Programmer's Guide

HP 83475B Lightwave Communications Analyzer

HP part number: 83475-90009
Printed in USA November 1995

Notice

The information contained in this document is subject to change without notice.

Hewlett-Packard makes no warranty of any kind with regard to this material, including but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Hewlett-Packard shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance, or use of this material.

1400 Fountaingrove Parkway, Santa Rosa, CA 95403-1799, USA

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 Institute of Standards and Technology, to the extent allowed by the Institute's calibration facility, and to the calibration facilities of other International Standards Organization members.

Regulatory Information

The performance characteristics and regulatory information chapter of the *HP 83475B User's Guide* contains regulatory information.

Warranty

This Hewlett-Packard instrument product is warranted against defects in material and workmanship for a period of one year from date of shipment. During the warranty period, Hewlett-Packard Company will, at its option, either repair or replace products which prove to be defective.

For warranty service or repair, this product must be returned to a service facility designated by Hewlett-Packard. Buyer shall prepay shipping charges to Hewlett-Packard and Hewlett-Packard shall pay shipping charges to return the product to Buyer. However, 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, or 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 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 OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Exclusive Remedies

THE REMEDIES PROVIDED HEREIN ARE 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.

Typeface conventions

Front-Panel Key	This represents a key physically located on the instrument.
`Softkey`	This indicates a “softkey,” a key whose label is determined by the firmware of the instrument.
`Screen Text`	This indicates text displayed on the instrument’s screen.

WARNING **This symbol will appear along with bold print to highlight a warning.**

CAUTION This symbol will appear when special care is required.

NOTE

This symbol will appear to call attention to an important point in the text.

References

This document provides information on programming the HP 83475B Lightwave Communications Analyzer. It is not meant to be all-inclusive and it should be used with the information contained in the following references:

- HP-IB Reference Manuals

  Tutorial Description of the Hewlett Packard Interface Bus
  HP Part Number 5021-1927
  Using the HP-IB Interface and Command Library
  HP Part Number 82335-90001

- *HP 83475B Lightwave Communications Analyzer User's Guide*
  HP Part Number 83475-90008

- Microsoft QuickC Manuals:

  C for Yourself (version 2.5)
  Up and Running (version 2.5)
  Tool Kit (version 2.5)

- Microsoft QuickBASIC Manuals for IBM Personal Computers and Compatibles

  Learning to Use Microsoft QuickBASIC (version 4.5)
  Programming in BASIC (version 4.5)
  Microsoft QuickBASIC BASIC Language Reference (version 4.5)

- HP Instrument BASIC Manuals:

  Installing and Using HP Instrument BASIC for Windows
  HP Part Number E2200-9000
  HP Instrument BASIC User's Handbook
  HP Part Number E2083-90000

In This Book

Part 1	Chapter 1	provides instructions for installing the remote programming module.
	Chapter 2	describes the additional functions available when using the remote programming module.
	Chapter 3	lists the characteristics of the analyzer when using the remote programming module.
Part 2	Chapter 4	provides an introduction to programming and describes the HP-IB and RS-232-C interface functions.
	Chapter 5	describes conventions used in programming the HP 83475B.
	Chapter 6	discusses the types of commands.
Part 3	Chapter 7	provides a dictionary of the commands available for the HP 83475B.
Part 4	Chapter 8	contains example programs.
	Appendix A	contains the remote optical channel calibration procedure.

Contents

1. Installation

To Install the Remote Programming Module 1-3
Analyzer Compatibility 1-4
To Configure the Interface 1-9
The `HP-IB menu` softkey 1-9
The `RS-232 Menu` softkey 1-10
Selecting Print Factors 1-10
Selecting Print Resolution 1-12

2. Operating the Remote Programming Module

Remote Programming Menu Trees 2-3
Math Functions 2-7
Function 1 menu 2-8
Function 2 menu 2-9
FFT measurement menu 2-13
Operating system requirements 2-13
DC value 2-13
Aliasing 2-14
Spectral leakage 2-15
FFT operation 2-15
Additional FFT function keys 2-20
To Save or Recall Traces 2-21
To Create a Label for a Trace Memory 2-24
To Set the Real-Time Clock 2-26
To Perform Custom Mask Tests 2-28
Polygons 2-28
Left-bottom reference polygon 2-29
Center reference polygon 2-30
Regions 2-31
Test 2-32
Language definition 2-32
Definitions 2-34
User-defined mask definition block 2-34
Custom mask test file examples 2-42
FDDI pulse mask example 2-42

Fixed amplitude E4 one pulse mask example 2-44

3. Reference Information

Operating Characteristics 3-3
Fast Fourier transforms 3-3
Test region 3-3
Inputs 3-3
Frequency cursor resolution 3-3
Points 3-3
Peak Find 3-3
Variable sensitivity and offset 3-3
Time record length 3-3
Magnifying the display 3-4
Selectable windows 3-4
Window characteristics 3-4
FFT frequency range 3-4
Frequency span control 3-4
Center frequency control 3-4
Move 0 Hz to left 3-4
FFT vector display 3-5
Display 3-5
Units/div 3-5
Reference level 3-5
Programmability 3-5
Trace memory (all nonvolatile) 3-7
1 through 3 3-7
4 through 100 3-7
Real-time clock 3-7
Hardcopy output 3-7
Printer/plotter supported 3-7
Option 202 only 3-7
RS-232 configurations 3-8
Connector type 3-8
Protocols 3-8
Data bits 3-8
Stop bits 3-8
Parity 3-8
Baud rates 3-8
Programmability 3-8

4. Remote Programming

Introduction to Programming 4-3
Talking to the instrument 4-4
Program message syntax 4-5
Output command 4-5
Device address 4-5
Instructions 4-6
Instruction header 4-6
White space (separator) 4-6
Program data 4-6
Header types 4-7
Simple command header 4-7
Compound command header 4-7
Common command header 4-7
Combining commands 4-8
Duplicate mnemonics 4-8
Query command 4-8
Program header options 4-9
Program data syntax rules 4-10
Character program data 4-11
Numeric program data 4-11
Embedded strings 4-11
Program message terminator 4-12
Selecting multiple subsystems 4-13
Programming: Getting Started 4-14
Initialization 4-15
Autoscale 4-15
Setting up the instrument 4-15
Example program 4-16
Program overview 4-16
Using the Digitize command 4-17
Receiving information from the analyzer 4-18
String variables 4-19
Numeric variables 4-20
Definite-length block response data 4-21
Multiple queries 4-21
Instrument status 4-22
Programming over HP-IB 4-23
Interface capabilities 4-23
Command and data concepts 4-23
Addressing 4-24

Communicating over the bus 4-25
Interface select code (selects interface) 4-25
Instrument address (selects instrument) 4-25
Lockout 4-26
Bus commands 4-26
Device clear 4-26
Interface clear 4-26
Programming over RS-232-C 4-27
Interface operation 4-28
Cables 4-28
Minimum 3-wire interface with software protocol . . 4-29
Extended interface with hardware handshake 4-30
Configuring the interface 4-31
Interface capabilities 4-31
Protocol 4-32
DTR 4-32
XON/XOFF 4-32
Data bits 4-32
Communicating over the RS-232-C bus 4-33
Lockout command 4-33

5. Programming and Documentation Conventions

Truncation rules 5-3
Front-panel to command cross-reference 5-4
The command tree 5-4
Command types 5-4
Common commands 5-5
Root level commands 5-5
Subsystem commands 5-5
Tree traversal rules 5-7
Examples 5-9
Infinity representation 5-11
Sequential and overlapped commands 5-11
Response generation 5-11
Notation conventions and definitions 5-12
Conventions 5-12
Definitions 5-12
Program examples 5-13
Command set organization 5-14

6. Command Types

Common Commands 6-3
Root Level Commands 6-5
System Commands 6-6
Acquire Commands 6-7
Acquire type and count 6-7
Normal mode 6-7
Averaging mode 6-8
Peak Detect mode 6-8
Channel Commands 6-9
Display Commands 6-10
External Commands (External Trigger) 6-11
Function Commands 6-12
Mask Commands 6-13
Mask setup 6-13
Mask scaling and placement 6-14
Eye:NRZ 6-14
Single valued 6-14
Mask testing modes 6-15
Conformance test mode 6-15
Run mode 6-15
Mask margins 6-16
Measure Commands 6-17
Measurement setup 6-20
Measurement error 6-20
Making measurements 6-21
Ocalibrate Commands 6-23
Timebase Commands 6-24
Trace Commands 6-25
Trigger Commands (Internal Trigger) 6-26
Waveform Commands 6-27
Waveform data and preamble 6-27
Data acquisition types 6-28
Normal data 6-29
Average data 6-29
Peak detect 6-30
Data conversion 6-31
Conversion from data value to voltage or watts . . . 6-31
Conversion from data value to time 6-31
Data format for HP-IB transfer 6-32
WORD format 6-32

BYTE format 6-33
ASCII format 6-33

7. Command Dictionary

Common commands 7-3
*CLS (Clear Status) 7-3
*ESE (Event Status Enable) 7-4
*ESR (Event Status Register) 7-5
*IDN (Identification Number) 7-6
*LRN (Learn) 7-6
*OPC (Operation Complete) 7-7
*RCL (Recall) 7-7
*RST (Reset) 7-8
*SAV (Save) 7-9
*SRE (Service Request Enable) 7-9
*STB (Status Byte) 7-11
*TRG (Trigger) 7-12
*TST (Test) 7-12
*WAI (Wait) 7-12
Instrument specific commands 7-13
:ACQuire:COMPlete 7-13
:ACQuire:COUNt 7-14
:ACQuire:POINts 7-15
:ACQuire:SETup 7-15
:ACQuire:TYPE 7-16
:ASTore 7-16
:AUToscale 7-17
:BLANk 7-17
:CHANnel:BWLimit 7-18
:CHANnel:COUPling 7-19
:CHANnel:INPut 7-20
:CHANnel:INVert 7-21
:CHANnel:IOPTical 7-21
:CHANnel:OFFSet 7-22
:CHANnel:PROBe 7-23
:CHANnel:PROTect 7-24
:CHANnel:RANGe 7-24
:CHANnel:SETup 7-26
:CHANnel:SKEW 7-27
:CHANnel:VERNier 7-27
:CHANnel:XEATtenuation 7-28

:CHANnel:XEAUnits 7-29
:CHANnel:XOATtn 7-29
:DIGitize 7-30
:DISPlay:COLumn 7-31
:DISPlay:GRID 7-31
:DISPlay:INVerse 7-32
:DISPlay:LINE 7-32
:DISPlay:PIXel 7-33
:DISPlay:ROW 7-34
:DISPlay:SETup 7-35
:DISPlay:TEXT 7-35
:DITHer 7-36
:ERASe 7-36
:EXTernal:COUPling 7-37
:EXTernal:INPut 7-38
:EXTernal:OFFSet 7-39
:EXTernal:PROBe 7-40
:EXTernal:PROTect 7-41
:EXTernal:SETup 7-41
:EXTernal:SKEW 7-42
:FUNCtion:CENTer 7-43
:FUNCtion:MOVE 7-44
:FUNCtion:OFFSet 7-45
:FUNCtion:OPERation 7-45
:FUNCtion:PEAKs 7-46
:FUNCtion:RANGe 7-47
:FUNCtion:REFerence 7-48
:FUNCtion:SOURce 7-49
:FUNCtion:SPAN 7-50
:FUNCtion:VIEW 7-50
:FUNCtion:WINDow 7-51
:MEASure:ALL 7-52
:MEASure:ASAMples 7-53
:MEASure:BPERiod 7-54
:MEASure:BRATe 7-54
:MEASure:DCD 7-55
:MEASure:DEFine DELay 7-56
:MEASure:DELay 7-57
:MEASure:DUTYcycle 7-58
:MEASure:ECRossing 7-59
:MEASure:EHEIght 7-60

:MEASure:ERATio 7-60
:MEASure:ERUNits 7-61
:MEASure:EWMax 7-62
:MEASure:EWMin 7-63
:MEASure:FALLtime 7-64
:MEASure:FREQuency 7-65
:MEASure:JPP 7-66
:MEASure:JRMS 7-66
:MEASure:LOWer 7-67
:MEASure:MASK :DEFine:BFIT 7-67
:MEASure:MASK :DEFine:DATA 7-68
:MEASure:MASK :DEFine:INVert 7-68
:MEASure:MASK :DEFine:MARGin 7-69
:MEASure:MASK :DEFine:MHTS 7-70
:MEASure:MASK :DEFine:OFAil 7-71
:MEASure:MASK :DEFine:OPASs 7-72
:MEASure:MASK :DEFine:RETest 7-73
:MEASure:MASK :DEFine:TIME 7-73
:MEASure:MASK :IDENtify 7-74
:MEASure:MASK:SHOW 7-74
:MEASure:MASK:STESt 7-75
:MEASure:MASK:TEST 7-75
:MEASure:MTYPe 7-76
:MEASure:NPP 7-76
:MEASure:NRMS 7-77
:MEASure:NWIDth 7-78
:MEASure:OLEVel 7-79
:MEASure:OPMeter 7-79
:MEASure:OVERshoot 7-80
:MEASure:PAMPlitude 7-81
:MEASure:PAVerage 7-82
:MEASure:PBASe 7-83
:MEASure:PERiod 7-84
:MEASure:PHASe 7-85
:MEASure:PMAX 7-86
:MEASure:PMET 7-86
:MEASure:PMIN 7-87
:MEASure:PPP 7-88
:MEASure:PREShoot 7-89
:MEASure:PRMS 7-90
:MEASure:PTIMe 7-90

:MEASure:PTOP 7-91
:MEASure:PUNIts 7-91
:MEASure:PWDELta 7-92
:MEASure:PWIDth 7-93
:MEASure:PWSTArt 7-94
:MEASure:PWSTop 7-94
:MEASure:RISetime 7-95
:MEASure:SCRatch 7-95
:MEASure:SHOW 7-96
:MEASure:SOURce 7-96
:MEASure:TDELta 7-97
:MEASure:THResholds 7-98
:MEASure:TPOWer 7-99
:MEASure:TREF 7-100
:MEASure:TREF :OPERation 7-100
:MEASure:TREF :T1 7-101
:MEASure:TREF :T2 7-101
:MEASure:TSAMples 7-102
:MEASure:TSTArt 7-103
:MEASure:TSTOp 7-103
:MEASure:TVOLt 7-104
:MEASure:UPPer 7-105
:MEASure:VAMPlitude 7-106
:MEASure:VAVerage 7-107
:MEASure:VBASe 7-108
:MEASure:VDELta 7-108
:MEASure:VMAX 7-109
:MEASure:VMIN 7-110
:MEASure:VPP 7-111
:MEASure:VRMS 7-112
:MEASure:VSTArt 7-113
:MEASure:VSTOp 7-113
:MEASure:VTIMe 7-114
:MEASure:VTOP 7-114
:MEASure:ZLEVel 7-115
:OCALibrate:CALibrate 7-115
:OCALibrate:ECALibrate :CHAN1 (or CHAN2) . . . 7-116
:OCALibrate:ECALibrate :SKIP 7-116
:OCALibrate:ECALibrate :XTRigger 7-117
:OCALibrate:ERATio :FCOR 7-117
:OCALibrate:ERATio :IOPT 7-118

:OCALibrate:ERATio :XE1 7-119
:OCALibrate:ERATio :XE2 7-120
:OCALibrate:KILL 7-120
:OCALibrate:LABel 7-121
:OCALibrate:LDEFault 7-121
:OCALibrate:OCALibrate :OCGain 7-121
:OCALibrate:OCALibrate :PDARk 7-122
:OCALibrate:OCALibrate :PMET:HHRange 7-123
:OCALibrate:OCALibrate :PMET:HLRange 7-123
:OCALibrate:OCALibrate :PMET:HMRange 7-124
:OCALibrate:OCALibrate :PMET:LHRange 7-124
:OCALibrate:OCALibrate :PMET:LLRange 7-125
:OCALibrate:OCALibrate :PMET:LMRange 7-125
:OCALibrate:SAVE 7-126
:OCALibrate:SCALibrate 7-126
:OCALibrate:SDISplay 7-127
:PRINt 7-127
:RUN 7-127
:STATus 7-128
:STOP 7-128
:SYSTem:DSP 7-128
:SYSTem:ERRor 7-129
:SYSTem:KEY 7-132
:SYSTem:LOCK 7-134
:SYSTem:SETup 7-135
:TER (Trigger Event Register) 7-136
:TIMebase:DELay 7-137
:TIMebase:MODE 7-138
:TIMebase:RANGe 7-139
:TIMebase:REFerence 7-139
:TIMebase:SETup 7-140
:TIMebase:VERNier 7-141
:TRACe:CLEAR 7-141
:TRACe:DATA 7-142
:TRACe:MODE 7-143
:TRACe:SAVE 7-143
:TRIGger:COUPling 7-144
:TRIGger:HOLDoff 7-144
:TRIGger:LEVel 7-145
:TRIGger:MODE 7-146
:TRIGger:NREJect 7-147

:TRIGger:POLarity 7-147
:TRIGger:REJect 7-148
:TRIGger:SETup 7-148
:TRIGger:SLOPe 7-149
:TRIGger:SOURce 7-149
:TRIGger:TVHFrej 7-150
:TRIGger:TVMode 7-150
:VIEW 7-151
:WAVeform:BYTeorder 7-151
:WAVeform:DATA 7-152
:WAVeform:FORMat 7-152
:WAVeform:POINts 7-153
:WAVeform:PREamble 7-154
:WAVeform:SOURce 7-155
:WAVeform:TYPE 7-155
:WAVeform:XINCrement 7-156
:WAVeform:XORigin 7-156
:WAVeform:XREFerence 7-156
:WAVeform:YINCrement 7-157
:WAVeform:YORigin 7-157
:WAVeform:YREFerence 7-157

8. Programming Examples

Introduction 8-3
HP BASIC with HP-IB 8-4
HP BASIC with RS-232 8-8
HP IBASIC for Windows with HP-IB 8-12
HP IBASIC for Windows with RS-232 8-17
QuickBASIC with HP-IB 8-22
QuickBASIC with RS-232 8-29
QuickC with HP-IB 8-36
Custom Mask Example Programs 8-42
Mask downloading programming example 8-42
Mask reading programming example 8-44

A. Remote Optical Channel Calibration

To Correct for Insertion Loss A-4
Remote Calibration Sequence A-6

Index

Figures

1-1. Installing the remote programming module. 1-3
1-2. Printout using the `Factors` On function. 1-11
1-3. Printout using the `Factors` Off function. 1-11
1-4. Printout using the `Resolution` High function on an HP LaserJet Series II printer. 1-12
2-1. Cursor/Masks softkey menus. 2-3
2-2. Math softkey menus. 2-4
2-3. Print/Utility softkey menus. 2-5
2-4. Trace softkey menus. 2-6
2-5. Integrate and offset. 2-11
2-6. An example of aliasing. 2-14
2-7. FFT measurements. 2-18
2-8. Left-bottom reference polygon. 2-29
2-9. Center reference polygon. 2-30
3-1. Frequency span and effective sampling rate vs. sweep speed. 3-6
4-1. Program message syntax. 4-5
4-2. Analyzer utility/print menu with HP-IB. 4-24
4-3. Analyzer Print/Utility menu with RS-232-C. 4-31
5-1. Programming command tree. 5-6
5-2. Programming command tree, continued. 5-7
A-1. Block diagram of a calibration setup. A-3

Tables

1-1. Cable Part Numbers 1-4
2-1. User-Defined Test Text File Codes 2-33
2-2. User-Defined Mask Definition 2-35
2-3. Polygon Definition 2-40
5-1. Mnemonic Truncation 5-3
5-2. Alphabetic Command Cross-Reference 5-15
7-1. Standard Event Status Enable Register 7-4
7-2. Standard Event Status Register 7-5
7-3. Analyzer Reset Conditions 7-8
7-4. Service Request Enable Register 7-10
7-5. Status Byte Register 7-11
7-6. Error Messages 7-130
7-7. Front-Panel Key Codes 7-133

1

Installation

Installation

What you'll find in this chapter

This chapter provides you with the information necessary to install the remote programming module on the analyzer. Information is also provided to connect and configure the module to the desired external devices (such as a printer, plotter, or computer) prior to local or remote operation.

To Install the Remote Programming Module

1. Turn off the analyzer.
2. Install the module as shown in Figure 1-1.

   The analyzer is reset after installation. The type of module installed is shown in the information displayed when the analyzer is first turned on.

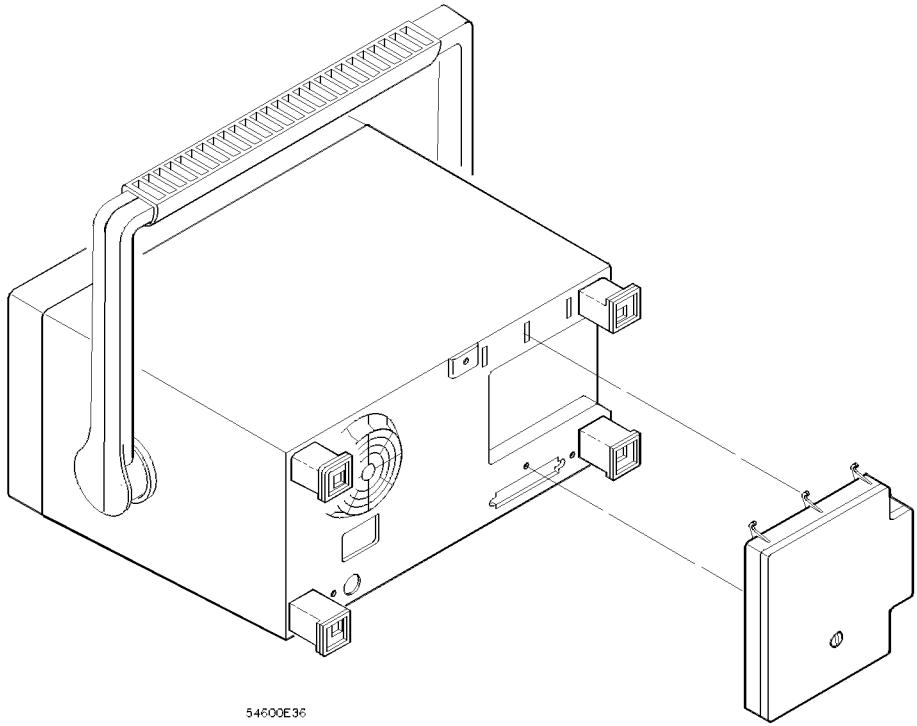


Figure 1-1. Installing the remote programming module.

Analyzer Compatibility

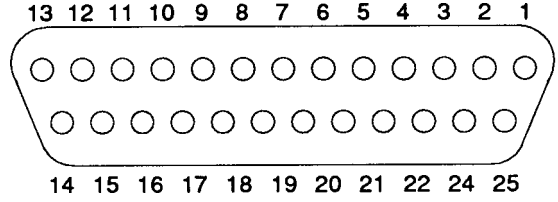
The remote programming module can be connected to a printer, a plotter, or a computer through the interface. The HP 83475B Option 201 has an HP-IB interface while the HP 83475B Option 202 has an RS-232 interface. The following cables may be used.

Table 1-1. Cable Part Numbers

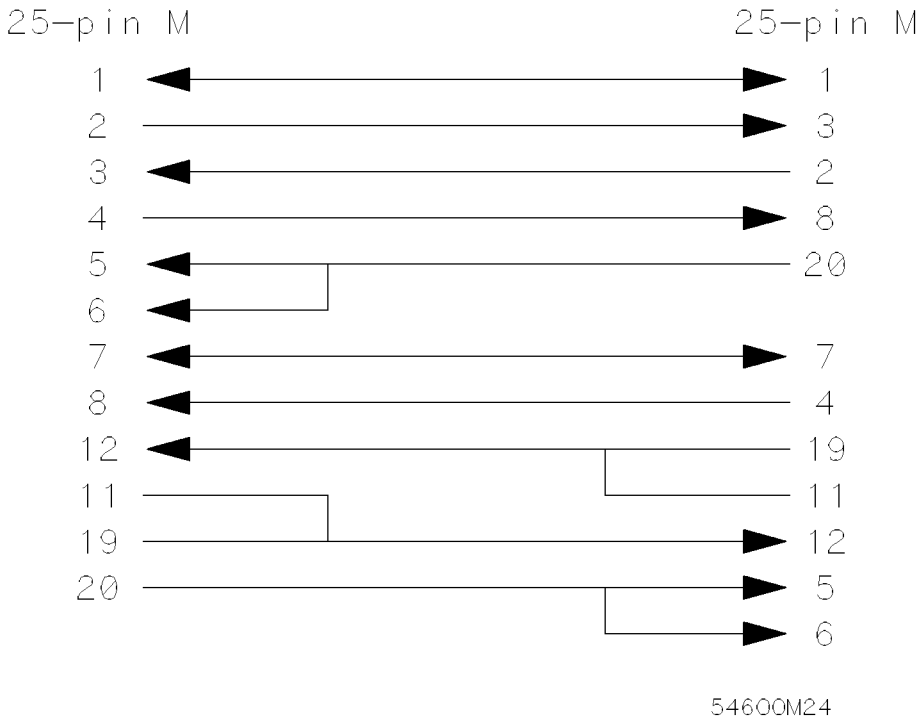
Cable Function	HP Part Number	Description
HP 83475B Option 201 (HP-IB)		
Analyzer to Printer/Plotter Controller	HP 10833A HP 10833B HP 10833C HP 10833D	1 meter (3.3 feet) 2 meters (6.6 feet) 4 meters (13.2 feet) 0.5 meter (1.6 feet)
HP 83475B Option 202 (RS-232)		
Analyzer to Printer/Plotter	HP 13242G	5 meters (16.7 feet)
HP Vectra 25-pin serial port	HP 17255M	1.2 meters (3.9 feet)
Analyzer to IBM 25-pin serial port	HP 92219J	5 meters (16.7 feet)
Analyzer to 9-pin serial port	HP 24542G	3 meters (9.9 feet)

The following figures show the pin outs of the RS-232 cables used with the HP 83475B Option 202. The signals for the RS-232 port on the HP 83475B are listed below:

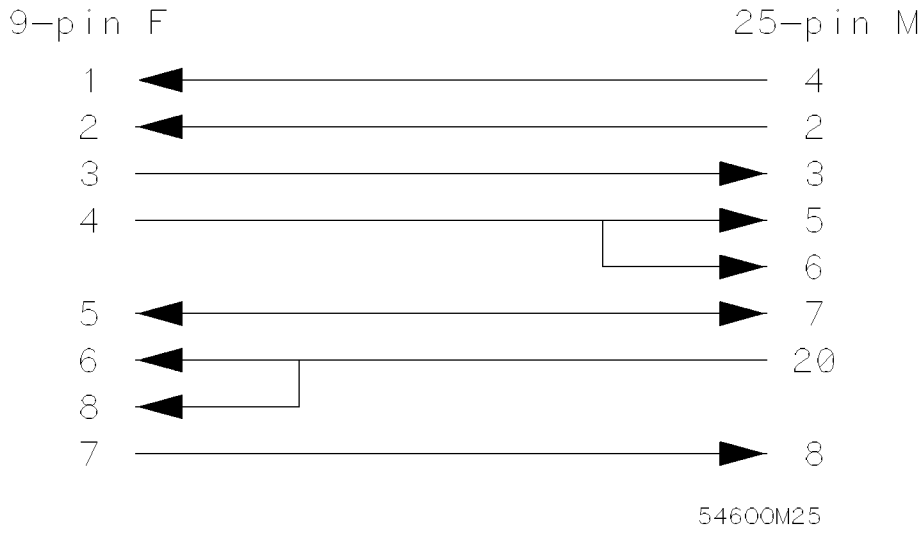
Pin Number	Signal
2	Transmit Data
3	Receive Data
4	Request to Send
5	Clear to Send
6	Data Set Ready
7	Signal Ground
8	Data Carrier Detect
20	Data Terminal Ready
SHELL	Ground



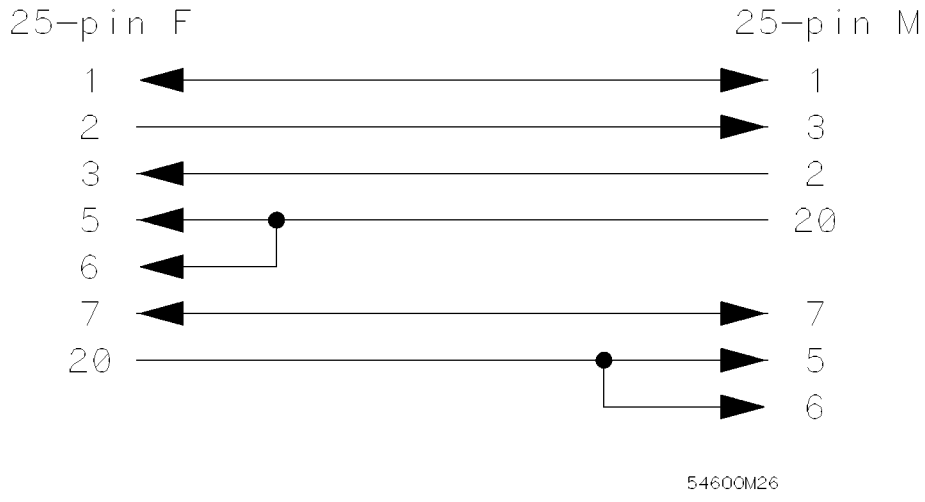
Pin out of RS-232 port in HP 83475B Option 202 looking into DB25 connector.



Pin out of RS-232 cable for connecting to Printer/Plotter/HP Vectra 25-pin serial port.



Pin out of RS-232 cable for connecting to 9-pin serial port.



Pin out of RS-232 cable for connecting to IBM PC/XT 25-pin serial port.

To Configure the Interface

1. Connect the remote programming module to a printer, plotter, or computer through a suitable cable. Refer to Table 1-1 for the HP part numbers of the proper cables.
2. Press (Print/Utility).

   When the module is installed, an additional softkey appears at the second softkey from the left. If you are using the HP 83475B Option 201, the softkey is labeled `HP-IB Menu`, continue with the procedure described in "The `HP-IB menu` softkey". If you are using the HP 83475B Option 202, the softkey is labeled `RS-232 Menu`, continue with the procedure described in "The `RS-232 Menu` softkey".

The `HP-IB menu` softkey

Press the `HP-IB Menu` softkey.

Four of the softkeys are related to the configuration of the HP-IB interface.

`Connect to`	Selects the device connected to the analyzer. Selections include HP plotter, HP printer, and computer.
`Factors`	Selects whether the printer or plotter output contains the information concerning the basic settings of the analyzer. See "Selecting Print Factors" later in this section for more information.
`Resolution`	Selects the resolution of the printer or plotter output. See "Selecting Print Resolution" later in this section for more information.
`Address`	Selects the HP-IB address of the instrument. Valid addresses range from 0 to 30. To change the address, press the softkey or turn the knob closest to the Cursors key.

The RS-232 Menu softkey

Press the RS-232 Menu softkey.

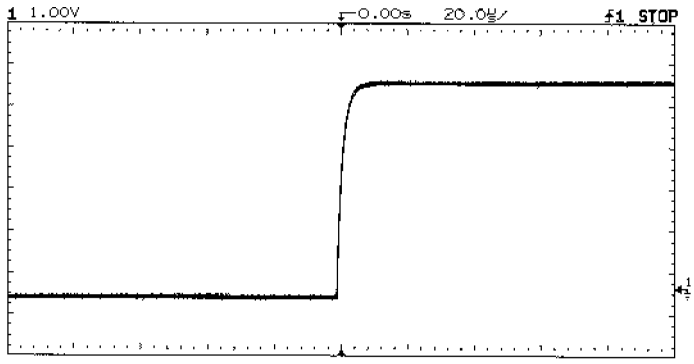
Five of the softkeys are related to the configuration of the RS-232 interface.

Connect to	Selects the device that the analyzer is connected to. Selections include HP plotter, HP printer, Epson compatible printer, and computer.
Factors	Selects whether the printer or plotter output contains the information concerning the basic settings of the analyzer. See "Selecting Print Factors" later in this section for more information.
Resolution	Selects the resolution of the printer or plotter output. See "Selecting Print Resolution" later in this section for more information.
Baud Rate	Selects the baud rate of the RS-232 port. Valid baud rates are 1200, 2400, 9600, and 19200.
Handshake	Selects the handshaking method used in the RS-232. DTR refers to the hardware handshaking using the DTR signal line. XON refers to the software handshaking using the XON/XOFF protocol.

Toggle the softkeys to select the correct configuration for your printer, plotter, or computer.

Selecting Print Factors

Analyzer factors may be turned on or off for hard copy prints and plots. All factors are printed on the hard copy when Factors On is selected. When factors are selected for a hard copy plot, the plot is in portrait mode, with the factors plotted at the bottom. When Factors Off is selected for hard copy plot, the plot is automatically plotted in landscape mode to maximize the plot area. Figure 1-2 shows an example of a printout using the Factors On function. Figure 1-3 shows an example of a printout using the Factors Off function.



	State	Volts/Div	Position	Couplg	BW Lim	Invert	Probe
Chan 1	On	1.000 V	-2.366 V	DC	Off	Off	1:1
Chan 2	Off	100.0mV	0.000 V	DC	Off	Off	1:1
Chan 3	Off	100.0mV	0.000 V	DC	---	---	1:1
Chan 4	Off	100.0mV	0.000 V	DC	---	---	1:1

	Mode	Main Time/Div	Main Delay	Time Ref	Delayed Time/Div	Delayed Delay
Horizontal	Normal	20.00us/	0.000 s	Cntr	--------	-----

	Mode	Source	Level	Holdoff	Slope	Couplg	Reject	NoiseRej
Trigger	AutoLvl	Ch 1	2.366 V	200.0ns	Pos	DC	Off	Off

Display Mode: Normal

Figure 1-2. Printout using the `Factors` On function.

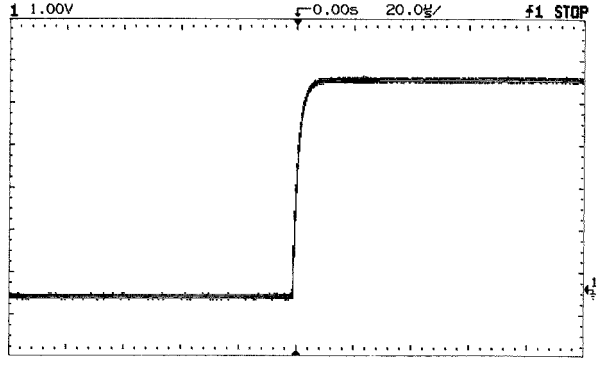


Figure 1-3. Printout using the `Factors` Off function.

Selecting Print Resolution

High or low hard copy resolution can be selected for hardcopy prints and plots. When `Resolution` High is selected, the full-bright and half-bright traces on the analyzer screen are printed or plotted on the hardcopy.

High resolution printing requires an HP-PCL compatible printer capable of printing at 300 dpi (dots per inch), such as an HP LaserJet Series printer. The half-bright trace is printed in gray shading while the full-bright trace is printed in black. Figure 1-4 shows an example of a high resolution print using the HP LaserJet II printer.

High resolution plotting uses two pens. Half-bright traces are plotted with plotter pen 1 and full-bright traces are plotted with plotter pen 2.

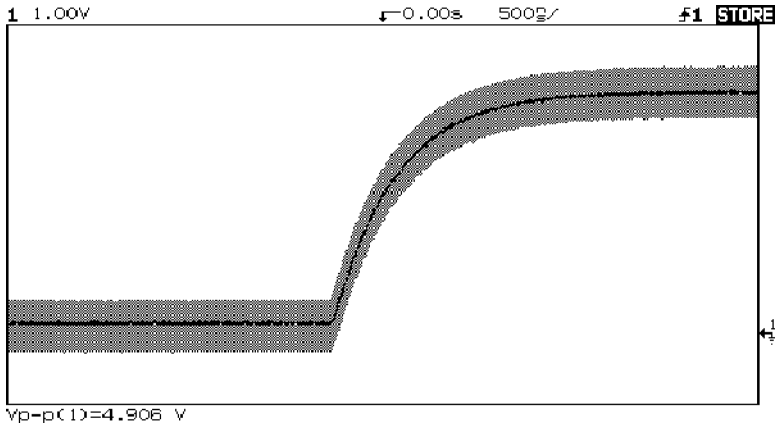


Figure 1-4.
Printout using the `Resolution` High function on an HP LaserJet Series II printer.

2

Operating the Remote Programming Module

Operating the Remote Programming Module

What you'll find in this chapter

This chapter provides you with the information necessary to use the features the Remote Programming Module provides. Basic operation for the analyzer is covered in the *HP 83475B Lightwave Communications Analyzer User's Guide*.

This chapter provides you with practical exercises and detailed information designed to guide you through operation of the following functions:

- Math Functions
- Trace Storage

When the remote programming module is installed the following menu trees are changed:

- Cursors/Masks
- Math
- Print/Utility
- Trace

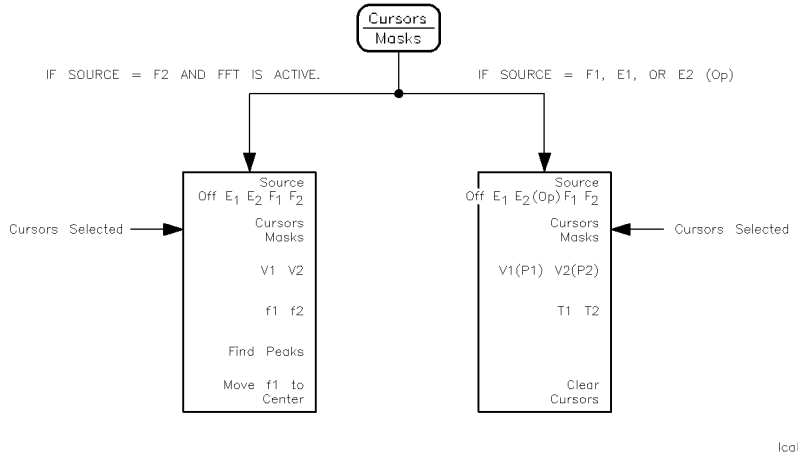


Figure 2-1. Cursor/Masks softkey menus.

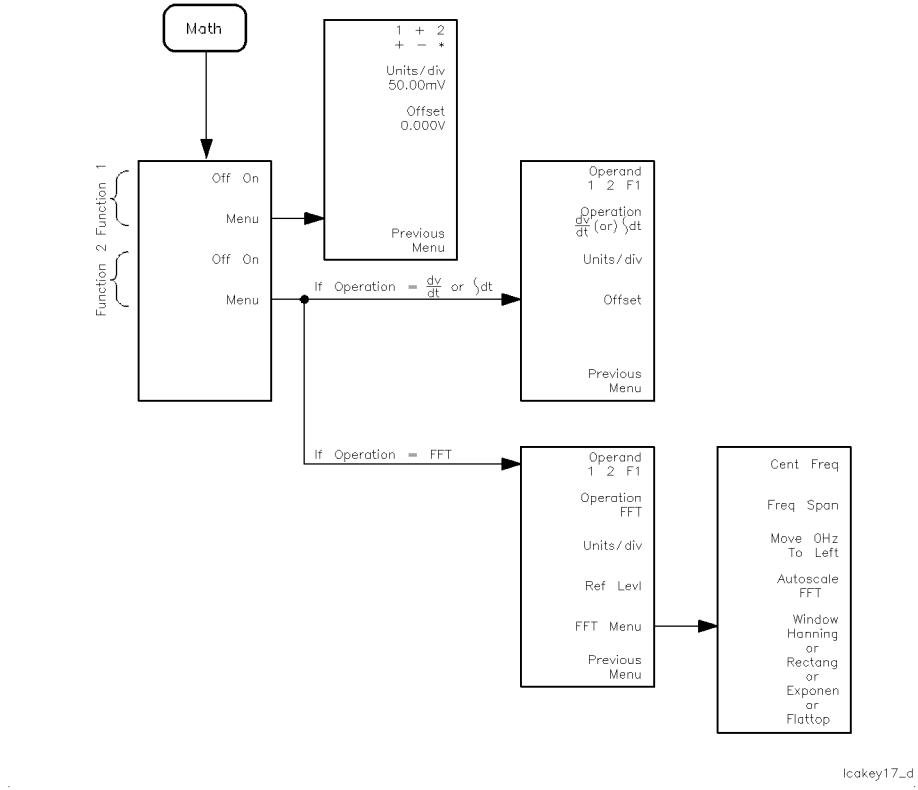


Figure 2-2. Math softkey menus.

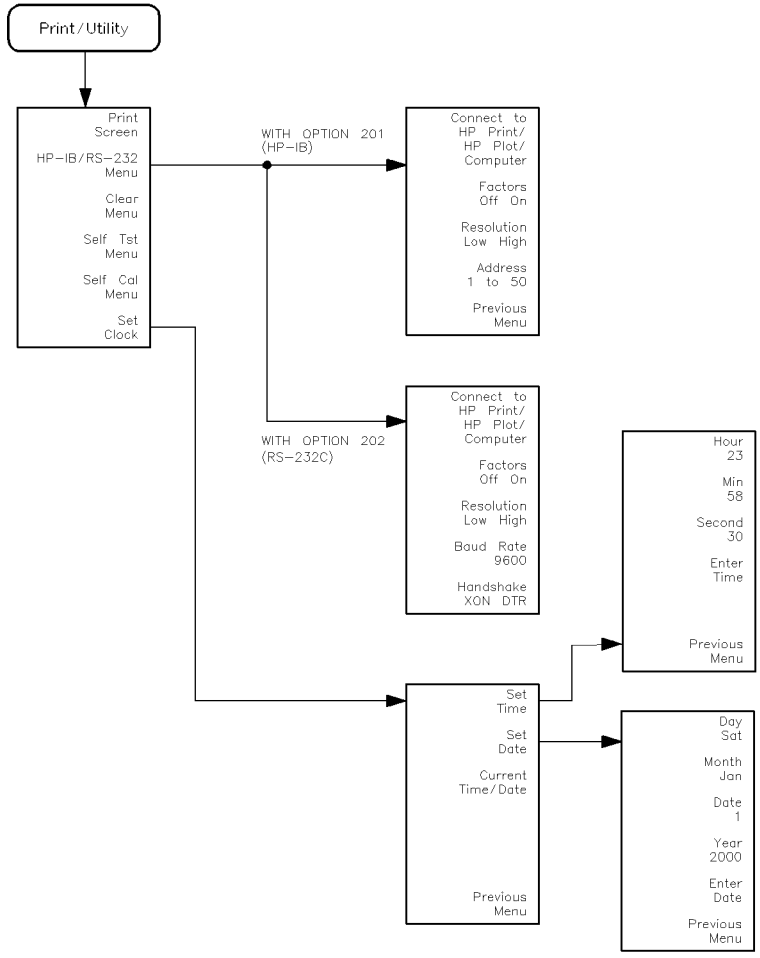


Figure 2-3. Print/Utility softkey menus.

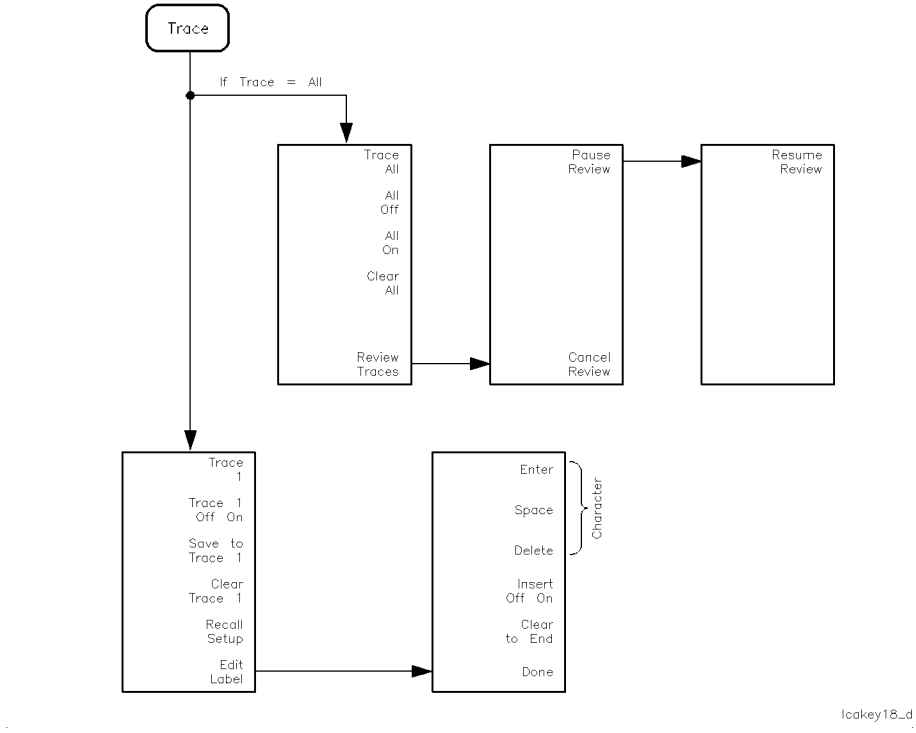


Figure 2-4. Trace softkey menus.

Math Functions

Without the remote programming module installed, addition and subtraction are the only math operations provided. In addition to the limited selections, the single function is performed on the pixel position of the data on the screen.

With the remote programming module installed, two functions define up to six operations that create mathematically altered waveforms (not pixel math).

- Function 1 will add (+), subtract (−), or multiply (*) the signals acquired on vertical inputs 1 and 2, then it will display the result as F1.
- Function 2 will integrate, differentiate, or perform an FFT on the signal acquired on input 1, input 2, or the result in F1; then it will display the result as F2.

The vertical range and offset of each function can be adjusted for ease of viewing and measurement considerations. Each function can be:

- displayed
- measured (with cursors)
- stored in trace memory
- output over the interface

NOTE

The math functions are *not* available when the optical channel is active or when the measurement type is set to Eye:NRZ. To activate electrical channel 2, press:

(Optical/Elec2)
`Chan 2 Off Op E2` E2

To set the measurement type to General Waveform, press:

(Power/Voltage)
`Meas Type Eye:NRZ Genl Wvfm` Genl Wvfm

Function 1 menu

1. To enable math function number 1, press:

   (±)
   `Function 1 Off On` On

2. Display the math functions by pressing:

   `Function 1 Menu`

   Toggle the `+ – *` softkey until the desired operation is selected. The results (F1) are displayed on the screen.

   All results are calculated on a point-by-point basis.

plus (+)	algebraically sum input 1 and input 2 (input 1 + input 2).
minus (–)	algebraically subtract input 2 from input 1 (input 1 – input 2).
multiply (*)	algebraically multiply input 1 with input 2 (input 1 × input 2).

3. To set the vertical sensitivity of the resulting waveform, press:

   `Units/div`

   Rotate the knob closest to the (Cursors/Masks) key to set the vertical sensitivity.

4. To set the offset (from the center graticule) of the resulting waveform, press:

   `Offset`

   Rotate the knob closest to the (Cursors/Masks) key to set the offset.

   Function waveform (F1) is available for viewing, measurement, or storage.

5. To return to the previous menu, press:

   `Previous Menu`

Function 1 operating hints

When multiply is the operation selected, the value displayed for units per division and offset is (V^2).

Offset is the value (in volts or volts^2) assigned to the center graticule for function 1. Normal screen position is 0 V offset, or at the center graticule (until changed).

See "Making Cursor Measurements" and "Saving and Recalling Traces" in this chapter for more information.

Function 2 menu

Function 2 will plot differential or integral waveforms, or perform an FFT using the input signals connected to the vertical inputs (1 and 2), or using the function 1 waveform.

1. To enable math function number 2, press:

   (±)

   `Function 2 Off On` On

2. Display the math functions by pressing:

   `Function 2 Menu`

3. Toggle the `Operand` softkey until the desired operand is selected. The results (F2) are displayed on the screen.

   F1 uses the result waveform in function 1.

4. Toggle the `Operation` softkey until the desired operation is selected.

   Results (F2) are displayed on the screen.

Differentiate (dV/dt)

Plots the derivative of the selected source using the "Central Difference" formula, as follows:

$$d_1 = \frac{c_1 - c_0}{\Delta t}$$

$$d_n = \frac{c(_{n+1}) - c(_{n-1})}{2\Delta t}$$

Where: d = differential waveform
c = input 1, 2, or function 1

Integrate ($\int$dt)

Plots the integral of the source using the "Trapezoidal Rule", as follows:

$$I_n = \frac{\Delta t}{2} \int_{i=1}^{i=n} c_i + ci + 1$$

Where: i = integral
c = input 1, 2, or function 1

The integrate calculation is relative to the currently selected source's input offset. The following examples illustrate any changes in offset level.

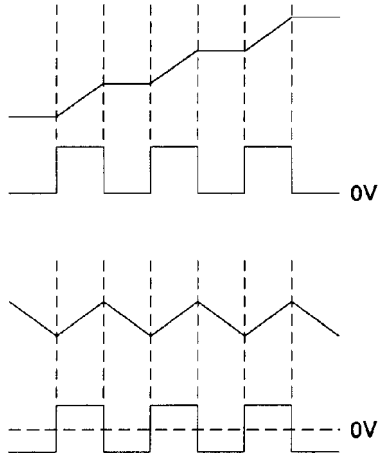


Figure 2-5. Integrate and offset.

FFT — The Fast Fourier Transform inputs the digitized time record of the source and transforms it to the frequency domain. The FFT spectrum is plotted on the analyzer display as dBV versus frequency. Selecting this function also adds the FFT menu. See "FFT Measurement" later in this chapter for more information.

5. To set the vertical sensitivity of the resulting waveform, press:

   `Units/div`

   Rotate the knob closest to the Cursors/Masks key to set the vertical sensitivity.

   Units per division changes from volts to dB when FFT is selected.

6. To set the offset (differentiate and integrate), press:

   `Offset`

   Rotate the knob closest to the (Cursors/Masks) key to set the offset (from the center graticule).

7. To set the reference level (FFT), press:

   `Ref Levl`

   Rotate the knob closest to the (Cursors/Masks) key to set the reference level (top graticule).

   Function waveform (F2) is available for viewing, measurement, or storage.

8. To return to the previous menu, press:

   `Previous Menu`

For FFT functions, an additional menu is available to set additional parameters. See "FFT Measurement" later in this chapter for more information.

Function 2 operating hints

Timebase must be set to `MAIN` when using function 2.

When differential is the operation selected, the value displayed for units per division and offset is volts per second (V/s). When integral is the operation selected, the value displayed for units per division and offset is volt-seconds (Vs).

Offset is the value (in volts per second or volt seconds) assigned to the center graticule for function 2. Normal screen position is 0 V offset, or at the center graticule (until changed).

Refer to "Making Cursor Measurements" and "Saving and Recalling Traces" in this chapter for more information.

FFT measurement menu

FFT (Function 2) is used to compute the fast Fourier transform using vertical inputs (1 and 2), or the Function 1 waveform. This function takes the digitized time record of the specified source and transforms it to the frequency domain. When the function is selected, the FFT spectrum is plotted on the analyzer display as dBV versus frequency. The readout for the horizontal axis changes from time to Hertz and the vertical readout changes from volts to dBV.

Operating system requirements

When the Remote Programming module is added to the HP 83475B lightwave analyzer, FFT measurement is based on repetitive sampling, with the number of points in the record fixed at 1024, and the FFT display is calibrated in dBV. This is a unit of measure that is referenced to 1 Vrms. If the display needs to be in dBm, the operator must switch to the 50Ω input mode and then perform the following conversion:

$$dBm = dBV + 13.01$$

DC value

The FFT computation produces a DC value that is incorrect. It does not take the offset at center screen into account and is 1.41421 times greater than its actual value. The DC value is not corrected in order to accurately represent frequency components near DC. All DC measurements should be performed in normal analyzer mode.

NOTE

The FFT function is not available if the optical channel is active. To use the FFT function select `Chan 2 Off Op E2` E2.

Aliasing

When using fast Fourier transforms, it is important to be aware of aliasing. This requires the operator have an idea of what the frequency domain should contain, and should also consider the effective sampling rate, frequency span, and analyzer vertical bandwidth when making FFT measurements. Sample rate is briefly displayed when the (±) key is pressed.

Aliasing happens when there are insufficient samples acquired on each cycle of the input signal to recognize the signal. This occurs whenever the frequency of the input signal is greater than the Nyquist frequency (sample frequency divided by 2). When a signal is aliased, the higher frequency components show up in the FFT spectrum at a lower frequency.

Figure 2-6 illustrates aliasing. In waveform A, the sample rate is set to 200 kSa/s, and the analyzer displays the correct spectrum. In waveform B, the sample rate is reduced by one-half (100 kSa/s), causing the components of the input signal above the Nyquist frequency to be mirrored (aliased) on the display.

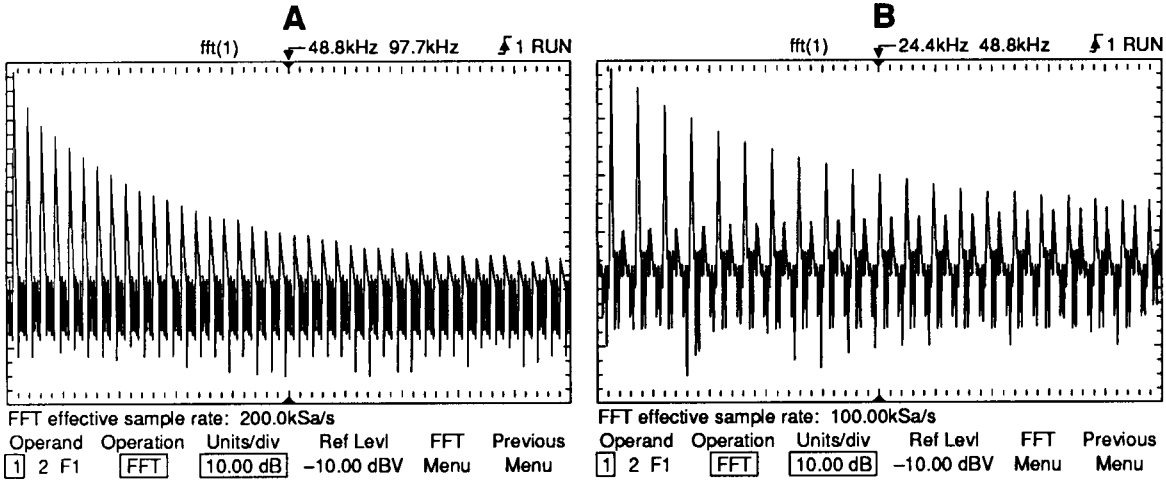


Figure 2-6. An example of aliasing.

Since the frequency span goes from ≈ 0 to the Nyquist frequency, the best way to prevent aliasing is to make sure that the frequency span is greater than the frequencies present in the input signal.

Spectral leakage

The FFT operation assumes that the time record repeats. Unless there is an integral number of cycles of the sampled waveform in the record, a discontinuity is created at the end of the record. This is referred to as leakage. In order to minimize spectral leakage, windows that approach zero smoothly at the beginning and end of the signal are employed as filters to the FFT. The remote programming module provides four windows: rectangular, exponential, Hanning, and flattop. For more information on leakage, see HP Application Note 243, *The Fundamentals of Signal Analysis*.

FFT operation

1. Enable math function 2 by pressing:

   (±)

   `Function 2 On Off` On

   `Function 2 Menu`

   Toggle the `Operand` softkey until the desired source is selected.

   F1 uses the result waveform in function 1.

2. Press the `Operation` softkey until FFT is selected.

   Results (F2) are displayed on the screen.

3. Set the vertical sensitivity of the resulting waveform by pressing:

   `Units/div`

   Rotate the knob closest to the (Cursors/Masks) key to set the vertical sensitivity.

4. Set the reference level (top graticule line) of the resulting waveform by pressing:

   `Ref Levl`

   Rotate the knob closest to the (Cursors/Masks) key to set the reference level.

5. Display the FFT menu by pressing:

   `FFT Menu`

The choices related to FFT are:

`Cent Freq`	Allows centering of the FFT spectrum to the desired frequency. Select and rotate the knob closest to the (Cursors/Masks) key to set the center frequency to the desired value.
`Freq Span`	Sets the overall width of the FFT spectrum (left graticule to right graticule). Select and rotate the knob closest to the (Cursors/Masks) key to set the center frequency to the desired value. See "FFT measurement hints" (later in this chapter) for information on using frequency span to magnify the display.
`Move 0Hz To Left`	Pressing this key changes the center frequency so that the left most graticule represents 0 Hz.
`Window`	Allows one of four windows to be selected. Select and rotate the knob closest to the (Cursors/Masks) key to set the desired window. The rectangular window is useful for transients signals and signals where there are an integral number of cycles in the time record. The Hanning window is useful for frequency resolution and general purpose use. It is good for resolving two frequencies that are close together or for making frequency measurements. The flattop window is the best window for making accurate amplitude measurements of frequency peaks. The exponential window is the best window for transients analysis.
`Previous Menu`	Returns you to the previous softkey menu.

6. Press the source channel number key to turn off its display and activate the vectors display. You now have an FFT display with vectors being drawn between the samples (connect the dots).

   FFT spectrum (F2) is available for viewing, measurement, or storage.

7. The (Cursors/Masks) key contains two additional selections that can be used to measure or move the FFT spectrum. Press (Cursors/Masks).

`Find Peaks` — Pressing this key sets Vmarker1 and the start marker (f1) on the peak with the highest amplitude and sets Vmarker2 and the stop marker (f2) on the peak with the next highest amplitude. Marker values in dBV or frequency (dependent on the active cursor) are automatically displayed at the bottom of the analyzer screen. The difference in dBV (ΔV) or frequency (Δ•f) between the two peaks is also displayed.

`Move f1 to Center` — Pressing this key changes the center graticule (or center frequency) to the current f1 marker frequency. If f1 cannot be found, a message is displayed on the screen.

The following FFT spectrum was obtained by connecting the front panel probe adjustment signal to input 1. Set Time/Div to 500 s/div, Volts/Div to 100 mV/div, Units/div to 10.00 dB, Ref Level to −10.00 dBv, Center Freq to 6.055 kHz, Freq Span to 12.21 kHz, and window to Hanning.

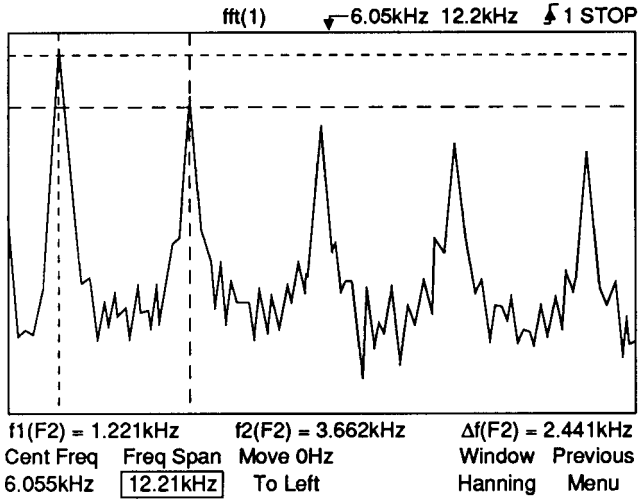


Figure 2-7. FFT measurements.

FFT measurement hints

It is easiest to view FFT's with vectors set to on. To do this, press:

Display

`Vectors Off On` On

`Stop`

The time and frequency domain will be displayed at the same time. Turning a channel `Off` will also provide vectors on the live FFT trace.

The number of points acquired for the FFT record is always 1024, and when frequency span is at maximum, all 1024 points are displayed. Once the FFT spectrum is displayed, the frequency span and center frequency controls are used much like the controls of a spectrum analyzer to examine the frequency of interest in greater detail. Place the desired part of the waveform at the center of the screen and decrease frequency span to increase the display resolution. As frequency span is decreased, the number of points shown is reduced, and the display is magnified.

While the FFT spectrum is displayed, use the cursor keys to switch between measurement functions and frequency domain controls in the FFT menu.

Decreasing the effective sampling rate, by selecting a slower sweep speed, will increase the low frequency resolution of the FFT display and also increase the chance that an alias will be displayed. Because there is always 1024 samples across the main sweep display window, the frequency resolution will be 2/1024 of the effective sampling rate. The actual resolution of the display will not be this fine because the shape of the window is the actual limiting factor in the FFT's ability to resolve two closely spaced frequencies. A good way to test the ability of the FFT to resolve two closely spaced frequencies is to examine the sidebands of an amplitude modulated sine wave. For example, at 2 MSa/sec effective sampling rate, a 1 MHz AM signal can be resolved to 2 kHz. Increasing the effective sampling rate to 4 MSa/sec reduces the resolution to 5 kHz.

For the best vertical accuracy on peak measurements:

- Make sure the source impedance and probe attenuation is set correctly. The impedance and probe attenuation are set from the Elec 1 or Optical/Elec 2 menu if the operand is a channel.
- Set the source sensitivity so the input signal is near full screen, but not clipped.
- Use the flattop window.
- Set the FFT sensitivity to a sensitive range, such as 2 dB/division.

FFT measurement hints (continued)

For best frequency accuracy on peaks:

- Use the Hanning window.
- Use cursors to place f1 cursor on the frequency of interest.
- Press the `Move f1 to Center` softkey.
- Adjust frequency span for better cursor placement.
- Return to the cursor menu to fine tune the f1 cursor.

For more information on the use of window, please refer to HP Application Note 243, *The Fundamentals of Signal Analysis* Chapter III, Section 5. Additional information can be found in Chapter 4 of *Spectrum and Network Measurements* by Robert A. Witte. (This book is available as HP part number 5960-5718.)

Additional FFT function keys

When the FFT function is selected (refer to Math Functions), two additional keys are available.

`Find Peaks` Sets Vmarker1 and the start marker (f1) on the FFT trace peak with the highest amplitude and sets Vmarker2 and the stop marker (f2) on the peak with the next highest amplitude. Marker values in dBV or frequency (dependent on the active cursor) are automatically displayed at the bottom of the analyzer screen. The difference in dBV (ΔV) or frequency (Δf) between the two peaks is also displayed.

`Move f1 To Center` Changes the center graticule (or center frequency) to the current f1 marker frequency. If f1 cannot be found, a message is displayed on the screen.

To Save or Recall Traces

With the remote programming module installed, the two volatile pixel memories are replaced with four high-speed nonvolatile memories.

In addition, 64 kilobytes of nonvolatile trace memory with data compression is also provided. A data compression algorithm maximizes the number of traces and front-panel setups that can be stored into this memory.

The total number of traces that can be saved depends on the complexity of each trace. At least four highly complex traces, or up to 96 simple traces, can be saved in this 64 KB of memory. Storage time is less than ten seconds.

1. Adjust analyzer controls for desired stable display.
2. To display the Trace menu, press:

   (Trace)

   The choices related to the trace memories of the module are:

`Trace`	Selects the trace memory (All, 1 to 100). When `Trace All 1 to 100` All is selected, the following additional softkeys are displayed:
`All Off`	Used to turn all 100 trace memories to off.
`All On`	Used to turn all 100 trace memories to on.
`Clear All`	Used to clear the contents of all 100 trace memories at one time.
`Review Traces`	Used to view the contents of all trace memories (with stored data) one at a time. View time is approximately three seconds. When selected, two additional softkeys are displayed. `Pause Review` pauses the review cycle on current trace until the key is pressed again, and `Cancel Review` cancels the review cycle.
`Trace X Off On`	Turns the selected memory number to on or off. When on is selected, trace status is temporarily displayed on the screen.

`Save to Trace X`	Saves the currently displayed waveform and front-panel setup to the selected trace memory location.
`Clear Trace X`	Erases a previously stored waveform and front-panel setup from the selected trace memory location.
`Recall Setup`	Recalls the previously stored front-panel setup that was saved with the waveform for the trace selected.
`Edit Label`	Used to enter a 20-character label that identifies the stored waveform. Refer to "To Create a Label for a Trace Memory" later in this chapter for more information.

3. Turn the knob closest to the (Cursors/Masks) key to move the cursor in the character area to select the desired trace location (from 1 to 100).

   Trace status is displayed on the screen. Information includes the edit label (or trace number if label has not been entered), and the time and date the memory was saved or cleared.

4. Select the desired softkey (`Save`, `Clear`, or `Recall`).

Saving traces hints

For traces 1 to 3, the trace is saved in the non-compressed state. For traces greater than 3, the trace is saved in 64 KB of nonvolatile trace memory with data compression. After a compressed trace is saved, the available area (in percent) is displayed.

The number of traces that can be saved in 64 KB of nonvolatile trace memory is dependent on the trace complexity and the current analyzer configuration. For example:

Up to 96 simple traces (for example, a square wave) can be stored.

As few as four complex traces (for example, an AM modulated sine wave) can be stored.

Saving a trace with autostore information or the grid requires more memory. If the information is not needed, turn it off to save memory.

The data compression algorithm uses run length encoding to compress the data. Where possible, runs of identical bits are stored as a single word (1 identification bit and 15 length bits).

The Review Traces function is useful when observing test failures.

To store only the front-panel setup, blank the screen prior to saving the trace. The edit label function can be used to identify the setup.

To Create a Label for a Trace Memory

Each trace stored in the remote programming module can have a label up to 20 characters long to identify the stored waveform.

1. To create a label, press:

   (Trace)
   `Edit Label`

   Five of the displayed softkeys are related to editing the message.

`Enter`	Enters the highlighted character in the character area into the message area under the cursor position.
`Space`	Enters a space in the message area under the cursor position.
`Delete`	Deletes the character in the message area under the cursor position.
`Insert`	Toggles the insert function on or off. With insert on, pressing `Enter` or `Space` inserts the character or space into the message before the character at the cursor position. The rest of the message is shifted to the right by one character. With insert off, pressing `Enter` or `Space` replaces the character highlighted in the character area.
`Clear to End`	Clears the characters from the cursor position to the end of the message including the one under the cursor.
`Done`	Exits the editing of the message.

2. Turn the Delay knob to move the cursor to the position in the label that you want to edit.
3. Turn the knob closest to the (Cursor/Masks) key to move the cursor in the character area to select the character that you want to enter.
4. Press the `Enter` softkey to enter the highlighted character into the user message, or press the `Space` softkey to enter a space into the user message.

5. Repeat procedures 2 through 4 until finished editing the message. The maximum length of the label is 20 characters.
6. Press the `Done` softkey.

Your message is displayed when you execute the various trace functions.

To Set the Real-Time Clock

Time (24-hour format) and date tagging of hard copy and nonvolatile trace memories is provided using a built-in, battery backed-up real-time clock.

1. To set time and date, or view current contents, press:

   (Print/Utility)
   `Set Clock`
   `Set Time`

   The choices related to setting the real-time clock time are:

`Hour`	Selects the hour digits (in 24-hour format). Current selection can be changed using the knob closest to the (Cursors/Masks) key, or by repeatedly pressing the softkey.
`Min`	Selects the minute digits. Current selection can be changed using the knob closest to the (Cursors/Masks) key, or by repeatedly pressing the softkey.
`Second`	Selects the second digits. Current selection can be changed using the knob closest to the (Cursors/Masks) key, or by repeatedly pressing the softkey.
`Enter Time`	Press to set the real-time clock time to the values displayed in the softkey area.
`Previous Menu`	Returns you to the previous softkey menu.

2. To set the date, press:

   `Set Date`

   The choices related to setting the real-time clock date are:

`Day`	Selects the day. Current selection can be changed using the knob closest to the (Cursors/Masks) key, or by repeatedly pressing the softkey.
`Month`	Selects the month. Current selection can be changed using the knob closest to the (Cursors/Masks) key, or by repeatedly pressing the softkey.

`Date`	Selects the day of the month. Current selection can be changed using the knob closest to the Cursors/Masks key, or by repeatedly pressing the softkey.
`Year`	Selects the year. Current selection can be changed using the knob closest to the Cursors/Masks key, or by repeatedly pressing the softkey.
`Enter Date`	Press to set the real-time clock date to the values displayed in the softkey area.
`Previous Menu`	Returns you to the previous softkey menu.

3. To display current real-time clock time and date information, press:

   `Current Time/Date`

Real-time clock hints

The real-time clock only allows selection of valid dates. If a day is selected and the month or year is changed so the day is now invalid, the day is automatically adjusted (for example, Feb 29).

To Perform Custom Mask Tests

Up to two user-defined masks can be downloaded into the non-volatile memory of the HP 83475B. The masks must be defined in an ASCII file. The ASCII file contains test parameter specifications and the geometrical description of the mask. The file is downloaded into the HP 83475B through HP-IB or RS-232 remote control. Examples of mask test files and of the HP-IB programs for downloading and reading mask information are provide in this section.

The test specifications, provided in the ASCII file for the mask description, are a set of parameters and thresholds which allow the analyzer software to automatically determine if a measurement has passed or failed the test. The test specification supports mask margins, fixed voltage or scaled masks, and mask hits measurements.

Masks are areas defined on the analyzer screen such that trace incursion will cause the analyzer to count hits inside the mask. The analyzer then compares the number of hits to a threshold of maximum allowed hits (which is also part of the test specification) to determine if the mask test has been passed.

There are three parts to a user-defined test file. These consist of polygons, regions, and tests.

Polygons

Polygons are sets of points that define areas on the analyzer screen. The points for the polygon are defined based on the definition given for the polygon. There are two possible references for a defined polygon. One is the center of the waveform (CENTER, CENTER). The second is the left bottom point of the waveform (T1, V BOTTOM). These references, defined in the mask definition file, provide the origin (point (0, 0)) for the Cartesian definition of the polygons. Points on a polygon are defined as (x, y) coordinates, where x is the time coordinate and y is the voltage coordinate. See Figure 2-8 and Figure 2-9 for more information.

Left-bottom reference polygon

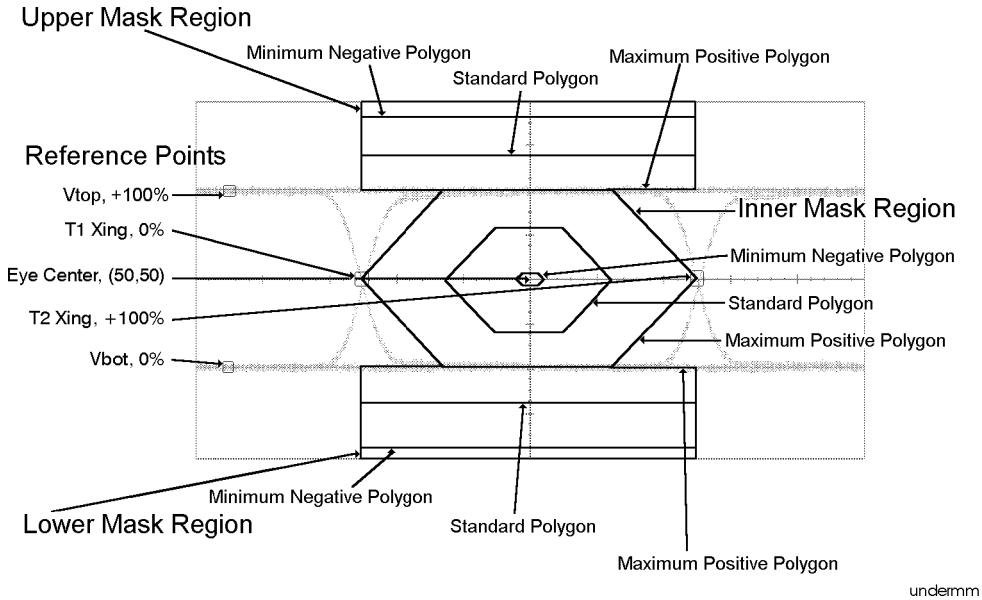


Figure 2-8. Left-bottom reference polygon.

Figure 2-8 shows the reference points for a mask defined with the origin in the voltage and time left-bottom point of the waveform. The time axis is in the X direction. The right time crossing of the eye (t_2) is 100%. The left time crossing of the eye (t_2) is 0%.

The Y axis is the amplitude axis of the analyzer. Where the high voltage of the eye crosses the Y axis defines the 100% value. Where the low voltage of the eye crosses the Y axis defines the 0% value. The last point of the polygon definition is automatically joined to the first, thereby creating the polygon.

Center reference polygon

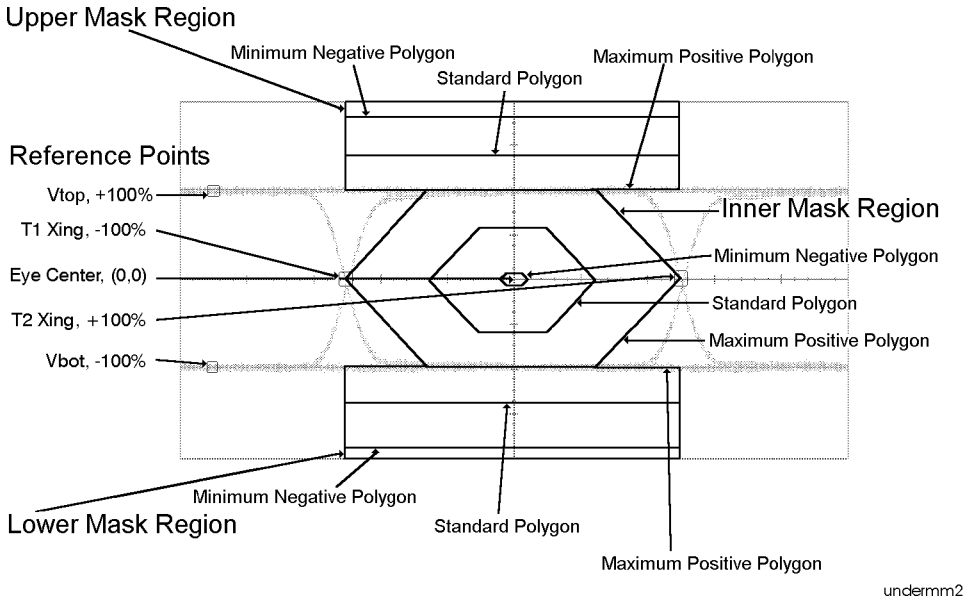


Figure 2-9. Center reference polygon.

Figure 2-9 shows the reference points for a mask defined with the origin in the voltage and time center of the waveform. The time axis is in the X direction. The right time crossing of the eye (t_2) is 100%. The left time crossing of the eye (t_1) is −100%.

The Y axis is the amplitude axis of the analyzer. Where the high voltage of the eye crosses the Y axis defines the 100% value. Where the low voltage of the eye crosses the Y axis defines the −100% value. The last point of the polygon definition is automatically joined to the first, thereby creating the polygon.

Regions

There are two possible types of regions, scaled and fixed amplitude. The mask scaling is defined by the type of region specified. A fixed amplitude region will scale to a specific waveform amplitude while a scaled region will be vertically normalized to fit any arbitrary waveform amplitude. Some pulse mask tests, such as the PDH tests, seek to incorporate both shape and amplitude measurements by defining the vertical mask scale in volts. Other mask tests, such as SONET and FDDI masks, only deal with waveform shape and define the vertical scale as a percentage of normalized waveform amplitude. Thus, in order to provide test versatility, both fixed and scaled mask regions can be defined with the HP 83475B.

Beyond scaling type, regions can be specified with or without margins. If no margin is specified, the mask region definition will consist of only one polygon. If margins are selected, two types of margins are possible, three-polygon and offset. Three-polygon margins allow for both positive and negative margins as a percentage of waveform amplitude and time unit interval. The unit interval is defined as the time width of the waveform under test. Typically it is close to the inverse of the bit rate. Offset margins are only specified for the vertical scale. These are specified in percentage of waveform amplitude.

For three-polygon margin masks a set of three polygons define a region. The first polygon is the maximum positive mask, the second is the standard mask, and the third is the minimum negative mask. See Figure 2-8 for more information.

The standard mask, typically, specifies the desired waveform shape. The positive mask defines how a positive mask margin will be applied to the standard mask to make the mask test more stringent.

The negative mask defines how a negative mask margin will be applied to the standard mask to make the mask test less stringent.

A 100% mask margin defines the positive mask, a 0% mask margin defines the standard mask, and a −100% margin defines the negative mask. All polygons used to define a region must contain exactly the same number of points.

Test

The test section of the user-defined mask test file specifies the mask parameters and thresholds. This includes, for example, the test bit rate, the test label, and the maximum number of mask hits allowed for a pass condition.

Language definition

Components of a user-defined test file must be organized in a specific order. The order is test, regions, and then polygons. Additional regions can be added if necessary with additional polygons within them. For example, an SDH/SONET mask would consist of a test section and three regions. Within each region, a set of polygons would be defined to specify the mask geometry and margin settings. Spaces and new lines (carriage returns) are ignored.

NOTE

It is recommended you use the same format as used in the examples provided at the end of this section.

The codes used in a user-defined test text file are as follows:

Table 2-1. User-Defined Test Text File Codes

Code	Definition
[header]	[header number]
[test]	TEST_BEGIN
	[version]
	,[test label]
	,[freq]
	,[waveform type]
	,[fit type]
	,[mask hit threshold]
	,[mask margin percent]
	,[test duration]
	,[time reference]
	,[time edge]
	,[voltage reference]
[region]	REGION_BEGIN
	,[region scaling]
	,[margin support]
[named polygon]	POLYGON_BEGIN
	,[polygon]
	POLYGON_END
[region end]	REGION_END
[test end]	TEST_END

Definitions

Definitions for the terms are given in Table 2-1 and provided below.

User-defined mask definition block

Input and output block form for the MEAS:MASK:DEF:DATA? query. The following table shows the data and descriptions of the fields. White space (space, tab, carriage return, newline) will be ignored between the various fields of the definition. All bytes must be counted in the number of bytes value in the header. All fields are comma delimited.

Table 2-2. User-Defined Mask Definition

Code	Description	Data
[header]	Header, where XXXX represents the number of bytes of data to follow. The first example shows a fixed field, 8 characters, definite length header. The analyzer will always respond with a definite length header equivalent to the first example. #0 is an indefinite length header. This block gets terminated with NL+EOI.	 #80000XXXX or #50XXXX or #4XXXX or #0
[test]	Test definition	TEST_BEGIN,
[version]	Version number. Fixed size field of one character. Only value allowed is {1}.	1,
[test label]	Label. Appears on SOFTKEY and as response to MEAS:MASK:IDENtify? Variable size set in quotes. Only the first nine characters are recorded.	"FC 266 MHz",
[freq]	Standard bit frequency. IEEE Floating point number.	266e6,
[waveform type]	Type of waveform on which measurements are to be taken. ASCII string required. Choices: {GENERAL | EYENRZ}	EYENRZ,
[fit type]	Type of fit algorithm to use. ASCII string required. Choices: {STANDARD}	STANDARD,
[mask hit threshold]	Default maximum number of hits allowed to pass. Integer. Range: 0–100000. May be limited by implementation.	1,

Table 2-2. User-Defined Mask Definition (continued)

Code	Description	Data
[mask margin percent]	Default mask margin. Decimal value representing percent. Range: −100 to +100. Valid only if the mask supports margins.	0,
[test duration]	Duration of conformance test. How long data is collected before a pass or fail is determined. Integer. Range: 1–999, implementation limits imposed.	5,
[time reference][1]	Time reference. The position used as a reference for time scaling. Choices: {CENTER, T1}. CENTER—half way between the 50% rise and fall (visa-versa). T1—at the left 50% rise or fall edge.	CENTER,
[time edge]	Time edge. Use the rise or fall edge. Choice: { RISE, FALL }	RISE,
[voltage reference][1]	Voltage reference. Choices: {TOP, BOT, CENTER}	CENTER,
[region begin]	Region begins. Region section must be repeated for each new region.	REGION_BEGIN,

1 There are only two possible combinations for the [time reference] and [voltage reference] parameters: CENTER, CENTER or T1, BOT.

Table 2-2. User-Defined Mask Definition (continued)

Code	Description	Data
[region scaling]	Specify the type of scaling for the region. Choices: {FIXED, SCALED} If FIXED, then the fixed VTOP and VBOT values must follow. These are floating point values, in VOLTS. If SCALED, then OMIT these values.	FIXED, 1.0, 0.0,
[margin support]	Type of margins supported. Choices: {THREE_POLYGONS, NO_MARGIN, OFFSET}. THREE_POLYGONS—three separate polygons will follow. All are positioned based on the first point, and in the following order: standard polygon, max positive polygon, negative polygon. NO_MARGIN—no margins supported. The standard polygon follows. OFFSET—margins are supported by offsetting the mask in the amplitude direction. The standard polygon is followed by the positive offset, then the negative offset. Examples of THREE_POLYGONS, NO_MARGIN and OFFSET are provided in the region examples given later in this section.	THREE_POLYGONS

Table 2-2. User-Defined Mask Definition (continued)

Code	Description	Data
[named polygon]	Polygons—points expressed in percentages. These points can be negative or positive. At least three points must be specified. The polygon is treated as a closed figure.	
	Define standard polygon	
	Polygon definition begins	POLYGON_BEGIN,
[polygon]	Points—specify time, then amplitude	|-70, 0|,|-30, 60|, |30, 60|,|70, 0|, |30,-60|,|-30, -60|,
	Polygon definition ends	POLYGON_END,

Table 2-3 continues with parameters defined for a custom mask test file. Specific examples are provided for definitions of regions and polygons.

Table 2-3. Polygon Definition

Description	Data
Define Max. Positive Polygon	POLYGON_BEGIN, |-100,0|, |-50,100|, |50,100|, |100,0|, |50,-100|,|-50,-100|, POLYGON_END,
Define Min. Negative Polygon	POLYGON_BEGIN, |-40, 0|, |-10, 20|, |10, 20|,|40, 0|, |10, -20|,|-10, -20|, POLYGON_END,
End of Region Define the Next Region Use as many regions as necessary. This region is defined for the top of the waveform.	REGION_END, REGION_BEGIN,
Region Scaling An example of an offset margin is given below.	OFFSET,
Standard Polygon	POLYGON_BEGIN, |-300,300|,|300, 300|, |300,140|,|-300, 140|, POLYGON_END,
Positive Offset – percentage.	-40, The sign indicates whether the positive offset should be applied in the upward or downward direction.

Table 2-3. Polygon Definition (continued)

Description	Data
Negative Offset End Region Definition	+40, REGION_END,
Define Bottom Mask This region does not support margins.	REGION_BEGIN, NO_MARGIN, POLYGON_BEGIN, |-300,-300|,|300,-300|, |300,-140|,|-300,-140|, POLYGON_END,
End of Mask Definition	TEST_END
Optional – finish indefinite block	NL+EOI

Custom mask test file examples

FDDI pulse mask example

The following example shows the specification of an FDDI pulse mask. The defined FDDI mask supports margins and is vertically normalized to the amplitude of the signal. Comments are preceded by the "*#*" sign. *Comments are not allowed in an actual mask definition file.*

```
#80000XXXX,                    # block length specifier
TEST_BEGIN,                    # specifies start of test
1,                             # version
"FDDI",                        # mask label
5.0e7,                         # mask bit rate
GENERAL,                       # waveform type
STANDARD,                      # standard fit type
0,                             # maximum number of hits allowed
0,                             # default mask margin percentage
5,                             # test duration (in seconds)
CENTER,                        # mask time reference
RISE,                          # time edge reference
CENTER,                        # voltage reference
REGION_BEGIN,                  # start of region specification
SCALED,                        # type of vertical fit
THREE_POLYGONS,                # region consists of three polygons

POLYGON_BEGIN,
 (-250, 3000), (-250, -95), (-124.25, -95),
 (-116.25, -80),(-99.625, 80), (-99.625, 150),
 (-90.125, 150), (-78, 105), (97.375, 105),
 (97.375, 80), (109.875,-80), (122, -95),
 (150,-95), (150, 39900),      # polygon geometrical specification
POLYGON_END,   # specification ends

POLYGON_BEGIN,
 (-250, 39900), (-250, -100), (-113.438, -100),
 (-109.438, -80), (-94.875, 80), (-94.875, 100),
 (-90.125, 100), (-78, 100), (94.875, 100),
 (94.875, 80), (105.125, -80), (122, -100),
 (150,-100), (150, 39900),     # polygon geometrical specification
```

```
POLYGON_END,                    # specification ends

POLYGON_BEGIN,
 (-250, 3000), (-250, -90), (-135.062, -90),
 (-123.062,-80), (-104.375, 80), (-104.375, 200),
 (-85.375, 200), (-73.25, 110), (99.675, 110),
 (99.675, 80), (114.625, -80), (122, -90),
 (150,-90), (150, 3000),        # polygon geometrical specification
POLYGON_END,                    # specification ends
REGION_END,                     # region specification ends

REGION_BEGIN,
 SCALED,
 THREE_POLYGONS,

POLYGON_BEGIN,
 (-250, -40100), (-250, -105), (-102.625, -105),
 (-102.625, -80), (-90.125, 80), (-78, 95),
 (75.75, 95), (92.375, 80), (100.375,-80),
 (100.375, -110), (109.875, -110), (122, -105),
 (150,-105), (150, -40100),
 POLYGON_END,

 POLYGON_BEGIN,
 (-250, -40100), (-250, -100), (-113.438, -100),
 (-109.438, -80), (-94.875, 85), (-78, 100),
 (82, 100), (94.875, 85), (105.125,-80),
 (105.125, -90), (105.125, -105), (122, -100),
 (150, -100), (150, -40100),
 POLYGON_END,

POLYGON_BEGIN,
 (-250, -40100), (-250, -110), (-91.8124, -110),
 (-91.8124, -80), (-85.375, 75), (-73.74, 90),
 (75.75, 90), (89.875, 75), (95.5,-80),
 (95.5, -115), (114.625, -115), (122, -110),
 (150,-110), (150, -40100),
 POLYGON_END,
REGION_END,
TEST_END,                       # test ends
```

Fixed amplitude E4 one pulse mask example

The following example specifies a fixed amplitude E4 one pulse mask. *Margins are not supported by the mask.*

```
TEST_BEGIN,              # specifies start of test
 1,                      # version
 "E4 ONE",               # mask label
 1.393E+08,              # mask bit rate
 GENERAL,                # waveform type
 STANDARD,               # standard fit type
 1,                      # maximum number of hits allowed
 0,                      # default mask margin percentage
 2,                      # test duration (in seconds)
 CENTER,                 # mask time reference
 FALL,                   # time edge reference
 BOT,                    # voltage reference
 REGION_BEGIN,           # start of region specification
FIXED,                   # fixed amplitude mask
0.5,                     # top voltage value for scaling
-0.5,                    # bottom voltage value for scaling
NO_MARGIN,               # margins not supported
POLYGON_BEGIN,
 (-75.0,90.0),(-65.3,80.0),(-51.4,0.0),(-51.4,-120.0),
 (57.0,-120.0),(57.0,0.0),(70.9,80.0),(75.0,90.0),(75.0,-600.0),
 (-75.0,-600.0),         # polygon geometrical specification
 POLYGON_END,            # specification ends
 REGION_END,             # region specification ends

 REGION_BEGIN,
 FIXED,
 0.5,
 -0.5,
 NO_MARGIN,
 POLYGON_BEGIN,
 (-75.0,120.0),(-48.6,120.0),(-48.6,0.0),(-34.7,-80.0),
 (-25.0,-90.0),(-18.8,-80.0),(18.8,-80.0),(25.0,-90.0),
 (29.1,-80.0),(43.0,0.0),(43.0,120.0),(75.0,120.0),
 (75.0,600.0),(-75.0,600.0),
 POLYGON_END,
 REGION_END,
 TEST_END              # test ends
```

3

Reference Information

Reference Information

What you'll find in this chapter

This chapter contains reference information for the remote programming module including its operating characteristics.

Operating Characteristics

Operating characteristics are specified with the remote programming module installed on the HP 83475B Lightwave Communication Analyzer. Refer to the *HP 83475B Lightwave Communications Analyzer User's Guide* for specifications on the HP 83475B.

Fast Fourier transforms

Test region	Each pixel is selectable to be tested or not.
Inputs	On either ch1, ch2, or F1
Frequency cursor resolution	From 1.22 mHz (millihertz) to 9.766 MHz
Points	Fixed at 1024
Peak Find	Peak Find automatically snaps cursor to the two largest peaks located anywhere in the displayed frequency span. Measurement information is automatically displayed at the bottom of the screen together with the difference in frequency between the two selected peaks.
Variable sensitivity and offset	Sensitivity and vertical offset (position) are controlled from the front panel to display an optimum view of the spectrum. Sensitivity is calibrated in dB per divisions; vertical offset is calibrated in dBV.
Time record length	10× main sweep speed.

Magnifying the display

As the frequency span is changed, the display is magnified about the center frequency so that you get an expanded view.

Selectable windows

Four windows are selectable.

1. Hanning, for best frequency resolution and general purpose use.
2. Flattop, for best amplitude accuracy.
3. Rectangular, for single-shot signals such as transients and signals where there are an integral number of cycles in the time record.
4. Exponential for best transient analysis.

Window characteristics

Window	Highest Side Lobe (dB)	3dB Bandwidth (bins)	6dB Bandwidth (bins)	Scallop Loss (dB)
Rectangular	−13	0.89	1.21	3.92
Hanning	−32	1.44	2.00	1.42
Flattop	−70	3.38	4.17	0.005

FFT frequency range

dc to 500 MHz

Frequency span control

This control allows you to specify the frequency span of the FFT display. When the span is adjusted, the display will expand or contract about the center frequency as set by the Center Frequency control. Refer to Figure 3-1 for the limits of the Frequency Span control.

Center frequency control

This control allows you to specify the frequency at the center of the FFT display. When the Frequency Span is changed, the FFT display will expand or contract about the frequency at the center of the display. Refer to Figure 3-1 for the limits of this control.

Move 0 Hz to left

Pressing this softkey will move the FFT display so that the left-hand edge of the display will be 0 Hz.

FFT vector display

When the time domain display is turned off, the FFT display will be displayed in vector drawing mode. The time domain display can be turned off by pressing the `Channel #` key a couple of times.

Display

FFT vertical units in dB.

Units/div

This control allows you to adjust the vertical scaling of the FFT display in a 1-2-5 sequence from 1 dB/div to 50 dB/div.

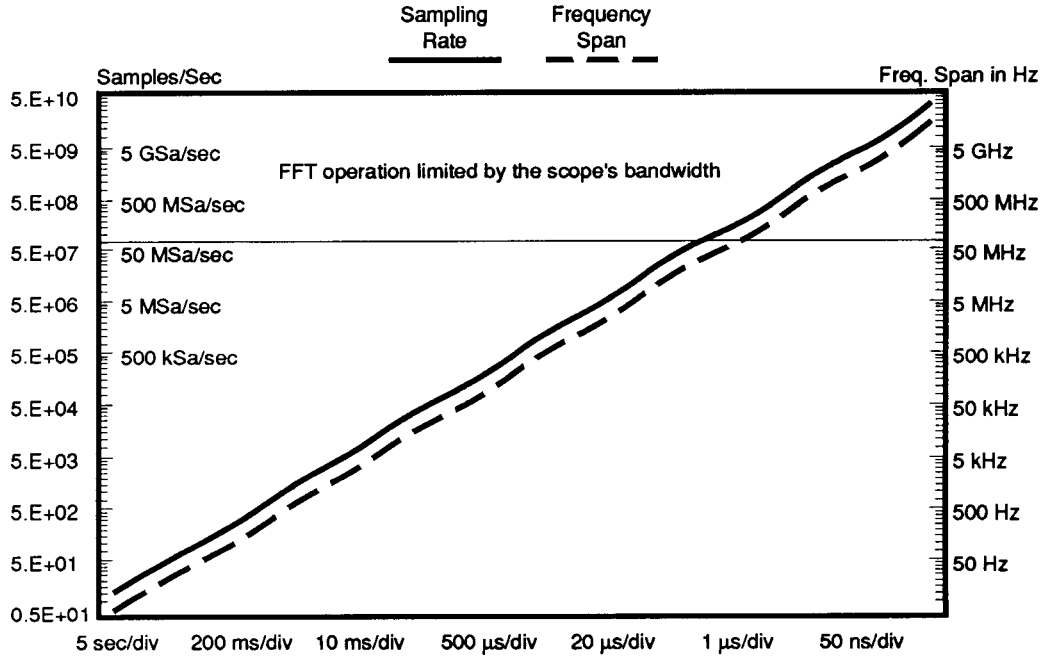
Reference level

This control adjusts the reference level of the FFT display across a range of 400 dBV. The minimum setting is −196 dB at 1 dBV\division, decreasing to 0 dB at 50 dBV/div. The maximum setting is 400 dB at 50 dBV/division, decreasing to 204 dB at 1 dBV/div.

Programmability

All front-panel controls are fully programmable over HP-IB (HP 83475B Option 201) or RS-232 (HP 83475B Option 202).

HP 54657A/54658A FFT Operation
Frequency Span and Effective Sampling Rate vs Sweep Speed



FFT inoperative at 2 ns/div

Sweep Speed	Sampling Rate	Frequency Span
5 sec/div	20 Hz	9.75 Hz
2 sec/div	50 Hz	24.4 Hz
1 sec/div	100 Hz	48.85 Hz
500 ms/div	200 Hz	97.5 Hz
200 ms/div	500 Hz	244 Hz
100 ms/div	1 kHz	488.5 Hz
50 ms/div	2 kHz	975 Hz
20 ms/div	5 kHz	2.44 kHz
10 ms/div	10 kHz	4.885 kHz
5 ms/div	20 kHz	9.75 kHz
2 ms/div	50 kHz	24.4 kHz
1 ms/div	100 kHz	48.85 kHz
500us/div	200 kHz	97.5 kHz
200 us/div	500 kHz	244 kHz

Sweep Speed	Sampling Rate	Frequency Span
100 us/div	1 MHz	488.5 kHz
50 us/div	2 MHz	975 kHz
20 us/div	5 MHz	2.44 MHz
10 us/div	10 MHz	4.885 MHz
5 us/div	20 MHz	9.75 MHz
2 us/div	50 MHz	24.4 MHz
1 us/div	100 MHz	48.85 MHz
500 ns/div	200 MHz	97.5 MHz
200 ns/div	500 MHz	244 MHz
100 ns/div	1 GHz	488.5 MHz
50 ns/div	2 GHz	975 MHz
20 ns/div	5 GHz	2.44 GHz
10 ns/div	10 GHz	4.885 GHz
5 ns/div	20 GHz	9.75 GHz

Figure 3-1. Frequency span and effective sampling rate vs. sweep speed.

Trace memory (all nonvolatile)

1 through 3	High-speed storage without compression.
4 through 100	Storage with compression, number of traces is a function of complexity. Storage time is less than 10 seconds.

Real-time clock

The real-time clock can be set from the front panel in a 24-hour format with battery back-up.

Hardcopy output

Printer/plotter supported	HP ThinkJet, HP QuietJet, HP PaintJet, and HP LaserJet printer. HP-GL compatible plotters.
Option 202 only	Epson FX-80 or compatible printer.

RS-232 configurations

Connector type	With the adapter cable connected, at the end of the cable is a 25-pin DTE port; a printer cable is required to connect it to hardcopy devices or a computer.
Protocols	XON/XOFF, hardware
Data bits	Eight (8)
Stop bits	One (1)
Parity	None
Baud rates	1200, 2400, 9600, 19200

Programmability

All instrument settings and operating modes may be remotely programmed via RS-232 and HP-IB (IEEE-488).

4

Remote Programming

Remote Programming

What you'll find in this chapter

This chapter introduces the basics for remote programming of an analyzer. Sections covering the following topics are included:

- provides an introduction to programming
- gets you started programming the analyzer
- describes the HP-IB interface functions
- describes the interface functions and programming over RS-232-C

Introduction to Programming

The programming instructions in this manual conform to the IEEE 488.2 Standard Digital Interface for Programmable Instrumentation. The programming instructions provide the means of remote control.

To program the HP 83475B Lightwave Communication Analyzer, it is necessary to add either an HB-IB or RS-232-C interface to the rear panel of the analyzer.

You can perform the following basic operations with a controller and an analyzer:

- Set up the instrument.
- Make measurements.
- Get data (waveform, measurements, configuration) from analyzer.
- Send information (pixel image, configurations) to analyzer.

Other tasks are accomplished by combining these basic functions.

NOTE

The programming examples for individual commands in this manual are written in HP BASIC 5.0 for an HP 9000 Series 200/300 Controller.

Talking to the instrument

Computers acting as controllers communicate with the instrument by sending and receiving messages over a remote interface. Instructions for programming normally appear as ASCII character strings embedded inside the output statements of a "host" language available on your controller. The input statements of the host language are used to read in responses from the analyzer.

For example, HP 9000 Series 200/300 BASIC uses the OUTPUT statement for sending commands and queries. After a query is sent, the response is usually read in using the ENTER statement.

Messages are placed on the bus using an output command and passing the device address, program message, and terminator. Passing the device address ensures that the program message is sent to the correct interface and instrument.

The following HP BASIC statement sends a command which sets the bandwidth limit of channel 1 on:

```
OUTPUT < device address > ;":CHANNEL1:BWLIMIT ON"<terminator>
```

The < device address > represents the address of the device being programmed. Each of the other parts of the above statement are explained in the following pages.

Program message syntax

To program the instrument remotely, you must have an understanding of the command format and structure expected by the instrument. The IEEE 488.2 syntax rules govern how individual elements such as headers, separators, program data, and terminators may be grouped together to form complete instructions. Syntax definitions are also given to show how query responses are formatted. Figure 4-1 shows the main syntactical parts of a typical program statement.

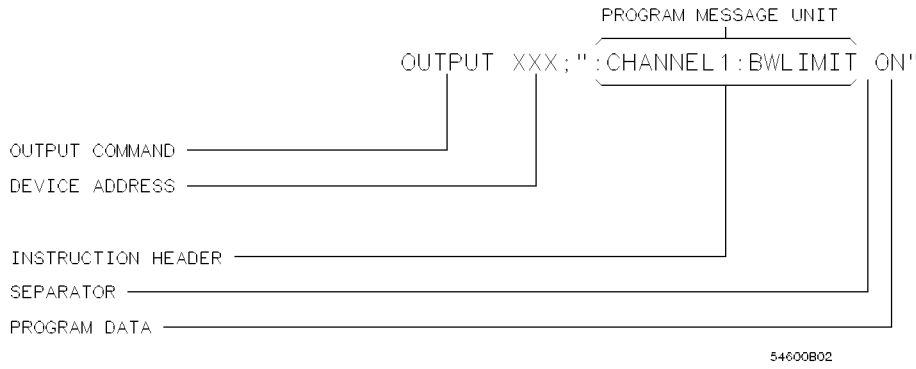


Figure 4-1. Program message syntax.

Output command

The output command is entirely dependent on the programming language. Throughout this manual, HP 9000 Series 200/300 BASIC 5.0 is used in the examples of individual commands. If you are using other languages, you will need to find the equivalents of HP BASIC commands like OUTPUT, ENTER, and CLEAR in order to convert the examples. The instructions listed in this manual are always shown between quotes in the example programs. This manual also has examples using Microsoft QuickBASIC and Microsoft Quick C.

Device address

The location where the device address must be specified is also dependent on the programming language you are using. In some languages, this may be specified outside the output command. In HP BASIC, this is always specified after the keyword OUTPUT. The examples in this manual assume the analyzer is at device address 707. When writing programs, the address varies according to how the bus is configured.

Instructions

Instructions (both commands and queries) normally appear as a string embedded in a statement of your host language, such as BASIC, Pascal, or C. The only time a parameter is not meant to be expressed as a string is when the instruction's syntax definition specifies <block data>, such as learnstring. There are only a few instructions which use block data.

Instructions are composed of two main parts:

- The header, which specifies the command or query to be sent.
- The program data, which provide additional information needed to clarify the meaning of the instruction.

Instruction header

The instruction header is one or more mnemonics separated by colons (:) that represent the operation to be performed by the instrument. The command trees in Figure 5-1 and Figure 5-2 illustrate how all the mnemonics can be joined together to form a complete header (see Chapter 5, "Programming and Documentation Conventions").

The example in Figure 4-1 is a command. Queries are indicated by adding a question mark (?) to the end of the header. Many instructions can be used as either commands or queries, depending on whether or not you have included the question mark. The command and query forms of an instruction usually have different program data. Many queries do not use any program data.

White space (separator)

White space is used to separate the instruction header from the program data. If the instruction does not require any program data parameters, you do not need to include any white space. In this manual, white space is defined as one or more spaces. ASCII defines a space to be character 32 (in decimal).

Program data

Program data is used to clarify the meaning of the command or query. It provides necessary information, such as whether a function should be on or off, or which waveform is to be displayed. Each instruction's syntax definition shows the program data, as well as the acceptable values. The section "Program Data Syntax Rules" in this chapter lists the general rules about acceptable values.

When there is more than one data parameter, the parameters are separated by commas (,). Spaces can be added around the commas to improve readability.

Header types

There are three types of headers:

- simple command headers
- compound command headers
- common command headers

Simple command header. Simple command headers contain a single mnemonic. AUTOSCALE and DIGITIZE are examples of simple command headers typically used in this instrument. The syntax is:

`<program mnemonic><terminator>`

When program data must be included with the simple command header (for example, :DIGITIZE CHAN1), white space is added to separate the data from the header. The syntax is:

`<program mnemonic><separator><program data><terminator>`

Compound command header. Compound command headers are a combination of two program mnemonics. The first mnemonic selects the subsystem, and the second mnemonic selects the function within that subsystem. The mnemonics within the compound message are separated by colons. For example:

To execute a single function within a subsystem:

`<subsystem>:<function><separator><program data><terminator>`

(For example :CHANNEL1:BWLIMIT ON)

Common command header. Common command headers control IEEE 488.2 functions within the instrument (such as clear status). Their syntax is:

`*<command header><terminator>`

No space or separator is allowed between the asterisk (*) and the command header. *CLS is an example of a common command header.

Combining commands

To execute more than one function within the same subsystem, a semi-colon (;) is used to separate the functions:

```
:<subsystem>:<function><separator><data>;<function><separator>
<data><terminator>
```

(For example :CHANNEL1:COUPLING DC;BWLIMIT ON)

Duplicate mnemonics

Identical function mnemonics can be used for more than one subsystem. For example, the function mnemonic RANGE may be used to change the vertical range or to change the horizontal range:

- To set the vertical range of channel 1 to 0.4 volts full scale, use:

```
:CHANNEL1:RANGE .4 V
```

- To set the horizontal time base to 1 second full scale, use:

```
:TIMEBASE:RANGE 1
```

CHANNEL1 and TIMEBASE are subsystem selectors and determine which range is being modified.

Query command

Command headers immediately followed by a question mark (?) are queries. After receiving a query, the instrument interrogates the requested function and places the answer in its output queue. The answer remains in the output queue until it is read or another command is issued. When read, the answer is transmitted across the bus to the designated listener (typically a controller).

For example, the query :TIMEBASE:RANGE? places the current time base setting in the output queue. In HP BASIC, the controller input statement:

```
ENTER <device address> ;Range
```

passes the value across the bus to the controller and places it in the variable Range.

Query commands are used to find out how the instrument is currently configured. They are also used to get results of measurements made by the instrument. For example, the command :MEASURE:RISETIME? instructs the instrument to measure the rise time of your waveform and place the result in the output queue.

The output queue must be read before the next program message is sent. For example, when you send the query :MEASURE:RISETIME? you must follow that query with an input statement. In HP BASIC, this is usually done with an ENTER statement immediately followed by a variable name. This statement reads the result of the query and places the result in a specified variable.

NOTE

Sending another command or query, before reading the result of a query, causes the output buffer to be cleared and the current response to be lost. This also generates a query interrupted error in the error queue.

Program header options

Program headers can be sent using any combination of uppercase or lowercase ASCII characters. Instrument responses, however, are always returned in uppercase.

Program command and query headers may be sent in either long form (complete spelling), short form (abbreviated spelling), or any combination of long form and short form.

```
TIMEBASE:DELAY 1US  - long form

TIM:DEL 1US  - short form
```

Programs written in long form are easily read and are almost self-documenting. The short form syntax conserves the amount of controller memory needed for program storage and reduces the amount of I/O activity.

NOTE

The rules for the short form syntax are shown in chapter 5, "Programming and Documentation Conventions."

Program data syntax rules

Program data is used to convey a variety of types of parameter information related to the command header. At least one space must separate the command header or query header from the program data.

```
<program mnemonic><separator><data><terminator>
```

When a program mnemonic or query has multiple program data, a comma separates sequential program data.

```
<program mnemonic><separator><data>,<data><terminator>
```

For example, :MEASURE:TVOLT 1.0V,2 has two program data: 1.0V and 2.

There are two main types of program data which are used in commands:

- character program data
- numeric program data

Character program data

Character program data is used to convey parameter information as alpha or alphanumeric strings. For example, the :TIMEBASE:MODE command can be set to normal, delayed, or xy. The character program data, in this case, may be NORMAL, DELAYED, or XY. :TIMEBASE:MODE DELAYED sets the time base mode to delayed.

The available mnemonics for character program data are always included with the instruction's syntax definition. When sending commands, either the long form or short form (if one exists) may be used. Uppercase and lowercase letters may be mixed freely. When receiving responses, uppercase letters are used exclusively.

Numeric program data

Some command headers require program data to be expressed numerically. For example, :TIMEBASE:RANGE requires the desired full scale range to be expressed numerically.

For numeric program data, you have the option of using exponential notation or using suffix multipliers to indicate the numeric value. The following numbers are all equal:

28 = 0.28E2 = 280e-1 = 28000m = 0.028K = 28e-3K

When a syntax definition specifies that a number is an integer, that means that the number should be whole. Any fractional part would be ignored, truncating the number. Numeric data parameters which accept fractional values are called real numbers.

All numbers are expected to be strings of ASCII characters. Thus, when sending the number 9, you would send a byte representing the ASCII code for the character "9" (which is 57). A three-digit number like 102 would take up three bytes (ASCII codes 49, 48, and 50). This is taken care of automatically when you include the entire instruction in a string.

Embedded strings

Embedded strings contain groups of alphanumeric characters which are treated as a unit of data by the analyzer. For example, the line of text written to the advisory line of the instrument with the :SYSTEM:DSP command:

```
:SYSTEM:DSP"This is a message."
```

Embedded strings may be delimited with either single (') or double (") quotes. These strings are case-sensitive and spaces act as legal characters just like any other character.

Program message terminator

The program instructions within a data message are executed after the program message terminator is received. The terminator may be either an NL (New Line) character, an EOI (End-Or-Identify) asserted in the HP-IB interface, or a combination of the two. Asserting the EOI sets the EOI control line low on the last byte of the data message. The NL character is an ASCII linefeed (decimal 10).

NOTE

The NL (New Line) terminator has the same function as an EOS (End Of String) and EOT (End Of Text) terminator.

Selecting multiple subsystems

You can send multiple program commands and program queries for different subsystems on the same line by separating each command with a semicolon. The colon following the semicolon enables you to enter a new subsystem. For example:

```
<program mnemonic><data>;:<program
mnemonic><data><terminator>

:CHANNEL1:RANGE 0.4 V;:TIMEBASE:RANGE 1
```

NOTE

Multiple commands may be any combination of compound and simple commands.

Programming: Getting Started

What you'll find in this section

This section provides information on:

- how to setup the instrument
- how to retrieve setup information
- how to retrieve measurement results
- how to digitize a waveform
- how to pass data to the controller

NOTE

The programming examples in this manual are written in HP BASIC 5.0 for an HP 9000 Series 200/300 Controller.

Initialization

To make sure the bus and all appropriate interfaces are in a known state, begin every program with an initialization statement. HP BASIC provides a CLEAR command which clears the interface buffer:

```
CLEAR 707  ! initializes the interface of the instrument
```

When you are using HP-IB, CLEAR also resets the analyzer's parser. The parser is the program which reads in the instructions which you send it.

After clearing the interface, initialize the instrument to a preset state:

```
OUTPUT 707;"*RST"  ! initializes the instrument to a preset state
```

NOTE

The actual commands and syntax for initializing the instrument are discussed in chapter 7, "Command Dictionary."

Refer to your controller manual and programming language reference manual for information on initializing the interface.

Autoscale

The AUTOSCALE feature of the analyzer performs a very useful function on unknown waveforms by setting up the vertical channel, time base, and trigger level of the instrument.

The syntax for the autoscale function is:

```
:AUTOSCALE<terminator>
```

Setting up the instrument

A typical analyzer setup would set the vertical range and offset voltage, the horizontal range, delay time, delay reference, trigger mode, trigger level, and slope. A typical example of the commands sent to the analyzer are:

```
:CHANNEL1:PROBE X10;RANGE 16 V;OFFSET 1.00 V<terminator>
```

```
:TIMEBASE:MODE NORMAL;RANGE 1E-3;DELAY 100E-6<terminator>
```

This example sets the timebase at 1 ms full-scale (100 μs/div) with delay of 100 μs. Vertical is set to 16 V full-scale (2 V/div) with center of screen at 1 V and probe attenuation of 10.

Example program

This program demonstrates the basic command structure used to program the analyzer.

```
10    CLEAR 707                              ! Initialize instrument interface
20    OUTPUT 707;"*RST"                      ! Initialize instrument to preset state
30    OUTPUT 707;":TIMEBASE:RANGE 5E-4"      ! Time base to 50 us/div
40    OUTPUT 707;":TIMEBASE:DELAY 0"         ! Delay to zero
50    OUTPUT 707;":TIMEBASE:REFERENCE CENTER" ! Display reference at center
60    OUTPUT 707;":CHANNEL1:PROBE X10"       ! Probe attenuation to 10:1
70    OUTPUT 707;":CHANNEL1:RANGE 1.6 V"     ! Vertical range to 1.6 V full scale
80    OUTPUT 707;":CHANNEL1:OFFSET -.4 V"    ! Offset to -0.4
90    OUTPUT 707;":CHANNEL1:COUPLING DC"     ! Coupling to DC
100   OUTPUT 707;":TRIGGER:MODE NORMAL"      ! Normal triggering
110   OUTPUT 707;":TRIGGER:LEVEL -.4 V"      ! Trigger level to -0.4
120   OUTPUT 707;":TRIGGER:SLOPE POSITIVE"   ! Trigger on positive slope
130   OUTPUT 707;":ACQUIRE:TYPE NORMAL"      ! Normal acquisition
140   OUTPUT 707;":DISPLAY:GRID OFF"         ! Grid off
150   END
```

Program overview

Line 10	initializes the instrument interface to a known state.
Line 20	initializes the instrument to a preset state.
Lines 30 through 50	set the time base mode to normal with the horizontal time at 50 μs/div with 0 s of delay referenced at the center of the graticule.
Lines 60 through 90	set the vertical range to 1.6 volts full scale with center screen at -0.4 volts with 10:1 probe attenuation and DC coupling.
Lines 100 through 120	configures the instrument to trigger at -0.4 volts with normal triggering.
Line 130	configures the instrument for normal acquisition.
Line 140	turns the grid off.

Using the Digitize command

The Digitize command is a macro that captures data satisfying the specifications set up by the ACQUIRE subsystem. When the digitize process is complete, the acquisition is stopped. The captured data can then be measured by the instrument or transferred to the controller for further analysis. The captured data consists of two parts: the waveform data record and the preamble.

NOTE

After changing the analyzer configuration, the waveform buffers are cleared. Before doing a measurement, the Digitize command should be sent to ensure new data has been collected.

When the DIGITIZE command is sent to an instrument, the specified channel signal is digitized with the current ACQUIRE parameters. To obtain waveform data, you must specify the WAVEFORM parameters for the waveform data prior to sending the :WAVEFORM:DATA? query.

The number of data points comprising a waveform varies according to the number requested in the ACQUIRE subsystem. The ACQUIRE subsystem determines the number of data points, type of acquisition, and number of averages used by the DIGITIZE command. This allows you to specify exactly what the digitized information contains. The following program example shows a typical setup:

```
OUTPUT 707;":ACQUIRE:TYPE AVERAGE"<terminator>
OUTPUT 707;":ACQUIRE:COMPLETE 100"<terminator>
OUTPUT 707;":WAVEFORM:SOURCE CHANNEL1"<terminator>
OUTPUT 707;":WAVEFORM:FORMAT BYTE"<terminator>
OUTPUT 707;":ACQUIRE:COUNT 8"<terminator>
OUTPUT 707;":WAVEFORM:POINTS 500"<terminator>
OUTPUT 707;":DIGITIZE CHANNEL1"<terminator>
OUTPUT 707;":WAVEFORM:DATA?"<terminator>
```

This setup places the instrument into the averaged mode with eight averages. This means that when the DIGITIZE command is received, the command will execute until the signal has been averaged at least eight times.

After receiving the :WAVEFORM:DATA? query, the instrument will start passing the waveform information when addressed to talk.

Digitized waveforms are passed from the instrument to the controller by sending a numerical representation of each digitized point. The format of the numerical representation is controlled with the :WAVEFORM:FORMAT command and may be selected as BYTE, WORD, or ASCII.

The easiest method of entering a digitized waveform depends on data structures, formatting available and I/O capabilities. You must scale the integers to determine the amplitude value of each point. These integers are passed starting with the left-most point on the instrument's display. For more information, refer to chapter 6, "Waveform Commands."

NOTE

When using HP-IB, a digitize operation may be aborted by sending a Device Clear over the bus (CLEAR 707).

Receiving information from the analyzer

After receiving a query (command header followed by a question mark), the analyzer interrogates the requested function and places the answer in its output queue. The answer remains in the output queue until it is read or another command is issued. When read, the answer is transmitted across the interface to the designated listener (typically a controller). The input statement for receiving a response message from an instrument's output queue typically has two parameters; the device address, and a format specification for handling the response message. For example, to read the result of the query command :CHANNEL1:COUPLING? you would execute the HP BASIC statement:

```
ENTER <device address>;Setting$
```

where <device address> represents the address of your device. This would enter the current setting for the channel one coupling in the string variable Setting$.

All results for queries sent in a program message must be read before another program message is sent. For example, when you send the query :MEASURE:RISETIME?, you must follow that query with an input statement. In HP BASIC, this is usually done with an ENTER statement.

Sending another command, before reading the result of the query, causes the output buffer to be cleared and the current response to be lost. This also causes an error to be placed in the error queue.

Executing an input statement, before sending a query, causes the controller to wait indefinitely.

The format specification for handling response messages is dependent on both the controller and the programming language.

String variables

The output of the instrument may be numeric or character data, depending on what is queried. Refer to the specific commands for the formats and types of data returned from queries.

NOTE

In HP BASIC 5.0, string variables are case sensitive and must be expressed exactly the same each time they are used.

NOTE

For the example programs, assume that the device being programmed is at device address 707. The actual address varies according to how you have configured the bus for your own application.

The following example shows the data being returned to a string variable:

```
10 DIM Rang$[30]
20 OUTPUT 707;":CHANNEL1:RANGE?"
30 ENTER 707;Rang$
40 PRINT Rang
50 END
```

After running this program, the controller displays:

```
+8.00000E-01
```

Numeric variables

The following example shows the data being returned to a numeric variable:

```
10 OUTPUT 707;":CHANNEL1:RANGE?"
20 ENTER 707;Rang
30 PRINT Rang
40 END
```

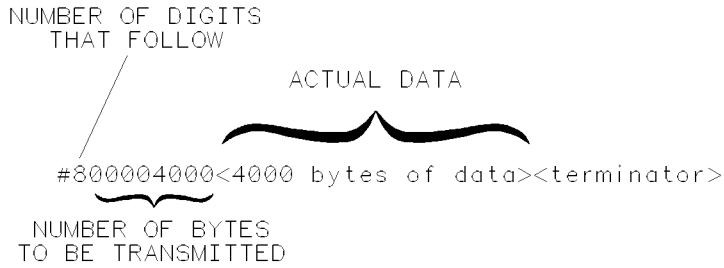
After running this program, the controller displays:

```
.8
```

Definite-length block response data

Definite-length block response data allows any type of device-dependent data to be transmitted over the system interface as a series of 8-bit binary data bytes. This is particularly useful for sending large quantities of data or 8-bit extended ASCII codes. The syntax is a pound sign (#) followed by a non-zero digit representing the number of digits in the decimal integer. After the non-zero digit is the decimal integer that states the number of 8-bit data bytes being sent. This is followed by the actual data.

For example, for transmitting 4000 bytes of data, the syntax would be:



The “8” states the number of digits that follow, and “00004000” states the number of bytes to be transmitted.

Multiple queries

You can send multiple queries to the instrument within a single program message, but you must also read them back within a single program message. This can be accomplished by either reading them back into a string variable or into multiple numeric variables. For example, you could read the result of the query :TIMEBASE:RANGE?;DELAY? into the string variable Results$ with the command:

```
ENTER 707;Results$
```

When you read the result of multiple queries into string variables, each response is separated by a semicolon. For example, the response of the query :TIMEBASE:RANGE?;DELAY? would be:

```
<range_value>; <delay_value>
```

Use the following program message to read the query :TIMEBASE:RANGE?;DELAY? into multiple numeric variables:

```
ENTER 707;Result1,Result2
```

Instrument status

Status registers track the current status of the instrument. By checking the instrument status, you can find out whether an operation has been completed, whether the instrument is receiving triggers, and more.

Programming over HP-IB

What you'll find in this section

This section describes the HP-IB interface functions and some general concepts. In general, these functions are defined by IEEE 488.1. They deal with general interface management issues, as well as messages which can be sent over the interface as interface commands.

Interface capabilities

The interface capabilities of the analyzer, as defined by IEEE 488.1, are SH1, AH1, T5, L4, SR1, RL1, PP1, DC1, DT1, C0, and E2.

Command and data concepts

The interface has two modes of operation:

- command mode
- data mode

The bus is in the command mode when the ATN line is true. The command mode is used to send talk and listen addresses and various bus commands, such as a group execute trigger (GET).

The bus is in the data mode when the ATN line is false. The data mode is used to convey device-dependent messages across the bus.

Addressing

By using the front-panel controls, the instrument must be placed in the "connect to computer" mode. Figure 4-2 is the Utility/Print menu after the HP-IB interface has been installed on the rear panel of the analyzer. Use this menu to set the HP-IB address for the analyzer and "connect to computer".

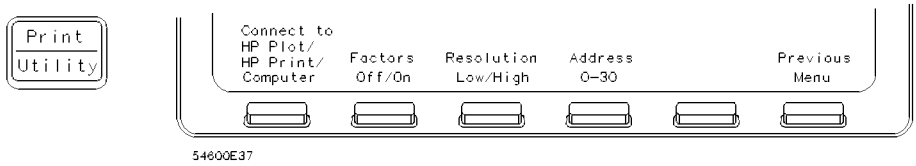


Figure 4-2. Analyzer utility/print menu with HP-IB.

- Each device on the HP-IB resides at a particular address, 0–30.
- The active controller specifies which devices talk and which listen.
- An instrument may be talk addressed, listen addressed, or unaddressed by the controller.

If the controller addresses the instrument to talk, the instrument remains configured to talk until it receives an interface clear message (IFC), another instrument's talk address (OTA), its own listen address (MLA), or a universal untalk command (UNT).

If the controller addresses the instrument to listen, the instrument remains configured to listen until it receives an interface clear message (IFC), its own talk address (MTA), or a universal unlisten command (UNL).

Communicating over the bus

Since HP-IB can address multiple devices through the same interface card, the device address passed with the program message must include not only the correct interface select code, but also the correct instrument address.

Interface select code (selects interface)

Each interface card has a unique interface select code. This code is used by the controller to direct commands and communications to the proper interface. The default is typically "7" for HP-IB controllers.

Instrument address (selects instrument)

Each instrument on an HP-IB must have a unique instrument address between decimal 0 and 30. The device address passed with the program message must include not only the correct instrument address, but also the correct interface select code.

```
DEVICE ADDRESS = (Interface Select Code * 100) + (Instrument Address)
```

For example, if the instrument address for the analyzer is 4 and the interface select code is 7, when the program message is passed, the routine performs its function on the instrument at device address 704.

For the analyzer, the instrument address is typically set to "7" at the factory. This address can be changed in the HP-IB menu.

NOTE

The examples in this manual assume the analyzer is at device address 707.

Lockout

A SYSTem:LOCK command may be used to disable front-panel control while a program is running. By default, the instrument accepts and executes bus commands, and the front panel is entirely active.

> **NOTE**
>
> Cycling the power also restores front panel control.

With HP-IB, the instrument is placed in the lockout mode by sending the local lockout command (LLO). The instrument can be returned to local by sending the go-to-local command (GTL) to the instrument.

Bus commands

The following commands are IEEE 488.1 bus commands (ATN true). IEEE 488.2 defines many of the actions which are taken when these commands are received by the instrument.

Device clear

The device clear (DCL) or selected device clear (SDC) commands clear the input and output buffers, reset the parser, and clear any pending commands. If either of these commands is sent during a digitize operation, the digitize operation is aborted.

Interface clear

The interface clear (IFC) command halts all bus activity. This includes unaddressing all listeners and the talker, disabling serial poll on all devices, and returning control to the system controller.

Programming over RS-232-C

What you'll find in this section

This section describes the interface functions and some general concepts of the RS-232-C interface. The RS-232-C interface on this instrument is Hewlett-Packard's implementation of EIA Recommended Standard RS-232-C, "Interface Between Data Terminal Equipment and Data Communications Equipment Employing Serial Binary Data Interchange." With this interface, data is sent one bit at a time and characters are not synchronized with preceding or subsequent data characters. Each character is sent as a complete entity without relationship to other events.

NOTE

IEEE 488.2 is designed to work with IEEE 488.1 as the physical interface. When RS-232-C is used as the physical interface, as much of IEEE 488.2 is retained as the hardware differences will allow. No IEEE 488.1 messages such as DCL, GET, and END are available.

Interface operation

The analyzer can be programmed with a controller over RS-232-C using either a minimum three-wire or extended hardwire interface. The operation and exact connections for these interfaces are described in more detail in the following sections. When you are programming an analyzer over RS-232-C with a controller, you are normally operating directly between two DTE (Data Terminal Equipment) devices as compared to operating between a DTE device and a DCE (Data Communications Equipment) device.

When operating directly between two RS-232-C devices, certain considerations must be taken into account. For three-wire operation, XON/XOFF must be used to handle protocol between the devices. For extended hardwire operation, protocol may be handled either with XON/XOFF or by manipulating the CTS and RTS lines of the analyzer. For both three-wire and extended hardwire operation, the DCD and DSR inputs to the analyzer must remain high for proper operation.

With extended hardwire operation, a high on the CTS input allows the analyzer to send data and a low on this line disables the analyzer data transmission. Likewise, a high on the RTS line allows the controller to send data and a low on this line signals a request for the controller to disable data transmission. Since three-wire operation has no control over the CTS input, internal pull-up resistors in the analyzer makes sure that this line remains high for proper three-wire operation.

Cables

Selecting a cable for the RS-232-C interface is dependent on your specific application. The following paragraphs describe which lines of the analyzer are used to control the operation of the RS-232-C bus relative to the analyzer. To locate the proper cable for your application, refer to the reference manual for your controller. This manual should address the exact method your controller uses to operate over the RS-232-C bus.

Minimum 3-wire interface with software protocol

With a 3-wire interface, the SOFTWARE (as compared to interface hardware) controls the data flow between the analyzer and the controller. This provides a much simpler connection between devices since you can ignore hardware handshake requirements. The analyzer uses the following connections on its RS-232-C interface for 3-wire communication:

- Pin-7 SGND (Signal Ground)
- Pin-2 TD (Transmit Data from analyzer)
- Pin-3 RD (Receive Data into analyzer)

The TD (Transmit Data) line from the analyzer must connect to the RD (Receive Data) line on the controller. Likewise, the RD line from the analyzer must connect to the TD line on the controller. Internal pull-up resistors in the analyzer insure the DCD, DSR, and CTS lines remain high when you are using a three-wire interface.

NOTE

The three-wire interface provides no hardware means to control data flow between the controller and the analyzer. XON/OFF protocol is the only means to control this data flow.

Extended interface with hardware handshake

With the extended interface, both the software and the hardware can control the data flow between the analyzer and the controller. This allows you to have more control of data flow between devices. The analyzer uses the following connections on its RS-232-C interface for extended interface communication:

- Pin-7 SGND (Signal Ground)
- Pin-2 TD (Transmit Data from analyzer)
- Pin-3 RD (Receive Data into analyzer)

The additional lines you use depends on your controller's implementation of the extended hardware interface.

- Pin-4 RTS (Request To Send) is an output from the analyzer which can be used to control incoming data flow.
- Pin-5 CTS (Clear To Send) is an input to the analyzer which controls data flow from the analyzer.
- Pin-6 DSR (Data Set Ready) is an input to the analyzer which controls data flow from the analyzer within two bytes.
- Pin-8 DCD (Data Carrier Detect) is an input to the analyzer which controls data flow from the analyzer within two bytes.
- Pin-20 DTR (Data Terminal Ready) is an output from the analyzer which is enabled as long as the analyzer is turned on.

The TD (Transmit Data) line from the analyzer must connect to the RD (Receive Data) line on the controller. Likewise, the RD line from the analyzer must connect to the TD line on the controller.

The RTS (Request To Send) is an output from the analyzer which can be used to control incoming data flow. A high on the RTS line allows the controller to send data and a low on this line signals a request for the controller to disable data transmission.

The CTS (Clear To Send), DSR (Data Set Ready), and DCD (Data Carrier Detect) lines are inputs to the analyzer which control data flow from the analyzer (Pin 2). Internal pull-up resistors in the analyzer assure the DCD and DSR lines remain high when they are not connected.

If DCD or DSR are connected to the controller, the controller must keep these lines and the CTS line high to enable the analyzer to send data to the controller. A low on any one of these lines will disable the analyzer data transmission. Dropping the CTS line low during data transmission will stop analyzer data transmission immediately. Dropping either the DSR or DCD line low during data transmission will stop analyzer data transmission, but as many as two additional bytes may be transmitted from the analyzer.

Configuring the interface

By using the front-panel menus, the RS-232-C interface can be placed in either the printer mode or the controller mode. The printer mode should be used when you want the instrument to talk directly to a printer over RS-232-C, without the aid of a controller. The controller mode is used when the instrument will operate in conjunction with a controller over RS-232-C. Figure 4-3 is the analyzer Print/Utility menu after the RS-232-C has been installed on the rear panel. Use this menu to "connect to computer", and assign the baud rate and protocol.

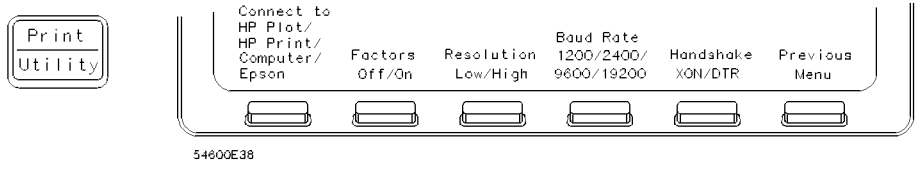


Figure 4-3. Analyzer Print/Utility menu with RS-232-C.

Interface capabilities

The baud rate, stop bits, parity, protocol, and data bits must be configured exactly the same for both the controller and the analyzer to properly communicate over the RS-232-C bus. The analyzer RS-232-C interface capabilities are listed below:

- Baud Rate: 1200, 2400, 9600, or 19.2 k
- Stop Bits: 1
- Parity: None
- Protocol: DTR or XON/XOFF
- Data Bits: 8

Protocol

DTR. With a three-wire interface, selecting Data Terminal Ready for the protocol does not allow the sending or receiving device to control data flow. No control over the data flow increases the possibility of missing data or transferring incomplete data.

With an extended hardwire interface, selecting DTR allows a hardware handshake to occur. With hardware handshake, hardware signals control data flow.

XON/XOFF. XON/XOFF stands for Transmit On/Transmit Off. With this mode the receiver (controller or analyzer) controls data flow and can request that the sender (analyzer or controller) stop data flow.

By sending XOFF (ASCII 17) over its transmit data line, the receiver requests that the sender disables data transmission. A subsequent XON (ASCII 19) allows the sending device to resume data transmission.

A controller sending data to the analyzer should send no more than 32 bytes of data after an XOFF.

The analyzer will not send any data after an XOFF is received until an XON is received.

Data bits

Data bits are the number of bits sent and received per character that represent the binary code of that character.

Information is stored in bytes (8 bits at a time) in the analyzer. Data can be sent and received just as it is stored, without the need to convert the data.

Communicating over the RS-232-C bus

Each RS-232-C interface card has its own interface select code. This code is used by the controller to direct commands and communications to the proper interface. Unlike HP-IB, which allows multiple devices to be connected through a single interface card, RS-232-C is only connected between two devices at a time through the same interface card. Because of this, only the interface code is required for the device address.

Generally, the interface select code can be any decimal value between 0 and 31, except for those interface codes which are reserved by the controller for internal peripherals and other internal interfaces. This value can be selected through switches on the interface card. For more information, refer to the reference manual for your interface card or controller.

For example, if your RS-232-C interface select code is 20, the device address required to communicate over the RS-232-C bus is 20.

Lockout command

To lockout the front panel controls, use the system command LOCK. When this function is on, all controls (except the power switch) are entirely locked out. Local control can only be restored by sending the command :SYSTEM:LOCK OFF.

NOTE

Cycling the power will also restore local control, but this will also reset certain RS-232-C states.

5

Programming and Documentation Conventions

Programming and Documentation Conventions

What you'll find in this chapter

This chapter covers conventions which are used in programming the instrument, as well as conventions used in the remainder of this manual. This chapter contains a detailed description of the command tree and command tree traversal.

Truncation rules

The truncation rule for the mnemonics used in headers and alpha arguments is:

The mnemonic is the first four characters of the keyword unless the fourth character is a vowel, in which case the mnemonic is the first three characters of the keyword.

This rule is *not* used if the length of the keyword is exactly four characters.

Some examples of how the truncation rule is applied to various commands are shown in Table 5-1.

Table 5-1. Mnemonic Truncation

Long Form	Short Form
RANGE	RANG
PATTERN	PATT
TIMEBASE	TIM
DELAY	DEL
TYPE	TYPE

Front-panel to command cross-reference

Table 5-2 lists the front-panel functions in alphabetical order with the corresponding programming commands.

The command tree

The command trees in Figure 5-1 and Figure 5-2 show all of the commands and the relationship of the commands to each other. The IEEE 488.2 common commands are not listed as part of the command tree since they do not affect the position of the parser within the tree. When a program message terminator (<NL>, linefeed – ASCII decimal 10) or a leading colon (:) is sent to the instrument, the parser is set to the "root" of the command tree.

Command types

This instrument has three types of commands:

- Common commands
- Root level commands
- Subsystem commands

Common commands

The common commands are the commands defined by IEEE 488.2. These commands control some functions that are common to all IEEE 488.2 instruments.

Common commands are independent of the tree, and do not affect the position of the parser within the tree. These commands differ from root level commands in that root level commands place the parser back at the root of the command tree.

Example:

```
*RST
```

Root level commands

The root level commands control many of the basic functions of the instrument. These commands reside at the root of the command tree. Root level commands are always parsable if they occur at the beginning of a program message, or are preceded by a colon.

Example:

```
:AUTOSCALE
```

Subsystem commands

Subsystem commands are grouped together under a common node of the command tree, such as the TIMEBASE commands. Only one subsystem may be selected at any given time. When the instrument is initially turned on, the command parser is set to the root of the command tree, therefore, no subsystem is selected.

Example:

```
:TIMEBASE
```

Truncation rules

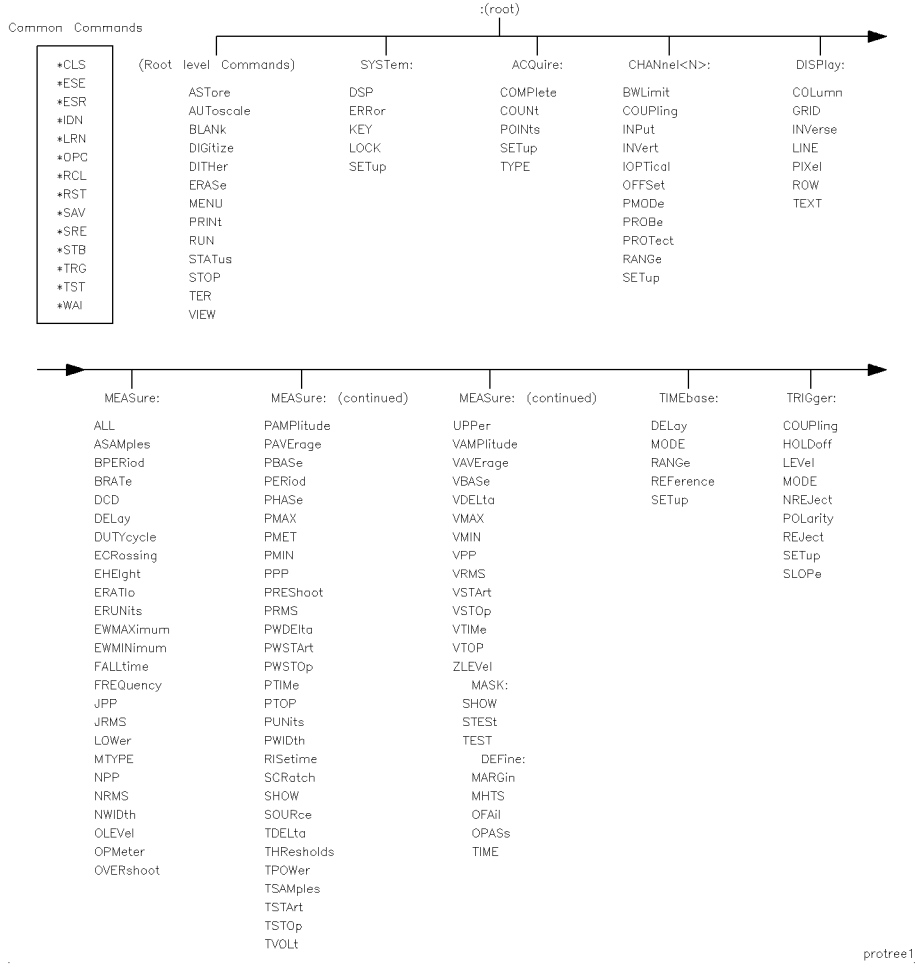


Figure 5-1. Programming command tree.

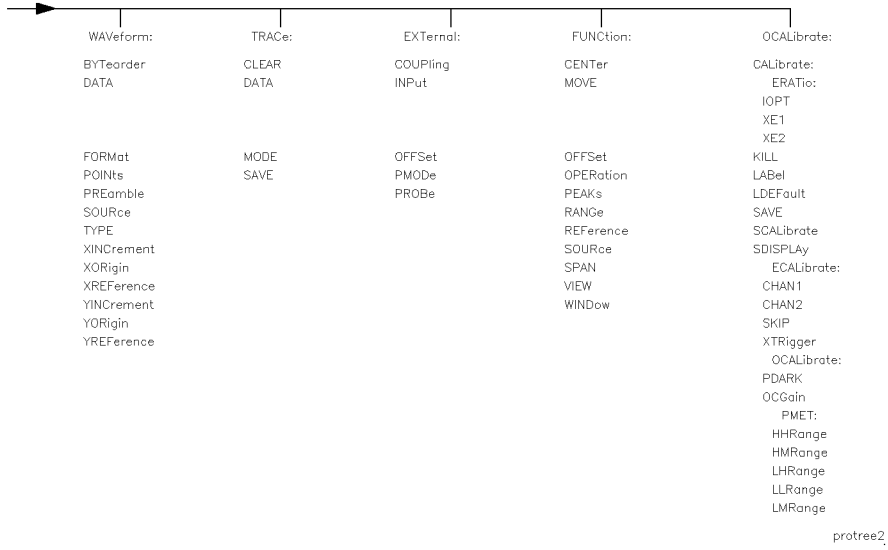


Figure 5-2. Programming command tree, continued.

Tree traversal rules

Command headers are created by traversing down the command tree. A legal command header from the command trees in Figure 5-1 would be :CHANNEL1:RANGE. This is referred to as a compound header. A compound header is a header made of two or more mnemonics separated by colons. The mnemonic created contains no spaces. The following rules apply to traversing the tree:

- A leading colon or a <program message terminator> (either an <NL> or EOI true on the last byte) places the parser at the root of the command tree. A leading colon is a colon that is the first character of a program header.

- Executing a subsystem command places you in that subsystem until a leading colon or a <program message terminator> is found. In Figure 5-1 and Figure 5-2, use the last mnemonic in the compound header as a reference point (for example, RANGE). Then find the last colon above that mnemonic (such as, CHANNEL<N>:). That is the point where the parser resides. Any command below that point can be sent within the current program message without sending the mnemonics which appear above them (for example, OFFSET).

Examples

The OUTPUT statements in the examples are written using HP BASIC 5.0 on a HP 9000 Series 200/300 Controller. The quoted string is placed on the bus, followed by a carriage return and linefeed (CRLF).

Example 1

```
OUTPUT 707;":CHANNEL1:RANGE 0.5 V;OFFSET 0 V"
```

Comments:

The colon between CHANNEL1 and RANGE is necessary because CHANNEL1:RANGE is a compound command. The semicolon between the RANGE command and the OFFSET command is the required program message unit separator. The OFFSET command does not need CHANNEL1 preceding it, since the CHANNEL1:RANGE command sets the parser to the CHANNEL1 node in the tree.

Example 2

```
OUTPUT 707;":TIMEBASE:REFERENCE CENTER;DELAY 0.00001"
```

or

```
OUTPUT 707;":TIMEBASE:REFERENCE CENTER"
OUTPUT 707;":TIMEBASE:DELAY 0.00001"
```

Comments:

In the first line of example 2, the “subsystem selector” is implied for the DELAY command in the compound command. The DELAY command must be in the same program message as the REFERENCE command, since the program message terminator places the parser back at the root of the command tree.

A second way to send these commands is by placing TIMEBASE: before the DELAY command as shown in the second part of example 2.

Example 3

```
OUTPUT 707;":TIMEBASE:REFERENCE CENTER;:CHANNEL1:OFFSET 0 V"
```

Comments:

The leading colon before CHANNEL1 tells the parser to go back to the root of the command tree. The parser can then see the CHANNEL1:OFFSET command.

Infinity representation

The representation of infinity is 9.9E+37. This is also the value returned when a measurement cannot be made.

Sequential and overlapped commands

IEEE 488.2 makes the distinction between sequential and overlapped commands. Sequential commands finish their task before the execution of the next command starts. Overlapped commands run concurrently. Commands following an overlapped command may be started before the overlapped command is completed. All of the commands are sequential.

Response generation

As defined by IEEE 488.2, query responses may be buffered for the following conditions:

- When the query is parsed by the instrument.
- When the controller addresses the instrument to talk so that it may read the response.

The responses to a query are buffered when the query is parsed.

Notation conventions and definitions

The following conventions and definitions are used in this manual in descriptions of remote operation:

Conventions

< >	Angle brackets enclose words or characters that symbolize a program code parameter or an interface command.
::=	"Is defined as," for example, <A> ::= <B> indicates that <A> can be replaced by <B> in any statement containing <A>.
|	"Or" indicates a choice of one element from a list. For example, <A> | <B> indicates <A> or <B>, but not both.
. . .	An ellipsis (trailing dots) indicates that the preceding element may be repeated one or more times.
[ ]	Square brackets indicate that the enclosed items are optional.
{ }	When several items are enclosed by braces, one, and only one, of these elements must be selected.

Definitions

d ::=	A single ASCII numeric character, 0–9.
n ::=	A single ASCII non-zero, numeric character, 1–9.
<**NL**> ::=	Newline or Linefeed (ASCII decimal 10).
<**sp**> ::=	<white space>
<**white space**> ::=	0 through 32 (decimal) except linefeed (decimal 10).

Program examples

The program examples given for each command in this manual were written on an HP 9000 Series 200/300 controller using the HP BASIC 5.0 programming language. The programs always assume the analyzer is at address 7 and the interface is at address 7 for a program address of 707. If a printer is used, it is always assumed to be at address 701.

In these examples, pay special attention to the way in which the command or query can be sent. The way the instrument is set up to respond to a command or query has no bearing on how you send the command or query. That is, the command or query can be sent using the long form or short form, if a short form exists for that command. You can send the command or query using uppercase (capital) letters or lowercase (small) letters.

If, however, you are sending an amplitude value, the value must be accompanied by the appropriate suffix. For example, when sending a Channel 2 range value, either a voltage (V, mV, uV) or wattage (W, mW, uW) suffix must be included.

As an example, set channel 2 range to 100 mV by sending one of the following:

- Commands in long form using the decimal format.

```
OUTPUT 707;":CHANNEL2:RANGE .1V"
```

- Commands in short form using an exponential format.

```
OUTPUT 707;":CHAN2:RANG 1E-1V"
```

- Commands using lowercase letters and short forms.

```
OUTPUT 707;":chan2:rang 100 mV"
```

NOTE

In these examples, the colon as the first character of the command is optional. The space between RANGE and the argument is required.

Command set organization

The command set is divided into common commands, root level commands, and several sets of subsystem commands. Each of the commands is listed alphabetically in this chapter with a brief description.

The commands are shown using upper and lowercase letters. As an example, AUToscale indicates that the entire command name is AUTOSCALE. In order to speed up the transfer, the short form AUT is also accepted by the analyzer. Each command listing contains a description of the command and its arguments, the command syntax, and a programming example.

The eight subsystems are listed below:

SYSTem	Controls some basic functions of the analyzer.
ACQuire	Sets the parameters for acquiring and storing data.
CHANnel	Controls all Y-axis analyzer functions.
DISPlay	Controls how waveforms, voltage and time markers, graticule, and text are displayed and written on the screen.
EXTernal	Defines the conditions for an external trigger.
FUNCtion	Controls the two functions, f1 and f2, available in the remote programming module.
MASK	Controls the mask and template testing.
MEASure	Selects the automatic measurements to be made.
OCALibrate	Controls optical calibration settings and functions.
TIMebase	Controls all horizontal analyzer functions.
TRACe	Controls the features used with the trace memories.
TRIGger	Controls the trigger modes and parameters for each internal trigger mode.
WAVeform	Provides access to waveform data.

Table 5-2 lists the commands in alphabetical order with their corresponding subsystem or command type.

Table 5-2. Alphabetic Command Cross-Reference

Command	Where Used
ALL	MEASure subsystem
ASAMples	MEASure subsystem
ASTore	Root level command
AUToscale	Root level command
BFIT	MEASure:MASK:DEFine subsystem
BLANk	Root level command
BPERiod	MEASure subsystem
BRATe	MEASure subsystem
BWLimit	CHANnel subsystem
BYTeorder	WAVeform subsystem
CALibrate	OCALibrate subsystem
CENTer[1]	FUNCtion subsystem
CHAN1	OCALibrate:ECALibrate subsystem
CHAN2	OCALibrate:ECALibrate subsystem
CLEAR	TRACe subsystem
*CLS	Common command
COLumn	DISPlay subsystem
COMPlete	ACQuire subsystem
COUNt	ACQuire subsystem
COUPling	CHANnel subsystem
COUPling	EXTernal subsystem
COUPling	TRIGger subsystem
DATA	MEASure:MASK:DEFine subsystem
DATA	TRACe subsystem
DATA	WAVeform subsystem
DCD	MEASure subsystem
DEFine	MEASure subsystem
DEFine	MEASure:MASK subsystem
DELay	MEASure subsystem
DELay	MEASure:DEFine subsystem
DELay	TIMebase subsystem

1 FFT function only

Table 5-2. Alphabetic Command Cross-Reference (continued)

Command	Where Used
DIGitize	Root level command
DITHer	Root level command
DSP	SYSTem subsystem
DUTYcycle	MEASure subsystem
ECALibrate	OCALibrate subsystem
ECRossing	MEASure subsystem
EHEIght	MEASure subsystem
ERASe	Root level command
ERATio	MEASure subsystem
ERATio	OCALibrate subsystem
ERRor	SYSTem subsystem
ERUNits	MEASure subsystem
*ESE	Common command
*ESR	Common command
EWMax	MEASure subsystem
EWMin	MEASure subsystem
FALLtime	MEASure subsystem
FORMat	WAVeform subsystem
FREQuency	MEASure subsystem
GRID	DISPlay subsystem
HHRange	OCALibrate:OCALibrate:PMET subsystem
HLRange	OCALibrate:OCALibrate:PMET subsystem
HMRange	OCALibrate:OCALibrate:PMET subsystem
HOLDoff	TRIGger subsystem
IDENtify	MEASure:MASK subsystem
*IDN	Common command
INPut	CHANnel subsystem
INPut	EXTernal subsystem

Table 5-2. Alphabetic Command Cross-Reference (continued)

Command	Where Used
INVerse	DISPlay subsystem
INVert	CHANnel subsystem
INVert	MEASure:MASK:DEFine subsystem
IOPT	OCALibrate:ERATio subsystem
IOPTical	CHANnel subsystem
JPP	MEASure subsystem
JRMS	MEASure subsystem
KEY	SYSTem subsystem
KILL	OCALibrate subsystem
LABel	OCALibrate subsystem
LDEFault	OCALibrate subsystem
LEVel	TRIGger subsystem
LHRange	OCALibrate:OCALibrate:PMET subsystem
LINE	DISPlay subsystem
LLRange	OCALibrate:OCALibrate:PMET subsystem
LMRange	OCALibrate:OCALibrate:PMET subsystem
LOCK	SYSTem subsystem
LOWer	MEASure subsystem
*LRN	Common command
MARGin	MEASure:MASK:DEFine subsystem
MASK	MEASure subsystem
MHTS	MEASure:MASK:DEFine subsystem
MODE	TIMebase subsystem
MODE	TRACe subsystem
MODE	TRIGger subsystem
MOVE[1]	FUNCtion subsystem
MTYPe	MEASure subsystem

1 FFT function only

Table 5-2. Alphabetic Command Cross-Reference (continued)

Command	Where Used
NPP	MEASure subsystem
NREJect	TRIGger subsystem
NRMS	MEASure subsystem
NWIDth	MEASure subsystem
OCALibrate	OCALibrate subsystem
OCGain	OCALibrate:OCALibrate subsystem
OFAil	MEASure:MASK:DEFine subsystem
OFFSet	CHANnel subsystem
OFFSet	EXTernal subsystem
OFFSet	FUNCtion subsystem
OLEVel	MEASure subsystem
OPASs	MEASure:MASK:DEFine subsystem
*OPC	Common command
OPERation	FUNCtion subsystem
OPERation	MEASure:TREF subsystem
OPMeter	MEASure subsystem
OVERshoot	MEASure subsystem
PAMPlitude	MEASure subsystem
PAVerage	MEASure subsystem
PBASe	MEASure subsystem
PDARk	OCALibrate:OCALibrate subsystem
PEAKs?[1]	FUNCtion subsystem
PERiod	MEASure subsystem
PHASe	MEASure subsystem
PIXel	DISPlay subsystem
PMAX	MEASure subsystem
PMET	MEASure subsystem
PMET	OCALibrate:OCALibrate subsystem
PMIN	MEASure subsystem

1 FFT function only

Table 5-2. Alphabetic Command Cross-Reference (continued)

Command	Where Used
POINts	ACQuire subsystem
POINts	WAVeform subsystem
POLarity	TRIGger subsystem
PPP	MEASure subsystem
PREamble	WAVeform subsystem
PRESHoot	MEASure subsystem
PRINt	Root level command
PRMS	MEASure subsystem
PROBe	CHANnel subsystem
PROBe	EXTernal subsystem
PROTect	CHANnel subsystem
PROTect	EXTernal subsystem
PTIMe	MEASure subsystem
PTOP	MEASure subsystem
PUNIts	MEASure subsystem
PWDELta	MEASure subsystem
PWIDth	MEASure subsystem
PWSTArt	MEASure subsystem
PWSTop	MEASure subsystem
RANGe	CHANnel subsystem
RANGe	FUNCtion subsystem
RANGe	TIMebase subsystem
*RCL	Common command
REFerence[1]	FUNCtion subsystem
REFerence	TIMebase subsystem
REJect	TRIGger subsystem
RETest	MEASure:MASK:DEFine subsystem
RISetime	MEASure subsystem

1 FFT function only

Table 5-2. Alphabetic Command Cross-Reference (continued)

Command	Where Used
ROW	DISPlay subsystem
*RST	Common command
RUN	Root level command
*SAV	Common command
SAVE	OCALibrate subsystem
SAVE	TRACe subsystem
SCALibrate	OCALibrate subsystem
SCRatch	MEASure subsystem
SDISplay	OCALibrate subsystem
SETup	ACQuire subsystem
SETup	CHANnel subsystem
SETup	DISPlay subsystem
SETup	EXTernal subsystem
SETup	SYSTem subsystem
SETup	TIMebase subsystem
SETup	TRIGger subsystem
SHOW	MEASure subsystem
SHOW	MEASure:MASK subsystem
SKEW	CHANnel subsystem
SKEW	EXTernal subsystem
SKIP	OCALibrate:ECALibrate subsystem
SLOPe	TRIGger subsystem
SOURce	FUNCtion subsystem
SOURce	MEASure subsystem
SOURce	TRIGger subsystem
SOURce	WAVeform subsystem
SPAN[1]	FUNCtion subsystem
*SRE	Common command
STATus	Root level command
*STB	Common command
STESt	MEASure:MASK subsystem
STOP	Root level command

1 FFT function only

Table 5-2. Alphabetic Command Cross-Reference (continued)

Command	Where Used
T1	MEASure:TREF subsystem
T2	MEASure:TREF subsystem
TDELta	MEASure subsystem
TER	Root level command
TEST	MEASure:MASK subsystem
TEXT	DISPlay subsystem
THResholds	MEASure subsystem
TIME	MEASure:MASK:DEFine subsystem
TPOWer	MEASure subsystem
TREF	MEASure subsystem
*TRG	Common command
TSAMples	MEASure subsystem
*TST	Common command
TSTArt	MEASure subsystem
TSTOp	MEASure subsystem
TVHFrej	TRIGger subsystem
TVMode	TRIGger subsystem
TVOLt	MEASure subsystem
TYPE	ACQuire subsystem
TYPE	WAVeform subsystem
UPPer	MEASure subsystem
VAMPlitude	MEASure subsystem
VAVerage	MEASure subsystem
VBASe	MEASure subsystem
VDELta	MEASure subsystem
VERNier	CHANnel subsystem
VERNier	TIMebase subsystem
VIEW	FUNCtion subsystem
VIEW	Root level command

Table 5-2. Alphabetic Command Cross-Reference (continued)

Command	Where Used
VMAX	MEASure subsystem
VMIN	MEASure subsystem
VPP	MEASure subsystem
VRMS	MEASure subsystem
VSTArt	MEASure subsystem
VSTOp	MEASure subsystem
VTIMe	MEASure subsystem
VTOP	MEASure subsystem
*WAI	Common command
WINDow[1]	FUNCtion subsystem
XE1	OCALibrate:ERATio subsystem
XE2	OCALibrate:ERATio subsystem
XEATtenuation	CHANnel subsystem
XEAUnits	CHANnel subsystem
XINCrement	WAVeform subsystem
XOATtn	CHANnel subsystem
XORigin	WAVeform subsystem
XREFerence	WAVeform subsystem
XTRigger	OCALibrate:ECALibrate subsystem
YINCrement	WAVeform subsystem
YORigin	WAVeform subsystem
YREFerence	WAVeform subsystem
ZLEVel	MEASure subsystem

1 FFT function only

6

Command Types

Command Types

What you'll find in this chapter

This chapter lists the types of commands available for the analyzer. For information on individual commands, refer to Chapter 7.

The types of commands are listed in this chapter in the following order:

Common commands
Root level commands
System commands

Acquire commands
Channel commands
Display commands
External (trigger) commands
Function commands
Mask commands
Measure commands
Ocalibrate commands
Timebase commands
Trace commands
Trigger (internal) commands
Waveform commands

Common Commands

The common commands are defined by the IEEE 488.2 standard. These commands are common to all instruments that comply with the IEEE 488.2 standard. They control some of the basic instrument functions, such as instrument identification and reset, reading the learn (instrument setup) string, how status is read and cleared, and how commands and queries are received and processed by the instrument.

The following common commands are implemented in the analyzer:

*CLS (Clear Status)
*ESE (Event Status Enable)
*ESR (Event Status Register)
*IDN (Identification Number)
*LRN (Learn)
*OPC (Operation Complete)
*RCL (Recall)
*RST (Reset)
*SAV (Save)
*SRE (Service Request Enable)
*STB (Status Byte)
*TRG (Trigger)
*TST (Test)
*WAI (Wait)

Common commands can be received and processed by the analyzer whether they are sent over the interface as separate program messages or within other program messages. If an instrument subsystem has been selected and a common command is received by the instrument, the instrument remains in the selected subsystem. For example, if the program message ":ACQUIRE:TYPE AVERAGE; *CLS; COUNT 256" is received by the instrument, the instrument sets the acquire type and count, then clears the status information without leaving the selected subsystem.

If some other type of command is received within the program message, you must reenter the original subsystem after the command. For example, the program message ":ACQUIRE:TYPE AVERAGE;:AUTOSCALE;:ACQUIRE:COUNT 256" sets the acquire type, completes the autoscale, then sets the acquire count. In this example, :ACQUIRE must be sent again after the AUTOSCALE command in order to reenter the Acquire subsystem and set the count.

NOTE

Each of the status registers mentioned in this chapter has an enable (mask) register. By setting the bits in the enable register, you can select the status information you wish to use.

Root Level Commands

Root Level commands control many of the basic operations of the analyzer. These commands are always recognized by the parser if they are prefixed with a colon, regardless of current command tree position. After executing a root level command, the parser is positioned at the root of the command tree.

The following Root Level commands are implemented:

ASTore
AUToscale
BLANk
DIGitize
DITHer
ERASe
PRINt
RUN
STATus
STOP
TER
VIEW

SYSTEM subsystem commands control the way in which query responses are formatted, simulate front panel key presses, and enable reading and writing to the advisory line of the instrument.

The System subsystem contains the following commands:

DSP
ERRor
KEY
LOCK
SETup

Acquire Commands

The ACQUIRE subsystem commands set up conditions for executing a DIGITIZE root level command to acquire waveform data.

The Acquire subsystem selects the type of data, the number of averages, the number of data points, and the completion criteria.

This subsystem contains the following commands:

COMPlete
COUNt
POINts
SETup
TYPE

Acquire type and count

The ACQUIRE subsystem is the only interface control for two display parameters: Display Mode and Number of Averages. There is a coupling between the front panel and the ACQUIRE subsystem parameters. This means that when the interface parameters for ACQUIRE TYPE or COUNT are changed, the front panel changes. Also, when the front-panel parameters are changed, the interface parameters change.

Normal mode

The :ACQUIRE:TYPE NORMAL command sets the analyzer to the normal mode when the instrument is operating in the repetitive timebase mode.

Averaging mode

The :ACQUIRE:TYPE AVERAGE command sets the analyzer to the Averaging mode.

COUNT can be set in the AVERAGE mode by sending the :ACQUIRE:COUNT command followed by the number of averages. In this mode, the value is one of three values: 8, 64, or 256. The COUNT value determines the number of averages that must be acquired.

To activate the averaging mode from the front panel, select the Display menu, then select Average. Changing the number of averages changes the COUNT value.

Peak Detect mode

The :ACQUIRE:TYPE PEAK command sets the analyzer to the Peak Detect mode. The :ACQUIRE:COUNT has no meaning in PEAK mode.

Channel Commands

The CHANNEL subsystem commands control the channel display and vertical or Y-axis of the analyzer. Channels are independently programmable for all offset, probe, coupling, and range functions. The channel number specified in the command selects the channel that is affected by the command.

The channel displays are toggled on and off with the root level commands VIEW and BLANK.

The Channel subsystem contains the following commands:

BWLimit
COUPling
INPut
INVert
IOPTical
OFFSet
PROBe
PROTect
RANGe
SETup
SKEW
VERNier
XEATtenuation
XEUnits
XOATtn

NOTE

When an amplitude or vertical scale parameter is set, the appropriate suffix, either volts or watts, must be included. Valid suffixes for volts are V, mV and uV. Valid suffixes for watts are W, mW and uW.

Display Commands

The DISPLAY subsystem is used to control the display of data, amplitude and time markers, text, and graticules.

NOTE

The Display mode can be selected with the :ACQUIRE:TYPE command. The number of averages can be specified with the :ACQUIRE:COUNT command.

The Display subsystem contains the following commands:

COLumn
GRID
INVerse
LINE
PIXel
ROW
SETup
TEXT

External Commands (External Trigger)

The commands in the EXTERNAL subsystem define the conditions for an external trigger. If the command is a valid command for the external trigger, that setting is accepted. If the command is not valid for the external trigger, an error is generated.

The External Trigger subsystem contains the following commands:

COUPling
INPut
OFFSet
PROBe
PROTect
SETup
SKEW

Function Commands

The FUNCTION subsystem commands control the two functions available in the remote programming module. The function number specified in the command selects the function that is affected by the command. The commands control the scaling, source, and state of the functions.

The Function subsystem contains the following commands:

CENTer	(FFT function only)
MOVE	(FFT function only)
OFFSet	
OPERation	
PEAKs	(FFT function only)
RANGe	
REFerence	(FFT function only)
SOURce	
SPAN	(FFT function only)
VIEW	
WINDow	(FFT function only)

NOTE

The advanced math functions are *not* available when the optical channel is active or when the measurement type is set to Eye:NRZ.

Mask Commands

The commands in the MEASURE:MASK subsystem are used to make mask and template measurements, and to set and report settings for mask testing. Some commands in this subsystem can also be used to adjust the Conformance test criteria.

The MEASURE:MASK subsystem includes the following commands:

DEFINE
DELay
IDENtify
MARGin
MHTS
OFAil
OPASs
SHOW
STESt
TEST
TIME

Mask setup

To make a mask measurement, the portion of the waveform to be mask tested must be placed on the screen. Refer to Chapter 7 for more information.

Mask measurements can only be performed on one source at a time. The source can be set to electrical channel 1, 2, or optical input channel 2.

Mask scaling and placement

The mask is scaled and placed based on critical positions on the selected waveform. The critical positions on the waveform are determined using the same algorithms that are used to make the parametric measurements.

> **NOTE**
>
> Turn the VERNIER timebase setting OFF for the best throughput during mask scaling and placement.

Eye:NRZ

If a multi-valued, NRZ, waveform mask is chosen, then the critical positions are determined by using the same algorithms as: MEAS:OLEVel MEAS:ZLEVel, and MEAS:ECRossing.

Single valued

If a single valued waveform mask or template is selected, then the critical positions on the waveform are determined with the equivalent algorithms as: MEAS:VTOP, MEAS:VBASe, and the 50% position of the rising or falling edge.

> **NOTE**
>
> When an eye diagram mask is selected, the measurment type is automatically set to Eye:NRZ. If a pulse mask is selected, the measurement type is automatically set to General Waveform. The automatic measurement algorithm is affected by the measurement type. For more information, refer to "Measure Commands".

Mask testing modes

The mask testing can be done in either of two modes, conformance test or run mode.

Conformance test mode

In Conformance Test mode, all sources but the selected SOURce are blanked and the selected source is displayed on the screen. The mask is scaled and a test is performed. The results are determined based on the specified pass/fail conditions.

Run mode

In Run mode, the mask placement positions will be taken from the selected source, but hits will be determined based on all full bright incursions into the mask exclusion regions. For best results in this mode, it is suggested you only display the source on which you wish to perform mask measurements.

NOTE

Mask testing and placement is not allowed when advanced math functions, such as FFT, integration, and differentiation, are active.

Mask margins

The mask margin is a value from −100% to +100% that is used to relax (negative mask margin) or tighten (positive mask margin) the mask testing specification.

When a margin other then 0% is used, the number of hits in the margin, and the standard mask are counted, and reported separately.

The total number of hits (MEASure:MASK:TOTHits?) is the value used to determine if the conformance test is passed or failed. The total number of hits is determined differently, depending on whether there is a negative or positive margin applied.

If the margin is negative, the total number of hits reported includes only the hits in the internal section of the mask which does not include the negative margin. By definition, this section is inside the standard mask.

If the margin is positive, the total number of hits reported is the number of hits in the standard mask and the number of hits in the margin.

Measure Commands

The commands in the MEASURE subsystem are used to make parametric measurements on displayed waveforms and to report the settings of the voltage and time markers. Some commands in this subsystem can be used to set the voltage and time markers to specified voltages, times, or events.

Two types of measurements can be made, Eye:NRZ for eye diagram waveforms or General Waveforms for general repetitive waveforms.

Amplitude measurements can be made in either power or voltage units. To make optical power measurements on channel 2, the optical channel must first be activated by pressing (Optical/Elec 2), `Chan 2 Off Op E2` Op or by sending the CHANnel2:IOPTtical ON command. If a power measurement query is sent, for example PAMPlitude, and the optical channel is off, an error value of 9.9e37 will be returned.

The Measure subsystem contains the following commands:

ALL	LOWer	PREShoot	TSTArt
ASAMples	MASK	PRMS	TSTOp
BPERiod	MTYPe	PTIMe	TVOLt
BRATe	NPP	PTOP	UPPer
DCD	NRMS	PUNIts	VAMPlitude
DEFine	NWIDth	PWDELta	VAVerage
DELay	OLEVel	PWIDth	VBASe
DUTYcycle	OPMeter	PWSTArt	VDELta
ECRossing	OVERshoot	PWSTop	VMAX
EHEIght	PAMPlitude	RISetime	VMIN
ERATio	PAVerage	SCRatch	VPP
ERUNits	PBASe	SHOW	VRMS
EWMax	PERiod	SOURce	VSTArt
EWMin	PHASe	TDELta	VSTOp
FALLtime	PMAX	THResholds	VTIMe
FREQuency	PMET	TPOWer	VTOP
JPP	PMIN	TREF	ZLEVel
JRMS	PPP	TSAMples	

The Measure commands are available in Eye:NRZ and General Waveform modes.

N O T E

The measurement type is automatically set to Eye:NRZ when an eye diagram mask is selected. It is set to General Waveform when a pulse mask is selected.

Measurement	General Waveform	Eye:NRZ
Time Measurements		
Bit Period	No	Yes
Bit Rate	No	Yes
DC Distortion	No	Yes
Delay	Yes	No
Duty Cycle	Yes	No
Fall Time	Yes	Yes
Frequency	Yes	No
Jitter-pp	No	Yes
Jitter-rms	No	Yes
Minus Width	Yes	No
Period	Yes	No
Phase	Yes	No
Plus Width	Yes	No
Rise Time	Yes	Yes
Tdelta	Yes	Yes
Tpower	Yes	No
Tstart	Yes	Yes
Tstop	Yes	Yes
Tvolt	Yes	No

Measurement	General Waveform	Eye:NRZ
Power and Voltage Measurements		
Extinction Ratio	No	Yes
Eye Crossing	No	Yes
Eye Height	No	Yes
Noise-pp	No	Yes
Noise-rms	No	Yes
One Level	No	Yes
Overshoot	Yes	Yes
Pamp	Yes	No
Pavg	Yes	No
Pbase	Yes	No
Pmax	Yes	Yes
Pmet	Yes	Yes
Pmin	Yes	Yes
Ppp	Yes	No
Preshoot	Yes	No
Prms	Yes	No
Ptime	Yes	Yes
Ptop	Yes	No
Pwdelta	Yes	Yes
Pwstart	Yes	Yes
Pwstop	Yes	Yes
Top level	Yes	No
Vamp	Yes	No
Vavg	Yes	No
Vbase	Yes	No
Vdelta	Yes	Yes
Vmax	Yes	Yes
Vmin	Yes	Yes
Vpp	Yes	No
Vrms	Yes	No
Vstart	Yes	Yes
Vstop	Yes	Yes
Vtime	Yes	Yes
Vtop	Yes	No
Zero Level	No	Yes

Measurement setup

To make a measurement, the portion of the waveform required for that measurement must be displayed on the analyzer:

- For a period or frequency measurement, at least one complete cycle must be displayed.
- For a pulse width measurement, the entire pulse must be displayed.
- For a rise time measurement, the leading (positive-going) edge of the waveform must be displayed.
- For a fall time measurement, the trailing (negative-going) edge of the waveform must be displayed.
- For an extinction ratio measurement, an entire eye must be displayed.

Measurement error

If a measurement cannot be made (typically because the proper portion of the waveform is not displayed), the value +9.9E+37 is returned for that parameter.

Making measurements

If more than one waveform, edge, or pulse is displayed, time measurements are made on the first (left-most) portion of the displayed waveform.

When any of the defined measurements are requested, the analyzer determines the top/one level (100%) and base/zero level (0%) voltages or powers of the waveform. Using this information, it can determine the other important voltage or power values (10%, 90%, and 50% voltage or power values) for making measurements.

The 10% and 90% amplitude values are typically used in the rise time and fall time measurements. The 50% voltage value is used for measuring frequency, period, pulse width, and duty cycle. The critical parameters on eye diagram measurements are the "1" level, "0" level, and crossing point values. These values are used when making bit rate, eye crossing percent and jitter measurements.

When the command form, or front-panel measurements is used, the instrument is placed into the continuous measurement mode. When the query form of these measurements is used, the measurement is made but not selected. Then the measurement is made one time, and the measurement result is returned.

Except for Vaverage and Vrms or Paverage and Prms measurements, which use only one cycle, general waveform voltage or power measurements are made using the entire display. If you want to make a measurement on a particular cycle, display only that cycle on the screen.

All eye diagram amplitude and time measurements are made on the first, left-most, eye diagram.

Amplitude measurement values are returned in volts, μW, or dBm, depending on the currently selected measurement units and source. Voltage values are measured using zero volts as the reference. Power values are measured using zero light as the reference.

Returned time values are in seconds. Returned time values are measured with the trigger point (time 0) as the reference. The value returned for TDELTA? is the time difference between the stop and start markers.

Measurements are made on the displayed waveforms specified by the SOURCE command.

For more information about measurement algorithms, refer to Appendix A of the *HP 83475B Lightwave Communications Analyzer User's Guide*.

Ocalibrate Commands

The OCALIBRATE subsystem commands control the optical channel vertical scale and power meter calibration functions and settings. Commands are also provided to perform and save extinction ratio offset calibrations on the optical and electrical channels.

The Ocalibrate subsystem contains the following commands:

CALibrate
ECALibrate
ERATio
KILL
LABel
LDEFault
OCALibrate
SAVE
SCALibrate
SDISplay

NOTE

Refer to Appendix A for more information on the execution of a remote optical channel calibration.

Timebase Commands

The TIMEBASE subsystem commands control the horizontal or X-axis analyzer functions.

The Timebase subsystem contains the following commands:

DELay
MODE
RANGe
REFerence
SETup
VERNier

Trace Commands

The TRACE subsystem commands are used to control the features used with trace memories. Trace memories 1-4 are stored as non-compressed images, while memories 5-100 are stored in a compressed format. In addition to the waveform, the current front-panel setup is saved.

The Trace subsystem contains the following commands:

CLEAR
DATA
MODE
SAVE

Trigger Commands (Internal Trigger)

The commands in the TRIGGER subsystem define the conditions for an internal trigger. Many of the commands in the Trigger subsystem are valid in more than one of the trigger modes. If the command is a valid command for a trigger mode, that setting is accepted. If the command is not valid for a trigger mode, an error is generated.

The Internal Trigger subsystem contains the following commands:

COUPling
HOLDoff
LEVel
MODE
NREJect
POLarity
REJect
SETup
SLOPe
SOURce
TVHFrej
TVMode

Waveform Commands

The WAVEFORM subsystem is used to transfer waveform data between a controller and the analyzer waveform memories. This subsystem contains the following commands:

BYTeorder
DATA
FORMat
POINts
PREamble
SOURce
TYPE
XINCrement
XORigin
XREFerence
YINCrement
YORigin
YREFerence

Waveform data and preamble

The waveform record is actually contained in two portions: the waveform data and the preamble. The waveform data is the actual data acquired for each point in the specified source. The preamble contains the information for interpreting the waveform data, which includes the number of points acquired, format of acquired data, and type of acquired data. The preamble also contains the X and Y increments, origins, and references for the acquired data, so that the raw data can be translated to time and voltage values.

The values set in the preamble are determined when the :DIGITIZE command is executed. The preamble values are based on the settings of variables in the ACQUIRE subsystem.

Although the preamble values can be changed with a controller, the way the data is acquired cannot be changed. Changing the preamble values cannot

change the type of data that was actually acquired, the number of points actually acquired, etc. Therefore, you must use extreme caution when changing any waveform preamble values to ensure the data is still useful.

The waveform data and preamble must be read by the controller or sent to the analyzer with two separate commands, DATA and PREAMBLE.

Sending consecutive :DIGITIZE commands may improve the data throughput. Refer to the Root Level Command :DIGITIZE in Chapter 7 for more information.

Data acquisition types

There are three types of waveform acquisition that can be selected with the :ACQUIRE:TYPE command: NORMAL, AVERAGE, PEAK. When the data is acquired using the DIGITIZE or RUN command, the data is placed in the channel buffer of the specified source.

After a DIGITIZE command, the instrument is stopped. If the instrument is restarted, over the HP-IB or the front panel, the data acquired with the DIGITIZE command is overwritten.

Depending on the analyzer configuration, a waveform record consists of either 4000 or 2000 acquired points. Most configurations acquire 4000 points per channel. The number of hardware points acquired may be queried using :ACQUIRE:POINTS?.

Fewer points may be used to speed data transfers and to minimize controller analysis time. The :WAVEFORM:POINTS may be varied even after data on a channel is acquired.

The number of points transferred to the computer is controlled using the :WAVEFORM:POINTS command. The number of points selected for transfer using :WAVEFORM:POINTS must be an even divisor of the number of points available as read using :ACQUIRE:POINTS?.

A waveform record consists of 4000 acquired points. :WAVEFORM:POINTS determines the number of points that will be transferred for :WAVEFORM:DATA? query. :WAVEFORM:POINTS determines the increment between timebuckets that will be transferred. :ACQUIRE:POINTS? must be an interger multiple of :WAVEFORM:POINTS. For example:

```
:WAVEFORM:POINTS 4000 returns time buckets 0, 1, 2, 3, 4, . . .3999.
:WAVEFORM:POINTS 2000 returns time buckets 0, 2, 4, 6, 8, . . .3998.
:WAVEFORM:POINTS 1000 returns time buckets 0, 4, 8, 12, 16, . .3992.
:WAVEFORM:POINTS 500 returns time buckets 0, 8, 16, 32, 64, . .3960.
```

Normal data

Normal data consists of the last data point (hit) in each time bucket. This data is transmitted over HP-IB in a linear fashion starting with time bucket 0 and going through time bucket n–1, where n is the number returned by the :WAVEFORM:POINTS? query. Only the magnitude values of each data point are transmitted. The first voltage value corresponds to the first time bucket on the left of the screen and the last value corresponds to the next to last time bucket on the right side of the screen. Time buckets that do not have data in them return 0.

The time values for each data point correspond to the position of the data point in the data array. These values are not transmitted.

Average data

Average data consists of the average of the first n hits in a time bucket, where n is the value returned by the :ACQUIRE:COUNT? query. Time buckets that have fewer than n hits return the average of what data they do have. If a time bucket does not have any data in it, it returns 0.

This data is transmitted over the interface linearly, starting with time bucket 0 and proceeding through time bucket n–1, where n is the number returned by the :WAVEFORM:POINTS? query. The first value corresponds to a point at the left side of the screen and the last value corresponds to one point away from the right side of the screen.

Peak detect

Peak detect display mode is used to detect glitches for timebase settings of 50 μs/div and slower. In this mode, the analyzer can sample more data than it can store and display. So, when peak detect is turned on, the analyzer scans through the extra data, picks up the minimum and maximum and then stores the data in an array. Each time bucket contains two data.

The array is transmitted over the interface bus linearly, starting with time bucket 0 and proceeds through time bucket (n/2)–1, where n is the number returned by the :WAVEFORM:POINTS? query. In each time bucket, two data are transmitted, the maximum followed by the minimum value. The first pair of values corresponds to the time bucket at the left-most side of the screen. The last pair of values corresponds to the time bucket at the far right of the screen.

Data conversion

Data sent from the analyzer must be scaled for useful interpretation. The values used to interpret the data are the X and Y references, X and Y origins, and X and Y increments. These values are read from the waveform preamble. Each channel has its own waveform preamble.

Conversion from data value to voltage or watts

The following formula converts a data value from a channel to a value in volts or watts:

voltage/watt = [(data value – yreference) ∗ yincrement] + yorigin

Watts data can only be obtained from the optical channel. To acquire watts data from channel 2, the command :Chan2:IOPT ON must have been sent. To obtain valid data, the correct optical wavelength calibration must be selected.

Conversion from data value to time

The time value of a data point can be determined by the position of the data point. As an example, the fourth data point sent with XORIGIN = 16 ns, XREFERENCE = 0, and XINCREMENT = 2 ns can be calculated using the following formula:

time = [(data point number – xreference) ∗ xincrement] + xorigin

This would result in the following calculation for time bucket 3:

time = [(3 – 0) ∗ 2 ns] + 16 ns = 22 ns

Data format for HP-IB transfer

There are three formats for transferring waveform data over the interface: BYTE, WORD, and ASCII.

BYTE and WORD formatted waveform records are transmitted using the arbitrary block program data format specified in IEEE 488.2. ASCII format block data does not use a block header.

When you use the block data format, the ASCII character string "#8&<DD...D>" is sent prior to sending the actual data. The 8 indicates how many D's follow. The D's are ASCII numbers which indicate how many data bytes follow.

For example, if 4000 points will be transferred and the WORD format was specified, the block header "#800008000" would be sent. The 8 indicates that eight length bytes follow, and 8000 indicates that 8000 binary data bytes follow.

WORD format

In the WORD format, the number of data bytes is twice the number of words (data points). The number of data points is the value returned by the :WAVEFORM:POINTS? query. The number of data bytes is followed by a sequence of bytes representing the data points, with the most significant byte of each word transmitted first. In this format the data is shifted so that the most significant bit after the sign bit contains the most significant bit of the data. If there is a hole in the data, the hole is represented by the 16-bit value of 0. The range of data in the WORD format is from 1 to 255. WORD format is useful in applications where the information is read directly into an integer array in a controller.

Use :WAVEFORM:BYTEORDER to determine if the least significant byte or most significant byte is to be transferred first. The most significant byte is always 0. The BYTEORDER command can be used to alter the transmit sequence to match the storage sequence of an integer in the programming language being used.

BYTE format

The BYTE format allows direct access to the 8-bit waveform data. If there is a hole in the data, the hole is represented by a value of 0.

The BYTE-formatted data transfers over the HP-IB faster than WORD-formatted data, since one byte per point is transferred in BYTE format and two bytes per point are transferred in WORD format.

ASCII format

Waveform records in the ASCII format are transmitted one value at a time, separated by a comma. The data values transmitted are the same as the values sent in the WORD format except that they are converted to an integer ASCII format (six or less characters) before being sent through the interface.

7

Command Dictionary

Command Dictionary

What you'll find in this chapter

This chapter is a dictionary reference of the remote programming commands. All commands are listed alphabetically by subsystem.

Common commands

*CLS
(Clear Status)

The *CLS (clear status) common command clears the status data structures, including the device-defined error queue. This command also clears the Request-for-OPC flag.

If the *CLS command immediately follows a program message terminator, the output queue and the MAV (message available) bit are cleared.

Command Syntax:	*CLS
Example:	OUTPUT 707;""CLS"

*ESE (Event Status Enable)

The *ESE command sets the bits in the Standard Event Status Enable Register. The Standard Event Status Enable Register contains a mask value for the bits to be enabled in the Standard Event Status Register. A one in the Standard Event Status Enable Register enables the corresponding bit in the Standard Event Status Register. A zero disables the bit. Refer to Table 7-1 for information about the Standard Event Status Enable Register bits, bit weights, and what each bit masks.

The *ESE query returns the current contents of the register.

Command Syntax:	*ESE <mask>
Where:	<mask> ::= 0 to 255
Example:[1]	OUTPUT 707;"*ESE 64"
Query Syntax:	*ESE?
Returned Format:	<mask><NL>
Where:	<mask> ::= 0 to 255 [integer-NR1 format]
Example:	OUTPUT 707;"*ESE?" ENTER 707;Event PRINT Event

1 In this example, the *ESE 64 command enables URQ (user request) bit 6 of the Standard Event Status Enable Register. Therefore, when a front-panel key is pressed, the ESB (event summary bit) in the Status Byte Register is also set.

Table 7-1. Standard Event Status Enable Register

Enables Standard Event Status Enable Register (High–Enables the ERS bit)		
Bit	**Weight**	**Enables**
7	128	NOT USED
6	64	URQ - User Request
5	32	CME - Command Error
4	16	EXE - Execution Error
3	8	DDE - Device Dependent Error
2	4	QYE - Query Error
1	2	TRG - Trigger Query
0	1	OPC - Operation Complete

*ESR
(Event Status Register)

The *ESR query returns the contents of the Standard Event Status Register.

When you read the Event Status Register, the value returned is the total bit weights of all of the bits that are high at the time you read the byte. Table 7-2 shows each bit in the Event Status Register and its bit weight.

Reading the register clears the Event Status Register.

Query Syntax:	*ESR?
Returned Format:	<status><NL>
Where:	<status> ::= 0 to 255 |integer-NR1 format|
Example:	OUTPUT 707;"*ESR?" ENTER 707;Event PRINT Event

Table 7-2. Standard Event Status Register

Bit	Bit Weight	Bit Name	Condition
7	128	—	NOT USED
6	64	URQ	0 = no front-panel key has been pressed. 1 = a front-panel key has been pressed.
5	32	CME	0 = no command errors. 1 = a command error has been detected.
4	16	EXE	0 = no execution error. 1 = an execution error has been detected.
3	8	DDE	0 = no device dependent errors. 1 = a device dependent error has been detected.
2	4	QYE	0 = no query errors. 1 = a query error has been detected.
1	2	TRG	0 = No trigger or AUTO |autolevel|. 1 = Trigger |can be 1 only in Normal, Single, TV trigger modes|.
0	1	OPC	0 = operation is not complete. 1 = operation is complete.
0 = False = Low 1 = True = High			

*IDN (Identification Number)

The *IDN query identifies the instrument type and software version by returning the following string:

"HEWLETT-PACKARD,83475B,0,<X.X>"

Where:

<X.X> ::= the software revision of the instrument.

An *IDN query must be the last query in a message. Any queries after the *IDN query in a program message are ignored.

Query Syntax:	*IDN?
Returned Format:	HEWLETT-PACKARD,83475B,0,X.X<NL>
Example:	DIM Id$[50] OUTPUT 707;"*IDN?" ENTER 707;Id$ PRINT Id$

*LRN (Learn)

The *LRN query returns a program message that contains the current state of the instrument.

This query performs the same function as the :SYSTEM:SETUP? query. It allows you to store an instrument setup in the controller. The stored setup can then be returned to the instrument at a later time using the :SYSTEM:SETUP command.

Query Syntax:	*LRN?
Returned Format:	#80000121<learn string><NL>
Where:	<learn string> :: = 121 data bytes in length.
Example:	DIM Lrn$[200] OUTPUT 707;"*LRN?" ENTER 707 USING "-K";Lrn$

*OPC (Operation Complete)

The *OPC (operation complete) command sets the operation complete bit in the Standard Event Status Register when all pending device operations have finished.

The *OPC query places an ASCII "1" in the output queue when all pending device operations have finished.

Command Syntax:	*OPC
Example:	OUTPUT 707;""*OPC"
Query Syntax:	*OPC?
Returned Format:	1<NL>
Example:	OUTPUT 707;":AUTOSCALE;*OPC?" ENTER 707;Op$

*RCL (Recall)

The *RCL command restores the state of the instrument from the specified save/recall register. An instrument setup must have been stored previously in the specified register.

Command Syntax:	*RCL { 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16}
Example:	OUTPUT 707;""RCL 3"

*RST
(Reset)

The *RST command places the instrument in a known state. Refer to Table 7-3 for the reset conditions.

Command Syntax:	*RST
Example:	OUTPUT 707;"*RST"

Table 7-3. Analyzer Reset Conditions

Timebase Menu		**Display Menu**	
time/division	100 μs	mode	norm
delay	0.00 s	grid	on
reference	cntr		
mode	main	**Cursors/Masks Menu**	
		ΔV markers	off
Channel Menu		Δt markers	off
Channel 1	on		
Channel 2	off	**Trace Menu**	
volts/division	100 mV	Trace	Mem 1
offset	0.00	Memory	off
coupling	dc		
probe attenuation	X1	**Measure Menu**	
vernier	off	Source	Channel 1
invert	off		
BW limit	off		
Trigger Menu			
Mode	auto level		
coupling	dc		
source	Channel 1		
level	0.0 V		
slope	positive		
reject and noise reject	off		
holdoff	200 ns		

*SAV
(Save)

The *SAV command stores the current state of the device in a save register. The data parameter is the number of the save register where the data will be saved. Registers 1 through 16 are valid for this command.

Command Syntax:	*SAV {1 |2 |3 |4 |5 |6 |7 |8 |9 |10 |11 |12 |13 |14 |15 |16 }
Example:	OUTPUT 707;""*SAV 3"

*SRE
(Service Request Enable)

The *SRE command sets the bits in the Service Request Enable Register. The Service Request Enable Register contains a mask value for the bits to be enabled in the Status Byte Register. A "1" in the Service Request Enable Register enables the corresponding bit in the Status Byte Register. A "0" disables the bit. Table 7-4 lists the bits in the Service Request Enable Register and what they mask.

The *SRE query returns the current value.

Command Syntax:	*SRE <mask>
Where:	<mask> ::= 0 to 255
Example:	OUTPUT 707;""*SRE 32"

Query Syntax:	*SRE?
Returned Format:	<mask><NL>
Where:	<mask> ::= sum of all bits that are set, 0 through 255 |integer-NR1 format|
Example:	OUTPUT 707;""*SRE?" ENTER 707;Value PRINT Value

Table 7-4. Service Request Enable Register

Service Request Enable Register (High - Enables the SRE bit)		
Bit	**Weight**	**Enables**
7	128	Not Used
6	64	RQS - Request Service
5	32	ESB - Event Status Bit
4	16	MAV - Message Available
3	8	Not Used
2	4	Not Used
1	2	Not Used
0	1	Not Used

*STB (Status Byte)

The *STB query returns the current value of the instrument's status byte. The MSS (Master Summary Status) bit is reported on bit 6 instead of the RQS (request service) bit. The MSS indicates whether or not the device has at least one reason for requesting service. Refer to Table 7-5 for the meaning of the bits in the status byte.

To read the instrument's status byte with RQS reported on bit 6, use the interface Serial Poll.

Query Syntax:	*STB?
Returned Format:	<value><NL>
Where:	<value> ::= 0 through 255 |integer-NR1|
Example:	OUTPUT 707;"*STB?" ENTER 707;Value PRINT Value

Table 7-5. Status Byte Register

Bit	Bit/Weight	Bit/Name	Condition
7	128	—	0 = not used.
6	64	RQS/MSS	0 = instrument has no reason for service. 1 = instrument is requesting service.
5	32	ESB	0 = no event status conditions have occurred. 1 = an enabled event status condition has occurred.
4	16	MAV	0 = no output messages are ready. 1 = an output message is ready.
3	8	—	0 = not used
2	4	—	0 = not used
1	2	—	0 = not used
0	1	—	0 = not used
0 = False = Low 1 = True = High			

*TRG
(Trigger)

The *TRG command has the same effect as if the RUN command had been sent.

Command Syntax:	*TRG
Example:	OUTPUT 707;"*TRG"

*TST
(Test)

The *TST query performs a self-test on the instrument. The result of the test is placed in the output queue.

A zero indicates the test passed and a non-zero value indicates the test failed.

If a test fails, refer to the troubleshooting section of the analyzer service manual.

Query Syntax:	*TST?
Returned Format:	<result><NL>
Where:	<result> ::= 0 or non-zero value
Where:	0 indicates the test passed. Non-zero indicates the test failed.
Example:	OUTPUT 707;"*TST?" ENTER 707;Result PRINT Result

*WAI
(Wait)

The *WAI command has no function in the analyzer, but is parsed for compatibility with other instruments.

Command Syntax:	*WAI
Example:	OUTPUT 707;"*WAI"

Instrument specific commands

:ACQuire:COMPlete

The :ACQuire:COMPlete command specifies the minimum completion criteria for an acquisition. The parameter determines what percentage of the time buckets need to be "full" before an acquisition is considered complete. If you are in the NORMAL mode, the instrument only needs one data bit per time bucket for that time bucket to be considered full.

The range for the :COMPlete command is 0 to 100 and indicates the minimum percentage of time buckets that must be "full" before the acquisition is considered complete. If the complete value is set to 100%, all time buckets must contain data for the acquisition to be considered complete. If the complete value is set to zero, then one acquisition cycle will take place.

NOTE

When the analyzer is in the averaging mode, it always has 100% completion for all acquisitions. The :ACQUIRE:COMPLETE does not have any effect.

The :COMPlete query returns the completion criteria for the currently selected mode.

For any of the horizontal vernier setting not equal to the 1-2-5 sequence, 100% completion cannot be achieved. Setting at 40% is suggested.

NOTE

The 2 ns/div (20 ns full scale) never reaches 100% complete. For this range, set complete to 40%.

Command Syntax:	:ACQuire:COMPlete <comp>
Where:	<comp> ::= 0 to 100 percent
Example:	OUTPUT 707;":ACQUIRE:COMPLETE 85"
Query Syntax:	:ACQuire:COMPlete?
Returned Format:	<comp><NL>
Where:	<comp> ::= 0 to 100 |integer-NR1 format|
Example:	OUTPUT 707;":ACQUIRE:COMPLETE?" ENTER 707;Cmp PRINT Cmp

:ACQuire:COUNt

In the average mode, the :ACQuire:COUNt command specifies the number of values to be averaged for each time bucket before the acquisition is considered complete for that time bucket.

When the acquisition type is set to AVERAGE, the count can be 8, 64, or 256. Any value can be sent in this mode; however, the value will be rounded to the nearest value of 8, 64, or 256.

The :COUNt query returns the currently selected count value for the averaging mode.

Command Syntax:	:ACQuire:COUNt <count>
Where:	<count> ::= 8, 64, or 256
Example:	OUTPUT 707;":ACQUIRE:TYPE AVERAGE" !select acquisition type OUTPUT 707;":ACQUIRE:COUNT 256"
Query Syntax:	:ACQuire:COUNt?
Returned Format:	<count><NL>
Where:	<count> ::= 8, 64, 256 |integer-NR1 format|
Example:	OUTPUT 707;":ACQ:COUNT?" ENTER 707;Cnt PRINT Cnt

:ACQuire:POINts

The :ACQuire:POINts query returns the number of points that the hardware will acquire from the input signal. To set the number of points to be transferred from the analyzer, use the command :WAVEform:POINts.

The :WAVEform:POINts query will return the number of points to be transferred from the analyzer. For the relationship between these two commands, see Chapter 6, "Waveform Commands".

Query Syntax:	:ACQuire:POINts?
Returned Format:	<points_argument><NL>
Where:	<points_argument> ::= 1 to 4000
Example:	DIM Pnts$[50] OUTPUT 707;":ACQUIRE:POINTS?" ENTER 707;Pnts$ PRINT Pnts$

:ACQuire:SETup

The :ACQuire:SETup query returns the current setup conditions for all commands in the ACQuire subsystem.

Query Syntax:	:ACQuire:SETup?
Returned Format:	ACQ:TYPE{NORM | AVER | PEAK};COUNT{1 to 1024}; POINTS{1 to 4000};COMP{0 to 100}<NL>
Example:	DIM Setup$[300} OUTPUT 707;":ACQUIRE:SETUP?" ENTER 707;Setup$ PRINT Setup$

:ACQuire:TYPE

The :ACQuire:TYPE command selects the type of acquisition that is to take place. There are three acquisition types: NORMAL, AVERAGE, and PEAK DETECT. For more information on these types, see Chapter 6.

The :TYPE query returns the current acquisition type.

Command Syntax:	:ACQuire:TYPE {NORMal | AVERage | PEAK }
Example:	OUTPUT 707;":ACQUIRE:TYPE AVER"
Query Syntax:	:ACQuire:TYPE?
Returned Format:	{NORM | AVER | PEAK}<NL>
Example:	DIM Tpe$[50] OUTPUT 707;":ACQUIRE:TYPE?" ENTER 707;Tpe$ PRINT Tpe$

:ASTore

The :ASTore command acquires data for the active waveform display. The data is acquired as defined by the trigger mode. If the trigger mode is in SINGLE, the :ASTore command enables the trigger once and displays the acquired data on the screen. This also occurs when the front-panel (Run) key is pressed while the instrument is STOPPED. If the trigger mode is set to AUTO, AUTLEVEL, or NORMAL, the :ASTore command enables the trigger repeatedly and displays the data it acquires continuously on the screen. This is the same thing that happens when the front-panel (Autostore) key is pressed. To turn off the ASTORE display mode, send the :RUN command.

Command Syntax:	:ASTore
Example:	OUTPUT 707;":ASTORE"

:AUToscale

The :AUToscale command evaluates all input signals and sets the correct conditions to display the signals. When the :AUToscale command is sent, the following conditions are set:

- main timebase mode
- vertical sensitivity
- vertical offset
- trigger to autolevel mode
- display mode to normal
- trigger level, holdoff, and slope
- sweep speed of the displayed channel

In addition, the :AUToscale command turns off the following items:

- Markers.
- All measurements.
- Trace memories.

If input signals are present on more than one vertical input, the timebase is triggered on the highest number channel. If no signals are found on any vertical input, the analyzer is returned to its former state.

Command Syntax:	:AUToscale
Example:	OUTPUT 707;":AUTOSCALE"

:BLANk

The :BLANk command turns off (stops displaying) the specified channel or pixel memory. To turn off a channel display, use the command :BLANk CHANNEL{1|2|}. To turn off a pixel memory display, use the command :BLANk PMEMORY{1|2}. Use the command :BLANk EXTERNAL to turn off the external trigger view.

Use the :VIEW command to turn on (start displaying) an active channel or pixel memory. Use the :VIEW command to turn on an active channel, pixel memory, or the external trigger view.

Command Syntax:	:BLANk <display>
Where:	<display> ::= {CHANnel{1 | 2} PMEMory{1 | 2 } EXTernal}
Example:	OUTPUT 707;":BLANK CHANNEL1"

:CHANnel:BWLimit

The :CHANnel <N>:BWLimit command controls an internal low-pass filter. When the filter is ON, the bandwidth of the specified channel is limited to approximately 20 MHz. The bandwidth limit filter can be used on channels 1 and 2.

The :BWLIMIT query returns the current setting.

Command Syntax:	:CHANnel <N>:BWLimit {ON | OFF}
Where:	<N> ::= 1 or 2
Example:	OUTPUT 707;":CHANNEL2:BWL ON"
Query Syntax:	:CHANnel& <N>:BWLimit?
Where:	<N> ::= 1 or 2
Returned Format:	{ON | OFF}<NL>
Example:	DIM Hf$[50] OUTPUT 707;":CHAN1:BWL?" ENTER 707;Hf$ PRINT Hf$

:CHANnel:COUPling

The :CHANnel<N>:COUPling command selects the input coupling for the specified channel. The coupling for the electrical channels can be set to AC, DC, or GND. The optical channel can be set to DC or GND.

The :COUPling query returns the current coupling for the specified channel.

Command Syntax:	:CHANnel<N>:COUPling {AC | DC | GND}
Where:	<N> ::= 1 or 2
Example:	OUTPUT 707;":CHAN2:COUP DC"
Query Syntax:	:CHANnel<N>:COUPling?
Where:	<N> ::= 1 or 2
Returned Format:	{AC | DC | GND}<NL> for electrical channels 1 and 2 {DC | GND}<NL> for optical channel 2
Example:	DIM Ch$[50] OUTPUT 707;":CHAN2:COUPLING?" ENTER 707;Ch$ PRINT Ch$

:CHANnel:INPut

The :CHANnel<N>:INPut command selects the impedance of the input source. FIFTY selects a 50Ω input impedance. ONEMEG selects a 1 MΩ input impedance.

When optical channel 2 is activated, the input cannot be set to 1 MΩ.

The :INPut query identifies the currently selected input source impedance.

Command Syntax:	:CHANnel<N>:INPut {FIFTY | ONEMEG}
Where:	<N> ::= 1 or 2
Example:	OUTPUT 707;":CHAN2:INP FIFTY"
Query Syntax:	:CHANnel<N>:INPut?
Where:	<N> ::= 1 or 2
Returned Format:	{FIFTY | ONEMEG}<NL>
Example:	DIM Input$[50] OUTPUT 707;":CHANNEL1:INPUT?" ENTER 707;Input$ PRINT Input$

:CHANnel:INVert

The :CHANnel<N>:INVert command selects the input inversion for the specified channel. The inversion may be ON or OFF. Inversion can be set for electrical channels 1 and 2.

Inversion is not allowed on channel 2 when the optical channel is active.

Command Syntax:	:CHANnel<N>:INVert {ON|OFF}
Where:	<N> ::= 1 or 2
Example:	OUTPUT 707;":CHAN1:INVERT ON"

Query Syntax:	:CHANnel<N>:INVert?
Where:	<N> = 1 or 2
Returned Format:	{ON | OFF}<NL>
Example:	DIM Ch$[50] OUTPUT 707;":CHAN2:INVERT?" ENTER 707;Ch$ PRINT Ch$

:CHANnel:IOPTical

The :CHANnel2:IOPTical command activates the calibrated watts vertical scale for channel 2. This command should be sent prior to the execution of automatic power measurements such as Pmax or Pbase.

The :IOPTical query returns the current state of the optical channel, ON or OFF.

Command Syntax:	:CHANnel2:IOPTical {ON | OFF}
Example:	OUTPUT 707;":CHAN2:IOPT ON"

Query Syntax:	:CHANnel2:IOPTical?
Returned Format:	{ON | OFF}<NL>
Example:	DIM Opt$[20] OUTPUT 707;":CHAN2:IOPT?" ENTER 707;Opt$ PRINT Opt$

:CHANnel:OFFSet

The :CHANnel<N>:OFFSet command sets the value represented at center screen for the selected channel. The range of legal values vary with the value set with the :RANGe command. If you select an offset value outside the legal range, the offset is automatically set to the nearest legal value.

When optical channel 2 is selected, the offset value can be set to a microwatt value.

The :OFFSet query returns the current offset value for the selected channel.

Command Syntax:	:CHANnel<N>:OFFSet <value> [W, mW, uW or V, mW, uW]
Where:	<N> ::= 1 or 2 <value> ::= offset value
Examples:	OUTPUT 707;":CHAN1:OFFS 200MV" OUTPUT 707;":CHAN2:OFFSET 20E-3V"
Query Syntax:	:CHANnel<N>:OFFSet?
Returned Format:	<value><NL> <value> ::= offset value in volts or watts [exponential-NR1-NR3 format]
Example:	OUTPUT 707;":CHANNEL2:OFFSET?" ENTER 707;Offset PRINT Offset

:CHANnel:PROBe

The :CHANnel<N>:PROBe command specifies the probe attenuation factor for the selected channel. The probe attenuation factor may be 1, 10, or 100. This command does not change the actual input sensitivity of the analyzer. It changes the reference constants for scaling the display factors, for making automatic measurements, for setting trigger levels, etc.

When optical channel 2 is selected, the :CHANnel<N>:PROBe command cannot be used. For information on optical attenuation correction, refer to the :CHANnel2:XOATtn command.

The :PROBe query returns the current probe attenuation factor for the selected channel.

Command Syntax:	:CHANnel<N>:PROBe <attenuation>
Where:	<N> ::= 1 or 2 <attenuation> ::= X1, X10, X100
Example:	OUTPUT 707;":CHANNEL2:PROBE X10"
Query Syntax:	:CHANnel<N>:PROBe?
Where:	<N> ::= 1 or 2
Returned Format:	<attenuation><NL>
Where:	<attenuation> ::= X1, X10, or X100
Example:	DIM Prb$[50] OUTPUT 707;":CHANNEL1:PROBE?" ENTER 707;Prb$ PRINT Prb$

:CHANnel:PROTect

The :CHANnel<N>:PROTect command turns ON or OFF 50Ω overvoltage protection.

The :PROTect query returns the current status of the overvoltage protection.

Command Syntax:	:CHANnel<N>:PROTect {OFF | ON}
Where:	<N> ::= 1 or 2
Example:	OUTPUT 707;":CHANNEL1:PROTECT ON"
Query Syntax:	:CHANnel<N>:PROTect?
Where:	<N> ::= 1 or 2
Returned Format:	{OFF | ON}
Example:	DIM Prot$[30] OUTPUT 707;":CHAN2:PROT?" ENTER 707;Prot$ PRINT Prot$

:CHANnel:RANGe

The :CHANnel<N>:RANGe command defines the full-scale vertical axis of the selected channel. The voltage RANGE for electrical channels 1 and 2 can be set to any value from 16 mV to 40 V when using 1:1 probe attenuation. If the probe attenuation is changed, the range value is multiplied by the probe attenuation factor.

When optical channel 2 is active, the power RANGE units are specified in microwatts. The minimum and maximum values depend on the currently selected optical λ calibration.

Optical Calibration	Min	Max
780 nm	20 μW	500 μW
1300 nm	2 μW	50 μW
1550 nm	2 μW	50 μW

The :RANGe query returns the current full-scale range setting for the specified channel.

Command Syntax:	:CHANnel<N>:RANGe <range> [W, mW, uW or V, mV, uV]
Where:	<N> ::= 1 or 2 <range> ::= full-scale range value |CHANNEL1 or CHANNEL2|
Examples:	:OUTPUT 707;":CHANNEL1:RANGE 0.64V" :OUTPUT 707;":CHANNEL2:RANGE 100 uW"
Query Syntax:	:CHANnel<N>:RANGe?
Returned Format:	<range><NL> <range> Channels 1 and 2 ::= full-scale range value |exponential-NR3 format|
Examples:	OUTPUT 707;":CHAN2:RANGE?" ENTER 707;Rng PRINT Rng DIM RNG$[50] OUTPUT 707;":CHAN1:RANGE?" ENTER 707;RNG$ PRINT RNG$

:CHANnel:SETup

The :CHANnel<N>:SETup query returns the current settings for all the Channel subsystem commands, except for the selected MATH.

Query Syntax:	:CHANnel:SETup?
Returned Format:	
For Channel1 or Channel2:	CHAN<n>:RANGE<range>;OFFSET<offset>; COUP{AC|DC|GND};BWLIMIT{ON|OFF};INVERT{ON|OFF}; VERNIER {ON | OFF};PROBE{X1|X10|X100}<NL>
Where:	<N>::= 1 or 2 <range>::=full-scale range in volts or watts |exponential-NR3 format| <offset>::=offset value in volts or watts |exponential-NR3 format|
For Channel1:	CHAN<1>:RANGE<range>;OFFSET<offset>;COUP{AC|DC|GND}; BWLIMIT{ON|OFF};INVERT{ON|OFF};VERNIER {ON | OFF}; PROBE{X1|X10|X100}; INPUT{FIFTY|ONEMEG}; PROTECT{OFF|ON}<NL>
Where:	<range>::=full-scale range in volts |exponential-NR3 format| <offset>::=offset value in volts |exponential-NR3 format|
For Channel2:	CHAN<2>:RANGE<range>;OFFSET<offset>;COUP{AC |DC|GND}; BWLIMIT{ON|OFF};INVERT{ON|OFF};VERNIER {ON | OFF}; PROBE{X1|X10|X100}; INPUT{FIFTY|ONEMEG}; PROTECT{OFF|ON}; SKEW <skew_value><NL>
Where:	<range>::=full-scale range in volts or watts |exponential-NR3 format| <offset>::=offset value in volts or watts |exponential-NR3 format| <skew_value>::=skew value in NR3 format
Example:	DIM Stp$[300] OUTPUT 707;":CHANNEL:SETUP?" ENTER 707; Stp$ PRINT Stp$

:CHANnel:SKEW

The :CHANnel2:SKEW command sets the skew between channels 1 and 2.

The :SKEW query identifies the current skew between channels 1 and 2.

Command Syntax:	:CHANnel2:SKEW <skew_value>
Where:	<skew_value> ::= is the value for the skew in NR3 format
Example:	OUTPUT 707;":CHAN2:SKEW 2.5E-008"
Query Syntax:	:CHANnel2:SKEW?
Returned Format:	<skew_value>
Example:	DIM Skew$[30] OUTPUT 707;":CHAN2:SKEW?" ENTER 707;Skew$ PRINT Skew$

:CHANnel:VERNier

The :CHANnel<N>:VERNier command selects the vernier mode of the Volts/Div or Watts/Div control. The command affects only the user control Volts/Div or Watts/Div knob and not the :CHAN <N>:RANGe command. :VERNier may be selected for channels 1 and 2.

Command Syntax:	:CHANnel<N>:VERNier {ON | OFF}
Where:	<N> ::= 1 or 2
Example:	OUTPUT 707;":CHAN1:VERN ON"
Query Syntax:	:CHANnel<N>:VERNier?"
Returned Format:	{ON | OFF}<NL>
Example:	DIM Ver$[50] OUTPUT 707;":CHAN2:VERNIER?" ENTER 707;Ver$ PRINT Ver$

:CHANnel:XEATtenuation

The :CHANnel:XEATtenuation command sets the channel external electrical attenuation.

The :XEATtenuation query returns the current external attenuation value setting.

Command Syntax:	:CHANnel<N>:XEATtenuation<attenuation>
Where:	<N> ::= 1 or 2
Where:	<attenuation> ::= floating point value of attenuation in dB or linear ratio, NR3 format.
Example:	OUTPUT 707;":CHANNEL2:XEATtenuation:"
Query Syntax:	:CHANnel<N>:XEATtenuation?
Returned Format:	<attenuation><NL>
Where:	<N> ::= 1 or 2
Where:	<attenuation> ::= floating point value of attenuation in dB, NR3 format.
Example:	OUTPUT 707;":CHANnel2:XEATtenuation?" ENTER 707;Rsp$ PRINT Rsp$

:CHANnel:XEAUnits

The :CHANnel:XEAUnits command and query sets the channel external electrical attenuation units. These units are used when inputting and outputting the channel external electrical attenuation.

Command Syntax:	:CHANnel<N>:XEAUnits<units>
Where:	<N> ::= 1 or 2
Where:	<units> ::= LINear or DB
Example:	OUTPUT 707;":CHANNEL2:XEAUnits:LIN"
Query Syntax:	:CHANnel<N>:XEAUnits?
Returned Format:	<units><NL>
Where:	<N> ::= 1 or 2
Where:	<units> ::= LINear or DB
Example:	OUTPUT 707;":CHANnel2:XEAUnits?" ENTER 707;Rsp$ PRINT Rsp$

:CHANnel:XOATtn

The :CHANnel2:XOATtn command sets the attenuation factor for the optical channel. The external attenuation factor can be a floating point value (in dB) between 0 and 20 dB. This command changes the reference constants for scaling the display factors, for making automatic measurements, for setting trigger levels and so forth. This command does *not* change the actual input sensitivity of the analyzer.

Command Syntax:	:CHANnel2:XOATtn<attenuation>
Where:	<attenuation> ::= floating point value of attenuation in dB, NR3 format.
Example:	OUTPUT 707;":CHANNEL2:XOATtn:3.1E00"
Query Syntax:	:CHANnel2:XOATtn?
Returned Format:	<attenuation><NL>
Where:	<attenuation> ::= floating point value of attenuation in dB, NR3 format.
Example:	DIM Att$[50] OUTPUT 707;":CHANnel2:XOATtn?" ENTER 707;Att$ PRINT Att$

:DIGitize

The :DIGitize command is used to acquire waveform data for transfer over the interface. Sending the command causes an acquisition to take place on the specified channels with the resulting data being placed in the channel buffer. Sending the :DIGitize command also turns off any unused channels.

The sources for the :DIGitize command are channels 1 and 2. The external trigger is not a measurement channel and cannot be digitized. The :DIGitize command will acquire data for all channels that are turned on when the :DIGitize is started. The parameters of :DIGitize are the channels that will acquire to the specified criteria.

Throughput can be improved by turning off (BLANK) any undesired channels. Be sure any desired channels are turned on (VIEW). :DIGitize works fastest for non-vernier 1-2-5 timebase settings. The 2 ns/div timebase setting also slows down the digitizing rate.

When the digitize operation is complete, the instrument is placed in the stopped mode. When the instrument is restarted, with a :RUN command or the front-panel (Run) key, the digitized data stored in the channel buffers is overwritten. Therefore, ensure all operations that require the digitized data are completed before restarting the analyzer.

The ACQUIRE subsystem commands are used to set up conditions such as TYPE, and COUNT for the next :DIGitize command. See the ACQUIRE subsystem for a description of these commands. To determine the actual number of points that are acquired and how the data is transferred, refer to the WAVEFORM subsystem commands.

Command Syntax:	:DIGitize CHANnel<N>[,CHANnel<N>]
Where:	<N> ::= 1 or 2.
Example:	OUTPUT 707;":DIGITIZE CHANNEL1, CHANNEL2"

:DISPlay:COLumn

The :DISPlay:COLumn command specifies the starting column for the subsequent :LINE command.

The :COLumn query returns the column where the next LINE will start.

Command Syntax:	:DISPlay:COLumn <number>
Where:	<number> ::= 0 through 63
Example:	OUTPUT 707;":DISPLAY:COLUMN 50"
Query Syntax:	:DISPlay:COLumn?
Returned Format:	<number><NL>
Where:	<number> ::= 0 through 63 |integer-NR1 format|
Example:	OUTPUT 707;":DISPLAY:COLUMN?" ENTER 707;Clmn PRINT Clmn

:DISPlay:GRID

The :DISPlay:GRID command selects the grid as ON or OFF.

The :GRID query returns the status of the grid.

Command Syntax:	:DISPlay:GRID {ON | OFF}
Example:	OUTPUT 707;":DISPLAY:GRID ON"
Query Syntax:	:DISPlay:GRID?
Returned Format:	{ON | OFF}<NL>
Example:	DIM Grd$[30] OUTPUT 707;":DISPLAY:GRID?" ENTER 707;Grd$ PRINT Grd$

:DISPlay:INVerse

The :DISPlay:INVerse command determines whether text sent with the :LINE command in the DISPLAY subsystem is to be written with the INVerse attribute. If the inverse attribute is ON, the text will be written in inverse video.

The :INVerse query returns the current state of this command.

Command Syntax:	:DISPlay:INVerse {ON | OFF}
Example:	OUTPUT 707;":DISPLAY:INVERSE OFF"
Query Syntax:	:DISPlay:INVerse?
Returned Format:	{ON | OFF}<NL>
Example:	DIM Iv$[30] OUTPUT 707;":DISP:INVERSE?" ENTER 707;Iv$ PRINT Iv$

:DISPlay:LINE

The :DISPlay:LINE command writes a text string to the screen. The text is displayed starting at the location of the current row and column. The row and column can be set by the :DISPlay:ROW and :DISPlay:COLumn commands prior to sending the :DISPlay:LINE command. Text can be written over the entire screen with the :LINE command.

If the text string is longer than the available space on the current line, the text wraps around to the start of the same line. In any case, the :ROW value is incremented by one and the :COLUMN value remains the same. The next :DISPlay:LINE command will write on the next line of the display starting at the same column as the previous text. After writing line 20, the last line in the display area, ROW is reset to 1.

Command Syntax:	:DISPlay:LINE <quoted string>
Where:	<quoted string> ::= any series of ASCII characters enclosed in quotes.
Example:	OUTPUT 707;":DISPLAY:LINE ""ENTER PROBE ATTENUATION"""

:DISPlay:PIXel

The :DISPlay:PIXel command sets the intensity of a pixel on the screen. The position of the pixel is determined by the x, y coordinates specified in the parameter.

The :PIXel query returns the current intensity of the specified pixel. Cycling the power on the analyzer will clear all the pixels.

A portion of the full-bright screen is part of the active waveform area and cannot be modified by the :PIXel command. This portion of the display is pixels with y-coordinates 12 to 303. Any pixel on the half-bright screen can be modified with the :PIXel command.

Command Syntax:	:DISPlay:PIXel <x>, <y>, <intensity>
Where:	<x>::= the x coordinate of the pixel to be set, 0 to 511 <y>::= the y coordinate of the pixel to be set, 0 to 303 <intensity>::= 0 to clear pixel 1 for half bright 2 for full bright other value to clear pixel
Example:	OUTPUT 707;":DISPLAY:PIXEL 100,100,1"
Query Syntax:	:DISPlay:PIXel?<x>,<y>
Where:	<x>::= the x-coordinate of the pixel <y>::= the y-coordinate of the pixel
Returned Format:	<intensity><NL>
Where:	<intensity> ::= " "0 for pixel off " "1 for pixel with half-bright on " "2 for pixel with full-bright on " "3 for pixel with both half-bright and full-bright on
Example:	OUTPUT 707;":DISP:PIX?100,100" ENTER 707;Intensity PRINT Intensity

:DISPlay:ROW

The :DISPlay:ROW command specifies the starting row on the screen for subsequent :LINE commands. The ROW number remains constant until another :ROW command is received, or until it is incremented by the :LINE command. The ROW value can be set to 1 through 20.

The :ROW query returns the current ROW number.

Command Syntax:	:DISPlay:ROW <row number>
Where:	<row number> ::= 1 through 20
Example:	OUTPUT 707;":DISPLAY:ROW 10"
Query Syntax:	:DISPLAY:ROW?
Returned Format:	<row number><NL>
Where:	<row number> ::= 1 through 20 [integer-NR1 format]
Example:	OUTPUT 707;":DISPLAY:ROW?" ENTER 707;Rw PRINT Rw

:DISPlay:SETup

The :DISPlay:SETup query returns the current settings for all commands in the DISPLAY subsystem.

Query Syntax:	:DISPlay:SETup?
Returned Format:	DISP:ROW {1 to 20};COL {0 TO 63};INVERSE {ON|OFF};GRID {ON|OFF};SOURCE PMEM{1|2}<NL>
Example:	DIM Stp$[300] OUTPUT 707;":DISPLAY:SETUP?" ENTER 707;Stp$ PRINT Stp$

N O T E

Source setting is not valid.

:DISPlay:TEXT

The :DISPlay:TEXT command blanks the user text area on the screen. When this command is sent, all text on the entire screen is blanked. This command has only one parameter, :BLANk.

There is no query form of this command.

Command Syntax:	:DISPlay:TEXT BLANk
Example:	OUTPUT 707;":DISPLAY:TEXT BLAN"

:DITHer

The :DITHer command selects an anti-aliasing function for slow sweep speeds. A small random time offset is added to the sampling clock. By default, after power up, the analyzer is not sampling at exact regular time intervals. Aliasing is much less likely to happen. The small time offset is not noticeable on the display of the analyzer.

Some data analysis tools that are sensitive to the regularity of sampling time may have problems. One example is the Fast Fourier Transform. The :DITHer:OFF command should be used to turn off the dithering when waveform data will be used by these data analysis tools.

Command Syntax:	:DITHer {ON | OFF}
Example:	OUTPUT 707;":DITHER ON"
Query Syntax:	:DITHer?
Returned Format:	{ON |OFF}<NL>
Example:	DIM Dither$[5] OUTPUT 707;":DITHER?" ENTER 707;Dither$ PRINT Dither$

:ERASe

The :ERASe command erases the screen. :ERASe is the same as pressing the front-panel Erase key.

Command Syntax:	:ERASe
Example:	OUTPUT 707;"ERASE"

:EXTernal:COUPling

The :EXTernal:COUPling command selects the input coupling for the selected external trigger source. The coupling can be set to DC, AC, or GND. You can use the AC setting only if the input is set to ONEMEG.

The :COUPling query returns the current selection.

Command Syntax:	:EXTernal:COUPling {DC | AC | GND}
Example:	OUTPUT 707;":EXTERNAL:COUPLING DC"
Query Syntax:	:EXTernal:COUPling?
Returned Format:	{DC | AC | GND}<NL>
Example:	DIM Coup$[50] OUTPUT 707;":EXT:COUP?" ENTER 707;Coup$ PRINT Coup$

:EXTernal:INPut

The :EXTernal:INPut command sets the input impedance for the external trigger. The input can be either fifty Ω or one MΩ.

The :INPut query returns the value of the input impedance for the external trigger.

Command Syntax:	:EXTernal:INPut { FIFTY | ONEMEG }
Where:	FIFTY is used for an external trigger with fifty ohm impedance. ONEMEG is used for an external triggers with one megohm impedance.
Example:	OUTPUT 707;":EXTERNAL:INPUT FIFTY"
Query Syntax:	:EXTernal:INPut?
Returned Format:	{FIFTY | ONEMEG}
Where:	FIFTY indicates the input impedance is fifty ohms. ONEMEG indicates the input impedance is one megohm.
Example:	DIM EXTINP$[50] OUTPUT 707;":EXTERNAL:INPUT?" ENTER 707;EXTINP$ PRINT EXTINP$

:EXTernal:OFFSet

The :EXTernal:OFFSet command sets the offset for the external trigger view. The :OFFSet query returns the offset for the external trigger.

Command Syntax:	:EXTernal:OFFSet <offset_value>
Where:	<offset_value> ::= offset value in volts |exponential-NR3 format|
Example:	OUTPUT 707;":EXTERNAL:OFFSET 50"
Query Syntax:	:EXTernal:OFFSet?
Returned Format:	<offset_value><NL>
Where:	<offset_value> ::= offset value in volts |exponential-NR3 format|
Example:	DIM OFFS$[50] OUTPUT 707;":EXTERNAL:OFFSET?" ENTER 707;OFFS$ PRINT OFFS$

:EXTernal:PROBe

The :EXTernal:PROBe command specifies the probe attenuation factor for the external trigger. The probe attenuation factor may be 1, 10, or 100. This command does not change the actual input sensitivity of the analyzer. It changes the reference constants for scaling the display factors, for making automatic measurements, for setting trigger levels, etc. This command also turns off the probe sense on the external trigger.

The :PROBe query returns the current probe attenuation factor for the external trigger.

Command Syntax:	:EXTernal:PROBe <attenuation>
Where:	<attenuation> ::= X1, X10, X100
Example:	OUTPUT 707;":EXTERNAL:PROBE X10"
Query Syntax:	:EXTernal:PROBe?
Returned Format:	<attenuation><NL>
Where:	<attenuation> ::= X1, X10, or X100
Example:	DIM Prb$[50] OUTPUT 707;":EXTERNAL:PROBE?" ENTER 707;Prb$ PRINT Prb$

:EXTernal:PROTect

The :EXTernal:PROTect command turns ON or OFF fifty Ω overvoltage protection.

The :PROTect query returns the overvoltage protection status.

Command Syntax:	:EXTernal:PROTect {OFF | ON}
Example:	OUTPUT 707;":EXTERNAL:PROT ON"
Query Syntax:	:EXTernal:PROTect?
Returned Format:	{OFF | ON}<NL>
Example:	DIM PROT$[30] OUTPUT 707;":EXT:PROT?" ENTER 707; PROT$ PRINT PROT$

:EXTernal:SETup

The :EXTernal:SETup query returns all current settings of the commands in the External subsystem.

Query Syntax:	:EXTernal:SETup?
Returned Format:	EXT:OFFSET <offset_value>; COUP {DC | AC | GND}; PROBE <attenuation>; INP {FIFTY | ONEMEG}; PROTECT {OFF | ON}; SKEW <skew_value>
Where:	<offset_value>::= external trigger offset value |exponential-NR3 format| <attenuation> ::= X1, X10, or X100 <skew_value>::= external trigger skew value |exponential-NR3 format|
Example:	DIM Set$[200] OUTPUT 707;":EXT:SET?" ENTER 707;Set$ PRINT Set$

:EXTernal:SKEW

The :EXTernal:SKEW command sets the skew between the external trigger view and Channel1.

The :SKEW query returns the current skew value.

Command Syntax:	:EXTernal:SKEW <skew_value>
Where:	<skew_value> ::= the external trigger skew |exponential-NR1-NR3 format|
Example:	OUTPUT 707;":EXTERNAL:SKEW +2.50000000E+002"
Query Syntax:	:EXTernal:SKEW?
Returned Format:	<skew_value><NL>
Example:	DIM SKEWs$[30] OUTPUT 707;":EXT:SKEW?" ENTER 707;SKEW$ PRINT SKEW$

:FUNCtion:CENTer

This command is only used when an FFT operation is selected. Refer to "FFT measurement menu" in Chapter 2 for additional information on FFT.

The :FUNCtion:CENTer command sets the center frequency when FFT is selected. If you set the center frequency to a value outside of the legal range, it is automatically set to the nearest legal value.

The operation of the :FUNCtion:CENTer command is identical to the `Cent Freq` softkey. Refer to "Math Functions" in Chapter 2 for additional information.

Step size is calculated as follows:

$$Step\ size = \left(\frac{1000}{1024}\right) Timebase\ Range$$

Command Syntax:	:FUNCtion2:CENTer <frequency>
Where:	<frequency> ::= current center frequency in Hertz (0 Hz to 10.00 GHz)
Example:	OUTPUT 707;":FUNC2:CENT 976.6"
Query Syntax:	:FUNCtion2:CENTer?
Returned Format:	<frequency><NL>
Where:	<frequency> ::= current center frequency in Hertz (0 Hz to 10.00 GHz) (exponential–NR3 format)
Example:	OUTPUT 707;":FUNCtion2:CENTer?" ENTER 707;Frq PRINT Frq

:FUNCtion:MOVE

This command is only used when an FFT operation is selected. Refer to "FFT measurement menu" in Chapter 2 for additional information on FFT.

:FUNCtion:MOVE LEFT changes the left most graticule (or beginning of the sweep) to 0 Hz. Center frequency is adjusted to one-half of the current span setting. If FFT is not active, a message is displayed on the screen.

The operation of the :FUNCtion:MOVE LEFT command is identical to the `Move 0 Hz To Left` softkey. Refer to "Math Functions" in Chapter 2 for additional information.

Command Syntax:	:FUNCtion2:MOVE {LEFT}
Where:	<frequency> ::= current center frequency in Hertz [0 Hz to 10.00 GHz]
Example:	OUTPUT 707;":FUNC2:MOVE LEFT"
Example:	OUTPUT 707;":FUNCtion2:MOVE LEFT"

:FUNCtion:OFFSet

The :FUNCtion:OFFSet command sets the value that is represented at center screen for the selected function.

The :OFFSet query outputs the current value offset value for the selected function.

If you set the offset to a value outside the legal range, the offset value is automatically set to the nearest legal value.

The operation of the :FUNCtion:OFFSet command is identical to the `Offset` softkey. Refer to "Math Functions" in Chapter 2 for additional information.

Command Syntax:	:FUNCtion<1 | 2>:OFFSet<offset>
Where:	<offset> ::= value at center of screen Range = ±10 times the current sensitivity of the selected function
Example:	OUTPUT 707;":FUNC1:OFFS 1.2"
Query Syntax:	:FUNCtion<1 | 2>:OFFSet?
Returned Format:	<offset><NL>
Where:	<offset> ::= value at the center of the screen |exponential-NR3 format|
Example:	OUTPUT 707;":FUNCtion2:CENTer?" ENTER 707;Frq PRINT Frq

:FUNCtion:OPERation

The :FUNCtion:OPERation command sets the desired operation for a function.

The :OPERation query outputs the current operation for the selected function.

Command Syntax:	:FUNCtion1:OPERation { ADD | SUBTract | MULTiply} :FUNCtion2:OPERation { INTegrate | DIFFerentiate | FFT }
Example:	OUTPUT 707;":FUNC1:OPER ADD"
Query Syntax:	:FUNCtion<N>:OPERation?
Returned Format:	{ADD | SUBTract | MULTiply | INTegrate | DIFFerentiate | FFT}

:FUNCtion:PEAKs

This query is only used when an FFT operation is selected. Refer to "FFT measurement menu" in Chapter 2 for additional information on FFT.

The :PEAKs? query causes a peak search to be performed and the queried value to be returned. A peak search sets FREQ1 and DB1 to the peak with the highest amplitude and sets FREQ2 and DB2 to the peak with the next highest amplitude.

When the active cursor is "V" (V1, V2, or both), the marker values in dBV are automatically displayed at the bottom of the analyzer screen. The difference in dBV (ΔV) between the two peaks is also displayed.

When the active cursor is "f" (f1, f2, or both), the marker values in frequency are automatically displayed at the bottom of the analyzer screen. The difference in frequency (Δf) between the two peaks is also displayed.

If FREQ1 is used, the frequency in hertz is returned for the peak with the highest amplitude. If DB1 is used, the amplitude in dBV is returned for the peak with the highest amplitude. If FREQ2 is used, the frequency in hertz is returned for the peak with the second highest amplitude. If DB2 is used, the amplitude in dBV is returned for the peak with the second highest amplitude. If a peak is not found, an invalid measurement (9.9E37) is returned.

Query Syntax:	:FUNCtion2:PEAKs? {FREQ1|DB1|FREQ2|DB2}
Returned Format:	<measurement> <NL>
Where:	<measurement> ::= value of the peak specified |exponential-NR3 format|
Example:	OUTPUT 707; ":FUNC2:PEAK? FREQ1"

:FUNCtion:RANGe

The :FUNCtion:RANGe command defines the full scale vertical axis for the selected function.

The :RANGe query outputs the current full scale range value for the selected function.

Command Syntax:	:FUNCtion<N>:RANGe <range>
Example:	OUTPUT 707;":FUNC1:RANG 20E-3"
Query Syntax:	:FUNCtion<N>:RANGe?
Returned Format:	<range> <NL>
Where:	<range> ::= full scale vertical axis value Range for function 1 is 8E-6 to 8E+6 Range for integrate function is 8E-9 to 400E+3 Range for differentiate function is 8E-6 to 1.6E11 Range for FFT function is 8 to 400 dB/div

:FUNCtion:REFerence

This command is only used when an FFT operation is selected. Refer to "FFT measurement menu" in Chapter 2 for additional information on FFT.

The :FUNCtion:REFerence command sets the reference level represented by the top graticule line when FFT is selected. If you set the reference level to a value outside of the legal range, it is automatically set to the nearest legal value. The operation of the :FUNCtion:REFerence command is identical to the `Ref Level` softkey. Refer to "Math Functions" in Chapter 2 for additional information.

The :REFerence query outputs the current reference level in dBV.

Command Syntax:	:FUNCtion2:REFerence <level>
Example:	OUTPUT 707;":FUNC2:REF 10"
Query Syntax:	:FUNCtion2:REFerence?
Returned Format:	<level> <NL>
Where:	<level> ::= current reference level in dBV |exponential-NR3 format| Range is —160.0 to 240 dBV in increments of 2.5 dBV

:FUNCtion:SOURce

The :FUNCtion2:SOURce command selects the source for function 2 operations.

The :SOURce query outputs the current source for function2 operations.

Choose between CHANnel1, CHANnel2 and FUNCtion1 to specify the desired source for FUNCtion2 DIFFerentiate, INTegrate, and FFT operations.

CHANnel1 uses the signal connected to the input 1 connector. CHANnel2 uses the signal connected to the input 2 connector. FUNCtion1 uses the result of function 1 as the source for FUNCtion2. This is either input 1 and 2 added, subtracted, or multiplied.

The operation of the :FUNCtion2:SOURce command is identical to the `Operand` softkey. Refer to "Math Functions" in Chapter 2 for additional information.

Command Syntax:	:FUNCtion2:SOURce {CHANnel1|CHANnel2|FUNCtion1}
Example:	OUTPUT 707;":FUNC2:SOURce CHAN1"
Query Syntax:	:FUNCtion2:SOURce?
Returned Format:	{CHANnel1|CHANnel2|FUNCtion1} <NL>

:FUNCtion:SPAN

This command is only used when an FFT operation is selected. Refer to "FFT measurement menu" in Chapter 2 for additional information on FFT.

The :SPAN query returns the current frequency span in hertz.

The :FUNCtion:SPAN command sets the frequency span of the display (left graticule to right graticule) when FFT is selected. If you set the frequency span to a value outside of the legal value, it is automatically set to the nearest legal value.

The operation of the :FUNCtion:SPAN command is identical to the `Freq Span` softkey. Refer to "Math Functions" in Chapter 2 for additional information.

Command Syntax:	:FUNCtion2:SPAN
Example:	OUTPUT 707;":FUNC2:SPAN 61.04 E3"
Query Syntax:	:FUNCtion2:SPAN?
Returned Format:	<NL>
Where:	::= 1.221 Hz to 9.766 GHz current frequency span in hertz [exponential-NR3 format]

:FUNCtion:VIEW

The :FUNCtion:VIEW command is used to turn the selected function ON or OFF.

The :VIEW query outputs the current state of the selected function.

The operation of the :FUNCtion:VIEW command is identical to the `Function 1 Off On` and `Function 2 Off On` softkeys. Refer to "Math Functions" in Chapter 2 for additional information.

Command Syntax:	:FUNCtion<N>:VIEW {ON | OFF}
Example:	OUTPUT 707;":FUNC2:VIEW ON"
Query Syntax:	:FUNCtion<N>:VIEW?
Returned Format:	{ON | OFF} <NL>

:FUNCtion:WINDow

This command is only used when an FFT operation is selected. Refer to "FFT measurement menu" in Chapter 2 for additional information on FFT.

The :FUNCtion:WINDow command allows selection of four windows for the FFT function. The operation of the :FUNCtion:WINDow command is identical to the `Window` softkey. Refer to "Math Functions" in Chapter 2 for additional information.

FFT operation assumes the time record repeats. Unless there is an integer number of cycles of the sampled waveform in the record, a discontinuity is created at the end of the record. This introduces additional frequency components into the spectrum about the actual peaks. This is referred to as leakage. In order to minimize leakage, windows that approach zero smoothly at the beginning and end of the record are employed as filters to the FFTs. The different windows are useful for various types of input signals.

The selection of the window is dependent on the type of input signal:

RECTangular window	Useful for transient signals and signals where there are an integral number of cycles in the time record.
HANNing window	Useful for frequency resolution and general purpose use. It is good for resolving two frequencies that are close together or for making frequency measurements.
FLATtop window	Best for making accurate amplitude measurements of frequency peaks.
EXPonent window	Best for transient analysis.

Command Syntax: :FUNCtion2:WINDow {RECTangular|HANNing|FLATtop|EXPonent}

Example: OUTPUT 707;":FUNC2:WIND RECT"

:MEASure:ALL

The :MEASure:ALL query makes a set of measurements on the displayed signal and buffers the measurement results for output over the interface. To make a measurement, the portion of the waveform required for the measurement must be displayed. Time measurements are made on the first (left-most) displayed edges of the waveforms. To obtain the most accurate measurement possible, use proper horizontal scaling. Refer to the individual commands for information on how the measurements are made and how the measurement results are returned.

The currently selected measurement type, either Eye:NRZ or General Waveform, determines which measurements are executed. Unsupported measurements will return the value +9.9E+37. Refer to "Measure Commands" in Chapter 6 for a list of the measurements supported by each measurement type.

All amplitude results are returned as either voltage or wattage values, depending on the vertical scale setting of channel 2.

Query Syntax:	:MEASure:ALL?
Returned Format:	<FREQ result>, <PERIOD result>, <PWIDTH result>, <NWIDTH result>, <RISE result>, <FALL result>, <PK-PK result>, <DUTY CYCLE result>, <RMS result>, <MAX result>, <MIN result>, <TOP result>, <BASE result>, <AVG result>, <AMPLIT result>, <OVERSHOOT result>, <PRESHOOT result>, <EXTINCTION RATIO result>, <ONE LEVEL result>, <ZERO LEVEL result>, <BIT RATE result>, <BIT PERIOD result>, <EYE HEIGHT result>, <JITTER RMS result>, <JITTER PK-PK result>, <NOISE RMS result>, <NOISE PK-PK result>, <EYE CROSSING % result>, <DCD result>, <POWER METER result>, <NL>
Where:	<result> ::= individual measurement results |exponential–NR3 format|
Example:	DIM Measure |1:30| OUTPUT 707;":MEASURE:ALL?" ENTER 707;Measure|*|

:MEASure:ASAMples

The :MEASure:ASAMples command is used to set the number of amplitude samples taken by the instrument when making an eye diagram voltage or power measurement.

The :ASAMples query returns the number of samples the instrument takes when making an eye diagram voltage or power measurement.

The number of samples can range from 200 to 10000. The number of samples presents a tradeoff between accuracy and speed of measurements, the lower the number, the faster the measurement. The defined number of amplitude samples is also used in the histogram algorithms when a mask or template is scaled to a waveform.

Command Syntax:	:MEASure:ASAMples <value><NL>
Where:	<value> :: = number of samples for the instrument to take when making an eye diagram amplitude measurement.
Example:	OUTPUT 707;":MEASure:ASAMples 1000"
Query Syntax:	:MEASure:ASAMples?
Returned Format:	<value><NL>
Where:	<value> :: = Number of samples the instrument takes when making an eye diagram amplitude measurement (exponential–NR3 format)
Example:	OUTPUT 707;":MEASure:ASAMples?" ENTER 707;Rsp PRINT Rsp

:MEASure:BPERiod

The :MEASure:BPERiod command puts the instrument into continuous measurement mode and measures the bit period of the input signal. This measurement is available when the instrument is in Eye:NRZ Waveform measurement mode.

The :BPERiod query returns the bit period of the input signal.

Command Syntax:	:MEASure:BPERiod
Query Syntax:	:MEASure:BPERiod?
Returned Format:	<value><NL>
Where:	<value> :: = bit period of the input signal |exponential-NR3 format|
Example:	OUTPUT 707;":MEAS:BPER?" ENTER 707;Rsp PRINT Rsp

:MEASure:BRATe

The :MEASure:BRATe command puts the instrument into continuous measurement mode and measures the bit rate for the signal. This measurement is available when the instrument is in Eye:NRZ Waveform measurement mode.

The :BRATe query returns the bit rate of the input signal.

Command Syntax:	:MEASure:BRATe
Query Syntax:	:MEASure:BRATe?
Returned Format:	<value><NL>
Where:	<value> :: = bit rate of the input signal |exponential-NR3 format|
Example:	OUTPUT 707;":MEAS:BRAT?" ENTER 707;Rsp PRINT Rsp

:MEASure:DCD

The :MEASure:DCD command puts the instrument into continuous measurement mode and measures the duty cycle distortion of the input signal. This measurement is available when the instrument is in Eye:NRZ Waveform measurement mode.

The :DCD query returns the duty cycle distortion of the input signal.

Command Syntax:	:MEASure:DCD
Query Syntax:	:MEASure:DCD?
Returned Format:	<value><NL>
Where:	<value> :: = the duty cycle distortion of the input signal |exponential–NR3 format|
Example:	OUTPUT 707;":MEAS:DCD?" ENTER 707;Rsp PRINT Rsp

:MEASure:DEFine DELay

The :MEASure:DEFine DELay command is used to define the starting and stopping delay measurement parameters.

The DELay query outputs the delay measurement start slope and edge (channel 1) and stop slope and edge (channel 2).

The sign of <edgeN> indicates a rising (+ or blank) or falling (−) edge. N is the selected edge number (1 to 5). The measurement is made with respect to the 50% amplitude level.

The operation of the :MEASure:DEFine DELay command is identical to the `Define Delay` softkey. To make delay measurements automatically, refer to Chapter 2 of the *HP 83475B Lightwave Communications Analyzer User's Guide*.

Command Syntax:	:MEASure:DEFine DELay,<edge1>,<edge2>
Example:	OUTPUT 707;":MEAS:DEF DEL,+1,-3"
Query Syntax:	:MEASure:DEFine DELay?
Returned Format:	<edge1>,<edge2> <NL>
Where:	<edgeN> ::= Selected edge number 1 to 5, for channel 1 and 2. The sign of <edgeN> indicates a rising (+ or blank) or falling (−) edge.

:MEASure:DELay

The :MEASure:DELay command puts the instrument into continuous measurement mode and starts a delay measurement. The delay measurement parameters are selected by using the :MEASURE:DEFine command. This measurement is available when the instrument is in General Waveform measurement mode.

The :DELay query returns the time delay between channels 1 and 2.

Command Syntax:	:MEASure:DELay
Query Syntax:	:MEASure:DELay?
Returned Format:	<value><NL>
Where:	<value> :: = time delay between channels 1 and 2.
Example:	OUTPUT 707;":MEAS:DEL?" ENTER 707;Rsp PRINT Rsp

:MEASure:DUTYcycle

The :MEASure:DUTYcycle command places the instrument in the continuous measurement mode and starts the duty cycle measurement. This measurement is available when the instrument is in General Waveform measurement mode.

The :DUTYcycle query measures and outputs the duty cycle of the signal specified by the :SOURce command. The signal must be displayed for the measurement to be made. The value returned for duty cycle is the ratio of the positive pulse width to the period.

The positive pulse width and the period of the specified signal are measured, then the duty cycle is calculated with the following formula:

$$duty\ cycle\ = +pulse\ width/period$$

Command Syntax:	:MEASure:DUTYcycle
Example:	OUTPUT 707;":MEASURE:DUTYCYCLE"
Query Syntax:	:MEASure:DUTYcycle?
Returned Format:	<value><NL>
Where:	<value> :: = ratio of positive pulse width to period [exponential-NR3 format]
Example:	OUTPUT 707;":MEASURE:DUTYCYCLE?" ENTER 707;Dc PRINT Dc

:MEASure:ECRossing

The :MEASure:ECRossing command puts the instrument into continuous measurement mode, and measures the eye crossing amplitude as a percentage of the "1" and "0" level amplitude difference of the eye diagram. This measurement is available when the instrument is in Eye:NRZ Waveform measurement mode.

The :ECRossing query returns the eye crossing amplitude percentage of the input signal.

Command Syntax:	:MEASure:ECRossing
Query Syntax:	:MEASure:ECRossing?
Returned Format:	<value><NL>
Where:	<value> ::= the eye crossing amplitude percentage of the input signal |exponential–NR3 format|
Example:	OUTPUT 707;":MEAS:ECR?" ENTER 707;Rsp PRINT Rsp

:MEASure:EHEIght

The :MEASure:EHEIght command puts the instrument into continuous measurement mode and measures the eye height of the input signal as a percentage of the "1" and "0" level eye diagram amplitude difference. This measurement is available when the instrument is in Eye:NRZ Waveform measurement mode.

The :EHEIght query returns the eye height of the input signal.

Command Syntax:	:MEASure:EHEIght
Query Syntax:	:MEASure:EHEIght?
Returned Format:	<value><NL>
Where:	<value> :: = the eye height of the input signal |exponential-NR3 format|
Example:	OUTPUT 707;":MEAS:EHEI?" ENTER 707;Rsp PRINT Rsp

:MEASure:ERATio

The :MEASure:ERATio command puts the instrument into continuous measurement mode and measures the extinction ratio of the input signal. This measurement is available when the instrument is in Eye:NRZ Waveform measurement mode.

The :ERATio query returns the extinction ratio of the input signal.

Command Syntax:	:MEASure:ERATio
Query Syntax:	:MEASure:ERATio?
Returned Format:	<value><NL>
Where:	<value> :: = the extinction ratio of the input signal |exponential-NR3 format|
Example:	OUTPUT 707;":MEAS:ERAT?" ENTER 707;Rsp PRINT Rsp

:MEASure:ERUNits

The :MEASure:ERUNits command is used to set the display units for the Extinction Ratio measurement.

The :ERUNits query returns the units being used to display the Extinction Ratio measurement.

Command Syntax:	:MEASure:ERUNits { LINEAR | dB | PERCENT }
Example:	OUTPUT 707;":MEASure:ERUNits dB"
Query Syntax:	:MEASure:ERUNits?
Returned Format:	{ LINEAR | dB | PERCENT }<NL>
Example:	DIM Rsp$[50] OUTPUT 707;":MEASure:ERUNits?" ENTER 707;Rsp$ PRINT Rsp$

:MEASure:EWMax

The :MEASure:EWMax command sets the upper limit of the vertical histogram window for the percentage of the eye bit period in which amplitude measurements are to be taken.

The :EWMax query returns the percentage for the stop point of the window which will be used for Eye:NRZ amplitude measurements.

The 0% point for the eye diagram is the first complete left-most crossing. The 100% point for the eye is the next crossing to the right. Valid eye window settings are between 0% and 100%. The minimum setting cannot be set below the minimum window limit.

Command Syntax:	:MEASure:EWMax <value>
Where:	<value> :: = Percentage of stop location of the window used for amplitude eye diagram measurements.
Example:	OUTPUT 707; ":MEASure:EWMax 40"
Query Syntax:	:MEASure:EWMax?
Returned Format:	<value> <NL>
Where:	<value> :: = Percentage of stop location of the window used for amplitude eye diagram measurements |exponential–NR3 format|
Example:	OUTPUT 707;":MEASure:EWMax?" ENTER 707;Lftpt PRINT Lftpt

:MEASure:EWMin

The :MEASure:EWMin command sets the lower limit of the vertical histogram window for the percentage of the eye bit period in which amplitude measurements are to be taken.

The :EWMin query returns the percentage for the start point of the window which will be used for Eye:NRZ amplitude measurements.

The 0% point for the eye diagram is the first complete left-most crossing. The 100% point for the eye is the next crossing to the right. Valid eye window settings are between 0% and 100%. The minimum setting cannot be set above the maximum window limit.

Command Syntax:	:MEASure:EWMin <value>
Where:	<value> ::= Percentage of start location of the window used for amplitude eye diagram measurements.
Example:	OUTPUT 707; ":MEASure:EWMin 40"
Query Syntax:	:MEASure:EWMin?
Returned Format:	<value> <NL>
Where:	<value> ::= Percentage of start location of the window used for amplitude eye diagram measurements [exponential–NR3 format]
Example:	OUTPUT 707;":MEASure:EWMin?" ENTER 707;Lftpt PRINT Lftpt

:MEASure:FALLtime

The :MEASURE:FALLTIME command places the instrument in the continuous measurement mode and starts a fall time measurement. This measurement is available when the instrument is in either General Waveform or Eye:NRZ Waveform measurement mode.

When the instrument is in the General Waveform measurement mode, the :FALLtime query measures and outputs the fall time of the first displayed falling (negative-going) edge. For highest measurement accuracy, set the sweep speed as fast as possible while leaving the falling edge of the waveform on the display. The fall time is determined by measuring the time at the upper threshold of the falling edge, then measuring the time at the lower threshold of the falling edge and calculating the fall time using the following formula (assuming 90% and 10% thresholds):

fall time = time at 10% level − time at 90% level

When the instrument is in Eye:NRZ measurement mode, the :FALLtime query measures and outputs the fall time of the first displayed eye diagram falling edge. At least one complete eye diagram must be displayed to make this measurement.

Command Syntax:	:MEASure:FALLtime
Example:	OUTPUT 707;":MEAS:FALL"
Query Syntax:	:MEASure:FALLtime?
Returned Format:	<value><NL>
Where:	<value> ::= Time in seconds between lower and upper amplitude levels [exponential–NR3 format]
Example:	OUTPUT 707;":MEASURE:FALLTIME?" ENTER 707;Fall PRINT Fall

:MEASure:FREQuency

The :MEASure:FREQuency command places the instrument in the continuous measurement mode and starts a frequency measurement. This measurement is available when the instrument is in General Waveform measurement mode.

The :FREQuency query measures and outputs the frequency of the first complete cycle on the screen. This command uses the 50% levels between defined thresholds.

The algorithm is:

If first edge on screen is rising, then:

$$frequency = \frac{1}{(time\ at\ second\ rising\ edge\ - time\ at\ first\ rising\ edge)}$$

Else:

$$frequency = \frac{1}{(time\ at\ second\ falling\ edge\ - time\ at\ first\ falling\ edge)}$$

Command Syntax:	:MEASure:FREQuency
Example:	OUTPUT 707;":MEASURE:FREQ"
Query Syntax:	:MEASURE:FREQuency?
Returned Format:	<value><NL>
Where:	<value> ::= frequency in hertz [exponential-NR3 format]
Example:	OUTPUT 707;":MEASURE:FREQUENCY?" ENTER 707;Frq PRINT Frq

:MEASure:JPP

The :MEASure:JPP command puts the instrument into continuous measurement mode, and measures the peak-to-peak jitter at the eye diagram crossing point. This measurement is available when the instrument is in Eye:NRZ Waveform measurement mode.

The :JPP query returns the peak-to-peak jitter of the input signal.

Command Syntax:	:MEASure:JPP
Query Syntax:	:MEASure:JPP?
Returned Format:	<value><NL>
Where:	<value> ::= The peak-to-peak jitter of the input signal.
Example:	OUTPUT 707;":MEASURE:JPP?" ENTER 707;Rsp PRINT Rsp

:MEASure:JRMS

The :MEASure:JRMS command puts the instrument into continuous measurement mode and measures the rms jitter at the eye diagram crossing point. This measurement is available when the instrument is in Eye:NRZ Waveform measurement mode.

The :JRMS query returns the rms jitter of the input signal.

Command Syntax:	:MEASure:JRMS
Query Syntax:	:MEASure:JRMS?
Returned Format:	<value><NL>
Where:	<value> ::= the rms jitter of the input signal
Example:	OUTPUT 707;":MEASURE:JRMS?" ENTER 707;Rsp PRINT Rsp

:MEASure:LOWer

The :MEASure:LOWer command sets the lower measurement threshold percentage. This measurement function is available when the instrument is in either General Waveform or Eye:NRZ measurement mode. This value and the UPPER value are used when THResholds are set to PERCent and are used for measurements such as rise time and fall time. Refer to the :MEASure:MTYPe command and Chapter 2 of the *HP 83475B Lightwave Communications Analyzer User's Guide* for more information on setting measurement threshholds.

The :MEASure:LOWer query outputs the current lower threshold percentage.

The operation of the :MEASure:LOWer command is identical to the `Lower` softkey. Refer to Chapter 2 of the *HP 83475B Lightwave Communications Analyzer User's Guide* for additional information.

Command Syntax:	:MEASure:LOWer <percent>
Example:	OUTPUT 707;":MEAS:LOW 20"
Query Syntax:	:MEASure:LOWer?
Returned Format:	<percent> <NL>
Where:	<percent> ::= user defined threshold in volts |exponential-NR3 format|

:MEASure:MASK :DEFine:BFIT

The MEASure:MASK:DEFine:BFIT command and query sets the best fit operation of mask testing.

Command Syntax:	:MEASure:MASK:DEFine:BFIT { ON | OFF }
Example:	OUTPUT 707;":MEASure:MASK:DEFine:BFIT ON"
Query Syntax:	:MEASure:MASK:DEFine:BFIT?
Returned Format:	<state><NL>
Where:	<state> ::= { ON | OFF }
Example:	OUTPUT 707;":MEASure:MASK:DEF:BFIT?" ENTER 707;Rsp$ PRINT Rsp$

:MEASure:MASK :DEFine:DATA

The MEASure:MASK:DEFine:DATA command is used for transferring mask definition block data to and from the analyzer. The command is sent prior to sending the mask definition data block, in the form of an ASCII file.

The :DATA query is used to extract a mask definition data block from the analyzer.

Command Syntax:	:MEASure:MASK:DEFine:DATA { UDM1 | UDM2 } <file size>
Example:	OUTPUT 707;":MEASure:MASK:DEFine:DATA UDM1"
Where:	<file size> ::= size of the file in bytes. Binary value, i.e., #4xxxx.
Query Syntax:	:MEASure:MASK:DEFine:DATA? <selected mask>
Returned Format:	<mask definition block><NL>
Where:	<selected mask> ::= { OC1 | STM0 | OC3 | STM1 | STS1 | STS3 | FC133 | FC266 | DS1 | DS1C | DS2 | DS3 | E1 | E2 | E3 | E4ONE | E4ZERO | E1_75 | E1_120 | STS1_EYE | STS3_IFC | STS3_TX | UDM1 | UDM2 }
Example:	Refer to Chapter 8, Programming Examples.

:MEASure:MASK :DEFine:INVert

The MEASure:MASK:DEFine:INVert command and query sets the inversion operation of fixed scale masks. This will cause a warning if invoked on scaled masks.

Command Syntax:	:MEASure:MASK:DEFine:INVert { ON | OFF }
Example:	OUTPUT 707;":MEASure:MASK:DEFine:INVert ON"
Query Syntax:	:MEASure:MASK:DEFine:INVert?
Returned Format:	<state><NL>
Where:	<state> ::= { ON | OFF }
Example:	OUTPUT 707;":MEASure:MASK:DEF:INVert?" ENTER 707;Rsp$ PRINT Rsp$

:MEASure:MASK
:DEFine:MARGin

The MEASure:MASK:DEFine:MARGin command sets the amount of margin to apply to a mask. The margin will be applied on the next placement of the mask. If the instrument is in continuous update mode, the margin will be added (or subtracted) on the next placement.

The :MARGin query returns the current mask margin setting.

Command Syntax:	:MEASure:MASK:DEFine:MARGin <value>
Where:	<value> ::= an integer from −100 to +100, representing the percentage of margin to apply to the mask.
Example:	OUTPUT 707;":MEASure:MASK:DEFine:MARG 20"
Query Syntax:	:MEASure:MASK:DEFine:MARGin?
Returned Format:	<value><NL>
Where:	<value> ::= the current mask margin setting [exponential–NR3 format]
Example:	OUTPUT 707;":MEASure:MASK:DEF:MARG?" ENTER 707;Rsp PRINT Rsp

:MEASure:MASK
:DEFine:MHTS

The :MEASure:MASK:DEFine:MHTS command sets the maximum number of hits allowed to pass the mask conformance test. A failure will register if the total number of hits is greater than this value. A pass will occur if the total number of hits is less than or equal to this value.

The :MHTS query returns the current maximum number of hits used for the conformance test.

Command Syntax:	:MEASure:MASK:DEFine:MHTS <value><NL>
Where:	<value> ::= the maximum number of hits allowed, for the conformance test to pass.
Example:	OUTPUT 707;":MEAS:MASK:DEFine:MHTS 0"
Query Syntax:	:MEASure:MASK:DEFine:MHTS?
Returned Format:	<value><NL>
Where:	<value> ::= the current setting of the maximum number of hits allowed for the conformance test to pass |exponential–NR3 format|
Example:	OUTPUT 707;":MEAS:MASK:DEF:MHTS?" ENTER 707;Rsp PRINT Rsp

:MEASure:MASK
:DEFine:OFAil

The :MEASure:MASK:DEFine:OFAil command sets the action to be performed when the instrument has determined that a conformance test has been failed.

The :OFAil query returns the current action that will be performed when a conformance test is failed.

The instrument can perform three possible actions based on a failed conformance test.

- It can STOP and hold the data on the screen.
- It can CONTinue collecting and displaying data.
- It can PRINt the results to a printer connected to the instrument.

In the print mode, the instrument is stopped until the printout is complete, then allowed to continue.

Command Syntax:	:MEASure:MASK:DEFine:OFAil { STOP | CONTinue | PRINt }
Example:	OUTPUT 707;":MEAS:MASK:DEFine:OFA PRIN"
Query Syntax:	:MEASure:MASK:DEFine:OFAil?
Returned Format:	{ STOP | CONTinue | PRINt }<NL>
Example:	DIM Rsp$[50] OUTPUT 707;":MEAS:MASK:DEF:OFA?" ENTER 707;Rsp$ PRINT Rsp$

:MEASure:MASK
:DEFine:OPASs

The :MEASure:MASK:DEFine:OPASs command sets the action to be performed when the instrument has determined that a conformance test has been passed.

The :OPASs query returns the current action that will be performed when a conformance test is passed.

The instrument can perform three possible actions based on a passed conformance test.

- It can STOP and hold the data on the screen.
- It can CONTinue collecting and displaying data.
- It can PRINt the results to a printer connected to the instrument.

In the print mode, the instrument is stopped until the printout is complete, then allowed to continue.

Command Syntax:	:MEASure:MASK:DEFine:OPASs { STOP | CONTinue | PRINt }
Example:	OUTPUT 707;":MEAS:MASK:DEFINE:OPAS CONT"
Query Syntax:	:MEASure:MASK:DEFine:OPASs?
Returned Format:	{ STOP | CONTinue | PRINt }<NL>
Example:	DIM Rsp$[50] OUTPUT 707;":MEAS:MASK:DEFine:OPAS?" ENTER 707;Rsp$ PRINT Rsp$

:MEASure:MASK :DEFine:RETest

The MEASure:MASK:DEFine:RETest command and query sets the retest operation of conformance testing. If set to auto, the complete test is performed every time. If set to manual, then hits will be counted only when a MEAS:MASK:TEST? query is executed. This will function as long as only trigger delay has been adjusted since the last automatic mask placement.

Command Syntax:	:MEASure:MASK:DEFine:RETest { AUTO | MAN }
Example:	OUTPUT 707;":MEASure:MASK:DEFine:RETest AUTO"
Query Syntax:	:MEASure:MASK:DEFine:RETest?
Returned Format:	<state><NL>
Where:	<state> ::= { AUTO | MAN }
Example:	OUTPUT 707;":MEASure:MASK:DEF:RETest" ENTER 707;Rsp$ PRINT Rsp$

:MEASure:MASK :DEFine:TIME

The :MEASure:MASK:DEFine:TIME command sets the time in seconds for the conformance test to run.

The :TIME query returns the time in seconds that the conformance test is set to run.

Command Syntax:	:MEASure:MASK:DEFine:TIME <conf_time>
Where:	<conf_time> ::= time in seconds for the conformance test to run. The minimum is 1 second, the maximum is 32 seconds.
Example:	OUTPUT 707;":MEASure:MASK:DEFine:TIME 2S"
Query Syntax:	:MEASure:MASK:DEFine:TIME?
Returned Format:	<conf_time><NL>
Where:	<conf_time> ::= the time in seconds that the conformance test is set to run.
Example:	OUTPUT 707;":MEASure:MASK:DEFine:TIME?" ENTER 707;Rsp PRINT Rsp

:MEASure:MASK :IDENtify

The MEASure:MASK:IDENtify query returns the identification string for the selected mask. Any mask selection is allowed.

Query Syntax:	:MEASure:MASK:IDENtify? <select mask>
Returned Format:	<select mask><NL>
Where:	<select mask> ::= { OC1 | STM0 | OC3 | STM1 | STS1 | STS3 | FC133 | FC266 | DS1 | DS1C | DS2 | DS3 | E1 | E2 | E3 | E4ONE | E4ZERO | E1_75 | E1_120 | STS1_EYE | STS3_IFC | STS3_TX | UDM1 | UDM2 }
Example:	OUTPUT 707;":MEASure:MASK:IDEN?" ENTER 707;Rsp$ PRINT Rsp$

:MEASure:MASK:SHOW

The :MEASure:MASK:SHOW sets the run mode operation of the mask measurements to Off, Only Once, or Continuous display and placement. The Only Once mode scales and places the mask based on a single data acquisition. The Continuous mode rescales the mask whenever a change in the waveform is detected.

The :SHOW query returns the current setting.

Command Syntax:	:MEASure:MASK:SHOW { OFF | ONLYonce | CONTinuous }
Query Syntax:	:MEASure:MASK:SHOW?
Returned Format:	<state><NL>
Where:	<state> ::= { OFF | ONLY | CONT }
Example:	DIM Rsp$[50] OUTPUT 707;":MEASure:MASK:SHOW?" ENTER 707;Rsp$ PRINT Rsp$

:MEASure:MASK:STESt

The :MEASure:MASK:STESt command is used to select a preprogrammed standard mask test in the instrument. If an eye diagram mask is selected, the measurement type is automatically set to Eye:NRZ. If a pulse mask is selected, the measurement type is automatically set to General Waveform.

The :STESt query returns the currently selected preprogrammed standard mask test.

Command Syntax:	:MEASure:MASK:STESt { OC1 | STM0 | OC3 | STM1 | STS1 | STS3 | FC133 | FC266 | DS1 | DS1C | DS2 | DS3 | E1 | E2 | E3 | E4ONE | E4ZERO | E1_75 | E1_120 | STS1_EYE | STS3_IFC | STS3_TX | UDM1 | UDM2 }
Example:	OUTPUT 707;":MEAS:MASK:STES STS1"
Query Syntax:	:MEASure:MASK:STESt?
Returned Format:	{ OC1 | STM0 | OC3 | STM1 | STS1 | STS3 | FC133 | FC266 | DS1 | DS1C | DS2 | DS3 | E1 | E2 | E3 | E4ONE | E4ZERO | E1_75 | E1_120 | STS1_EYE | STS3_IFC | STS3_TX | UDM1 | UDM2 }
Example:	DIM Rsp$[50] OUTPUT 707;":MEAS:MASK:STES?" ENTER 707;Rsp$ PRINT Rsp$

:MEASure:MASK:TEST

The :MEASure:MASK:TEST query performs the conformance test, and responds with the results.

Query Syntax:	:MEASure:MASK:TEST?
Returned Format:	{PASSED | FAILED}, <TOTHits result>, <STDHits result>, <MARHits result><NL>
Where:	<result> ::= individual integral measurement results
Example:	DIM Rsp$[50] OUTPUT 707;":MEASure:MASK:TEST?" ENTER 707;Rsp$ PRINT Rsp$

:MEASure:MTYPe

The :MEASure:MTYPe command is used to set the measurement type which selects algorithms for the automatic measurements. The waveform type is checked during all continuous, and single shot measurement cycles, and only certain measurements are allowed while in General Waveform or Eye:NRZ measurement modes. Check each command to determine if the measurement is made for the measurement type selected. Refer to "Measure Commands" in Chapter 6 for more information. If you select a measurement that is not supported for a waveform type, the measurement will return +9.9e+37.

Command Syntax:	:MEASure:MTYPe { GENERAL | EYENRZ }
Query Syntax:	:MEASure:MTYPe?
Returned Format:	{ GENERAL | EYENRZ } <NL>
Example:	DIM Rsp$[50] OUTPUT 707;":MEAS:MTYP?" ENTER 707;Rsp$ PRINT Rsp$

:MEASure:NPP

The :MEASure:NPP command puts the instrument into continuous measurement mode and measures the peak-to-peak noise of the input signal. This measurement is available when the instrument is in Eye:NRZ Waveform measurement mode.

The :NPP query returns the peak-to-peak noise of the eye diagram.

Command Syntax:	:MEASure:NPP
Query Syntax:	:MEASure:NPP?
Returned Format:	<value><NL>
Where:	<value> ::= the peak-to-peak noise of the input signal.
Example:	OUTPUT 707;":MEAS:NPP?" ENTER 707;Rsp PRINT Rsp

:MEASure:NRMS

The :MEASure:NRMS command puts the instrument into continuous measurement mode and measures the rms noise of the eye diagram. This measurement is available when the instrument is in Eye:NRZ measurement mode.

The :NRMS query returns the rms noise of the input signal.

Command Syntax:	:MEASure:NRMS
Query Syntax:	:MEASure:NRMS?
Returned Format:	<value><NL>
Where:	<value> ::= the rms noise of the input signal.
Example:	OUTPUT 707;":MEAS:NRMS?" ENTER 707;Rsp PRINT Rsp

:MEASure:NWIDth

The :MEASure:NWIDth command places the instrument in the continuous measurement mode and starts an NWIDTH measurement.

The :NWIDth query measures and outputs the width of the first negative pulse on the screen using the 50% levels. The algorithm is:

If the first edge on screen is rising, then:

width = (time at second rising edge − time at first falling edge)

Else:

width = (time at first rising edge − time at first falling edge)

Command Syntax:	:MEASure:NWIDth
Example:	OUTPUT 707;":MEAS:NWIDTH"
Query Syntax:	:MEASure:NWIDth?
Returned Format:	<value><NL>
Where:	<value> ::= negative pulse width in seconds |exponential-NR3 format|
Example:	OUTPUT 707;":MEASURE:NWIDTH?" ENTER 707;Nwd PRINT Nwd

:MEASure:OLEVel

The :MEASure:OLEVel command puts the instrument into continuous measurement mode and measures the one level of the eye diagram. This measurement is available when the instrument is in Eye:NRZ measurement mode.

The :OLEVel query returns the one level of the eye diagram.

Command Syntax:	:MEASure:OLEVel
Query Syntax:	:MEASure:OLEVel?
Returned Format:	<value><NL>
Where:	<value> ::= the one level of the eye diagram |exponential–NR3 format|
Example:	OUTPUT 707;":MEAS:OLEV?" ENTER 707;Rsp PRINT Rsp

:MEASure:OPMeter

The :MEASure:OPMeter command turns the optical power meter on and off. When the power meter is on, the instrument is placed in continuous measurement mode.

The :OPMeter query returns the current state of the power meter, ON or OFF. The power meter should be activated prior to sending the :OPMeter query.

Command Syntax:	:MEASure:OPMeter {ON | OFF}
Example:	OUTPUT 707;":MEASure:OPMeter ON"
Query Syntax:	:MEASure:OPMeter?
Returned Format:	{ ON | OFF } <NL>
Where:	<value> ::= the current state of the power meter.
Example:	OUTPUT 707;":MEASure:OPMeter?" ENTER 707;Rsp PRINT Rsp

:MEASure:OVERshoot

The :MEASure:OVERshoot command puts the instrument into continuous measurement mode and measures the overshoot of the eye diagram. This measurement is available when the instrument is in either Eye:NRZ or General Waveform measurement mode.

The :OVERshoot query returns the overshoot of the input signal.

Three different amplitude measurements are made and then overshoot, for a general voltage waveform, is calculated using the following formula:

$$overshoot = 100\left(\frac{Vmax - Vtop}{Vtop - Vbase}\right)$$

Command Syntax:	:MEASure:OVERshoot
Example:	OUTPUT 707;":MEAS:OVER"
Query Syntax:	:MEASure:OVERshoot?
Returned Format:	<value> <NL>
Where:	<value> ::= overshoot of the input signal in percent [exponential-NR3 format]
Example:	OUTPUT 707;":MEAS:OVER?" ENTER 707;Rsp PRINT Rsp

:MEASure:PAMPlitude

The :MEASure:PAMPlitude command places the instrument into continuous measurement mode and measures the amplitude of the input signal. This measurement can be made when the instrument is in either Eye:NRZ or General Waveform measurement mode.

The :PAMPlitude query measures and outputs the power amplitude of the first waveform on the screen. The method used to determine amplitude is to measure two levels and then calculate amplitude as follows:

$$amplitude = Ptop - Pbase (or\ Pone - Pzero)$$

Command Syntax:	:MEASure:PAMPlitude
Example:	OUTPUT 707;":MEAS:PAMP"
Query Syntax:	:MEASure:PAMPlitude?
Returned Format:	<value> <NL>
Where:	<value> :: = amplitude of selected waveform in watts [exponential–NR3 format]
Example:	OUTPUT 707;":MEAS:PAMP?" ENTER 707;Rsp PRINT Rsp

:MEASure:PAVerage

The :MEASure:PAVerage command places the instrument in the continuous measurement mode and starts a Paverage measurement. This measurement is available when the instrument is in General Waveform measurement mode.

The :PAVerage query measures the average amplitude of the first cycle of the displayed power signal. If a complete cycle is not present, the analyzer averages all data points.

Command Syntax:	:MEASure:PAVerage
Example:	OUTPUT 707;":MEAS:PAV"
Query Syntax:	:MEASure:PAVerage?
Returned Format:	<average_power><NL>
Where:	<average_power> ::= calculated average power level [exponential-NR3 format]
Example:	OUTPUT 707;":MEAS:PAV?" ENTER 707;Pavg PRINT Pavg

:MEASure:PBASe

The :MEASure:PBASe command places the instrument in the continuous measurement mode and starts a Pbase measurement. This measurement is available when the instrument is in General Waveform measurement mode.

The :PBASe query measures and outputs the power value at the base of the waveform. The base level of a pulse is normally not the same as the minimum value.

Command Syntax:	:MEASure:PBASe
Example:	OUTPUT 707;":MEAS:PBASE"
Query Syntax:	:MEASure:PBASe?
Returned Format:	<base_power><NL>
Where:	<base_power> ::= power at base of selected waveform |exponential-NR3 format|
Example:	OUTPUT 707;":MEASURE:PBASE?" ENTER 707;Pbase PRINT Pbase

:MEASure:PERiod

The :MEASure:PERiod command places the instrument in the continuous measurement mode and selects the period measurement. This measurement is available when the instrument is in General Waveform measurement mode.

The :PERiod query measures and outputs the period of the first complete cycle on the screen. The period is measured at the 50% amplitude level.

The algorithm for this measurement is:

If the first edge on screen is rising, then:

period = time at second rising edge − time at first rising edge

Else:

period = time at second falling edge − time at first falling edge

Command Syntax:	:MEASure:PERiod
Example:	OUTPUT 707;":MEAS:PERIOD"
Query Syntax:	:MEASure:PERiod?
Returned Format:	<value><NL>
Where:	<value> ::= waveform period in seconds |exponential-NR3 format|
Example:	OUTPUT 707;":MEASURE:PERIOD?" ENTER 707;Prd PRINT Prd

:MEASure:PHASe

The :MEASure:PHASe command puts the instrument into continuous measurement mode and measures the phase between channel 1 and 2. This measurement is available when the instrument is in General Waveform measurement mode.

The :PHASe query measures and outputs the phase between channels 1 and 2. The phase measurement is made from the first rising edge of the signal connected to input 1, to the first rising edge of the signal connected to input 2. The operation of the :MEASure:PHASe command is identical to the `Phase` softkey.

Phase is determined by measuring the delay and the period, then calculating phase using the following formula:

$$Phase = \left(\frac{delay}{period\ of\ input\ 1} \right) 360$$

Command Syntax:	:MEASure:PHASe
Example:	OUTPUT 707;":MEAS:PHAS"
Query Syntax:	:MEASure:PHASe?
Returned Format:	<value> <NL>
Where:	<value> ::= phase difference between channel 1 and channel 2 |exponential-NR3 format|
Example:	OUTPUT 707;":MEAS:PHAS?" ENTER 707;Rsp PRINT Rsp

:MEASure:PMAX

The :MEASure:PMAX command places the instrument in the continuous measurement mode and starts a Pmax measurement. This measurement is available when the instrument is in either Eye:NRZ or General Waveform measurement mode.

The :PMAX query measures and outputs the absolute maximum power present on the selected waveform.

Command Syntax:	:MEASure:PMAX
Example:	OUTPUT 707;":MEAS:PMAX"
Query Syntax:	:MEASure:PMAX?
Returned Format:	<value><NL>
Where:	<value> ::= maximum power of selected waveform [exponential-NR3 format]
Example:	OUTPUT 707;":MEASURE:PMAX?" ENTER 707;Pmax PRINT Pmax

:MEASure:PMET

The :PMET query returns the current optical power meter measurement. This measurement is available when the instrument is in either Eye:NRZ or General Waveform measurement mode. To ensure a valid reading, adequate time must be allowed for the power meter to settle. Sending the :MEASure:OPMeter ON command prior to sending the :PMET query will ensure a valid reading. If the query is sent prior to the :MEASure:OPMeter ON command, an error value of 9.9e37 will be returned.

Query Syntax:	:MEASure:PMET?
Returned Format:	<value><NL>
Where:	<value> ::= current optical power meter measurement in watts or dBm [exponential-NR3 format]
Example:	OUTPUT 707;":MEAS:PMET?" ENTER 707;Pmet PRINT Pmet

:MEASure:PMIN

The :MEASure:PMIN command places the instrument in the continuous measurement mode and starts a PMIN measurement. This measurement is available when the instrument is in either Eye:NRZ or General Waveform measurement mode.

The :PMIN query measures and outputs the absolute minimum power present on the selected waveform.

Command Syntax:	:MEASure:PMIN
Example:	OUTPUT 707;":MEAS:PMIN"
Query Syntax:	:MEASure:PMIN?
Returned Format:	<value><NL>
Where:	<value> ::= minimum power value of the selected waveform |exponential–NR3 format|
Example:	OUTPUT 707;":MEASURE:PMIN?" ENTER 707;Pmin PRINT Pmin

:MEASure:PPP

The :MEASure:PPP command places the instrument in the continuous measurement mode and starts a peak-to-peak power measurement. This measurement is available when the instrument is in General Waveform measurement mode.

The :PPP query measures the maximum and minimum power levels for the selected source, then calculates the peak-to-peak power and outputs that value. The peak-to-peak power (Ppp) is calculated with the following formula:

$$Ppp = Pmax - Pmin$$

Pmax and Pmin are the maximum and minimum power levels present on the selected source.

Command Syntax:	:MEASure:PPP
Example:	OUTPUT 707;":MEAS:PPP"
Query Syntax:	:MEASure:PPP?
Returned Format:	<value><NL>
Where:	<value> ::= peak-to-peak power |exponential–NR3 format|
Example:	OUTPUT 707;":MEAS:PPP?" ENTER 707;Ppp PRINT Ppp

:MEASure:PREShoot

The :MEASure:PREShoot command places the instrument in the continuous measurement mode and starts a preshoot measurement. The operation of the :MEASure:PREShoot command is identical to the `Preshoot` softkey. This measurement is available when the instrument is in the General Waveform measurement mode.

The :PREShoot query measures and outputs the preshoot of the first edge displayed on the screen.

Three different amplitude measurements are made and then preshoot is calculated using the following formula:

$$preshoot = 100\left(\frac{Base - Min}{Top - Base}\right)$$

Command Syntax:	:MEASure:PREShoot
Example:	OUTPUT 707;":MEAS:PRES"
Query Syntax:	:MEASure:PREShoot?
Returned Format:	<value> <NL>
Where:	<value> ::= preshoot of the selected waveform in percent |exponential–NR3 format|

:MEASure:PRMS

The :MEASure:PRMS command places the instrument in the continuous measurement mode and starts a dc rms power measurement. This measurement is available when the instrument is in General Waveform measurement mode.

The :PRMS query measures and outputs the dc rms power of the selected waveform. The dc rms power is measured on the first cycle of the displayed signal. If a complete cycle is not present, the analyzer computes the rms value on all displayed data points.

Command Syntax:	:MEASure:PRMS
Example:	OUTPUT 707;":MEAS:PRMS"
Query Syntax:	:MEASure:PRMS?
Returned Format:	<value><NL>
Where:	<value> ::= calculated dc rms power level |exponential-NR3 format|
Example:	OUTPUT 707;":MEASURE:PRMS?" ENTER 707;Prms PRINT Prms

:MEASure:PTIMe

The :PTIMe query returns the power at a specified time. The specified time must be on screen and is referenced to the trigger event.

Query Syntax:	:MEASure:PTIMe? <time>
Where:	<time> ::= displayed time from trigger in seconds
Returned Format:	<power><NL>
Where:	<power> ::= power at specified time |exponential-NR3 format|
Example:	OUTPUT 707;":MEASURE:PTIME? .001" ENTER 707;Ptm PRINT Ptm

:MEASure:PTOP

The :MEASure:PTOP command places the instrument in the continuous measurement mode and starts a Ptop measurement. This measurement is available when the instrument is in General Waveform measurement mode.

The :PTOP query returns the power at the top level of a waveform. The top level is the most prevalent high level of the waveform.

Command Syntax:	:MEASure:PTOP
Example:	OUTPUT 707;":MEASURE:PTOP"
Query Syntax:	:MEASure:PTOP?
Returned Format:	<value><NL>
Where:	<value> ::= power at the top of the waveform |exponential-NR3 format|
Example:	OUTPUT 707;":MEASURE:PTOP?" ENTER 707;Ptp PRINT Ptp

:MEASure:PUNIts

The :MEASure:PUNIts command set the internal power measurements display units.

The :PUNIts query returns the current power measurements display units.

Command Syntax:	:MEASure:PUNIts { WATTS | dBM }
Example:	OUTPUT 707;":MEASURE:PUNIts WATTS"
Query Syntax:	:MEASure:PUNIts?
Returned Format:	{ WATTS | dBM }<NL>
Example:	OUTPUT 707;":MEASURE:PUNIts?" ENTER 707;Rsp PRINT Rsp

:MEASure:PWDELta

The :PWDELta query outputs the power difference between PMarker 1 and PMarker 2. No measurement is made when the :PWDELta query is received by the analyzer. The delta power value that is output is the current value. This is the same value as the front panel delta P value. This measurement is available when the instrument is in either Eye:NRZ or General Waveform measurement mode.

PWDELTA = Power at PMarker 2 − Power at PMarker 1

Query Syntax:	:MEASure:PWDELta?
Returned Format:	<value><NL>
Where:	<value> ::= delta P value in watts [exponential–NR3 format]
Example:	OUTPUT 707;":MEAS:PWDELTA?" ENTER 707;Pwdelta PRINT Pwdelta

:MEASure:PWIDth

The :MEASure:PWIDth command places the instrument in the continuous measurement mode and starts the Pwidth measurement.

The :PWIDth query measures and outputs the width of the first displayed positive pulse. Pulse width is measured at the 50% amplitude level.

The algorithm for this measurement is:

If the first edge on screen is falling, then:

width = time at second falling edge − time at first rising edge

Else:

width = time at first falling edge − time at first rising edge

Command Syntax:	:MEASure:PWIDth
Example:	OUTPUT 707;":MEAS:PWIDTH"
Query Syntax:	:MEASure:PWIDth?
Returned Format:	<value><NL>
Where:	<value> ::= width of positive pulse in seconds [exponential-NR3 format]
Example:	OUTPUT 707;":MEASURE:PWIDTH?" ENTER 707;Pwd PRINT Pwd

:MEASure:PWSTArt

The :MEASure:PWSTArt command moves PMarker 1 to the specified power corresponding to the current source. The source can be selected by the :MEASure:SOURce command.

The :PWSTArt query returns the current power level of PMarker 1.

Command Syntax:	:MEASure:PWSTArt <power> [W, mW, uW or V, mW, uW]
Where:	<power> ::= power value for Pmarker 1
Example:	OUTPUT 707;":MEAS:PWSTA 10uW"
Query Syntax:	:MEASure:PWSTArt?
Returned Format:	<value><NL>
Where:	<value> ::= power at PMarker 1 [exponential-NR3 format]
Example:	OUTPUT 707;":MEASURE:PWSTART?" ENTER 707;Pwst PRINT Pwst

:MEASure:PWSTop

The :MEASure:PWSTop command moves PMarker 2 to the specified power corresponding to the current source. The source can be selected by the :MEASure:SOURce command.

The :PWSTop query returns the current power level of PMarker 2.

Command Syntax:	:MEASure:PWSTop <power> [W, mW, uW or V, mW, uW]
Where:	<voltage> ::= power value for Pmarker 2
Example:	OUTPUT 707;":MEAS:PWST -100uW"
Query Syntax:	:MEASure:PWSTop?
Returned Format:	<value><NL>
Where:	<value> ::= power at PMarker 2 [exponential-NR3 format]
Example:	OUTPUT 707;":MEASURE:PWSTOP?" ENTER 707;Pwst PRINT Pwst

:MEASure:RISetime

The :MEASure:RISetime command places the instrument in the continuous measurement mode and starts a rise time measurement. This measurement is available when the instrument is in either Eye:NRZ or General Waveform measurement mode.

When the instrument is in General Waveform measurement mode, the :RISetime query measures and outputs the rise time of the first displayed rising (positive-going) edge. For maximum measurement accuracy set the sweep speed as fast as possible while leaving the leading edge of the waveform on the display. The rise time is determined by measuring the time at the lower threshold level of the rising edge and the time at the upper threshold level of the rising edge, and then calculating the rise time with the following formula:

$$rise\ time = time\ at\ upper\ level - time\ at\ lower\ level$$

When the instrument is in Eye:NRZ measurement mode, the :RISetime query measures and outputs the risetime of the first displayed eye diagram. For the measurements to be made, at least one complete eye diagram must be shown in the display.

Command Syntax:	:MEASure:RISetime
Example:	OUTPUT 707;":MEAS:RIS"
Query Syntax:	:MEASure:RISetime?
Returned Format:	<value><NL>
Where:	<value> ::= rise time in seconds |exponential–NR3 format|
Example:	OUTPUT 707;":MEAS:RIS?" ENTER 707;Rs PRINT Rs

:MEASure:SCRatch

The :MEASure:SCRatch command clears selected measurements and selected markers from the screen.

Command Syntax:	:MEASure:SCRatch
Example:	OUTPUT 707;":MEASURE:SCRATCH"

:MEASure:SHOW

The :MEASure:SHOW command enables markers for tracking measurements.

The :SHOW query returns the status of the markers.

Command Syntax:	:MEASure:SHOW {ON | OFF}
Example:	OUTPUT 707;"MEASURE:SHOW ON"
Query Syntax:	:MEASure:SHOW?
Returned Format:	{ON | OFF} <NL>
Example:	DIM Sh$[50] OUTPUT 707;":MEASURE:SHOW?" ENTER 707; Sh$ PRINT Sh$

:MEASure:SOURce

The :MEASure:SOURce command selects the sources for the measurements. The specified source becomes the source for the MEASURE subsystem commands.

The :SOURce query returns the current source selection.

Command Syntax:	:MEASure:SOURce <source>
Where:	<source> ::= CHANnel{1 | 2}
Example:	OUTPUT 707;":MEASURE:SOURCE CHANNEL1"
Query Syntax:	:MEASure:SOURce?
Returned Format:	<source><NL>
Where:	<source> :: = CHAN {1 | 2}
Example:	DIM Src$[50] OUTPUT 707;":MEAS:SOUR?" ENTER 707;Src$ PRINT Src$

:MEASure:TDELta

The :TDELta query returns the time difference between the start and stop time markers:

$$Tdelta = Tstop - Tstart$$

Tstart is the time at the start marker and Tstop is the time at the stop marker. No measurement is made when the :TDELta query is received by the analyzer. The delta time value that is output is the current value. This is the same value as the front panel delta t value.

Query Syntax:	:MEASure:TDELta?
Returned Format:	<value><NL>
Where:	<value> ::= difference between stop and start markers [exponential-NR3 format]
Example:	OUTPUT 707;":MEASURE:TDELTA?" ENTER 707;Td1 PRINT Td1

:MEASure:THResholds

The :MEASure:THResholds command selects the thresholds used when making time measurements. This measurement function is available when the instrument is in either Eye:NRZ or General Waveform measurement mode. The operation of the :MEASure:THResholds command is identical to the `Thresholds` softkey. Refer to the :MEASure:MTYPe command and Chapter 2 of the *HP 83475B Lightwave Communications Analyzer User's Guide* for more information on setting thresholds.

The :THResholds query outputs the current thresholds selected when making time measurements.

Command Syntax:	:MEASure:THResholds {T1090 | T2080 | PERCent}
Example:	OUTPUT 707;":MEAS:THR PERC"
Query Syntax:	:MEASure:THResholds?
Returned Format:	{T1090 | T2080 | PERCent} <NL>
Where:	T1090 uses the 10 and 90% levels of the selected waveform. T2080 uses the 20 and 80% levels of the selected waveform. PERCent uses the upper and lower percentage thresholds set with the :UPPer and :LOWer commands on the selected waveform.

:MEASure:TPOWer

When the :TPOWer query is sent, the displayed signal is searched for the defined amplitude level and transition. The time interval between the trigger event and this defined occurrence is returned as the response to this query. This measurement is available when the instrument is in either Eye:NRZ or General Waveform measurement mode.

The power must be specified as a positive value. The sign of the slope selects a rising (+) or falling (–) edge. If no sign is specified for the slope, it is assumed to be the rising edge.

The magnitude of occurrence defines the occurrence to be reported. For example, +3 returns the time for the third time the waveform crosses the specified amplitude level in the positive direction. Once this amplitude crossing is found, the analyzer outputs the time at that crossing in seconds, with the trigger point (time zero) as the reference.

If the specified crossing cannot be found, the analyzer outputs +9.9E+37. This value is returned if:

- the waveform does not cross the specified power
- the waveform does not cross the specified power level the specified number of times in the specified direction.

Query Syntax:	:MEASure:TPOWer? <amplitude> [W,mW, uW], <slope><occurrence>
Where:	<amplitude> ::= positive power level the waveform must cross. <slope> ::= direction of the waveform when <amplitude> is crossed— rising |space or +| or falling |–|. <occurrence> ::= number of crossings to be reported |if one—the first crossing is reported, if two—the second crossing is reported, and so forth.|
Returned Format:	<time><NL>
Where:	<time> ::= time in seconds of specified amplitude crossing |exponential–NR3 format|
Example:	OUTPUT 707;":MEASURE:TPOW? 10uW,+3" ENTER 707;Tpow PRINT Tpow

:MEASure:TREF

The :MEASure:TREF command and query sets the Time Reference operation. :MEASure:TREF defaults to the operation :MEASure:TREF:OPERation. Automatic operation will cause the analyzer to use internal algorithms to find the crossing point for an eye diagram, or the 50% rising and falling edges of a general waveform. Manual operation allows the user to tell the analyzer what the crossing or rising edges are.

Command Syntax:	:MEASure:TREF {AUTO | MAN}
Example:	OUTPUT 707;":MEASURE:TREF AUTO"
Query Syntax:	:MEASure:TREF?
Returned Format:	{AUTO | MAN} <NL>
Example:	OUTPUT 707;":MEASURE:TREF?" ENTER 707; Rsp$ PRINT Rsp$

:MEASure:TREF :OPERation

The :MEASure:TREF:OPERation command and query is the default for :MEASure:TREF. Automatic operation will cause the analyzer to use internal algorithms to find the crossing point for an eye diagram, or the 50% rising and falling edges of a general waveform. Manual operation allows the user to tell the analyzer what the crossing or rising edges are.

Command Syntax:	:MEASure:TREF:OPER {AUTO | MAN}
Example:	OUTPUT 707;":MEASURE:TREF:OPER AUTO"
Query Syntax:	:MEASure:TREF:OPER?
Returned Format:	{AUTO | MAN} <NL>
Example:	OUTPUT 707;":MEASURE:TREF:OPER?" ENTER 707; Rsp$ PRINT Rsp$

:MEASure:TREF :T1

The :MEASure:TREF:T1 command and query sets the first manual timing reference point. This value will be limited to the left of the second timing reference value and the right of the left hand edge of the screen.

Command Syntax:	:MEASure:TREF:T1 <reference location>
Where:	<reference location> ::= time of reference in seconds
Example:	OUTPUT 707;":MEASURE:TREF:T1 2s"
Query Syntax:	:MEASure:TREF:T1?
Returned Format:	<value><NL>
Where:	<value> ::= location of time reference marker in seconds
Example:	OUTPUT 707;":MEASURE:TREF:T1?" ENTER 707; Rsp$ PRINT Rsp$

:MEASure:TREF :T2

The :MEASure:TREF:T2 command and query sets the second manual timing reference point. This value will be limited to the right of the first timing reference value and the left of the right hand edge of the screen.

Command Syntax:	:MEASure:TREF:T2 <reference location>
Where:	<reference location> ::= time reference in seconds
Example:	OUTPUT 707;":MEASURE:TREF:T2 2s"
Query Syntax:	:MEASure:TREF:T2?
Returned Format:	<value><NL>
Where:	<value> ::= location of time reference marker in seconds.
Example:	OUTPUT 707;":MEASURE:TREF:T2?" ENTER 707; Rsp$ PRINT Rsp$

:MEASure:TSAMples

The :MEASure:TSAMples command sets the number of samples the analyzer will take to when making a time measurement on an eye diagram. The defined number of time samples is also used in the histogram algorithms when a mask or template is scaled to a waveform.

The :TSAMples query returns the number of samples used to make a time measurement on an eye diagram.

This number presents a trade off between speed and accuracy of the measurement.

Command Syntax:	:MEASure:TSAMples <value><NL>
Example:	OUTPUT 707;":MEASure:TSAMples 500"
Where:	<value> :: = Number of samples the instrument will take to make a time measurement on an eye diagram.
Query Syntax:	:MEASure:TSAMples?
Returned Format:	<value><NL>
Where:	<value> :: = Number of samples the instrument will take to make a time measurement on an eye diagram [exponential–NR3 format]
Example:	OUTPUT 707;":MEASure:TSAMples?" ENTER 707;Rsp PRINT Rsp

:MEASure:TSTArt

The MEASure:TSTArt command moves the start marker to the specified time with respect to the trigger time. This measurement is available when the instrument is in either Eye:NRZ or General Waveform measurement mode.

The :TSTArt query returns the time at the start marker.

Command Syntax:	:MEASure:TSTArt <start marker time>
Where:	<start marker time> :: = time at start marker in seconds
Example:	OUTPUT 707;":MEASURE:TSTART 30 NS"
Query Syntax:	:MEASure:TSTArt?
Returned Format:	<value><NL>
Where:	<value> :: = time at start marker in seconds |exponential–NR3 format|
Example:	OUTPUT 707;":MEASURE:TSTART?" ENTER 707;Tst PRINT Tst

:MEASure:TSTOp

The :MEASure:TSTOp command moves the stop marker to the specified time with respect to the trigger time. This measurement is available when the instrument is in either Eye:NRZ or General Waveform measurement mode.

The :TSTOp query returns the time at the stop marker.

Command Syntax:	:MEASure:TSTOp <stop marker time>
Where:	<stop marker time> ::= time at stop marker in seconds
Example:	OUTPUT 707;":MEAS:TSTOP 40 NS"
Query Syntax:	:MEASure:TSTOp?
Returned Format:	<value><NL>
Where:	<value> ::= time at stop marker in seconds |exponential–NR3 format|
Example:	OUTPUT 707;":MEASURE:TSTOP?" ENTER 707;Tst PRINT Tst

:MEASure:TVOLt

When the :TVOLt query is sent, the displayed signal is searched for the defined amplitude level and transition. This measurement is available when the instrument is in either Eye:NRZ or General Waveform measurement mode. The time interval between the trigger event and this defined occurrence is returned as the response to this query.

The voltage can be specified as a negative or positive voltage. To specify a negative voltage, use a minus (–) sign. The sign of the slope selects a rising (+) or falling (–) edge. If no sign is specified for the slope, it is assumed to be the rising edge.

The magnitude of occurrence defines the occurrence to be reported. For example, +3 returns the time for the third time the waveform crosses the specified amplitude level in the positive direction. Once this amplitude crossing is found, the analyzer outputs the time at that crossing in seconds, with the trigger point (time zero) as the reference.

If the specified crossing cannot be found, the analyzer outputs +9.9E+37. This value is returned if:

- the waveform does not cross the specified voltage
- the waveform does not cross the specified voltage the specified number of times in the specified direction.

Query Syntax:	:MEASure:TVOLt? <amplitude> [V, mV, uV] <slope><occurrence>
Where:	<amplitude> ::= positive or negative voltage level the waveform must cross. <slope> ::= direction of the waveform when <amplitude> is crossed— rising |space or +| or falling |–|. <occurrence> ::= number of crossings to be reported |if one—the first crossing is reported, if two—the second crossing is reported, and so forth.|
Returned Format:	<time><NL>
Where:	<time> ::= time in seconds of specified amplitude crossing |exponential–NR3 format|
Example:	OUTPUT 707;":MEASURE:TVOL? -.250V,+3" ENTER 707;Tvol PRINT Tvol

:MEASure:UPPer

The :MEASure:UPPer command sets the upper measurement threshold percentage. This measurement function is available when the instrument is in either Eye:NRZ or General Waveform measurement mode. This value and the LOWer value are used when THResholds are set to PERCentage. The operation of the :MEASure:UPPer command is identical to the `Upper` softkey. Refer to the :MEASure:MTYPe command and Chapter 2 of the *HP 83475B Lightwave Communications Analyzer User's Guide* for more information on setting thresholds.

The :UPPer query outputs the current upper threshold level.

Command Syntax:	:MEASure:UPPer <percent>
Example:	OUTPUT 707;":MEAS:UPP 85"
Query Syntax:	:MEASure:UPPer?
Returned Format:	<percent> <NL>
Where:	<percent> :: = user-defined upper threshold in percent [exponential-NR3 format]

:MEASure:VAMPlitude

The MEASure:VAMPlitude command places the instrument into continuous measurement mode and measures the amplitude of the input signal. This measurement is available when the instrument is in either Eye:NRZ or General Waveform measurement mode.

The :VAMPlitude query measures and outputs the voltage amplitude of the first waveform on the screen. The method used to determine voltage amplitude is to measure VTOP (or VONE) level and VBASE (or VZERO) level, then calculate voltage amplitude as follows:

$$voltage\ amplitude\ = Vtop\ - Vbase\ (or\ Vone\ - Vzero)$$

Command Syntax:	:MEASure:VAMPlitude
Example:	OUTPUT 707;":MEAS:VAMP"
Query Syntax:	:MEASure:VAMPlitude?
Returned Format:	<value> <NL>
Where:	<value> :: = amplitude of selected waveform in volts [exponential-NR3 format]
Example:	OUTPUT 707;":MEAS:VAMP?" ENTER 707;Rsp PRINT Rsp

:MEASure:VAVerage

The :MEASure:VAVerage command places the instrument in the continuous measurement mode and starts a Vaverage measurement. This measurement is available when the instrument is in General Waveform measurement mode.

The :VAVerage query measures the average voltage of the first cycle of the displayed signal. If a complete cycle is not present, the analyzer averages all data points.

Command Syntax:	:MEASure:VAVerage
Example:	OUTPUT 707;":MEAS:VAV"
Query Syntax:	:MEASure:VAVerage?
Returned Format:	<average_voltage><NL>
Where:	<average_voltage> ::= calculated average voltage |exponential-NR3 format|
Example:	OUTPUT 707;":MEAS:VAV?" ENTER 707;Vavg PRINT Vavg

:MEASure:VBASe

The :MEASure:VBASe command places the instrument in the continuous measurement mode and starts a Vbase measurement. This measurement is available when the instrument is in General Waveform measurement mode.

The :VBASe query measures and outputs the voltage value at the base of the waveform. The base voltage of a pulse is normally not the same as the minimum value.

Command Syntax:	:MEASure:VBASe
Example:	OUTPUT 707;":MEAS:VBASE"
Query Syntax:	:MEASure:VBASe?
Returned Format:	<base_voltage><NL>
Where:	<base_voltage> ::= voltage at base of selected waveform [exponential-NR3 format]
Example:	OUTPUT 707;":MEASURE:VBASE?" ENTER 707;Vbase PRINT Vbase

:MEASure:VDELta

The :VDELta query outputs the voltage difference between VMarker 1 and VMarker 2. No measurement is made when the :VDELta query is received by the analyzer. The delta voltage value that is output is the current value. This is the same value as the front panel delta V value.

VDELTA = Voltage at VMarker 2 − Voltage at VMarker 1

Query Syntax:	:MEASure:VDELta?
Returned Format:	<value><NL>
Where:	<value> ::= delta V value in volts [exponential-NR3 format]
Example:	OUTPUT 707;":MEAS:VDELTA?" ENTER 707;Vdelta PRINT Vdelta

:MEASure:VMAX

The :MEASure:VMAX command places the instrument in the continuous measurement mode and starts a Vmax measurement. This measurement is available when the instrument is in either Eye:NRZ or General Waveform measurement mode.

The :VMAX query measures and outputs the absolute maximum voltage present on the selected waveform.

Command Syntax:	:MEASure:VMAX
Example:	OUTPUT 707;":MEAS:VMAX"
Query Syntax:	:MEASure:VMAX?
Returned Format:	<value><NL>
Where:	<value> ::= maximum voltage of selected waveform |exponential–NR3 format|
Example:	OUTPUT 707;":MEASURE:VMAX?" ENTER 707;Vmax PRINT Vmax

:MEASure:VMIN

The :MEASure:VMIN command places the instrument in the continuous measurement mode and starts a VMIN measurement. This measurement is available when the instrument is in either Eye:NRZ or General Waveform measurement mode.

The :VMIN query measures and outputs the absolute minimum voltage present on the selected waveform.

Command Syntax:	:MEASure:VMIN
Example:	OUTPUT 707;":MEAS:VMIN"
Query Syntax:	:MEASure:VMIN?
Returned Format:	<value><NL>
Where:	<value> ::= minimum voltage value of the selected waveform |exponential-NR3 format|
Example:	OUTPUT 707;":MEASURE:VMIN?" ENTER 707;Vmin PRINT Vmin

:MEASure:VPP

The :MEASure:VPP command places the instrument in the continuous measurement mode and starts a VPP measurement. This measurement is available when the instrument is in General Waveform measurement mode.

The :VPP query measures the maximum and minimum voltages for the selected source, then calculates the peak-to-peak voltage and outputs that value. The peak-to-peak voltage (Vpp) is calculated with the following formula:

$$Vpp = Vmax - Vmin$$

Vmax and Vmin are the maximum and minimum voltages present on the selected source.

Command Syntax:	:MEASure:VPP
Example:	OUTPUT 707;":MEAS:VPP"
Query Syntax:	:MEASure:VPP?
Returned Format:	<value><NL>
Where:	<value> ::= voltage peak to peak [exponential-NR3 format]
Example:	OUTPUT 707;":MEAS:VPP?" ENTER 707;Vpp PRINT Vpp

:MEASure:VRMS

The :MEASure:VRMS command places the instrument in the continuous measurement mode and starts a dc rms voltage measurement. This measurement is available when the instrument is in General Waveform measurement mode.

The :VRMS query measures and outputs the dc rms voltage of the selected waveform. The dc rms voltage is measured on the first cycle of the displayed signal. If a complete cycle is not present, the analyzer computes the rms value on all displayed data points.

Command Syntax:	:MEASure:VRMS
Example:	OUTPUT 707;":MEAS:VRMS"
Query Syntax:	:MEASure:VRMS?
Returned Format:	<value><NL>
Where:	<value> ::= calculated dc rms voltage |exponential-NR3 format|
Example:	OUTPUT 707;":MEASURE:VRMS?" ENTER 707;Vrms PRINT Vrms

:MEASure:VSTArt

The :MEASure:VSTArt command moves VMarker 1 to the specified voltage corresponding to the current source. The source can be selected by the :MEASure:SOURce command.

The :VSTArt query returns the current voltage level of VMarker 1.

Command Syntax:	:MEASure:VSTArt <voltage> [V, mV, uV]
Where:	<voltage> ::= voltage value for Vmarker 1
Example:	OUTPUT 707;":MEAS:VSTA -10MV"
Query Syntax:	:MEASure:VSTArt?
Returned Format:	<value><NL>
Where:	<value> ::= voltage at VMarker 1 [exponential-NR3 format]
Example:	OUTPUT 707;":MEASURE:VSTART?" ENTER 707;Vst PRINT Vst

:MEASure:VSTOp

The :MEASure:VSTOp command moves VMarker 2 to the specified voltage corresponding to the current source. The source can be selected by the :MEASure:SOURce command.

The :VSTOp query returns the current voltage level of VMarker 2.

Command Syntax:	:MEASure:VSTOp <voltage> [V, mV, uV]
Where:	<voltage> ::= voltage value for Vmarker 2
Example:	OUTPUT 707;":MEAS:VSTO -100MV"
Query Syntax:	:MEASure:VSTOp?
Returned Format:	<value><NL>
Where:	<value> ::= voltage at VMarker 2 [exponential-NR3 format]
Example:	OUTPUT 707;":MEASURE:VSTOP?" ENTER 707;Vst PRINT Vst

:MEASure:VTIMe

The :VTIMe query returns the voltage at a specified time.

The specified time must be on screen and is referenced to the trigger event.

Query Syntax:	:MEASure:VTIMe? <time>
Where:	<time> ::= displayed time from trigger in seconds
Returned Format:	<voltage><NL>
Where:	<voltage> ::= voltage at specified time [exponential-NR3 format]
Example:	OUTPUT 707;":MEASURE:VTIME? .001" ENTER 707;Vtm PRINT Vtm

:MEASure:VTOP

The :MEASure:VTOP command places the instrument in the continuous measurement mode and starts a Vtop measurement. This measurement is available when the instrument is in General Waveform measurement mode.

The :VTOP query returns the voltage at the top level of a waveform. The top level is the most prevalent high level of the signal.

Command Syntax:	:MEASure:VTOP
Example:	OUTPUT 707;":MEASURE:VTOP"
Query Syntax:	:MEASure:VTOP?
Returned Format:	<value><NL>
Where:	<value> ::= voltage at the top of the waveform [exponential-NR3 format]
Example:	OUTPUT 707;":MEASURE:VTOP?" ENTER 707;Vtp PRINT Vtp

:MEASure:ZLEVel

The MEASure:ZLEVel command puts the instrument into continuous measurement mode and measures the zero level of the eye diagram signal. This measurement is available when the instrument is in Eye:NRZ Waveform measurement mode.

The :ZLEVel query returns the zero level of the input signal. The zero level is the most prevalent or the mean low level of the signal.

Command Syntax:	:MEASure:ZLEVel
Query Syntax:	:MEASure:ZLEVel?
Returned Format:	<value><NL>
Where:	<value> ::= the zero level of the input signal.
Example:	OUTPUT 707;":MEAS:ZLEV?" ENTER 707;Rsp PRINT Rsp

:OCALibrate:CALibrate

The :CALibrate query begins the calibration sequence for the selected calibration. Either standard or user-defined calibrations may be selected and changed.

WARNING

The instrument *will accept* other remote commands during the execution of the calibration sequence. If operation of the analyzer is suspect, it may be necessary to cycle the line power off, then on, to ensure the analyzer is in a known operational state.

Query Syntax:	:OCALibrate:CALibrate?
Returned Format:	{CONT | ERR} <NL>
Where:	CONT indicates the last step of the calibration was successful and the calibration procedure can continue. ERR indicates an unrecoverable error has occurred at the last step in the calibration. The calibration is aborted and the instrument is reset to the state it was in before the calibration was performed.

:OCALibrate:ECALibrate
:CHAN1 (or CHAN2)

The :ECALibrate:CHAN1 (CHAN2) query performs an electrical calibration of channel 1 (or 2).

WARNING

Before executing this query, make sure the rear panel `DC OUTPUT` is connected to the front panel `CHANNEL 1 (or 2) INPUT`.

Query Syntax:	:OCALibrate:ECALibrate:CHAN1?
Returned format:	{CONT | ERR} <NL>
Where:	CONT indicates the last step of the calibration was successful and the calibration procedure can continue. ERR indicates an unrecoverable error has occurred at the last step in the calibration. The calibration is aborted and the instrument is reset to the state it was in before the calibration was performed.

:OCALibrate:ECALibrate
:SKIP

The :ECALibrate:SKIP query allows the controller to omit (skip) the electrical calibration steps. When selected, the next step performed should be OCAL:OCAL:PDAR.

Query Syntax:	:OCALibrate:ECALibrate:SKIP?
Returned format:	{CONT | ERR} <NL>
Where:	CONT indicates the last step of the calibration was successful and the calibration procedure can continue. ERR indicates this query was executed in the wrong sequence. It must be executed in the sequence described in Appendix A. This is the only place in the remote optical calibration sequence the calibration sequence can continue after ERR is returned.

:OCALibrate:ECALibrate
:XTRigger

The :ECALibrate:XTRigger query performs an electrical calibration of the external trigger circuits.

WARNING

Before executing this query, make sure the rear panel DC OUTPUT is connected to the front panel EXTERNAL TRIGGER INPUT.

Query Syntax:	:OCALibrate:ECALibrate:XTRigger?
Returned format:	{CONT | ERR} <NL>
Where:	CONT indicates the last step of the calibration was successful and the calibration procedure can continue. ERR indicates an unrecoverable error has occurred at the last step in the calibration. The calibration is aborted and the instrument is reset to the state it was in before the calibration was performed.

:OCALibrate:ERATio
:FCOR

The :OCALibrate:ERATio:FCOR command sets the extinction ratio frequency correction factor and allows you to activate frequency correction.

The :ERATio:FCOR query returns the current setting of the frequency correction value.

Command Syntax:	:OCALibrate:ERATio:FCOR { OP | E1 | E2 } <value> { ON | OFF }
Where:	<value> ::= floating point frequency correction from 0 to 100
Example:	OUTPUT 707;":OCAL:ERAT:FCOR 10 ON"
Query Syntax:	:OCALibrate:ERATio:FCOR? { OP | E1 | E2 }
Returned format:	{ ON | OFF } <value>
Where:	<value> ::= floating point frequency correction from 0 to 100

:OCALibrate:ERATio
:IOPT

The :OCALibrate:ERATio:IOPT query adjusts the zero level, with no signal input, taking into account the current residual dc offset of the optical channel.

It returns either COMPlete or ERRor.

When this measurement is made there should be no input into the optical channel.

Query Syntax:	:OCALibrate:ERATio:IOPT?
Returned format:	{COMP | ERR} <NL>
Where:	COMP indicates the adjustment was successful. ERR indicates an unrecoverable error has occurred at the last step in the calibration. The calibration is aborted and the instrument is reset to the state it was in before the calibration was performed.
Example:	OUTPUT 707;":OCALibrate:ERATio:IOPT?" ENTER 707;Rsp PRINT Rsp

:OCALibrate:ERATio
:XE1

The :OCALibrate:ERATio:XE1 command adjusts the zero level, with no signal input, taking into account the current residual dc offset of the Elec 1 channel.

The :ERATio:XE1 query returns either COMPlete or ERRor.

When this measurement is made, there should be no input into the Elec 1 channel.

Command Syntax:	:OCALibrate:ERATio:XE1
Example:	OUTPUT 707;":OCALibrate:ERATio:XE1"
Query Syntax:	:OCALibrate:ERATio:XE1?
Returned format:	{COMP | ERR} <NL>
Where:	COMP indicates the adjustment was successful. ERR indicates an unrecoverable error has occurred at the last step in the calibration. The calibration is aborted and the instrument is reset to the state it was in before the calibration was performed.
Example:	OUTPUT 707;":OCALibrate:ERATio:XE1?" ENTER 707;Rsp PRINT Rsp

:OCALibrate:ERATio :XE2

The :OCALibrate:ERATio:XE2 command adjusts the zero level, with no signal input, taking into account the current residual dc offset of the Elec 2 channel.

The :ERATio:XE2 query returns either COMPlete or ERRor.

When this measurement is made, there should be no input into the Elec 1 channel.

Command Syntax:	:OCALibrate:ERATio:XE2
Example:	OUTPUT 707;":OCALibrate:ERATio:XE2"
Query Syntax:	:OCALibrate:ERATio:XE2?
Returned format:	{COMP | ERR} <NL>
Where:	COMP indicates the adjustment was successful. ERR indicates an unrecoverable error has occurred at the last step in the calibration. The calibration is aborted and the instrument is reset to the state it was in before the calibration was performed.
Example:	OUTPUT 707;":OCALibrate:ERATio:XE2?" ENTER 707;Rsp PRINT Rsp

:OCALibrate:KILL

The :KILL query aborts the calibration sequence and resets the instrument to the state it was in before the calibration sequence was initiated.

Query Syntax:	:OCALibrate:KILL?
Returned Format:	{ERR} <NL>
Where:	ERR indicates the calibration is aborted and the instrument is reset to the state it was in before the calibration sequence was initiated.
Example:	OUTPUT 707;":OCALibrate:KILL?" ENTER 707;Rsp PRINT Rsp

:OCALibrate:LABel

The :OCALibrate:LABel command labels the currently selected calibration and displays the label on the softkey. The calibration to be labeled can be selected using the :SCALibrate command.

Command Syntax:	:OCALibrate:LABel <string>
Example:	OUTPUT 707;":OCALibrate:LABel User Cal 1"

:OCALibrate:LDEFault

The :OCALibrate:LDEFault command loads the standard factory startup values as the calibration factors. It is important this is done *before* executing the complete optical channel calibration sequence because a complete sequence includes an electrical system vertical calibration.

Command Syntax:	:OCALibrate:LDEFault
Example:	OUTPUT 707;":OCALibrate:LDEFault"

:OCALibrate:OCALibrate
:OCGain

The :OCALibrate:OCGain query performs the optical channel gain calibration. To perform a calibration, a known input power, close to 160 μW, must be fed into the optical input.

WARNING

Before executing this query:

- Connect the laser source to the `OPTICAL INPUT`.
- For 1300 nm or 1550 nm calibrations, set the laser power as close to 160 μW as possible. For best calibration results, do not exceed 160 μW (−7.9 dBm).
- For 780 nm calibrations, set the laser power as close to 1000 μW as possible. For best calibration results, do not exceed 1000 μW (0 dBm).
- The optical input damage level is 2.5 mW (=4 dBm).

Query Syntax:	:OCALibrate:OCALibrate:OCGain <power>[W, mW, uW]?
Returned format:	{CONT | ERR} <NL>
Where:	<power> ::= known power level input into the optical channel. CONT indicates the last step of the calibration was successful and the calibration procedure can continue. ERR indicates an unrecoverable error has occurred at the last step in the calibration. The calibration is aborted and the instrument is reset to the state it was in before the calibration was performed.

:OCALibrate:OCALibrate :PDARk

The :OCALibrate:PDARk query performs, as part of a calibration, a measurement of the dark noise on the optical channel.

WARNING

Before executing this query make sure:

- The `DEMOD OUTPUT` OR `EXT O/E` converter is connected to the `CHANNEL 2 INPUT`
- There is no signal applied to the optical input
- The source is deactivated or a dark cap is placed on the optical input connector

Query Syntax:	:OCALibrate:OCALibrate:PDARk?
Returned format:	{CONT | ERR} <NL>
Where:	CONT indicates the last step of the calibration was successful and the calibration procedure can continue. ERR indicates an unrecoverable error has occurred at the last step in the calibration. The calibration is aborted and the instrument is reset to the state it was in before the calibration was performed.

:OCALibrate:OCALibrate
:PMET:HHRange

The :OCALibrate:PMET:HHRange query measures and saves the optical power high value for the high range calibration.

Query Syntax:	:OCALibrate:PMET:HHRange <power> [W, mW, uW]?
Returned format:	{OFFS | ERR | CONT} <NL>
Where:	<power> ::= known power level input into the optical channel.
	CONT indicates the last step of the calibration was successful and the calibration procedure can continue.
	ERR indicates an unrecoverable error has occurred at the last step in the calibration. The calibration is aborted and the instrument is reset to the state it was in before the calibration was performed.
	OFFS indicates the last calibration phase was indeterminate. The last step must be tried again, varying the input criteria until the analyzer senses the power level and can return CONT.

:OCALibrate:OCALibrate
:PMET:HLRange

The :OCALibrate:PMET:HLRange query measures and saves the optical power high value for the low range calibration.

Query Syntax:	:OCALibrate:PMET:HLRange <power> [W, mW, uW]?
Returned format:	{OFFS | ERR | CONT} <NL>
Where:	<power> ::= known power level input into the optical channel.
	CONT indicates the last step of the calibration was successful and the calibration procedure can continue.
	ERR indicates an unrecoverable error has occurred at the last step in the calibration. The calibration is aborted and the instrument is reset to the state it was in before the calibration was performed.
	OFFS indicates the last calibration phase was indeterminate. The last step must be tried again, varying the input criteria until the analyzer senses the power level and can return CONT.

:OCALibrate:OCALibrate
:PMET:HMRange

The :OCALibrate:PMET:HMRange query measures and saves the optical power high value for the midrange calibration.

Query Syntax:	:OCALibrate:PMET:HMRange <power> [W, mW, uW]?
Returned format:	{OFFS | ERR | CONT} <NL>
Where:	<power> ::= known power level input into the optical channel. CONT indicates the last step of the calibration was successful and the calibration procedure can continue. ERR indicates an unrecoverable error has occurred at the last step in the calibration. The calibration is aborted and the instrument is reset to the state it was in before the calibration was performed. OFFS indicates the last calibration phase was indeterminate. The last step must be tried again, varying the input criteria until the analyzer senses the power level and can return CONT.

:OCALibrate:OCALibrate
:PMET:LHRange

The :OCALibrate:PMET:LHRange query measures and saves the optical power low value for the high range calibration.

Query Syntax:	:OCALibrate:PMET:LHRange <power> [W, mW, uW]?
Returned format:	{OFFS | ERR | CONT} <NL>
Where:	<power> ::= known power level input into the optical channel. CONT indicates the last step of the calibration was successful and the calibration procedure can continue. ERR indicates an unrecoverable error has occurred at the last step in the calibration. The calibration is aborted and the instrument is reset to the state it was in before the calibration was performed. OFFS indicates the last calibration phase was indeterminate. The last step must be tried again, varying the input criteria until the analyzer senses the power level and can return CONT.

:OCALibrate:OCALibrate
:PMET:LLRange

The :OCALibrate:PMET:LLRange query measures and saves the optical power low value for the low range calibration.

Query Syntax:	:OCALibrate:PMET:LLRange <power> [W, mW, uW]?
Returned format:	{OFFS | ERR | CONT} <NL>
Where:	<power> ::= known power level input into the optical channel.
	CONT indicates the last step of the calibration was successful and the calibration procedure can continue.
	ERR indicates an unrecoverable error has occurred at the last step in the calibration. The calibration is aborted and the instrument is reset to the state it was in before the calibration was performed.
	OFFS indicates the last calibration phase was indeterminate. The last step must be tried again, varying the input criteria until the analyzer senses the power level and can return CONT.

:OCALibrate:OCALibrate
:PMET:LMRange

The :OCALibrate:PMET:LMRange query measures and saves the optical power low value for the midrange calibration.

Query Syntax:	:OCALibrate:PMET:LMRange <power> [W, mW, uW]?
Returned format:	{OFFS | ERR | CONT} <NL>
Where:	<power> ::= known power level input into the optical channel.
	CONT indicates the last step of the calibration was successful and the calibration procedure can continue.
	ERR indicates an unrecoverable error has occurred at the last step in the calibration. The calibration is aborted and the instrument is reset to the state it was in before the calibration was performed.
	OFFS indicates the last calibration phase was indeterminate. The last step must be tried again, varying the input criteria until the analyzer senses the power level and can return CONT.

:OCALibrate:SAVE

The :SAVE query saves the calibration after the required measurement sequence has been correctly performed.

Query Syntax:	:OCALibrate:SAVE?
Returned format:	{COMP | ERR} <NL>
Where:	COMP indicates the calibration was successful and the data successfully stored. ERR indicates an unrecoverable error has occurred at the last step in the calibration. The calibration is aborted and the instrument is reset to the state it was in before the calibration was performed.

:OCALibrate:SCALibrate

The :OCALibrate:SCALibrate command selects the wavelength calibration inside the instrument for average optical power meter and watts vertical scale measurements.

The :OCALibrate:SCALibrate query returns the currently selected wavelength calibration.

Command Syntax:	:OCALibrate:SCALibrate {D0780 | D1310 | D1550 | U0780 | U1310 | U1550}
Example:	OUTPUT 707;":OCALibrate:SCALibrate D0780"
Query Syntax:	:OCALibrate:SCALibrate?
Returned Format:	{ D0780 | D1310 | D1550 | U0780 | U1310 | U1550 }<NL>
Example:	DIM Rsp$[50] OUTPUT 707;":OCALibrate:SCALibrate?" ENTER 707;Rsp$ PRINT Rsp$

:OCALibrate:SDISplay

The :OCALibrate:SDISplay command display either the user or default calibration choices on the screen.

Command Syntax:	:OCALibrate:SDISplay {USER | DEFAULT}
Example:	OUTPUT 707;":OCALibrate:SDISplay USER"

:PRINt

The :PRINT query outputs a copy of the display as soon as the analyzer is addressed to talk. The portion of the waveform to be copied must be placed on the display before sending the :PRINT query. The printer output includes the displayed waveforms, the graticule, time and voltage markers, and measurement results.

There is an optional parameter for high resolution mode. When :PRINT? HIRES is sent to the analyzer, the analyzer will return the print data at high resolution mode as soon as it is addressed to talk. The high resolution mode contains both half-bright and full-bright display information.

The high resolution print data will work with printers that are compatible with HP-PCL and have a resolution of 300 dpi or higher, such as the HP LaserJet series.

Query Syntax:	:PRINT? [HIRes]
Example:	OUTPUT 707;":PRINT?" SEND 7;UNT UNL SEND 7;LISTEN 1 !Assumes the printer is at address 1 SEND 7;TALK 7 SEND 7;DATA

:RUN

The :RUN command starts the acquisition of data for the active waveform display. It has the same effect as pressing the front-panel (Run) key.

The :RUN command also turns off the Autostore function.

Command Syntax:	:RUN
Example:	OUPUT 707;":RUN"

:STATus

The :STATus query indicates if a channel is ON or OFF. STATUS also indicates if the external trigger view is ON or OFF.

Query Syntax:	:STATus? <display>
Where:	<display> ::= {CHANnel{1 | 2} | PMEMory{1 | 2} | EXTernal}
Returned Format:	{ON | OFF}<NL>
Example:	DIM Status$[5] OUTPUT 707;":STATUS? CHANNEL1" ENTER 707;Status$ PRINT Status$

:STOP

The :STOP command stops the data acquisition. This command also turns AUTOSTORE off.

The :RUN command must be executed to restart the acquisition.

Command Syntax:	:STOP
Example:	OUTPUT 707;":STOP"

:SYSTem:DSP

The :SYSTem:DSP command writes a quoted string, excluding quotes, to the message and measurement line of the screen.

Command Syntax:	:SYSTem:DSP <quoted ASCII string>
Example:	OUTPUT 707;":SYSTEM:DSP ""This is a message"""

:SYSTem:ERRor

The :ERRor query outputs the next error number in the error queue over the interface. This instrument has an error queue that is 15 errors deep and operates on a first-in, first-out basis. Repeatedly sending the query :SYSTem:ERRor? returns the error numbers in the order that they occurred until the queue is empty. Any further queries then return zeros until another error occurs. See Table 7-6 for the error numbers and descriptions.

Query Syntax:	:SYSTem:ERRor?
Returned Format:	<error><NL>
Where:	<error> ::= an integer error code.
Example:	OUTPUT 707;":SYSTEM:ERROR?" ENTER 707;Emsg PRINT Emsg

Table 7-6. Error Messages

Error Number	Description
-100	Command error [unknown command]
-101	Invalid character
-102	Syntax error
-103	Invalid separator
-104	Data type error
-105	GET not allowed
-108	Parameter not allowed
-109	Missing parameter
-112	Program mnemonic too long
-113	Undefined header
-121	Invalid character in number
-123	Numeric overflow
-124	Too many digits
-128	Numeric data not allowed
-130	Suffix error
-131	Invalid suffix
-138	Suffix not allowed
-140	Character data error
-141	Invalid character data
-144	Character data too long
-148	Character data not allowed

Table 7-6. Error Messages (continued)

Error Number	Description
-150	String data error
-151	Invalid string data
-158	String data not allowed
-160	Block data error
-161	Invalid block data
-168	Block data not allowed
-170	Expression error
-171	Invalid expression
-178	Expression data not allowed
-200	Execution error
-211	Trigger ignored
-221	Settings conflict
-222	Data out of range
-223	Too much data
-310	System error
-350	Too many errors
-400	Query error
-410	Query INTERRUPTED
-420	Query UNTERMINATED
-430	Query DEADLOCKED
-440	Query UNTERMINATED after indefinite response

:SYSTem:KEY

The :SYSTem:KEY command simulates the pressing of a specified front-panel key. Key commands may be sent over the interface in any order that are legal key presses from the front panel. Make sure the instrument is in the desired state before executing the :KEY command.

The :KEY query returns the key code for the last key pressed from the front panel. Key codes range from −1 to 50.

Refer to Table 7-7 for key codes.

The keycode −1 is returned if no new key has been pressed.

Command Syntax:	:SYSTem:KEY <keycode>
Where:	<keycode> ::= 0 to 16 or 19 to 50
Example:	OUTPUT 707;":SYSTEM:KEY 2"
Query Syntax:	:SYSTem:KEY?
Returned Format:	<keycode><NL>
Where:	<keycode> ::= integer, −1 to 16 or 19 to 50
Example:	OUTPUT 707;":SYSTEM:KEY?" ENTER 707;Input PRINT Input

Table 7-7. Front-Panel Key Codes

Key	Key Code	Key[1]	Key Code
AUTOSCALE	0	SOFTKEY_4	26
ELEC1	1	SOFTKEY_5	27
OPTICAL/ELEC2	2	SOFTKEY_6	28
NA	3	CH1_VOLT_CW	29
OPTICAL CAL	4	CH1_VOLT_CCW	30
+/-	5	CH1_POS_CW	31
TRG_SRC	6	CH1_POS_CCW	32
TRG_MODE	7	CH2_AMP_CW	33
TRG_SLOPE	8	CH2_AMP_CCW	34
MAIN/DELAYED	9	CH2_POS_CWW	35
TIME	10	CH2_POS_CCW	36
POWER/VOLTAGE	11	N/A	37
CURSORS/MASKS	12	N/A	38
SAVE_TRACE	13	N/A	39
SAVE_SETUP	14	N/A	40
DISPLAY	15	S/DIV_CW	41
PRINT/UTILITY	16	S/DIV_CCW	42
NA	17	DELAY_CW	43
NA	18	DELAY_CCW	44
RUN	19	TRG_LEVEL_CW	45
AUTOSTORE	20	TRG_LEVEL_CCW	46
STOP	21	TRG_HOLD_CW	47
ERASE	22	TRG_HOLD_CCW	48
SOFTKEY_1	23	GENERAL_PURPOSE_KNOB_CW	49
SOFTKEY_2	24	GENERAL_PURPOSE_KNOB_CCW	50
SOFTKEY_3	25	NO KEY	–1

1 CW denotes clockwise rotation of the knob. CCW denotes counterclockwise rotation of the knob.

:SYSTem:LOCK

The :SYSTem:LOCK command disables the front panel. LOCK ON is the equivalent of sending a local lockout message over HP-IB.

The :LOCK query returns the lock status of the front panel.

Command Syntax:	:SYSTem:LOCK {ON | OFF}
Example:	OUTPUT 707;":SYST:LOCK ON"
Query Syntax:	:SYSTem:LOCK?
Returned Format:	{ON | OFF}<NL>
Example:	DIM Lock$[5] OUTPUT 707;":SYSTEM:LOCK?" ENTER 707;Lock$ PRINT Lock$

:SYSTem:SETup

The :SYSTem:SETup command sets the analyzer as defined by the data in the setup (learn) string sent from the controller. The setup string contains 144 bytes of setup data. The 144 bytes do not include the header "**#800000121**".

The setup string does not change the interface mode, interface address, waveform format, or waveform source.

The :SETup query operates the same as the *LRN query. It outputs the current analyzer setup in the form of a learn string to the controller.

The setup (learn) string is sent and received as a binary block of data. The format for the data transmission is the # format defined in the IEEE 488.2 specification.

The logical order for this instruction is to send the query, followed by the command. The query causes the learn string to be sent to the controller and the command causes the learn string to be returned to the analyzer.

Command Syntax:	:SYSTem:SETup <setup>
Where:	<setup> ::= #800000141<setup data string>
Query Syntax:	:SYSTem:SETup?
Returned Format:	<setup><NL>
Where:	<setup> ::= #80000141<setup data string>
Example:	10 DIM Set$[200] 20 !Setup the instrument as desired 30 OUTPUT 707;":SYSTEM:SETUP?" 40 !Transfer the instrument setup to controller 50 ENTER 707 USING "-K";Set$!Store the setup 60 PAUSE 70 OUTPUT 707 USING "#,K";":SYST:SETUP ";Set$ 80 !Returns the instrument to the first setup 90 END

:TER
(Trigger Event Register)

The :TER query reads the trigger event bit in the Event Status Register. After the trigger event bit is read, it is cleared. A one indicates a trigger has occurred. A zero indicates a trigger has not occurred.

A Service Request (SRQ) can be generated when TRG bit of the ESR (Event Status Register) transitions and both ESE and SRE bits have been set.

The bit must be cleared each time you would like a new Service Request to be generated.

Query Syntax:	:TER?
Returned Format:	{1 | 0}<NL>
Example:	OUTPUT 707;":TER?" ENTER 707;Trg_event PRINT Trg_event

:TIMebase:DELay

The :TIMebase:DELay command sets the time base delay. This delay is the internal time between the trigger event and the onscreen delay reference point. The delay reference point is one division from the left edge of the display, and is set with the :TIMebase:REFerence command. When the delayed timebase is selected, the :DELay command sets the internal time between the trigger event and the onscreen delay reference point in the delayed timebase.

The :DELay query returns the current delay value.

Command Syntax:	:TIMebase:DELay <delay>
Where:	<delay> ::= Time in seconds from trigger to the on screen delay reference point. The maximum value depends on the time/division setting.
Example:	OUTPUT 707;":TIM:DEL 2MS"
Query Syntax:	:TIMebase:DELay?
Returned Format:	<delay><NL>
Where:	<delay> ::= Time from trigger to display reference in seconds. Display reference is left or center [exponential–NR3 format]
Example:	OUTPUT 707;":TIMEBASE:DELAY?" ENTER 707;Delay PRINT Delay

:TIMebase:MODE

The :TIMebase:MODE command selects the time base mode. There are four timebase modes:

- NORMal - The Normal timebase mode is the main timebase. It is the default timebase mode after the *RST (reset) command.
- DELayed - In the Delayed timebase mode, the range and delay commands set the values in the delayed timebase instead of the main (Normal) timebase. No waveform data is available through the interface bus. Measurements will be made in the delayed timebase if possible, otherwise the measurements will be made in the main timebase.
- XY - In the XY mode, the :RANGe, :DELay, :REFerence and :VERNier commands are not available. No measurement or waveform data are available.
- ROLL - In the Roll mode, data moves continuously across the display from left to right. The analyzer runs continuously and is untriggered. The time reference selection changes from LEFT | CENTer to CENTer | RIGHT.

The :MODE query returns the current time base mode.

N O T E

Delayed and Roll modes are not available when the measurement type is set to Eye:NRZ.

Command Syntax:	:TIMebase:MODE {NORMal | DELayed | XY | ROLL}
Example:	OUTPUT 707;":TIM:MODE NORMAL"
Query Syntax:	:TIMebase:MODE?
Returned Format:	{NORM | DEL | XY | ROLL}<NL>
Example:	DIM Mode$[30] OUTPUT 707;":TIMEBASE:MODE?" ENTER 707;Mode$ PRINT Mode$

:TIMebase:RANGe

The :TIMebase:RANGe command sets the full-scale horizontal time in seconds. The RANGe value is ten times the time per division. When the delayed timebase is selected, the :RANGe command will set the full-scale horizontal time of the delayed timebase.

The :RANGe query returns the current full-scale range value.

Command Syntax:	:TIMebase:RANGe <range>
Where:	<range> ::= 10 ns to 50 s
Example:	OUTPUT 707;":TIM:RANG 100 MS"
Query Syntax:	:TIMebase:RANGe?
Returned Format:	<range><NL>
Where:	<range> ::= 10 ns to 50 s [exponential-NR3 format]
Example:	OUTPUT 707;":TIMEBASE:RANGE?" ENTER 707;Rng PRINT Rng

:TIMebase:REFerence

The :TIMebase:REFerence command sets the display reference to one division from the left side of the screen or to the center of the screen.

The :REFerence query returns the current display reference.

N O T E

When the analyzer timebase is in ROLL mode, the time reference selection changes to CENTer | RIGHT.

Command Syntax:	:TIMebase:REFerence {LEFT | CENTer}
Example:	OUTPUT 707;":TIMEBASE:REFERENCE LEFT"
Query Syntax:	:TIMebase:REFerence?
Returned Format:	{LEFT | CENT}<NL>
Example:	DIM Rf$[30] OUTPUT 707;":TIMEBASE:REFERENCE?" ENTER 707;Rf$ PRINT Rf$

:TIMebase:SETup

The :TIMebase:SETup query returns the current setting of all commands in the Timebase subsystem.

Query Syntax:	:TIMebase:SETup?
Returned Format:	TIMEBASE:MODE {NORM | DEL | XY};RANGE <range>;DELAY <delay>; REF {LEFT | CENT};VERN {ON | OFF}<NL>
Where:	<range>::= 20 ns to 50 s |exponential-NR3 format|
	<delay>::= time from trigger to delay reference in seconds |exponential-NR3 format|
Example:	DIM Stp$[300] OUTPUT 707;":TIMEBASE:SETUP?" ENTER 707;Stp$ PRINT Stp$

:TIMebase:VERNier

The :TIMebase:VERNier command selects vernier for the timebase. The command affects only the user control and not the :TIMebase:RANGe command.

The :VERNier query returns the current vernier mode.

Command Syntax:	:TIMebase:VERNier {ON | OFF}
Example:	OUTPUT 707;":TIMEBASE:VERN ON"
Query Syntax:	:TIMebase:VERNier?
Returned Format:	{ON | OFF}<NL>
Example:	DIM Smp$[30] OUTPUT 707;":TIMEBASE:VERNIER?" ENTER 707;Smp$ PRINT Smp$

:TRACe:CLEAR

The :TRACe:CLEAR command clears the contents of the selected trace memory. The operation of the :TRACe:CLEAR command is identical to the `Clear Trace` softkey. Refer to "To Save or Recall Traces" in Chapter 2 for additional information.

Command Syntax:	:TRACe:CLEAR <N>
Example:	OUTPUT 707;":TRAC:CLEAR 2"
Where:	<N> :: = 1 to 100

:TRACe:DATA

The :TRACe:DATA command downloads trace memory data to the trace memory number specified. Compressed images (5–100) may not be downloaded into one of the non-compressed trace memories (1–4). Trace labels and time-tags are also set.

The :DATA query outputs trace data, trace labels, and time-tags from the specified trace memory.

Command Syntax:	:TRACe:DATA <N>
Example:	OUTPUT 707;":TRAC:DATA 4"
Query Syntax:	:TRACe:DATA? <N>
Returned Format:	<header> <trace_data> <NL>
Where:	<N> :: = The trace memory number (1 to 100) <header> :: = The block header (10 bytes) contains the ASCII characters #8000nnnnn and is sent prior to the data (nnnnn is the number of bytes in the data string). <trace_data> :: = A maximum of 16,342 bytes of data, setup, and label information that represents the current trace.

:TRACe:MODE

The :TRACe:MODE command enables or disables displaying the selected trace memory waveform on the analyzer display. The operation of the :TRACe:MODE command is identical to the `Trace On Off` softkey. Refer to "To Save or Recall Traces" in Chapter 2 for additional information.

The :MODE query outputs the selected trace memory state. When ON is selected, trace memory is displayed on the analyzer screen.

Command Syntax:	:TRACe:MODE <N> {ON | OFF}
Example:	OUTPUT 707;":TRAC:MODE 4 ON"
Query Syntax:	:TRACe:MODE? <N>
Returned Format:	{ON | OFF} <NL>
Where:	<N> :: = The trace memory number |1 to 100| ON | OFF :: = Select ON to enable trace memory display, and OFF to disable trace memory display.

:TRACe:SAVE

The :TRACe:SAVE command saves the current display and front-panel setup in the trace memory specified.

For traces 1 to 3, the trace is saved in the non-compressed state. For traces 4 through 100, the trace is saved in 64 KB of nonvolatile trace memory with data compression. Storage time for compressed data is less than ten seconds.

The operation of the :TRACe:SAVE command is identical to the `Save to Trace` softkey. Refer to "To Save or Recall Traces" in Chapter 2 for additional information.

Command Syntax:	:TRACe:SAVE <N>
Where:	<N> :: = The trace memory number |1 to 100|
Example:	OUTPUT 707;":TRAC:SAVE 2"

:TRIGger:COUPling

The :TRIGger:COUPling command selects the input coupling for the selected external trigger source. The coupling can be set to AC or DC.

The :COUPling query returns the current selection.

Command Syntax:	:TRIGger:COUPling {AC | DC}
Example:	OUTPUT 707;":TRIGGER:COUPLING DC"
Query Syntax:	:TRIGger:COUPling?
Returned Format:	{AC | DC}<NL>
Example:	DIM Mode$[50] OUTPUT 707;":TRIG:COUP?" ENTER 707;Mode$ PRINT Mode$

:TRIGger:HOLDoff

The :TRIGger:HOLDoff command allows a holdoff value to be entered in terms of time.

The :HOLDoff query returns the value of the holdoff time for the current mode.

Command Syntax:	:TRIGger:HOLDoff <holdoff_time>
Where:	<holdoff_time> ::= 200 ns to 13.42 s
Example:	OUTPUT 707;":TRIGGER:HOLDOFF 216 NS"
Query Syntax:	:TRIGger:HOLDoff?
Returned Format:	<holdoff_time><NL>
Where:	<holdoff_time> ::= 200 ns to 13.42 s |exponential-NR3 format|
Example:	DIM Ho[50] OUTPUT 707;":TRIGGER:HOLD?" ENTER 707;Ho PRINT Ho

:TRIGger:LEVel

The :TRIGger:LEVel command sets the trigger level voltage or power of the active trigger. If the optical channel is the active trigger, the trigger level can be specified in watts.

The :LEVel query returns the trigger level of the current trigger mode.

Command Syntax:	:TRIGger:LEVel <level> [W, mW, uW or V, mV, uV]
Where:	<level> ::= for internal triggers, ±1.5 × full-scale value from center screen. <level> ::= for external triggers, ±2 volts with probe attenuation at 1:1.
Examples:	OUTPUT 707;":TRIGGER:LEVEL .30" OUTPUT 707;":TRIGGER:LEV 300MV" OUTPUT 707;":TRIG:LEV 3E-1"
Query Syntax:	:TRIGger:LEVel?
Returned Format:	<level><NL>
Where:	<level> ::= trigger level in volts or watts [exponential–NR3 format]
Example:	DIM TLevel[50] OUTPUT 707;":TRIGGER:LEVEL?" ENTER 707;TLevel PRINT TLevel

:TRIGger:MODE

The :TRIGger:MODE command selects the trigger mode.

If trigger is lost or AUTLEVEL command is sent, the trigger level is set to either center of the signal (when DC coupled) or to middle of screen (when AC coupled).

If the AUTO mode is selected, a baseline is provided on the display in the absence of a signal. If a signal is present but the analyzer is not triggered, the unsynchronized signal is displayed instead of a baseline.

If the NORMAL mode is selected and no trigger is present, the instrument does not sweep, and the data acquired on the previous trigger remains on the screen.

If the SINGLE mode is selected, the screen is cleared and the instrument is stopped. The :RUN command arms the trigger, and data is acquired when the trigger is found. The :RUN command must be sent to make a single acquisition.

The :MODE query returns the current trigger mode.

Command Syntax:	:TRIGger:MODE {AUTLevel | AUTO | NORMal | SINGle | TV}
Example:	OUTPUT 707;":TRIG:MODE AUTO"
Query Syntax:	:TRIGger:MODE?
Returned Format:	{ AUTLevel | AUTO | NORMal | SINGle | TV }<NL>
Example:	DIM Mode$[30] OUTPUT 707;":TRIGGER:MODE?" ENTER 707;Mode$ PRINT Mode$

:TRIGger:NREJect

The :TRIGger:NREJect command allows the noise reject filter to be turned on or off.

The :NREJect query returns the current mode of the noise reject filter.

Command Syntax:	:TRIGger:NREJect {OFF | ON}
Example:	OUTPUT 707;":TRIGGER:NREJ ON"
Query Syntax:	:TRIGger:NREJect?
Returned Format:	{OFF | ON}<NL>
Example:	DIM Nrej$[50] OUTPUT 707;":TRIGGER:NREJECT?" ENTER 707;Nrej$ PRINT Nrej$

:TRIGger:POLarity

The :TRIGger:POLarity command sets the polarity for the TV trigger.

The :POLarity query returns the TV trigger polarity.

Command Syntax:	:TRIGger:POLarity {POSitive | NEGative}
Example:	OUTPUT 707;":TRIGGER:POL POS"
Query Syntax:	:TRIGger:POLarity?
Returned Format:	{POS | NEG}<NL>
Example:	DIM Pol$[30] OUTPUT 707;":TRIG:POL?" ENTER 707;Pol$ PRINT Pol$

:TRIGger:REJect

The :TRIGger:REJect command allows the low frequency or high frequency reject filter to be turned on or off. One of the two filters can be turned on.

The :REJect query returns the current status of the two filters.

Command Syntax:	:TRIGger:REJect {OFF | LF | HF}
Example:	OUTPUT 707;":TRIGGER:REJ HF"
Query Syntax:	:TRIGger:REJect?
Returned Format:	{OFF | LF | HF}<NL>
Example:	DIM Re$[50] OUTPUT 707;":TRIG:REJ?" ENTER 707;Re$ PRINT Re$

:TRIGger:SETup

The :TRIGger:SETup query returns all current settings of the commands in the Trigger subsystem.

Query Syntax:	:TRIGger:SETup?
Returned Format:	TRIG:MODE {AUT | AUTO | NORM | SING | TV}; SOURCE {CHAN{1|2} | EXT | LINE}; LEVEL<level>; HOLD<time>; SLOPE{POS | NEG}; COUP{AC | DC}; REJ {OFF | LF | HF}; NREJ {ON | OFF}; POL{POS | NEG}; TVMODE {FIELD1 | FIELD2 | LINE | VERT}; TVHF{ON | OFF}<NL>
Where:	<level>::= trigger level in volts or watts |exponential-NR3 format| <time>::= 200 ns to 13.42 s |exponential-NR3 format|
Example:	DIM Set$[200] OUTPUT 707;":TRIG:SET?" ENTER 707;Set$ PRINT Set$

:TRIGger:SLOPe

The :TRIGger:SLOPe command specifies the slope of the edge for the trigger.

The :SLOPe query returns the current trigger slope.

Command Syntax:	:TRIGger:SLOPe {NEGative | POSitive}
Example:	OUTPUT 707;":TRIGGER:SLOPE POSITIVE"
Query Syntax:	:TRIGger:SLOPe?
Returned Format:	{NEG | POS}<NL>
Example:	DIM Ts$[50] OUTPUT 707;":TRIG:SLOP?" ENTER 707;Ts$ PRINT Ts$

:TRIGger:SOURce

The :TRIGger:SOURce command selects the channel that actually produces the trigger. Channels 1, 2, external and line can be selected.

The :SOURce query returns the current source.

Command Syntax:	:TRIGger:SOURce {CHANnel1 | CHANnel2 | EXTernal | LINE}
Example:	OUTPUT 707;":TRIGGER:SOURCE CHAN2"
Query Syntax:	:TRIGger:SOURce?
Returned Format:	{CHAN1 | CHAN2 | EXT | LINE}<NL>
Example:	DIM Src$[30] OUTPUT 707;":TRIGGER:SOURCE?" ENTER 707;Src$ PRINT Src$

:TRIGger:TVHFrej

The :TRIGger:TVHFrej command allows the high frequency filter in the TV trigger mode to be turned on or off.

The :TVHFrej query returns the current mode of the high frequency filter in the TV trigger mode.

Command Syntax:	:TRIGger:TVHFrej {OFF | ON}
Example:	OUTPUT 707;":TRIG:TVHF ON"
Query Syntax:	:TRIGger:TVHFrej?
Returned Format:	{OFF | ON}<NL>
Example:	DIM TVhf$[30] OUTPUT 707;":TRIG:TVHF?" ENTER 707;TVhf$ PRINT TVhf$

:TRIGger:TVMode

The :TRIGger:TVMode command selects the TV trigger mode. There are four TV trigger modes: LINE, FIELD1, FIELD2, and VERTical.

The :TVMode query returns the current TV trigger mode.

Command Syntax:	:TRIGger:TVMode {LINE | FIELD1 | FIELD2 | VERTical}
Example:	OUTPUT 707;":TRIGGER:TVMODE LINE"
Query Syntax:	:TRIGger:TVMode?
Returned Format:	{LINE | FIELD1 | FIELD2 | VERT}<NL>
Example:	DIM TVms$[3 0] OUTPUT 707;":TRIGGER:TVMODE?" ENTER 707;TVm$ PRINT TVm$

:VIEW

The :VIEW command turns on (starts displaying) an active channel. :VIEW also turns on the external trigger view.

To display a channel use the command :VIEW:CHANnel{1 | 2}. To display the external trigger view use the command :VIEW:EXTernal.

Use the :BLANk command to turn off (stop displaying) a specified channel. The :BLANk command also turns off the external trigger view.

Command Syntax:	:VIEW <display>
Where:	<display> ::= {CHANnel{1 | 2} |PMEMory{1 | 2} | EXTernal}
Example:	OUTPUT 707;":VIEW CHANNEL1"

:WAVeform:BYTeorder

The :WAVeform:BYTeorder command sets the output sequence of the word data. The parameter MSBFIRST sets the most significant byte to be transmitted first. The parameter LSBFIRST sets the least significant byte to be transmitted first. This command affects the transmitting sequence only when :WAVeform:FORMat WORD is selected.

The :BYTeorder query returns the current output sequence.

Command Syntax:	:WAVeform:BYTeorder {LSBFirst | MSBFirst}
Example:	OUTPUT 707;":WAVEFORM:BYTEORDER MSBF"
Query Syntax:	:WAVeform:BYTeorder?
Returned Format:	{LSBF |MSBF}<NL>
Example:	DIM Byt$[30] OUTPUT 707;":WAV:BYT?" ENTER 707;Byt$ PRINT Byt$

:WAVeform:DATA

The :DATA query outputs a waveform record to the controller over the interface that is stored in a channel buffer previously specified with the :WAVeform:SOURce command. An example of the :DATA query is included in Chapter 8.

Command Syntax:	:WAVeform:DATA <binary block data in # format>
Query Syntax:	:WAVeform:DATA?
Returned Format:	<binary block length bytes><binary block><NL>

:WAVeform:FORMat

The :WAVeform:FORMat command sets the data transmission mode for waveform data output. This command controls how the data is formatted when sent from the analyzer.

When the ASCII mode is selected, the data consists of ASCII digits with each data value separated by a comma.

WORD formatted data transfers as two bytes, with the upper byte set to 0. The :BYTeorder command can be used to specify whether the upper byte or lower byte is transmitted first. The default (no command sent) is the upper byte transmitted first.

BYTE formatted data is transferred as 8-bit bytes.

The :FORMat query returns the current output format for the transfer of waveform data.

Command Syntax:	:WAVeform:FORMat {ASCii | WORD | BYTE}
Example:	OUTPUT 707;":WAVEFORM:FORMAT WORD"
Query Syntax:	:WAVeform:FORMat?
Returned Format:	{ASC | WORD | BYTE}<NL>
Example:	DIM Frmt$[30] OUTPUT 707;":WAV:FORMAT?" ENTER 707;Frmt$ PRINT Frmt$

:WAVeform:POINts

The :WAVeform:POINts command sets the number of points to be transferred using the :WAVeform:DATA query. The number of time buckets available is returned by the :ACQuire:POINts query. The points value is the number of time buckets contained in the waveform selected with the :WAVeform:SOURce command.

The :POINts query returns the number of points to be transferred when using the :WAVeform:DATA query.

Command Syntax:	:WAVeform:POINts <value>
Where:	<value> ::= {100 | 200 | 250 | 400 | 500 | 800 | 1000 | 2000 | 4000}
Example:	OUTPUT 707;":WAVEFORM:POINTS 1000"
Query Syntax:	:WAVeform:POINts?
Returned Format:	{100 | 200 | 250 | 400 | 500 | 800 |1000 | 2000 | 4000} <NL>
Example:	OUTPUT 707;":WAV:POINTS?" ENTER 707;Pts PRINT Pts

:WAVeform:PREamble

The :PREamble query sends a waveform preamble to the controller from the waveform source. The preamble data contains information concerning the vertical and horizontal scaling of the data of the corresponding channel.

Query Syntax: :WAVeform:PREamble?

Returned Format: <preamble block><NL>

Where: <preamble block> ::=
<format NR1>,<type NR1>,<points NR1>, <count NR1>,
<xincrement NR3>,<xorigin NR3>,<xreference NR1>,
<yincrement NR3>,<yorigin NR3>,<yreference NR1>

Where: <format> ::=
0 for ASCII format
1 for BYTE format
2 for WORD format

<type> ::=
0 for AVERAGE type
1 for NORMAL type
2 for PEAK DETECT type

<count>::= 1, always 1 and is present for compatibility only

Example:
```
DIM Preamble[1:10]
OUTPUT 707;":WAVEFORM:PREAMBLE?"
ENTER 707 ;Preamble[*]
OUTPUT 707;":WAVEFORM:PREAMBLE ";Preamble[*]
END
```

:WAVeform:SOURce

The :WAVeform:SOURce command selects the channel and function to be used as the source for the waveform commands.

The :SOURce query returns the currently selected source for the waveform commands.

Command Syntax:	:WAVeform:SOURce CHANnel {1 | 2} FUNCtion {1 | 2}
Example:	OUTPUT 707;":WAV:SOURCE CHANNEL2"
Query Syntax:	:WAVeform:SOURce?
Returned Format:	CHAN{1 | 2} FUNCtion {1 | 2} <NL>
Example:	DIM Src$[30] OUTPUT 707;":WAVEFORM:SOURCE?" ENTER 707;Src$ PRINT Src$

:WAVeform:TYPE

The :TYPE query returns the currently selected acquisition mode.

The acquisition mode is set by the :ACQuire:TYPE command.

Query Syntax:	:WAVeform:TYPE?
Returned Format:	{NORM | PEAK | AVER}<NL>
Example:	DIM Type$[30] OUTPUT 707;":WAVEFORM:TYPE? ENTER 707;Type$ PRINT Type$

:WAVeform:XINCrement

The :XINCrement query returns the current x-increment value in the preamble for the current specified source. This value is the time difference between consecutive data points.

Query Syntax:	:WAVeform:XINCrement?
Returned Format:	<value><NL>
Where:	<value> ::= x-increment in the current preamble |exponential–NR3 format|
Example:	OUTPUT 707;":WAV:XINCREMENT?" ENTER 707;Xin PRINT Xin

:WAVeform:XORigin

The :XORigin query returns the current x-origin value in the preamble for the current specified source. :XORigin is always the first data point in the memory.

Query Syntax:	:WAVeform:XORigin?
Returned Format:	<value><NL>
Where:	<value> ::= x-origin value in the current preamble |exponential–NR3 format|
Example:	OUTPUT 707;":WAV:XORIGIN?" ENTER 707;Xor PRINT Xor

:WAVeform:XREFerence

The :XREFerence query returns the current x-reference value in the preamble for the current specified source. This value specifies the data point associated with the x-origin data value.

Query Syntax:	:WAVeform:XREFerence?
Returned Format:	<value><NL>
Where:	<value> ::= x-reference value in the current preamble |integer–NR1 format|
Example:	OUTPUT 707;":WAV:XREFERENCE?" ENTER 707;Xrf PRINT Xrf

:WAVeform:YINCrement

The :WAVeform:YINCrement query returns the current y-increment value in the preamble for the current specified source. This value is the voltage or power difference between consecutive data points.

Query Syntax:	:WAVeform:YINCrement?
Returned Format:	<value><NL>
Where:	<value> ::= y-increment value in the current preamble |exponential–NR3 format|
Example:	OUTPUT 707;":WAV:YINCREMENT?" ENTER 707;Yin PRINT Yin

:WAVeform:YORigin

The :YORigin query returns the current y-origin value in the preamble for the current specified source. This value is the voltage or power at the center of the screen.

Query Syntax:	:WAVeform:YORigin?
Returned Format:	<value><NL>
Where:	<value> ::= y-origin in the current preamble |exponential —NR3 format|
Example:	OUTPUT 707;":WAV:YORIGIN?" ENTER 707;Yorg PRINT Yorg

:WAVeform:YREFerence

The :YREFerence query returns the current y-reference value in the preamble for the current specified source. This value specifies the data point where the y-origin occurs.

Query Syntax:	:WAVeform:YREFerence?
Returned Format:	<value><NL>
Where:	<value> ::= y-reference value in the current preamble |integer–NR1 format|
Example:	OUTPUT 707;":WAV:YREFERENCE?" ENTER 707;Yrf PRINT Yrf

8

Programming Examples

Programming Examples

What you'll find in this chapter

This chapter contains example programs using the command set for the HP 83475B Lightwave Communications Analyzer. In general, they use the long form of the command with each command having a separate output statement for clarity. To optimize speed, switch to the concatenated short form.

Throughout these examples, for analyzers with an HP-IB interface module, the address of the analyzer is assumed to be 7, and the system bus at 700. For analyzers with a RS-232 interface module, the baud rate of the analyzer is assumed to be 9600, and the communication port in the PC to be COM1 and the communication port for HP 200/300 controller is at address 9.

The input signal used is the AC CAL signal from the front panel of the instrument connected to channel 1 with a 10:1 probe.

Two of the examples demonstrate the general purpose features of the analyzer. The examples, for both HP-IB and RS-232 interfaces, are provided for the following platforms:

- HP BASIC 5.0 running on an HP Series 200/300 controller.
- Microsoft QuickBASIC 4.5 running on IBM PC or compatible.
- Microsoft QuickC 2.5 running on IBM PC or compatible.

Other examples demonstrate the communications measurement capabilities of the analyzer and mask programs. These examples are provided in IBASIC for Windows^{TM} A.00 for both HP-IB and RS-232 interfaces.

Introduction

Two 3.5 inch, 1.44 Mb floppy disks are included with this manual. One disk contains files of the examples in this chapter and the other disk contains examples of user defined mask programs.

The example programs for HP BASIC have been converted from LIF format to the DOS format. To retrieve the program, you need to convert the DOS format back to LIF format before the program can be read by the HP Series 200/300 controller. The files are stored in sub-directories as listed below:

- HP-HPIB contains examples on HPBASIC using HP-IB interface.
- HP-RS232 contains examples on HPBASIC using RS-232 interface.
- IB-HPIB contains communications test examples in IBASIC for Windows, using HP-IB interface.
- IB-RS232 contains communications test examples in IBASIC for Windows, using RS-232 interface.
- QB-HPIB contains examples on QuickBASIC using HP-IB interface.
- QB-RS232 contains examples on QuickBASIC using RS-232 interface.
- QC-HPIB contains examples on QuickC using HP-IB interface.

The files on the second disk, examples of user defined mask programs, are stored in the root directory.

To use the QuickBASIC and QuickC programs on the HP-IB interface, you must have the HP 82335A HP-IB Command Library installed in your PC. See the manual for 82335A for how to use the library with QuickBASIC and QuickC.

There are more example files contained in the diskette. In each sub-directory, there is a text file called README. It contains the latest information concerning the example files.

This sample program demonstrates some of the commands used to setup the HP-IB interface and analyzer, then makes a simple measurement.

```
10    !\HP_HPIB\INIT.BAS       program in HP Basic for HP-IB interface
20    !
30    !MAIN PROGRAM
40    !
50    GOSUB Initialize                !initialize interface and analyzer
60    GOSUB Get_waveform              !digitize signal
70    GOSUB Measure                   !measure and print frequency
80    STOP
90    !
100   !INITIALIZE INTERFACE AND ANALYZER
110   !
120 Initialize:                       !
130   ASSIGN @Anlzr TO 707            !analyzer address
140   ASSIGN @Isc TO 7                !HP-IB address
150   CLEAR @Isc                      !clear HPIB interface
160   OUTPUT @Anlzr;"*RST"            !set analyzer to default config
170   OUTPUT @Anlzr;":AUTOSCALE"      !AUTOSCALE
180   OUTPUT @Anlzr;":CHAN1:PROBE X10" !set probe to 10
190   CLEAR SCREEN                    !clear screen
200   RETURN
210     !
220     !DIGITIZE waveform to acquire data and stop analyzer for further
230     !measurement. Measurement is NOT displayed on front panel.
240     !
250 Get_waveform:                             !
260   OUTPUT @Anlzr;":WAVEFORM:SOURCE CHAN1" !set source to channel 1
270   OUTPUT @Anlzr;":DIGITIZE CHAN1"        !macro to acquire data & stop
280   RETURN
290     !
300     !make frequency measurement and read results into computer
310     !
320 Measure:                                  !
330   OUTPUT @Anlzr;":MEASURE:FREQUENCY?''    !FREQUENCY query
340   ENTER @Anlzr;Value                      !read from analyzer
350   PRINT "FREQUENCY = ";Value;"Hz"
360   RETURN
370   END
```

This program digitizes and transfers waveform data to the computer, saves the data to a file, retrieves it from a file, and display it on the computer's screen. The integrate routine performs an integration math function and display the results.

```
10    !\HP_HPIB\DIGI.BAS        HP Basic program using HPIB interface
20    !
30    !MAIN PROGRAM
40    !
50    REAL Preamble(1:10)               !array for preamble information
60    !
70    GOSUB Initialize                  !initialize interface and analyzer
80    GOSUB Get_waveform                !dig & acquire signal
90    GOSUB Save_waveform               !store to disk
100   !
110   PRINT "The waveform data and preamble information have now been"
120   PRINT "read from the analyzer and stored in the computer's disk."
130   PRINT "This information will now be retrieved from the disk, and"
140   PRINT "will be used to plot the waveform, calculate and plot the"
150   PRINT "integral, as well as calculate scaling information."
160   PRINT "Press CONTINUE to continue."
170   PAUSE
180   GOSUB Retrieve_wave               !retrieve from disk
190   GOSUB Graph                       !draw waveform
200   GOSUB Integrate                   !integrate & draw
210   STOP
220   !
230   !INITIALIZE INTERFACE AND ANALYZER
240 Initialize:                         !
250   ASSIGN @Isc TO 7                  !Interface Select Code = 7
260   ASSIGN @Anlzr TO 707              !analyzer address
270   ASSIGN @Fast TO 707;FORMAT OFF    !turn controller formatting off
280   CLEAR @Isc                        !clear HPIB interface
290   OUTPUT @Anlzr;"*RST"              !set analyzer to default config
300   OUTPUT @Anlzr;":AUTOSCALE"        !AUTOSCALE
310   OUTPUT @Anlzr;":CHAN1:PROBE X10"  !set probe to 10
320   CLEAR SCREEN
330   RETURN
340   !
350   !complete analyzer configuration; DIGITIZE and acquire waveform data
360   !and preamble (voltage/timing) information into computer.
370   !
380 Get_waveform:                       !
390     !
400     !the following 4 commands are the default configuration setting
```

```
410     !that the RST sets up; but, they are included here for clarity and
420     !completeness. This ensures analyzer is configured for DIGITIZE, if
430     !*RST was not done.
440     !
450   OUTPUT @Anlzr;":WAVEFORM:SOURCE CHAN1"  !set source to channel 1
460   OUTPUT @Anlzr;":ACQUIRE:TYPE NORMAL"    !set to normal acquisition mode
470   OUTPUT @Anlzr;":ACQUIRE:COMPLETE 100"   !set complete criteria
480   OUTPUT @Anlzr;":WAVEFORM:POINTS 4000"   !set # of pts to 4000
490     !
500   OUTPUT @Anlzr;":WAVEFORM:FORMAT WORD"   !acquired data in word format
510   OUTPUT @Anlzr;":DIGITIZE CHAN1"         !macro to acquire data & stop
520   OUTPUT @Anlzr;":WAVEFORM:DATA?"         !query analyzer for data
530   ENTER @Anlzr USING "#,2A,8D";Header$,Bytes  !strip off header & size
540   PRINT "reading";Bytes;" bytes from analyzer"
550   !
560   !allocate an array for the waveform data. After the array has been
570   !read in, one extra byte read is done to input the line feed (10)
580   !attached to the end of the analyzer's output buffer.
590   !
600   ALLOCATE INTEGER Waveform (1:Bytes/2)
610   ENTER @Fast;Waveform(*)              !read waveform information
620   ENTER @Anlzr USING "-K,B";End$       !get last byte (line feed)
630   OUTPUT @Anlzr;":WAVEFORM:PREAMBLE?"  !query for preamble
640   ENTER @Anlzr;Preamble(*)             !read preamble information
650   RETURN
660   !
670   !plot waveform data to display
680   !
690 Graph:                                 !
700   GCLEAR                               !initialize graphics
710   CLEAR SCREEN
720   GINIT
730   GRAPHICS ON
740   VIEWPORT 0,130,35,100
750   WINDOW 1,Preamble(3),0,255
760   FRAME
770   PEN 4
780   MOVE 0,0
790   FOR I=1 TO Preamble(3)              !plot data points
800     MOVE I,Waveform(I)
810     DRAW I,Waveform(I)
820   NEXT I
830   PRINT TABXY(0,18), "V/Div =";(32*Preamble(8));" V";TAB(50),
        "Offset = ";(128-Preamble(10))*Preamble(8)+Preamble(9);" V"
840   PRINT TABXY(0,19),"S/Div =";Preamble(3)*Preamble(5)/10;" S";TAB(50),
        "Delay = ";(Preamble(3)/2-Preamble(7))*Preamble(5)+Preamble(6);" S"
```

```
850   RETURN
860   !
870   !calculate and plot the integral of the waveform using the preamble
880   !
890 Integrate                              !
900   REAL Math(0:4000)
910   MOVE 0,0
920   Math(0)=0
930   FOR I=0 TO Preamble(3)-1
940    Math(I+1)=Math(I)+(Waveform(I+1)-Preamble(10))*Preamble(8)+Preamble(9)
950   NEXT I
960   Max=Math(1)
970   Min=Math(1)
980   FOR I=1 TO Preamble(3)
990     IF Math(I)Max THEN Max=Math(I)
1000    IF Math(I)    THEN Min=Math(I)
1010  NEXT I
1020  PEN 5
1030    !plot integral
1040  WINDOW 1,Preamble(3),Min,Max
1050  FOR I=1 TO Preamble(3)
1060    MOVE I,Math(I)
1070    DRAW I,Math(I)
1080  NEXT I
1090  RETURN
1100    !
1110    !save waveform data and preamble information to computer disk
1120    !
1130 Save_waveform:                         !
1140  ON ERROR GOTO 1160
1150  PURGE "WAVESAMPLE"
1160  OFF ERROR
1170  CREATE BDAT "WAVESAMPLE",1,4080
1180  ASSIGN @Path TO "WAVESAMPLE"
1190  OUTPUT @Path;Waveform(*),Preamble(*)
1200  RETURN
1210   !
1220   !retrieve waveform data and preamble information from disk
1230   !
1240 Retrieve_wave:                         !
1250  ASSIGN @Path TO "WAVESAMPLE"
1260  ENTER @Path;Waveform(*),Preamble(*)
1270  RETURN
1280  END
```

This sample program demonstrates some of the commands used to setup the RS-232 interface and analyzer, then makes a simple measurement.

```
10     !\HP_RS232\INIT.232        program in HP basic for RS232 interface
20     !
30     !MAIN PROGRAM
40     !
50     GOSUB Initialize                    !initialize interface and analyzer
60     GOSUB Get_waveform                  !digitize signal
70     GOSUB Measure                       !measure and print frequency
80     STOP
90     !
100    !INITIALIZE INTERFACE AND ANALYZER
110    !
120 Initialize:                            !
130    CONTROL 9,3;9600                    !set RS232 baud rate to 9600
140    CONTROL 9,4;3                       !8 bit;1stop;par disable;odd par
150    ASSIGN @Anlzr TO 9                  !analyzer address
160    OUTPUT @Anlzr;"*RST"                !set analyzer to default config
170    OUTPUT @Anlzr;":AUTOSCALE"          !AUTOSCALE
180    OUTPUT @Anlzr;":CHAN1:PROBE X10"    !set probe to 10
190    CLEAR SCREEN                        !clear screen
200    RETURN
210      !
220      !DIGITIZE waveform to acquire data and stop analyzer for further
230      !measurement. Measurement is NOT displayed on front panel.
240      !
250 Get_waveform                                  !
260    OUTPUT @Anlzr;":WAVEFORM:SOURCE CHAN1" !set source to channel 1
270    OUTPUT @Anlzr;":DIGITIZE CHAN1"        !macro to acquire data & stop
280    RETURN
290      !
300      !made a frequency measurement and read results into computer
310      !
320 Measure:                                  !
330    OUTPUT @Anlzr;":MEASURE:FREQUENCY?"    !FREQUENCY query
340    ENTER @Anlzr;Value                     !read from analyzer
350    PRINT "FREQUENCY = ";Value;"Hz"
360    RETURN
370    END
```

This program digitizes and transfers waveform data to the computer, saves the data to a file, retrieves it from a file, and displays it on the computer's screen. The integrate routine performs an integration math function and displays the results.

```
10    !\HP_RS232\DIGI.232       HP Basic program using RS232 interface
20    !RS232 program is very similar to HP-IB example. Need to initilize port
30    !port 9 and set @Anlzr to 9.
40    !
50    !MAIN PROGRAM
60    !
70    REAL Preamble(1:10)             !array for preamble information
80    !
90    GOSUB Initialize                !initialize interface and analyzer
100   GOSUB Get_waveform              !dig & acquire signal
110   GOSUB Save_waveform             !store to disk
120   !
130   PRINT "The waveform data and preamble information have now been"
140   PRINT "read from the analyzer and stored in the computer's disk."
150   PRINT "This information will now be retrieved from the disk, and"
160   PRINT "will be used to plot the waveform, calculate and plot the"
170   PRINT "integral, as well as calculate scaling information."
180   PRINT "Press CONTINUE to continue."
190   PAUSE
200   GOSUB Retrieve_wave             !retrieve from disk
210   GOSUB Graph                     !draw waveform
220   GOSUB Integrate                 !integrate & draw
230   STOP
240   !
250   !INITIALIZE INTERFACE AND ANALYZER
260 Initialize:                         !
270   CONTROL 9,3;9600                  !set RS232 baud rate to 9600
280   CONTROL 9,4;3                     !8 bit;1stop;par disable;odd par
290   ASSIGN @Anlzr TO 9                !analyzer address
300   OUTPUT @Anlzr;"*RST"              !set analyzer to default config
310   OUTPUT @Anlzr;":AUTOSCALE"        !AUTOSCALE
320   OUTPUT @Anlzr;":CHAN1:PROBE x10"  !set probe to 10
330   CLEAR SCREEN
340   RETURN
350   !
360   !complete analyzer configuration; DIGITIZE and acquire waveform data
370   !and preamble (voltage/timing) information into computer.
380   !
390 Get_waveform:                       !
400     !
```

```
410     !the following 4 commands are the default configuration setting
420     !that the RST sets up; but, they are included here for clarity and
430     !completeness. This ensures analyzer is configured for DIGITIZE, if
440     !*RST was not done.
450     !
460   OUTPUT @Anlzr;":WAVEFORM:SOURCE CHAN1"  !set source to channel 1
470   OUTPUT @Anlzr;":ACQUIRE:TYPE NORMAL"    !set to normal acquisition mode
480   OUTPUT @Anlzr;":ACQUIRE:COMPLETE 100"   !set complete criteria
490   OUTPUT @Anlzr;":WAVEFORM:POINTS 4000"   !set # of pts to 4000
500     !
510   OUTPUT @Anlzr;":WAVEFORM:FORMAT BYTE"   !acquired data in byte format
520   OUTPUT @Anlzr;":DIGITIZE CHAN1"         !macro to acquire data & stop
530   OUTPUT @Anlzr;":WAVEFORM:DATA?"         !query analyzer for data
540   ENTER @Anlzr USING "#,2A,8D";Header$,Bytes  !strip off header & size
550   PRINT "reading ";Bytes;" bytes from analyzer"
560   !
570   !allocate an array for the waveform data. The array is one byte
580   !more than the number of points to allow for the line feed (10) that
590   !is attached to the end of the output buffer to be read. This
600   !character SHOULD not be used. The value in preamble(3) reports
610   !the correct number of points. Note that all future calculations
620   !utilize preamble(3), the number of points, for loop control.
630   !
640   ALLOCATE INTEGER Waveform(1:Bytes+1)
650   ENTER @Anlzr USING "#,B";Waveform(*)  !read waveform information
660   OUTPUT @Anlzr;":WAVEFORM:PREAMBLE?"   !query for preamble
670   ENTER @Anlzr;Preamble(*)              !read preamble information
680   RETURN
690   !
700   !plot waveform data to display
710   !
720 Graph:                                  !
730   GCLEAR                                !initialize graphics
740   CLEAR SCREEN
750   GINIT
760   GRAPHICS ON
770   VIEWPORT 0,130,35,100
780   WINDOW 1,Preamble(3),0,255
790   FRAME
800   PEN 4
810   MOVE 0,0
820   FOR I=1 TO Preamble(3)                !plot data points
830     MOVE I,Waveform(I)
840     DRAW I,Waveform(I)
850   NEXT I
860   PRINT TABXY(0,18),"V/Div =";(32*Preamble(8));" V";TAB(50),
        "Offset =";(128-Preamble(10))*Preamble(8)+Preamble(9);" V"
```

```
870   PRINT TABXY(0,19),"S/Div =";Preamble(3)*Preamble(5)/10;" S";TAB(50),
        "Delay =";(Preamble(3)/2-Preamble(7))*Preamble(5)+Preamble(6);" S"
880   RETURN
890   !
900   !calculate and plot the integral of the waveform using the preamble
910   !
920 Integrate:                                !
930   REAL Math(0:4000)
940   MOVE 0,0
950   Math(0)=0
960   FOR I=0 TO Preamble(3)-1
970    Math(I+1)=Math(I)+(Waveform(I+1)-Preamble(10))*Preamble(8)+Preamble(9)
980   NEXT I
990   Max=Math(1)
1000  Min=Math(1)
1010  FOR I=1 TO Preamble(3)
1020    IF Math(I)Max THEN Max=Math(I)
1030    IF Math(I)  THEN Min=Math(I)
1040  NEXT I
1050  PEN 5
1060    !plot integral
1070  WINDOW 1,Preamble(3),Min,Max
1080  FOR I=1 TO Preamble(3)
1090    MOVE I,Math(I)
1100    DRAW I,Math(I)
1110  NEXT I
1120  RETURN
1130    !
1140    !save waveform data and preamble information to computer disk
1150    !
1160 Save_waveform:                           !
1170  ON ERROR GOTO 1190
1180  PURGE "WAVESAMPLE"
1190  OFF ERROR
1200  CREATE BDAT "WAVESAMPLE",1,4080
1210  ASSIGN @Path TO "WAVESAMPLE"
1220  OUTPUT @Path;Waveform(*),Preamble(*)
1230  RETURN
1240   !
1250   !retrieve waveform data and preamble information from disk
1260   !
1270 Retrieve_wave:                           !
1280  ASSIGN @Path TO "WAVESAMPLE"
1290  ENTER @Path;Waveform(*),Preamble(*)
1300  RETURN
1310  END
```

This program sets up the instrument and performs a lightwave transmitter conformance test. The test consists of a mask conformance measurement for the SDH/SONET STM-1/OC-3 standard, an extinction ratio measurement, and an average optical power measurement. The measurement results are analyzed and the result reported.

```
10    !\IB_HPIB\CONF.BAS          HP IBasic program using HPIB interface
20    !
30    !MAIN PROGRAM
40    !
50    DIM Res1$[50],Parsed$(1:4)[10]
60    INTEGER Last_match
70    !
80    PRINT "This example program will perform the following tasks:"
90    PRINT "              a.  initialize interface and analyzer"
100   PRINT "              b.  set up and perform an STM-1/OC-3 mask test"
110   PRINT "              c.  set up and perform an extinction ratio test"
120   PRINT "              d.  set up and perform an average power measurement"
130   PRINT "              e.  report the conformace test results"
140   PRINT
150   PRINT "The program performs a typical transmitter conformance test for"
160   PRINT "the STM-1/OC-3 SDH/SONET rate of 155.52 Mb/s.  It assumes the"
170   PRINT "following conditions:"
180   PRINT
190   PRINT "               1. a 1330 nm signal at the optical input"
200   PRINT "               2. a 155.52 Mb/s random or pseudo-random optical input"
210   PRINT "               3. an external trigger input signal"
220   PRINT
230   PRINT "Click Continue when ready to start"
240   PAUSE
250   GOSUB Initialize                    !initialize interface and analyzer
260   GOSUB Mask_test                     !perform a mask test
270   GOSUB Ext_ratio                     !perform an extinction ratio test
280   GOSUB Avg_power                     !perform an average power measurement
290   GOSUB Report                        !report results
300   !
310   STOP
320   !
330   !
340   !INITIALIZE INTERFACE AND ANALYZER
350 Initialize:                          !
360   ASSIGN @Isc TO 7                    !Interface Select Code = 7
```

```
370   ASSIGN @Anlzr TO 707                !analyzer address
380   ASSIGN @Fast TO 707;FORMAT ON       !turn controller formatting on
390   CLEAR @Isc                          !clear HPIB interface
400   OUTPUT @Anlzr;"*RST"                !set analyzer to default config
410   OUTPUT @Anlzr;":CHAN2:INPUT FIFTY" !set chan2 to 50 Ohm input
420   DISP "Attach STM-1/OC-3 reference filter, click Cont when ready"
430   PAUSE
440   OUTPUT @Anlzr;":AUTOSCALE"          !AUTOSCALE
450   OUTPUT @Anlzr;":BLANK CHAN1"        !de-activate Channel 1
460   OUTPUT @Anlzr;":BLANK CHAN2"        !de-activate Channel 2
470   OUTPUT @Anlzr;":VIEW CHAN2"         !activate Channel 2
480   OUTPUT @Anlzr;":TIMEBASE:RANGE 1E-8"!set time base for one eye diagram
490   DISP "Adjust delay to center waveform on screen, click Cont when ready"
500   PAUSE
510   OUTPUT @Anlzr;":MEAS:SOURCE CHAN2"     !set channel 2 as the meas source
520   OUTPUT @Anlzr;":CHAN2:IOPTICAL ON"     !activate the optical channel
530   OUTPUT @Anlzr;":MEAS:MTYPE EYENRZ"     !set the measurement type for eye diagrams
540   OUTPUT @Anlzr;":MEAS:TSAMPLES 400"     !set the # of time histogram samples to 400
550   OUTPUT @Anlzr;":MEAS:ASAMPLES 1000"    !set the # of voltage histogram samples to 1000
560   OUTPUT @Anlzr;":OCAL:SCAL D1310"       !select 1310 nm default calibration
570   OUTPUT @Anlzr;":MEAS:PUNITS dBm"       !set power measurements to dBm
580   CLEAR SCREEN
590   RETURN
600   !
610   ! PERFORM AN STM-1/OC-3 MASK TEST
620 Mask_test:                                !
630   !
640   OUTPUT @Anlzr;":MEASURE:MASK:STEST STM1"  !selects the STM1/OC3 mask
650   OUTPUT @Anlzr;":MEAS:MASK:DEF:TIME 5"     !sets the test time to 5 sec
660   OUTPUT @Anlzr;":MEAS:MASK:DEF:MARGIN 20"  !sets a 20% margin
670   OUTPUT @Anlzr;":MEAS:MASK:DEF:MHTS 0"     !sets 0 as the max # of hits
680   OUTPUT @Anlzr;":MEAS:MASK:DEF:OPASS CONT" !continue acquisition if pass
690   OUTPUT @Anlzr;":MEASURE:MASK:TEST?"       !test is performed
700   ENTER @Anlzr;Res1$                        !read results from the analyzer
710   OUTPUT @Anlzr;"MEASURE:MASK:SHOW OFF"     !de-activate the mask
720   REPEAT
730      I=I+1
740      Delimeter$=","
750      CALL Universal_parse(Delimeter$,Res1$,Parsed$(I),Last_match)
760   UNTIL Last_match
770   CLEAR SCREEN
780   !
790   !
```

```
800   RETURN
810   !
820   ! PERFORM AN EXTINCTION RATIO MEASUREMENT
830 Ext_ratio:                              !
840   !
850   DISP "Disable signal into optical input, click Cont when ready"
860   PAUSE
870   OUTPUT @Anlzr;":OCAL:ERAT:IOPT?"      !calibrate optical chan DC offset
880   ENTER @Anlzr;String$                  !read the results into string
890   !
900   REPEAT                                !begin a loop to check validity of
910   I=I+1                                 !calibration results
920   IF UPC$(TRIM$(String$))="ERR" THEN    !if an error in calibration, re-measure
930      DISP "Calibration error, disable optical input and re-measure, Click Cont when ready"
940      PAUSE
950      OUTPUT @Anlzr;":MEASURE:ERAT:IOPT?"  !calibrate DC offset
960      ENTER @Anlzr;String$                  !read the results into string
970      Last_match=0                          !repeat measurement or abort
980      IF I=3 THEN           !three calibrations are allowed before abort
990         DISP "Unable to calibrate DC offset, measurement aborted"
1000        RETURN
1010     END IF
1020  ELSE
1030     DISP "Input signal into the Optical Channel, click Cont when ready"
1040     PAUSE
1050     DISP "Make sure eye diagram is centered on screen, click Cont when ready"
1060     PAUSE
1070     Last_match=1         !calibration is good, loop set for termination
1080  END IF
1090  UNTIL Last_match        !loop continues if calibration is invalid
1100  !
1110  OUTPUT @Anlzr;":MEAS:EWMIN 30"        !sets measurement for middle 40%
1120  OUTPUT @Anlzr;":MEAS:EWMAX 70"        !of the eye bit period.
1130  OUTPUT @Anlzr;":MEAS:ERUNITS dB"      !set the measurement units to dB
1140  OUTPUT @Anlzr;":MEAS:ERATIO?"         !measure the extinction ratio
1150  ENTER @Anlzr;Res2                     !get and save the result
1160  !
1170  RETURN
1180  !
```

```
1190  !PERFORM AN AVERAGE POWER MEASUREMENT
1200 Avg_power:                              !
1210  !
1220  OUTPUT @Anlzr;":MEAS:OPMETER ON"    !activates the power meter
1230  OUTPUT @Anlzr;"*OPC?"               !operation complete sent to analyzer
1240  ENTER @Anlzr;String$                !reads result to variable
1250  OUTPUT @Anlzr;"MEAS:PMET?"          !queries for a measurement
1260  ENTER @Anlzr;Res3                   !reads measurement into variable
1270  OUTPUT @Anlzr;":MEAS:PMET?"         !queries for another measurement
1280  ENTER @Anlzr;Res4                   !reads other measurement
1290  WHILE ABS(Res4-Res3)>.2
1300    OUTPUT @Anlzr;":MEAS:PMET?"       !The power measurement is
1310    ENTER @Anlzr;Res3                 !iterated until the power
1320    OUTPUT @Anlzr;":MEAS:PMET?"       !meter has settled
1330    ENTER @Anlzr;Res4                 !
1340  END WHILE
1350  OUTPUT @Anlzr;":MEAS:OPMETER OFF"   !de-activate power meter
1360  !
1370  RETURN
1380  !
1390  ! REPORT THE CONFORMANCE TEST RESULTS
1400 Report:                               !
1410  !
1420  IF UPC$(TRIM$(Parsed$(1)))="FAIL" OR Res2<13 OR ABS(Res4)>15 THEN
1430   PRINT "THE TRANSMITTER FAILED THE CONFORMANCE TEST."
1440   PRINT
1450  ELSE
1460   PRINT "THE TRANSMITTER PASSED THE CONFORMACE TEST."
1470   PRINT
1480  END IF
1490  PRINT "THE TEST RESULTS ARE:"
1500  PRINT
1510  PRINT "   FOR THE MASK TEST:  Total hits    =";TRIM$(Parsed$(2))
1520  PRINT "                       Standard hits =";TRIM$(Parsed$(3))
1530  PRINT "                       Margin hits   =";TRIM$(Parsed$(4))
1540  PRINT
1550  PRINT "   EXTINCTION RATIO = ";Res2;" dB"
1560  PRINT "   AVERAGE OPTICAL POWER = ";Res4;" dBm"
1570  !
1580  RETURN
1590  !
1600  END
1610  !
1620  ! SUBROUTINE FOR PARSING DATA STRINGS
1630  SUB Universal_parse(Delimeter$,In_n_out$,String_parsed$,INTEGER Last_match)
1640  !
1650  INTEGER Delim_loc,Len_delim
1660  Len_delim=LEN(Delimeter$)           !length of delimeter determined
1670  Delim_loc=POS(In_n_out$,Delimeter$) !position of delimeter determined
```

```
1680  IF Delim_loc=0 THEN                !if no delimeter found, string is
1690    String_parsed$=In_n_out$         !not parsed.
1700    Last_match=1
1710  ELSE                               !if delimeter present, string is parsed
1720    String_parsed$=TRIM$(In_n_out$[1,Delim_loc-1])
1730    In_n_out$=TRIM$(In_n_out$[Delim_loc+Len_delim])
1740    Last_match=0
1750  END IF
1760  SUBEND
```

This program sets up the instrument and performs a lightwave transmitter conformance test. The test consists of a mask conformance measurement for the SDH/SONET STM-1/OC-3 standard, an extinction ratio measurement, and an average optical power measurement. The measurement results are analyzed and the result reported.

```
10    !\IB_HPIB\CONF.BAS          HP IBasic program using HPIB interface
20    !
30    !MAIN PROGRAM
40    !
50    DIM Res1$[50],Parsed$(1:4)[10]
60    INTEGER Last_match
70    !
80    PRINT "This example program will perform the following tasks:"
90    PRINT "              a.  initialize interface and analyzer"
100   PRINT "              b.  set up and perform an STM-1/OC-3 mask test"
110   PRINT "              c.  set up and perform an extinction ratio test"
120   PRINT "              d.  set up and perform an average power measurement"
130   PRINT "              e.  report the conformace test results"
140   PRINT
150   PRINT "The program performs a typical transmitter conformance test for"
160   PRINT "the STM-1/OC-3 SDH/SONET rate of 155.52 Mb/s.  It assumes the"
170   PRINT "following conditions:"
180   PRINT
190   PRINT "               1. a 1330 nm signal at the optical input"
200   PRINT "               2. a 155.52 Mb/s random or pseudo-random optical input"
210   PRINT "               3. an external trigger input signal"
220   PRINT
230   PRINT "Click Continue when ready to start"
240   PAUSE
250   GOSUB Initialize                    !initialize interface and analyzer
260   GOSUB Mask_test                     !perform a mask test
270   GOSUB Ext_ratio                     !perform an extinction ratio test
280   GOSUB Avg_power                     !perform an average power measurement
290   GOSUB Report                        !report results
300   !
310   STOP
320   !
330   !
340   !INITIALIZE INTERFACE AND ANALYZER
350 Initialize:                          !
360   CONTROL 9,3;9600                    !set RS232 baud rate to 9600
```

```
370   CONTROL 9,4;3                      !8 bit;1stop;par disable;odd par
380   ASSIGN @Anlzr TO 9                 !set analyzer address
390   OUTPUT @Anlzr;"*RST"               !set analyzer to default config
400   OUTPUT @Anlzr;":CHAN2:INPUT FIFTY" !set chan2 to 50 Ohm input
410   DISP "Attach STM-1/OC-3 reference filter, click Cont when ready"
420   PAUSE
430   OUTPUT @Anlzr;":AUTOSCALE"          !AUTOSCALE
440   OUTPUT @Anlzr;":BLANK CHAN1"        !de-activate Channel 1
450   OUTPUT @Anlzr;":BLANK CHAN2"        !de-activate Channel 2
460   OUTPUT @Anlzr;":VIEW CHAN2"         !activate Channel 2
470   OUTPUT @Anlzr;":TIMEBASE:RANGE 1E-8"!set time base for one eye diagram
480   DISP "Adjust delay to center waveform on screen, click Cont when ready"
490   PAUSE
500   OUTPUT @Anlzr;":MEAS:SOURCE CHAN2"     !set channel 2 as the meas source
510   OUTPUT @Anlzr;":CHAN2:IOPTICAL ON"     !activate the optical channel
520   OUTPUT @Anlzr;":MEAS:MTYPE EYENRZ"     !set the measurement type for eye diagrams
530   OUTPUT @Anlzr;":MEAS:TSAMPLES 400"     !set the # of time histogram samples to 400
540   OUTPUT @Anlzr;":MEAS:ASAMPLES 1000"    !set the # of voltage histogram samples to 1000
550   OUTPUT @Anlzr;":OCAL:SCAL D1310"       !select 1310 nm default calibration
560   OUTPUT @Anlzr;":MEAS:PUNITS dBm"       !set power measurements to dBm
570   CLEAR SCREEN
580   RETURN
590   !
600   ! PERFORM AN STM-1/OC-3 MASK TEST
610 Mask_test:                             !
620   !
630   OUTPUT @Anlzr;":MEASURE:MASK:STEST STM1"  !selects the STM1/OC3 mask
640   OUTPUT @Anlzr;":MEAS:MASK:DEF:TIME 5"     !sets the test time to 5 sec
650   OUTPUT @Anlzr;":MEAS:MASK:DEF:MARGIN 20"  !sets a 20% margin
660   OUTPUT @Anlzr;":MEAS:MASK:DEF:MHTS 0"     !sets 0 as the max # of hits
670   OUTPUT @Anlzr;":MEAS:MASK:DEF:OPASS CONT" !continue acquisition if pass
680   OUTPUT @Anlzr;":MEAS:MASK:DEF:OFAIL PRINT"!print waveform on fail
690   OUTPUT @Anlzr;":MEASURE:MASK:TEST?"       !test is performed
700   ENTER @Anlzr;Res1$                        !read results from the analyzer
710   OUTPUT @Anlzr;"MEASURE:MASK:SHOW OFF"     !de-activate the mask
720   REPEAT
730      I=I+1
740      Delimeter$=","
750      CALL Universal_parse(Delimeter$,Res1$,Parsed$(I),Last_match)
760   UNTIL Last_match
770   CLEAR SCREEN
780   !
790   RETURN
800   !
```

```
810   ! PERFORM AN EXTINCTION RATIO MEASUREMENT
820 Ext_ratio:                              !
830   !
840   DISP "Disable signal into optical input, click Cont when ready"
850   PAUSE
860   OUTPUT @Anlzr;":OCAL:ERAT:IOPT?"      !calibrate optical chan DC offset
870   ENTER @Anlzr;String$                  !read the results into string
880   !
890   REPEAT                                !begin a loop to check validity of
900   I=I+1                                 !calibration results
910   IF UPC$(TRIM$(String$))="ERR" THEN    !if an error in calibration, re-measure
920      DISP "Calibration error, disable optical input and re-measure, Click Cont when ready"
930      PAUSE
940      OUTPUT @Anlzr;":MEASURE:ERAT:IOPT?"  !calibrate DC offset
950      ENTER @Anlzr;String$                  !read the results into string
960      Last_match=0                          !repeat measurement or abort
970      IF I=3 THEN           !three calibrations are allowed before abort
980         DISP "Unable to calibrate DC offset, measurement aborted"
990         RETURN
1000     END IF
1010  ELSE
1020     DISP "Input signal into the Optical Channel, click Cont when ready"
1030     PAUSE
1040     DISP "Make sure eye diagram is centered on screen, click Cont when ready"
1050     PAUSE
1060     Last_match=1         !calibration is good, loop set for termination
1070  END IF
1080  UNTIL Last_match        !loop continues if calibration is invalid
1090  !
1100  OUTPUT @Anlzr;":MEAS:EWMIN 30"        !sets measurement for middle 40%
1110  OUTPUT @Anlzr;":MEAS:EWMAX 70"        !of the eye bit period.
1120  OUTPUT @Anlzr;":MEAS:ERUNITS dB"      !set the measurement units to dB
1130  OUTPUT @Anlzr;":MEAS:ERATIO?"         !measure the extinction ratio
1140  ENTER @Anlzr;Res2                     !get and save the result
1150  !
1160  RETURN
1170  !
1180  !PERFORM AN AVERAGE POWER MEASUREMENT
1190 Avg_power:                            !
1200  !
1210  OUTPUT @Anlzr;":MEAS:OPMETER ON"    !activates the power meter
1220  OUTPUT @Anlzr;"*OPC?"               !operation complete sent to analyzer
1230  ENTER @Anlzr;String$                !reads result to variable
1240  OUTPUT @Anlzr;"MEAS:PMET?"          !queries for a measurement
1250  ENTER @Anlzr;Res3                   !reads measurement into variable
1260  OUTPUT @Anlzr;":MEAS:PMET?"         !queries for another measurement
1270  ENTER @Anlzr;Res4                   !reads other measurement
```

```
1280  WHILE ABS(Res4-Res3)>.2
1290    OUTPUT @Anlzr;":MEAS:PMET?"       !The power measurement is
1300    ENTER @Anlzr;Res3                 !iterated until the power
1310    OUTPUT @Anlzr;":MEAS:PMET?"       !meter has settled
1320    ENTER @Anlzr;Res4                 !
1330  END WHILE
1340  OUTPUT @Anlzr;":MEAS:OPMETER OFF"   !de-activate power meter
1350  !
1360  RETURN
1370  !
1380  ! REPORT THE CONFORMANCE TEST RESULTS
1390 Report:                              !
1400  !
1410  IF UPC$(TRIM$(Parsed$(1)))="FAIL" OR Res2<13 OR ABS(Res4)>15 THEN
1420   PRINT "THE TRANSMITTER FAILED THE CONFORMANCE TEST."
1430   PRINT
1440  ELSE
1450   PRINT "THE TRANSMITTER PASSED THE CONFORMACE TEST."
1460   PRINT
1470  END IF
1480  PRINT "THE TEST RESULTS ARE:"
1490  PRINT
1500  PRINT "   FOR THE MASK TEST:  Total hits    =";TRIM$(Parsed$(2))
1510  PRINT "                       Standard hits =";TRIM$(Parsed$(3))
1520  PRINT "                       Margin hits   =";TRIM$(Parsed$(4))
1530  PRINT
1540  PRINT "   EXTINCTION RATIO = ";Res2;" dB"
1550  PRINT "   AVERAGE OPTICAL POWER = ";Res4;" dBm"
1560  !
1570  RETURN
1580  !
1590  END
1600  !
```

```
1610  ! SUBROUTINE FOR PARSING DATA STRINGS
1620  SUB Universal_parse(Delimeter$,In_n_out$,String_parsed$,INTEGER Last_match)
1630  !
1640  INTEGER Delim_loc,Len_delim
1650  Len_delim=LEN(Delimeter$)           !length of delimeter determined
1660  Delim_loc=POS(In_n_out$,Delimeter$) !position of delimeter determined
1670  IF Delim_loc=0 THEN                 !if no delimeter found, string is
1680    String_parsed$=In_n_out$          !not parsed.
1690    Last_match=1
1700  ELSE                                !if delimeter present, string is parsed
1710    String_parsed$=TRIM$(In_n_out$[1,Delim_loc-1])
1720    In_n_out$=TRIM$(In_n_out$[Delim_loc+Len_delim])
1730    Last_match=0
1740  END IF
1750  SUBEND
```

This program sets up the HP-IB interface and the analyzer, then makes a simple measurement.

```
DECLARE SUB get.waveform ()
DECLARE SUB measure ()
DECLARE SUB Initialize ()
DECLARE SUB send (cmd$)
'Filename : \QB-HPIB\INIT.BAS

REM $INCLUDE: 'c:\hpib\qbsetup'
DIM SHARED isc&, Anlzr&
CALL Initialize                          'Initialize interface and analyzer
CALL get.waveform                        'Tell analyzer to acquire data
CALL measure                             'Measure the freq. of the input signal
END

SUB get.waveform
   send (":digitize channel1")           'tell analyzer to acquire data
END SUB

SUB Initialize
SHARED PCIC.ERR, PCIB.BASERR, NOERR
   isc& = 7                             'select code of HP-IB interface
   Anlzr& = isc& * 100 + 7              'address of the analyzer
   CALL ioreset(isc&)                   'set interface to start-up state
   IF PCIB.ERR      NOERR THEN ERROR PCIB.BASERR    'check for error
   CALL iotimeout(isc&, 10)             'set timeout to 10 seconds
   IF PCIB.ERR <>
     NOERR THEN ERROR PCIB.BASERR       'check for error
   CALL ioclear(isc&)                   'clear devices attached to known state
   IF PCIB.ERR <>
     NOERR THEN ERROR PCIB.BASERR       'check for error
   send ("*RST")                        'reset the analyzer
   send (":autoscale")                  'autoscale the input signal
   send (":channel1:probe x10")         'set probe factor to 10
   CLS                                  'clear PC screen
END SUB
```

```
SUB measure
   send (":measure:frequency?")      'Query for result of frequency
                                      measurement
   CALL ioenter(Anlzr&, freq!)       'Get the number from the analyzer
   IF PCIB.ERR <>
     NOERR THEN ERROR PCIB.BASERR    'Check if any error occur
   PRINT "Frequency = "; freq!; "Hz" 'Print out the result
END SUB
SUB send (cmd$)
   CALL iooutputs(Anlzr&, cmd$,
      LEN(cmd$))                     'send command to analyzer

   IF PCIB.ERR <>
     NOERR THEN ERROR PCIB.BASERR    'check for any error
END SUB
```

This program digitizes and transfers waveform data to a PC, saves the waveform data to a file, retrieves it from a file, and displays it on the PC's screen. The integrate waveform routine will perform an integration math function on the waveform data and display the results on the PC's screen. The graphics routines in this program are written for EGA/VGA graphic adaptors. There is an example written for CGA mode on the example disk. Refer to the README file for more detail.

```
DECLARE SUB initialize ()
DECLARE SUB get.waveform ()
DECLARE SUB save.waveform ()
DECLARE SUB retrieve.waveform ()
DECLARE SUB graph.waveform ()
DECLARE SUB integrate.waveform ()
DECLARE SUB send (cmd$)
'
'Filename : \QB-HPIB\DIGI_EGA.BAS
'
REM $INCLUDE: 'c:\hpib\qbsetup'

DIM SHARED preamble!(10)            'variable holding the preamble data
DIM SHARED waveform%(4000)          'variable holding the waveform data
DIM SHARED Anlzr&, isc&             'variable for analyzer address

CALL initialize                     'initialization of interface and analyzer
CALL get.waveform                   'get the waveform from analyzer
CALL save.waveform                  'save the waveform into a file
CALL retrieve.waveform              'retrieve the waveform from the file
CALL graph.waveform                 'display the waveform on the screen
CALL integrate.waveform             'integration of the waveform

END
```

```
SUB get.waveform
SHARED pcib.err, pcib.baserr, noerr
   '
   'The following four commands is added for clarity.  They are the
   'default values after the *RST command.
   '
   send (":waveform:points 4000")     'get 4000 points from analyzer
   send (":acquire:type normal")      'use normal acquisition
   send (":acquire:complete 100")     'get the data until 100% completion
   send (":waveform:source channel1") 'get the data from channel 1

   send (":waveform:format word")     'get the data in 16-bit word format
   send (":digitize channel1")        'tell the analyzer to gather data
   send (":waveform:preamble?")       'get the informations about the data
   max.len% = 10
   actual.len% = 0
   CALL ioentera(Anlzr&, SEG preamble!(1), max.len%, actual.len%)
   IF pcib.err <> noerr THEN ERROR pcib.baserr

   send (":waveform:data?")           'ask for the waveform data
   CALL iomatch(isc&, CHR$(10), 0)    'disable matching for transfer of
                                         binary data
   max.len% = 8000                    'max. is 8000 bytes
   CALL ioenterab(Anlzr&, SEG waveform%(1), max.len%, actual.len%, 2)
   IF pcib.err <> noerr THEN ERROR pcib.baserr

   max.len% = 1                       'get the newline char
   actual.len% = 0
   msg$ = SPACE$(max.len%)            'allocate the space
   CALL ioenters(Anlzr&, msg$, max.len%, actual.len%)
   IF pcib.err <> noerr THEN ERROR pcib.baserr
END SUB
```

```
SUB graph.waveform
   SCREEN 9                              'set screen mode to EGA 64k
                                         '640 x 350 x 16
   VIEW (1, 1)-(638, 256), , 15          'set viewport and draw the border
   WINDOW (0, 0)-(10, 8)                 'prepare to draw the grid
   COLOR 8                               'color dark gray
   FOR i% = 1 TO 9                       'draw the grid
     LINE (i%, 0)-(i%, 8)
   NEXT i%
   FOR i% = 1 TO 7
     LINE (0, i%)-(10, i%)
   NEXT i%

   WINDOW (1, 0)-(preamble!(3), 255)  'use window to make the co-ordinates
                                       device independent
   COLOR 10                              'bright green color
   FOR i% = 1 TO preamble!(3)
     PSET (i%, waveform%(i%))            'draw the pixel
   NEXT i%
   '
   ' Use the preamble data to calculate the timebase and vertical settings
   '
   WIDTH 80, 25
   VIEW PRINT 20 TO 24
   LOCATE 20, 1
   PRINT "V/Div  = "; 32 * preamble!(8); "V";
   PRINT "Offset = "; (128 - preamble!(10)) * preamble!(8) + preamble!(9); "V";

   LOCATE 20, 41
   PRINT "S/Div  = "; preamble!(3) * preamble!(5) / 10; "S"
   LOCATE 21, 41
   PRINT "Delay  = "; (preamble!(3) / 2 - preamble!(7)) *
     preamble!(5) + preamble!(6); "S";
END SUB
```

```
SUB initialize
SHARED pcib.err, pcib.baserr, noerr
  isc& = 7                              'assume HP-IB select code
  Anlzr& = isc& * 100 + 7               'address of analyzer
  CALL ioreset(isc&)                    'reset the HP-IB interface
  IF pcib.err <> noerr THEN ERROR pcib.baserr

  CALL iotimeout(isc&, 10)              'set timeout to 10 seconds
  IF pcib.err <> noerr THEN ERROR pcib.baserr

  CALL ioclear(isc&)                    'clear devices attached to interface
                                          to known state

  IF pcib.err <> noerr THEN ERROR pcib.baserr

  send ("*RST")                          'reset analyzer
  send (":autoscale")
  send (":channel1:probe x10")          'change probe factor to 10
  CLS                                   'clear screen of PC
END SUB

SUB integrate.waveform
  DIM math!(preamble!(3))               'define the array to hold
the math data
  math!(0) = 0                          'starting from zero
  FOR i% = 1 TO preamble!(3)            'each elt. add the current elt. to
                                         the last total
    math!(i%) = math!(i%-1) + (waveform%(i%) - preamble!(10)) * preamble!(8)
    + preamble!(9)
  NEXT i%
  max! = math!(1)                       'find out the min & max of integrate
  min! = math!(1)                       'in order to have the proper scale
  FOR i% = 1 TO preamble!(3)
    IF math!(i%) > max! THEN max! = math!(i%)
    IF math!(i%) < min! THEN min! = math!(i%)
  NEXT i%
  WINDOW (1,min!)-(preamble!(3),max!)  'set up proper scale
  COLOR 12                              'use red color
  FOR i% = 1 TO preamble!(3)            'draw the integrate
    PSET (i%, math!(i%))
  NEXT i%
  LOCATE 22, 1                          'print out information about integrate
  PRINT "Integration   V*S/Div = "; (max! - min!) / 8 * preamble!(5)
  PRINT "              Offset  = "; (max! + min!) / 2 * preamble!(5)
END SUB
```

```
SUB retrieve.waveform
  PRINT "Waveform data has been saved to a file.  The program will retrieve"
  PRINT "the data, graph it, and perform integration.  All susequent
    operations"
  PRINT "are based on the saved information.  Press <enter> to
    continue."
  OPEN "wave.dat" FOR BINARY AS #2
  FOR count% = 1 TO 10
    GET #2, , preamble!(count%)
  NEXT count%
  FOR count% = 1 TO preamble!(3)
    GET #2, , waveform%(count%)
  NEXT count%
  CLOSE #2
END SUB

SUB save.waveform
  OPEN "wave.dat" FOR BINARY AS #2
  FOR count% = 1 TO 10
    PUT #2, , preamble!(count%)
  NEXT count%
  FOR count% = 1 TO preamble!(3)
    PUT #2, , waveform%(count%)
  NEXT count%
  CLOSE #2
END SUB

SUB send (cmd$)
SHARED pcib.err, pcib.baserr, noerr
  CALL iooutputs(Anlzr&, cmd$, LEN(cmd$))
  IF pcib.err <> noerr THEN ERROR pcib.baserr
END SUB
```

This sample program sets up the RS-232 port and the analyzer, then makes a simple measurement.

```
DECLARE SUB get.waveform ()
DECLARE SUB measure ()
DECLARE SUB enter (number!)
DECLARE SUB initialize ()
'
' Filename : \QB-RS232\INIT.BAS
'
DIM SHARED lf$

CALL initialize                         'initilaize interface and analyzer
CALL get.waveform                       'tell analyzer to acquire data
CALL measure                            'measure the freq. of input signal
END

SUB enter (number)
'*************************************************************************'
'This routine will read a real number from the RS-232 port defined as file
'#1. Either comma or LF will be treated as delimiter
'*************************************************************************'
  quit = 0
  msg$ = ""
  DO                                    'skip the leading LF and comma
   DO
     ch$ = INPUT$(1, #1)                'get a byte from RS-232 port
     IF ch$ <> lf$ AND
        ch$ <> "," THEN                 'LF and comma are seperators
        msg$ = msg$ + ch$               'append byte to string
     ELSE

        quit = 1
     END IF
   LOOP UNTIL quit = 1                  'until comma or LF is found
  LOOP UNTIL LEN(msg$) <> 0             'until valid content is found
  number = VAL(msg$)                    'convert the string into number
END SUB

SUB get.waveform
   PRINT #1, ":digitize channel1"
END SUB
```

```
SUB initialize
   lf$ = CHR$(10)                    'define LF char
   ' analyzer is connected to COM1 at 9600 baud
   OPEN "COM1:9600,n,8,1,ASC,CD1000,CS1000,DS1000" FOR RANDOM AS #1 LEN = 2
   PRINT #1, "*RST";lf$;
   PRINT #1, ":autoscale"; lf$;      'autoscale the input signal
   PRINT #1, ":channel1:probe x10";
     lf$;                            'set probe factor to 10
   CLS                               'clear screen
END SUB

SUB measure
   PRINT #1, ":measure:frequency?"; lf$;
   CALL enter(freq!)
   PRINT "Frequency = "; freq!; " Hz";
END SUB
```

This program digitizes and transfers waveform data to a PC, saves the waveform data to a file, retrieves it from a file, and displays it on the PC's screen. The integrate waveform routine will perform an integration math function on the waveform data and display the results on the PC's screen. The graphics routines in this program are written for EGA/VGA graphic adaptors. There is an example written for CGA mode on the example disk. Refer to the README file for more detail.

```
DECLARE SUB initialize ()
DECLARE SUB get.waveform ()
DECLARE SUB save.waveform ()
DECLARE SUB retrieve.waveform ()
DECLARE SUB graph.waveform ()
DECLARE SUB integrate.waveform ()
DECLARE SUB enter (number!)
'
' Filename : \QB-RS232\DIGI_EGA.BAS
'
DIM SHARED preamble!(10)              'variable holding the preamble data
DIM SHARED waveform%(4000)            'variable holding the waveform data
DIM SHARED lf$                        'will be defined as LF char

CALL initialize                       'initialize interface and analyzer
CALL get.waveform                     'get waveform from analyzer
CALL save.waveform                    'save waveform to a file
CALL retrieve.waveform                'retrieve waveform from file
CALL graph.waveform                   'display the waveform data
CALL integrate.waveform               'integrate the waveform data
END

SUB enter (number!)
'*************************************************************************
'This routine will read a real number from the RS-232 port defined as file
' #1.  Either comma or LF will be treated as delimiter
'*************************************************************************
   quit = 0
   msg$ = ""
   DO                                 'skip the leading LF and comma
         DO
              ch$ = INPUT$(1, #1)     'get a byte from RS-232 port
              IF ch$ <> lf$ AND
                ch$ <> "," THEN 'LF and comma are separators
                msg$ = msg$ + ch$     'append byte to string
              ELSE
```

```
            quit = 1
          END IF
        LOOP UNTIL quit = 1          'until comma or LF is found
    LOOP UNTIL LEN(msg$) <> 0        'until valid content is found
    number = VAL(msg$)               'convert the string into number
END SUB

SUB get.waveform
   '
   'The following four commands are added for clarity only. They are
   'the default values after the *RST command
   '
   PRINT #1, ":waveform:points 4000"; lf$;  'get 4000 points from analyzer
   PRINT #1, ":acquire:type normal"; lf$;   'set acquire to be normal
   PRINT #1, ":acquire:complete 100"; lf$;  'get data until 100% completion
   PRINT #1, ":waveform:source chan1"; lf$; 'get data from channel 1

   PRINT #1, ":waveform:format word"; lf$;  'get data in 16-bit word format
   PRINT #1, ":digitize channel1"; lf$;     'tell analyzer to gather data
   '
   ' Get the preamble data for the waveform
   '
   PRINT #1, ":waveform:preamble?"; lf$;    'ask for preamble informations
   FOR count% = 1 TO 10                     'get the preambles
      CALL enter(preamble!(count%))
   NEXT count%
   '
   ' Get the waveform from the analyzer in word format
   '
   PRINT #1, ":waveform:data?"; lf$;        'ask for the waveform data
   msg$ = INPUT$(10, #1)                    'the arbitary block header
   length% = VAL(RIGHT$(msg$, 8)) / 2       'length is specified in last 8
                                             bytes of header
                                            'length is measured in bytes
   FOR count% = 1 TO length%                'get the waveform
      GET #1, , waveform%(count%)
   NEXT count%
   msg$ = INPUT$(1, #1)                     'get the last NL char.
END SUB

SUB graph.waveform
   SCREEN 9                                 ' set screen mode to EGA  64k
                                            ' 640 x 350 x 16
   VIEW (1, 1)-(638, 256), , 15             ' set viewport and draw the
                                              border
   WINDOW (0, 0)-(10, 8)                    ' prepare to draw the grid
   COLOR 8                                  ' color dark gary
```

```
FOR i% = 1 TO 9                          ' draw the grid
      LINE (i%, 0)-(i%, 8)
      NEXT i%
      FOR i% = 1 TO 7
      LINE (0, i%)-(10, i%)
   NEXT i%

   WINDOW (1, 0)-(preamble!(3), 255)       ' use window to make the
                                             co-ordinates to be
                                           ' device independent
   COLOR 10                                ' bright green color
   FOR count% = 1 TO preamble!(3)
         PSET (count%, waveform%(count%))  ' draw the pixel
   NEXT count%
   '
   ' Use the preamble data to calculate the timebase and vertical settings
   '
   COLOR 15
   WIDTH 80, 25
   VIEW PRINT 20 TO 24
   LOCATE 20, 1
   PRINT "V/Div  = "; 32 * preamble!(8); "V";

   PRINT "Offset = "; (128 - preamble!(10)) * preamble!(8) + preamble!(9);
         V";

   LOCATE 20, 41
   PRINT "S/Div  = "; preamble!(3) * preamble!(5) / 10; "S";
   LOCATE 21, 41
   PRINT "Delay  = "; (preamble!(3) / 2 - preamble!(7)) * preamble!(5)
         + preamble!(6); "S";
END SUB

SUB initialize
   lf$ = CHR$(10)                          ' define the LF char
                        ' analyzer is connected to COM1 at 9600 baud

   OPEN "COM1:9600,n,8,1,ASC,CD1000,CS1000,DS1000" FOR RANDOM AS #1 LEN = 2
   PRINT #1, "*RST"; lf$;                  ' reset the analyzer
   PRINT #1, ":autoscale"; lf$;            ' autoscale the input signal
   PRINT #1, ":channel1:probe x10"; lf$;   ' set probe factor to 10
   CLS
END SUB

SUB integrate.waveform
   DIM math!(preamble!(3))                 'define the array to hold the
                                            math data
```

```
   math!(0) = 0                                'starting from zero
   FOR i% = 1 TO preamble!(3)                  'each elt. add the current elt.
                                                to the last total
      math!(i%) = math!(i%-1) + (waveform%(i%) - preamble!(10)) *
          preamble!(8) + preamble!(9)
   NEXT i%
   max! = math!(1)                     'find out the min & max of integrate
   min! = math!(1)                     'in order to have the proper scale
   FOR i% = 1 TO preamble!(3)
      IF math!(i%) > max! THEN max! = math!(i%)
      IF math!(i%) < min! THEN min! = math!(i%)
   NEXT i%
   WINDOW (1, min!)-(preamble!(3), max!)    'set up proper scale
   COLOR 12                                 'use red color
   FOR i% = 1 TO preamble!(3)               'draw the integrate
      PSET (i%, math!(i%))
   NEXT i%
   LOCATE 22, 1                             'print out information about
                                             integrate
   PRINT "Integration   V*S/Div = "; (max! - min!) / 8 *
preamble!(5)
   PRINT "              Offset  = "; (max! + min!) / 2 *
preamble!(5)
END SUB
SUB retrieve.waveform
   PRINT "Waveform data has been saved to a file.  The program will retrieve"
   PRINT "the data, graph it, and perform integration.  All subsequent operations"
   PRINT "are based on the saved information.  Press <enter> to continue."
   LINE INPUT dummy$
   CLS
   OPEN "wave.dat" FOR BINARY AS #2
   FOR count% = 1 TO 10
       GET #2, , preamble!(count%)
   NEXT count%
   FOR count% = 1 TO preamble!(3)
       GET #2, , waveform%(count%)
   NEXT count%
   CLOSE #2
END SUB
```

```
SUB save.waveform
   OPEN "wave.dat" FOR BINARY AS #2
   FOR count% = 1 TO 10
         PUT #2, , preamble!(count%)
   NEXT count%
   FOR count% = 1 TO preamble!(3)
         PUT #2, , waveform%(count%)
   NEXT count%
   CLOSE #2
END SUB
```

This sample program sets up the HP-IB interface and the analyzer, then makes a simple measurement.

```
#include "c:\qc25\chpib.h"
#include "c:\qc25\cfunc.h"
#include <graph>
#include <stdio.h>
send (char *cmd);
initialize();
measure();
/*
      Filename : \QC-HPIB\INIT.C
*/
#define ISC     7L                      /* select code of HP-IB interface */
#define ANLZR   707L                    /* address of analyzer */
#define TRUE    1
main()
{
   initialize();                        /* initialization of interface &
                                           analyzer */
   get_waveform();
   measure();                           /* make measurement and print result */
}
error_handler(int error, char *routine)
char    ch;
if (error != NOERR)
   {
         printf("Error in call to %s \n", routine);
         printf("%d %s \n", error, errstr(error));
         printf("Press Enter to continue: ");
         scanf ("%c", &ch);
   }
}
initialize()
{
   error_handler(IORESET(ISC), "ioreset");          /* reset interface */
   error_handler(IOTIMEOUT(ISC, 10), "iotimeout"); /* set timeout to 10 sec */
   error_handler(IOCLEAR(ISC), "ioclear");          /* clear analyzer to known
                                                       state */
```

```
   send ("*RST");                                    /* reset analyzer to
                                                        default */
   send (":autoscale");                              /* autoscale input
                                                        signal */
   send (":channel1:probe x10");                     /* set probe factor to
                                                        x10 */
   _clearscreen(_GCLEARSCREEN);                      /* clear screen of PC */
}

measure()
{
   float        freq;

   send (":measure:frequency?");                     /* query frequency of
                                                        input signal */
   error_handler(IOENTER(Anlzr, &freq), "ioenter"); /* get the freq. from
                                                        analyzer */
                                                     /* noted that freq must
                                                        be passed by ref */
   printf ("Frequency = %f Hz \n", freq);            /* print it out */
}

send(char *cmd)
{  /* this subroutine send the char string pointed by cmd to analyzer */
   error_handler(IOOUTPUTS(Anlzr, cmd, strlen(cmd)), cmd);
}

get_waveform()
{
   send (":digitize channel1");
}
```

This program digitizes and transfers waveform data to a PC, saves the waveform data to a file, retrieves it from a file, and displays it on the PC's screen. The integrate waveform routine will perform an integration math function on the waveform data and display the results on the PC's screen. The graphics routines in this program are written for EGA/VGA graphic adaptors. There is an example written for CGA mode on the example disk. Refer to the README file for more detail.

```
#include "c:\qc25\chpib.h"
#include "c:\qc25\cfunc.h"
#include <stdio.h>
#include <graph.h>
initialize();
get_waveform();
save_waveform();
retrieve_waveform();
graph_waveform();
integrate_waveform();
send (char *cmd);
/*
   Filename : \QC-HPIB\DIGI_EGA.C
*/
#define ISC     7L                      /* select code of HP-IB interface */
#define Anlzr   707L                    /* address of analyzer */
#define TRUE    1
float           preamble[10];           /* array to hold the preamble data */
unsigned char   waveform[4000];         /* array to hold the waveform data */
void main()
{
   initialize();                        /* initialize analyzer and interface */
   get_waveform();                      /* get the waveform from analyzer */
   save_waveform();                     /* save the waveform to a file */
   retrieve_waveform();                 /* retrieve the waveform from file */
   graph_waveform();                    /* display the data */
   integrate_waveform();                /* integrate the waveform and
                                           display it */
}
error_handler(int error, char *routine)
{
   char    ch;
   if (error != NOERR) {                            /* check for errors */
       printf("Error in call to %s \n", routine);   /* print error info*/
       printf("%d %s \n", error, errstr(error));
       printf("Press Enter to continue: ");
```

```
        printf ("%c", ch = getchar());              /* wait for enter key */
    }
}
initialize()
{
    error_handler(IORESET(ISC),"ioreset");         /* reset HP-IB interface */
    error_handler(IOTIMEOUT(ISC,10),"iotimeout"); /* set timeout to 10 sec */
    error_handler(IOCLEAR(ISC),"ioclear");         /* clear analyzer to known
                                                      state */

    send ("*RST");                                 /* reset analyzer */
    send (":autoscale");                           /* autoscale input signal */
    send (":channel1:probe x10");                  /* set probe factor to 10 */
    _clearscreen (_GCLEARSCREEN);                  /* clear PC's screen */
}
send(char *cmd

{  /* this routine is added to make sending the command to analyzer easier to
      read */
    error_handler(IOOUTPUTS(Anlzr, cmd, strlen(cmd)), cmd);
}
get_waveform()
{
    int  element, length, i;
    char endline;
    char msg[20];

    /* The following four commands are added for clarity.  They are the
       default values after the *RST command
    */
    send (":waveform:points 4000");       /* get 4000 points from analyzer */
    send (":acquire:type normal");        /* set acquisition type to normal */
    send (":acquire:complete 100");       /* get data until 100% completion */
    send (":waveform:source channel1");   /* get data from channel 1 */

    send (":waveform:format byte");       /* use the 8-bit byte format */
    send (":digitize channel1");          /* tell analyzer to gather data */
    send (":waveform:preamble?");         /* ask for info about data */
    length = 10;                          /* totally 10 parameters */
    error_handler(IOENTERA(Anlzr,
       preamble,&length), "ioentera");
    send (":waveform:data?");             /* ask for the waveform data */
    error_handler(IOMATCH(ISC
        ,endline,0), "iomatch");          /* disable matching for transfer
                                             of binary data */
```

```
   length = 4000;                          /* max. is 4000 bytes */
   error_handler(IOENTERAB(Anlzr,waveform,&length,1),
"ioenterab");
   length = 1;                             /* get the last newline char */
   error_handler(IOENTERS(Anlzr,msg,&length), "ioenters");
}
save_waveform()
{
   FILE *fp;
   fp = fopen("wave.dat", "wb");           /* open a file for storage */
   fwrite(preamble,sizeof
      (preamble[0]),10,fp);                /* write the preambles to file */
   /* write the data, preamble[2] contains the number of data */
   fwrite (waveform, sizeof(waveform[0]), (unsigned)preamble[2], fp);
   fclose (fp);                            /* close the file */
}
retrieve_waveform()
{
   FILE *fp;
       printf ("Waveform data has been saved to a file.  The program
               will retrieve\n");
       printf ("the data, graph it, and perform integration.  All
               subsequent operations\n");
       printf ("are based on the svaed information.  Press <enter>
               to continue.\n");
       printf ("%c", ch = getchar());
   fp = fopen("wave.dat", "rb");                   /* open the file */
   fread(preamble,sizeof(preamble[0]),10,fp);      /* get the preambles */
   /* retrieve the waveform data, preamble[2] contains the number of points */
   fread (waveform, sizeof(waveform[0]), (unsigned)preamble[2], fp);
   fclose(fp);                                     /* close the file */
}
graph_waveform()
{
   int          i;
   _setvideomode(_ERESCOLOR);                /* set screen mode to EGA */
   _rectangle (_GBORDER, 0,0, 639,257);      /* draw border */
   _setviewport(1,1,638,256);                /* set viewport */
   _setwindow (TRUE, 0,0,10,8);              /* set mapping co-ordinates */
   _setcolor (8);                            /* set color to dark gray */
   for (i=1;i<11;i++) {                      /* draw the grid */
         _moveto_w(i,0);
         _lineto_w(i,8);
   }
```

```
for (i=1;i<9;i++) {
          _moveto_w(0,i);
          _lineto_w(10,i);
   }
   _setwindow(TRUE, 1,0,preamble[2],255);   /* set mapping co-ordinates */
   _setcolor(10);                           /* set color to bright green */
   for (i=0;i<preamble[2];i++) {            /* draw the pixels */
         _setpixel_w (i,waveform[i]);
   }
   _settextwindow(20,1,24,80);              /* set text window */
   _settextcolor (15);                      /* set color to bright white */
   _settextposition(1,1);                   /* position cursor */
   printf ("V/Div = %f V \n", 32*preamble[7]);
   printf ("Offset= %f V \n", (128-preamble[9])*preamble[7]+preamble[8]);
   _settextposition(1,41);
   printf ("S/Div = %f S \n", preamble[2]*preamble[4]/10);
   _settextposition(2,41);
   printf ("Delay = %f S \n", (preamble[2]/2 -preamble[6])*preamble[4]+
        preamble[5]);
}
integrate_waveform ()
{
   float        math[4001], min, max;     /* define array to hold math data */
   int          i;
   math[0] = 0;                           /* starting from zero */
   for (i=0;i<preamble[2];i++) {          /* each elt. = last elt + curr
                                             wf data */
       math[i+1] = math[i] + (waveform[i]-preamble[9])*preamble[7] +
                 preamble[8];
   }
   max = math[1];                    /* find out the max & min of integrate */
   min = math[1];                    /* in order to have the proper scale */
   for (i=1; i<=preamble[2]; i++) {
         if (math[i] > max) {max = math[i];}
         if (math[i] < min) {min = math[i];}
   }
   _setwindow(TRUE, 1, min,
           preamble[2], max);             /* set up proper mapping */
   _setcolor(12);                         /* use red color */
   for (i=1; i<=preamble[2]; i++)         /* draw the integrate */
         _setpixel_w(i, math[i]);
   _settextposition (3,1);                /* print out information about
                                             integrate */
   _settextcolor (12);
   printf ("Integration  V*S/Div = %f\n", ((max-min)/8*preamble[4]));
   printf ("             Offset  = %f", ((max+min)/2*preamble[4]));
}
```

Custom Mask Example Programs

Mask downloading programming example

The following BASIC program downloads a user-defined mask test file named "FDDI3.IN" into the non-volatile memory of the HP 83475B.

```
10    ! SEND MASK DATA TO THE ANALYZER
20    !
30    ! CONFIGURATION SECTION
40    !
50    DIM S1$[1],Out$[25],Cmd$[35],Cmd2$[35]
60    DIM T$[100]
70    In$="FDDI3.IN"                        ! File path definition
80    !
90    Cmd$="MEAS:MASK:DEF:DATA UDM2,"       ! Define command for mask downloading
100   Cmd2$="MEAS:MASK:IDEN? UDM2"          ! Define command for reading mask
105                                         ! identification
110   !
120   ASSIGN @I TO In$;FORMAT OFF           ! Open file
130   ASSIGN @S TO 707                      ! Set analyzer address
140   ASSIGN @S1 TO 707;FORMAT OFF
150   ON TIMEOUT 7,10 GOTO Timeout_error    ! Set time-out condition
160   !
170                                         ! Get the file size in bytes
180   STATUS @I,3;N
190   STATUS @I,4;R
200   L=N*R-1
210   L$=VAL$(L)
220   Ll=LEN(L$)
230   Ll$=VAL$(Ll)
240   Beg$="#"&Ll$&L$                       ! Define a string with the file size
250   !
260   !
270   !
280   OUTPUT @S;Cmd$&Beg$;                  ! Send the data input command
290                                         ! with the file size
300   LOOP                                  ! Begin to read the mask definition
310   ON ERROR GOTO End_data                ! file into a string and then send the
```

```
320   ENTER @I USING "#,K";T$              ! string to the analyzer.
330   OUTPUT @S1 USING "#,K";T$            ! When finished, go to End_data.
340   PRINT USING "#,K";T$
350   END LOOP
360 End_data: !
370   OUTPUT @S;"",END                     ! Send terminator to analyzer
380   OUTPUT @S;Cmd2$                      ! Query for user-defined mask label
390   ENTER @S;Resp$
400   PRINT Resp$                          ! Print the user-defined mask label
410   PRINT "ALL DONE!!"
420   STOP
430 Timeout_error:  PRINT "TIME OUT!! EXITING..."
440  END
```

Mask reading programming example

The following BASIC program reads the E4 one pulse mask definition from the HP 83475B and places it in an ASCII file.

```
10    ! USE MEAS:MASK:DEF:DATA COMMAND TO GET MASK DATA FROM ANALYZER
20    !
30    !    THIS VERSION GETS ONE CHARACTER AT A TIME AND PUTS IT INTO
40    !    THE OUTPUT FILE.
50    !
60    ! CONFIGURATION SECTION
70    !
80    Bl=80
90    DIM S1$[1],Out$[25],Cmd$[35],Cmd2$[35],Rst$[35],Buf$[80]
110   Out$="MMDD.OUT"
120   !
130   Rst$="*RST;AUTOSCALE;ERASE"          ! Define program commands
140   Cmd$="MEAS:MASK:DEF:DATA? E4ONE"
150   Cmd2$="MEAS:MASK:IDEN? E4ONE"
160   !
170   ASSIGN @S TO 707                     ! Assign analyzer I/O path
180   ASSIGN @T TO 707;FORMAT OFF          ! Assign another path without format
190   !
200   ON TIMEOUT 7,10 GOTO Timeout_error   ! Set time-out condition
210   !
220   !
230   !
240   ON ERROR GOTO Off_error              ! Get output file ready to go
250   PURGE Out$
260 Off_error: OFF ERROR
270   CREATE Out$,1                        ! Open the output file
280   ASSIGN @O TO Out$;FORMAT OFF         ! Assign I/O file path
290   !
300   !
310   CALL Send_out(@S,Rst$,0)             ! Re-set the analyzer
320   !
330   !
340   PRINT "SENDING >>";Cmd$;"<<"
350   OUTPUT @T;Cmd$,END                   ! Send the data generating command
360   !
370   !
390   ENTER @T USING "#,K";S1$             ! Get the response and pick-off the
400   PRINT "THIS SHOULD BE A # -> ";S1$   ! # sign
410   !
```

```
420   !
430   ENTER @T USING "#,K";S1$              ! Assesses how many numbers are in
435                                         ! the length specifier
440   PRINT "NUMBER OF CHARACTERS IN THE NUMBER = ";S1$
450   !
460   !
470   IF S1$="0" THEN                       ! Find out if it is of determinate length
480     PRINT "UNSUPPORTED - INDEFINITE LENGTH BLOCK INPUT"
490   ELSE
500     ALLOCATE Sn$[VAL(S1$)]
510     ENTER @T USING "#,K";Sn$            ! Find the number of characters to be read
520     !
530     PRINT "NUMBER OF CHARACTERS TO BE READ IN = ";Sn$
540     L=VAL(Sn$)
550     PRINT "GETTING DATA";
560     N=INT(L/B1)
570     FOR I=1 TO N STEP 1                 ! Begin to read the data from the analyzer
580       ENTER @T USING "#,K";Buf$         ! into a string and then from the string to
590       OUTPUT @O USING "#,K";Buf$        ! the output file
600       PRINT ".";
610     NEXT I
620     FOR I=1 TO L-N*B1+1 STEP 1          ! Get the last few bytes
630       ENTER @T USING "#,K";S1$
640       OUTPUT @O USING "#,K";S1$
650       PRINT ".";
660     NEXT I
670     PRINT
690   END IF
700   PRINT "TRANSFER COMPLETE"             ! Indicate transfer is done
730   ASSIGN @Out TO *                      ! Close the output file
740   CALL Send_out(@S,Cmd2$,1)             ! Re-set the analyzer and check for errors
750   PRINT "FINISHED!!!"
760   STOP
770 Timeout_error: PRINT "TIME OUT!! EXITING..."
780   ASSIGN @Out TO *                      ! Close the output file after time-out
790   END
800   SUB Send_out(@S,Str$,Query)           ! Sub-program that sends a command to the
810     PRINT "SENDING >>";Str$;"<<"        ! analyzer and checks for errors
820     OUTPUT @S;Str$
850     IF (Query=1) THEN
860       ON TIMEOUT 7,1 GOTO Get_errors
870       ENTER @S;Resp$
880       PRINT "RECIEVED RESPONSE >>";Resp$;"<<"
890     END IF
900 Get_errors: OFF ERROR
910     OUTPUT @S;":SYST:ERR?"
920     ENTER @S;Err
930     IF (NOT (Err=0)) THEN
940       PRINT "ERROR # ";Err;"OCCURRED DURING COMMAND"
```

```
950       REPEAT
960         OUTPUT @S;":SYST:ERR?"
970         ENTER @S;Err
980         IF (NOT (Err=0)) THEN
990           PRINT "ANOTHER ERROR # ";Err
1000        END IF
1010      UNTIL Err=0
1020    END IF
1030  SUBEND
```

A

Remote Optical Channel Calibration

Remote Optical Channel Calibration

The HP 83475B lightwave communication analyzer allows you to perform optical channel calibrations. In addition to the three factory default calibrations, three user calibrations can be performed and saved. To ensure the watts vertical scale functions properly, the user calibrations should be performed *at least ±20 nanometers* from the standard wavelengths of 780 nm, 1310 nm, and 1550 nm.

Each calibration requires making a sequence of power measurements. The first measurement is for the calculation of the receiver conversion gain. The next measurement measures the system dark noise level. High and low measurements are then performed for each of the three power meter ranges.

Required Equipment

HP 815545M	1310/1550 nm source
HP 81530A	450–1020 nm detector
HP 81532A	800–1700 nm detector
HP 8153A	lightwave multimeter
HP 8158B	Option 001 600–1200 nm attenuator (MM, FC-PC connector)
HP 8158B	Option 002 1200–1650 nm attenuator (MM, FC-PC connector)

A block diagram of a calibration setup is shown in Figure A-1.

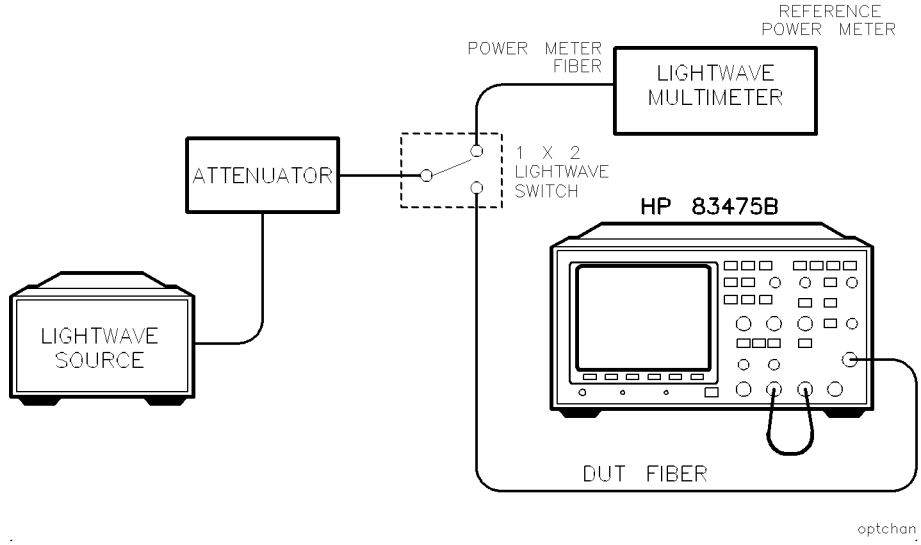


Figure A-1. Block diagram of a calibration setup.

To improve calibration accuracy, lightwave switches are recommended for signal routing. Clean connectors are necessary to assure the best accuracy. Refer to Chapter 1 of the *HP 83475B Lightwave Communications Analyzer User's Guide* for information on lightwave connector care.

To Correct for Insertion Loss

If the setup shown in Figure A-1 is used, the insertion loss of the power meter and device-under-test (DUT) signal paths should be corrected for. This provides a more accurate calibration. The following procedure can be used to correct for the insertion loss of the signal paths.

1. Adjust the attenuator to a set level (0.0 dB is good).
2. Connect the DUT fiber to the reference power meter.
3. Measure and record the power at the DUT fiber signal path output.

   _____ μW
4. Connect the DUT fiber to the optical input of the HP 83475B.
5. Connect the power meter fiber to the reference power meter.
6. Measure and record the power at the power meter fiber signal path output.

   _____ μW

WARNING

Do not disconnect the power meter fiber from the reference power meter. It is important to maintain the connection so repeatability uncertainity is minimized.

7. Calculate the insertion loss correction, in percent, using the following formula:

$$Correction\ Factor = \frac{P\ at\ DUT\ path - P\ at\ power\ meter\ path}{P\ at\ DUT\ path}$$

where:

P = power, in microwatts, at the specified points

During the calibration procedure, several power levels must be input into the HP 83475B optical channel. The instrument must be "told" what these power levels are. Use the following formula to calculate the actual power levels and enter them into the HP 83475B.

Use the correction factor, as calculated earlier, in the following equation to determine the actual power input, in microwatts, to the HP 83475B optical channel:

$$Input\ power\ to\ HP\ 83475B = \frac{Reference\ power\ meter\ reading}{1 - correction\ factor}$$

Remote Calibration Sequence

This calibration must be performed in the sequence shown below. Do not change the order of the commands and do not execute any other remote commands between the commands.

The calibration sequence commands are all queries. When the step is complete one of the following responses will be returned.

ERRor	indicates an unrecoverable error occurred during the last step of the calibration. The calibration is aborted and the analyzer is reset to the state existing before the calibration was called.
CONTinue	indicates the last step of the calibration was successful. Continue with the calibration procedure.
OFFScreen	indicates the last step of the calibration was not successful. Vary the input criteria and retry the last step until the instrument senses the power level and returns `CONT` or `COMP`.
COMPlete	indicates the calibration is complete and the calibration table has been saved.

WARNING

The instrument *will accept* other remote commands during the execution of the calibration sequence. If operation of the analyzer is suspect, it may be necessary to cycle the line power off, then on, to ensure the analyzer is in a known operational state.

The :OCAL:KILL command can be used at any point in the sequence to abort the calibration sequence and reset the instrument to the state it was in before the calibration sequence was initiated.

Refer to Chapter 7 for a complete description of the commands used in the remote optical channel calibration.

1. :OCAL:CALibrate?

   The :OCALibrate:CALibrate query begins the calibration sequence for the selected calibration. Either standard or user-defined calibrations may be used.

   Returns CONT or ERR.

2. :OCAL:ECAL:SKIP?

   The :OCALibrate:ECALibrate:SKIP query allows the controller to omit (skip) the electrical calibration steps. When selected, the next step performed is OCAL:OCAL:PDAR.

   Returns CONT or ERR. ERR indicates this query was executed in the wrong sequence. This is the only place in the remote optical calibration sequence the calibration sequence can continue after ERR is returned.

3. :OCAL:ECAL:XTRigger?

   The :OCALibrate:ECALibrate:XTRigger query performs an electrical calibration of the external trigger circuits.

WARNING Before executing this query, make sure the rear panel `DC OUTPUT` is connected to the front panel `EXTERNAL TRIGGER INPUT`.

   Returns ERR or CONT.

4. :OCAL:ECAL:CHAN1?

   The :OCALibrate:ECALibrate:CHAN1 query performs an electrical calibration of channel 1.

WARNING Before executing this query, make sure the rear panel `DC OUTPUT` is connected to the front panel `CHANNEL 1 INPUT`.

   Returns ERR or CONT.

5. :OCAL:ECAL:CHAN2?

   The :OCALibrate:ECALibrate:CHAN2 query performs an electrical calibration of channel 2.

WARNING Before executing this query, make sure the rear panel `DC OUTPUT` is connected to the front panel `CHANNEL 2 INPUT`.

   Returns ERR or CONT.

6. :OCAL:OCAL:PDARk?

   The :OCALibrate:OCALibrate:PDARk query initiates a measurement of the optical channel dark noise level.

   Returns ERR or CONT.

WARNING

Before executing this query, make sure:

- The `DEMOD OUTPUT` is connected to the `CHANNEL 2 INPUT`
- There is no signal applied to the optical input
- The source is de-activated or a dark cap is placed on the optical input connector

NOTE

Input power level can be determined from the reference power meter in Figure A-1. You may choose to apply an insertion loss correction as described in "To Correct for Insertion Loss".

7. :OCAL:OCAL:OCGain <power> [W, mW, uW]?

   The :OCALibrate:OCALibrate:OCGain query performs the optical channel gain calibration. The <power> string should indicate the laser output power.

WARNING

Before executing this query:

- Connect the laser source to the `OPTICAL INPUT`.
- For 1310 nm or 1550 nm calibrations, set the laser power as close to 160 μW as possible. For best calibration results, do not exceed 160 μW (-7.9 dBm).
- For 780 nm calibrations, set the laser power as close to 1000 μW as possible. For best calibration results, do not exceed 1000 μW (0 dBm).
- The optical input damage level is 2.5 mW (=4 dBm).

Returns ERR or CONT.

8. :OCAL:OCAL:PMET:LLRange <power> [W, mW, uW]?

   The :OCALibrate:PMET:LLRange query makes and saves the optical power low value for the low range calibration.

   Returns OFFS, ERR or CONT.

9. :OCAL:OCAL:PMET:HLRange <power> [W, mW, uW]?

   The :OCALibrate:PMET:HLRange query makes and saves the optical power high value for the low range calibration.

   Returns OFFS, ERR or CONT.

10. :OCAL:OCAL:PMET:LMRange <power> [W, mW, uW]?

    The :OCALibrate:PMET:LMRange query makes and saves the optical power low value for the midrange calibration.

    Returns OFFS, ERR or CONT.

11. :OCAL:OCAL:PMET:HMRange <power> [W, mW, uW]?

    The :OCALibrate:PMET:HMRange query makes and saves the optical power high value for the midrange calibration.

    Returns OFFS, ERR or CONT.

12. :OCAL:OCAL:PMET:LHRange <power> [W, mW, uW]?

    The :OCALibrate:PMET:LHRange query makes and saves the optical power low value for the high range calibration.

    Returns OFFS, ERR or CONT.

13. :OCAL:OCAL:PMET:HHRange <power> [W, mW, uW]?

    The :OCALibrate:PMET:HHRange query makes and saves the optical power high value for the high range calibration.

    Returns OFFS, ERR or CONT.

14. :OCAL:SAVE?

    The :OCALibrate:SAVE query saves the calibration data and generates a calibration table. This table can be used for both instantaneous and average power measurements.

    Returns COMP or ERR. If the calibration sequence has not been performed correctly, ERR will be returned.

Index

Index

A
:ACQuire
COMPlete, 7-13
COUNt, 7-14
POINts, 7-15
SETup, 7-15
TYPE, 7-16
Acquire
command types, 6-7–8
parameters, 4-17
aliasing, 2-14
alphabetic command cross-reference, 5-14
:ASTore, 7-16
:AUToscale, 7-17
Autoscale, 4-15
averaged mode, 4-17

B
basic command structure, 4-16
:BLANk, 7-17
block response data, 4-21

C
cable interface
RS-232, 1-5–8
calibration
remote optical channel, A-2
setup, A-2
center reference polygon, 2-30
:CHANnel
BWLimit, 7-18
COUPling, 7-19
INPut, 7-20
INVert, 7-21
IOPTical, 7-21
OFFSet, 7-22
PROBe, 7-23
PROTect, 7-24
RANGe, 7-24
SETup, 7-26
SKEW, 7-27
VERNier, 7-27
XEATtenuation, 7-28
XEAUnits, 7-29

XOATtn, 7-29
Channel command types, 6-9
character program data, 4-11
*CLS (Clear Status) command, 7-3
codes, user-defined test, 2-33
combining commands, 4-8
command
alphabetic cross-reference, 5-14
basic structure, 4-16
combining, 4-8
common, 5-5
header, 5-7
instructions, 4-6
lockout, 4-33
overlapped, 5-11
program examples, 5-13
query, 4-8
root level, 5-5
sequential, 5-11
set organization, 5-14
subsystem, 5-5, 5-7
types, 5-4
command header
common, 4-7
compound, 4-7
simple, 4-7
command tree, 5-6
description, 5-4
common command, 5-4
header, 4-7
types, 6-3–4
Common command
*CLS (Clear Status), 7-3
*ESE (Event Status Enable), 7-4
*ESR (Event Status Register), 7-5
*IDN (Identification Number), 7-6
*LRN (Learn), 7-6
*OPC (Operation Complete), 7-7
*RCL (Recall), 7-7
*RST (Reset), 7-8
*SAV (Save), 7-9
*SRE (Service Request Enable), 7-9
*STB (Status Byte), 7-11
*TRG (Trigger), 7-12
*TST (Test), 7-12
*WAI (Wait), 7-12
common commands
definition, 5-5
compound

command header, 4-7
header, 5-7
custom mask
test, 2-28–44
test file, 2-42

D
data bits, 4-32
DC value, 2-13
definition
language, 2-32
polygon, 2-39
user-defined mask, 2-34
definitions
terms, 2-34
delay
reference, 4-15
time, 4-15
device address, 4-5, 4-18
:DIGitize, 7-30
Digitize command, 4-17
:DISPlay
COLumn, 7-31
GRID, 7-31
INVerse, 7-32
LINE, 7-32
PIXel, 7-33
ROW, 7-34
SETup, 7-35
TEXT, 7-35
Display command types, 6-10
:DITHer, 7-36
duplicate mnemonics, 4-8

E
embedded strings, 4-11
:ERASe, 7-36
error
messages, 7-129–131
queue, 4-19
*ESE (Event Status Enable) command, 7-4
*ESR (Event Status Register) command, 7-5
examples, programming, 8-2
:EXTernal
COUPling, 7-37
INPut, 7-38
OFFSet, 7-39
PROBe, 7-40
PROTect, 7-41
SETup, 7-41

SKEW, 7-42
External (trigger) command types, 6-11
Eye:NRZ
power/voltage measurement, 6-19
time measurement, 6-18

F Factors softkey, 1-10
Fast Fourier Transforms (FFT), 3-3–7
FDDI pulse mask example, 2-42
FFT measurement
aliasing, 2-14
center frequency control, 3-4
DC value, 2-13
display, 3-4–5
frequency cursor resolution, 3-3
frequency range, 3-4
frequency span control, 3-4
input, 3-3
magnification, 3-4
menu, 2-13
operating system, 2-13
operation, 2-15
peak find, 3-3
points, 3-3
programming, 3-5
requirements, 2-13
sensitivity, 3-3
spectral leakage, 2-15
test region, 3-3
time record length, 3-3
vertical offset, 3-3
window, 3-4
fixed amplitude pulse mask example, 2-44
front-panel menu
RS-232-C interface, 4-31
:FUNCtion
CENTer, 7-43
MOVE, 7-44
OFFSet, 7-45
OPERation, 7-45
PEAKs, 7-46
RANGe, 7-47
REFerence, 7-48
SOURce, 7-49
SPAN, 7-50
VIEW, 7-50
WINDow, 7-51
Function command types, 6-12

G General Waveform
power/voltage measurement, 6-19
time measurement, 6-18

H header types, 4-7
horizontal range, 4-15
HP BASIC with HP-IB, 8-4
HP BASIC with RS-232, 8-8
HP-IB
address, 4-24
bus command, 4-26
command mode, 4-23
configuration, 1-9
data mode, 4-23
device clear command, 4-26
instrument address, 4-25
interface capabilitities, 4-23
interface clear command, 4-26
interface select code, 4-25
lockout mode, 4-26
menu softkey, 1-9
remote program interface, 1-4, 4-3
HP IBASIC for Windows with HP-IB, 8-12
HP IBASIC for Windows with RS-232, 8-17

I *IDN (Identification Number) command, 7-6
infinity representation, 5-11
initialization statement, 4-15
insertion loss correction, A-4
instrument status, 4-22
interface configuration, 1-9

L language definition, 2-32
left-bottom reference polygon, 2-29
lockout command, 4-33
*LRN (Learn) command, 7-6

M mask
downloading program, 8-42
polygons, 2-28
reading program, 8-44
regions, 2-31
user-defined test, 2-32
user-defined test codes, 2-33
Mask command types, 6-13–16
math functions, 2-7–20

:MEASure
ALL, 7-52
ASAMples, 7-53
BPERiod, 7-54
BRATe, 7-54
DCD, 7-55
DEFine DELay, 7-56
DELay, 7-57
DUTYcycle, 7-58
ECRossing, 7-59
EHEIght, 7-60
ERATio, 7-60
ERUNits, 7-61
EWMax, 7-62
EWMin, 7-63
FALLtime, 7-64
FREQuency, 7-65
JPP, 7-66
JRMS, 7-66
LOWer, 7-67
MASK:DEFine:BFIT, 7-67
MASK:DEFine:DATA, 7-68
MASK:DEFine:INVert, 7-68
MASK:DEFine:MARGin, 7-69
MASK:DEFine:MHTS, 7-70
MASK:DEFine:OFAil, 7-71
MASK:DEFine:OPASs, 7-72
MASK:DEFine:RETest, 7-73
MASK:DEFine:TIME, 7-73
MASK:IDENtify, 7-74
MASK:SHOW, 7-74
MASK:STESt, 7-75
MASK:TEST, 7-75
MTYPe, 7-76
NPP, 7-76
NRMS, 7-77
NWIDth, 7-78
OLEVel, 7-79
OPMeter, 7-79
OVERshoot, 7-80
PAMPlitude, 7-81
PAVerage, 7-82
PBASe, 7-83
PERiod, 7-84
PHASe, 7-85
PMAX, 7-86
PMET, 7-86
PMIN, 7-87
PPP, 7-88

PREShoot, 7-89
PRMS, 7-90
PTIMe, 7-90
PTOP, 7-91
PUNIts, 7-91
PWDELta, 7-92
PWIDth, 7-93
PWSTArt, 7-94
PWSTop, 7-94
RISetime, 7-95
SCRatch, 7-95
SHOW, 7-96
SOURce, 7-96
TDELta, 7-97
THResholds, 7-98
TPOWer, 7-99
TREF, 7-100
TREF:OPERation, 7-100
TREF:T1, 7-101
TREF:T2, 7-101
TSAMples, 7-102
TSTArt, 7-103
TSTOp, 7-103
TVOLt, 7-104
UPPer, 7-105
VAMPlitude, 7-106
VAVerage, 7-107
VBASe, 7-108
VDELta, 7-108
VMAX, 7-109
VMIN, 7-110
VPP, 7-111
VRMS, 7-112
VSTArt, 7-113
VSTOp, 7-113
VTIMe, 7-114
VTOP, 7-114
ZLEVel, 7-115
Measure command types, 6-17–22
mnemonics
duplicate, 4-8
truncation, 5-3
mode, averaged, 4-17
multiple
commands, 4-13
queries, 4-21
subsystems, selecting, 4-13

N notation
convections/definitions, 5-12
numeric
program data, 4-11
variable, 4-20

O :OCALibrate
CALibrate, 7-115, A-7
ECALibrate:CHAN1, A-7
ECALibrate:CHAN1 (or CHAN2), 7-116
ECALibrate:CHAN2, A-7
ECALibrate:SKIP, 7-116, A-7
ECALibrate:XTRigger, 7-117, A-7
ERATio:FCOR, 7-117
ERATio:IOPT, 7-118
ERATio:XE1, 7-119
ERATio:XE2, 7-120
KILL, 7-120, A-6
LABel, 7-121
LDEFault, 7-121
OCALibrate:OCGain, 7-121, A-8
OCALibrate:PDARk, 7-122, A-7–8
OCALibrate:PMET:HHRange, 7-123, A-10
OCALibrate:PMET:HLRange, 7-123, A-9
OCALibrate:PMET:HMRange, 7-124, A-9
OCALibrate:PMET:LHRange, 7-124, A-9
OCALibrate:PMET:LLRange, 7-125, A-9
OCALibrate:PMET:LMRange, 7-125, A-9
SAVE, 7-126, A-10
SCALibrate, 7-126
SDISplay, 7-127
Ocalibrate command types, 6-23
offset voltage, 4-15
*OPC (Operation Complete) command, 7-7
output
command, 4-5
queue, 4-9
overlapped command, 5-11

P parameters
acquire, 4-17
waveform, 4-17
parser, 4-15, 5-4
peak measurement
frequency accuracy, 2-20
vertical accuracy, 2-19
plotter

HP-IB interface, 1-4
RS-232 interface, 1-4
polygon, 2-28
center reference, 2-30
definitions, 2-39
left-bottom reference, 2-29
power/voltage measurement
Eye:NRZ, 6-19
General Waveform, 6-19
preamble, 4-17
:PRINt, 7-127
print
factors, 1-10
resolution, 1-12
printer
HP-IB interface, 1-4
RS-232 interface, 1-4
printer/plotter output, 3-7
program data, 4-6
character, 4-11
numeric, 4-11
syntax, 4-10
program header
options, 4-9
program message
syntax, 4-5
terminator, 4-12
programming command tree, 5-6
programming examples, 8-2
HP BASIC with HP-IB, 8-4
HP BASIC with RS-232, 8-8
HP IBASIC for Windows with HP-IB, 8-12
HP IBASIC for Windows with RS-232, 8-17
mask downloading, 8-42
mask reading, 8-44
QuickBASIC with HP-IB, 8-22
QuickBASIC with RS-232, 8-29
QuickC with HP-IB, 8-36
programming guidelines, 4-2
programming instructions, header, 4-6
program overview, 4-16

Q query
command, 4-8
instructions, 4-6
multiple, 4-21
receiving, 4-18
response, 5-11
QuickBASIC with HP-IB, 8-22
QuickBASIC with RS-232, 8-29
QuickC with HP-IB, 8-36

R *RCL (Recall) command, 7-7
real-time clock
operating, 3-7
setting, 2-26
receiver conversion gain measurement, A-2
regions, 2-31
regulatory information, iii
remote calibration sequence, A-6
remote optical channel calibration, A-2
remote program interface
HP-IB, 4-3
RS-232, 4-3
remote programming
device address, 4-5
instructions, 4-6
output command, 4-5
program data, 4-6
program message syntax, 4-5
white space (separator), 4-6
remote programming module
installing, 1-3
math function, 2-7
operating, 2-2
operating characteristics, 3-3–8
Resolution softkey, 1-12
root level command
types, 6-5
root level commands
definition, 5-5
RS-232
baud rates, 3-8
cables, 1-5–8, 4-28
capabilities, 4-31
configuration, 1-10, 3-8, 4-31
connector, 3-8
controller mode, 4-31
data bits, 3-8, 4-32
device address, 4-33

front-panel menu, 4-31
hardware interface, 4-30
lockout, 4-33
menu softkey, 1-10
parity, 3-8
printer mode, 4-31
protocol, 3-8, 4-32
remote program interface, 4-3
remote programming, 1-4
select code, 4-33
stop bits, 3-8
three-wire interface, 4-29
*RST (Reset) command, 7-8
:RUN, 7-127

S
*SAV (Save) command, 7-9
sequential command, 5-11
Service Request Enable Register, 7-9
simple command header, 4-7
slope, 4-15
spectral leakage, 2-15
*SRE (Service Request Enable) command, 7-9
:STATus, 7-128
Status Byte Register, 7-11
*STB (Status Byte) command, 7-11
:STOP, 7-128
string variable, 4-19
subsystem commands, 5-7
definition, 5-5
:SYSTem
DSP, 7-128
ERRor, 7-129
KEY, 7-132
LOCK, 7-134
SETup, 7-135
System command types, 6-6
system dark noise level measurement, A-2

T
:TER (Trigger Event Register), 7-136
:TIMebase
DELay, 7-137
MODE, 7-138
RANGe, 7-139
REFerence, 7-139
SETup, 7-140
VERNier, 7-141
Timebase command types, 6-24
time measurement

Eye:NRZ, 6-18
General Waveform, 6-18
:TRACe
CLEAR, 7-141
DATA, 7-142
MODE, 7-143
SAVE, 7-143
trace
label, 2-24
recalling, 2-21
saving, 2-21
Trace command types, 6-25
trace memory
operating, 3-7
trace storage, 2-21–27
tree traversal rules, 5-7
*TRG (Trigger) command, 7-12
:TRIGger
COUPling, 7-144
HOLDoff, 7-144
LEVel, 7-145
MODE, 7-146
NREJect, 7-147
POLarity, 7-147
REJect, 7-148
SETup, 7-148
SLOPe, 7-149
SOURce, 7-149
TVHFrej, 7-150
TVMode, 7-150
Trigger command types, 6-26
trigger level, 4-15
trigger level/mode, 4-15
truncation rule, 5-3
*TST (Test) command, 7-12

U
user-defined
mask definition, 2-34
test file, 2-32
test file, codes, 2-33

V variable
numeric, 4-20
string, 4-19
vertical
channel, 4-15
range, 4-15
:VIEW, 7-151

W *WAI (Wait) command, 7-12
:WAVeform
BYTeorder, 7-151
DATA, 7-152
FORMat, 7-152
POINts, 7-153
PREamble, 7-154
SOURce, 7-155
TYPE, 7-155
XINCrement, 7-156
XORigin, 7-156
XREFerence, 7-156
YINCrement, 7-157
YORigin, 7-157
YREFerence, 7-157
waveform
data record, 4-17
parameters, 4-17
Waveform command types, 6-27–33
white space (separator), 4-6

X XON/XOFF, 4-32