for pattern duration follow a preset sequence which varies depending on the current sample rate and duration mode (less-than or greater-than). To view the value for your current configuration, enter the value you want with the DURation command. Then send the DURation query to view the actual displayed value.

If a pattern is specified across boards (HP 16515A/16516A), less-than (LT) duration is no longer available.

Command Syntax: :TRACe:DURation {LT|GT},<duration_value>

where:

GT ::= greater than
LT ::= less than
<duration_value> ::= real number

Example:
```
OUTPUT XXX;":TRACE:DURATION LT,10ns"
```

DURation

Query Syntax: :TRACe:DURation?

Returned Syntax: [TRACe:DURation] {LT|GT},<duration_value><NL>

Example:
```
10 DIM Setting$[100]
20 OUTPUT XXX;":TRACE:DURATION?"
30 ENTER XXX:Setting$
40 PRINT Setting$
50 END
```

EDGE

command/query

The EDGE command specifies the edge patterns for the 1 GHz timing analyzer to trigger on. Each command deals with only one label. Therefore, an edge specification over multiple labels will require multiple commands. The first parameter specifies the label and the second parameter specifies the edge specification for the given label. The edge specification uses the characters R, F, and X to indicate the edges or don't cares as follows:

- "R" - Rising edge
- "F" - Falling edge
- "X" - don't care (ignore the channel)

The position of these characters in the string correspond with the position of the channels in the label. All channels without Xs are ORed together to form the edge specification. The EDGE query returns the current edge specification for the specified label.

Command Syntax: :TRACe:EDGE <label_name>,<edge_spec>

where:

<label_name> ::= string of up to 6 alphanumeric characters
<edge_spec> ::= string of characters {R|F|X}

Example: OUTPUT XXX;":TRACE:EDGE 'A','XRFXXXRFXXX"

Note

The <edge_spec> must contain precisely the same number of characters as there are channels in specified label.

EDGE

Query Syntax: :TRACe:EDGE? <label_name>

Returned Format: [:TRACe:EDGE] <label_name>,<edge_spec><NL>

Example:

```
10 DIM Pattern$[100]
20 OUTPUT XXX;":TRACE:EDGE? 'A'"
30 ENTER XXX;Pattern$
40 PRINT Pattern$
50 END
```

MATCh

command/query

The MATCh command specifies whether the analyzer will trigger on a pattern equal (or not equal) to the pattern specified in the Trace menu. The MATCh query returns the current setting.

Note

Selecting NEQual (not equal) with the MATCh command forces the duration setting to Greater-Than (GT).

Command Syntax: :TRACe:MATCh {EQUal|NEQual}

Example:
```
OUTPUT XXX;":TRACE:MATCH EQUAL"
```

Query Syntax: :TRACe:MATCh?

Returned Format: [:TRACe:MATCh] {EQUal|NEQual}<NL>

Example:
```
10 DIM Setting$[100]
20 OUTPUT XXX;":TRACE:MATCH?"
30 ENTER XXX;Setting$
40 PRINT Setting$
50 END
```

PATTern command/query

The PATTern command specifies trace patterns for the 1 GHz timing analyzer. The first parameter specifies the label and the second parameter specifies the pattern for the label. Since each command deals with only one label, a complete pattern specification over multiple labels will require multiple commands. Each label can contain up to 32 bits with a range between 0 and FFFFFFFF hexadecimal. Since a pattern can contain don't cares (X) and be represented in several bases, the pattern specification parameter is handled as a string of characters.

Note

If a pattern is specified across boards (HP 16515A/16516A), less-than (LT) duration is no longer available.

The PATTern query returns the current pattern for the specified label in the base defined for the label.

Note

If negative polarity has been specified by the LABel command (see the LABel command in the chapter "Format Subsystem"), a 1 bit in a pattern specifies a level below threshold at the pod and a 0 bit specifies a level above threshold at the pod.

Command Syntax: :TRACe:PATTern <label_name>,<pattern>

where:

<label_name> ::= string of up to 6 alphanumeric characters

<pattern> ::= string in one of the following forms:
"#B01X" for binary
"#Q01234567X" for octal
"#H0123456789ABCDEFX" for hexadecimal
"0123456789" for decimal
(don't cares "X" not allowed in decimal)

Example: OUTPUT XXX;":TRACE:PATTERN 'A','#B01X00X01'"

Query Syntax: :TRACe:PATTern? <label_name>

Returned Format: [:TRACe:PATTern] <label_name>,<pattern><NL>

Example:
```
10 DIM Pattern$[100]
20 OUTPUT XXX;":TRACE:PATTERN? 'A'"
30 ENTER XXX;Pattern$
40 PRINT Pattern$
50 END
```

WAVeform Subsystem 4

Introduction

The WAVeform subsystem contains all of the commands available for the HP 16515A/16516A Waveforms menu. These include commands for setting up the waveform display and making marker measurements. Figure 4-1 shows the syntax diagram for these commands.

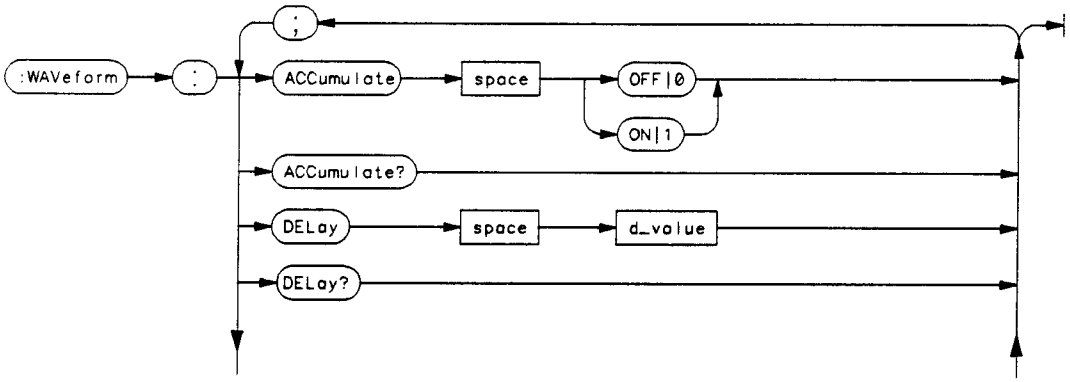


Figure 4-1. Waveform Subsystem Commands Syntax Diagram

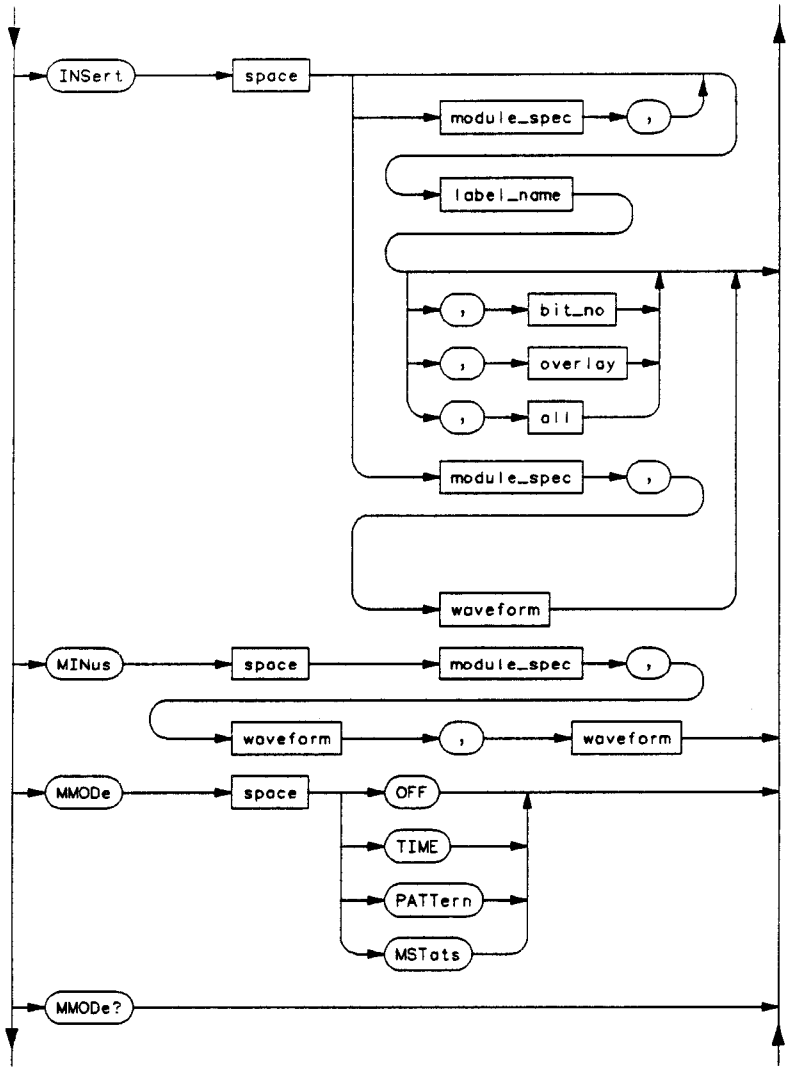


Figure 4-1. Waveform Subsystem Commands Syntax Diagram (Continued)

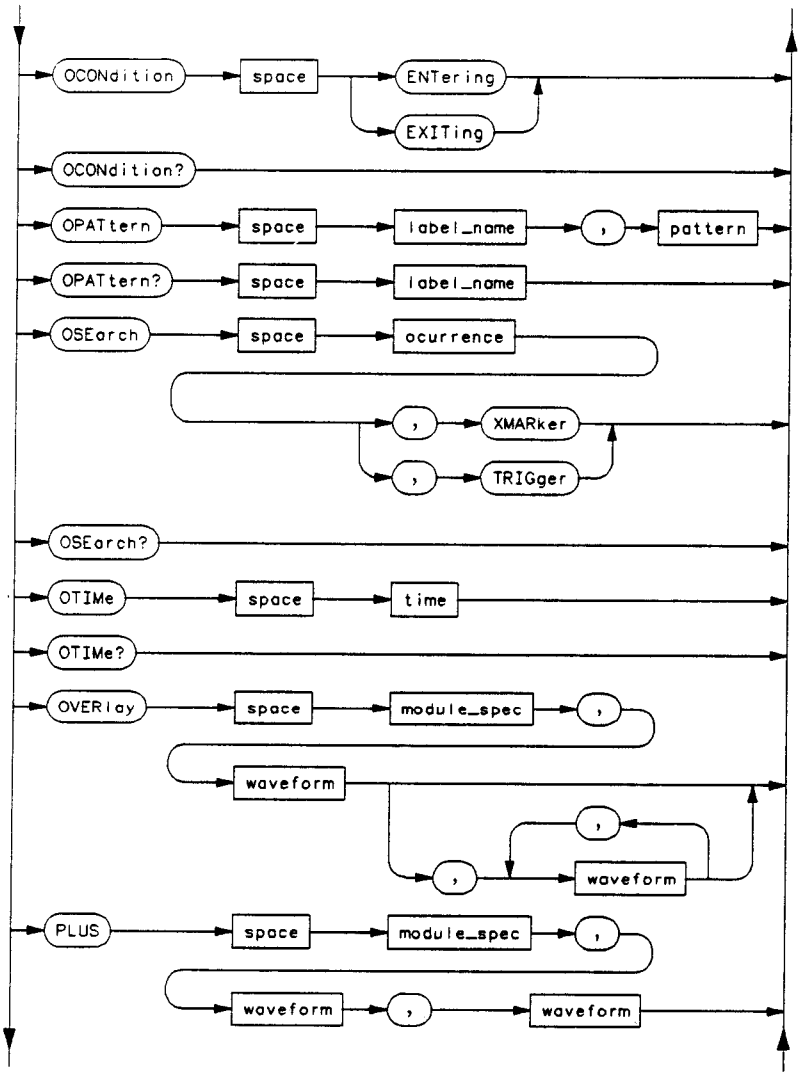


Figure 4-1. Waveform Subsystem Commands Syntax Diagram (Continued)

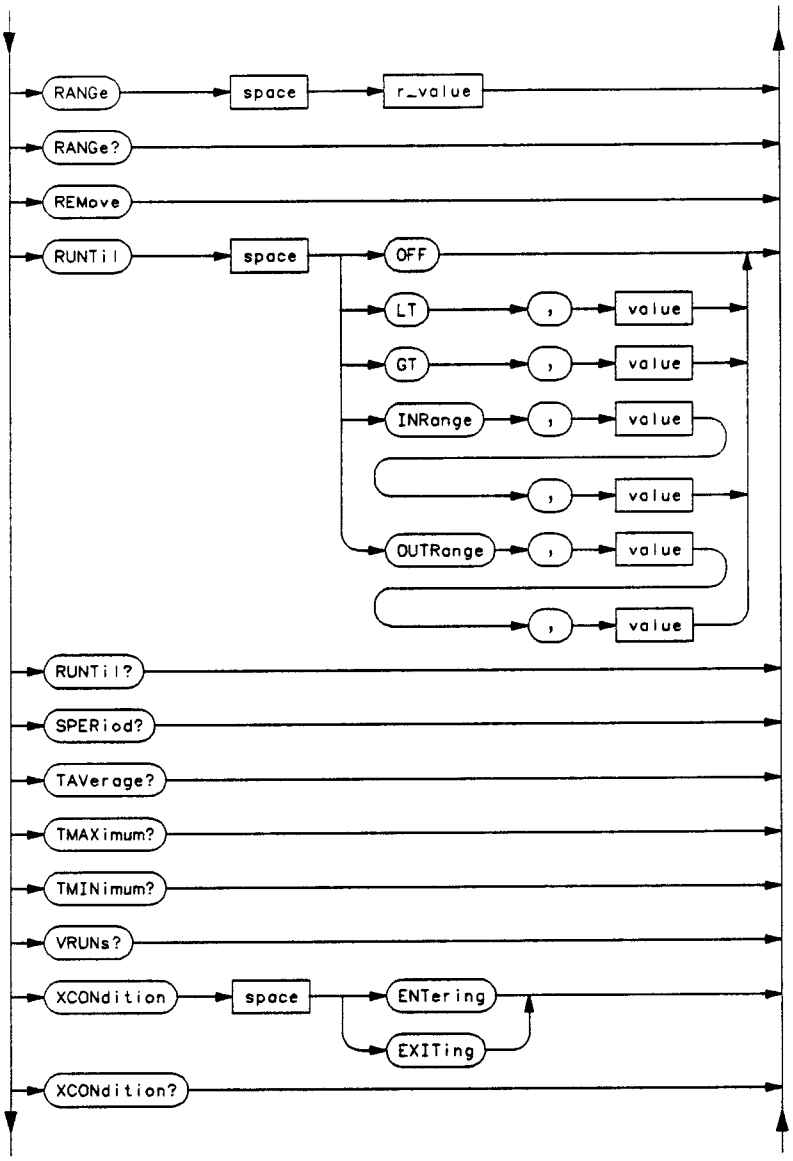
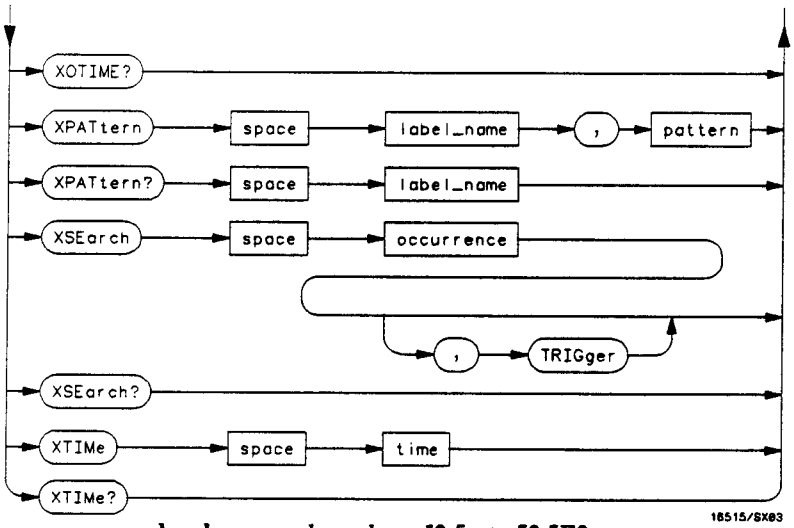


Figure 4-1. Waveform Subsystem Commands Syntax Diagram (Continued)



d_value = *real number, -12.5 s to 53.5E3 s.*
module_spec = *integer, 1 to 5.*
label_name = *string of up to 6 alphanumeric characters.*
bit_no = *integer, 0 to 31.*
waveform = *string of one alpha and one numeric character.*
pattern = *string in one of the following forms:*
"#B01X" for binary
"#Q01234567X" for octal
"#H0123456789ABCDEFX" for hexadecimal
"0123456789" for decimal
(don't cares "X" not allowed in decimal)
occurrence = *integer, -8192 to 8192.*
time = *real number, -13.0 s to 53.5E3 s.*
r_value = *real number, 10E-9 s to 10 s.*
value = *real number, -53.5E3 s to 53.5E3 s.*

Figure 4-1. Waveform Subsystem Commands Syntax Diagram (Continued)

ACCumulate command/query

The ACCumulate command controls whether the waveform display gets erased between individual runs or whether it accumulates. The ACCumulate query returns the current setting.

Command Syntax: :WAVeform:ACCumulate {{ON|1}|{OFF|0}}

Example:
```
OUTPUT XXX;":WAVEFORM:ACCUMULATE ON"
```

Query Syntax: :WAVeform:ACCumulate?

Returned Format: [:WAVeform:ACCumulate] {0|1}<NL>

Example:
```
10 DIM Setting$[100]
20 OUTPUT XXX;":WAVEFORM:ACC?"
30 ENTER XXX;Setting$
40 PRINT Setting$
50 END
```

DELay command/query

The DELay command specifies the amount of time between the timing trigger and the horizontal center of the screen in the waveforms display. The allowable values for delay are -12.5 s to +53.5 ks. The DELay query returns the current offset value (delay) from the trigger.

Command Syntax: :WAVeform:DELay <d_value>

where:

<d_value> ::= -12.5 s to +53.5E3 s (real number)

Example:
```
OUTPUT XXX;":WAVEFORM:DELAY 400E-9"
```

Query Syntax: :WAVeform:DELay?

Returned Format: [:WAVeform:DELay] <d_value><NL>

Example:
```
10 DIM Delay_value$[100]
20 OUTPUT XXX;":WAVEFORM:DELAY?"
30 ENTER XXX;Delay_value$
40 PRINT Delay_value$
50 END
```

INSert command

The INSert command inserts time correlated waveforms in the waveform display. The waveforms are added directly below the last waveform on screen. When the maximum number of waveforms are present on screen (24), inserting an additional waveform replaces the last waveform on screen.

Time correlated waveforms from the oscilloscope and other timing analyzers in the HP 16500A system can also be inserted in the 1 GHz timing analyzer's display. When inserting waveforms from the oscilloscope or other timing analyzer modules, the optional first parameter must be used. 1...5 corresponds to the modules in slot A...E. If the module specifier is not selected, the currently selected module is assumed.

The second parameter specifies the label for the waveforms to be added.

The optional third parameter specifies the label bit number, OVERlay, or ALL. If a number is specified, only the waveform for that bit number is added to the screen.

If OVERlay is specified, all the bits of the label are displayed as a composite overlayed waveform. If ALL is specified, all the bits of the label are displayed sequentially. If the third parameter is not specified, ALL is assumed.

To insert a waveform from another timing analyzer to the 1 GHz timing analyzer's display:

Command Syntax: :WAVeform:INSert [<module_spec>,]<label_name>[,{<bit_no>|OVERlay|ALL}]

where:

<module_spec> ::= 1 through 5 (integer)
<label_name> ::= string of up to 6 alphanumeric characters
<bit_no> ::= 0 through 31 (integer)

Examples:
```
OUTPUT XXX;":WAVEFORM:INSERT 2,'A',OVERLAY"
OUTPUT XXX;":WAV:INSERT 'A'"
```

To insert a waveform from an oscilloscope module to the 1 GHz timing analyzer's display:

Command Syntax: :WAVeform:INSert <module_spec>,<waveform>

where:

<module_spec> ::= 1 through 5 for slot in which timebase card is installed (integer)
<waveform> ::= <acquisition_spec>{1|2} (string format)
<acquisition_spec> ::= {A|B|C|D} (slot where acquisition card is located)

Example:
```
OUTPUT XXX;":WAVEFORM:INSERT 4,'C1'"
```

MINus command

The MINus command inserts time correlated A-B (A minus B) oscilloscope waveforms on the screen. The first parameter is the module specifier where 1...5 corresponds to the slot A...E in which the module is located. The next two parameters specify the oscilloscope waveforms to be subtracted from each other.

Command Syntax: :WAVeform:MINus <module_spec>,<waveform>,<waveform>

where:

<module_spec> ::= 1 through 5 for slot in which oscilloscope timebase card is installed (integer)
<waveform> ::= <acquisition_spec>{1|2}
<acquisition_spec> ::= {A|B|C|D} (slot where acquisition card is located)

Example: OUTPUT XXX;":WAVEFORM:MINUS 2,'A1','B1'"

Note

This command is only available for oscilloscope waveforms from an oscilloscope module in the same mainframe.

MMODe command/query

The MMODe (Marker Mode) command selects the mode for controlling marker movement and the display of marker readouts. When the marker mode is OFF, the markers do not appear on screen and the current sample period is displayed. When PATTern is selected, the markers are placed on the specified patterns. When TIME is selected, the markers are positioned in time. In MSTats, the markers are placed on patterns, but the readouts will be time statistics.

The MMODe query returns the current marker mode.

Command Format: :WAVeform:MMODe {OFF|TIME|PATTern|MSTats}

Example:
```
OUTPUT XXX;":WAVEFORM:MMODE TIME"
```

Query Syntax: :WAVeform:MMODe?

Returned Format: [:WAVeform:MMODe] {OFF|TIME|PATTern|MSTats}<NL>

Example:
```
10 DIM Setting$[100]
20 OUTPUT XXX;":WAVEFORM:MMODE?"
30 ENTER XXX;Setting$
40 PRINT Setting$
50 END
```

OCONdition command/query

The OCONdition command specifies whether the marker is to be placed on the entry or exit point of the pattern when you are in the PATTern marker mode. The OCONdition query returns the current setting.

Command Syntax: :WAVeform:OCONdition {ENTering|EXITing}

Example: OUTPUT XXX;":WAVEFORM:OCONDITION EXITING"

Query Syntax: :WAVeform:OCONdition?

Returned Format: [:WAVeform:OCONdition] {ENTering|EXITing}

Example:
```
10 DIM Setting$[100]
20 OUTPUT XXX;":WAVEFORM:OCONDITION?
30 ENTER XXX;Setting$
40 PRINT Setting$
50 END
```

OPATtern

command/query

The OPATtern command specifies a search pattern for the O marker which is then used with the OSEarch criteria and OCONdition for placing the marker on the pattern. The first parameter specifies the label and the second parameter specifies the search pattern for the label. Since each command deals with only one label, a specification over multiple labels requires multiple commands.

Each label can contain up to 32 bits with a range between 0 and FFFFFFFFh (hexadecimal). Since the search pattern for the label can contain don't cares (X) and be represented in several bases, the pattern specification parameter for the label is handled as a string.

In the PATTern marker mode, the OPATtern query returns the pattern specification for a given label. In the TIME marker mode, the query returns the pattern under the O marker for a given label.

Command Syntax: :WAVeform:OPATtern <label_name>,<pattern>

where:

<label_name> ::= string of up to 6 alphanumeric characters
<pattern> ::= string in one of the following forms:
"#B01X" for binary
"#Q01234567X" for octal
"#H0123456789ABCDEFX" for hexadecimal
"0123456789" for decimal
(don't cares "X" not allowed in decimal)

Example: OUTPUT XXX;":WAVEFORM:OPATTERN 'A','#B01001XX011'"

OPATtern

Query Syntax: :WAVeform:OPATtern? <label_name>

Returned Format: [:WAVeform:OPATtern] <label_name>,<pattern><NL>

Example:

```
10 DIM Pattern$[100]
20 OUTPUT XXX;":WAV:OPATTERN? 'A'"
30 ENTER XXX;Pattern$
40 PRINT Pattern$
50 END
```

OSEarch

command/query

The OSEarch command specifies the search criteria for the O marker which is then used with the associated OPATtern specification and the OCONdition for placing markers on patterns. The first parameter determines which occurrence of the OPATtern specification, relative to the starting point, the marker actually searches for. An occurrence of 0 (zero) places a marker on the selected starting point (trigger or X marker). The second parameter specifies whether the search will begin with the trigger or X marker.

The OSEarch query returns the search criteria for the O marker.

Command Syntax: :WAVeform:OSEarch <occurrence>[,{XMARker|TRIGger}]

where:

<occurrence> ::= -8192 to +8192 (integer)

Example:
```
OUTPUT XXX;":WAVEFORM:OSEARCH 5,TRIGGER"
```

Query Syntax: :WAVeform:OSEarch?

Returned Format: [:WAVeform:OSEarch] <integer>,{XMARker|TRIGger}<NL>

Example:
```
10 DIM Setting$
20 OUTPUT XXX;":WAVEFORM:OSEARCH?"
30 ENTER XXX;Setting$
40 PRINT Setting$
50 END
```

OTIMe command/query

The OTIMe command places the O marker at a specified position in time when the marker mode is TIME.

The OTIMe query returns the current position of the O marker. If there is no valid data, 9.9E37 is returned.

Command Syntax: :WAVeform:OTIMe <time>

where:

<time> ::= -13.0 s to +53.5E3 s (real number)

Example:
```
OUTPUT XXX;":WAVEFORM:OTIME 540ns"
```

Query Syntax: :WAVeform:OTIMe?

Returned Format: [:WAVeform:OTIMe] <time><NL>

Example:
```
10 DIM O_value$[100]
20 OUTPUT XXX;":WAVEFORM:OTIME?"
30 ENTER XXX;O_value$
40 PRINT O_value$
50 END
```

OVERlay command

The OVERlay command adds time correlated overlayed oscilloscope waveforms to the screen. The first parameter is the module specifier. 1...5 specifies the slot A...E where the module that has waveforms to be inserted is located. The next parameters specify the oscilloscope waveforms to be overlayed.

Command Syntax: :WAVeform:OVERlay <module_spec>,<waveform>,[<waveform>...]

where:

<module_spec> ::= 1 through 5 for slot in which oscilloscope timebase card is installed (integer)
<waveform> ::= <acquisition_spec>{1|2}
<acquisition_spec> ::= {A|B|C|D} (slot where oscilloscope acquisition card is located)

Example: OUTPUT XXX;":WAVEFORM:OVERLAY 2,'A1','A2'"

PLUS command

The PLUS command inserts time correlated A + B (A plus B) oscilloscope waveforms on the screen. The first parameter is the module specifier where 1...5 corresponds to the slot location A...E of the module that has waveforms to be inserted. The next two parameters specify which oscilloscope waveforms are to be added together.

Command Syntax: :WAVeform:PLUS <module_spec>,<waveform>,<waveform>

where:

<module_spec> ::= 1 through 5 for slot in which oscilloscope timebase card is installed (integer)
<waveform> ::= <acquisition_spec>{1|2}
<acquisition_spec> ::= {A|B|C|D|E} (slot where oscilloscope acquisition card is located)

Example: OUTPUT XXX;":WAVEFORM:PLUS 2,'A1','B1'"

Note

This command is only available for oscilloscope waveforms from an oscilloscope module in the same mainframe.

RANGe command/query

The RANGe command specifies the full-scale range of the waveform display. This is equivalent to ten times the seconds per division setting on the display. The allowable values for RANGe are from 10 ns to 10 s.

The RANGe query returns the current full-scale range.

Command Syntax: :WAVeform:RANGe <r_value>

where:

<r_value> ::= 10E-9 s to 10 s (real number)

Example:
```
OUTPUT XXX;":WAVEFORM:RANGE 4E-8"
```

Query Syntax: :WAVeform:RANGe?

Returned Format: [:WAVeform:RANGe] <r_value><NL>

Example:
```
10 DIM Setting$[100]
20 OUTPUT XXX;":WAVEFORM:RANGE?
30 ENTER XXX;Setting$
40 PRINT Setting$
50 END
```

REMove command

The REMove command removes all the waveforms from the display.

Command Syntax: :WAVeform:REMove

Example: OUTPUT XXX;":WAVEFORM:REMOVE"

RUNTil

command/query

The RUNTil (Run Until) command defines the stop criteria based on the time between the X and O markers when the trace mode is repetitive. When RUNTil is set to OFF, the analyzer will run until either the "STOP" touchscreen field is touched or the STOP command is sent. Run until the time between X and O marker options are:

- Less Than (LT) a specified time value
- Greater Than (GT) a specified time value
- IN the RANGE between two time values
- OUT of the RANGE of the two time values

The end points for INRange and OUTRange must not be the same value. The RUNTil query returns the current value.

Command Syntax: :WAVeform:RUNTil {OFF|LT,<value>|GT,<value>|INRange,<value>,<value>|OUTRange,<value>,<value>}"

where:

<value> ::= -53.5E3 s to +53.5E3 s (real number)

Examples:
```
OUTPUT XXX;":WAVEFORM:RUNTIL OFF"
OUTPUT XXX;":WAV:RUNT LT,10E-9"
OUTPUT XXX;":WAVEFORM:RUNTIL GT,10E-9"
OUTPUT XXX;":WAVEFORM:RUNTIL INRANGE,10E-9,20E-9"
OUTPUT XXX;":WAV:RUNT OUTRANGE,10E-9,20E-9"
```

RUNTil

Query Syntax: :WAVeform:RUNTil?

Returned Format: [:WAVeform:RUNTil] {OFF|LT,<value>|GT,<value>|INRange,<value>,<value>|OUTRange,<value>,<value>}

Example:

```
10 DIM Setting$[100]
20 OUTPUT XXX;":WAVEFORM:RUNTIL?"
30 ENTER XXX;Setting$
40 PRINT Setting$
50 END
```

SPERiod

query

The SPERiod query returns the sample period of the last run.

Query Syntax: WAVeform:SPERiod?

Returned Format: [:WAVeform:SPERiod] <time_value> <NL>

where:

<time_value> ::= 1E-9 s to 1.6E-3 s (real number)

Example:

```
10 DIM Value$[100]
20 OUTPUT XXX;":WAVEFORM:SPERIOD?"
30 ENTER XXX;Value$
40 PRINT Value$
50 END
```

TAVerage query

The TAVerage query returns the value of the average time between the X and O markers. If there is no valid data, the query returns 9.9E37.

Query Syntax: :WAVeform:TAVerage?

Returned Format: [:WAVeform:TAVerage] <value> <NL>

where:

<value> ::= -13.1072 s to +13.1072 s (real number)

Example:

```
10 DIM Value$[100]
20 OUTPUT XXX;":WAVEFORM:TAVERAGE?"
30 ENTER XXX;Value$
40 PRINT Value$
50 END
```

TMAXimum query

The TMAXimum query returns the value of the maximum time between the X and O markers. If there is no valid data, the query returns 9.9E37.

Query Syntax: :WAVeform:TMAXimum?

Returned Format: [:WAVeform:TMAXimum] <value> <NL>

where:

<value> ::= -13.1072 s to +13.1072 s (real number)

Example:

```
10 DIM Max_value$[100]
20 OUTPUT XXX;":WAVEFORM:TMAXIMUM?"
30 ENTER XXX;Max_value$
40 PRINT Max_value$
50 END
```

TMINimum

query

The TMINimum query returns the value of the minimum time between the X and O markers. If there is no valid data, the query returns 9.9E37.

Query Syntax: :WAVeform:TMINimum?

Returned Format: [:WAVeform:TMINimum] <value><NL>

where:

<value> ::= -13.1072 s to +13.1072 s (real number)

Example:

```
10 DIM Min_value$[100]
20 OUTPUT XXX;":WAVEFORM:TMINIMUM?"
30 ENTER XXX;Min_value$
40 PRINT Min_value$
50 END
```

VRUNs query

The VRUNs query returns the total number of runs and the number of valid runs made. Valid runs are those where the pattern search for both the X and O markers was successful, resulting in valid delta time measurements.

Query Syntax: :WAVeform:VRUNs?

Returned Format: [:WAVeform:VRUNs] <total_runs>,<valid_runs><NL>

where:

<total_runs> ::= positive integer (integer)
<valid_runs> ::= positive integer (integer)

Example:

```
10 DIM Runs$[100]
20 OUTPUT XXX;":WAVEFORM:VRUNS?"
30 ENTER XXX;Runs$
40 PRINT Runs$
50 END
```

XCONdition command/query

The XCONdition command specifies whether the marker is to be placed on the entry or exit point of the pattern when you are in the PATTern marker mode.

The XCONdition query returns the current setting.

Command Syntax: :WAVeform:XCONdition {ENTering|EXITing}

Example:
```
OUTPUT XXX;":WAVEFORM:XCONDITION EXITING"
```

Query Syntax: :WAVeform:XCONdition?

Returned Format: [:WAVeform:XCONdition] {ENTering|EXITing}

Example:
```
10 DIM X_setting$[100]
20 OUTPUT XXX;":WAVEFORM:XCONDITION?"
30 ENTER XXX;X_setting$
40 PRINT X_setting$
50 END
```

XOTime

query

The XOTime query returns the time from the X marker to the O marker. If the data is not valid, the query returns 9.9E37.

Query Syntax: :WAVeform:XOTime?

Returned Format: [:WAVeform:XOTime] <time_value> <NL>

where:

<time_value> ::= real number

Example:

```
10 DIM Time$[100]
20 OUTPUT XXX;":WAVEFORM:XOTIME?"
30 ENTER XXX;Time$
40 PRINT Time$
50 END
```

XPATtern command/query

The XPATtern command specifies a search pattern for the X marker which is then used with the XSEarch criteria and XCONdition for placing the marker on the pattern. The first parameter specifies the label and the second parameter specifies the search pattern for the label. Since each command deals with only one label, a specification over multiple labels requires multiple commands.

Each label can contain up to 32 bits with a range between 0 and FFFFFFFFh (hexadecimal). Since the search pattern for the label can also contain don't cares (X) and be represented in several bases, the pattern specification parameter for the label is handled as a string.

In the PATTern marker mode, the XPATtern query returns the pattern specification for a given label. In the TIME marker mode, the query returns the pattern under the X marker for a given label. If the X marker is not on valid data, don't cares (Xs) are returned.

Command Syntax: :WAVeform:XPATtern <label_name>,<pattern>

where:

<label_name> ::= string of up to 6 alphanumeric characters
<pattern> ::= string in one of the following forms:
"#B01X" for binary
"#Q01234567X" for octal
"#H0123456789ABCDEFX" for hexadecimal
"0123456789" for decimal
(don't cares "X" not allowed in decimal)

Example: OUTPUT XXX;":WAVEFORM:XPATTERN 'A','#B01001XX011'"

Query Syntax: :WAVeform:XPATtern? <label_name>

Returned Format: [:WAVeform:XPATtern] <label_name>,<pattern><NL>

Example:
```
10 DIM Pattern$[100]
20 OUTPUT XXX;":WAVEFORM:XPATTERN? 'A'"
30 ENTER XXX;Pattern$
40 PRINT Pattern$
50 END
```

XSEarch command/query

The XSEarch command specifies the search criteria for the X marker which is then used with the associated XPATtern specification and the XCONdition for placing markers on patterns. The first parameter for this command determines which occurrence of the XPATtern specification, relative to the trigger, the marker actually searches for. An occurrence of 0 (zero) places a marker on the selected starting point (trigger). The second parameter is optional since the X marker pattern is always referenced from the trigger point. The XSEarch query returns the search criteria for the X marker.

Command Syntax: :WAVeform:OSEarch <occurrence> [,TRIGger]

where:

<occurrence> ::= -8192 to +8192 (integer)

Example:
```
OUTPUT XXX;":WAVEFORM:XSEARCH 5,TRIGGER"
```

Query Syntax: :WAVeform:XSEarch?

Returned Format: [:WAVeform:XSEarch] <occurrence>,TRIGger<NL>

Example:
```
10 DIM X_setting$[100]
20 OUTPUT XXX;":WAVEFORM:XSEARCH?"
30 ENTER XXX;X_setting$
40 PRINT X_setting$
50 END
```

XTIMe command/query

The XTIMe command places the X marker at a specified position in time when the marker mode is TIME.

The XTIMe query returns the current position of the X marker. If there is no valid data, 9.9E37 is returned.

Command Syntax: :WAVeform:XTIMe <time>

where:

<time> ::= -13.0 s to +53.5E3 s (real number)

Example:
```
OUTPUT XXX;":WAVEFORM:XTIME 540ns"
```

Query Syntax: :WAVeform:XTIMe?

Returned Format: [:WAVeform:XTIMe] <time><NL>

Example:
```
10 DIM X_value$[100]
20 OUTPUT XXX;":WAVEFORM:XTIME?"
30 ENTER XXX;X_value$
40 PRINT X_value$
50 END
```

SYMBol Subsystem 5

Introduction

The SYMBol subsystem contains the commands that allow you to define symbols on the controller and download them to the HP 16515A/16516A 1 GHz Timing Analyzer module.

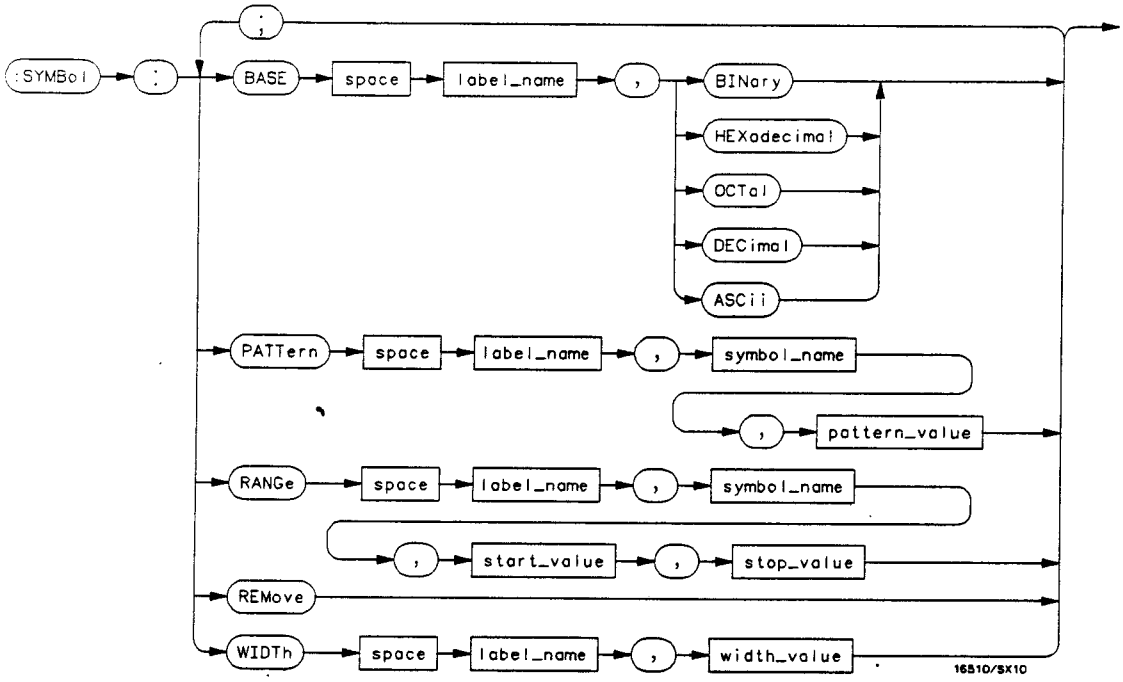


Figure 5-1. SYMBol Subsystem Syntax Diagram

<label_name> = *string of up to 6 alphanumeric characters*
<symbol_name> = *string of up to 16 alphanumeric characters*
<pattern_value> = *string of one of the following forms:*
"#B01X..." for binary
"#Q01234567X.." for octal
"#H0123456789ABCDEFX..." for hexadecimal
"0123456789..." for decimal
(don't cares "X" not allowed in decimal)
<start_value> = *string of one of the following forms:*
"#B01..." for binary
"#Q01234567.." for octal
"#H0123456789ABCDEF..." for hexadecimal
"0123456789..." for decimal
(don't cares "X" not allowed in decimal)
<stop_value> = *string of one of the following forms:*
"#B01..." for binary
"#Q01234567.." for octal
"#H0123456789ABCDEF..." for hexadecimal
"0123456789..." for decimal
(don't cares "X" not allowed in decimal)
<width_value> = *integer form 1 to 16*

Figure 5-1. SYMBol Subsystem Syntax Diagram (Continued)

The BASE command sets the base in which symbols for the specified label will be displayed in the symbol menu. It also specifies the base in which the symbol offsets are displayed when symbols are used.

Note

BINary is not available for labels with more than 20 bits assigned. In this case the base will default to HEXadecimal.

Command Syntax: :SYMBol:BASE <label_name>,<base_value>

where:

<label_name> ::= string of up to 6 alphanumeric characters
<base_value> ::= {BINary | HEXadecimal | OCTal | DECimal | ASCii}

Example: OUTPUT XXX;":SYMBol:BASE 'DATA',HEXadecimal"

PATTern command

The PATTern command allows you to create a pattern symbol for the specified label. The pattern may contain "don't cares" in the form of Xs.

Command Syntax: :SYMBol:PATTern <label_name>, <symbol_name>, <pattern_value>

where:

<label_name> ::= string of up to 6 alphanumeric characters
<symbol_name> ::= string of up to 16 alphanumeric characters
<pattern_value> ::= string of one of the following forms:
"#B01X" for binary
"#Q01234567X" for octal
"#H0123456789ABCDEFX" for hexadecimal
"0123456789" for decimal
(don't cares "X" not allowed in decimal)

Example: OUTPUT XXX;":SYMBol:PATTern 'STAT', 'MEM_RD','#H01XX'"

The RANGe command allows you to create a range symbol containing a start value and a stop value for the specified label.

Note

Don't cares "X" are not allowed in range symbols.

Command Syntax: :SYMBol:RANGe <label_name>,<symbol_name>,
<start_value>,<stop_value>

where:

<label_name> ::= string of up to 6 alphanumeric characters
<symbol_name> ::= string of up to 16 alphanumeric characters
<start_value> ::= string of one of the following forms:
"#B01" for binary
"#Q01234567" for octal
"#H0123456789ABCDEF" for hexadecimal
"0123456789" for decimal
(don't cares "X" not allowed in decimal)
<stop_value> ::= string of one of the following forms:
"#B01" for binary
"#Q01234567" for octal
"#H0123456789ABCDEF" for hexadecimal
"0123456789" for decimal
(don't cares "X" not allowed in decimal)

Example: OUTPUT XXX;":SYMBol:RANGe 'STAT', 'IO_ACCESS','#H0000','#H000F'"

REMove command

The REMove command deletes all of the current symbols.

Command Syntax: :SYMBol:REMove

Example: OUTPUT XXX;":SYMBol:REMove"

The WIDTh command specifies the width (number of characters) in which the symbol names will be displayed when symbols are used.

Note

The WIDTh command does not affect the displayed length of the symbol offset value.

Command Syntax: :SYMBol:WIDTh <label_name>,<width_value>

where:

<label_name> ::= string of up to 6 alphanumeric characters
<width_value> ::= integer from 1 to 16

Example: OUTPUT XXX;":SYMBol:WIDTh 'DATA',9 "

DATA and SETup Commands A

Introduction

The DATA and SETup commands are SYSTem subsystem commands that allow you to send and receive instrument configuration and acquired data to and from a controller in block form. These commands allow you to save an entire block of data for processing in the controller, or for later analysis in the timing analyzer. This appendix explains how to use these commands.

The block data for the DATA command and query is broken down into bytes and byte positions. This is intended primarily for processing of data in the controller.

Note

Do not change the block data in the controller if you intend to process the block data later in the timing analyzer. Changes made to block data in the controller may cause unpredictable results in the timing analyzer.

SYSTem:DATA command/query

The SYSTem:DATA command transmits the data part of the setup configuration for the selected module (1 GHz timing analyzer). This is transmitted in block data format and contains the contents of the acquisition buffer, including all measurement results. The SYSTem:DATA query returns the current contents of the acquisition buffer to the controller.

Note

When you reload data in the analyzer with the SYSTem:DATA command, make sure the exact analyzer configuration that was present when data was sent to the controller is still present. To ensure the same configuration is still present, save the configuration with the SYSTem:SETup command and reload the configuration in the analyzer before you reload the data.

Command Syntax: :SYSTem:DATA <block data in # format>

Query Syntax: :SYSTem:DATA?

Returned Format: [:SYSTem:DATA] <block data in # format> <NL>

Definition of Block Data

Block data in the # format is made up of a block length specifier and a variable number of sections.

<block length specifier> <section 1> <section 2> ... <section N>

The block length specifier is defined as follows:

#8<length>

where:

<length> :: = the total length of all sections in byte format (must be represented with 8 digits)

For example, if the total length of the block (all the sections) is 144 bytes, the block length specifier would be "#800000144" since the length must be represented with 8 digits.

Sections consist of a section header followed by the section data as follows:

<section header> <section data>

where:

<section header> :: = 10 bytes for the section name
1 byte reserved (always 0)
1 byte for the module ID code (1 for 1 GHz timing)
4 bytes for the length of the section data in bytes

The section data format varies for each section and may be any length.

Note

The total length of a section is 16 (for the section header) plus the length of the section data. Thus, when calculating the length of a block of configuration data, care should be taken to not forget to add the length of the section headers.

SYSTem:DATA

HP-IB Example:

```
10 DIM Block$[32000]  !allocate enough memory for block data
20 DIM Specifier$[2]
30 OUTPUT XXX;":EOI ON"
40 OUTPUT XXX;":SYSTEM:HEAD OFF"
50 OUTPUT XXX;":SELECT 4"  !select module
60 OUTPUT XXX;":SYSTEM:DATA?"  !send data query
70 ENTER XXX USING "#,2A";Specifier$  !read in #8
80 ENTER XXX USING "#,8D";Blocklength  !read in block length
90 ENTER XXX USING "-K";Block$  !read in data
100 END
```

Data Command Configuration

The SYSTem:DATA command for the HP 16515A/16A 1 GHz Timing Analyzer consists of the following section. Numbers are listed in the left hand column to specify the byte in the data block that the following value starts in.

16 bytes - Section Header

1 **10 bytes** - section name, "DATA " (six trailing spaces)

11 **1 byte** - reserved (always 0)

12 **1 byte** - module ID (1 for HP 16515A/16A)

13 **4 bytes** - length of section data in bytes
~~(8192 * 2)~~ + 24 for HP 16515A
~~(8192 * 4)~~ + 24 for HP 16515A/16516A

24 bytes - Preamble Information

17 **2 bytes** - how many pods are contained in the stored data.

19 **2 bytes** - index of the leftmost on-screen sample (72 to 572).

21 **4 bytes** - reserved

25 **2 bytes** - the pixel location of the leftmost sample (-1 to +8191). This returns -1 if the waveform is off screen to the right.

27 **4 bytes** - reserved

31 **4 bytes** - index of the sample that is the trigger point.

35 **2 bytes** - index of the sample period (see table A-1)

Table A-2. Index Values for Sample Period

Index	Sample Period		Index	Sample Period
0	1 ns		10	1.6 µs
1	2 ns		11	3.2 µs
2	4 ns		12	8 µs
3	8 ns		13	16 µs
4	16 ns		14	32 µs
5	32 ns		15	80 µs
6	80 ns		16	160 µs
7	160 ns		17	320 µs
8	320 ns		18	800 µs
9	800 ns		19	1.6 ms

37 **1 byte** - 1 if acquisition is valid, 0 if acquisition is not valid

38 **1 byte** - reserved

39 **2 bytes** - number of samples in the acquisition (always 8192)

40 **8192 bytes** - data for pod 2 of the HP 16515A

8232 **8192 bytes** - data for pod 1 of the HP 16515A

16424 **8192 bytes** - data for pod 2 of the HP 16516A (if an HP 16516A expansion card is installed)

24616 **8192 bytes** - data for pod 1 of the HP 16516A (if an HP 16516A expansion card is installed)

Note

Each byte of the 8192 bytes for each pod represents one sample, with channel 7 the most significant bit and channel 0 the least significant bit.

SYSTem:SETup command/query

The SYSTem:SETup command configures the selected module (1 GHz Timing Analyzer) as defined by the block data sent by the controller. The SYSTem:SETup query returns a block of data that contains the current configuration for the selected module to the controller.

Command Syntax: :SYSTem:SETup <block data in # format>

Query Syntax: :SYSTem:SETup?

Returned Format: [:SYSTem:SETup] <block data in # format> <NL>

Definition of Block Data Block data in the # format is made up of a block length specifier and a variable number of sections.

<block length specifier> <section 1> <section 2>

The block length specifier is defined as follows:

#8<length>

where:

<length> :: = the total length of all sections in byte format (must be represented with 8 digits)

For example, if the total length of the block (all the sections) is 144 bytes, the block length specifier would be "#800000144" since the length must be represented with 8 digits.

Sections consist of a section header followed by the section data as follows:

<section header> <section data>

where:

<section header> :: = 10 bytes for the section name
1 byte reserved (always 0)
1 byte for the module ID code (1 for 1 GHz timing)
4 bytes for the length of the section data in bytes

The section data format varies for each section and may be any length.

Note

The total length of a section is 16 (for the section header) plus the length of the section data. Thus, when calculating the length of a block of configuration data, care should be taken to not forget to add the length of the section headers.

HP-IB Example:

```
10 DIM Block$[32000]  !allocate enough memory for block data
20 DIM Specifier$[2]
30 OUTPUT XXX;":EOI ON"
40 OUTPUT XXX;":SYSTEM:HEAD OFF"
50 OUTPUT XXX;":SELECT 4"  !select module
60 OUTPUT XXX;":SYSTEM:SETUP?"  !send setup query
70 ENTER XXX USING "#,2A";Specifier$  !read in #8
80 ENTER XXX USING "#,8D";Blocklength  !read in block length
90 ENTER XXX USING "-K";Block$  !read in data
100 END
```

Index

A

ACCumulate, 4-6

B

BASE, 5-3
Block Data, A-3, A-7

C

CARDcage, 1-5
CESR, 1-11
Command
  ACCumulate, 4-6
  BASE, 5-3
  DELay, 4-7
  DURation, 3-3
  EDGE, 3-5
  INSert, 4-8
  LABel, 2-3
  MATCh, 3-7
  MENU, 1-6
  MESE, 1-12
  MINus, 4-10
  MMODe, 4-11
  OCONdition, 4-12
  OPATtern, 4-13
  OSEarch, 4-15
  OTIMe, 4-16
  OVERlay, 4-17
  PATTern, 3-8, 5-4
  PLUS, 4-18
  RANGe, 4-19, 5-5
  REMove, 2-5, 4-20, 5-6
  RMODe, 1-6
  RUNTil, 4-21
  SELect, 1-2, 1-6
  STARt, 1-6
  STOP, 1-6
  SYSTem:DATA, A-2
  SYSTem:PRINt, 1-7
  SYSTem:SETup, A-7
  THReshold, 2-6
  WIDTh, 5-7
  XCONdition, 4-28
  XPATtern, 4-30
  XSEarch, 4-32
  XTIMe, 4-33
Command cross-reference, 1-10
Command set organization, 1-8
Command structure, 1-2
Command tree, 1-9

D

DATA command, 1-1
DELay, 4-7
DURation, 3-3

E

EDGE, 3-5
Error messages, 1-1

F

FORMat subsystem, 2-1

I

ID number, 1-5
INSert, 4-8
INTermodule subsystem, 1-7

L

LABel, 2-3
Longform, 1-8

M

Mainframe commands, 1-5
Marker mode, 4-11
MATCh, 3-7
MENU, 1-6
MESE, 1-12
MESR, 1-14
MINus, 4-10
MMEMory subsystem, 1-7
MMODe, 4-11

O

OCONdition, 4-12
OPATtern, 4-13
OSEarch, 4-15
OTIMe, 4-16
OVERlay, 4-17

P

PATTern, 3-8, 5-4
PLUS, 4-18

Q

Query
- ACCumulate, 4-6
- CARDcage, 1-5
- DELay, 4-7
- DURation, 3-3
- EDGE, 3-5
- LABel, 2-3
- MATCh, 3-7
- MENU, 1-6
- MESE, 1-12
- MESR, 1-14
- MMODe, 4-11
- OCONdition, 4-12
- OPATtern, 4-13
- OSEarch, 4-15
- OTIMe, 4-16
- PATTern, 3-8
- RANGe, 4-19
- RMODe, 1-6
- RUNTil, 4-21
- SELect, 1-6

SPERiod, 4-23
SYSTem:DATA, A-2
SYSTem:ERRor, 1-7
SYSTem:PRINt, 1-7
SYSTem:SETup, A-7
TAVerage, 4-24
THReshold, 2-6
TMAXimum, 4-25
TMINimum, 4-26
VRUNs, 4-27
XCONdition, 4-28
XOTime, 4-29
XPATtern, 4-30
XSEarch, 4-32
XTIMe, 4-33

R

RANGe, 4-19, 5-5
REMove, 2-5, 4-20, 5-6
RMODe, 1-6
Run Until, 4-21
RUNTil, 4-21

S

SELect, 1-2, 1-6
SETup command, 1-1
Shortform, 1-8
SPERiod, 4-23
STARt, 1-6
Status reporting, 1-11
STOP, 1-6
Subsystem
FORMat, 2-1
INTermodule, 1-7
MMEMory, 1-7
SYMBol, 5-1
TRACe, 3-1
WAVeform, 4-1
SYMBol subsystem, 5-1
Syntax diagram
FORMat subsystem, 2-2
SYMBol subsystem, 5-1 / 5-2
TRACe subsystem, 3-1 / 3-2
WAVeform subsystem, 4-1 / 4-5
SYSTem:DATA, A-2
SYSTEM:DATA command, 1-1
SYSTem:ERRor, 1-7
SYSTem:PRINt, 1-7
SYSTem:SETup, A-7
SYSTEM:SETUP command, 1-1

T

TAVerage, 4-24
THReshold, 2-6
TMAXimum, 4-25
TMINimum, 4-26
TRACe subsystem, 3-1

V

VRUNs, 4-27

W

WAVeform subsystem, 4-1
WIDth, 5-7

X

XCONdition, 4-28
XOTime command, 4-29
XPATtern, 4-30
XSEarch, 4-32
XTIMe, 4-33