Errata

Title & Document Type: 75000 Series C / E1740A Time Interval Analyzer User's Guide

Manual Part Number: E1740-90005

Revision Date: June 1, 1994

HP References in this Manual

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

About this Manual

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

Support for Your Product

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

www.tm.agilent.com

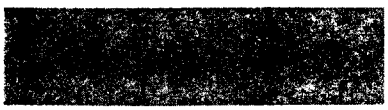
Search for the model number of this product, and the resulting product page will guide you to any available information. Our service centers may be able to perform calibration if no repair parts are needed, but no other support from Agilent is available.

Agilent Technologies

HEWLETT PACKARD

User's Guide

HP 75000 Series C
HP E1740A
Time Interval Analyzer



User's Guide

This guide describes how to configure, install, program, and service the HP E1740A Time Interval Analyzer VXI Module. The information in this guide applies to instruments having the number prefix listed below, unless accompanied by a "Manual Updating Changes" package indicating otherwise.

SERIAL PREFIX NUMBER: **3323 and above**

HP 75000 SERIES C

HP E1740A Time Interval Analyzer

Printed: June 1994

Printed in USA

Manual part number E1740-90005

Certification and Warranty

Certification

Hewlett-Packard Company certifies that this product met its published specification 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 (formerly National Bureau of Standards), to the extent allowed by the Institute's calibration facility, and to the calibration facilities of other International Standards Organization members.

Warranty

This Hewlett-Packard instrument product is warranted against defects in material and workmanship for a period of three years 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 detailed warranty information, see back matter.

Safety Considerations

General

This product and related documentation must be reviewed for familiarization with this safety markings and instructions before operations.

Warning Symbols That May Be Used In This Book

Instruction manual symbol; the product will be marked with this symbol when it is necessary for the user to refer to the instruction manual.

Indicates hazardous voltages.

Indicates earth (ground) terminal.

or

Indicated terminal is connected to chassis when such connection is not apparent.

Indicates Alternating current.

Indicates Direct current

Safety Considerations (Cont'd.)

WARNING

BODILY INJURY OR DEATH MAY RESULT FROM FAILURE TO HEED A WARNING. DO NOT PROCEED BEYOND A WARNING UNTIL THE INDICATED CONDITIONS ARE FULLY UNDERSTOOD AND MET.

CAUTION

Damage to equipment, or incorrect measurement data, may result from failure to heed a caution. Do not proceed beyond a CAUTION until the indicated conditions are fully understood and met.

Safety Earth Ground

An uninterruptible safety earth ground must be maintained from the mains power source to the product's ground circuitry.

WARNING

ANY INTERRUPTION OF THE PROTECTIVE GROUNDING CONDUCTOR (INSIDE OR OUTSIDE THE PRODUCT'S CIRCUITRY) OR DISCONNECTING THE PROTECTIVE EARTH TERMINAL WILL CAUSE A POTENTIAL SHOCK HAZARD THAT COULD RESULT IN PERSONAL INJURY. (GROUNDING ONE CONDUCTOR OF A TWO CONDUCTOR OUTLET IS NOT SUFFICIENT PROTECTION.) WHENEVER IT IS LIKELY THAT THE PROTECTION HAS BEEN IMPAIRED, THE PRODUCT MUST BE MADE INOPERATIVE AND BE SECURED AGAINST ANY UNINTENDED OPERATION.

For additional safety and acoustic noise information, see back matter.

Hewlett-Packard Company
Santa Clara Division
5301 Stevens Creek Boulevard
Santa Clara, California 95052-8059

Contents

In This Guide

This guide contains information needed to configure, install, program, and service the HP E1740A Time Interval Analyzer VXI Module.

The programming commands for the Time Interval Analyzer (TIA) conform to the *Standard Commands for Programmable Instruments (SCPI) Standard Version 1992.0*.

If you have programmed any HP instruments that have been introduced over the last few years, you will have seen a general trend toward the techniques specified in the SCPI standard. For example, several instruments are already using a hierarchy of commands that is similar to the command structure defined by the SCPI standard.

How to Use This Guide

How you use this guide depends upon how much you already know about programming instruments and how complex your measurement requirements are. Let's start by establishing your programming background, and then discuss the type of measurements you want to perform.

New Users

What You Should Understand

As a new user, you should understand that you must have some understanding of a high-level language such as Pascal, BASIC, C, or FORTRAN before you can use the command set defined in this guide to control the TIA. (In Chapter 15 there are programming examples provided in HP BASIC and Borland® Turbo C.) However, whatever language you use, command strings that control the TIA remain the same.

Learning to Program the TIA

To learn how to program the TIA, perform the following:

- Read Chapter 1, "Getting Started," for an introduction to the TIA features, start-up procedures, and a list of power-up default values.

- Scan summary tables 12-1 and 13-1 at the beginning of Chapter 12, "Definitions of the TIA IEEE 488.2 Common Commands," and Chapter 13, "Definitions of the TIA Subsystem Commands," (respectively) to get a feeling for the number and structure of commands available to you.
- Read Chapter 2, "Overview of How to Use and Program the TIA," for an overview of how to start programming the TIA, and an overview of the SCPI concepts as they relate to the HP E1740A TIA.
- Read Chapter 3, "Using the Measurement Instruction Commands," for steps needed to perform simplified programming of the TIA using the , :CONFigure, :FETCh?, :MEASure?, and :READ? commands.
- Read Chapters 4 through 9, that describe the five steps used to make most measurements with the TIA (i.e., select the time interval measurement, specify how the data will be acquired, configure the input channels for your signals, set the span and resolution, and acquire and analyze the data.).
- Read Chapter 10, "Data Formats," for information on how to specify the output format (ASCII, real, or integer) of your measurement for transferring numeric and array information to a computer for further analysis.
- Read Chapter 15, "Programming Examples," which provides programming examples for common measurement tasks you will want your TIA to perform.
- Modify some of the programming examples to select specific measurement functions. If the programs work, consider yourself an experienced programmer and use Chapter 12 and 13 as references for detailed information of all the TIA's SCPI commands.

Experienced Programmers

If you have programmed other instruments, you will probably be familiar with many of the concepts and techniques discussed in this guide.

Because the SCPI command set and some of the status reporting techniques are new, you may want to use the following sequence to learn the TIA programming requirements:

- Look over the steps for a new user and perform any that you think are applicable to your current level of knowledge. In particular, read Chapter 1, "Getting Started," and look at the programming examples provide in Chapter 15, "Programming Examples."
- Review summary tables 12-1 and 13-1 in Chapter 12 and 13, respectively. If this chapter contains sufficient information to get you started, write some programs to explore the TIA's capabilities. If you need additional information on any command, refer to the applicable command description in Chapter 12 or Chapter 13.
- Review the remaining information in this guide to determine what is applicable to your programming requirements.

If you need more information than is contained in this guide, see the section in this chapter titled "Related Documentation."

Applications

After you have read the appropriate information and written some measurement programs, you may want to expand the scope of your applications. The following two techniques are explained in detail:

- If you are going to write interrupt-driven programs (or if you just want to determine the status of the TIA), read Chapter 11, "Status Reporting."
- If you are going to write programs to transfer data between the TIA and an external computer, read the section titled "Overview of Response Message Formats" in Chapter 2, and Chapter 10, "Data Format."

User's Guide Contents

The following information is contained in this guide:

- Table of Contents
- This preface, "In This Guide," introduces you to the user's guide.
- Chapter 1, "Getting Started," introduces you to the HP E1740A TIA and provides configuration, installation, a power-up procedure, and introductory programming information.
- Chapter 2, "Overview of How to Use and Program the TIA," provides some overview information about the steps that are commonly used to make a measurement, briefly explains the SCPI elements and formats, and SCPI syntax conventions used in this guide.
- Chapter 3, "Using the Measurement Instruction Commands," describes how to simplify your programming using the Measurement Instruction commands.
- Chapter 4, "Selecting the Time Interval Measurement," describes how to select different types of time interval measurements that the TIA can make.
- Chapter 5, "Specifying How the Data Will be Acquired," describes how to set the TIA to collect your data of interest.
- Chapter 6, "Configuring Channel 1 and 2 Inputs," describes how to set input conditions of the TIA's input channels.
- Chapter 7, "Setting the Span and Resolution," describes tradeoffs when setting span and resolution.
- Chapter 8, "Initiating Measurements," describes how to initiate measurements and acquire data.
- Chapter 9, "Analyzing Data With Window Margin," describes window margin analysis.
- Chapter 10, "Data Format," describes how to specify the format for various types of data retrieval.
- Chapter 11, "Status Reporting," describes the status reporting structure of the TIA, and describes how to program the TIA for status reporting.

- Chapter 12, "Definitions of the TIA IEEE 488.2 Common Commands," provides a summary list of all the Common commands, and detailed descriptions of each Common command that is available for the TIA.
- Chapter 13, "Definitions of the TIA Subsystem," provides a summary list of all the Subsystem commands, and detailed descriptions of each Subsystem command that is available for the TIA.
- Chapter 14, "Error Messages," provides listings of error messages that the TIA could generate along with descriptions of possible causes for the errors.
- Chapter 15, "Programming Examples," provides programming examples for some tasks that you will want your TIA to perform.
- Chapter 16, "Maintenance," provides service information; such as performance tests, calibration procedures, troubleshooting and diagnostic tests, disassembly and reassembly procedures, and replaceable parts information.
- Chapter 17, "Specifications," lists the specifications and characteristics of the TIA.
- Index

Related Documentation

This section contains a list of documentation related to the use of the TIA. Additional information that you may find useful can be found in the following publications:

1. **Beginner's Guide to SCPI (HP Part Number H2325-90001,** July 1990 Edition).
2. **Beginner's Guide to SCPI, Barry Eppler** (Hewlett-Packard Press, Addison-Wesley Publishing Co. 1991).
3. **Standard Commands for Programmable Instruments (SCPI), Version 1992.0.**

   This standard is a guide for the selection of messages to be included in programmable instrumentation. It is primarily intended for instrument firmware engineers. However, you may find it useful if you are programming more than one instrument that claims conformance to the SCPI standard. You can verify the use of standard SCPI commands in different instruments.

   To obtain a copy of this standard, contact:

   SCPI Consortium
   8380 Hercules, Suite P3
   La Mesa, CA 91942
   Phone: (619) 697-8790
   FAX: (619) 697-5955

4. **The International Institute of Electrical Engineers and Electronic Engineers, IEEE Standard 488.1-1987, IEEE Standard Digital Interface for Programmable Instrumentation.**

   This standard defines the technical details required to design and build an HP-IB (IEEE 488.1) interface. This standard contains electrical specification and information on protocol that is beyond the need of most programmers. However, it can be useful to clarify formal definitions of certain terms used in related documents.

   To obtain a copy of this standard, write to:

   The Institute of Electrical and Electronic Engineers Inc.
   345 East 47th Street
   New York, NY 10017 USA

5. **The International Institute of Electrical Engineers and Electronic Engineers, IEEE Standard 488.2-1987, IEEE Standard Codes, Formats, Protocols, and Common Commands for Use with ANSI/IEEE Std 488.1-1987 Programmable Instrumentation.**

   This standard defines the underlying message formats and data types used in SCPI. It is intended more for firmware engineers than for instrument users/programmers. However, it can be useful if you need to know the precise definition of specific message formats, data type, or common commands.

   To obtain a copy of this standard, write to:

   The Institute of Electrical and Electronic Engineers Inc.
   345 East 47th Street
   New York, NY 10017 USA

6. **Hewlett-Packard Company, BASIC 5.0/5.1 Interfacing Techniques Vol 2., Specific Interfaces, 1987.**

   This HP BASIC manual contains a good non-technical description of the HP-IB (IEEE 488.1) interface in Chapter 12, "The HP-IB Interface." Subsequent revisions of HP BASIC may use a slightly different title for this manual or chapter. This manual is the best reference on I/O for HP BASIC programmers.

   To obtain a copy of this manual, contact your nearest Hewlett-Packard Sales office.

7. **Hewlett-Packard Company, Tutorial Description of the Hewlett-Packard Interface Bus, 1987.**

   To obtain a copy of this manual, contact your nearest Hewlett-Packard Sales office.

1

Getting Started

Chapter Contents

This chapter describes how to configure, install, and begin using the HP E1740A Time Interval Analyzer (TIA). The main sections in this chapter are:

What You Need to Use the TIA

The HP E1740A Time Interval Analyzer (TIA) is a two-slot, C-size VXI module. It is capable of measuring and analyzing time intervals in a variety of ways. As with any VXI instrument module, other elements are needed in order to operate the TIA. The following list summarizes these necessary elements:

- a VXI C-size mainframe
- an embedded VXI computer or an external computer with a Slot 0 command module

VXI C-Size Mainframe

The number of mainframe slots required for your application is the typical deciding factor when selecting an appropriate mainframe. Other specifications include the dc and dynamic current available from each power supply, along with the cooling capabilities of the mainframe.

Embedded Personal Computer (PC)

For dedicated use, an embedded VXI computer provides enhanced data transfer performance and flexibility. This computer is installed into the mainframe and includes the VXI interface needed for communicating with and controlling other device modules in the mainframe.

External Personal Computer (PC) with Command Module

When using an external computer, you need a command module which plugs into the mainframe in Slot 0. The command module provides Slot 0 and Resource Manager capabilities as well as an external interface to the PC. This makes possible communication between the computer and any VXI modules in the mainframe.

Communicating With the TIA Module

The HP E1740A uses the SCPI command language. SCPI (Standard Commands for Programmable Instruments) is an ASCII-based instrument command language designed for test and measurement instruments. The commands are documented in this user's guide. Use these commands in the programs you develop to control the TIA.

Communication with the TIA usually requires a library of commands to perform the I/O tasks necessary to send and receive data from the TIA. The command library is normally matched to the interface card and the programming language preferred by you.

Embedded PC

An embedded PC has the interface for controlling the TIA built into the computer. The command library and the programming language need to be supplied.

External PC

For example, when using an external PC as the TIA controller, the following items would support communication with the TIA:

- HP 82335 HP-IB Interface Card.
- HP-IB Command Library for your programming language.
- BASIC, Pascal, or C language of choice.
- HP E1406A Command Module.

Preparing the TIA for Use

Before you can use the TIA, it must be configured correctly and installed in the mainframe. This section describes the TIA faceplate (LEDs and connectors), how to configure the hardware settings, and how to install the TIA. An introduction to programming and a simple program for the TIA is provided near the end of this chapter.

HP E1740A Faceplate Description

Refer to Figure 1-1 for the location of the LEDs and connectors described in this section.

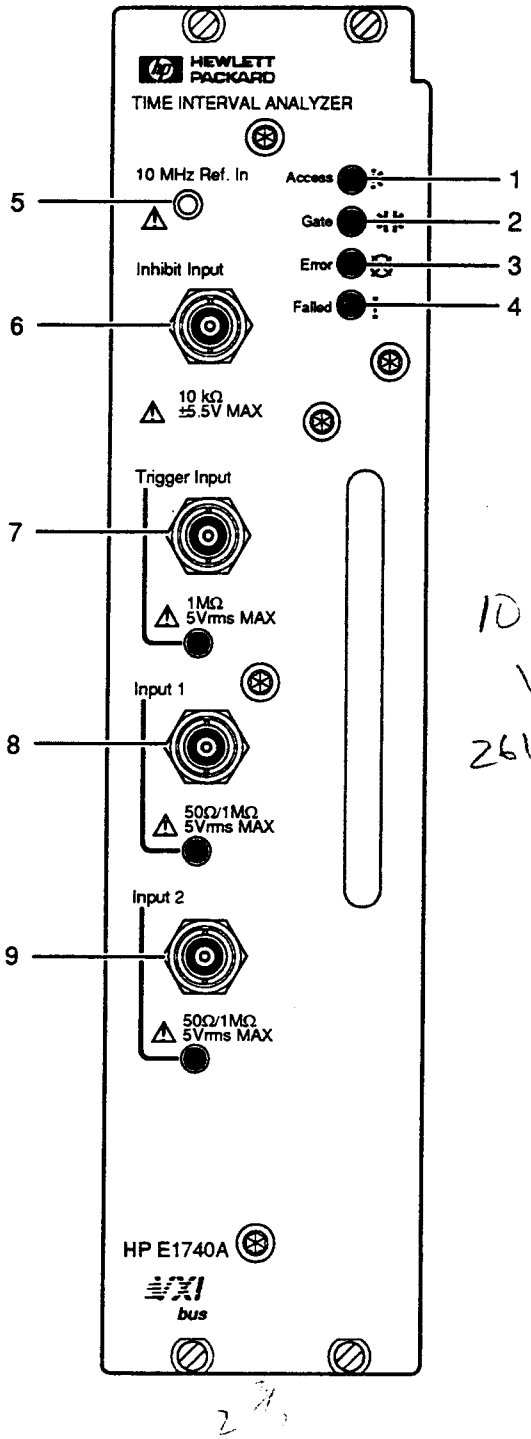


Figure 1-1. The HP E1740A TIA Faceplate

Faceplate Annunciator LEDs

1 Access

This green LED lights to indicate a data transfer across the TIA's VXIbus interface is taking place.

2 Gate

This green LED indicates when the measurement gate is open. Whenever the gate is open, the TIA can acquire data.

3 Error

This red LED lights when a programming error has occurred. Use the SYST:ERR? command to read the error queue. This LED also indicates when the hardware is being reconfigured, such as when the resolution is changed.

4 Failed

This red LED lights when a non-recoverable VXIbus error or a self-test failure occurs. Cycle power to the mainframe as a first step in recovering from this failure state.

Faceplate Connectors

5 10 MHz Ref. In

An SMC input for phase-locking the TIA to an external 10 MHz reference.

6 Inhibit Input

A BNC input for a signal you can use to selectively prevent the TIA from making measurements.

7 Trigger Input

A BNC input for the signal you want to use to synchronize the start of an acquisition. The green LED flashes when the signal is being detected by the TIA.

8 Input 1

A BNC input for the signal to be measured. Typically, the data signal is measured at Input 1. See Chapter 17, "Specifications," for a summary of the specifications and operating characteristics. The green LED flashes when the signal is being detected by the TIA.

9 Input 2

A BNC input for a signal to be measured. Typically, the clock signal is measured at Input 2. Chapter 17 contains the specifications of the inputs. The green LED flashes when the signal is being detected by the TIA.

Device Summary Information

Device Information
Device type: message-based servant
C-size (2 slot)
Connectors: P1 and P2
Addressing modes: A16/A24/D16 Master, A16/D16 Slave
Not dynamically configurable
Non-interrupter/non-interrupt handler
VXIbus Revision Compliance: 1.3
SCPI Revision: 1992.0
See side of module for power/cooling requirements

HP E1740A VXIbus Factory Settings

The HP E1740A TIA is configured at the factory as shown in Table 1-1.

Table 1-1. HP E1740A VXIbus System Factory Settings.

Parameter	Setting
Logical Address	48
Bus Request Level	3

TIA Logical Address

In a VXIbus system, every device must be in the servant area of a commander (with the exception of the top-level commander). A commander is a command module, such as the HP E1406A Command Module, or an embedded controller. The HP E1740A TIA logical address is used to place the TIA in the servant area of a commander and to address the TIA (see "Addressing the TIA" or "Using an Embedded Computer" later in this chapter).

Assigning the TIA to a Commander

Be aware of the following when assigning the HP E1740A TIA to a commander:

- A commander's servant area begins at the next logical address after its own and includes the number of contiguous logical addresses specified by the servant area size setting. For example, if the command module has a logical address of 0 and a servant pointer setting of 255, the command module will serve as the interface for all instrument modules with logical addresses between 1 and 255.
- The HP E1406A Command Module has a factory-set logical address of 0 and a servant area switch setting of 255. Using these factory settings, it is not necessary to change the logical address of the TIA (48) to place it in the servant area of the command module.

- If your system uses an external computer and the HP E1406A Command Module, put the TIA in the servant area of the command module. This enables the command module to function as the HP-IB interface to the TIA.

The factory-set HP E1740A TIA logical address switch is shown in Figure 1-2.

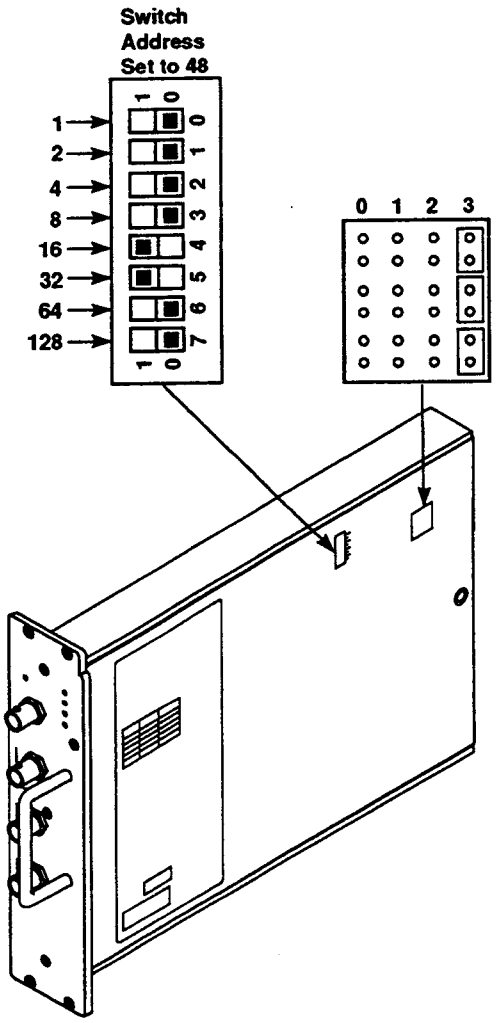


Figure 1-2. Logical Address Switch and Bus Request Setting

TIA Bus Request Level

The bus request level is the priority at which the HP E1740A TIA can request the use of the Data Transfer Bus.

- There are four bus request lines (0–3) from which one is selected (Figure 1-2). Bus request line 3 has the highest priority, bus request line 0 has the lowest priority.
- Bus request line 3 is normally used unless there is a particular module within the mainframe that requires a different data interface priority than other modules.
- It is normally not necessary to change the bus request level setting (3) on the TIA.

A drawing of the bus request jumper settings is shown in Figure 1-2. A bus request level of 3 is set.

Installing/Removing the TIA Module

The HP E1740A TIA can be installed in any mainframe slot except Slot 0.

To Install the TIA Module

Always check that the mainframe power is OFF before installing any boards or instruments into the backplane connectors.

1 **Inspect the pins on the back of the module for straightness before inserting into mainframe.**

2 **Install the module by lining up the module with the slot guides and then pressing the module firmly into the backplane connectors as shown in Figure 1-3.**

NOTE

The screws located at the top and bottom of the TIA module can be used to secure the installed module to the mainframe.

When installing a VXI module into a mainframe that aligns the modules vertically (orientation shown in Figure 1-3), sometimes the top installation screws may tilt down and become blocked by the mainframe. This prevents the module from connecting to the backplane. A remedy for this is to pull these screws toward the front of the module before performing the installation.

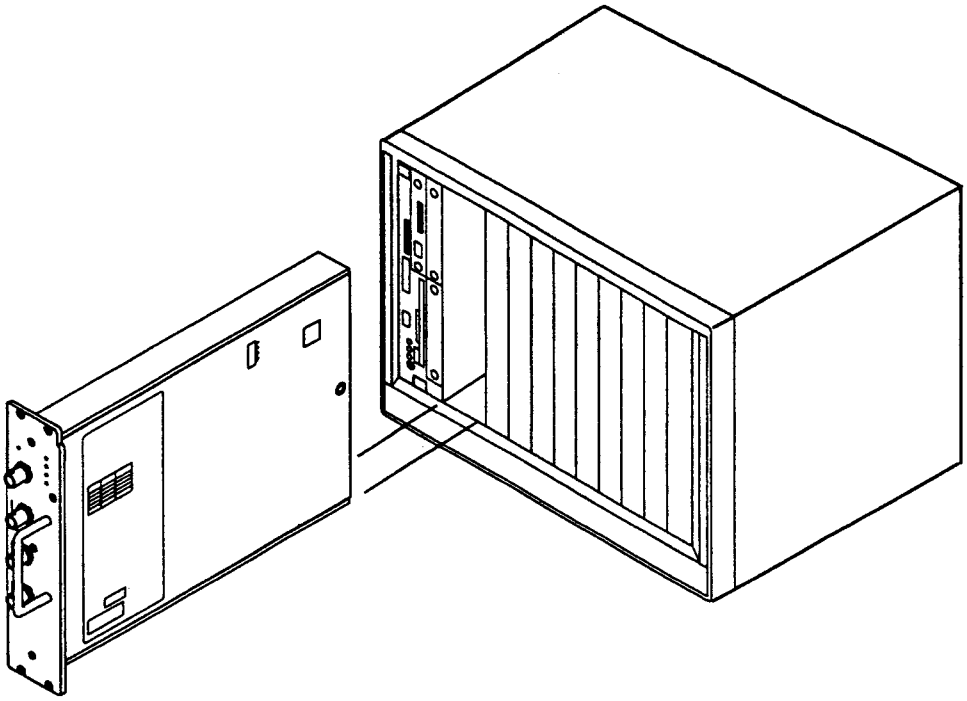


Figure 1-3 Installing the HP E1740A Time Interval Analyzer

To Remove the TIA Module

1 **Loosen the four screws that attached the module to the mainframe.**

2 **Remove the module by pulling on the faceplate handle of the TIA. This extracts the module from the backplane connectors allowing the module to be removed.**

A slight side-to-side motion while pulling may make it easier to remove the module.

Addressing the TIA

To Address the TIA When Using an Embedded PC Controller

The HP E1740A can be programmed across the VXIbus backplane from an embedded computer. Consult the documentation supplied with your embedded PC for information on how to address the TIA. The specifics will depend on the programming language you are using and the development tools support for that language provided by the manufacturer of the PC.

To Address the TIA When Using an External Controller and Command Module

This description assumes you are using an external controller with an HP-IB interface card and a command module to control the TIA. There are three parts to the address:

- the interface select code.
- the command module's primary HP-IB address.
- the TIA module's secondary HP-IB address.

This form of an address in an HP BASIC statement appears as:

```
OUTPUT 70906;"CONFigure:XTIMe:TINTerval;:TRIGger:SOURce EXTernal"
```

Interface Select Code (7). Determined by the address of the HP-IB interface card in the controller. In most Hewlett-Packard controllers, this card has a factory-set address of 7.

Primary HP-IB Address (09). This is the address of the HP-IB port on the command module. Valid addresses are 0 to 30. The HP E1406A Command Module has a factory-set address of 9.

Secondary HP-IB Address. This address is derived from the logical address of the TIA by dividing the logical address by 8. Thus, for the HP E1740A factory-set logical address of 48, the secondary address is 6.

Commands to Start Using the TIA

The introductory commands in this section include:

- TIA Self-Test
- Resetting the TIA and clearing its status registers
- Querying the TIA power-on/reset settings
- Checking for errors
- Making measurements

Power-up and Health Check

If TIA is not already operating, follow these start-up procedures.

Start

1 **If you are using an external PC, turn ON the computer before the VXIbus mainframe.**

2 **Turn ON the display monitor.**

3 **Turn ON the VXIbus mainframe.**

Expected Behavior

If you are using an embedded PC, the PC should execute a self-test routine.

The HP E1740A TIA will also execute a self-test routine when power is turned on.

All LEDs on the HP E1740A will stay off until the end of the self-test. The "Access" and "Gate" LEDs will flash once at the end of the self-test.

Verify Communication With the TIA

A convenient way to test the interface with the TIA is to send a request for the TIA to identify itself. Send the command, "*IDN?", and the TIA should return the following ASCII response:

HEWLETT-PACKARD,E1740A,0,xxxx

- **HEWLETT-PACKARD** is the manufacturer.
- **E1740A** is the model number.
- **0** is returned instead of the TIA serial number.
- **xxxx** is the firmware version.

TIA Self-Test

The self-test query causes the TIA to execute an internal self-test and report if any errors are detected. The TIA self-test is executed with the command:

```
*TST?
```

Upon completion of the test, one of the self-test codes listed in Table 1-2 is returned.

Table 1-2. Self-Test Results

Self-Test Code	Description
0	Test passed
1	Test failed. An error message describes the failure.

Resetting and Clearing the TIA

The commands used to reset and clear the TIA are:

```
*RST
*CLS
```

Resetting the TIA sets it to its power-on configuration and clearing the TIA clears its status registers. Status register programming is covered in Chapter 11 in this guide. More complete descriptions of the reset and clear commands are included in Chapter 12, "Definitions of the TIA IEEE 488.2 Common Commands."

Querying the TIA Configuration

After resetting the TIA or cycling power, the TIA parameters are set to their power-on values. These values are listed in the Table 1-3, located at the end of this chapter.

Unless otherwise stated, the commands used to set parameters can be used to query their current state by adding a question mark to the command. An example of this follows:

```
[:SENSe]:EVENt2:LEVel?
```

This command query will return the current voltage threshold setting for Input 2.

Checking for Errors

The query :SYSTem:ERRor? will request the first error in the TIA's error queue. The error messages are described in the appendix.

Example Commands

This section introduces the programming structure used to make measurements with the TIA. Additional details and descriptions about overall control of the TIA are included in Chapter 2, "Overview of How to Use and Program the TIA." Program segments include comments that describe the action of each command line.

Single-Source Time Interval Program

The following program segment shows how to configure the TIA to make a one-channel time interval acquisition and return statistical results of the measurement data. An actual program would have addressing, enter, and output statements.

Example Program Segment

```
*RST
*CLS
CONFigure:XTIMe:TINTerval DEFault,DEFault,(@1)
FORMat ASCii
INPut1:COUPling DC;IMPedance 1E6;ROUTe SEParate
EVENt1:LEVel 0.15;SLOPe POSitive
TRIGger:SOURce EXTernal;LEVel TTL;SLOPe POSitive
TRIGger:DELay:SOURce TIMer;TIMer 10E-3
ACQuisition:PACing STEP;STEP 3
ACQuisition:SOURce MCOunt;MCOunt 1000
TRIGger:COUNt 100
INITiate
FETCh:TINTerval:MEAN?
```

Program Description

```
*RST
*CLS
```

These two commands reset the TIA and clear the status registers.

```
:CONF:XTIM:TINT DEF,DEF,(@1)
:FORM ASC
```

The CONFigure command selects sequential time interval measurements to be acquired on Input 1. The FORMat command specifies the ASCII output format for measurement data.

```
:INP1:COUP DC;IMP 1E6;ROUT SEP
:EVEN1:LEV 0.15;SLOP POS
```

The Input command sets signal conditioning for Input 1. Coupling is set to DC, input impedance to 1 MΩ, and the input channels are set to separate. The next command line sets the Input 1 trigger conditions for the input signal that will be detected by the TIA. A 10:1 probe is being used on a TTL input signal so "0.15" volts is specified. The desired slope of the signal is set as the rising edge.

```
:TRIG:SOUR EXT;LEV TTL;SLOP POS
:TRIG:DEL:SOUR TIM;TIM 10E-3
```

An external TTL signal is used at the Trigger input to trigger the start of the acquisition on a rising edge. A time delay of 10 ms is specified to occur after the Trigger signal. This delay holds off the start of the data acquisition. The TTL level is 1.5 V.

```
:ACQ:PAC STEP;STEP 3
:ACQ:SOUR MCO;MCO 1000
:TRIG:COUN 100
```

Time interval measurements are set to start on every third edge of the input signal on Input 1. One thousand measurements will be collected for each acquisition. This will occur 100 times. A trigger signal is required to begin each acquisition.

```
:INIT
```

The INITiate command makes the TIA ready to start the first acquisition.

```
:FETC:TINT:MEAN?
```

Once the measurements are completed, the FETCh? command will transfer the ASCII statistical mean result from the TIA's internal memory to the TIA's output buffer. The result can then be read into your controller. The returned value is the mean for 100,000 measurements (100 acquisitions of 1000 measurements).

*RST Response

The IEEE 488.2 *RST command sets the instrument to a specified state. Use this command at the beginning of your program to put the TIA into a known state. (Use *CLS to clear the status event registers and the SCPI error queue.)

The states of commands affected by *RST command are listed in Table 1-3.

Table 1-3. HP E1740A *RST State

Command Header	Parameter	State
:CALCulate:WMARgin:EXTRapolation:RANGe:LOWer	<numeric_value>	-18.0
:CALCulate:WMARgin:EXTRapolation:RANGe:UPPer	<numeric_value>	-1.5
:CALCulate:WMARgin:MLEVel	<numeric_value>	-10.0
:CALCulate:WMARgin:SEGMents:CENTer	<numeric_value>	20E-9
:CALCulate:WMARgin:SEGMents:COUNt	<numeric_value>	7
:CALCulate:WMARgin:SEGMents:ENABle	<non-decimal numeric> | <numeric_value> |ALL	ALL
:CALCulate:WMARgin:SEGMents:OFFSet	<numeric_value>	0.0
:CALCulate:WMARgin:SEGMents:WIDTh	<numeric_value>	10E-9
:CONF?		"XTIM:VOLT:TINT 0,2048, (@1), (@1)"
*EMC	<NR1>	0 (i.e., disabled) 1 (i.e., enabled)
:FORMat[:DATA]	ASCii | REAL | INTeger	ASCii
:INPut[1|2]:COUPling	AC | DC	DC
:INPut[1|2]:IMPedance	<numeric_value> [OHM]	1E6 OHM
:INPut[1|2]:ROUTe	COMMon | SEParate	SEParate
[:SENSe]:ACQuisition:MCOunt	<numeric_value>	1000.0
[:SENSe]:ACQuisition:PACing[:SOURce]	IMMediate | RANDom | STEP	IMMediate
[:SENSe]:ACQuisition:PACing:STEP	<numeric_value>	2
[:SENSe]:ACQuisition:SOURce	MCOunt | TIMer	MCOunt
[:SENSe]:ACQuisition:TIMer	<numeric_value> [S]	10E-6
[:SENSe]:EVENt[1|2]:HYSTeresis:RELative	<numeric_value> [PCT]	50 PCT
[:SENSe]:EVENt[1|2]:LEVel	<numeric_value> [V]	0
[:SENSe]:EVENt[1|2]:SLOPe	POSitive | NEGative	POSitive
[:SENSe]:FUNCtion	<sensor_function>	"XTIM:TINT 1"
[:SENSe]:HISTogram:ACCumulate[:STATe]	<Boolean>	OFF
[:SENSe]:HISTogram:RANGe:OFFSet	<numeric_value>	0.0
[:SENSe]:HISTogram:RANGe:RESolution	<numeric_value>	48.8E-12
[:SENSe]:HISTogram:RANGe[:UPPer]?		100E-9

Chapter Contents

This chapter provides an overview of how to use the TIA (Time Interval Analyzer). Details on using the TIA are provided in Chapters 3 through 11. The following is provided in this chapter:

2

Overview of How to Use and Program the TIA

Starting to Program the TIA

How you program the TIA will be determined by the level of control you need over how your measurements are acquired. The SCPI command language provides commands for simplified programming of instruments. They are :CONFigure, :FETCh? :MEASure? and :READ? which are briefly described in this section. (Refer to Chapter 3, "Using the Measurement Instruction Commands," and the section titled "Measurement Instructions" in Chapter 13, "Definitions of the TIA Subsystems," for detailed information.)

The purpose of these commands is to acquire data using a set of high-level instructions. These commands are structured to allow you to trade off interchangeability with fine control of the measurement process. The :MEASure? query provides a complete capability by configuring the TIA, taking a measurement, and putting the results in the Output Queue in one operation.

When more precise control of the measurement is required, the :CONFigure and :READ? commands can be used. :CONFigure performs the configuration portion of the measurement. :READ? performs the data acquisition and post processing (if any), and then it places the results in the Output Queue. This allows generic configuration of the instrument using :CONFigure, and then customization of the measurement with other commands (for example, from the [:SENSe] subsystem). :READ? completes the measurement process.

The :READ? command, in turn, is composed of the :INITiate[:IMMediate] and :FETCh? commands. :INITiate[:IMMediate] performs the data acquisition. :FETCh? performs the post-processing function (if any) and places the result in the Output Queue. This allows more than one :FETCh? on a single set of acquired data.

Table 2-1 summarizes the Measurement Instruction commands.

How to Use the TIA

To make most measurements, the following five-step sequence is suggested:

1. Select the time interval measurement.
2. Specify how the data will be acquired.
3. Configure the input channels for your signals.
4. Set the span and resolution.
5. Acquire and analyze the data.

Chapters 4 through 8 describe the commands used to accomplish the five tasks listed above.

The following section titled "Starting to Program the TIA," provides a general description of the Measurement Instruction commands that can be used to acquire data by using many of the TIA's default settings. At first, this is useful to verify preliminary setup and data transfer between the TIA and your controller. Then you can investigate more detailed commands as you become familiar with the features and want to customize the TIA setup to meet your needs.

Overview of the Command Types and Formats

There are two types of HP E1740A programming commands: IEEE 488.2 Common Commands and Standard Commands for Programmable Instruments (SCPI). The IEEE 488.2 Common Commands control and manage communications between the HP E1740A and the controller or personal computer. The SCPI commands control instrument functions. The format of each type of command is described in the following paragraphs.

Common Command Format

The IEEE 488.2 Standard defines the Common commands as commands that perform functions like reset, self-test, status byte query, and identification. Common commands always begin with the asterisk (*) character, and may include parameters. The command keyword is separated from the first parameter by a space character. Some examples of Common commands are as follows:

```
*RST        *IDN?        *ESE 32
```

SCPI Command and Query Format

SCPI commands perform functions like instrument setup. A subsystem command has a hierarchical structure that usually consists of a top level (or root) keyword, one or more lower-level keywords, and parameters. The following example shows a command and its associated query:

```
:INPut:COUPling AC
:INPut:COUPling?
```

INPut is a root-level keyword with COUPling the second level keyword, and AC is the command parameter.

SCPI Conformance Information

The SCPI commands used in the HP E1740A are in conformance with the SCPI Standard Version 1992.0.

Table 2-1. Measurement Instruction Commands

Summary of the Measurement Instruction Commands	
:MEASure?	This command is the simplest to use, but allows few additional possibilities. This command lets the TIA configure *itself* for an optimal measurement, initiate a measurement, and return the result; that is, it provides a complete measurement sequence (:MEAS query is equivalent to the :CONF, :INIT, :FETC? command sequence, but with no flexibility.)
:CONFigure :READ?	The combined use of these two commands allows for more control when the TIA performs measurement, initiates measurement, and returns the result.
:CONFigure :INITiate :FETCh?	The combination of these commands allows for the most flexibility . This command sequence configures the TIA, initiates the measurement as specified, and returns the result.

Refer to Chapter 3, "Using the Measurement Instruction Commands," for more information.

Abbreviated Commands

The command syntax shows most keywords as a mixture of upper and lower case letters. Upper case letters indicate the abbreviated spelling for the command. For better program readability, you may send the entire keyword. The HP E1740A accepts either command form and is not case sensitive.

For example, if the command syntax shows CALCulate, then CALC and CALCULATE are both acceptable forms. Other forms of CALCulate, such as CALCU or CALCULA will generate an error. You may use upper and/or lower case letters. Therefore, CALCULATE, calculate, and CaLcUlAtE are all acceptable.

Keyword Separator

A colon (:) always separates one keyword from the next lower-level keyword as shown below:

```
:INPut:COUPling?
```

Optional Keyword

Optional keywords are those which appear in square brackets ([ ]) in the command syntax. (Note that the brackets are not part of the command and are not sent to the TIA.)

Suppose you send a second level keyword without the preceding optional keyword. In this case, the TIA assumes you intend to use the optional keyword and responds as if you had sent it.

Examine the portion of the [:SENSe] subsystem shown below:

```
[:SENSe]
  :ROSCillator
    :SOURce EXTernal
```

The root-level keyword [:SENSe] is an optional keyword. To set the TIA's reference oscillator source to external, you can use either of the following:

```
:SENS:ROSCillator:SOUR EXT
                or
:ROSCillator:SOUR EXT
```

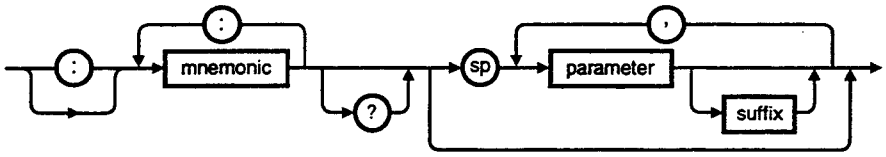
Elements of SCPI Commands

A program command or query is composed of functional elements that include a header (or keywords with colon separators), program data, and terminators. These elements are sent to the TIA over the VXIbus as a sequence of ASCII data messages. Examples of a typical Common Command and Subsystem Command are:

```
OUTPUT 70906;"*CLS"
OUTPUT 70906;":INP1:COUP AC;IMP 1.0 MOHM"
```

Subsystem Command Syntax

Figure 2-1 shows the simplified syntax of a Subsystem Command. You must use a space (SP) between the last command mnemonic and the first parameter in a Subsystem Command. Note that if you send more than one parameter with a single command, you must separate adjacent parameters with a comma.

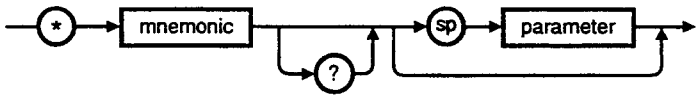


NOTE: sp = space. ASCII character decimal 32

Figure 2-1. Simplified Program Command Syntax Diagram

Common Command Syntax

Figure 2-2 shows the simplified syntax of a Common Command. You must use a space (SP) between the command mnemonic and the parameter in a Common Command.



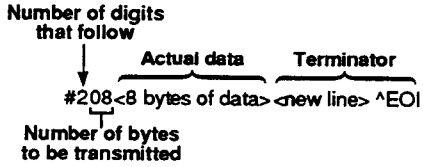
NOTE: sp = space. ASCII character decimal 32

Figure 2-2. Simplified Common Command Syntax Diagram

Parameter Types

Table 2-2 contains explanations and examples of parameter types. Parameter types may be numeric value, Boolean, literal, NRf, string, non-decimal numeric, or arbitrary block.

Table 2-2. Command and Query Parameter Types

TYPE	EXPLANATIONS AND EXAMPLES
<numeric value>	Accepts all commonly used decimal representation of numbers including optional signs, decimal points, and scientific notation: 123, 123e2, -123, –1.23e2, .123, 1.23e–2, 1.23000E–01.
<Boolean>	Represents a single binary condition that is either true or false: 1 or ON, 0 or OFF (Query response returns only 1 or 0.)
<literal>	Selects from a finite number of choices. These parameters use mnemonics to represent each valid setting. An example is the INPut:COUPling AC | DC command parameters (AC | DC).
<NRf>	Flexible numeric representation. Only positive integers are used for NRf parameters in the TIA.
<string>	A string parameter is delimited by either single quotes or double quotes. Within the quotes, any characters in the ASCII 7-bit code may be specified. The following HP BASIC program statement sends a command containing a <string> parameter: `OUTPUT 70906;"FUNC 'XTIM:TINT'"`
<non-decimal numeric>	Format for specifying hexadecimal (#H1F), octal (#Q1077), and binary (#B10101011) numbers using ASCII characters. May be used in :STATus subsystem commands.
<arbitrary block>	The syntax is a pound sign (#) followed by a non-zero digit representing the number of digits in the subsequent decimal integer. The decimal integer specifies the number of 8-bit data bytes being sent. This is followed by the actual data. The terminator is a line feed asserted with EOI. For example, for transmitting 8 bytes of data, the syntax might be:  The "2" indicates the number of digits that follow and the two digits "08" indicate the number of *data* bytes to be transmitted. A zero-length block has the syntax: #0<new line>^EOI

Implied Channel (Optional Numeric Keyword Suffix)

Some commands allow specifying a channel with an optional numeric keyword suffix. These commands will show the channel numbers within square brackets. The brackets are not part of the command and are not sent to the TIA.

For example, :INPut[1|2]:COUPling AC | DC represents coupling commands for channels 1 and 2:

```
:INPut[1]:COUPling  AC | DC
:INPut2:COUPling  AC | DC
```

If you do not specify the channel number, the implied channel is 1. For example, you can send either of the following to configure channel 1's coupling to AC:

```
:INPut1:COUPling AC
          or
:INPut:COUPling AC
```

Suffix Multipliers

Table 2-3 lists the suffix multipliers that can be used with suffix elements (except PCT and DEG).

Table 2-3. Suffix Multipliers

DEFINITION	MNEMONIC	NAME
1E18	EX	EXA
1E15	PE	PETA
1E12	T	TERA
1E9	G	GIGA
1E6	MA (or M for OHM and HZ)*	MEGA
1E3	K	KILO
1E-3	M (except for OHM and HZ)*	MILLI
1E-6	U	MICRO
1E-9	N	NANO
1E-12	P	PICO
1E-15	F	FEMTO
1E-18	A	ATTO

*The suffix units, MHZ and MOHM, are special cases that should not be confused with <suffix multiplier>HZ and <suffix multiplier>OHM.

Command Terminator

A command may be terminated with a <new line> (ASCII character decimal 10), an EOI (End-of-Identify) asserted concurrent with last byte, or an EOI asserted concurrent with a <new line> as the last byte.

Parameter Separator

If you send more than one parameter with a single command, you must separate adjacent parameters with a comma.

Suffixes

A suffix is the combination of suffix elements and multipliers that can be used to interpret the <numeric value> sent. If a suffix is not specified, the TIA assumes that <numeric value> is unscaled (that is, Volts, seconds, etc.)

For example, the following two commands are equivalent:

```
OUTPUT 70906;"INP:IMP 1 MOHM"
OUTPUT 70906;"INP:IMP 1E+6"
```

Suffix Elements

Suffix elements, such as HZ (Hertz), S (seconds), V (volts), OHM (Ohms), PCT (percent), and DEG (degrees) are allowed within this format.

For example, sending :INP:COUP AC;IMP 50 is equivalent to sending:

```
:INP:COUP AC
:INP:IMP 50
      or
:INP:COUP AC;:INP:IMP 50
```

The ":" must be present to distinguish another root level command. For example:

```
:INP:COUP AC;:OUTP:ECLT0 ON
```

is equivalent to sending:

```
:INP:COUP AC
:OUTP:ECLT0 ON
```

If the ":"(which is following the ";" and is in front of OUTP) is omitted, the TIA assumes that the second command is ":INP:OUTP:ECLT0 ON" and generates a syntax error.

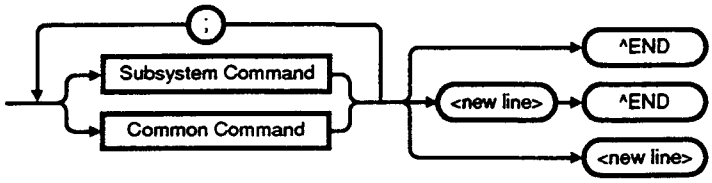
Using Multiple Commands

Program Messages

Program Messages are a combination of one or more properly formatted SCPI Commands. Program messages always go from a computer to the TIA. They are sent to the TIA as a sequence of ASCII data messages.

Program Message Syntax

Figure 2-3 shows the simplified syntax of a program message. You can send Common Commands and Subsystem Commands in the same program message. If you send more than one command in one message, you must separate adjacent commands with a semicolon.



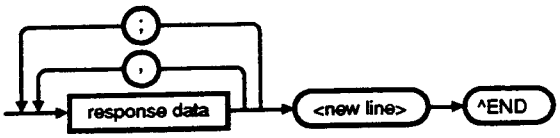
NOTE:
<new line> = ASCII character decimal 10
^END = EOI asserted concurrent with last byte

Figure 2-3. Simplified Program Message Syntax Diagram

When using IEEE 488.2 Common commands with SCPI Subsystem commands on the same line, use a semicolon between adjacent commands. For example:

```
*RST;:INP:COUP AC
```

When multiple subsystem commands are sent in one program message, the first command is always referenced to the root node. Subsequent commands, separated by ";", are referenced to the same level as the preceding command if no ":" is present immediately after the command separator (the semicolon).



NOTE:
<new line> = ASCII character decimal 10
^END = EOI asserted concurrent with last byte
; = multiple response separator
, = data separator within a response (ASCII character decimal 44)

Figure 2-4. Simplified Response Message Syntax Diagram

Overview of Response Message Formats

Response Messages

Response messages are data sent from the TIA to a computer in response to a query. (A query is a command followed by a question mark. Queries are used to find out how the TIA is currently configured and to transfer data from the TIA to the computer.)

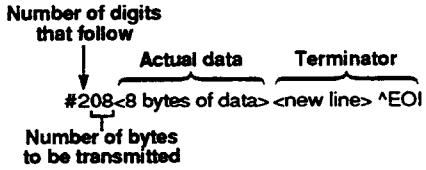
After receiving a query, the TIA interrogates the requested configuration and places the response in its output queue. The output message remains in the queue until it is read or another command is issued. When read, the message is transmitted across the interface to the computer. You read the message by using some type of enter statement that includes the device address and an appropriate variable. Use a print or display statement to display the message. The following HP BASIC example illustrates how to query the TIA and display the message:

```
10 OUTPUT 70906;":INP:COUP?"
20 ENTER 70906; A$
30 PRINT A$
40 END
```

Response Message Syntax

Figure 2-4 shows the simplified syntax of a Response Message. Response messages may contain both commas and semicolon separators. When a single query command returns multiple values, a comma is used to separate each item. When multiple queries are sent in the same program message, the groups of data corresponding to each query are separated by a semicolon. Note that a <new line> ^END is always sent as a response message terminator.

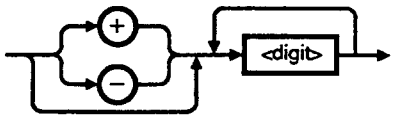
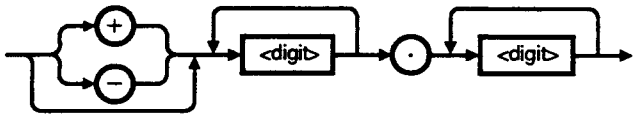
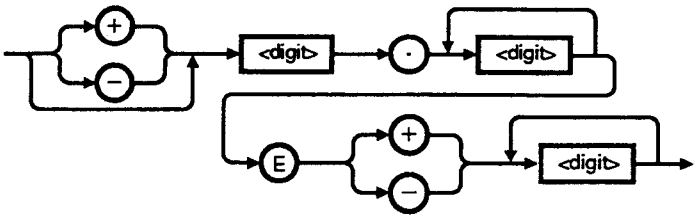
Table 2-4. Response Message Data Types (Continued)

Type	Description
Not a Number	Not a Number is represented by the value 9.91E37. 9.91E37 is defined by IEEE 754 to represent "Not a Number." The TIA responds with this value when queried for a floating point number it cannot provide.
<Boolean>	A single ASCII-encoded byte, 0 or 1, is returned for the query of settings that use <Boolean> parameters.
<literal>	ASCII-encoded bytes corresponding to the short form of the literal used as the command parameter. For example, if the :ROSC:SOUR INTernal command is sent to the TIA, the :ROSC:SOUR? response would be INT.
<string>	A string response consists of ASCII characters enclosed by double quotes. For example, string data is used for the "<error description>" portion of :SYST:ERR? response and for [:SENS]:FUNC? response.
<definite length block>	The syntax is a pound sign (#) followed by a non-zero digit representing the number of digits in the subsequent decimal integer. The decimal integer specifies the number of 8-bit data bytes being sent. This is followed by the actual data. The terminator is a line feed asserted with EOI. For example, for transmitting 8 bytes of data, the syntax might be:  The "2" indicates the number of digits that follow and the two digits "08" indicate the number of *data* bytes to be transmitted. A zero-length block has the syntax: #0<new line>^EOI

Response Message Data Types

Table 2-4 contains explanations of response data types.

Table 2-4. Response Message Data Types

Type	Description
<NR1>	This numeric representation has an implicit radix point.  The maximum number of characters in an <NR1> response is 22 (maximum 15 mantissa digits, 2 signs, 1 decimal point, 1 'E' character, 3 exponent digits).
<NR2>	This numeric representation has an explicit radix point.  The maximum number of characters in an <NR2> response is 22 (maximum 15 mantissa digits, 2 signs, 1 decimal point, 1 'E' character, 3 exponent digits).
<NR3>	This numeric representation has an explicit radix point and an exponent.  The maximum number of characters in an <NR3> response is 22 (maximum 15 mantissa digits, 2 signs, 1 decimal point, 1 'E' character, 3 exponent digits).

Some SCPI Syntax Conventions

[ ]	An element inside brackets is optional.
1 | 2	Means use either 1 or 2.
<numeric_value>	Means enter a number.
SENSe	Means you MUST use either all the upper case letters or the entire word. The lower case letters are optional. For example, SENS and SENSE are both valid. However, SEN is not valid. (Note SENSe is used here as an example, but this convention is true for all SCPI commands.) The actual letters used may be upper or lower cause.

NOTE

When you see quotation marks in the command's parameter (shown in the "Parameter Form" column in Table 13-1), you must send the quotation marks with the command. Refer to the section titled "Using HP BASIC" in Chapter 15 of this guide for details on how to use double quotes or single quotes to enclose the string parameter of a command.

Overview of the Measurement Instruction Commands

The Measurement Instruction commands includes the following SCPI commands:

- :MEASure?
- :CONFigure
- :READ?
- :FETCh?

Each of these commands are described in this chapter. **The detailed information for each of these commands is provided in the "Measurement Instructions" section of Chapter 13.**

The easiest way to have the TIA make a measurement is by using the :MEASure command. However, this command does not offer much flexibility. When you execute the command, the TIA uses the current settings of the selected function and immediately performs the measurement. The results are sent to the output buffer.

Sending the :MEASure? command is the same as sending a :CONFigure command followed immediately by a :READ? command.

One disadvantage of :MEASure? is that it keeps control of the bus until the acquisition is completed. The use of :CONFigure, :INITiate, and :FETCh? provides the possibility of including an interrupt routine after the :INITiate command that can notify the bus after the acquisition terminates. The bus is free to service other requests during the acquisition.

Descriptions of the Measurement Instruction Commands Parameters

The following is an example :MEASure? command with the possible parameters:

:MEASure:XTIMe:TINTerval? [<start_meas #>[,<# of meas>[,<source_list>]]]

3

Using the Measurement Instruction Commands

How to Use the :MEASure Command

There are three types of measurements that can be made with the :MEASure command:

- Time Interval
- Timestamp
- Histogram

Refer to Chapter 4, "Selecting the Time Interval Measurement," for a description of how to make each type of measurement mentioned above using the [:SENSe]:FUNCtion subsystem commands.

To make single-source time interval measurements,

Send :MEASure:XTIMe:TINTerval? 0,100,(@1)

This command line specifies that time interval measurements be made on a signal at Input 1. The measurements will be acquired and recorded in a sequence. This contrasts with the Histogram function where measurements are sorted into histogram bins as they are acquired, and the sequence of measurements is lost. One hundred measurements will be transferred from the TIA's memory to the output buffer. Use an ENTER statement in an HP BASIC program, or some equivalent command in another language, to send the data to a controller.

To make dual-source time interval measurements,

Send :MEASure:XTIMe:TINTerval? 0,100,(@1),(@2)

This command will cause the TIA to make time interval measurements between the signals at Inputs 1 and 2. The first 100 measurements will be sent to the output buffer.

To make timestamp measurements,

Send :MEASure:XTIMe:TSTamp? 0,400,(@1),(@2)

This command will cause the TIA to record the time of occurrence of the signal edges on Inputs 1 and 2. Each "timestamp" represents the time at which the signal edge occurred on Input 1 or 2. It represents the elapsed time from the beginning of the acquisition. The difference between each pair of "timestamps" represents the time interval. Use this command when you want access to this unprocessed time data.

The <start_meas #> Parameter

Use this parameter to specify the place (the sequential location) in the measurement results from which to begin retrieving data. This parameter exists because of the very large number of time interval measurements that can be acquired. Up to 512K single-channel measurements can be stored in the memory of the TIA. This parameter complements <# of meas> parameter, described below. Specifying "DEF" will start data retrieval at the measurement.

NOTE

This parameter will not be used very often with the :MEASure? command. This is because each time the command is executed, a new acquisition takes place. The parameter is more useful with the :FETCh? command (described later in this chapter) where a large number of measurements can be acquired and then transferred in segments into arrays that you create.

The <# of meas> Parameter

Use this parameter to specify the number of measurements to retrieve. This parameter complements <start_meas #> parameter described above. Specifying "DEF" will retrieve 2048 measurements.

NOTE

This parameter will not be used very often with the :MEASure? command. This is because each time the command is executed, a new acquisition takes place. The parameter is more useful with the :FETCh? command (described later in this chapter) where a large number of measurements can be acquired and then transferred in segments into arrays that you create. For large acquisitions, it is best to use FETCh? for data retrieval. This way, all of the data from the same acquistion can be retrieved in segments.

The <source_list>

Use “(@1)” for a single-source measurement on Input 1.

Use “(@2)” for a single-source measurement on Input 2.

Use “(@1),(@2)” for a dual-source measurement between Inputs 1 and 2.

See the descriptions of the parameters in the introductory section titled "Overview of the Measurement Instruction Commands" (starting at page 3-2) for descriptions of the parameters.

The following example describes a time interval measurement on Input 1. Several of the default settings are changed before the acquisition begins.

*RST	*Reset the TIA to a known state.*
:CONF:XTIM:TINT DEF,DEF,(@1)	*Select time interval on Input 1.*
:TRIG:SOUR EXT	*Select an external trigger.*
:TRIG:LEV 1.5	*Set trigger level voltage to 1.5V.*
:READ? 0, 100	*Initiate the acquisition and place the first 100 results in the output buffer.*

OR THE NEXT TWO COMMANDS CAN TAKE THE PLACE OF THE READ? COMMAND.

:INIT[:IMM]	*Initiate the acquisition.*
:FETC:XTIM:TINT? 0,100	*Retrieve the measurement results from memory and place it in the output buffer.*

Use an ENTER statement in an HP BASIC program, or some equivalent command in another language, to send the data to a controller.

How to Use the :READ? Command

The :READ? command initiates a measurement and puts the results in the output buffer.

Sending the :READ? command is like sending the :INITiate command followed immediately by the :FETCh? command.

NOTE

Be aware that you need to monitor for time overflows which can occur. Any input channel offsets must also be factored into the measurement computation.

NOTE

Since it is possible to produce time interval measurements from the timestamp data, you can make TSTamp measurements but retrieve time interval measurements using the following :FETCh query:

```
:FETC:XTIM:TINT?
```

To make histogram measurements,

Send `:MEASure:XTINterval:HISTogram? 0,2048,(@1)`

This command line specifies that time interval measurements be acquired into a histogram. These are single-channel measurements, such as would be appropriate for a data-to-data signal measurement. The number of measurements in each of 2048 histogram bins will be returned.

How to Use :CONFigure Command

For more programming flexibility, use the :CONFigure command. When you execute this command, the TIA is configured for the selected function. However, the acquisition is not automatically started, and you can change measurement parameters before starting the acquisition. The TIA offers a variety of low-level commands in the :INPut, [:SENSe], :CALCulate, and :TRIGger subsystems. These commands are documented in the following chapters. (You can also use the [:SENSe]:FUNCtion command to change the measurement function without using :MEASure or :CONFigure.)

Use this command when most of the default settings are acceptable for your measurement. The INPut, TRIGger, and SENSe subsystem commands should be used to set particular parameters.

Use the INITiate or READ? command to initiate the measurement.

```
:CONFigure:XTIMe:TINTerval [<start_meas#>[,< meas#>[,<source_list>]]]
```

How to Use the :INITiate and :FETCh? Commands

The :INITiate and :FETCh? commands provide the lowest level of control (with the most flexibility) of measurement triggering and result retrieval. Use the :INITiate command to start a measurement after you have configured the TIA. Measurements will begin when the specified trigger conditions are satisfied after the :INITiate command is received. The results are placed in the TIA's internal memory and can be retrieved with a :FETCh? command.

Measurement Complete Flag

After you execute an :INITiate command, you can monitor bit 4 of the Operation Status Register for completion of the acquisition by sending STAT:OPER:COND?. It is possible to insert an interrupt routine after the :INITiate command that can use the SRQ (service request) bit of the status register to inform the bus when the acquisition is completed. The bus can then service other requests during the acquisition.

Use the :FETCh? command to transfer the readings from the TIA's internal memory to the TIA's output buffer where you can read them into your bus controller.

Chapter Contents

This chapter describes the types of time interval measurements the TIA can make. **The detailed information for each of these measurements is provided in the "[:SENSe] Subsystem" section of Chapter 13.**

4

Selecting the Time Interval Measurement

To Select a Single-Source Time Interval < 12.5 ns

When you want to measure edges occurring faster than an 80 MHz rate (12.5 ns period), not every edge can be captured, but there is a special measurement mode that allows single-channel intervals to be measured down to 8 ns. The tradeoff with this mode is that following the short-interval measurement, there will be about 30 ns required before the next edge can be captured. The capture pattern will always be such that the time between the first two edges can be < 12.5 ns, then a long interval will follow, then a short one, and so on.

Send the following to select single-source time interval < 12.5 ns measurements:

```
[:SENSe]:"XTIMe:TINTerval 1,2"

[:SENSe]:TSTamp:NDELay
```

You may want to use random pacing with this measurement mode. This provides a way to measure consecutive edges on a single-channel input down to 8 ns, and (this is the random pacing part) ensures that all intervals in the distribution (e.g., a random data pattern) are measured in correct proportion.

Time Interval Measurements

Time interval measurements can be made on Input 1, Input 2, or from Input 1 to Input 2.

To Select Single-Source Sequential Time Interval Measurements

For single-channel measurements, consecutive edges can be recorded into memory as long as they are 12.5 ns or more apart. If the interval you are trying to measure is < 12.5 ns, you may miss an edge and then record the following one.

A special recording mode exists that allows pairs of edges down to 8 ns to be measurable on a single channel.

Figure 4-1 shows a single-source time interval measurement.

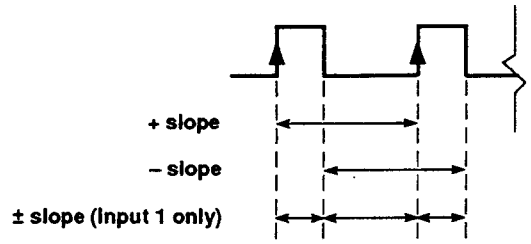


Figure 4-1. Single-Source Time Interval Measurement

Send the following to select single-source time interval measurements:

```
[:SENSe]:FUNCtion "XTIMe:TINTerval 1"
```

- Use this for data-to-data measurements when you want to record the sequence in which the measurements were acquired. This contrasts with the histogram modes where the measurements are directly sorted into histogram bins, thus the order in which the measurements were acquired is lost.

To Select Single-Source Histogram Measurements

Single-source time intervals are collected and sorted into a histogram. The histogram consists of 2,048 bins each with a width determined by the resolution setting (see Chapter 7). The histogram data results are the number of measurements that occur in each bin.

Send the following to select single-source histogram measurements:

```
[:SENSe]:FUNCtion "XTINterval:HISTogram 1"
```

- The time interval measurements are acquired into a histogram. No measurement sequence data is available, but measurement intervals are histogrammed at the incoming data rate.
- Frequency results are not available from histogram results.

To Select Dual-Source Histogram

Dual-source time intervals from Input 1 to Input 2 are collected and sorted into a histogram. The histogram data is the number of measurements sorted into each of the 2,048 bins.

Send the following to select dual-source histogram measurements:

```
[:SENSe]:FUNCtion "XTINterval:HISTogram 1,2"
```

- This mode sorts the interval measurement data directly into a histogram. Use this mode when you want to make histogram measurements as quickly as possible. Histogram measurement modes also support window margin computation.
- Frequency results are not available from histogram results.

To Select Single-Source Timestamps

In this mode, the TIA will acquire timestamps that are the time of occurrence of events on a single input. An event is an edge of the input signal where the selected edge and voltage level are controlled with other commands. Consecutive edges will be timestamped referenced to the elapsed time since the first edge. The first edge can be the trigger edge or the first edge of the input signal being measured.

To Select Dual-Source Sequential Time Interval Measurements

Time interval measurements from Input 1 to Input 2 can be made up to a 40 MHz rate. Intervals down to 0 seconds can be measured. See the diagram below.

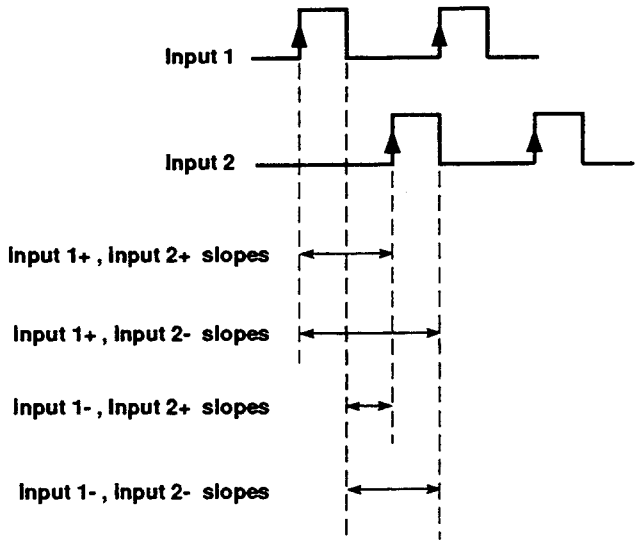


Figure 4-2. Dual-Source Time Interval Measurement

Send the following to select dual-source time interval measurements:

```
[:SENSe]:FUNCtion "XTIMe:TINTerval 1,2"
```

- Use this for data-to-clock or any two-channel measurements when you want to acquire data into sequential memory to preserve the time-order in which the measurements were acquired.
- A typical connection for this type of measurement would be to connect the data signal to Input 1 and the clock signal to Input 2.

Time interval results can be produced from this data. It will be necessary to correct the timestamp data for any time overflow that occurs.

Send the following to select single-source timestamps measurements:

```
[:SENSe]:FUNCtion "XTIMe:TSTamp 1"
```

To Select Dual-Source Timestamps

The TIA will acquire timestamps that are the time of occurrence of adjacent events on Inputs 1 and 2. An event is an edge of the input signal. Pairs of adjacent edges will be timestamped referenced to the elapsed time since the first edge on Input 1 occurred. The sequence of timestamps will always be: Input 1, Input 2, Input 1, Input 2, ...

Send the following to select dual-source timestamps measurements:

```
[:SENSe]:FUNCtion "XTIMe:TSTamp 1, 2"
```

As with single-channel timestamp measurements, to convert the timestamps to intervals will require overflow correction, scaling by the resolution, and (only for dual-channel measurements) adjustment for channels 1 and 2 path length differences.

To Select Frequency Measurements

Frequency can be derived from single-channel time interval measurements. It can only be retrieved once you have made a single-source time interval measurement with the appropriate arming conditions.

Send the following to convert time interval results to frequency:

```
:FETCh:XTIMe:FREQuency?
```

The pacing conditions that do not support frequency are:

- Random pacing.
- Any pacing that causes signal edges to be missed.

Chapter Contents

This chapter describes how to set the TIA to collect your data of interest. It includes the following:

The detailed information for each of the trigger conditions is provided in the ":TRIGger Subsystem" section of Chapter 13, and detailed information for setting measurement pacing is provided in the "[:SENSe] Subsystem" section of Chapter 13.

5

Specifying How the Data will be Acquired

Delaying From the Trigger

Specify how long to delay after the trigger until ready to acquire first data.

To Select Minimum Delay From the Trigger

Send :TRIGger:DELay:SOURce IMMediate

Provides for minimum delay after the trigger edge.

To Select Delay From the Trigger By Time

Send :TRIGger:DELay:SOURce TIMer

:TRIGger:DELay:TIMer <numeric_value>

The <numeric_value> identifies the delay time. Refer to ":TRIGger:DELay:TIMer" in Chapter 13 for range values.

To Delay From the Trigger By Number of Edges

Send :TRIGger:DELay:SOURce INPut

:TRIGger:DELay:ECOunt <numeric_value>

The <numeric_value> identifies the number of edges to delay. The conditions for counting edges are controlled with the [:SENS]:EVENt commands. Refer to ":TRIGger:DELay:ECOunt" in Chapter 13 for range values.

Specifying the Trigger

To Select Free Run (No Trigger)

The TIA will start measuring as soon as it is ready.

Send :TRIGger:SOURce IMMediate

To Specify Trigger

Specify the start of an acquisition.

A trigger can be an edge applied to the **Trigger Input** on the TIA faceplate (front panel) or an edge from either of the VXIbus ECL trigger lines.

To select the signal connected to **Trigger Input** on the front panel as the trigger source,

Send :TRIGger:SOURce EXTernal

To select the ECL Trigger Line from the VXIbus as the trigger source,

Send :TRIGger:SOURce ECLTrg0|ECLTrg1

To Select Start Trigger Level and Slope

To specify the level at which a trigger will be accepted at the TIA's front-panel **Trigger Input**,

Send :TRIGger:LEVel <GND|ECL|ECL10|TTL|TTL10> or <numeric_ value>

A specified numeric value will correspond to only one of the five discrete values: GND, ECL, ECL10, TTL, and TTL10.

To specify the slope that will be used to identify the trigger (applies to EXT, ECLTRG0, ECLTRG1).

Send :TRIGger:SLOPe POSitive|NEGative

To Measure After Counting a Specified Number of Edges

Send [:SENSe]:ACQuisition:PACing[:SOURce] STEP

[:SENSe]:ACQuisition:PACing:STEP <numeric_value>

The <numeric_value> specifies how many edges to count until the next edge is measured. Refer to "[:SENSe]:ACQuisition:PACing:STEP" in Chapter 13 for range values.

Figures 5-3 and 5-4 illlustrate measuring after counting a specified number of edges for single- and dual-source time interval measurements, respectively.

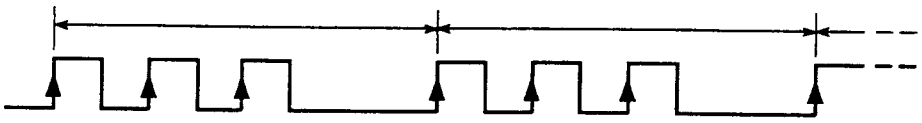


Figure 5-3. Every Nth Edge Pacing (N=3) Measurement for Single-Source Time Interval (Interval >12.5 ns)

In Figure 5-4, a minimum of 30 ns is needed after the end of one measurement before another measurement can begin.

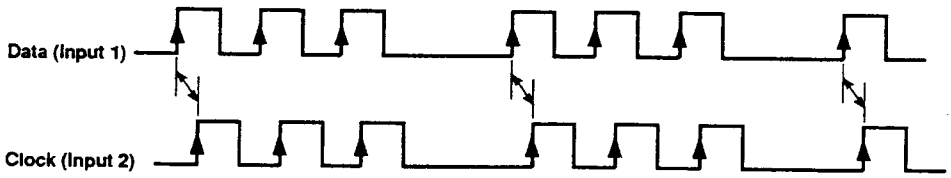


Figure 5-4. Every Nth Edge Pacing (N=3 edges of Input 1) Measurement for Dual-Source Time Interval.

To Measure After Counting a Random Number of Edges

Send [:SENSe]:ACQuisition:PACing[:SOURce] RANDom

Acquires a measurement after counting a random number of edges between measurements.

Setting Measurement Pacing

Measurement pacing allows you to specify how often to acquire data during an acquisition. You can specify to measure on **every edge**, **every Nth edge**, or on **random edges** of the input signal.

NOTE

Examples in Figures 5-1 through 5-6 show the trigger being kept as the *rising* edge.

To Measure as Fast as Possible

Send [:SENSe]:ACQuisition:PACing[:SOURce] IMMediate

Figure 5-1 and 5-2 illustrate measuring on every-edge for single- and dual-source time interval measurements, respectively.

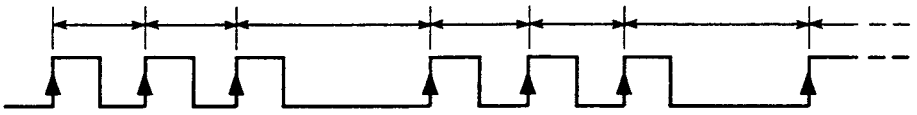


Figure 5-1. Every Edge Pacing Measurement for Single-Source Time Interval (Interval >12.5 ns)

In Figure 5-2, a minimum of 30 ns is needed after the end of one measurement before another measurement can begin. The measured interval can be as short as 0 ns.

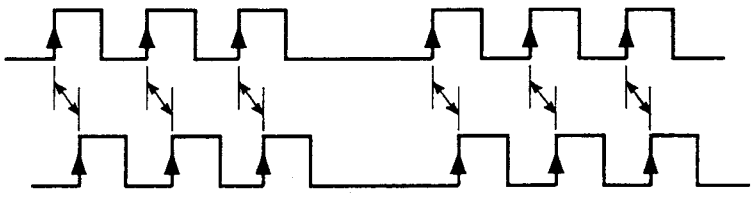


Figure 5-2. Every Edge Pacing Measurement for Dual-Source Time Interval

Specifying the Length and Number of Acquisitions

To Specify the Length of Acquisitions

By Time

Send [:SENSe]:ACQuisition:SOURce TIMer

[:SENSe]:ACQuisition:TIMer <numeric_value>

The <numeric_value> identifies the duration of the acquisition. Refer to "[:SENSe]:ACQuisition:TIMer" in Chapter 13 for range values.

By Number of Measurements

Send [:SENSe]:ACQuisition:SOURce MCOunt

[:SENSe]:ACQuisition:MCOunt <numeric_value>

The <numeric_value> identifies how many measurements will be in each acquisition. Refer to "[:SENSe]:ACQuisition:MCOunt" in Chapter 13 for range values.

To Specify the Number of Acquisitions

When you specify the number of acquisitions, you are specifying the number of times to repeat the acquisition.

Send :TRIGger:COUNt <numeric_value>

The <numeric_value> identifies the number of acquisitions. Refer to ":TRIGger:COUNt" in Chapter 13 for range values.

The time from the end of an acquisition until the TIA is ready to begin the next one is <500 ns.

Figures 5-5 and 5-6 illlustrate measuring after counting a random number of edges for single- and dual-source time interval measurements, respectively.

In Figure 5-5, *note that Input 1 can measure intervals as short as 8 ns non-continuously. A minimum of 30 ns is needed following every even-numbered edge until the next edge can be captured *and* a random number of edges of Input 1 will be counted before another measurement can begin.

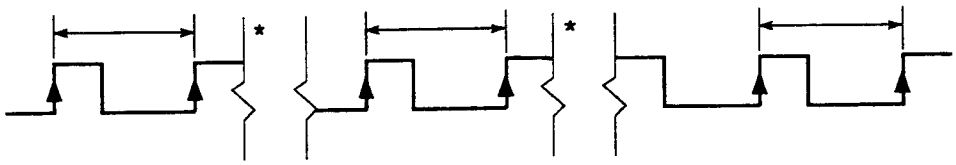


Figure 5-5. Random Edges Measurement for Single-Source Time Interval (Interval <12.5 ns)

In Figure 5-6, *note that a minimum of 30 ns is needed from the end of the prior measurement *and* a random number of edges of Input 1 will be counted before another measurement can begin.

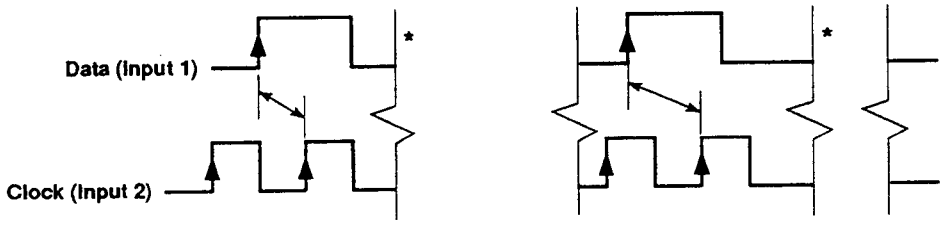


Figure 5-6. Random Edges Measurement for Dual-Source Time Interval

Using the Inhibit Input

Inhibit is a feature that can selectively suppress the data collection process. **An external signal is required at the TIA's** Inhibit Input connector. One of three voltage thresholds (ECL, GND, or TTL) can be specified, and the enabling condition can be set as above or below any of the three thresholds.

Using Inhibit While Making Single-Source Measurements

For **data-edge** to **data-edge** intervals, when Inhibit is active, the measurement underway is extended producing a measured interval at least as long as the time period of the "inhibit active." Any extended or "inhibited" intervals are not included in histogram calculations.

Using Inhibit While Making Dual-Source Measurements (From Input 1 to Input 2)

For **data-edge** to **clock-edge** intervals, when Inhibit is active, the measurement underway is not recorded, and any partial data is discarded. Partial data can occur when once Inhibit is inactive, an edge occurs first on Input 2. Discarding the partial data ensures that measurements are always made from Input 1 to 2.

Controlling the Inhibit Input Data Within an Acquisition.

To Disable / Enable Inhibit

Send [:SENSe]:INHibit[:STATe] ON | OFF

The shortest command to enable inhibit is :INH ON.

To Inhibit When Inhibit Input is Above or Below a Specified Level

Send [:SENSe]:INHibit:ACTive HIGH | LOW

To Select the Inhibit Level

Send [:SENSe]:INHibit:LEVel ECL | GND | TTL

6

Configuring Channel 1 and 2 Inputs

Selecting Channels 1 and 2 Input Conditioning

This chapter lists the steps you may need to perform or consider when setting up the TIA input conditioning.

Detailed information for each of the input conditioning selections is provided in the sections titled ":INPut[1|2] Subsystem" and "[:SENSe] Subsystem" in Chapter 13.

To Select Input Coupling

Send :INPut[1|2]:COUPling AC|DC

To Select Input Impedance

Send :INPut[1|2]:IMPedance 50 | 1.0E+6 [OHM]

To Select Input Threshold Level

Send [:SENSe]:EVENt[1|2]:LEVel <numeric_value> [v]

The <numeric_value> identifies the threshold voltage level. Refer to "[:SENSe]:EVENt[1|2]:LEVel" in Chapter 13 for range values.

To Select Input Event Slope

Send [:SENSe]:EVENt[1|2]:SLOPe POSitive|NEGative|EITHer

To Select Input Event Hysteresis (Sensitivity)

Send [:SENSe]:EVENt[1|2]:HYSTeresis:RELative <numeric_value> [PCT]

The <numeric_value> identifies the relative hysteresis level. Refer to "[:SENSe]:EVENt[1|2]:HYSTeresis:RELative" in Chapter 13 for range values.

To Select Input Route (Common/Separate)

Send :INPut[1|2]:ROUTe COMMon|SEParate

7

Setting the Span and Resolution

Chapter Contents

This chapter describes the tradeoffs when selecting span and resolution. The following topics are included:

Detailed information for setting the range and resolution for time interval sequential measurements is provided in the "[:SENSe]:TINTerval:RANGe" section of Chapter 13. Detailed information for setting the span and resolution for histogram is provided in the "[:SENSe]:HISTogram:RANGe" section of Chapter 13.

Setting Time Interval Range and Resolution for Sequential Measurements

The TIA has a control for selecting resolution based on the maximum time interval you need to measure. This allows you to optimize the resolution of your measurement results. The default setting provides the best resolution. It specifies a maximum time interval of 3.2 µs that can be measured with a resolution of approximately 50 ps.

The range and resolution can be set in power-of-two steps. The maximum time interval that can be measured is 26.2 ms with a corresponding resolution of 400 ns.

Table of Range and Resolution

When measuring intervals over 3.2 µs, select the range that allows measurement of the largest expected time interval. The maximum time interval and the resolution for that range are coupled, so that selecting a time interval range automatically selects the corresponding resolution shown in Table 7-1.

NOTE

If intervals occur that are longer than the specified maximum time interval, they will be erroneously measured. What happens in such a case is the counter monitoring the interval overflows and an interval shorter than the actual one will be reported.

Table 7-1 shows the supported maximum time interval and resolution values for the TIA. Here is an example of how to use the table: If you know that the maximum time interval you will measure is less than 3.2 µs, use the default value of 3.2 µs and the results will be measured with a resolution of 50 ps. The resolution specification is the smallest difference in measurement values that the TIA can resolve.

NOTE

The actual instrument resolution will be $\frac{12.5\text{ ns}}{256} = 48.8828125$ ps.

For convenience let's call this resolution value 50 ps.

Table 7-1. Sequential Time Interval Range and Resolution

Maximum Time Interval	Resolution	Maximum Time Interval	Resolution
3.2 µs	50 ps	410 µs	6.25 ns
6.4 µs	100 ps	819 µs	12.5 ns
12.8 µs	200 ps	1.64 ms	25.0 ns
25.6 µs	390 ps	3.28 ms	50.0 ns
51.2 µs	780 ps	6.55 ms	100 ns
102 µs	1.56 ns	13.1 ms	200 ns
205 µs	3.13 ns	26.2 ms	400 ns

To Set Time Interval Range

The time interval range defines the maximum value that can be measured with a particular resolution.

Send [:SENSe]:TINTerval:RANGe[:UPPer] <numeric_value>

The <numeric_value> defines the maximum interval to be measured.

To Set Time Interval Resolution

The time interval resolution is the smallest difference in measurement values that can be resolved by the TIA.

Send [:SENSe]:TINTerval:RANGe:RESolution <numeric_value>

The <numeric_value> defines the resolution.

Setting Histogram Span and Resolution

The TIA has a control for selecting resolution based on the maximum histogram span that you need to measure. The resolution of the measurement results can be optimized based on the required range of time interval values to be measured.

Table of Span and Resolution

Table 7-2 shows the supported span for a histogram and the resolution available for that span. If you know that all the intervals to measure are 100 ns or less, you can take advantage of the best resolution of 48.8 ps. Now if the intervals are greater that 100 ns, but all the intervals occur within a 100 ns span, you can use an offset command to "slide" the 100 ns span to where the intervals are to be measured. For example, say you are measuring intervals of from 50 ns to 150 ns. This is still a span of 100 ns, but not 0 to 100 ns. To move the start of the span to 50 ns and still take advantage of the best resolution, use the following command:

```
[:SENS]:HIST:RANG:OFFS 50E-9      !This command moves the 100 ns
                                  span to: 50 ns to 150 ns.
```

The span and resolution values are coupled values. When setting span, its corresponding resolution is automatically selected.

Table 7-2. Histogram Span and Resolution

Span	Resolution	Span	Resolution
100 ns	50 ps	12.8 µs	6.25 ns
200 ns	100 ps	25.6 µs	12.5 ns
400 ns	200 ps	51.2 µs	25.0 ns
800 ns	390 ps	102 µs	50.0 ns
1.6 µs	780 ps	205 µs	100 ns
3.2 µs	1.56 ns	410 µs	200 ns
6.4 µs	3.13 ns	819 µs	400 ns

To Set Histogram Resolution

The histogram resolution is the width of each of the 2,048 bins in the histogram. Similar to the time interval resolution, it represents the smallest difference in measurement values that the TIA can resolve.

Send [:SENSe]:HISTogram:RANGe:RESolution <numeric_value>

To Set Histogram Offset

The histogram offset command allows you to "move" the range of the histogram to where the time interval values are expected.

Send [:SENSe]:HISTogram:RANGe:OFFSet <numeric_value>

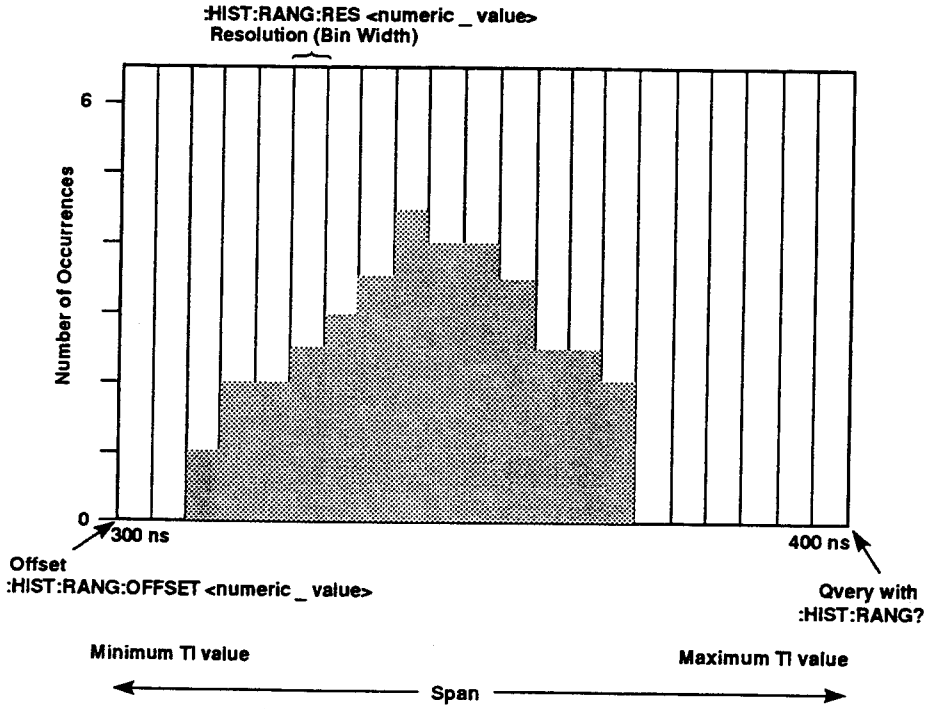


Figure 7-1. Illustration of Histogram Span and Offset

8

Initiating Measurements

Acquisition Controls

Controlling Acquisition

This chapter lists the steps you need to perform or consider when initiating measurements, and acquiring data.

Detailed information for aborting measurements in progress is provided in the ":ABORt Subsystem" section of Chapter 13. Detailed information for initiating and acquiring measurements is provided in the ":INITiate Subsystem" and ":FETCh Subsystem" sections of Chapter 13, respectively.

To Abort Measurements in Progress

The :ABORt subsystem command aborts any measurement in progress.

Send :ABORt

To Initiate Measurements

The :INITiate subsystem is used to control the start of a measurement acquisition.

Send :INITiate

Fetch?

The :FETCh subsystem commands retrieve the measurements acquired with the :INITiate commands and places them in the output buffer.

Send FETCh? [<start_meas#>[,<meas# >[,source_list]]]

Omitting the parameters will cause the TIA to retrieve all measurements.

Read more about the FETCh command in chapters 3 and 13.

9

Analyzing Data With Window Margin

Chapter Contents

This chapter provides a general discussion of window margin and lists some steps you may want to perform or consider when using window margin. The following is provided in this chapter:

Detailed information of all of the window margin commands is provided in the section titled ":CALCulate Subsystem" in Chapter 13.

What is Window Margin?

Window margin analysis is a technique to measure the intrinsic error rate of a disk drive readback process. The results indicate whether sufficient timing margin is available in the system to achieve the desired error rate goal. Window margin analysis and phase margin analysis were invented to overcome the practical limitation of actually waiting long enough for a drive to read 10^{11} or 10^{12} bits to establish a bit error rate of 10^{-10}. In addition, these techniques also give insight into the available margin in the design, rather than just pass/fail error rate information.

For both phase margin analysis and window margin analysis, a relatively small number of bits can be measured relative to the error rate of interest. By using techniques to accelerate error rates, the timing margin for an error to occur can be projected for low error rates. Phase margin analysis and window margin analysis *differ primarily in the way that error rates are accelerated*. Phase margin analyzers narrow the size of the decoding window to accelerate the occurrence of timing errors. This requires the phase margin analyzer to have complete control over the data separator/phase lock loop circuitry. Time Interval Analyzers compute window margin results from histograms of time interval measurements.

The HP E1740A use histograms of either data-to-data or data-to-clock intervals as the fundamental information to compute window margin. A data error effectively occurs if timing jitter or displacement of a data edge is great enough to cause the edge to be detected outside of its intended decoding window. The histograms effectively represent the timing displacement (for example, peak shift, timing asymmetry) and the variation (for example, read noise, write noise) of the data edges relative to the decoding window.

The difference between the measured mean value of a histogram distribution and its expected value based on the clock rate and the encoding scheme indicates the amount of timing offset or displacement. The standard deviation of the distribution is a measure of the aggregate timing noise or jitter in the system. Both of these timing effects combine to reduce the amount of timing margin in the system and increase the probability that an error will occur.

A window margin plot, shown in Figure 9-1, is essentially a cumulative probability representation of the time interval histogram. The histogram is mathematically integrated from the outside edges of the decoding window towards the center. This integration is accomplished by successively summing the histogram bin counts and normalizing by the total number of measurements in the histogram distribution. This bin-by-bin integration is how the error rate is effectively accelerated to derive timing margin for error rates corresponding to the number of intervals actually measured. To produce margin information down to a part in 10E10 without having to measure 10E10 bits, the plot is mathematically extrapolated.

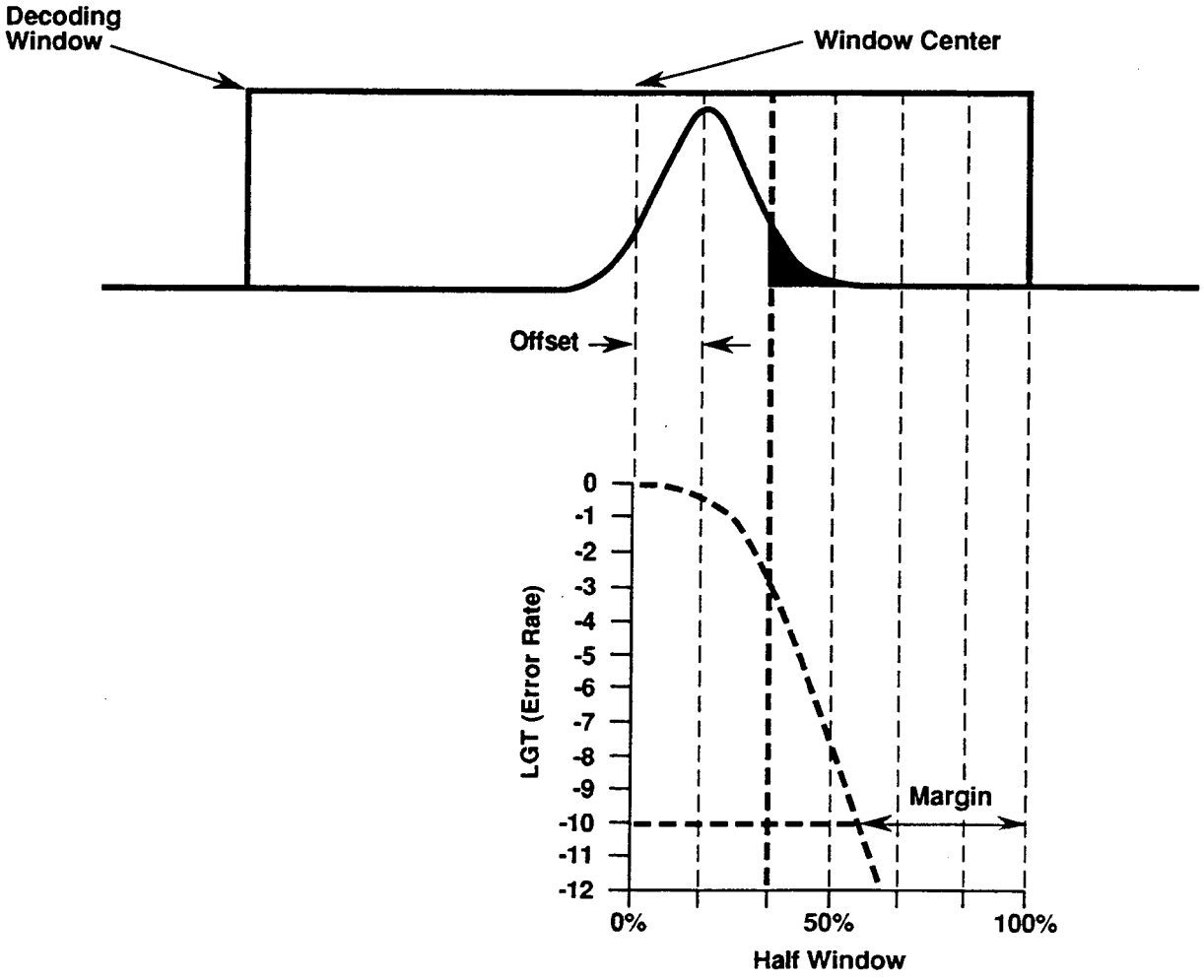


Figure 9-1. A Window Margin Plot

Both the early and late sides of the resulting cumulative probability density curves are extrapolated to the error rate of interest. Mathematically, the function is actually erfc^{-1}.

$$\text{error rate (w)} = k\int_{w=edge}^{\infty} e^{\frac{-(t-ofst)^2}{2\sigma 2}}\, dt$$

Recognizing that this function is linear in terms or error rate sigmas versus the decoding window, a least-squares linear curve fit can be made to the acquired data to extrapolate to the required error rate. The x-axis intercept is the timing offset in the system and the inverse slope is the aggregate noise in the system. The difference between the edge of the decoding window and the y-intercept of the fitted line at the desired error rate represents the timing margin. These data points are converted to an error rate versus decode window display. Be aware that this processing assumes that the original histograms follow a normal or Gaussian shape.

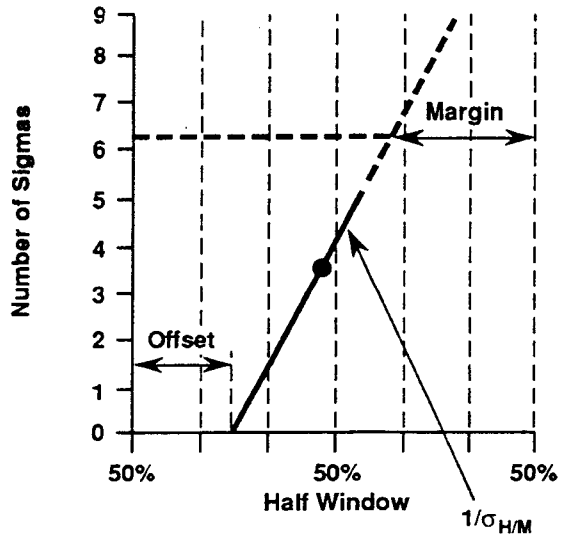


Figure 9-2. Extrapolation Based on Distribution Sigmas

The HP E1740A allows you to control the portion of the data used to extrapolate or calculate the curve fit by using the :CALCulate:WMARgin:EXTRapolation commands with the TIA configured for histogram time intervals (:CONFigure:XTINterval:HISTogram). This may be useful when you want to ignore certain data points that may skew the extrapolation. However, at least five data points (for each side) must fall within the extrapolation levels you specify.

A combined window margin result is computed by "folding" the time interval histogram about the defined decoding window center. Overlaid bins are summed, and the resulting one-sided histogram is integrated as described above to achieve the combined window margin result.

Using Window Margin

In the HP E1740A TIA, the :CALCulate subsystem is used for computation of window margin results, which are derived from acquired histogram measurements. Along with the window margin results, all of the parameters associated with defining how the window margin should be computed are controlled within the :CALCulate subsystem.

NOTE

Window margin results are only available when the configuration is histogram time intervals (CONFigure:XTINterval:HISTogram).

NOTE

See Chapter 13 for a more detailed description of the commands discussed in this chapter.

To Enable Automatic Window Margin Analysis

Send :CALCulate:WMARgin[:STATe] <Boolean>

This command accepts a Boolean which activates (or de-activates) automatic calculation of the entire window margin group of commands. The significance of this is that when the state is set to ON, window margin results will automatically be computed upon acquisition of new data, whereas when the state is OFF, window margin results are only computed when the result is requested.

To Select Code Spacings (Segments)

Segment controls provide the ability to define portions of the histogram time axis for individual segment processing.

1 **Specify the center of the first segment by sending:**

:CALCulate:WMARgin:SEGMents:CENTer <numeric_value>

This command defines the expected center (in time) of the first segment.

2 **Specify the number of segments by sending:**

:CALCulate:WMARgin:SEGMents:COUNt <numeric_value>

This command controls how many segments should be defined. Segments are connected end-to-end (in time) so that the end of a given

segment is in common with the start of the next segment. The first segment is always the one whose center is defined by the :CENTer command.

3 **Specify the segment width by sending:**

:CALCulate:WMARgin:SEGMents:WIDTh <numeric_value>

This command defines the time width of each segment. All segments have the same width. Typically, this width will be equal to the data decode window width, which is related to the clock from which it is derived.

4 **Select the segments to be included in window margin analysis by sending:**

:CALCulate:WMARgin:SEGMents:ENABle <non-decimal numeric> | <numeric_value> | ALL

This command provides a way to selectively include segments in window margin computations. The parameter for making the selection works much like selecting bits in the status byte: each consecutive segment has a bit assigned to it, with an associated value equal to the power-of-2 of it's associated. bit position.

To Query the Offset

Send :CALCulate:WMARgin:OFFSet?

This query returns the time difference between the center of the decoding window and the mean of the distribution. The result returned incorporates data from all enabled segments.

To Adjust Offset

Send :CALCulate:WMARgin:SEGMents:OFFSet <numeric_value>

This command provides a way to time-shift all of the defined segments by an equal amount. This can also be accomplished by changing the CENTer value, but the intent of OFFSet is to provide a way of tweaking the segment locations without having to move the CENTer value, which may be based on an ideal design value.

To Specify the Error Rate

Send :CALCulate:WMARgin:MLEVel <numeric_value>

Enter a log (error rate) value on which margin calculations are based. For example, if you wanted to determine available margin at 1 in 1E6, enter -6.

To Control Use of Extrapolated Data

When calculating window margin,

Send :CALCulate:WMARgin:EXTRapolation:STATe <Boolean>

When set to ON (default state) extrapolated data will be used, if necessary, in calculating window margin. If set to OFF, an attempt to read window margin will produce an error if no data is present at the margin level. This command does not prevent reading extrapolated data.

To Modify Extrapolation Range

You can specify the range of margin data to use when determining the extrapolation curve. This is valuable when you want to limit this range to the portion of the window margin curve that appears to be exhibiting the dominant noise process.

Define the Upper Boundary

Send :CALCulate:WMARgin:EXTRapolation:RANGe:UPPer <numeric_value>

Defines the upper boundary of the error rate range used as the basis for extrapolation. If -4.5 is entered, the upper data boundary is set to 10E-4.5.

Define the Lower Boundary

Send :CALCulate:WMARgin:EXTRapolation:RANGe:LOWer <numeric_value>

Defines the lower boundary of the error rate range used as the basis for extrapolation. This is the ending point of the extrapolation curve. If -12 is entered, the lower extrapolation boundary is set to 10E-12.

To Specify One-sided or Two-sided Window Margin

Send :CALCulate:WMARgin:SIDes ONE | TWO

Specifying "ONE" will fold early and late margins together. Selecting "TWO" will keep early and late margins separate.

Obtaining One-sided Window Margin Results

All the enabled distributions are folded about the window center to produce a single curve.

1 **To obtain one-sided window margin result,**

Send :CALCulate:WMARgin:MARGin?

The window margin result is based upon the setting of the error rate value. The result returned is the margin at the specified error rate.

Use this command to retrieve the margin of one-sided results.

2 **To obtain one-sided window margin noise result,**

Send :CALCulate:WMARgin:NOISe?

This command retrieves the aggregate timing "noise" of random bit shift effects of a one-sided window margin.

3 **To obtain one-sided window margin data,**

Send :CALCulate:WMARgin:DATA?

Use this command to retrieve the window margin data of one-sided results. See Table 10-1 for information on the format of the window margin data.

4 **To obtain the time reference for window margin data,**

Send :CALCulate:WMARgin:DATA:REFerence?

This query returns a reference value for the first value returned by the DATA query.

5 **To obtain the time reference of the extrapolated curve,**

Send :CALCulate:WMARgin:EDATa?

Use this command to retrieve the extrapolated data for one-sided window margin results.

6 To obtain the starting point of the extrapolated curve,

Send :CALCulate:WMARgin:EDATa:REFerence?

Obtaining Two-sided Window Margin Results

1 To obtain two-sided window margin results,

a. Send the following command to retrieve the early margin of a two-sided window margin:

:CALCulate:WMARgin:MARGin:EARLy?

b. Send the following command to retrieve the late margin of a two-sided window margin:

:CALCulate:WMARgin:MARGin:LATE?

2 To obtain two-sided window margin noise results,

a. Send the following command to retrieve the early-side noise result of a two-sided window margin:

:CALCulate:WMARgin:NOISe:EARLy?

b. Send the following command to retrieve the late-side noise result of a two-sided window margin:

:CALCulate:WMARgin:NOISe:LATE?

3 To obtain two-sided window margin data,

a. Send the following command to retrieve the early-side window margin data of two-sided results:

:CALCulate:WMARgin:DATA:ESIDe?

b. Send the following command to retrieve the late-side window margin data of two-sided results:

:CALCulate:WMARgin:DATA:LSIDe?

4 To obtain the time reference for early-side window margin data,

Send :CALCulate:WMARgin:DATA:ESIDe:REFerence?

This query returns the starting point of the early-side curve with respect to the window center.

5 To obtain the time reference for late-side window margin data,

Send :CALCulate:WMARgin:DATA:LSIDe:REFerence?

Late-side data always starts at the window center (time zero). Therefore, this command always returns zero—the command is provided only for consistency and never needs to be used.

6 To obtain the two-sided extrapolated window margin data,

a. Send the following query to retrieve the extrapolated data for early-side window margin results:

:CALCulate:WMARgin:EDATa:ESIDe?

b. Send the following query to retrieve the extrapolated data for late-side window margin results.

:CALCulate:WMARgin:EDATa:LSIDe?

7 To obtain the time reference for early-side extrapolated data,

Send :CALCulate:WMARgin:EDATa:ESIDe:REFerence?

This query returns the starting point of the early-side extrapolated curve with respect to the window center.

8 To obtain time reference for late-side extrapolated data,

Send :CALCulate:WMARgin:EDATa:LSIDe:REFerence?

Late side data always starts at the window center (time zero). Therefore, this command always returns zero— the command is provided for consistency and never needs to be used.

10

Data Formats

Handling Data

Chapter Contents

This chapter describes how to retrieve acquired data and move it to a computer for further analysis. The following are included in this chapter:

Using the FORMat Subsystem to Specify Output Format

The :FORMat [:DATA] subsystem sets the data format for transferring numeric and array information.

The :FORMat[:DATA] <types> selects the output format for measurement or source data (that is, in response to :FETC?,:MEAS?, :READ, :CALC:WMARgin:DATA?, or :CALC:WMARgin:EDATa?). Valid <types> are ASCii, REAL, and INTeger.

The REAL type defaults to <float64> and the INTeger type defaults to <int16>. When ASCii formatting is selected the output stream data is buffered, whereas REAL formatting retrieves unbuffered data (yields faster throughput).

Thus, use FORMat[:DATA] ASCii | REAL | INTeger to set or query the data format type.

*RST state for :FORMat[:DATA] is ASCii.

:FORMat affects the response format of the following commands:

:CALCulate:WMARgin:DATA?
:CALCulate:WMARgin:DATA:ESIDe?
:CALCulate:WMARgin:DATA:LSIDe?
:CALCulate:WMARgin:EDATa?
:CALCulate:WMARgin:EDATa:ESIDe?
:CALCulate:WMARgin:EDATa:LSIDe?
:FETCh?
:MEASure?
:READ?

Refer to the following section for details on data format retrieval.

Description of the Types of Format for Various Data Retrieval

Table 10-1 shows how format selection affects each type of data retrieval for the different measurement-function configurations.

Each configuration listed in the table is discussed in detail on the following pages.

Table 10-1. Types of Format for Various Data Retrieval

CONFigure	Data Retrieval	Format Types		
		ASCii	REAL*	INTeger**
:XTIM:TST	FETCh? or FETC:XTIM:TST? FETC:XTIM:TINT?	timestamps <NR1> intervals <NR3>	——————— intervals <float64>	timestamps <int16> timestamp differences <int16>
:XTIM:TINT	FETCh? or FETC:XTIM:TINT? FETC:XTIN:HIST?	intervals <NR3> bin counts <NR1>	intervals <float64> ———	timestamps differences <int16> bin counts <int32>
:XTIN:HIST (:HIST:ACC OFF)	FETCh? or FETC:XTIN:HIST? FETC:XTIM:TINT?	bin counts <NR1> intervals <NR3>	——————— intervals <float64>	bin counts <int32> timestamp differences <int16>
:XTIN:HIST (:HIST:ACC ON)	FETCh? or FETC:XTIN:HIST? FETC:XTIM:TINT?	bin counts <NR1> intervals <NR3>	bin counts <float64> intervals <float64>	——————— timestamp differences <int16>
Window Margin Data (**Note:** Configuration must be XTIN:HIST)	CALC:WMAR:DATA? CALC:WMAR:DATA:ESID? CALC:WMAR:DATA:LSID? CALC:WMAR:EDAT? CALC:WMAR:EDAT:ESID? CALC:WMAR:EDAT:LSID?	<NR3> <NR3> <NR3> <NR3> <NR3> <NR3>	<float64> <float64> <float64> <float64> <float64> <float64>	——————— ——————— ——————— ——————— ——————— ———
Frequency Results	FETC:XTIM:FREQ?	frequencies <NR3>	frequencies <float64>	———

* In all cases, REAL results are returned as 8-byte (64-bit) floating point values.

** Where supported, histogram results are 4-byte (32-bit) integers. All other INT format results are 2-byte (16-bit) integers.

Configuration :CONF:XTIM:TST

This configuration directs the HP E1740A to capture the data as a sequence of timestamps. As with all configurations, retrieving the data in the same form as the configuration only requires a :FETCh?. In this case, :FETCh? and :FETCh:XTIM:TST? are equivalent.

The timestamp values are unprocessed numbers that represent the value of an internal counter at the time the data was latched. To convert timestamp data to time intervals requires the following steps (If you're only interested in intervals, use the :CONF:XTIM:TINT configuration or retrieve the data with :FETC:XTIM:TINT?):

1 **Subtract current timestamp from the next timestamp.**

2 **If the result from step 1 is negative, add 65536 to it. (A negative result indicates that the counter overflowed between the timestamps, so one counter cycle must be added.)**

You now have the interval in units of counter ticks.

3 **To convert to time, multiply by the resolution ([:SENS]:TINT:RANG:RES?).**

4 **If the measurement is two-channel (Input 1 to 2), each interval needs to be corrected for timing differences between the channels.**

The query CAL:INP:TIM? provides the channel difference (in units of time). For each interval from 1 to 2 (the 1st, 3rd, ... odd intervals), you should *subtract* the channel difference. For each interval from 2 to 1 (the 2nd, 4th, ... even intervals), you should *add* the channel difference. For [:SENS]:TST:NDEL ON a similar correction is needed. The correction value for this case is obtained from the CAL:INP:NDEL? query.

You can also have the timestamps converted to intervals by the TIA by using the :FETC:XTIM:TINT? query. For ASCii and REAL formats the results will be time intervals (in units of time). For INTeger format the results will be timestamp differences (in units of counter ticks).

Configuration :CONF:XTIM:TINT

This configuration directs the TIA to capture the data as a sequence of time intervals. As with all configurations, retrieving the data in the same form as the configuration only requires a :FETCh?. In this case, :FETCh? and :FETC:XTIM:TINT? are equivalent.

For ASCii and REAL formats, the results will be time intervals (in units of time) for INTeger format the results will be intervals in units of counter ticks.

You can also retrieve the data as a time interval histogram (:FETC:XTIN:HIST?). If your primary interest is histogram data, the :CONF:XTIN:HIST? configuration is recommended since it provides more control of the histogram results than configuring for sequential time interval and retrieving as histogram (that is, MCOunt limit for :CONF:XTIM:TINT is smaller than :CONF:XTIN:HIST).

Configuration :CONF:XTIN:HIST

These configurations direct the TIA to utilize its high-speed histogramming capability to directly record the intervals between events as number of occurrences at each interval value. As with all configurations, retrieving the data in the same form as the configuration only requires a :FETCh? . In this case, :FETCh? and :FETC:XTIN:HIST? are equivalent.

When histogram accumulation is disabled ([:SENS]:HIST:ACC OFF), which is the *RST state, the histogram data can be retrieved as in either ASCii or INTeger formats. If integer format is selected, the results will be 4-byte (32-bit) integers.

When histogram accumulation is enabled ([:SENS]:HIST:ACC ON), the histogram data can be retrieved in either ASCii or REAL formats. As with all outputs in REAL mode, the structure is 8-byte (64-bit) floating point.

To scale and reference the histogram data use the [:SENS]:HIST:RANG:RES? and [:SENS]:HIST:RANG:OFFS? commands. To compute the interval being represented by any specific histogram bin, use the expression:

[resolution * (bin number -1)] + reference,

where resolution is the value retrieved with the [:SENS]:HIST:RANG:RES? query, bin number is the bin of interest (first bin = 1), and reference is the value retrieved with the [:SENS]:HIST:RANG:OFFS? command.

NOTE

The exact resolution will be $\frac{12.5\ ns}{256} \times 2^n$

where n is determine by the program resolution. Exact resolution is 48.8828125 ps

You can also retrieve the data as sequential time interval by using :FETC:XTIM:TINT?. If your primary interest is sequential time interval data, the [:SENS]:XTIM:TINT configuration is recommended over the approach of configuring for histogram and retrieving as sequential time interval. The reason for this recommendation is that it's possible to get unexpected results when mixing configuration and retrieval modes. For example, if you've configured for a histogram measurement of 50 million data values, the sequential data will overflow (its limit is 1/2 million data values).

Window Margin

Each of the window margin data results have the same format capabilities as shown in the table. INTeger format does not make sense for this type of data; therefore, it is not supported.

Frequency Results

Sequential frequency results can be retrieved for single-channel measurements with the :FETC:XTIM:FREQ? query. Valid frequency results will not be obtained if the random pacing mode is selected ([:SENS]:ACQ:PAC RAND) or the every edge pacing mode is selected ([:SENS]:ACQ:PAC IMM) and the continuous counting rate is exceeded by the input. An example of this would be a 100 MHz input, which exceeds the 80 MHz every edge counting rate. The way to get valid frequency readings above 80 MHz is to use every Nth edge pacing ([:SENS]:ACQ:PAC STEP). Any value greater than 3 will ensure that every edge is accounted for.

11

Status Reporting

Chapter Contents

This chapter describes the status reporting structure for the TIA. The status reporting structure allows you to monitor specified events in the TIA (such as when an error occurs or a message is available).

The following information is provided in this chapter:

HP E1740A TIA Status Registers Structure

The HP E1740A status registers conform to the SCPI and IEEE 488.2 standards.

Figure 11-1 shows all the status system register groups and queues in the HP E1740A Time Interval Analyzer (TIA). This is a high level drawing that does not show all the registers that are contained in each group. It is intended as a guide to the bits used in each of these register groups to monitor the TIA's status. Note that besides the Operation Status and the Questionable Data/Signal Register groups, a summary of the Standard Status Structure Registers (defined by IEEE 488.2-1987) is shown.

Refer to the section in this chapter titled "How to Program the TIA for Status Reporting" and the flowchart in Figure 11-5 for detailed information on programming the status reporting system.

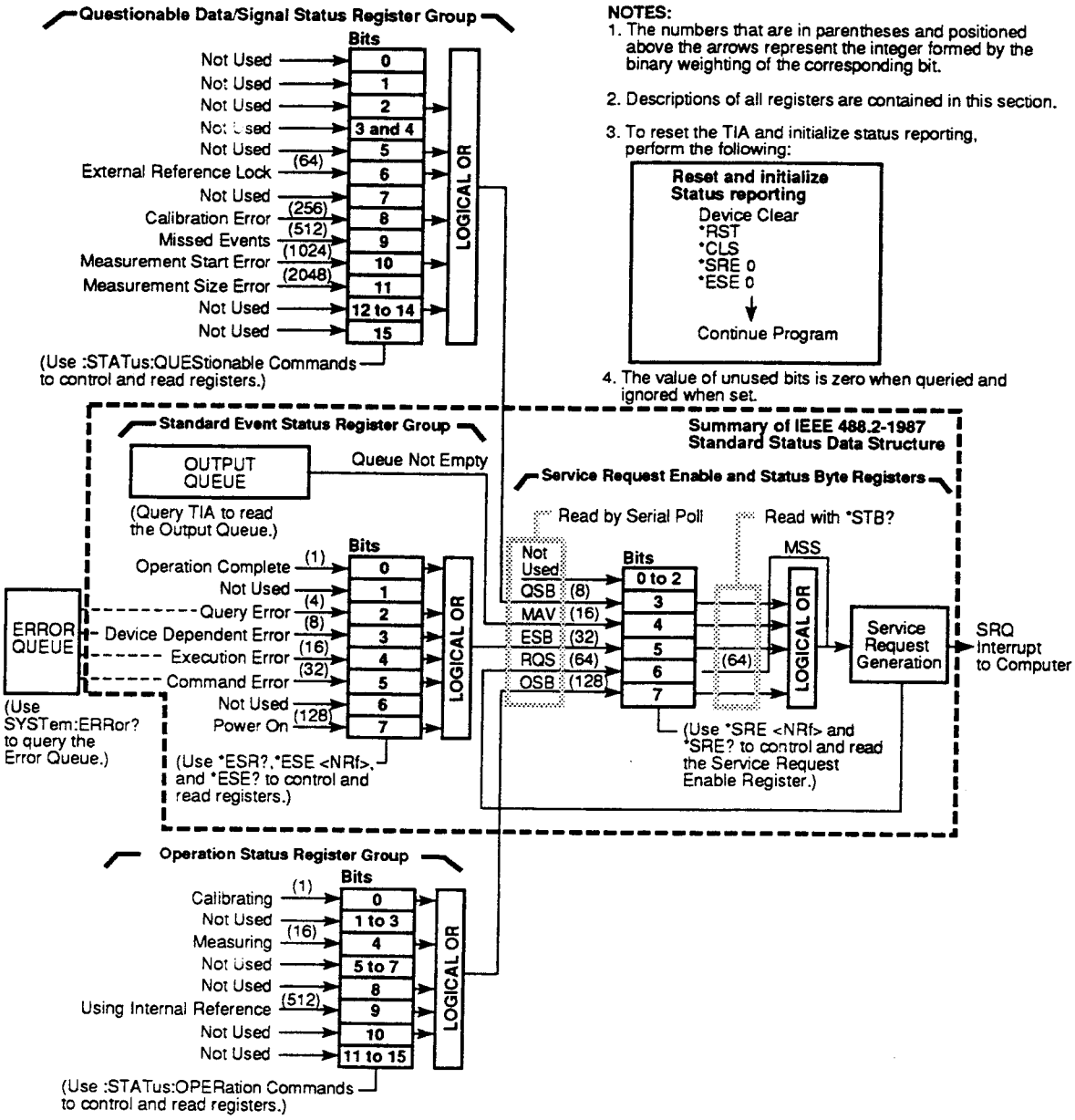


Figure 11-1. HP E1740A SCPI Status Reporting Summary Functional Diagram

Status Byte Register and Service Request Enable Register

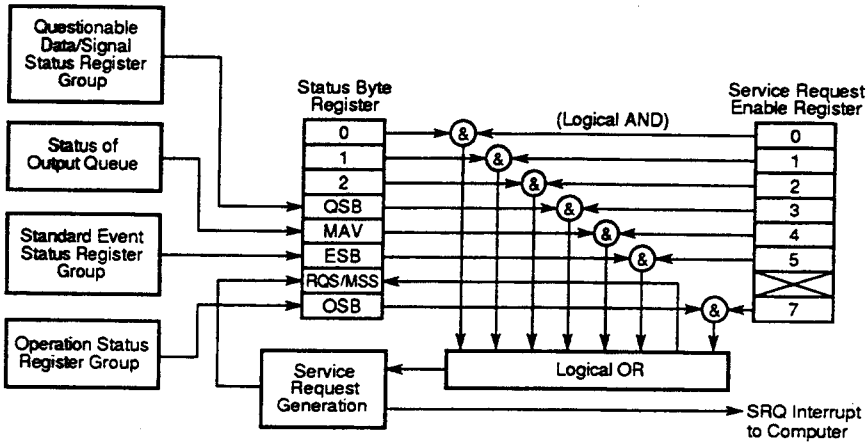


Figure 11-2 Status Byte and Service Request Enable

Status Byte Register

The Status Byte Register is the summary-level register in the status reporting structure. It contains summary bits that monitor activity in the other status registers and queues as shown in Figure 11-2. The Status Byte Register is a live register—its summary bits are set TRUE or FALSE (one or zero) by the presence or absence of the condition which is being summarized.

The Status Byte Register can be read with either a serial poll or the *STB? query.

The Status Byte Register is altered only when the state of the overlying status data structures is altered.

The entire Status Byte Register can be cleared by sending the *CLS command, by itself or in a program message, to the TIA.

Table 11-1 lists the Status Byte Register bits and briefly describes each bit.

Table 11-1. Status Byte Register

BIT	WEIGHT	SYMBOL	DESCRIPTION
0			Not used
1			Not used
2			Not used
3	8	QSB	Questionable Data/Signal Status Register Summary Bit
4	16	MAV	Message Available Summary Bit
5	32	ESB	Standard Event Status Register Summary Bit
6	64	RQS/MSS	Request Service/Master Status Summary Bit
7	128	OSB	Operation Status Register Summary Bit

A detailed description of each bit in the Status Byte Register follows:

- **Bits 0 - 2** are not used.
- **Bit 3 (QSB)** summarizes the Questionable Data/Signal Status Event Register.

  This bit indicates whether or not one or more of the enabled Questionable Data/Signal events have occurred since the last reading or clearing of the Questionable Data/Signal Status Event Register.

  This bit is set TRUE (one) when an enabled event in the Questionable Data/Signal Status Event Register is set TRUE. Conversely, this bit is set FALSE (zero) when no enabled events are set TRUE.

- **Bit 4 (MAV)** summarizes the Output Queue.

  This bit indicates whether or not the Output Queue is empty.

  This bit is set TRUE (one) when the TIA is ready to accept a request by the external computer to output data byte(s); that is, the Output Queue is not empty. This bit is set FALSE (zero) when the Output Queue is empty.

- **Bit 5 (ESB)** summarizes the Standard Event Status Register.

  This bit indicates whether or not one of the enabled Standard Event Status Register events have occurred since the last reading or clearing of the Standard Event Status Register.

  This bit is set TRUE (one) when an enabled event in the Standard Event Status Register is set TRUE. Conversely, this bit is set FALSE (zero) when no enabled events are set TRUE.

- **Bit 6 (RQS/MSS)** summarizes IEEE 488.1 RQS and Master Summary Status.

  When a serial poll is used to read the Status Byte Register, the RQS bit indicates if the device was sending SRQ TRUE. The RQS bit is set FALSE by a serial poll.

  When *STB? is used to read the Status Byte Register, the MSS bit indicates the Master Summary Status. The MSS bit indicates whether or not the TIA has at least one reason for requesting service.

- **Bit 7 (OSB)** summarizes the Operation Status Event Register.

  This bit indicates whether or not one or more of the enabled Operation events have occurred since the last reading or clearing of the Operation Status Event Register.

  This bit is set TRUE (one) when an enabled event in the Operation Status Event Register is set TRUE. Conversely, this bit is set FALSE (zero) when no enabled events are set TRUE.

Service Request Enable Register

The Service Request Enable Register selects which summary bits in the Status Byte Register may cause service requests as shown in Figure 11-2.

Use *SRE to write to this register and *SRE? to read this register.

Use *SRE 0 to clear the register. A cleared register does not allow status information to generate service requests. (Power-on also clears this register.)

Standard Event Status Register Group

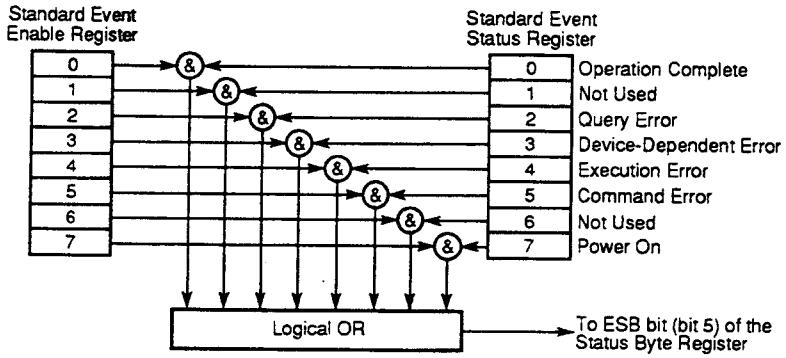


Figure 11-3. Standard Event Status Reporting

Standard Event Status Register

The Standard Event Status Register contains bits that monitor specific IEEE 488.2-defined events as shown in Figure 11-3.

Use *ESR? to read this register.

Use *ESR? or *CLS to clear this register.

Table 11-2 lists the Standard Event Status Register bits and briefly describes each bit.

Table 11-2. Standard Event Status Register

BIT	WEIGHT	SYMBOL	DESCRIPTION
0	1	OPC	Operation Complete
1		(RQC)	Not used because this instrument cannot request permission to become active IEEE 488.1 controller-in-charge.
2	4	QYE	Query Error
3	8	DDE	Device-Specific Error
4	16	EXE	Execution Error
5	32	CME	Command Error
6		(URQ)	Not used because this instrument does not define any local controls as "User Request" controls.
7	128	PON	Power On

A detailed description of each bit in the Standard Event Status Register follows:

- **Bit 0 (Operation Complete)** is an event bit which is generated in response to the *OPC command. This bit indicates that the TIA has completed all pending operations.

  Specifically, this event bit indicates that the pending operation condition has transitioned from TRUE to FALSE.

- **Bit 1** is not used.

- **Bit 2 (Query Error)** is an event bit which indicates that either 1) an attempt was made to read the Output Queue when it was empty or 2) data in the Output Queue has been lost.

  Errors -400 through -499 are query errors.

- **Bit 3 (Device-Specific Error)** is an event bit which indicates an operation did not properly complete due to some condition of the TIA.

  Errors -300 through -399 and all those with positive error numbers (+2000 and above) are device-specific errors.

- **Bit 4 (Execution Error)** is an event bit which indicates that a command could not be executed 1) because the parameter was out of range or inconsistent with the TIA's capabilities, or 2) because of some condition of the TIA.

  Errors -200 through -299 are execution errors.

- **Bit 5 (Command Error)** is an event bit which indicates one of the following has occurred: 1) an IEEE 488.2 syntax error, 2) a semantic error indicating an unrecognized command, or 3) a Group Execute Trigger was entered into the input buffer inside of a program message.

- **Bit 6** is not used.

- **Bit 7 (Power On)** is an event bit which indicates that an off-to-on transition has occurred in the power supplied to the TIA.

Standard Event Status Enable Register

The Standard Event Status Enable Register selects which events in the Standard Event Status Register are reflected in the ESB summary bit (bit 5) of the Status Byte Register as shown in Figure 11-2.

Use *ESE to write to this register and *ESE? to read this register.

Use *ESE 0 to clear the register. (Power-on also clears this register.)

Operation Status Register Group and Questionable Data/Signal Status Register Group

The Operation Status Register Group and the Questionable Data/Signal Status Register Group each have a complete set of registers that consist of the following:

- a condition register
- an event register
- an event enable register

Figure 11-4 shows the model that these register groups follow.

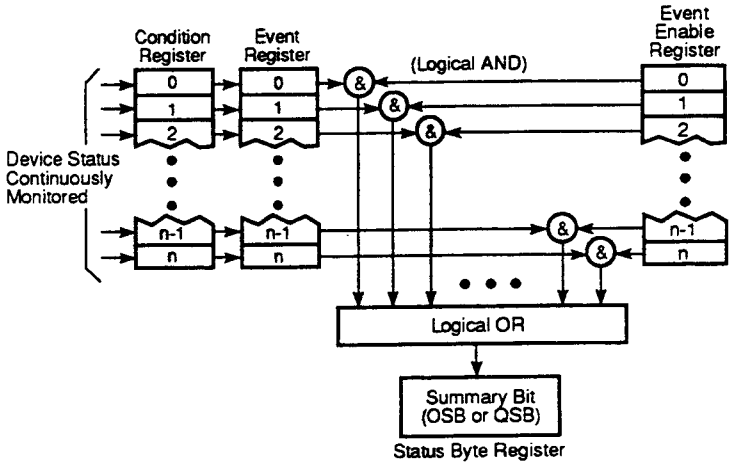


Figure 11-4. Operation and Questionable Status Reporting Model

Condition Register

A condition register continuously monitors the hardware and firmware status of the TIA. There is no latching or buffering for this register; it is updated in real time. Reading a condition register does not change its contents.

To read the condition registers use:

```
:STATus:OPERation:CONDition?
:STATus:QUEStionable:CONDition?
```

Event Register

An event register captures changes in conditions.

An event register bit (event bit) is set TRUE when an associated event occurs. These bits, once set, are "sticky." That is, they cannot be cleared (even if they do not reflect the current status of a related condition) until they are read.

To read the event registers use:

```
:STATus:OPERation[:EVENt]?
:STATus:QUEStionable[:EVENt]?
```

Use event register queries or *CLS to clear event registers.

Event Enable Register

An event enable register selects which event bits in the corresponding event register can generate a summary bit.

To write the event enable registers use:

```
:STATus:OPERation:ENABle <value>
:STATus:QUEStionable:ENABle <value>
```

where the <value> indicates which bits are selected. Sum the weighting values in tables 11-3 or 11-4 to determine the appropriate value.

To read the event enable registers use:

:STATus:OPERation:ENABle?
:STATus:QUEStionable:ENABle?

The event enable registers are cleared by setting:

:STATus:OPERation:ENABle 0 and
:STATus:QUEStionable:ENABle 0 or by power-on.

Operation Status Register Group

The Operation Status Register Group monitors conditions which are part of the TIA's normal operation.

Table 11-3 lists the Operation Status Register bits and briefly describes each bit.

Table 11-3. Operation Status Register

BIT	WEIGHT	DESCRIPTION
0	1	Calibrating
1		Not used
2		Not used
3		Not used
4	16	Measuring
5		Not used
6		Not used
7		Not used
8		Not used
9	512	Using Internal Reference
10		Not used
11-14		Not used
15		Not used since some controllers may have difficulty reading a 16-bit unsigned integer. The value of this bit shall always be 0.

A detailed description of each bit in the Operation Status Register follows:

- **Bit 0 (Calibrating)** is a condition bit which indicates the TIA is currently performing a calibration.

  The condition bit is TRUE (one) during a calibration and FALSE (zero) otherwise.

- **Bits 1-3** are not used.
- **Bit 4 (Measuring)** is a condition bit which indicates the TIA is actively measuring.

  The condition bit is TRUE (one) during a measurement and FALSE (zero) otherwise.

- **Bits 5-8** are not used.
- **Bit 9 (Using Internal Reference)** is a condition bit which indicates the TIA is using the internal reference.

  The condition bit is TRUE (one) while the TIA is using the internal reference. The condition bit is FALSE (zero) while the TIA is using an external reference either EXTernal or CLK10, see [:SENSe]:ROSCillator:SOURce command.

  When this bit is FALSE, it doesn't imply that the TIA is locked to the external reference. Bit 6 of the Questionable Data/Signal Status Register Group can be used to monitor the lock status of an external reference.

- **Bits 10-15** are not used.

Questionable Data/Signal Status Register Group

The Questionable Data/Signal Status Register Group monitors SCPI-defined conditions.

Table 11-4 lists the Questionable Data/Signal Status Register bits and briefly describes each bit.

Table 11-4. Questionable Data/Signal Status Register

BIT	WEIGHT	DESCRIPTION
0		Not used
1		Not used
2		Not used
3		Not used
4		Not used
5		Not used
6	64	External Reference Lock
7		Not used
8	256	Calibration Error
9	512	Missed Event
10	1024	Start
11	2048	Size
12-13		Not used
14	16384	Command Warning
15		Not used since some controllers may have difficulty reading a 16-bit unsigned integer. The value of this bit is always 0.

A detailed description of each bit in the Questionable Data/Signal Status Register Group follows:

- **Bits 0-5** are not used.
- **Bit 6 (External Reference Lock)** is an event which indicates that the external 10 MHz reference has been phase-locked by the HP E1740A. This indication is meaningful for either front-panel external reference input (selected with [:SENS]:ROSC:SOUR EXT) or VXI backplane CLK10 input ([:SENS]:ROSC:SOUR CLK10).
- **Bit 7** is not used.
- **Bit 8 (Calibration Error)** is an event bit which indicates that a calibration has failed.
- **Bit 9 (Missed Event)** is an event bit which indicates when a count event goes beyond 80 MHz. This only works in a range from 80 MHz to about 90 MHz.
- **Bit 10 (Start)** is an event which indicates that the starting meas# parameter of the data retrieval commands (such as :READ?, :FETCh?, and :MEAS?) exceeds the total measurements.
- **Bit 11 (Size)** indicates that the <meas#> parameter of the data retreival commands (:READ?, :FETCh?, :MEAS?) exceeded the total measurement.
- **Bits 12-13** are not used.
- **Bit 14 (Command Warning)** is an event bit indicating a command, such as :CONFigure or :MEASure, ignored a parameter during execution.
- **Bit 15** is not used.

How to Program the TIA for Status Reporting

Determining the Condition of the TIA

The TIA has status registers that are used to indicate its condition. There are four register groups that can be examined individually, or used to alert a computer. These registers, shown in Figure 11-1, are:

- Operation Status Register Group
- Questionable Data/Signal Register Group
- Standard Event Status Register Group
- Status Byte Register Group

The first three groups all have event registers that can be fed into the Status Byte Register. The Status Byte Register can be used to assert the SRQ line of the HP-IB and thus alert the computer that the TIA needs attention. The following examples show how each of the register groups can be used. (Figure 11-5 is a flowchart diagram of how to program the TIA for Status Reporting.)

Resetting the TIA and Clearing the VXI Interface—Example 1

Before attempting any programming, it is a good idea to set the TIA to a known state . The following command grouping shows how to reset the TIA. Before issuing these commands, execute a device clear to reset the interface and TIA. Consult your interface card's documentation for how to issue a device clear since the device clear command will be specific to the interface you are using. Perform the following:

1. Issue a Device Clear. (See your computer or interface card documentation for how to issue this command.)
2. Issue the following commands:

```
*RST
*CLS
*SRE 0
*ESE 0
```

Using the Standard Event Status Register to Trap an Incorrect command—Example 2

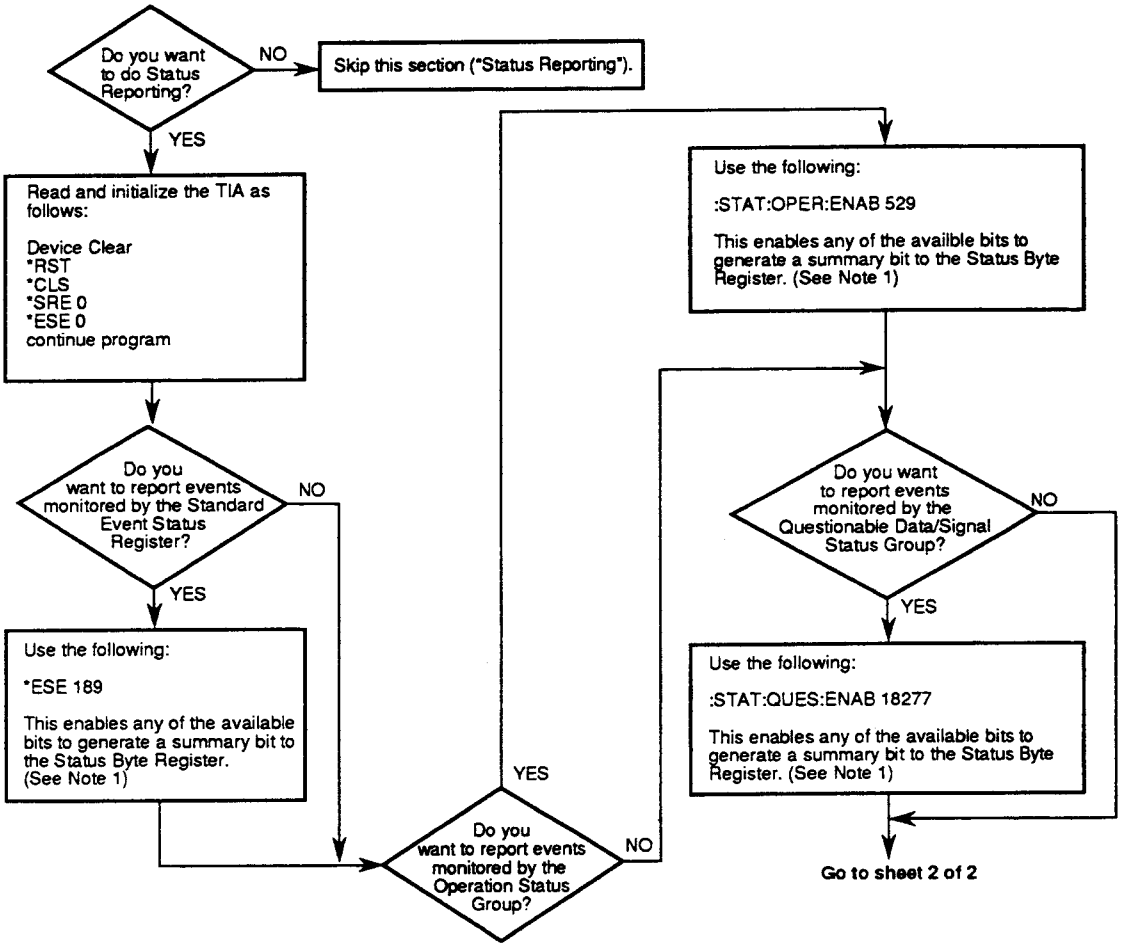
The following command grouping shows how to use the Standard Event Status Register and the Status Byte Register to alert the computer when an incorrect command is sent to the TIA. The command *ESE 32 tells the TIA to summarize the command error bit (bit 5 of the Event Status Register) in the Status Byte Register. The command error bit is set when an incorrect command is received by the TIA. The command *SRE 32 tells the TIA to assert the SRQ line when the Event Status Register summary bit is set to 1. If the TIA is serial polled after a command error, the serial poll result will be 96.

*ESE 32	Enable for bad command.
*SRE 32	Assert SRQ from Standard Event Status Register summary.

Using the Questionable Data/Signal Status Register to Alert the Computer When the TIA Has Locked to an External 10 MHz Reference—Example 3

:STAT:QUES:ENAB 64	Tells the TIA to summarize the detected event (transition to phase locked condition) in the Status Byte Register.
*SRE 8	Tells the TIA to assert SRQ when the summary b for the Questionable Data Register is set to 1.

How to Program the TIA for Status Reporting



NOTES:
1. All of the enable commands, used on this figure, enable all of the active bits for each event register and the Status Byte Register. You do not have to enable all bits. Read the description of each register to determine which bits you want to enable.

2. Additional analysis should be performed if more than one bit is set.

Figure 11-5. Status Reporting Flowchart (1 of 2)

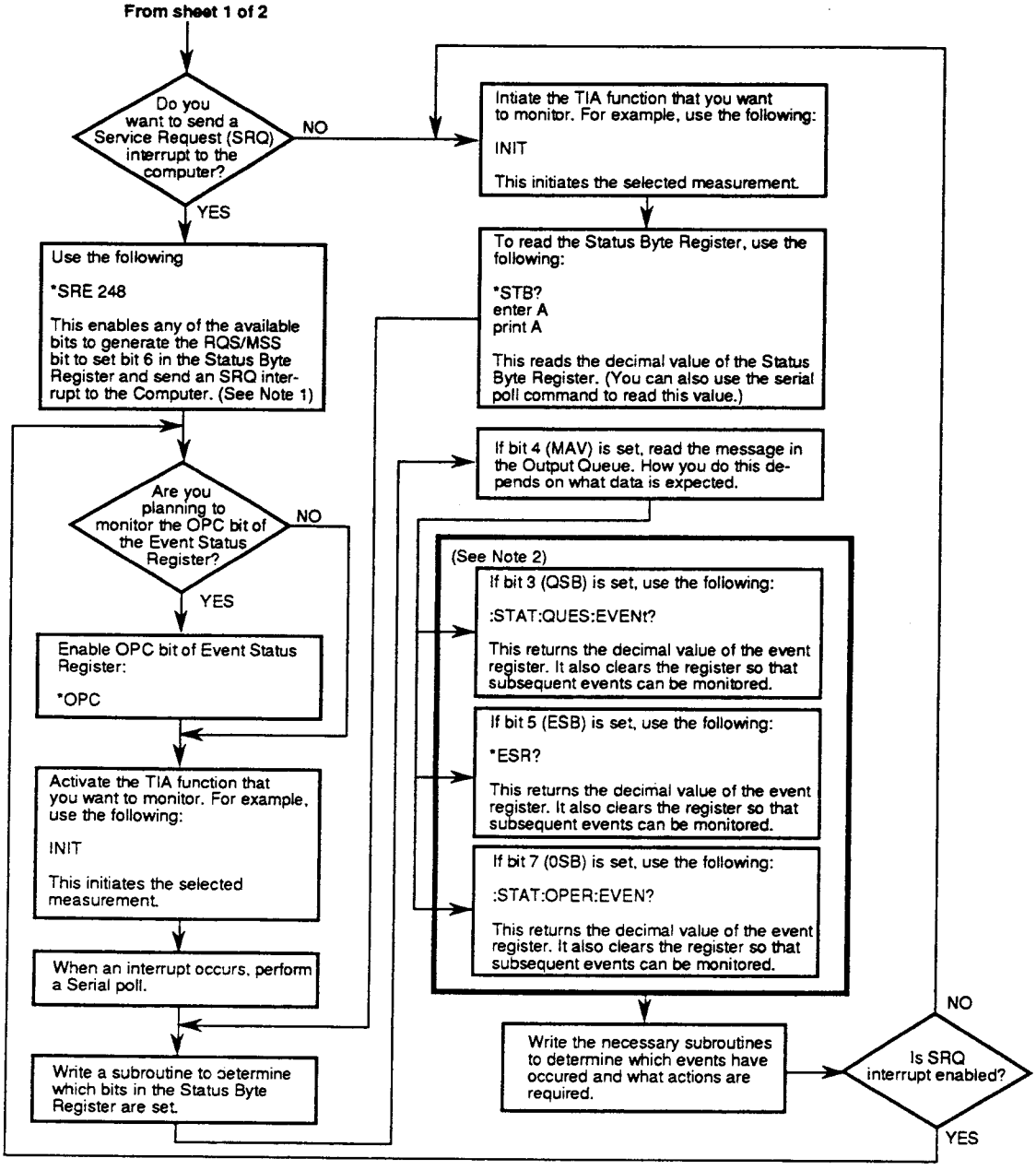


Figure 11-5. Status Reporting Flowchart (2 of 2)

12

Definitions of the TIA IEEE 488.2 Common Commands

Common Commands Summary

Table 12-1. IEEE 488.2 Common Commands

Mnemonic	Command Name	Function
*CLS	Clear Status	Clears Status data structures (Event Registers and Error Queue).
*DMC <string>, <arbitrary block>	Define Macro Command	Assigns a sequence of zero or more commands/queries to a macro label. No query form.
*EMC <NRf>	Enable Macro Command	Enables and disables expansion of macros. Non-zero value enables; zero value disables.
*EMC?	Enable Macro Query	Queries whether macros are enabled.
*ESE <NRf>	Standard Event Status Enable	Sets the Standard Event Status Enable Register.
*ESE?	Standard Event Status Enable Query	Queries the Standard Event Status Enable Register.
*ESR?	Event Status Register Query	Queries the Standard Event Status Register.
*GMC? <string>	Get Macro Contents Query	Queries the current definition of a currently defined macro label.
*IDN?	Identification Query	Queries the TIA identification.
*LMC?	Learn Macro Query	Queries the currently defined macro labels.
Note: Pending operations include measurements in progress.		

Table 12-1. IEEE 488.2 Common Commands (Continued)

Mnemonic	Command Name	Function
*OPC	Operation Complete	Causes TIA to set the operation complete bit in the Standard Event Status Register when all pending operations (see Note) are finished.
*OPC?	Operation Complete Query	Places an ASCII "1" in the Output Queue when all pending operations (see Note) are completed.
*PMC	Purge Macro Command	Deletes all macros previously defined using the *DMC command.
*RST	Reset	Resets the TIA to a known state.
*SRE <NRf>	Service Request Enable	Set the Service Request Enable register.
*SRE?	Service Request Enable Query	Queries the Service Request Enable register.
*STB?	Status Byte Query	Queries the Status Byte and Master Summary Status bit.
*TST?	Self-Test Query	Executes an internal self-test and reports the results.
*WAI	Wait-to-Continue	Makes TIA wait until all pending operations (see Note) are completed before executing commands following *WAI command.

Note: Pending operations include measurements in progress.

*CLS
(Clear Status)

Clears all event registers summarized in the status byte (Standard Event Status Register, Operation Event Status Register, and Questionable Data Event Status Register) and clears the Error Queue. The *CLS command will not clear data or any other settings.

It also places the instrument in "Operation Complete Idle State" and "Operation Complete Query Idle State" (IEEE 488.2). This results in the disabling of any prior *OPC command.

If *CLS immediately follows a program message terminator, the output queue and the MAV bit are cleared because "any" new program message after a program message terminator clears the output queue.

This command will clear the error LED (turn it off) on the front panel.

*DMC <string>, <arbitrary block> (Define Macro Command)

This command assigns a sequence of zero or more commands/queries to a macro label. The sequence is executed when the label is received as a command or query.

The <string> parameter specifies the macro label. The macro label may not be a common command/query header. It may be the same as an instrument-specific command/query header; in this case, provided macros are enabled, the macro expansion is executed and the instrument-specific command/query may be executed by disabling macros.

The <arbitrary block> contains the sequence of commands/queries being labeled.

Parameters may be passed to the sequence during execution. Placeholders for parameters appear in the sequence as a dollar sign followed by a single digit in the range one to nine inclusive. The first parameter following the macro label is substituted for the parameter placeholder labeled $1, and so on up to nine parameters.

Comments

- The maximum macro label length is 12 characters.
- Redefining an existing macro causes an execution error.
- The TIA allows up to four levels of recursion.
- There is no query form. Use *GMC? (see page 12-10) to query the current definition of a macro label.

*EMC <NRf>
(Enable Macro Command)
*EMC?
(Enable Macro Query)

Sets or queries the Enable for defined Macros.

Macro definitions are not affected by this command. One use of this command is to turn off macro expansion in order to execute an instrument-specific command with the same name as a macro.

The value of the numeric parameter determines whether the defined macros are enabled or disabled. A value that rounds to an integer value of zero disables any defined macros. A value that rounds to an integer value not equal to zero enables any defined macros.

<NRf> Range -32767 to +32767

<NRf> Resolution 1

Query Response

- Single ASCII-encoded byte, 0 or 1.
- A value of zero indicates that macros are disabled and a value of one indicates that macros are enabled.

Comments

- *RST value is 1 (enabled)
- Macros are enabled at power-on.

*ESE <NRf>
(Standard Event Status Enable)

*ESE?
(Standard Event Status Enable Query)

Sets or queries the Standard Event Status Enable Register, shown in Figure 12-1.

The parameter and query response value, when rounded to an integer value and expressed in base 2 (binary), represents the bit values of the Standard Event Status Enable Register. The value of unused bits is zero when queried and ignored when set.

This register is used to enable a single or inclusive OR group of Standard Event Status Register events to be summarized in the Status Byte Register (bit 5).

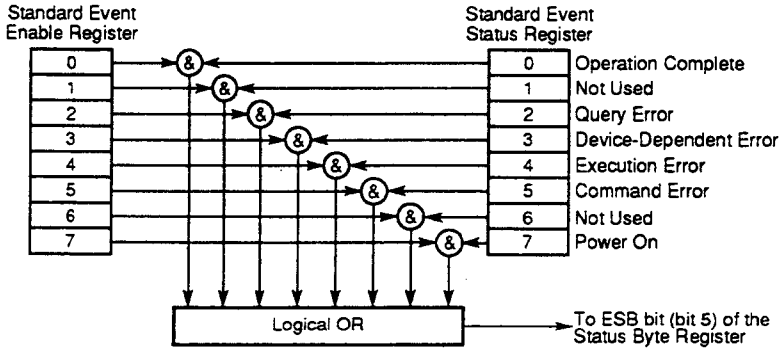


Figure 12-1. The Standard Event Status Enable Register

See the section titled "Standard Event Status Register Group," page 11-8, in Chapter 11 of this guide for a detailed description of the Standard Event Status Register.

*ESE?
(Standard Event Status Enable Query)

Query Response Numeric data transferred as ASCII bytes in <NR1> format.

<NRf> Range 0 to 255

<NRf> Resolution 1

Comments

- At power-on, the Standard Event Status Enable Register is cleared (value is 0).
- This value is unaffected by *RST.

*ESR?
(Event Status Register Query)

Queries the Standard Event Status Register, shown in Figure 12-2.

This event register captures changes in conditions, by having each event bit correspond to a specific condition in the instrument. An event becomes TRUE when the associated condition makes the defined transition. The event bits, once set, are "sticky." That is, they aren't cleared, even if they do not reflect the current status of the related condition, until they are read.

This register is cleared by *CLS, by *ESR?, and at power-on. Note that the instrument's power-on sequence initially clears the register, but then records any subsequent events during the power-on sequence including setting the PON (power on) bit.

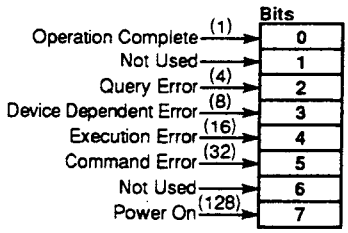


Figure 12-2. Standard Event Status Register

See the section titled "Standard Event Status Register Group," page 11-8, in Chapter 11 of this guide for a detailed description of the Standard Event Status Register.

Query Response

- Numeric data transferred as ASCII bytes in <NR1> format.
- Range is 0 to 255.
- The query response is an integer formed by the binary-weighting of the bits.

  For example, a response of 12 indicates that a query error (bit 2, value of 4) and a device dependent error (bit 3, value of 8) have occurred.

*GMC? <string> (Get Macro Contents Query)

Queries the current definition of a macro.

The <string> parameter must be a currently defined macro label.

Query Response

- The query response is a <definite length block> containing the command/query sequence which is executed when the macro label is received.
- A zero-length block response indicates that no command sequence is stored by the specified label.

*IDN?
(Identification Query)

Queries the TIA identification.

Query Response

A sequence of ASCII-encoded bytes:

HEWLETT-PACKARD, E1740,0,XXXX

terminated with a new line and EOI.

XXXX represents the firmware date code.

Comments

This query should be the last query in a terminated program message; otherwise, error -440 is generated.

*LMC?
(Learn Macro Query)

Queries the currently defined macro labels.

Query Response

- A sequence of one or more strings separated by commas.
- If no macros are defined, the response is a null string (two consecutive double quote marks).

*OPC
(Operation Complete)

This event command enables the OPC bit in the Standard Event Status Register (bit 0) to be set upon the transition of the measurement cycle from measuring to idle. See the section titled "Standard Event Status Register Group," page 11-8, in Chapter 11 of this guide for a detailed description of the Standard Event Status Register's Operation Complete bit.

This event command is "disabled" by *CLS, *RST, Device Clear (page 13-31), power-on, or upon the transition of the measurement cycle from measuring to idle.

This event command has no query form.

*OPC?
(Operation Complete Query)

This query places a single ASCII-encoded byte for "1" in the Output Queue upon the transition of the measurement cycle from measuring to idle. This allows synchronization between a controller and the instrument using the MAV bit in the Status Byte Register or a read of the Output Queue. (Note that it does not actually "read" a state, as most queries do.)

Since this query will not respond until the measurement cycle transitions from measuring to idle, the only way to cancel the query "holdoff" is by Device Clear (page 13-31) or power-on.

Query Response Single ASCII-encoded byte, 1.

*PMC
(Purge Macro Command)

The Purge Macros command deletes *all* macros previously defined using the *DMC command. Use :SYST:PIMacro <string> (Purge Immediate Macro) to purge a *single* macro (see :SYSTem:PIMacro in Chapter 13 of this guide for detailed information).

*RST
(Reset)

This event command performs an instrument reset.

The reset performs the following:

- sets instrument settings to their *RST states,
- disables macros,
- places instrument in "Operation Complete Idle State" and "Operation Complete Query Idle State", and

The reset does not affect:

- the macros defined with *DMC,
- the calibration data,
- the Service Register Enable or the Standard Event Status Enable,
- the Output Queue, and
- the IEEE 488.1 address or the state of the IEEE 488.1 interface.

See the section titled "*RST Response," page 1-19, in Chapter 1 of this guide for a complete listing of the *RST states.

Where applicable, command definitions in this chapter (Chapter 12) and Chapter 13 include the *RST state in the "Comment" portion of the definition.

*SRE <NRf>
(Service Request Enable)
*SRE?
(Service Request Enable Query)

Sets or queries the Service Request Enable Register, shown in Figure 12-3.

The parameter and query response value, when rounded to an integer value and expressed in base 2 (binary), represents the bit values of the Service Request Enable Register.

This register is used to enable a single or inclusive OR group of Status Byte Register events to generate an SRQ.

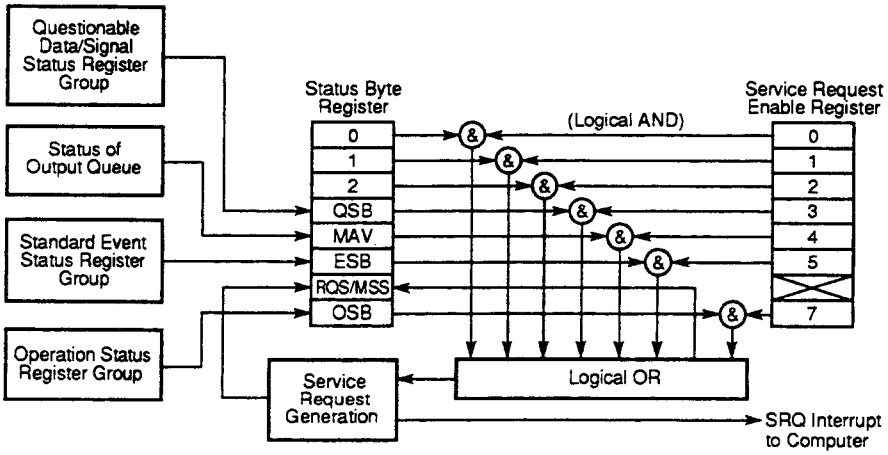


Figure 12-3. The Service Request Enable Register

See the section titled "Status Byte Register and Service Request Enable Register," page 11-5, in Chapter 11 of this guide for a detailed description of the Service Request Enable Register.

*SRE?
(Service Request Enable Query)

<NRf> Range

- 0 to 255
- The value of bit 6 and unused bits is ignored when set.

<NRf> Resolution

1

Query Response

- Numeric data transferred as ASCII bytes in <NR1> format.
- The value of bit 6 and unused bits is zero when queried.

Comments

- At power-on, this value is cleared (set to 0).
- This value is unaffected by *RST.

*STB?
(Status Byte Query)

Queries the Status Byte Register, shown in Figure 12-4.

This register is cleared at power-on.

This query does not directly alter the Status Byte Register (including the MSS/RQS bit) or anything related to the generation of SRQ.

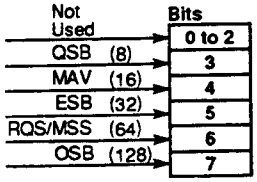


Figure 12-4. The Status Byte Register

See the section titled "Status Byte Register and Service Request Enable Register," page 11-5, in Chapter 11 of this guide for a detailed description of the Status Byte Register.

Query Response

- Numeric data transferred as ASCII bytes in <NR1> format.
- Range is 0 to 255.
- The response value when rounded to an integer value and expressed in base 2 (binary), represents the bit values of the Status Byte Register.
- The value of unused bit is zero when queried.
- The Master Summary Status, not the RQS message, is reported on bit 6. Master Summary Status indicates that the TIA has at least one reason for requesting service. (The Master Summary Status is not sent in response to a serial poll; the IEEE 488.1 RQS message is sent instead.) It is the inclusive OR of the bitwise combination (excluding bit 6) of the Status Byte Register and the Service Request Enable Register.

*TST?
(Self-Test Query)

This query causes an internal self-test and the response indicates whether any errors were detected.

Error -330 is generated if the self-test fails.

Query Response

- Numeric data transferred as ASCII bytes in <NR1> format.
- A response value of zero indicates the self-test has completed without errors detected, while a non-zero value indicates the self-test was not completed or was completed with errors detected.

Comments

The following are tested:

Shared RAM,
EPROM,
EEPROM,
NVRAM,
FPGAs,
Histogram IC's, and
Video RAMs.

*WAI (Wait-to-Continue)

This command prevents the instrument from executing any further commands or queries until the measurement cycle transitions from measuring to idle. The only way to cancel this "holdoff" is by device clear or power-on. (*RST and *CLS have no affect on *WAI operation.)

***WAI**

(Wait-to-Continue)

13

Definitions of the TIA Subsystem Commands

Introduction

HP E1740A SCPI Subsystem Commands

This chapter describes the SCPI Subsystem commands and the Device Clear command for the HP E1740A.

SCPI Subsystem commands include all measurement functions and some general purpose functions. SCPI Subsystem Commands use a hierarchical relationship between keywords that is indicated by a ":" (colon). For example, in the SYST:ERR? query, the ":" between SYST and ERR? indicates ERR? is subordinate to SYST.

Table 13-1 lists the SCPI Subsystem Commands in alphabetical order by the command keyword. The table shows the Subsystem commands hierarchical relationship, and related parameters (if any).

Std/New Column

The **Std/New** column in Table 13-1 gives the status of the command with respect to the SCPI standard. The "Std" commands operate as defined in the SCPI standard and as defined in this guide.

The category of "New" consists of commands that could be:

- SCPI approved but are not yet in the SCPI manual
- HP approved and submitted for SCPI approval.
- Not approved at all.

The new commands operate as defined in this guide.

Parameter Form Column

Refer to the section titled "Parameter Types" on page 2-10 in Chapter 2, "Overview of How to Use and Program the TIA," for descriptions of the different parameter types (such as <Boolean>, <NRf>, <arbitrary block>, etc.). Also information on command syntax and query response types can be found in Chapter 1.

See Table 1-3 in Chapter 1, "Getting Started" for *RST or power-up values.

Subsystem Commands Summary

Table 13-1. HP E1740A SCPI Commands Summary

Keyword/Syntax	Parameter Form	Std/New
:ABORt		Std
:CALCulate		Std
:WMARgin		New
:DATA?		New
:DATA		New
:ESIDe?		New
:ESIDe		New
:REFerence?		New
:LSIDe?		New
:LSIDe		New
:REFerence?		New
:REFerence?		New
:EDATa?		New
:EDATa		New
:ESIDe?		New
:ESIDe		New
:REFerence?		New
:LSIDe?		New
:LSIDe		New
:REFerence?		New
:REFerence?		New
:EXTRapolation		New
:RANGe		New
:LOWer		New
:LOWer?	<numeric_value>	New
:UPPer		New
:UPPer?	<numeric_value>	New
:STATe	<Boolean>	New
:STATe?		New
:MARGin?		New
:MARGin		New
:EARLy?		New
:LATE?		New
:MLEVel	<numeric_value>	New
:MLEVel?		New
:NOISe?		New
:NOISe		New
:EARLy?		New
:LATE?		New
:OFFSet?		New

Table 13-1. HP E1740A SCPI Commands Summary (Continued)

Keyword/Syntax	Parameter Form	Std/New
:CALCulate (Cont.)		**Std**
:WMARgin (Cont.)		New
:SEGMents		New
:CENTer	<numeric_value>	New
:CENTer?		New
:COUNt	<numeric_value>	New
:COUNt?		New
:ENABle	<non-decimal numeric> | <numeric_value> | ALL	New
:ENABle?		New
:OFFSet	<numeric_value>	New
:OFFSet?		New
:WIDTh	<numeric_value>	New
:WIDTh?		New
:SIDes	One | TWO	New
:SIDes?		New
[:STATe]	<Boolean>	New
[:STATe]?		New
:CALibration		New
:INPut[1|2]		New
:GAIN		New
:EXECute		New
:EXECute?		New
:NDELay?		New
:NDELay		New
:EXECute		New
:EXECute?		New
:OFFSet		New
:EXECute		New
:EXECute?		New
:TIMing?		New
:TIMing		New
:EXECute		New
:EXECute?		New
:INTerpolator		New
:EXECute		New
:EXECute?		New
:RECall		New
:SAVE		New
:TIMebase		New
:EXECute		New
:EXECute?		New

Table 13-1. HP E1740A SCPI Commands Summary (Continued)

Keyword/Syntax	Parameter Form	Std. New
:CONFigure?		Std
:CONFigure		Std
:XTIMe		Std
[:VOLTage]		Std
:TINTerval	[<start_meas#>[,<# of meas>[, <source_list>]]]	Std
:TSTamp	[<start_meas#>[,<# of meas>[, <source_list>]]]	Std
:XTINterval		New
[:VOLTage]		New
:HISTogram	[<start_meas#>[,<# of meas>[, <source_list>]]]	New
DIAGnostic		**Std**
	_____DIAGNOSTIC TEST NAMES_____	
:TEST? 0	Test All	New
:TEST? 1	SRAM	New
:TEST? 2	EPROM	New
:TEST? 3	EEPROM	New
:TEST? 4	Non-volatile RAM	New
:TEST? 5	XiLINX	New
:TEST? 6	Histogram	New
:TEST? 7	VRAM (video RAM)	New
:TEST? 8	Correction RAM	New
:TEST? 9	79.9 MHz	New
:TEST? 10	Event Pacing Counter	New
:TEST? 11	No test assigned	New
:TEST? 12	Randomizer	New
:TEST? 13	Input board Register	New
:TEST? 14	Trigger Level and Event Count (Input 1 & 2)	New
:TEST? 15	Inhibit	New
:TEST? 16	Trigger Input	New
:TEST? 17	Timebase	New
:FETCh?		**Std**
:FETCh		**Std**
:MAX?	[<source_list>]	New
:MIN?	[<source_list>]	New
:PTPeak?	[<source_list>]	New
[:SCALar]		Std
[:VOLTage]		Std
:TINTerval		New
:MAXimum?	[<start_block#>[,<# of block>]]	New
:MEAN?	[<start_block#>[,<# of block>]]	New
:MINimum?	[<start_block#>[,<# of block>]]	New
:SDEViation?	[<start_block#>[,<# of block>]]	New
:TINTerval?	[<start_meas#>[,<# of meas>]]	New
:TSTamp?	[<start_meas#>[,<# of meas>]]	New

Table 13-1. HP E1740A SCPI Commands Summary (Continued)

Keyword/Syntax	Parameter Form	Std/New
:FETCh? (Cont.)		
:XTIMe		New
[:VOLTage]		New
:FREQuency?	[<start_meas#>[,<# of meas>]]	New
:TINTerval?	[<start_meas#>[,<# of meas>]]	New
:TSTamp?	[<start_meas#>[,<# of meas>]]	New
:XTINterval		New
[:VOLTage]		New
:HISTogram?	[<start_meas#>[,<# of meas>]]	New
:FORMat		**Std**
[:DATA]	ASCii | INTeger | REAL	Std
[:DATA]?		Std
:INITiate		Std
[:IMMediate]		Std
:INPut[1|2]		Std
:COUPling	AC | DC,	Std
:COUPling?		Std
:IMPedance	<numeric_value>	Std
:IMPedance?		Std
:ROUTe	COMMon | SEParate	New
:ROUTe?		New
:MEASure		Std
:XTIMe		New
[:VOLTage]		New
:TINTerval?	[<start_meas#>[,<# of meas>[, <source_list>]]]	New
:TSTamp?	[<start_meas#>[,<# of meas>[, <source_list>]]]	New
:XTINterval		New
[:VOLTage]		New
:HISTogram?	[<start_meas#>[,<# of meas>[, <source_list>]]]	New
:OUTPut		New
:ECLTrg[0|1]		Std
[:STATe]	<Boolean>	Std
[:STATe]?		Std
:READ?		Std
:READ		Std
[:SCALar]		Std
[:VOLTage]		Std
:TINTerval		New
:MAXimum?	[<start_block#>[,<# of block>]]	New
:MEAN?	[<start_block#>[,<# of block>]]	New
:MINimum?	[<start_block#>[,<# of block>]]	New
:SDEViation?	[<start_block#>[,<# of block>]]	New

Table 13-1. HP E1740A SCPI Commands Summary (Continued)

Keyword/Syntax	Parameter Form	Std/New
:READ? (Continued		
:XTIMe		New
[:VOLTage]		New
:FREQuency?	[<start_meas#>[,<# of meas>]]	New
:TINTerval?	[<start_meas#>[,<# of meas>]]	New
:TSTamp?	[<start_meas#>[,<# of meas>]]	New
:XTINterval		New
[:VOLTage]		New
:HISTogram?	[<start_meas#>[,<# of meas>]]	New
[:SENSe]		Std
:ACQuisition		New
:BLOCk?	BSTart | INHibit, <start_#>, <# of blocks or inhibits>	New
:LENGth?	DATA | BSTart | INHibit	New
:MCOunt	<numeric_value>	New
:MCOunt?		New
:PACing		New
[:SOURce]	IMMediate | RANDom | STEP	New
[:SOURce]?		New
:STEP	<numeric_value>	New
:STEP?		New
:SOURce	MCOunt | TIMer	New
:SOURce?		New
:TIMer	<numeric_value>	New
:TIMer?		New
:EVENt[1|2]		New
:HYSTeresis		New
:RELative	<numeric_value>	New
:RELative?		New
:LEVel	<numeric_value>	New
:LEVel?		New
:AUTO		New
:AUTO?		New
:SLOPe	POSitive | NEGative | EITHer	New
:SLOPe?		New
:FUNCtion	"<sensor_function>"	New
:FUNCtion?		New
:HISTogram		New
:ACCumulate		New
[:STATe]	<Boolean>	New
[:STATe]?		New
:CLEar		New
:COUNt?		New

Table 13-1. HP E1740A SCPI Commands Summary (Continued)

Keyword/Syntax	Parameter Form	Std/New
[:SENSe] (Cont.)		Std
:ROSCillator		New
:SOURce?		New
:SOURce	EXTernal | INTernal | CLK10	New
:STATus?		New
:TINTerval		New
:RANGe		New
:RESolution	<numeric_value>	New
:RESolution?		New
[:UPPer]	<numeric_value>	New
[:UPPer]?		New
:TSTamp		New
:ETRigger		New
[:STATe]	<Boolean>	New
[:STATe]?		New
:NDELay		New
[:STATe]	<Boolean>	New
[:STATe]?		New
:STATus		Std
:OPERation		Std
:CONDition?		Std
:ENABle	<NRf> | <non-decimal_numeric>	Std
:ENABle?		Std
[:EVENt]?		Std
:QUEStionable		Std
:CONDition?		Std
:ENABle	<NRf> | <non-decimal_numeric>	Std
:ENABle?		Std
[:EVENt]?		Std
:SYSTem		Std
:ERRor?		Std
:PIMacro	<string>	Std
:VERSion?		Std

Table 13-1. HP E1740A SCPI Commands Summary (Continued)

Keyword/Syntax	Parameter Form	Std/New
:TRIGger		Std
:COUNt	<numeric_value>	Std
:COUNt?		Std
:DELay		New
:ECOunt	<numeric_value>	New
:ECOunt?		New
:SOURce	TIMer | INPut | IMMediate	New
:SOURce?		New
:TIMer	<numeric_value>	New
:TIMer?		New
:LEVel	<numeric_value> | TTL | ECL | TTL10 | ECL10 | GND	Std
:LEVel?		Std
:SLOPe	POSitive | NEGative	Std
:SLOPe?		Std
:SOURce	EXTernal | IMMediate | ECLT0 | ECLTRG0 | ECLT1 | ECLTRG1	Std
:SOURce?		Std

:ABORt

This command is an event that causes the TIA to abort any measurement in progress, as quickly as possible.

The :ABORt command is not complete until the current measurement is stopped. The execution of an ABORt command sets false any Pending Operation Flags that were set true by initiation of measuring.

Comments

When a measurement or block of measurements is aborted, the Measuring bit in the Operation Status Register will be set false.

:CALCulate Subsystem

In the HP E1740A TIA, the :CALCulate subsystem is used for computation of window margin results, which are derived from acquired histogram measurements. Along with the window margin results, all of the parameters associated with defining how the window margin should be computed are controlled within the :CALCulate subsystem.

NOTE

Window margin results are only available when the configuration is histogram time intervals (CONFigure:XTINterval:HISTogram).

:CALCulate:WMARgin:DATA?

Queries the window margin data when SIDes is set to ONE. The data can be read out in REAL or ASCii formats (see :FORMat). The data content is the sequence of log error rate values, working from left to right. The visual model is to picture the window margin data as it might be plotted, then the margin data comes out with the error rate value for the leftmost x-axis position first and ends with the error rate value for the rightmost x-axis position.

Query Response

A binary block or an ASCII array of 64-bit floating point numbers <NR3>, depending on the :FORMat subsystem.

Comments

- Query only.
- The length of the output depends on the histogram resolution ([:SENS]:HIST:RANG:RES) and the segment width value (:CALC:WMAR:SEGM:WIDT).
- Sending this query with SIDes set to TWO will cause the TIA to read the entire array (early followed by late).
- Returns window margin data that is derived from acquired data. There is no extrapolation information returned by this query.

:CALCulate:WMARgin:DATA:ESIDe?

Queries the early side window margin data when SIDes is set to TWO. The data can be read in either REAL or ASCii formats (see :FORMat). The data content is the sequence of log error rate values, working from left to right. The visual model is to picture the window margin data as it might be plotted, then the margin data comes out with the error rate value for the leftmost x-axis position first and ends

with the error rate value for the rightmost x-axis position. The early side data starts toward the early side segment boundary and sequences toward the segment center.

Query Response A binary block or an ASCII array of 64-bit floating point numbers <NR3>, depending on the :FORMat subsystem.

Comments

- Query only.
- The length of the output depends on the histogram resolution ([:SENS]:HIST:RANG:RES) and the segment width value (:CALC:WMAR:SEGM:WIDT).
- Sending this query with SIDes set to ONE will cause the TIA to return an empty string in ASCII format.
- This query only returns window margin data that is derived from acquired data. There is no extrapolation information returned by this query.

:CALCulate:WMARgin:DATA:ESIDe:REFerence?

Queries the reference value for the window margin data that can be read with the :CALC:WMAR:DATA:ESID? query. The reference defines the starting position of the first data point. Thus, to compute the x-axis location of a given data value, use the computation:

reference_value + [(i-1) − resolution],

where reference_value is the result from this query and i is the sample number being investigated (e.g., for i=1, which is the first sample, the x-axis location is the reference_value). The resolution can be obtained from the [:SENS]:HIST:RANG:RES? query.

Query Response Numeric data transferred as ASCII bytes in floating point <NR3> format.

Comments Query only.

:CALCulate:WMARgin:DATA:LSIDe?

Queries the late side window margin data when SIDes is set to TWO. The data can be read in REAL or ASCII formats (see :FORMat). The data content is the sequence of log error rate values, working from left to right. The visual model is to picture the window margin data as it might be plotted, then the margin data comes out with the error rate

value for the leftmost x-axis position first and ends with the error rate value for the rightmost x-axis position. The late side margin data starts at the segment center and sequences out toward the late side segment boundary.

Query Response A binary block or an ASCII array of 64-bit floating point numbers <NR3>, depending on the :FORMat subsystem.

Comments

- Query only.
- The length of the output depends on the histogram resolution ([:SENS]:HIST:RANG:RES) and the segment width value (:CALC:WMAR:SEGM:WIDT).
- Sending this query with SIDes set to ONE will cause the TIA to read out the one-sided data.
- This query returns only window margin data that is derived from acquired data. There is no extrapolation information returned by this query.

:CALCulate:WMARgin:DATA:LSIDe:REFerence?

Queries the reference value for the window margin data that can be read with the :CALC:WMAR:DATA:LSID? query. The reference defines the starting position of the first data point. Thus, to compute the x-axis location of a given data value use the computation:

reference_value + [(i-1) − resolution],

where reference_value is the result from this query and i is the sample number being reviewed (e.g., for i=1 which is the first sample, the x-axis location is the reference_value). The resolution can be obtained from the [:SENS]:HIST:RANG:RES? query.

Query Response Numeric data transferred as ASCII bytes in floating point <NR3> format.

Comments Query only. Result will always be 0.

:CALCulate:WMARgin:DATA:REFerence?

Queries the reference value for the window margin data that can be read with the :CALC:WMAR:DATA? query. The reference defines the starting position of the first data point. Thus, to compute the x-axis location of a given data value, use the computation

reference_value + [(i-1) − resolution],

where reference_value is the result from this query, i is the sample number being reviewed (e.g., for i=1 which is the first sample, the x-axis location is simply the reference_value). The resolution can be obtained from the [:SENS]:HIST:RANG:RES? query.

Query Response Numeric data transferred as ASCII bytes in floating point <NR3> format.

Comments Query only.

:CALCulate:WMARgin:EDATa?

Queries the window margin extrapolation data when SIDes is set to ONE. Other than returning extrapolated window margin data instead of window margin directly derived from acquired histogram data, this query is similar to the :CALC:WMAR:DATA? command.

Query Response A binary block or an ASCII array of 64-bit floating point numbers <NR3>, depending on the :FORMat subsystem.

Comments

- Query only.
- The data returned by this query is affected by the :CALC:WMAR:EXTRapolation group commands. The Extrapolated data *is* available even if :CALC:WMAR:EXTR:STAT is OFF. :EXTR:STAT only affects the reading of window margin. Also, the :RANGe:LOWer and :RANGe:UPPer commands can modify the range of non-extrapolated window margin that is used to create the window margin extrapolation data.
- This query returns only extrapolated window margin data. There is no mixing of extrapolated and non-extrapolated data.

:CALCulate:WMARgin:EDATa:REFerence?

Queries the reference value for the extrapolated window margin data that can be read with the :CALC:WMAR:EDAT? query. See the :CALC:WMAR:DATA:REF? description for information about the reference value.

Query Response Numeric data transferred as ASCII bytes in floating point <NR3> format.

Comments

Query only.

:CALCulate:WMARgin:EDATa:ESIDe?

Queries the early side window margin extrapolation data when SIDes is set to TWO. Other than returning extrapolation window margin data instead of window margin directly derived from acquired histogram data, this query is similar to the :CALC:WMAR:DATA:ESID? command.

Query Response

A binary block or an ASCII array of 64-bit floating point numbers <NR3>, depending on the :FORMat subsystem.

Comments

- Query only.
- The data returned by this query is affected by the :CALC:WMAR:EXTRapolation group commands. The Extrapolated data *is* available even if :CALC:WMAR:EXTR:STAT is OFF. :EXTR:STAT only affects the reading of window margin. Also, the :RANGe:LOWer and :RANGe:UPPer commands can modify the range of non-extrapolated window margin that is used to create the window margin extrapolation data.
- This query only returns extrapolated window margin data. Extrapolated and non-extrapolated data are not mixed.

:CALCulate:WMARgin:EDATa:ESIDe:REFerence?

Queries the reference value for the extrapolated window margin data that can be read with the :CALC:WMAR:EDATa:ESID? query. See the :CALC:WMAR:DATA:REF? description for information about the reference value.

Query Response

Numeric data transferred as ASCII bytes in floating point <NR3> format.

Comments

Query only.

:CALCulate:WMARgin:EDATa:LSIDe?

Queries the late side window margin extrapolation data when SIDes is set to TWO. Other than returning extrapolated window margin data instead of window margin directly derived from acquired histogram data, this query is similar to the :CALC:WMAR:DATA:LSID? command.

Query Response A binary block or an ASCII array of 64-bit floating point numbers <NR3>, depending on the :FORMat subsystem.

Comments

- Query only.
- The data returned by this query is affected by the :CALC:WMAR:EXTRapolation group commands. The Extrapolated data *is* available even if :CALC:WMAR:EXTR:STAT is OFF. :EXTR:STAT only affects the reading of window margin. Also, the :RANGe:LOWer and :RANGe:UPPer commands can modify the range of non-extrapolated window margin that is used to create the window margin extrapolation data.
- This query returns only extrapolated window margin data. Extrapolated and non-extrapolated data are not mixed.

:CALCulate:WMARgin:EDATa:LSIDe:REFerence?

Queries the reference value for the extrapolated window margin data that is read with the :CALC:WMAR:EDAT:LSID? query. See the :CALC:WMAR:DATA:REF? description for information about the reference value.

Query Response Numeric data transferred as ASCII bytes in floating point <NR3> format.

Comments Query only. Result will always be 0.

:CALCulate:WMARgin:EXTRapolation:STATe <Boolean>

Controls whether or not use of extrapolated data is allowed when computing window margin. If the STATe is ON, then the HP E1740A will extrapolate, if necessary, to determine the margin values. If the STATe is OFF, then no extrapolation from the data will be attempted, and an error will be reported if the acquired data does not extend below margin level.

Query Response

- Single ASCII-encoded byte, 0 or 1.
- A value of 0 indicates OFF; a value of 1 indicates ON
- ON enables extrapolation. OFF disables extrapolation.

Comments

- *RST state is ON (extrapolation will occur).

- Extrapolation is only used for margin computation when the acquired data does not extend below the user-defined margin level (the error rate level at which the margin is determined, set with the :CALC:WMAR:MLEV <value> command). If the STATe is OFF and the data does not extend below the specified margin level, then an error is produced upon an attempt to read window margin.
- Having the STATe ON does not guarantee that extrapolation will occur. This is also dependent on the presence of window margin data within the user-selectable extrapolation range (controlled via the :CALC:WMAR:EXTR:RANG:LOWer and CALC:WMAR:EXTR:RANG:UPPer commands). If no data exists within these specified boundaries, no extrapolation can occur.
- This state has no effect on whether or not extrapolated data is available via the :CAL:WMAR:EDATa? family of commands.

:CALCulate:WMARgin:EXTRapolation:RANGe:LOWer <numeric_value>

Controls the lower limit of the range of data on which extrapolation is based. The value is set as a log error rate. The range, defined by the LOWer and UPPer commands, identifies a range of error rate values from which to base the extrapolation. For example, if the LOWer value is -15.0 and the UPPer value is -6.0, then only window margin data in the error rate range from -6.0 (1 part in 10^{6}) down to -15.0 (1 part in 10^{15}) will be considered when developing the extrapolation data. Reasons for wanting to control this range include the need to specifically avoid peak shift data in the extrapolation (lower the UPPer value) or to single out a single noise process in an aggregate distribution to discover how that process might be contributing to the overall result.

<numeric_value> Range

-0.5 to -20.0 (the values are log error rate, so -1.0 means 1 part in $10^{-1.0} = 0.1$)

Query Response

Numeric data transferred as ASCII bytes in floating point <NR3> format.

Comments

- *RST value is -18.0. The default essentially means that all of the lower error rate data will be included in the extrapolation projection unless you specifically change this value.
- If the LOWer value is not below the UPPer value, there will be no data on which to base extrapolation.

:CALCulate:WMARgin:EXTRapolation:RANGe:UPPer <numeric_value>

Controls the upper limit of the range of data on which extrapolation is based. The value is set as the log error rate. The range, defined by the UPPer and LOWer commands, identifies a range of error rate values from which to base the extrapolation. For example, if the UPPer value is -6.0 and the LOWer value is -9.0, then only window margin data in the error rate range from -6.0 (1 part in 10^6) down to -9.0 (1 part in 10^9) will be considered when developing the extrapolation data. Reasons for wanting to control this range include the need to specifically avoid peak shift data in the extrapolation (lower the UPPer value) or to single out a single noise process in an aggregate distribution in order to discover how that process might be contributing to the overall result.

<numeric_value> Range

-0.5 to -20.0 (the values are log error rate, so -1.0 means 1 part in $10^{-1.0} = 0.1$)

Query Response

Numeric data transferred as ASCII bytes in floating point <NR3> format.

Comments

- *RST value is -1.5. The default is set below zero to avoid automatically including time shifted data in the extrapolation projection. This data produces a flat top in the lower error rate portions of the window margin data and is not indicative of any noise process, which is generally what you want to include in the extrapolation projection.
- If the LOWer value is not below the UPPer value, there will be no data on which to base extrapolation.

:CALCulate:WMARgin:MARGin?

Queries the window margin at the user-specified margin level (:CALC:WMAR:MLEVel) when SIDes is set to ONE. The margin value is reported as a time value (seconds) and is essentially the time from the segment boundary to the intersection of the window margin curve with the margin level. Another quantity that is often used, bit shift, can easily be derived from the margin value by subtracting the margin value from 1/2 the segment width. These values can be converted to a percent by dividing by 1/2 the segment width and multiplying by 100%.

Query Response Numeric data transferred as ASCII bytes in floating point <NR3> format.

Comments

- Query only.
- If SIDes is set to TWO, 0 is returned and an error is reported.
- The margin value is based on either window margin data directly derived from the histographic data or from the extrapolation projection taken from the window margin data. You can easily determine what data was used. If the window margin data derived from the histographic data crosses the indicated margin level, then that data is the basis for the reported margin value. Extrapolation data is only used when the directly derived data does not extend to the specified margin level.
- If you want to ensure that margin is only reported based on window margin data taken directly from the histographic data, turn off extrapolation with the :CALC:WMAR:EXTR:STAT OFF command. Subsequently, if the direct window margin data does not extend to the specified margin level, 0 is returned, and an error is reported.

:CALCulate:WMARgin:MARGin:EARLy?

Queries the early-side window margin at the user-specified margin level (see :CALC:WMAR:MLEVel for setting this value) when SIDes is set to TWO. The margin value is reported as a time value (seconds) and is essentially the time from the segment boundary to the intersection of the window margin curve with the margin level. Another quantity that is often used, bit shift, can easily be derived from the margin value by subtracting the margin value from 1/2 the segment width. These values can be converted to a percent by dividing by 1/2 the segment width and multiplying by 100%.

Query Response Numeric data transferred as ASCII bytes in floating point <NR3> format.

Comments

- Query only.
- If SIDes is set to ONE, 0 is returned and an error is reported.
- The margin value is based on either window margin data directly derived from the histographic data or from the extrapolation projection taken from the window margin data. You can easily determine what data was used. If the window margin data derived

from the histographic data crosses the indicated margin level, then that data is the basis for the reported margin value. Extrapolation data is only used when the directly derived data does not extend to the specified margin level.

- If you want to ensure that margin is only reported based on window margin data derived from the histographic data, turn off extrapolation using the :CALC:WMAR:EXTR:STAT OFF command. Subsequently, if the direct window margin data does not extend to the specified margin level, 0 is returned and an error is reported.

:CALCulate:WMARgin:MARGin:LATE?

Queries the late-side window margin at the user-specified margin level (see :CALC:WMAR:MLEVel for setting this value) when SIDes is set to TWO. The margin value is reported as a time value (seconds) and is essentially the time from the segment boundary to the intersection of the window margin curve with the margin level. Another quantity that is often used, bit shift, can easily be derived from the margin value by subtracting the margin value from 1/2 the segment width. These values can be converted to a percent by dividing by 1/2 the segment width and multiplying by 100%.

Query Response

Numeric data transferred as ASCII bytes in floating point <NR3> format.

Comments

- Query only.
- If SIDes is set to ONE, 0 is returned and an error is reported.
- The margin value may be based on either window margin data directly derived from the histographic data or from the extrapolation projection taken from the window margin data. You can easily determine what data was used. If the window margin data derived from the histographic data crosses the indicated margin level, then that data is the basis for the reported margin value. Extrapolation data is only used when the directly derived data does not extend to the specified margin level.
- If you want to ensure that margin is only reported based on window margin data taken directly from the histographic data, turn off extrapolation using the :CALC:WMAR:EXTR:STAT OFF command. Subsequently, if the direct window margin data does not extend to the specified margin level, 0 is returned, and an error is reported.

:CALCulate:WMARgin:MLEVel <numeric_value>

This command controls the setting of the margin level, which is the error rate level at which the margin result is determined. The value is set as log error rate. For example, setting it to -9.0 means that the margin will be evaluated at an error rate of 1 part in 10^9.

<numeric_value> Range -0.5 to -20.0

Query Response

Numeric data transferred as ASCII bytes in floating point <NR3> format.

Comments

- *RST value is -10.0.
- The setting of the margin level affects the margin results (:CALC:WMAR:MARG?, :CALC:WMAR:MARG:EARL? and :CALC:WMAR:MARG:LATE?)

:CALCulate:WMARgin:NOISe?

Queries for an estimate of the aggregate noise in the enabled data segments. This query applies to the case when SIDes is set to ONE. The histographic data from all segments is combined, and then folded about the segment center. The noise estimate is derived as part of extrapolated data projection process.

Query Response Numeric data transferred as ASCII bytes in floating point <NR3> format.

Comments

- Query only.
- If SIDes is set to TWO, 0 is returned and an error is reported.
- The NOISe? result is dependent on the extrapolation range, which is controlled by the :CALC:WMAR:EXTR:RANG:LOWer and :CALC:WMAR:EXTR:RANG:UPPer commands.

:CALCulate:WMARgin:NOISe:EARLy?

Queries for an estimate of the aggregate noise in the enabled data segments. This query applies to the case when SIDes is set to TWO, and returns the noise associated with the early side of the distribution (the data that precedes the segment center). The histographic data from all segments is combined. The noise estimate is derived as part of the extrapolation data projection process.

Query Response Numeric data transferred as ASCII bytes in floating point <NR3> format.

Comments

- Query only.
- If SIDes is set to ONE, 0 is returned and an error is reported.
- The NOISe? result is dependent on the extrapolation range, controlled by the :CALC:WMAR:EXTR:RANG:LOWer and :CALC:WMAR:EXTR:RANG:UPPer commands.

:CALCulate:WMARgin:NOISe:LATE?

Queries for an estimate of the aggregate noise in the enabled data segments. This query applies when SIDes is set to TWO, and returns the noise associate with the late side of the distribution (the data that follows the segment center). The histographic data from all the segments is combined. The noise estimate is derived as part of the extrapolation data projection process.

Query Response Numeric data transferred as ASCII bytes in floating point <NR3> format.

Comments

- Query only.
- If SIDes is set to ONE, 0 is returned and an error is reported.
- The NOISe? result is dependent on the extrapolation range controlled by the :CALC:WMAR:EXTR:RANG:LOWer and :CALC:WMAR:EXTR:RANG:UPPer commands.

:CALCulate:WMARgin:OFFSet?

Queries a value consisting of the time difference between the segment center and the mean of the histographic distribution that was used to generate the window margin data. For example, if the distribution mean is 1.3 ns to the left of the segment center (i.e., on the early side), the offset value will be -1.3 ns.

Query Response Numeric data transferred as ASCII bytes in floating point <NR3> format.

Comments

- Query only.
- The SIDes state does not affect the result. The offset is based on the two-sided distribution.

- The distribution mean is computed using data from all enabled segments, specified with the :CALC:WMAR:SEGM:ENABle command, and adjusting the data from each segment with respect to that segment's center.

:CALCulate:WMARgin:SEGMents Subtree

The :SEGMents subtree of window margin commands define the various characteristics of a segment. In brief, a segment is a region of time over which data is expected to occur. Usually, a clock in the device under test provides the basis for determining where data is expected to occur. Since, data may not be present in every location where it is possible, the expected time interval spacings for the data are at multiples of the clock period that is being used to recover the data.

:CALCulate:WMARgin:SEGMents:CENTer <numeric_value> [S]

This command defines the expected center (in time) of the first segment.

<numeric_value> Range The CENTer value can be from 0 to 10 µseconds (10E-6)

Query Response Numeric data transferred as ASCII bytes in floating point <NR3> format.

Comments *RST value is 20E-9 (20 ns).

:CALCulate:WMARgin:SEGMents:COUNt <numeric_value>

This command controls the number of segments. Segments are connected end-to-end (in time) so that the end of a given segment is in common with the start of the next segment. The first segment is always the one whose center is defined by the :CENTer command.

<integer> Range 1 to 16

Query Response Numeric data transferred as ASCII bytes in <NR1> format.

Comments *RST value is 7.

:CALCulate:WMARgin:SEGMents:ENABle <non-decimal> numeric | <numeric_value> | ALL

This command provides a way to selectively include (enable) segments in window margin computations. The parameter for making the selection works much like selecting bits in the status byte: each consecutive segment has a bit assigned to it, with an associated value equal to the power-of-2 of it's associated bit position.

Segment	Bit Position	Value to Enable the Segment	Segment	Bit Position	Value to Enable the Segment
1	0	1	9	8	256
2	1	2	10	9	512
3	2	4	11	10	1024
4	3	8	12	11	2048
5	4	16	13	12	4096
6	5	32	14	13	8192
7	6	64	15	14	16384
8	7	128	16	15	32768

Thus, to enable segments 1, 3, and 4, the value needed to enable these segments is 1+4+8=13; hence, the command would be :CALC:WMAR:SEGM:ENAB 13.

Range 0 to 65535

Query Response Numeric data transferred as ASCII bytes in <NR1> format.

Comments

- At *RST, all segments are enabled.
- For an enabled segment to actually be used in a window margin computation, that segment must also be within the defined :COUNt range. For example, if :COUNt is set to 5 (meaning that 5 segments are defined) enabling the 6th segment (:CALC:WMAR:SEGM:ENAB 32) will not cause this segment to be used in window margin computations. The COUNt value would have to be 6 or greater.
- :CALC:WMAR:SEGM:ENAB ALL is a shortcut method for enabling all of the segments.

- Hexadecimal and binary notation and can also be used with this command, which for some users may be an easier way to visualize the setting of this parameter, For example, to enable segments 1, 7, and 8 any of the following are allowed:
  - **Decimal method:** :
    :CALCulate:WMARgin:SEGMents:ENABle 193
  - **Hexadecimal method:** :
    :CALCulate:WMARgin:SEGMents:ENABle 0C1H
  - **Binary method:**
    :CALCulate:WMARgin:SEGMents:ENABle 11000001B

:CALCulate:WMARgin:SEGMents:OFFSet <numeric_value> [S]

This command provides a way to time-shift all of the defined segments by an equal amount. This can also be accomplished by changing the CENTer value, but OFFSet provides a way of tweaking the segment locations without having to move the CENTer value, which may be based on an ideal design value.

<numeric_value> Range -1E-5 to +1E-5 (i.e., ± 10 µseconds)

Query Response Numeric data transferred as ASCII bytes in floating point <NR3> format.

Comments

- *RST value is 0.0.
- Note that this parameter can be adjusted either positively or negatively. Its greatest use is with two channel measurements (data-to-clock) where an adjustment may be needed to compensate for timing differences between the two channels.

:CALCulate:WMARgin:SEGMents:WIDTh <numeric_value> [S]

This command defines the time width of each segment. All segments have the same width. This width will be equal to the data decode window width, which is related to the clock from which it is derived.

<numeric_value> Range 1E-9 to 1E-5

Query Response Numeric data transferred as ASCII bytes in floating point <NR3> format.

Comments *RST value is 10E-9.

:CALCulate:WMARgin:SIDes ONE | TWO

This command controls whether the various window margin results are based on a single (ONE) or dual-sided (TWO) computation.

Two-Sided Window Margin

When SIDes is TWO, the early (the portion of the data that occurs prior to that particular segment's window center) and late (the portion of the data that occurs after that particular segment's window center) data is kept separate, which leads to pairs of margin results associated with the early and late sides.

One-Sided Window Margin

When SIDes is ONE, the early and late sides are combined into a single distribution from which window margin results are determined.

Query Response

A sequence of ASCII-encoded bytes: ONE or TWO

Comments

- *RST state is ONE.
- Setting the SIDes parameter to ONE does not simply cause the margin result to be the worst case of the individual early and late margin results (obtained with SIDes set to TWO). The combining process that occurs when SIDes is set to ONE starts with the original histogram data, combining it by folding about the appropriate segment center. The window margin is generated from this combined data.

:CALCulate:WMARgin[:STATe] <Boolean>

This command accepts a Boolean which activates (or de-activates) the entire window margin group of commands. For example, when the state is set to ON, window margin results will automatically be computed upon acquisition of new data, whereas when the state is OFF, window margin results are computed when the result is requested.

Query Response

- Single ASCII-encoded byte, 0 or 1.
- A value of 0 indicates OFF; a value of 1 indicates ON

Comments

- *RST state is OFF.
- Note that [:STATe] is optional. Therefore, :CALC:WMAR:STAT ON and :CALC:WMAR ON are equivalent.

:CALibration Subsystem

All of the following calibrations have been performed at the factory and the results permanently stored in the TIA; therefore, you are unlikely to need to use these commands. Procedures for each calibration are provided in the Chapter 16, "Maintenance," of this guide.

:CALibration:INPut[1|2]:GAIN:EXECute

Performs a gain calibration for either Input 1 or 2.

Requires a 5 Volt DC level into the input being calibrated.

Query Response

- Single ASCII-encoded byte, 0 or 1.
- A value of 0 indicates FAIL; a value of 1 indicates PASS

:CALibration:INPut[1|2]:NDELay?

Queries the time difference between consecutive edges of input 1 that is applicable to the case where [:SENS]:TST:NDEL is ON. The result was determined from a prior :CAL:INP:NDEL:EXEC command.

Query Response

Numeric data transferred as ASCII bytes in floating point <NR3> format.

:CALibration:INPut[1|2]:NDELay:EXECute

Performs the Input 1 consecutive edge timing difference calibration that is applicable to the case where [:SENS]:TST:NDEL is ON. An external input is required for this calibration.

Query Response

- Single ASCII-encoded byte, 0 or 1.
- A value of 0 indicates FAIL; a value of 1 indicates PASS

:CALibration:INPut[1|2]:OFFSet:EXECute

Performs an offset calibration for either Input 1 or 2.

Requires grounding the input being calibrated for offset.

Query Response

- Single ASCII-encoded byte, 0 or 1.
- A value of 0 indicates FAIL; a value of 1 indicates PASS

:CALibration:INPut[1|2]:TIMing?

Queries the time difference between inputs 1 and 2, which was determined by a prior :CAL:INP:TIM:EXEC command.

Query Response Numeric data transferred as ASCII bytes in floating point <NR3> format.

Comments Query only.

:CALibration:INPut[1|2]:TIMing:EXECute

Performs the Input 1 to 2 timing difference calibration. External inputs are required.

Query Response

- Single ASCII-encoded byte, 0 or 1.
- A value of 0 indicates FAIL; a value of 1 indicates PASS

:CALibration:INTerpolator:EXECute

Performs a calibration of the timing interpolator in the TIA. Executing this calibration could improve rms resolution performance if you are going to operate the TIA in environments where the temperatures are over 10 degrees away from 25° C.

No external inputs are needed for this calibration.

Query Response

- Single ASCII-encoded byte, 0 or 1.
- A value of 0 indicates FAIL; a value of 1 indicates PASS

:CALibration:RECall

Recalls the most recently saved calibration (cal) values (see :CAL:SAVE). The only time this would change the cal values being used by the TIA would be if you had performed a calibration, but not saved it, during the current power-on session.

:CALibration:SAVE

Saves all of the calibration results to permanent flash memory in the TIA. This way, the calibration (cal) values will automatically be used on subsequent power-ons of the TIA. If you do not SAVE after executing a calibration, the new cal values will only be active as long as the TIA remains powered on. On subsequent power-ons the TIA will revert to the last saved cal values.

:CALibration:TIMebase:EXECute

Performs a calibration of the TIA's internal 10 MHz oscillator.

An external 10 MHz source is needed to Input 1.

Query Response

- Single ASCII-encoded byte, 0 or 1.
- A value of 0 indicates FAIL; a value of 1 indicates PASS

:CONFigure Subsystem

Refer to the Measurement Instructions section of this chapter for description of :CONFigure.

Device Clear

The full capability of the Device Clear IEEE 488.1 interface function is implemented in the TIA. This function allows a device to be initialized to a cleared state. The device-dependent effect of this command is described below.

In response to either the Device Clear message or the Selected Device Clear message, the TIA:

- clears the input buffer and Output Queue,
- resets the parser, execution control, and response formatter,
- clears any command that would prevent processing a *RST or other commands,
- disables the effect of a prior *OPC command, and
- terminates the holdoff action of a *WAI, *OPC?, or data query waiting for pending operation to complete.

:DIAGnostic Subsystem

The :DIAGnostics subsystems are defined in this section. Refer to Chapter 16, "Maintenance," of this guide for instructions on how to use or execute these commands.

:DIAGnostic:TEST? <numeric_value> Subtree

The commands in this subtree execute selected internal diagnostic tests with the integer value identifying the specific test.

Diagnostic Execution Notes:

1. If run via the "SCPI Control of E1740A" dialog box, set time out to long.*
2. For those diagnostics that require an input signal, not applying one will cause an error.
3. The HP E1741A "Result" buffer will hold only 220 characters. Some diagnostics have results messages longer than this. For complete results feedback, consult this section. The complete results can also be seen by running a diagnostic in the "SCPI Control of E1740" dialog box.*
4. For details on what to do in the event of a diagnostic failure, consult Chapter 16, "Maintainence," in this guide. Executing these diagnostics and making repair decisions WITHOUT reference to the troubleshooting process may lead to improper service actions, higher repair costs, and delayed repairs.

:DIAGnostic:TEST? 0

Performs a self test of the instrument. It runs all of the diagnostic tests **that require no user intervention**. The tests are:

Test 1. SRAM,
Test 2. EPROM,
Test 3. EEPROM,
Test 4. Non Volatile RAM,
Test 5. XILINX,
Test 6. Histogram,
Test 7. VRAM,
Test 8. Correction RAM,
Test 9. 79.9 MHz,
Test 10. Event Pacing Counter,
Test 11. No test assigned to this number
Test 12. Randomizer, and
Test 13. Input board Register.

* Comment applies to performing diagnostic tests with HP E1741A software.

Setup Procedure No user intervention required.

Query Response Quoted string:

"Self Test PASSED"

or

"Self Test FAILED: EEPROM HIST VRAM CORRAM MEAS79 INPUT"

Comments The fail message is an example of what will be seen. If any tests fail, they are listed in the return message.

:DIAGnostic:TEST? 1

Performs a non-destructive RAM test on the two system SRAMs (U91,U90).

Setup Procedure No user intervention required.

Query Response Quoted string:

"SRAM Test PASSED"

or

"Measurement SRAM FAILED (U90=MSB,U91=LSB): addr=FFED7A data=FFFF (AAAA)"

Comments The returned failure message shows the address, data read and data written (in parenthesis) of the first failure detected.

:DIAGnostic:TEST? 2

Computes a checksum for each of the two bootup EPROMs (U13,U14).

These are compared with the expected checksums that are stored in the last two locations of each EPROM. If the computed checksum does not match the expected checksum or if the checksum is equal to zero, the test fails.

Setup Procedure No user intervention required.

Query Response Quoted string:

"EPROM Test PASSED"

or

"Odd EPROM checksum FAILED (U14): checksum=24384E (243852)"

"Even EPROM checksum FAILED (U13): checksum=3B375A (3B3728)"

Comments The returned failure message shows which EPROM part failed it's checksum test, the checksum that was read, and the expected checksum in parenthesis.

:DIAGnostic:TEST? 3

Computes a checksum for each of the four system EEPROMs (U93,U21,U74,U28).

These are compared with the expected checksums that are stored in each EEPROM. If the computed checksum does not match the expected checksum or if the checksum is equal to zero, the test fails.

Setup Procedure No user intervention required.

Query Response Quoted string:

"EEPROM Test PASSED"

or

"Odd EEPROM 1 checksum FAILED (U93): checksum=2674EB (267335)"

"Even EEPROM 1 checksum FAILED (U21): checksum=2674EB (267335)"

"Odd EEPROM 2 checksum FAILED (U74): checksum=2674EB (267335)"

"Even EEPROM 2 checksum FAILED (U28): checksum=2674EB (267335)"

Comments The returned failure message shows which EEPROM part failed it's checksum test, the checksum that was read, and the expected checksum in parenthesis.

:DIAGnostic:TEST? 4

Tests the EEPROM that functions as Non-volatile RAM (U94). A checksum is computed for each record that is stored in the NVRAM. If any of the computed checksums do not match the expected checksum, the test fails.

Setup Procedure No user intervention required.

Query Response

Quoted string:

"NVRAM Test PASSED: last_record = 5 erases = 1"

or

"NVRAM Test FAILED: No records found (erases = 1)"

"NVRAM Test FAILED: bad recs = 5"

1st bad rec = 0 calc = 2E43 read = 34F5
last bad rec = 10 calc = 3D20 read = 3C34
last_record = 10 erases = 5

NOTE

Data will vary according to the amount of activity. The pass or fail message is the key.

Comments

If no records were found in the NVRAM, and error message indicating this is returned. If records are found that have bad checksums, a message is returned indicating how many records were bad, the calculated and read checksums for the first and last failures, and the last record number and the number of total NVRAM erasures that have occurred.

:DIAGnostic:TEST? 5

Checks to make sure that the two XILINX ICs (A1U47,U73) are programmed by checking the DONE bits from each. This is accomplished by reading an I/O port from U26.

Setup Procedure

No user intervention required.

Query Response

Quoted string:

"XILINX Test PASSED"

or

"XILINX Test FAILED: U47->[unprogrammed] U73->[unprogrammed]"

Comments

If either of the XILINX ICs are not programmed, an error is returned indicating which IC's are programmed.

:DIAGnostic:TEST? 6

Performs a test of the histogram hardware. For each of the four histogram circuits, three tests are preformed. A histogram clear operation is performed. This test fails if any histogram RAM locations are not set to zero after the clear operation.

Setup Procedure No user intervention required.

Query Response Quoted string:

"HIST1: PASSED
HIST2: PASSED
HIST3: PASSED
HIST4: PASSED"

or

"HIST1:
FAILED Clear: ...
FAILED Register Test: ...
FAILED Register Test: ...
FAILED RAM Test failures ...
FAILED RAM Test failures ..."

NOTE The FAILED message is repeated for each histogram.

Comments If any histogram circuit fails it's clear test, the address of the first location that is not cleared is returned along with the data that was read. If any of the histogram ICs register tests fail, a message is returned with the expected value in parenthesis followed by a list of read back values. If any of the histogram circuit's RAM test fails, a message is returned that lists the number of failures detected out of 2048 locations tested, the pattern that was expected followed by the address and data of the first failure.

:DIAGnostic:TEST? 7

Performs a data destructive RAM test on the eight Video RAMs that are used to store the measurement data (U42,U39,U46,U45,U43, U40,U44,U41).

Setup Procedure No user intervention required.

Query Response

Quoted string:

"VRAM Test PASSED"

or

"VRAM Test FAILED:

Data: addr=200000 data= (ffff)
Status: addr=300000 data= (ffff)"

Comments

If any address locations fail the VRAM data or status test, the address, data read and data written (in parenthesis) are reported in the failure message. Also, a list of up to 20 failures from the addresss line tests are reported. Addresses that appear to be mapped to the same location are listed in square brackets.

:DIAGnostic:TEST? 8

This test first saves away the current contents of the correction RAM. It then fills it with the value AA. Then each location is read, compared with AA and then set to 55. Then each location is read, compared with 55 and then set to 00. The contents of the RAM are then restored.

Setup Procedure

No user intervention required.

Query Response

Quoted string:

"Correction RAM PASSED"

or

"Correction RAM FAILED: addr=00 data=FF (AA) failures=1"

Comments

If any locations fail the test, the address of the first failure, data read, data expected (in parenthesis) and total number of failures are reported.

:DIAGnostic:TEST? 9

Performs a continuous histogram measurement on the internal 79.9 MHz oscillator.The measurement takes one block of 1000000 samples.

79.9 MHz is a high enough frequency where noise might cause time intervals to increase about 80 MHz. When this happens, measurements are missed and will show up as time intervals that are twice the expected value. These are expected and are ignored for the mean calculation.

Setup Procedure No user intervention required.

Query Response Quoted string:

"79.9 MHz Test PASSED:

Sdev = 32.304pS Mean = 12.565nS Min = 12.402nS Max = 12.744nS"

or

"79.9 MHz Test FAILED:

Sdev = 32.304pS Mean = 12.900nS Min = 12.402nS Max = 13.044nS"

NOTE The statistics above will vary. The pass or fail message is the key. The test fails if the measurement mean of the main lobe is more than 1% off of the expected time interval or if more than 10% of the measurements are not in this main lobe. The return message reports the standard deviation, mean, minimum, and maximum of the main lobe.

:DIAGnostic:TEST? 10

Performs a continuous histogram measurement on every fifth cycle of the internal 79.9 MHz oscillator. The measurement takes one block of 1000000 event paced samples.

Setup Procedure No user intervention required.

Query Response

Quoted string:

"Event Pacing Counter Test PASSED:

Sdev = 44.264pS Mean = 62.627nS Min = 62.451nS Max = 62.793nS"

or

"Event Pacing Counter Test FAILED:

Sdev = 93.652pS Mean = 65.534nS Min = 62.352nS Max = 67.423nS"

NOTE

The statistics above will vary. Key on the pass or fail message.The test fails if the measurement mean of the main lobe is more than 1% off of the expected time interval or if more than 10% of the measurements are not in this main lobe. Both messages return the standard deviation, mean, minimum, and maximum of the main lobe.

:DIAGnostic:TEST? 11 (NOT ASSIGNED)

:DIAGnostic:TEST? 12

Performs a continuous histogram measurement on every random cycles of the internal 79.9 MHz oscillator. The measurement takes one block of 1000000 randomly paced samples. The measurements will clump together into 6 distributions: 6,7,8,9,10 and 17 times the time interval of the input frequency.

Setup Procedure

No user intervention required.

Query Response

Quoted string:

"Randomizer Test PASSED"

or

"Randomizer Test FAILED:

Sdev = 111.589pS Mean = 75.288nS Min = 73.047nS Max = 77.148nS"

Comments

The test fails if the measurements do not distribute in the expected manner. The message returned will contain a line for each distribution expected. Each reports the standard deviation, mean, mininum and maximum for that time interval region. It also reports the proportional size of each distribution normalized to a total measurement size of 32.

The total number of measurements found is also reported.

:DIAGnostic:TEST? 13

Performs a clear test, walking ones test, set-all-bit test, 55 and AA pattern tests on the register of the input amplifier.

Setup Procedure

No user intervention required.

Query Response

Quoted string:

"Input Board Register Test PASSED"

or

"Input Board Register Test FAILED: Bits = [80]"

Comments

If the test fails, a message is returned that shows which bits failed.

:DIAGnostic:TEST? 14

Sets the trigger level DACs of channel 1 and 2 to values 1.500, -1.500, 0.015, -0.015 volts and checks the trigger light value against the expected result for an input of 0 volts. Then a timestamp measurement is taken where the sign of the trigger level is cycled 100 times. 100 events should be recorded.

Setup Procedure

Remove inputs from Channels 1 and 2.

Query Response

Quoted string:

"Trigger Level and Event Count Test PASSED"

or

"Trigger Level and Event Count Test FAILED:

Chan A: 1.500V failed

Chan A: 0.015V failed

Chan B: 0.500V failed

Chan B: 0.015V failed

Events = 0 [100]"

Comments

If the test fails, a message is returned that reports which trigger levels produced unexpected results and also returns the number of events recorded versus the number expected.

:DIAGnostic:TEST? 15

Sets up the hardware to make one continuous time stamped measurement of 10000 samples. The data is then searched to find inhibits. The inhibit signal should be at least 100 kHz.

Setup Procedure

Connect a 100 kHz TTL trigger signal to the **Inhibit Input** on the front panel of the HP E1740A.

Query Response

Quoted string:

"Inhibit Test PASSED: Measurements = 10000 Inhibits = 234"

or

"Inhibit Test FAILED: Measurements = 10000 Inhibits = 0"

NOTE

For Inhibit PASSED, number of inhibits will vary depending on inhibit frequency and timing. Any number OTHER than zero is passing.

Comments

Test fails if no inhibits are found or less than 10000 measurements are taken.

:DIAGnostic:TEST? 16

Sets up the hardware to make one externally armed continuous histogram measurement of 1000 samples. The internal 79.9 MHz signal is used. The external arm trigger level is set to TTL levels.

Setup Procedure

Setup Procedure: Connect a 100 kHz TTL trigger signal to the **Trigger Input** on the front panel of the HP E1740A.

Query Response

Quoted stringe:

"Trigger Input Test PASSED"

or

"Trigger Input Test FAILED"

Comments

If no trigger occurs, the measurement will timeout and return the failed message.

:DIAGnostic:TEST? 17

Performs three continuous histogram measurement of 1000000 samples. The first with the internal timebase, the second with the external oscillator and the third with the 10 MHz signal from the VXI backplane.

Setup Procedure

Connect 10 MHz signal to external timebase input.

Query Response

Quoted string:

"Timebase Test PASSED"

or

"Timebase Test FAILED"
Internal Timebase FAILED"
External Timebase FAILED [Unlocked]"
CLK10 Timebase FAILED [Unlocked]"

Comments

The test fails if any of the measurements do not finish or if the mean of any of the measurements are out of limits, or if the external or CLK10 timebase did not lock.

:FETCh? Subsystem

Refer to the Measurement Instructions section of this chapter for description of :FETCh? query.

:FORMat Subsystem

This subsystem sets the data format for transferring numeric and array information. This data format is used for response data by those commands that are specifically designated to be affected by the :FORMat subsystem.

:FORMat[:DATA] ASCii | REAL | INTeger

The :FORMat[:DATA] <types> selects the output format for measurement or source data (that is, in response to :FETC?,:MEAS?, :READ?, :CALC:WMARgin:DATA?, or :CALC:WMARgin:EDATa?). Valid <types> are ASCii, REAL, and INTeger.

The REAL type defaults to <float64> and the INTeger type defaults to <int16>. When ASCii formatting is selected the output stream data is buffered, whereas REAL formatting retrieves unbuffered data (yields faster throughput). In general, INT and REAL modes will provide faster output than ASC.

Query Response

A sequence of ASCII-encoded bytes: ASC, REAL, or INT

Comments

- *RST state is ASCii.
- This command affects the response format of the following commands:

  :CALCulate:WMARgin:DATA?
  :CALCulate:WMARgin:DATA:ESIDe?
  :CALCulate:WMARgin:DATA:LSIDe?
  :CALCulate:WMARgin:EDATa?
  :CALCulate:WMARgin:EDATa:ESIDe?
  :CALCulate:WMARgin:EDATa:LSIDe?
  :FETCh?
  :MEASure?
  :READ?

NOTE

Refer to Chapter 10, "Data Formats," in this guide for detailed information on the types of output format for various data retrieval.

:INITiate Subsystem

This subsystem controls the initiation of a measurement.

:INITiate[:IMMediate]

This event command causes the TIA to initiate a block of measurements.

Comments

This command is an overlapped command (see IEEE 488.2, Section 12). It is possible to send commands to the HP E1740A TIA while it is acquiring measurements. For example, you could query the "measuring" bit (bit 4) in the Status Operation Register during the measurement. Any command that changes the state of the hardware during the acquisition will abort the measurement.

Beginning a measurement or block of measurements with an :INITiate[:IMMediate] sets the Pending Operation Flag to true. Completing the measurement or block of measurements (normally or by aborting) sets Pending Operation Flag to false.

:INPut[1|2] Subsystem

This subsystem controls the characteristics of the TIA's input ports. :INPut1 corresponds to channel 1 input port and :INPut2 corresponds to channel 2 input port.

:INPut[1|2]:COUPling AC | DC

Sets or queries the coupling for the specified input.

Query Response A sequence of ASCII-encoded bytes: AC or DC

Comments *RST state is DC.

:INPut[1|2]:IMPedance <numeric_value> [OHM]

Sets or queries the input impedance for the input, specified in Ohms (50 Ω or 1 MΩ).

<numeric_value> Range 50 or 1E6

Query Response Numeric data transferred as ASCII bytes in floating point <NR3> format with six significant digits.

Comments

- *RST value is 1E6 OHM.
- Units are Ohms.

:INPut[1|2]:ROUTe COMMon | SEParate

Sets or queries the internal signal routing. It is only activated on Input 1. If COMMon is selected Input 1 acts as the signal source to both inputs (internally). If SEParate is selected the inputs become independent.

Query Response A sequence of ASCII-encoded bytes: COMM or SEP

Comments *RST state is SEParate.

:MEASure? Subsystem

Refer to the Measurement Instructions section in this chapter for a description of :MEASure? query.

Measurement Instructions (:CONFigure, :FETCh?, :MEASure?, :READ?)

The Measurement Instruction commands are structured to allow you to trade off interchangeability for fine control of the measurement process. The key commands in this group of commands are:

- CONFigure:<function> [<start_meas#>[,<# of meas>, [,<source_list]]]
- MEASure:<function>? [<start_meas#>[,<# of meas>, [,<source_list]]]

The **<function>** paramaters define the measurement operation to be used by :MEASure?, :CONFigure, :READ?, and :FETCh? instructions and are used directly in combination with these instructions.

The <function> parameters and its layers are as follows:

<function> = <presentation layer>:<fundamental measurement layer>:<measurement function layer>

The **presentation** layer for the HP E1740A are:

- XTIMe (Versus Time), or
- XTINterval (Versus Time Interval).

XTIMe specifies that the sensor functions are acquired with respect to time (that is, the X-axis is in units of time).

XTINterval specifies that the sensor functions are acquired with respect to time intervals (that is, the X-axis is in units of time intervals). The XTINterval presentation layer accommodates histogram measurements which are bin count values of successive time intervals.

The **fundamental measurement** layer specifies the fundamental signal characteristics being measured, which is [:VOLTage] in the HP E1740A.

The **measurement function** layer presents the specific function being measured. In the HP E1740A the measurement function layers are either :TINTerval, :HISTogram, or :TSTamp.

(For block measurements, the characteristic of each block can be set using the [:SENSe]:ACQuisition and :TRIGger subsystems.)

Description of the Parameters

The <function> Parameter

The <function> parameter allows selection of the TIA's measurement functions.

The allowable <function>s are:

XTIMe[:VOLTage]:TINTerval	*contiguous time interval on input1*
XTIMe[:VOLTage]:TSTamp	*contiguous timestamping on input1*
XTINterval[:VOLTage]:HISTogram	*histograms from input1 time intervals*

The [<start_meas#>[, <# of meas>]] Parameters

Use the **<start_meas #>** parameter to specify the place (the sequential location) in the measurement results from which to begin retrieving data. Zero identifies the first measurement.

This parameter exists because of the very large number of time interval measurements that can be acquired. Up to 512K single-channel measurements can be stored in the memory of the TIA. This parameter complements <# of meas> described below. "DEF" (Default) will set the <start_meas#> parameter to zero.

Use the **<# of meas>**parameter to specify the number of measurements to retrieve. This parameter complements <start_meas #> described above. "DEF" (Default) will retrieve up to 2048 measurements.

The [<start_block#>[, <# of block>]] Parameters

Use the **<start_block #>** parameter to specify the starting block from which to compute statistics results. This parameter is only valid for:

:FETCh:TINT:XXX? and :READ:TINT:XXX? subsystems, where XXX equals MAX, MEAN, MIN, or SDEV.

Use the **<# of block>** parameter to specify how many blocks should be included in the statistics computation. This parameter is only valid for:

:FETCh:TINT:XXX? and :READ:TINT:XXX? subsystems, where XXX equals MAX, MEAN, MIN, or SDEV.

Description of the <source_list>

The <source_list> represents a channel list for each measurement function. For a TI 1->2 (a dual channel measurement), the channel list is represented as (@1), (@2). For a single channel measurement, it can altogether be defaulted or represented as a single (@1) or as a dual-channel measurement (@1), (@1). The possible permutations for the source list are as follows:

(@1)	Input 1 measurement only
(@1),(@1)	Input 1 measurement only
(@1),(@2)	Input 1->2 measurement
(@2)	Input 2 measurement only
(@2),(@2)	Input 2 measurement only

The <source_list> parameter has the same syntax as SCPI <channel_list> syntax. For example, a one-channel function would use (@1) to specify channel 1, whereas a two-channel function would use (@1), (@2) to specify a measurement between channel 1 and channel 2.

If the instrument receives a parameter which is unexpected, it shall process the command, ignoring the unexpected parameter, and set the "Command Warning" bit of the Data Questionable status reporting structure.

The response format for :MEASure?, :READ?, and :FETCh? is determined by the :FORMat subsystem. If no valid data is available, error -230 (Data corrupt or stale) is generated.

:CONFigure:<function> [<parameters>[,<source_list>]]

For more programming flexibility, use the :CONFigure command. When you execute this command, the TIA is configured for the selected function. However, the acquisition is not automatically started, and you can change measurement parameters before starting the acquisition. The TIA offers a variety of low-level commands in the :INPut, [:SENSe], :CALCulate, and :TRIGger subsystems. These commands are documented in this chapter. (You can also use the [:SENSe]:FUNCtion command to change the measurement function without using :MEASure? or :CONFigure.)

Use this command when most of the settings are acceptable for your measurement. The :INPut, :TRIGger, and [:SENSe] subsystem commands should be used to set particular parameters.

:CONFigure:XTIMe:TINTerval [<start_meas#>[, <# of meas>[,source_list]]]

Configures the TIA to perform a time interval measurement.

A subsequent :READ? operation would actually perform the time interval measurement and retrieve the data. Parameters may be defaulted from right to left.

Comments

- Refer to Chapter 3, "Using the Measurement Instruction Commands," for descriptions of each measurement function.
- Refer to the section titled "Description of the <function> Parameter" for a list of each of the measurement functions.

:CONFigure:XTIMe:TSTamp [<start_meas#>[, <# of meas>[,source_list]]]

Configures the TIA to take timestamps of inputs as specified by the source lists.

A subsequent :READ? operation would actually perform the timestamping and would retrieve the measurement.

Since it is possible to restore the time interval information via the timestamps it is possible to configure TSTamp but retrieve the time interval using the following READ query: READ:XTIM[:VOLT]:TINT?

Comments

- Refer to Chapter 3, "Using the Measurement Instruction Commands," for descriptions of each measurement function.
- Refer to the section titled "Description of the <function> Parameter" for a list of each of the measurement functions.

:CONFigure:XTINterval:HISTogram [<start_meas#>[, <# of meas>[,source_list]]]

Configures the TIA to perform a histogram of the acquired time interval measurements. The defaulted form is a histogram of the time intervals on Input 1. The width of the bins are determined through the [:SENS]:TINT:RANGe:RESolution command.

A subsequenct :READ? operation would actually perform the time interval measurements and order them in the appropriate bins according to the resolution of the bins as specified by the [:SENS]:HIST:RANG:RESolution command.

Comments

- Refer to Chapter 3, "Using the Measurement Instruction Commands," for descriptions of each measurement function.
- Refer to the section titled "Description of the <function> Parameter" for a list of each of the measurement functions.

:CONFigure?

Queries the function configured by the last :CONFigure or :MEASure? query. If the instrument state has changed through commands other than :CONFigure or :MEASure? query, the instrument will track these changes, and the query response will reflect these changes.

Query Response

A string of the form: "<function> <parameters>,<source_list>", omitting the leading colon from the <function>.

Comments

- Refer to Chapter 3, "Using the Measurement Instruction Commands," for descriptions of each measurement function.
- Refer to the section titled "Description of the <function> Parameter" for a list of each of the measurement functions.

:FETCh?

This query returns the result of the already initiated and acquired measurement. The :FETCh? query will return data any time that the last reading is valid. Data becomes invalid under the following conditions:

- When a *RST is executed.
- When an INITiate is executed.
- When there is any reconfiguration of the signal routing, measurement function, signal generation, and/or trigger blocks.
- When the sensor begins acquisition of a new reading.

FETCh? can also be used in conjunction with a <function> name not necessarily **that** of the configured measurement:

FETCh[:<function>]? <parameters>

where <parameters> = [<start_meas#>[, <# of meas>[,source_list]]]

This capability allows FETCh? to return the value of the <function> derived from the acquired data of a preconfigured/preinitiated measurement. If <parameters> is omitted :FETCh? will retrieve the entire acquisition. DEFault is accepted as a parameter. The <start_meas#> parameter specifies the measurement number the data retrieval should start from, and the <# of meas> parameter specifies in numbers of measurements the number of consecutive data points to be retrieved. This is true for cases where multiple data points are available. Multiple channel measurements like Time Interval measurements from Input 1 to Input 2 can be specified by the <source_list> parameter as (@1), (@2); however, they must match those of the preconfigured measurement. Parameters may be defaulted from the right by omitting them.

In case of statistics computation, use the <start_block #> parameter to specify the starting block from which to compute the statistics results. Use the <# of block> parameter to specify how many blocks should be included in the statistics computation.

Issuing this query while a measurement is in progress has the effect of holding off further commands from being processed until the measurement completes. This hold-off action can only be canceled by the measurement completing, Device Clear, or power-on.

Query Response

- Result will be formatted according to :FORMat[:DATA] ASCii | REAL | INTeger setting. (Refer to Chapter 10, "Data Formats," for details.)
- When ASCii format is used, numeric data is transferred as ASCII bytes in <NR3> format.

Comments

- Refer to Chapter 3, "Using the Measurement Instruction Commands," for descriptions of each measurement function.
- Refer to the section titled "Description of the <function> Parameter" for a list of each of the measurement functions.

:FETCh:TINTerval:MAXimum? [<start_block#>[, <# of block>]]

Queries the maximum acquired data value.

:FETCh:TINTerval:MEAN? [<start_block#>[, <# of block>]]

Queries the mean of the acquired data.

:FETCh:TINTerval:MINimum? [<start_block#>[, <# of block>]]

Comments

<source_list> representations are as follows:
(@1) indicates Input 1
(@2) indicates Input 2
a defaulted parameter indicates Input 1

Query Response

Numeric data is transferred as ASCII bytes in <NR3> format.

:FETCh:MINimum? [<source_list>]

These query commands cause the HP E1740A to evaluate the input signal at the selected input and return the peak minimum voltage. All current input settings for the selected input will apply during the evaluation. For example, if the input impedance is set to 50 ohms, the minimum voltage evaluation will use this impedance.

Comments

<source_list> representations are as follows:
(@1) indicates Input 1
(@2) indicates Input 2
a defaulted parameter indicates Input 1

Query Response

Numeric data is transferred as ASCII bytes in <NR3> format.

:FETCh:PTPeak? [<source_list>]

These query commands cause the HP E1740A to evaluate the input signal at the selected input and return the peak-to-peak voltage. All current input settings for the selected input will apply during the evaluation. For example, if the input impedance is set to 50 ohms, the peak-to-peak voltage evaluation will use this impedance.

Comments

<source_list> representations are as follows:
(@1) indicates Input 1
(@2) indicates Input 2
a defaulted parameter indicates Input 1

Query Response

Numeric data is transferred as ASCII bytes in <NR3> format.

:FETCh:TINTerval:MAXimum? [<start_block#>[, <# of block>]]

Queries the maximum acquired data value.

:FETCh:TINTerval:MEAN? [<start_block#>[, <# of block>]]

Queries the mean of the acquired data.

:FETCh:TINTerval:MINimum? [<start_block#>[, <# of block>]]

Queries the minimum acquired data value.

:FETCh:TINTerval:SDEVation? [<start_block#>[, <# of block>]]

Queries the standard deviation of the acquired data.

:FETCh:XTIMe:FREQuency? [<start_meas#>[, <# of meas>]]

Queries the sequential frequency data for single-channel measurements.

Valid frequency results will not be obtained if the random pacing mode is selected (:ACQ:PAC RAND) or every edge pacing mode is selected (:ACQ:PAC IMM), and the continuous counting rate is exceeded by the input. An example of this would be a 100 MHz input, which exceeds the 80 MHz every edge counting rate. The way to get valid frequency readings above 80 MHz is to use every Nth edge pacing (:ACQ:PAC STEP). Any value greater than 3 will ensure that every edge is accounted for.

:FETCh:XTIMe:TINTerval? [<start_meas#>[, <# of meas>]]

Queries the block of time interval data.

:FETCh:XTIMe:TSTamp? [<start_meas#>[, <# of meas>]]

Queries the block of timestamp data.

:FETCh:XTINterval:HISTogram? [<start_meas#>[, <# of meas>]]

Queries the block of histogram data.

:MEASure:<function>? [<parameters>[,<source_list>]]

The :MEASure query is identical to sending:

:ABORt;
CONFigure:<function> [<parameters> [,<source_list>]]
READ:<function>? [<parameters> [,<source_list>]]

:MEASure:XTIMe:TINTerval? [<start_meas#>[, <# of meas>[,source_list]]]

This query selects the sequential time interval capability. It configures the hardware, initiates the measurement and puts the data in the output queue once the measurement has completed. The returned data is strictly sequential acquired time intervals. The <source_list> indicates whether a time interval from Input 1 to Input 2 (@1), (@2), a contiguous time interval measurement on Input 1 (@1), or a contiguous time interval measurement on Input 2 (@2), will take place. In the defaulted form a contiguous time interval measurement of Input 1 will occur.

Parameters (other than <source_list>) may be defaulted from the right by omitting them, or anywhere by substituting the keyword DEFault. The <source_list> parameter may be defaulted by omitting it. The default values are specified by the particular function description.

Issuing this query while a measurement is in progress will result in this query aborting the current measurement before initiating the desired measurement, and then waiting for the measurement to complete. Consequently, this has the effect of holding off further commands from being processed until the desired measurement completes. This hold-off action can only be canceled by the measurement completing, Device Clear, or power-on.

Query Response

- Result will be formatted according to :FORMat[:DATA] ASCii | REAL setting.
- When ASCii format is used, numeric data is transferred as ASCII bytes in <NR3> format.

Comments

- Refer to Chapter 3, "Using the Measurement Instruction Commands," for descriptions of each measurement function.
- Refer to the section titled "Description of the <function> Parameter" for a list of each of the measurement functions.

:MEASure:XTIMe:TSTamp? [<start_meas#>[, <# of meas>[,source_list]]]

This query selects the sequential timestamping capability. It configures the hardware, initiates the measurement and puts the data in the output queue once the measurement has completed. The returned data is strictly a timestamp representation over the time of the inputs selected by the <source _list>. The <source_list> indicates whether a timestamp Input 1 to Input 2 (@1), (@2) or a contiguous timestamping of Input 1 (@1) will take place. In the defaulted form a contiguous timestamping of Input 1 will occur.

Parameters (other than <source_list>) may be defaulted from the right by omitting them, or anywhere by substituting the keyword DEFault. The <source_list> parameter may be defaulted by omitting it. The default values are specified by the particular function description.

Issuing this query while a measurement is in progress will result in this query aborting the current measurement before initiating the desired measurement, and then waiting for the measurement to complete. Consequently, this has the effect of holding off further commands from being processed until the desired measurement completes. This hold-off action can only be canceled by the measurement completing, Device Clear, or power-on.

Query Response

- Result will be formatted according to :FORMat[:DATA] ASCii | REAL setting.
- When ASCii format is used, numeric data is transferred as ASCII bytes in <NR3> format.

Comments

- Refer to Chapter 3, "Using the Measurement Instruction Commands," for descriptions of each measurement function.
- Refer to the section titled "Description of the <function> Parameter" for a list of each of the measurement functions.

:MEASure:XTINterval:HISTogram? [<start_meas#>[, <# of meas>[,source_list]]]

This query selects the fast hardware histogram capability. It configures the hardware, initiates the measurement and puts the data in the output queue once the measurement has completed. The returned data is strictly a time bin count representation of the acquired intervals done on the fly. The <source_list> indicates

whether a time interval from Input 1 to Input 2 (@1), (@2) or a contiguous time interval measurement on Input 1 (@1) will take place. In the defaulted form a contiguous time interval measurement of Input 1 will occur.

Parameters (other than <source_list>) may be defaulted from the right by omitting them, or anywhere by substituting the keyword DEFault. The <source_list> parameter may be defaulted by omitting it. The default values are specified by the particular function description.

Issuing this query while a measurement is in progress will result in this query aborting the current measurement before initiating the desired measurement, and then waiting for the measurement to complete. Consequently, this has the effect of holding off further commands from being processed until the desired measurement completes. This hold-off action can only be canceled by the measurement completing, Device Clear, or power-on.

Query Response

- Result will be formatted according to :FORMat[:DATA] ASCii | REAL setting.
- When ASCii format is used, numeric data is transferred as ASCII bytes in <NR3> format.

Comments

- Refer to Chapter 3, "Using the Measurement Instruction Commands," for descriptions of each measurement function.
- Refer to the section titled "Description of the <function> Parameter" for a list of each of the measurement functions.

:READ[:<function>]? [<parameters>]

This command initiates a measurement of the configured measurement and return the value of the <function> derived from the acquired data. If <parameters> is omitted it is assumed to be those currently in use. DEFault is accepted as parameter.

A common application is to use this command in conjunction with a :CONFigure to provide a capability like :MEASure? in which the application programmer is allowed to provide fine adjustments to the instrument state by issuing the corresponding commands between the :CONFigure and :READ?.

:READ? can also be used in conjunction with a <function> name not necessarily that of the configured measurement

By specifying a <function>, the instrument will retrieve the requested value *derived* from the data taken by the implied :INITiate.

Issuing this query while a measurement is in progress will result in this query aborting the current measurement and idling the measurement cycle before initiating the desired measurement, and then waiting for the measurement to complete. Consequently, this has the effect of holding off further commands from being processed until the desired measurement completes. This hold-off action can only be canceled by the measurement completing, Device Clear, or power-on.

Query Response

- Result will be formatted according to :FORMat[:DATA] ASCii | REAL setting.
- When ASCii format is used, numeric data is transferred as ASCII bytes in <NR3> format.

Comments

- Refer to Chapter 3, "Using the Measurement Instruction Commands," for descriptions of each measurement function.
- Refer to the section titled "Description of the <function> Parameter" for a list of each of the measurement functions.

:READ:TINTerval:MAXimum? [<start_block#>[, <# of block>]]

Initiates the configured measurement and queries the maximum of the acquired data.

:READ:TINTerval:MEAN? [<start_block#>[, <# of block>]]

Initiates the configured measurement and queries the mean of the acquired data.

:READ:TINTerval:MINimum? [<start_block#>[, <# of block>]]

Initiates the configured measurement and queries the minimum of the acquired data.

:READ:TINTerval:SDEViation? [<start_block#>[, <# of block>]]

Initiates the configured measurement and queries the standard deviation of the acquired data.

:READ:XTIMe:FREQuency? [<start_meas#>[, <# of meas>]]

Initiates and queries the sequential frequency data for single-channel measurements.

Valid frequency results will not be obtained if the random pacing mode is selected (:ACQ:PAC RAND) or every edge pacing mode is selected (:ACQ:PAC IMM), and the continuous counting rate is exceeded by the input. An example of this would be a 100 MHz input, which exceeds the 80 MHz every edge counting rate. The way to get valid frequency readings above 80 MHz is to use every Nth edge pacing (:ACQ:PAC STEP). Any value greater than 3 will ensure that all edges are accounted for.

:READ:XTIMe:TINTerval? [<start_meas#>[, <# of meas>]]

Initiates and queries the block of time interval data.

:READ:XTIMe:TSTamp? [<start_meas#>[, <# of meas>]]

Initiates and queries the block of timestamp data.

:READ:XTINterval:HISTogram? [<start_meas#>[, <# of meas>]]

Initiates and queries the block of histogram data.

:OUTPut Subsystem

This subsystem controls the characteristics of the sourcing output port, in this case the ECLtrigger lines (VXI backplane). The TIA can source the triggering signal onto the backplane.

:OUTPut:ECLTrg[0][:STATe] <Boolean>

Controls whether or not the trigger event will be sourced onto the ECL Trigger Line 0 of the VXI backplane. If the STATe is ON, the trigger event will be sourced.

Query Response

- Single ASCII-encoded byte, 0 or 1.
- A value of 0 indicates OFF, a value of 1 indicates ON.

Comments

- *RST state is OFF (trigger event will not be sourced on ECL Trigger Line 0).
- Only one trigger line may be active at a time; therefore, if :OUTP:ECLT0[:STATe] ON is selected, :OUTP:ECLT1[:STATe] will be set to OFF (if it isn't already).
- The sourced trigger event is always a positive going edge.

:OUTPut:ECLTrg1[:STATe] <Boolean>

Controls whether or not the trigger event will be sourced onto the ECL Trigger Line 1 of the VXI backplane. If the STATe is ON, the trigger event will be sourced.

ECLtrg1 may be used in place of ECLT1.

Query Response

- Single ASCII-encoded byte, 0 or 1.
- A value of 0 indicates OFF, a value of 1 indicates ON.

Comments

- *RST is OFF (trigger event will not be sourced on ECL Trigger Line 1).
- Only one trigger line may be active at a time; therefore, if :OUTP:ECLT1[:STATe] ON is selected, :OUTP:ECLT0[:STATe] will be set to OFF (if it isn't already).
- The sourced trigger event is always a positive going edge.

:READ? Subsystem

Refer to the Measurement Instructions section of this chapter for description of :READ? query.

[:SENSe] Subsystem

The [:SENSe] subsystem deals with controls that directly affect device-specific settings of the TIA and not those related to the signal-oriented characteristics. This is an optional node.

[:SENSe]:ACQuisition Subtree

The [:SENSe]:ACQusition subtree handles the parameters that affect the duration of the acquisition.

[:SENSe]:ACQuisition:BLOCk? BSTart | INHibit, <start_#>, <# of blocks or inhibits>

Queries block position information. The returned value depends on the selected terminating parameter.

[:SENS]:ACQ:BLOCk? BST returns an array of values that identify the first measurement location in each acquisition block. For a single block acquisition (:TRIG:COUNt set to 1, which is the default), this query's usefulness is limited since the block start location will always be 1. When the measurement contains multiple acquisitions (i.e. :TRIG:COUNt set >1), this command provides a useful way to separate the data into original acquisition blocks. The most useful mode for this query is when the block acquisitions are based on time (see [:SENS]:ACQ:SOUR TIM), because in this case, the number of measurements in each acquisition block is unknown until the measurement is completed.

[:SENS]:ACQ:BLOCk? INH returns an array of values that identify the locations where inhibit was applied. The locations returned are the positions of the sampled data just prior to the occurrence of the inhibit. For more information about the inhibit feature, see the [:SENS]:INHibit commands.

Query Response

- The <start_#> parameter identifies which block should be the first one reported by the query (whether it is INH or BST).

  The <# of blocks or inhibits> parameter identifies how many block locations should be returned. The maximum size is 4096. therefore, if you have a measurement with, for example, 6000 blocks (that is, TRIG:COUN 6000), you can retrieve the first 4096 block start locations with :ACQ:BLOC? BST, 0, 4096 and the rest with :ACQ:BLOC? BST, 4096, 1904.

The same comments apply to retrieving INHibit locations.

- These queries are not needed unless you plan to use either the multiple block acquisition feature or the inhibit feature of the HP E1740A.

[:SENSe]:ACQuisition:LENGth? DATA | BSTart | INHibit

Queries for three different types of information, depending on the selected terminating parameter.

[:SENS]:ACQ:LENG? BSTart returns the number of acquisition blocks that occurred during an acquisition. This is related to the multiple trigger feature of the HP E1740A, which is selected using the :TRIG:COUNt command.

[:SENS]:ACQ:LENG? DATA returns the actual number of measured time intervals for a given acquisition.

To query the length of histogram data acquisition, use [SENSe]:HIST:COUNt?

[:SENS]:ACQ:LENG? INH returns the number of inhibited regions that occurred during a given acquisition. The inhibit feature of the HP E1740A is controlled with the [:SENS]:INHibit commands.

Query Response Numeric data transferred as ASCII bytes in <NR1> format.

Comments Query only.

[:SENSe]:ACQuisition:MCOunt <numeric value>

This command selects the number of acquired measurements when the [:SENS]:ACQ:SOUR command is set to MCOunt. For example, the command [:SENS]:ACQ:MCO 50 specifies to acquire 50 measurements.

<numeric_value> Range

		FUNCTION TYPE					
		XTIM:TST 1	XTIM:TST 1,2	XTIM:TINT 1	XTIM:TINT 1,2	XTIN:HIST 1	XTIN:HIST 1,2
# OF BLOCKS	= 1	524287	524287	520.000	258.000	1E12	1E12
	> 1	524287	524287	524287	262143	1E6	1E6

MCOunt upper limits per function type, input channel, and number of measurement blocks.

Query Response

Numeric data transferred as ASCII bytes in floating point<NR3> format.

Comments

- *RST value is 1000.0
- The upper limit of this parameter is dependent on the function, the input channel, and the number of measurement blocks selected.

  This value is reduced for sequential measurements to ensure that the product of the number of measurements (set with MCOunt) and the number of triggers per measurement (set with the :TRIG:COUNt command) is less than the limits described in the Range heading for sequential time-interval measurements.

[:SENSe]:ACQuisition:PACing[:SOURce] IMMediate | RANDom | STEP

This command controls how the samples are taken during the acquisition portion of the measurement. IMMediate specifies that every data edge should be collected, if possible. RANDom specifies that a random pacing should be used. With RANDom, after a data edge is sampled, a delay of a random number of data edges is applied before the next data edge is sampled. This mode is useful when the data input exceeds the maximum sampling rate, preventing improper weighting of the various distributions. Finally, STEP specifies that a sample should be taken every N data edges, where N is specified with the [:SENS]:ACQ:PAC:STEP command. This mode is useful for detecting a recurring edge in a repeating data pattern sequence or for creating a longer effective gate time (to improve resolution of frequency measurements).

Query Response

A sequence of ASCII-encoded bytes: IMM or RAND or STEP

Comments

- *RST state is IMMediate. The HP E1740A attempts to sample every edge.
- Note that the SOURce portion of the command is optional. For example, the [:SENS]:ACQ:PAC IMM and [SENS]:ACQ:PAC:SOUR IMM commands are functionally equivalent.

[:SENSe]:ACQuisition:PACing:STEP <numeric_value>

This command sets the number of edges to count until the next edge is sampled. This will only take effect if the [SENS]:ACQ:PAC[:SOUR] is set to STEP. For example, a value of 5 would cause every 5th data edge to be sampled.

<numeric_value> Range 2 to 16,777,215 edges in steps of 1.

Query Response Numeric data transferred as ASCII bytes in integer <NR1> format.

Comments *RST value is 2.

NOTE As the STEP value is increased, the possibility of time-aliasing is increased. Time aliasing causes the interval shown to be much shorter than it actually is. This effect happens when the time between two sampled edges exceeds the maximum time-interval value for the selected resolution. When this occurs, the counter that tracks the time between the intervals overflows and the actual time between the sampled edges is incorrectly recorded. For example, at the best resolution setting (50 ps), the maximum time interval that can be correctly measured is 3.2 µs. Depending on the incoming data rate, the selection of the STEP value could exceed this limit.

A simple example would be sampling a 10 MHz constant frequency, with a step value greater than 32. Thirty-two periods of 100 ns (period of 10 MHz) would cause a sample to occur every 3.2 µs, which is right at the maximum time-interval limit for the selected resolution. STEP values greater than this would be time-aliased. To remedy this problem, select a coarser resolution, which will provide a greater maximum time-interval limit.

[:SENSe]:ACQuisition:SOURce MCOunt | TIMer

This command selects which of two methods is used to control the duration of the measurement acquisition. MCOunt specifies that the duration will be based on a number of measurements (see [:SENS]:ACQ:MCOunt). TIMer specifies that the duration will be based on an amount of time (see the related command [:SENS]:ACQ:TIMer).

Query Response A sequence of ASCII-encoded bytes: MCO or TIM

Comments

*RST state is MCOunt (measurement acquisition duration based on the number of measurements).

[:SENSe]:ACQuisition:TIMer <numeric_value> [S]

This command selects the duration time for the measurement. This command is used when the [:SENS]:ACQ:SOUR is set to TIMer.

<numeric_value> Range

25 ns to 26 ms, in steps of 25 ns.

Query Response

Numeric data transferred as ASCII bytes in floating point <NR3> format.

Comments

*RST value is 10 µs.

[:SENSe]:EVENt [1|2] Subtree

The [:SENSe]:EVENt subtree collects together the commands used to define transitions of the signal through a specific voltage level. The events may be used to specify the start and stop of a time interval measurement.

[:SENSe]:EVENt [1|2]:HYSTeresis:RELative <numeric_value> [PCT]

This command provides the means for controlling the channel 1 and channel 2 hysteresis which is used to set the noise sensitivity. The greater the hysteresis setting, the more noise tolerant that channel will be, causing it less likely to take a sample on a noise-related crossing of the specified threshold level. However, increasing the hysteresis also decreases the sensitivity of that input, so if you only have a small signal to start with, you may not be able to use a large hysteresis setting. The hysteresis value is set as a percent, with 0 being the minimum available amount and 100 the maximum. The syntax shown above covers both channels, with EVENt 1 or EVEN1 both acceptable for channel 1 control and EVENt 2 or EVEN2 acceptable for channel 2 control.

Range

0 to 100 PCT, in 1% increments.

Query Response

Numeric data transferred as ASCII bytes in <NR1> format.

Comments

*RST value is 50 PCT.

[:SENSe]:EVENt [1|2]: LEVel <numeric_value> [V]

This command provides the means for controlling the channel 1 and 2 threshold level settings. For each input, there is the voltage level at which you want that channel to sense the input. The syntax shown above covers both channels, with EVENt1 or EVEN1 both acceptable for channel 1 control and EVENt2 or EVEN2 acceptable for channel 2 control. For example, to set the channel 2 threshold level to 2.3 volts, send the command [:SENS]:EVEN2:LEV 2.3 to the HP E1740A.

<numeric_value> Range

Both channels 1 and 2 can be set from -10 volts to +10 volts, with 2.5 mv settability.

Query Response

Numeric data transferred as ASCII bytes in floating point <NR3> format.

Comments

*RST value is 0 V for both channels 1 and 2.

[:SENSe]:EVENt [1|2]:LEVel:AUTO

This command causes the HP E1740A to evaluate the input signal at the specified input (1 or 2) and set the voltage threshold level to the midpoint between the peak signal maximum and minimum voltages. The query form causes the action just described and returns the voltage threshold level that has been set. The voltage threshold level can be manually set using the :EVENt [1|2]:LEVel <numeric_value> command. To query the current threshold level, use :EVENt [1|2] :LEVel?

Query Response

Numeric data transferred as ASCII bytes in floating point <NR3> format.

Comments

Both command and query form perform the action of auto threshold determination.

[:SENSe]:EVENt [1|2]: SLOPe POSitive | NEGative | EITHer

These commands provide the means for controlling the channel 1 and channel 2 slope settings. For each input, this is the slope direction upon which you want that channel to sense the input. The syntax shown above covers both channels, with EVENt1 or EVEN1 both acceptable for channel 1 control and EVENt2 or EVEN2 acceptable for channel 2 control. For example, to set channel 1 to sense only negative going transitions at the specified LEVel, send the command [:SENS]:EVENt1:SLOP NEG to the E1740A.

Range

Channel 1 accepts POSitive, NEGative, or EITHer. Channel 2 only accepts POSitive or NEGative.

Query Response

A sequence of ASCII-encoded bytes: POS or NEG or EITH

Comments

- *RST state is POSitive for both channels.
- The EITHer selection (allowed for channel 1 only,) causes channel 1 to sense transitions at the specified LEVel in EITHer slope direction.

[:SENSe]:FUNCtion Subtree

This subtree is the mechanism for selecting the desired function of a sense block. The <sensor_function> has the following syntax:

"<presentation_layer> <function_name> [<input_block>,<input_block>]"

The allowable <sensor_function>s are:

"XTIMe[:VOLTage]:TINTerval"	*contiguous time interval on input1*
"XTIMe[:VOLTage]:TINTerval 1"	*contiguous time interval on input1*
"XTIMe[:VOLTage]:TINTerval 1,2"	*time interval from input1 to input2*
"XTIMe[:VOLTage]:TINTerval 2"	*contiguous time interval on input2*
"XTIMe[:VOLTage]:TSTamp"	*contiguous timestamping on input1*
"XTIMe[:VOLTage]:TSTamp 1"	*contiguous timestamping on input1*
"XTIMe[:VOLTage]:TSTamp 1,2"	*timestamping from input1 to input2*
"XTIMe[:VOLTage]:TSTamp 2"	*contiguous timestamping on input2*
"XTINterval[:VOLTage]:HISTogram"	*histograms from input1 time intervals*
"XTINterval[:VOLTage]:HISTogram 1"	*histograms from input1 time intervals*
"XTINterval[:VOLTage]:HISTogram 1,2"	*histograms from TI input1 to input2*
"XTINterval[:VOLTage]:HISTogram 2"	*histograms from input2 time intervals*

Query Response

Returns the short form of the function names including the input block.

Comments

*RST state is "XTIM:TINT 1"

[:SENSe]:FUNCtion "sensor_function"

This command provides a mechanism for selecting the desired measurement function. This can also be accomplished with :CONFigure commands that are part of the Measurement Instruction commands.

The specific choices for the "sensor_function" string are:

"XTIMe:TINTerval" specifies the acquisition of sequential time interval measurements. Channel 1 is the default, but you can specify a channel for the input source. For example,

"XTIMe:TINTerval 1" explicitly specifies time interval measurements are taken from channel 1.

"XTIMe:TINTerval 1, 2" specifies that the time intervals are taken from channel 1 to channel 2, and

"XTIMe:TINTerval 2" specifies time interval measurements are taken from channel 2.

"XTIMe:TSTamp" also specifies sequential measurements, but instead of intervals (the time difference between adjacent time events), this string specifies that the time of occurrence of each sample is stored. These times are taken from a continuously running 16 bit time counter whose resolution you can control with the [:SENS]:TINT:RANG:RES command. This counter can overflow, so special processing of the timestamp data is needed if you want to convert it to intervals. The other specific sensor functions of this type are "XTIMe:TSTamp 1" (explicitly specifies time interval measurements are taken from channel 1), "XTIMe:TSTamp 1, 2" which provides timestamps for channel 1 followed by channel 2, and "XTIMe:TSTamp 2" which specifies the measurement input as channel 2.

"XTINterval:HISTogram" specifies to store measurements in a histogram. This general form will use channel 1 as the input. Other possibilities are "XTINterval:HISTogram1" (explicitly calls out channel 1 as the measurement source), "XTINterval:HISTogram 1, 2" which specifies that the time intervals are from channel 1 to channel 2, "XTINterval:HISTogram 2" which specifies the measurement input as channel 2.

Query Response Queries the short form of the function names, including the input block, "XTIM:TINT 1" for example.

Comments *RST state is "XTIM:TINT 1"

[:SENSe]:HISTogram Subtree

This subtree handles setting the range for a histogram measurement, and the logistics of histogram accumulate (STATe and CLEar).

[:SENSe]:HISTogram:ACCumulate[:STATe] <Boolean>

This command provides a way to accumulate histogram results for multiple acquisitions. The accumulation process sums the occurrences in a given bin after each update. Since the bin count is not cleared between updates, the resultant histogram continues to add each new update into its ever-growing bins. Setting the Boolean to ON will cause the histogram to accumulate after each update. Setting it to OFF will cause the histogram to clear for each new update.

Query Response

- Single ASCII-encoded byte, 0 or 1.
- A value of 0 indicates OFF; a value of 1 indicates ON

Comments

*RST state is OFF.

[:SENSe]:HISTogram:CLEar

This command can be used to clear an accumulated histogram result. It resets all of the bins to 0 occurrences. If the ACCumulate STATe is ON, the histograms will still accumulate on subsequent measurement completions after you have sent a :CLEar command.

[:SENSe:HISTogram:COUNt?

Queries the length of a histogram data acquisition.

Query Response

Numeric data transferred as ASCII bytes in floating point <NR3> format.

Comment

Query only.

[:SENSe]:HISTogram:RANGe:OFFSet <numeric_value> [S]

Sets or queries the minimum time-interval that will be included by the histogram. A simple way to think of this value is, if you are looking at a histogram in its typical orientation, this parameter controls the minimum x-axis value.

<numeric_value> Range

Can be stepped in increments equal to the resolution.

Query Response

Numeric data transferred as ASCII bytes in floating point <NR3> format.

Comments

- *RST value is 0 ns.
- This parameter may affect the RESolution parameter value. For example, if you select an OFFSet value that cannot be supported without a change of resolution, the resolution will be changed to the closest value that supports the requested OFFSet value. The key factor in determining if a different resolution needs to be used is whether the maximum time-interval contained within the histogram span can still be supported with the current RESolution. If it cannot, the RESolution value will be changed. To check if the resolution has changed, query the RESolution parameter.
- This parameter will alter the UPPer histogram parameter value. This is because these parameters are highly inter-related. Assuming the RESolution doesn't change as a result of a change in the UPPer value, then the OFFSet value will move exactly the same amount as the UPPer value.

[:SENSe]:HISTogram:RANGe:RESolution <numeric_value> [S]

Sets or queries the time resolution for histogram measurements. A simple way to think of the resolution is that it is the width of a single bin in the histogram.

<numeric_value> Range

RESolution can be set in power of two steps, starting at 48.8 ps, ending at 400 ns

Query Response

Numeric data transferred as ASCII bytes in floating point <NR3> format.

Comments

- *RST value is 48.8 ps.
- There is a coupling between the selected resolution and the histogram span. The span (the time interval range that will be histogrammed) is always 2048 times the resolution. For the *RST resolution of 48.8 ps, this leads to a histogram span of 100 ns.

[:SENSe]:HISTogram:RANGe[:UPPer]?

Queries the maximum time-interval that will be included by the histogram. A simple way to think of this value is, if you are looking at a histogram in its typical orientation, this parameter identifies the maximum x-axis value.

Query Response Numeric data transferred as ASCII bytes in floating point <NR3> format.

Comments

- *RST value is 100 ns.
- Query only.

[:SENSe]:INHibit Subtree

This subtree controls the variables that condition the inhibit signal (STATe, LEVel, ACTive).

[:SENSe]:INHibit:ACTive HIGH | LOW

Sets or queries which side of the specified LEVel will actually cause measurement inhibit to occur. For example, [:SENS]:INH:ACT LOW means that when the signal applied to the inhibit input is below the specified LEVel (see next command), measurements will be inhibited. When the signal is above this LEVel, measurements won't be inhibited.

Query Response A sequence of ASCII-encoded bytes: HIGH or LOW

Comments

- *RST state is LOW.
- After a measurement has occurred, you can get a list of the location of each inhibit activation. See [:SENS]:ACQ:BLOC? INH query.

[:SENSe]:INHibit:LEVel ECL | GND | TTL [V]

Sets or queries the voltage level at which the inhibit transition is sensed. For example, [:SENS]:INH:LEV GND means that an inhibit transition (either from inhibiting to not inhibiting or vice versa - see prior command) will occur when the signal applied to the inhibit input crosses 0V.

Query Response Numeric data transferred as ASCII bytes in floating point <NR3> format.

Comments

- *RST state is GND.
- After a measurement has occurred, you can get a list of the location of each inhibit activation. See [:SENS]:ACQ:BLOC? INH query.
- Nominal voltage for ECL transition is -1.3V.
- Nominal voltage for TTL transition is +1.5V.

[:SENSe]:INHibit[:STATe] <Boolean>

Sets or queries whether or not the inhibit feature of the HP E1740A is active. [:SENS]:INH ON means that the inhibit input on the front of the HP E1740A will be active. The inhibit input accepts an external gating signal that can selectively enable/disable acquisition of measurements in the E1740A. The specifics of the enable/disable decision are determined by the other inhibit commands.

Query Response

- Single ASCII-encoded byte, 0 or 1.
- A value of 0 indicates OFF; a value of 1 indicates ON

Comments

- *RST state is OFF.
- After a measurement has occurred, you can get a list of the location of each inhibit activation. See [:SENSe]:ACQuisition:BLOCk? INHibit query.

[:SENSe]:ROSCillator Subtree

This subtree handles the reference oscillator source selection and phase-locking status when an external reference is used.

[:SENSe]:ROSCillator:SOURce EXTernal | INTernal | CLK10

This command controls the source of the 10 MHz reference for the HP E1740A. This reference provides the basic timing reference from which all other measurement related timing is derived. INTernal specifies that the E1740A should use its internal 10 MHz reference. EXTernal specifies that the 10 MHz reference will be provided at the SMC connector on the front panel of the HP E1740A (the HP E1740A will phase-lock to this reference). CLK10 specifies that the HP E1740A should phase-lock to the 10 MHz CLK10 line on the VXI backplane, which is sourced from the Slot-0 card in the VXI card cage. Generally, the Slot-0 card can either generate CLK10 internally or accept an external reference which it then passes onto the CLK10 line.

Query Response A sequence of ASCII-encoded bytes: EXT or INT or CLK10.

Comments

- *RST state is INTernal.
- The HP E1740A can phase lock to an input within 100 ppm of 10 MHz.

[:SENSe]:ROSCillator:SOURce:STATus?

Queries the external reference sources (CLK10 or EXTernal) for phase locked status. The query is always based on the selection made from the [:SENS]:ROSC:SOUR command. For example, if the EXTernal reference input has been selected ([:SENS]:ROSC:SOUR EXT), the :ROSC:SOUR:STAT? query will check if the input applied at the external input (if any) has actually been phase locked by the HP E1740A.

Query Response

- A sequence of ASCII-encoded bytes: LOCK or UNL
- LOCK is returned if the reference is phase locked. UNL is returned if the phase is unlocked.

Comments Query only.

NOTE

It is important not to program the measurement setup parameters when there is no 10 MHz signal available to the HP E1740A because some of the parameters rely on the presence of this input to properly set the parameter value. The safest way to ensure this is to set the SOURce to INTernal before changing the measurement setup, but it is acceptable to be set to an external reference (CLK10 or EXTernal) as long as you have first verified that this reference has been phase locked by the HP E1740A.

[:SENSe]:TINTerval Subtree

This subtree controls the time interval measuring capabilities of the TIA.

[:SENSe]:TINTerval:RANGe:RESolution <numeric_value> [S]

Sets or queries time-interval resolution for a sequential measurement. A sequential measurement is one where the time-interval measurements are stored with timing relationships between each measurement maintained. This is opposed to a histogram measurement, where the time-intervals are binned by value into number of occurrences of each interval (in which case the time sequence information is lost).

<numeric_value> Range

48.8 ps to 400 ns, in power of two steps

Query Response

Numeric data transferred as ASCII bytes in floating point <NR3> format.

Comments

- *RST value is 48.8 ps.
- There is a direct relationship between the :TINT:RANG[:UPP] value and the :TINT:RANG:RES value. If one is doubled, so is the other. Thus, for any given measurement, you need to balance the need for measurement resolution with the need for time-interval measurement range.

[:SENSe]:TINTerval:RANGe[:UPPer] <numeric_value> [S]

Sets or queries the maximum time-interval that can be measured with a sequential measurement. A sequential measurement is one where the time-interval measurements are stored with the timing relationships between each measurement maintained. This is opposed

to a histogram measurement, where the time-intervals are binned by value into number of occurrences of each interval (in which case the time sequence information is lost).

<numeric_value> Range 3.2 µs to 26 ms, in power of two steps

Query Response Numeric data transferred as ASCII bytes in floating point <NR3> format.

Comments

- *RST value is 3.2 µs.
- There is a direct relationship between the :TINT:RANG[:UPP] value and the :TINT:RANG:RES value. If one is doubled, so is the other. Thus, for any given measurement, you need to balance the need for measurement resolution with the need for time-interval measurement range.

[:SENSe]:TSTamp Subtree

This subtree handles the timestamping related areas such as the state of the external trigger and selecting time interval measurements on Input 1 of less than 12.5 ns.

[:SENSe]:TSTamp:ETRigger[:STATe] <Boolean>

Controls whether or not the external trigger event is timestamped as part of a sequential data acquisition. When :ETRigger is ON, each block of data will begin with the timestamp for the external trigger, followed by the data timestamps for the block of data. The number of blocks of data is controlled by the TRIG:COUN <numeric_value> command.

Query Response

- Single ASCII-encoded byte, 0 or 1.
- A value of 0 indicates OFF; a value of 1 indicates ON.

Comments

- *RST state is OFF.
- External trigger is enabled with the :TRIG:SOUR EXT command.
- The external trigger timestamp has the same resolution and maximum time interval range as the rest of the data. If the maximum time interval is exceeded between the external trigger timestamp and the following data timestamp, the interval

between these events will be erroneously measured. Remedy this by selecting a longer maximum time interval ([:SENS]:TINT:RANG[:UPP] <numeric_value>).

[:SENSe]:TSTamp:NDELay[:STATe] <Boolean>

This command is used to select a special measurement mode that allows intervals on channel 1 to be measured below 12.5 ns. This mode provides measurement down to 8 ns for a pair of consecutive events on channel 1. The reason you wouldn't always select this mode is that, following the interval less than 12.5 ns, about 30 ns are required to get ready for the next one. Contrast this with [:SENS]:TST:NDEL OFF, where every edge can be measured if the spacings are 12.5 ns or greater. The real value of this mode is that it provides the only way to obtain single-channel, consecutive event measurement below 12.5 ns.

Query Response

- Single ASCII-encoded byte, 0 or 1.
- A value of 0 indicates OFF; a value of 1 indicates ON.

Comments

- *RST state is OFF.
- If you read data back as timestamps (as opposed to intervals) when [:SENS]:TST:NDEL is ON, you must apply a calibration correction to every second timestamp. You can query for the calibration value with the :CAL:INP:NDEL? command. After you've converted the timestamps to time values (multiply by resolution and correct for overflow, if any), subtract the calibration value from every second timestamp. If you read the data back as intervals, no corrections are needed.

:STATus Subsystem

The :STATus subsystem commands allow you to specify or examine the status of the Operation Status Register group and the Questionable Data/Signal Register group.

:STATus:OPERation Subtree

This subtree commands allow you to examine the status of the TIA monitored by the Operation Status Register group, shown in Figure 13-1. The Operation Status Register group consists of a condition register, an event register, and an enable register. The commands in this subtree allow you to control and monitor these registers.

See the section titled "Operation Status Register Group and Questionable Data/Signal Status Register Group" on page 11-11 in Chapter 11 for a detailed description of the Operation Status Register Group.

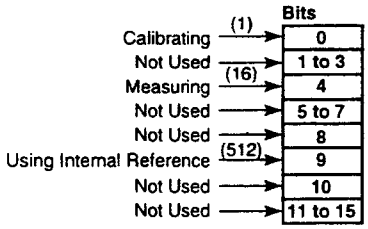


Figure 13-1. The Operation Status Register Group

:STATus:OPERation:CONDition?

Queries the status of the Operation Condition Status Register.

Bits are not cleared when read.

Query Response

- Numeric data transferred as ASCII bytes in <NR1> format.
- Range is 0 to 65,535.
- The query response value is an integer formed by the binary-weighting of the bits. The value of unused bits is zero.

Comments

The Operation Condition Status Register is cleared at power-on.

:STATus:OPERation:ENABle <non-decimal numeric> | <NRf>

Sets or queries the Operation Event Status Enable Register.

The parameter and query response value, when rounded to an integer value and expressed in base 2 (binary), represents the bit values of the Operation Event Status Enable Register. The value of unused bits is zero when queried and ignored when set.

This register is used to enable a single or inclusive OR group of Operation Event Status Register events to be summarized in the Status Byte Register (bit 7).

Range

The range for the <non-decimal numeric> or <NRf> parameter is 0 to 65,535.

Query Response

Numeric data transferred as ASCII bytes in <NR1> format.

Comments

At power-on, the Operation Event Status Enable Register is cleared (value is 0).

:STATus:OPERation[:EVENt]?

Queries the status of the Operation Event Status Register.

The Operation Event Status Register captures changes in conditions by having each event bit correspond to a specific condition bit in the Operation Condition Status Register. An event becomes TRUE when the associated condition makes the transition specified by the transition filters. The event bits, once set, are "sticky." That is, they cannot be cleared, even if they do not reflect the current status of a related condition, until they are read.

Query Response

- Numeric data transferred as ASCII bytes in <NR1> format.
- Range is 0 to 65,535.
- The query response value is an integer formed by the binary-weighting of bits. The value of unused bits is zero.

Comment

The Operation Event Status Register is cleared by *CLS, by :STAT:OPER[:EVEN]?, and at power-on.

:STATus:QUEStionable Subtree

The :STATus:QUEStionable subtree commands allow you to examine the status of the TIA monitored by the Questionable Data/Signal Status Register group, shown in Figure 13-2. The Questionable Status group consists of a condition register, an event register, and an enable register. The commands in this subtree allow you to control and monitor these registers.

See the section titled "Operation Status Register Group and Questionable Data/Signal Status Register Group" on page 11-11 in Chapter 11 for a detailed description of the Questionable Data/Signal Status Register Group.

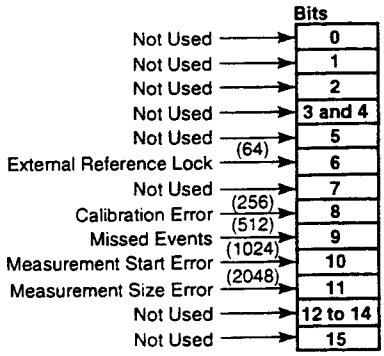


Figure 13-2. The Questionable Data/Signal Status Register Group

:STATus:QUEStionable:CONDition?

Queries the status of the Questionable Data Condition Status Register.

Bits are not cleared when read.

Query Response

- Numeric data transferred as ASCII bytes in <NR1> format.
- Range is 0 to 65,535.
- The query response value is an integer formed by the binary-weighting of the bits. The value of unused bits is zero.

Comments

The Questionable Data Condition Status Register is cleared at power-on.

:STATus:QUEStionable:ENABle <non-decimal numeric> | <NRf>

Sets or queries the Questionable Data Event Status Enable Register.

The parameter and query response value, when rounded to an integer value and expressed in base 2 (binary), represents the bit values of the Questionable Data Event Status Enable Register. The value of unused bits is zero when queried and ignored when set.

This register is used to enable a single or inclusive OR group of Questionable Data Event Status Register events to be summarized in the Status Byte Register (bit 3).

Range

The range of the <non-decimal numeric> or <NRf> parameter is 0 to 65,535.

Query Response

Numeric data transferred as ASCII bytes in <NR1> format.

Comments

At power-on, the Questionable Data Event Status Enable Register is cleared (value is 0).

:STATus:QUEStionable[:EVENt]?

Queries the status of the Questionable Data Event Status Register.

The Questionable Data Event Status Register captures changes in conditions by having each event bit correspond to a specific condition bit in the Questionable Data Condition Status Register. An event becomes TRUE when the associated condition makes a transition from 0 to 1. The event bits, once set, are "sticky." That is, they cannot be cleared, even if they do not reflect the current status of a related condition, until they are read.

The Questionable Data Event Status Register is cleared by *CLS, by :STAT:QUES[:EVEN]?, and at power-on.

Query Response

- Numeric data transferred as ASCII bytes in <NR1> format.
- Range is 0 to 65,535.
- The query response value is an integer formed by the binary-weighting of bits. The value of unused bits is zero.

:SYSTem Subsystem

This subsystem collects together the capabilities that are not related to instrument performance.

:SYSTem:ERRor?

Queries the oldest error in the Error Queue and removes that error from the queue (first in, first out).

See the Appendix in this guide for detailed error information.

Query Response

- <NR1>, string quoted: <error_number>,"<error_description>"
- The <error_number> is an integer in the range [-32768, 32767]. The negative error numbers are defined by the SCPI standard; positive error numbers are particular to this TIA. An error number value of zero indicates that the Error Queue is empty.
- The maximum length of the <error_description> is 255 characters.

Comments

- The queue is cleared (emptied) on *CLS, power-on, or upon reading the last error from queue.
- If the Error Queue overflows, the last error in the queue is replaced with the error -350, "Queue overflow." Any time the queue overflows, the least recent errors remain in the queue and the most recent error is discarded. The maximum length of the Error Queue is 30.
- This query clears the Error LED on the TIA front panel if the Error Queue has become empty as a result of the query.
- The Error Queue is unaffected by *RST.

:SYSTem:PIMacro <string>

The Purge Immediate Macro event deletes the macro described by the string name. If the string is not defined, error **-270, "Macro error"**, will be returned

Use the *PMC command to delete all macros.

:SYSTem:VERSion?

Queries the SCPI version number with which the TIA complies.

Query Response

- Numeric data transferred as ASCII bytes in <NR2> format.
- The response is an <NR2> formatted numeric value which has the form YYYY.V, where YYYY represents the year (1993) and the V represents the approved version for that year (0).

Comments

The instrument complies with SCPI Standard 1992.0 and returns this value as the response to this query.

:TRIGger Subsystem

The trigger subsystem is used to synchronize device actions with events. The TRIGger model presents an independent level of event detection that needs to be satisfied before the measurement acquisition can begin. For the triggered condition to be satisfied, either an event on the external source needs to be detected followed by the appropriate delay or no waiting for an event occurs.

SEQUENCE OF EXECUTION. Upon initiation the triggering condition needs to be satisfied. The TIA can be triggered immediately or via an external source. The trigger source can be selected using the :TRIG:SOURce command. Once triggered the TIA proceeds to satisfy any pre-acquisition requirements. Namely, it can delay acquisition for a certain amount of time :TRIG:DEL:TIM<time> or for a certain number of events on Input 1 :TRIG:DEL:ECO <#events>. The selection of the source of the delay is done via the :TRIG:DEL:SOUR <TIM | INP> command.

:TRIGger:COUNt <numeric_value>

Set or queries the number of acquisition blocks in the measurement. Each block is initiated by a trigger (defined by :TRIG:SOUR command). When COUNt is set >1, there is a gap (<500 ns) at the end of each block while the hardware resets so the next trigger can be accepted.

<numeric_value> Range

- 1 to 1,048,575 in steps of 1 for fast histogram measurements.
- 1 to a maximum of 262,144 for sequential measurements.

For sequential measurements (:CONF:XTIM:TINT or :CONF:XTIM:TST), the upper bound is dependent on the number of measurements in each block since the total number of measurements must always be less than or equal to 524,287 for sequential measurements. Thus, if the measurement block is set to 500,000 measurements, :TRIG:COUN must be 1.

Query Response Numeric data transferred as ASCII bytes in <NR1> format.

Comments

- *RST value is 1.
- Setting COUNt > 1 is useful when you have a sequence of triggers that you want to capture in a single hardware sweep.
- When COUNt is > 1, you can read the starting location of each triggered data block with the [:SENS]:ACQ:BLOC? BSTart query.

:TRIGger:DELay:ECOunt <numeric_value>

Sets or queries the value for the number of input events to delay from the trigger event until the start of the data acquisition portion of the measurement. The counted input event is always Input 1 in a two channel measurement and is the measured input in a single channel measurement (could be Input 1 or 2). The signal conditions for the counted event are the same as for the measured event (i.e., slope, trigger level are the same). For this parameter to be active, the :TRIG:DEL:SOURce must be set to INPut.

<numeric_value> Range 0 to 1,048,575 in steps of 1

Query Response Numeric data transferred as ASCII bytes in <NR1> format.

Comments *RST value is 1.

:TRIGger:DELay:SOURce INPut | TIMer | IMMediate

Sets or queries the method for delay following the trigger event.

The INPut parameter states that a counted number of input events will determine the delay (see :TRIG:DEL:ECOunt command).

The TIMer parameter indicates that the delay from the trigger event should be an amount of time (see :TRIG:DEL:TIMer command).

The IMMediate parameter states that the delay from the trigger event should be as short as possible.

Query Response A sequence of ASCII-encoded bytes: INP, TIM, or IMM

Comments *RST state is IMMediate.

:TRIGger:DELay:TIMer <numeric_value> [S]

This command controls the amount of time delay from the trigger event until the data acquisition phase of the measurement. For this parameter to be active, the :TRIG:DEL:SOURce must be set to TIMer.

<numeric_value> Range 75 ns to 26.214 ms, in increments of 25 ns.

Query Response Numeric data transferred as ASCII bytes in floating point <NR3> format.

Comments *RST value is 100 ns.

:TRIGger:LEVel <numeric_value > | TTL | ECL | TTL10 | ECL10 | GND [V]

Sets or queries the voltage level at which the external trigger (input on front panel or faceplate of the HP E1740A) will respond. TTL10 and ECL10 represent appropriate levels for these logic families when a times 10 probe is being used. The numeric value specified will correspond to only one of the five discrete values: TTL, ECL, TTL10, ECL10 and GND.

Query Response Numeric data transferred as ASCII bytes in floating point <NR3> format.

Comments

- *RST state is TTL.
- This parameter is active only when the :TRIG:SOURce state is EXTernal.

:TRIGger:SLOPe POSitive | NEGative

Sets or queries the slope direction that will cause the trigger event to occur when the :TRIG:SOUR state is anything other than IMMediate. A POSitive slope means that the trigger will occur when the signal crosses the threshold level (set via :TRIG:LEVel command for the case when the SOURce is the front panel input) from a lower to a higher voltage.

Query Response A sequence of ASCII-encoded bytes: POS or NEG

Comments *RST state is POSitive.

:TRIGger:SOURce EXTernal | IMMediate | ECLT0 | ECLTRG0 | ECLT1 | ECLTRG1

Sets or queries the source of the trigger.

EXTernal specifies that the trigger input on the front panel of the HP E1740A will be the trigger source.

IMMediate means, in effect, that no trigger source is required. The HP E1740A will immediately trigger when a measurement is initiated.

ECLT0 and ECLTRG0 are equivalent commands. They specify that the trigger source will be the ECL trigger bus 0 line on the VXI backplane. Similarly, ECLT1 and ECLTRG1 are equivalent commands, that specify the trigger source to be the ECL trigger bus 1 line on the VXI backplane. For the VXI backplane trigger sources and faceplate source (EXTernal), the slope can be set with the :TRIG:SLOPe command.

Query Response A sequence of ASCII-encoded bytes: EXT, IMM, ECLTRG0, or ECLTRG1.

Comments *RST state is IMMediate.

14

Error Messages

Introduction

This chapter explains how to read any errors from the TIA, discusses the types of errors, and provides a table of all of the TIA's errors and their probable causes.

Reading an Error

Executing the :SYSTem:ERRor? command reads the oldest error from the error queue and erases that error from the queue. The :SYST:ERR? response has the form:

<error number>, <error string>

An example response is:

-113,"Undefined header"

Positive error numbers are specific to the TIA. Negative error numbers are command language related and are discussed later in this chapter.

All errors set a corresponding bit in the Standard Event Status Register (see the section titled "Standard Event Status Register Group" on page 11-8 of Chapter 11).

The following short program reads all errors (one at a time, oldest to newest) from the error queue. After each error is read, it is automatically erased from the error queue. When the error queue is empty (that is, all errors have been read from the queue), further queries return the **+0,"No error"** response.

```
10  ASSIGN @Tia TO 70906
20  !Assign path name
30  DIM Err_string$[255]
40  !Creates array for error string
50  REPEAT
60  !Repeats until error queue is empty
70   OUTPUT @Tia;"SYST:ERR?"
80   !Read error number and string
90   ENTER @Tia;Err_num,Err_string$
100          !Enter error number and string
110          PRINT Err_num,Err_string$
120          !Print error number and string
130 UNTIL Err_num = 0
140 END
```

Error Queue

As errors are detected, they are placed in an error queue. This queue is first in, first out. That is, if there has been more than one error, the first one in the queue is read out with :SYST:ERR?. Subsequent responses continue until the queue is empty.

If the error queue overflows, the last error in the queue is replaced with error **–350, "Queue overflow"**. Any time the queue overflows, the least recent errors remain in the queue, and the most recent error is discarded. The length of the TIA's error queue is 30 (29 positions for the error messages, and 1 position for the "Queue overflow" error). Reading an error from the head of the queue removes that error from the queue, and opens a position at the tail of the queue for a new error, if one is subsequently detected.

When all errors have been read from the queue, further error queries return **+0, "No error"**.

The error queue is cleared when any of the following occur:

- Upon power-on.
- Upon receipt of a *CLS command.
- Upon reading the last item from the queue.

Error Types

Error numbers are categorized by type as shown in Table 14-1. Each and every error is listed in Table 14-2.

Table 14-1. Error Types

Error Number	Error Type
+0	No Error
–100 to –199	Command Errors
–200 to –299	Execution Errors
–300 to –350	Device-Specific Errors
–400 to –499	Query Errors
+2000 to +2019	TIA-Specific Errors

The first error described in each class (for example, -100, -200, -300, -400) is a "generic" error.

No Error

The :SYST:ERR? response **+0, "No error"** indicates that the TIA has no errors. The error queue is empty when every error in the queue has been read (:SYST:ERR? query) or the queue was cleared by power-on or *CLS.

Command Error

An <error number> in the range [–100 to –199] indicates that an IEEE 488.2 syntax error has been detected by the TIA's parser. The occurrence of any error in this class causes the command error bit (bit 5) in the Event Status Register to be set. One of the following events has occurred:

- An IEEE 488.2 syntax error has been detected by the parser. That is, a controller-to-TIA message was received that is in violation of the IEEE 488.2 Standard. Possible violations include a data

element that violates the TIA listening formats or whose type is unacceptable to the TIA.

- An unrecognized header was received. Unrecognized headers include incorrect TIA-specific headers and incorrect or unimplemented IEEE 488.2 Common Commands.
- A Group Execute Trigger (GET) was entered into the input buffer inside of an IEEE 488.2 program message.

Events that generate command errors do not generate execution errors, device-specific errors, or query errors.

Execution Error

An <error number> in the range [–200 to –299] indicates that an error has been detected by the TIA's execution control block. The occurrence of any error in this class causes the execution error bit (bit 4) in the Event Status Register to be set. One of the following events has occurred:

- A <PROGRAM DATA> element following a header was evaluated by the TIA as outside of its legal input range or is otherwise inconsistent with the TIA's capabilities.
- A valid program message could not be properly executed due to some TIA condition.

Execution errors are reported by the TIA after rounding and expression evaluation operations have been taken place. Rounding a numeric data element, for example, is not reported as an execution error. Events that generate execution errors do not generate command errors, device-specific errors, or query errors.

Device- or TIA-Specific Error

An <error number> in the range [–300 to –399] or [+1 to +32767] indicates that the TIA has detected an error that is not a command error, a query error, or an execution error; some TIA operations did not properly complete, possibly due to an abnormal hardware or firmware condition. These codes are also used for self-test response errors. The occurrence of any error in this class causes the device-specific error bit (bit 3) in the Event Status Register to be set.

Query Error

An <error number> in the range [–400 to –499] indicates that the output queue control of the TIA has detected a problem with the message exchange protocol. The occurrence of any error in this class should cause the query error bit (bit 2) in the Event Status Register to be set. One of the following is true:

- An attempt is being made to read data from the output queue when no output is either present or pending.
- Data in the output queue has been lost.

Table 14-2. Errors

Number	Error String	Cause
+0	No error	The error queue is empty. Every error in the queue has been read (:SYSTem:ERRor? query) or the queue was cleared by power-on or *CLS.
-100	Command error	This is the generic syntax error used if the TIA cannot detect more specific errors.
-101	Invalid character	A syntactic element contains a character that is invalid for that type. For example, a header containing an ampersand, :INP:COUP& AC.
-102	Syntax error	An unrecognized command or data type was entered.
-103	Invalid separator	The parser was expecting a separator and entered an illegal character.
-104	Data type error	The parser recognized a data element different than one allowed. For example, numeric or string data was expected, but block data was received.
-105	GET not allowed	A Group Execute Trigger was received within a program message.
-108	Parameter not allowed	More parameters were received than expected for the header.
-109	Missing parameter	Fewer parameters were received than required for the header.
-112	Program mnemonic too long	The header or character data element contains more than twelve characters.
-113	Undefined header	The header is syntactically correct, but it is undefined for the TIA. For example, *XYZ is not defined for the TIA.
-120	Numeric data error	This error, as well as errors -121 through -129, are generated when parsing a data element which appears to be numeric, including the non-decimal numeric types. This particular error message is used when the TIA cannot detect a more specific error.
-121	Invalid character in number	An invalid character for the data type being parsed was entered. For example, a "9" in octal data.
-123	Exponent too large	Numeric overflow.
-124	Too many digits	The mantissa of a decimal numeric data element contained more than 255 digits excluding leading zeros.
-128	Numeric data not allowed	A legal numeric data element was received, but the TIA does not accept one in this position for the header.
-131	Invalid suffix	The suffix does not follow the syntax described in IEEE 488.2 or the suffix is inappropriate for the TIA.
-134	Suffix too long	The suffix contained more than 12 characters.
-138	Suffix not allowed	A suffix was entered after a numeric element that does not allow suffixes.
-141	Invalid character data	The character data element contains an invalid character.
-148	Character data not allowed	A legal character data element was entered where prohibited by the TIA.
-150	String data error	This error can be generated when parsing a string data element. This particular error message is used if the TIA cannot detect a more specific error.

Table 14-2. Errors (Continued)

Number	Error String	Cause
-151	Invalid string data	A string data element was expected but was invalid for some reason. For example, an END message was received before the terminal quote character.
-158	String data not allowed	A string data element was entered but was not allowed by the TIA at this point in parsing.
-160	Block data error	This error can be generated when parsing a block data element. This particular error message is used if the TIA cannot detect a more specific error.
-161	Invalid block data	A block data element was expected, but it was not allowed by the TIA at this point in parsing.
-168	Block data not allowed	A legal block data element was entered but was not allowed by the TIA at this point in parsing.
-170	Expression error	This error can be generated when parsing an expression data element. It is used if the TIA cannot detect a more specific error.
-171	Invalid expression	The expression data element was invalid (see IEEE 488.2). For example, unmatched parentheses or an illegal character.
-178	Expression data not allowed	Expression data was entered but was not allowed by the TIA at this point in parsing.
-181	Invalid outside macro definition	Indicates that a macro parameter placeholder ($<number>) was entered outside of a macro definition.
-183	Invalid inside macro definition	Indicates that the program message unit sequence, sent with a *DMC command, is syntactically invalid.
-200	Execution error	This is the generic syntax error if the TIA cannot detect more specific errors. This code indicates only that an Execution Error has occurred.
-210	Trigger error	Used if the TIA cannot detect a more specific error from the :INIT, :TRIG, or :ABOR subsystems.
-211	Trigger ignored	Indicates that a GET or *TRG was received and recognized by the TIA but was ignored.
-213	Init ignored	Indicates that a request for a measurement initiation was ignored as another measurement was in progress.
-220	Parameter error	Indicates that a program data element related error occurred. This error is used when the TIA cannot detect more specific errors.
-221	Settings conflict	Indicates that a legal program data element was parsed but could not be executed due to the current TIA state.
-222	Data out of range	Indicates that a legal program data element was parsed but could not be executed because the interpreted value is outside the legal range defined by the TIA. Typically, the value is clipped to legal limit.
-223	Too much data	Indicates that a legal program data element of block, expression, or string type was received that contained more data than the TIA could handle due to memory or related TIA-specific requirements.

Table 14-2. Errors (Continued)

Number	Error String	Cause
-224	Illegal parameter value	Used where exact value, from a list of possible values, was expected.
-230	Data corrupt or stale	No valid data available. New measurement started but not completed.
-240	Hardware error	Indicates that a legal program command or query could not be executed because of a hardware problem in the TIA.
-241	Hardware missing	Indicates that a legal program command or query could not be executed because of missing TIA hardware.
-272	Macro execution error	Indicates that a syntactically legal macro program data sequence could not be executed due to some error in the macro definition.
-273	Illegal macro label	Indicates that the macro label defined in the *DMC command was a legal string syntax, but it could not be accepted by the TIA. For example, the label was too long, the same as a common command header, or contained invalid header syntax.
-276	Macro recursion error	Indicates that a syntactically legal macro program data sequence could not be executed because the TIA found the maximum recursion level of four was exceeded.
-277	Macro redefinition not allowed	Indicates that a syntactically legal macro label in the *DMC command could not be executed because the macro label was already defined (see IEEE 488.2).
-278	Macro header not found	Indicates that a syntactically legal macro label in the *GMC? query could not be executed because the header was not previously defined.
-300	Device-specific error	This is the generic device-dependent error.
-310	System error	Indicates that a system error occurred.
-321	Out of memory	Indicates that the TIA has detected that insufficient memory is available. For example, this error will eventually occur on a *DMC, once the macro memory is filled with previously defined macros.
-330	Self-test failed	Indicates at least one failure occurred when *TST? was executed.
-331	Self-test failed; CPU kernal failure	Power-on self test detected this hardware failure.
-332	Self-test failed; ROM checksum failure	Power-on self test detected this hardware failure.
-333	Self-test failed; RAM address lines failure	Power-on self test detected this hardware failure.
-334	Self-test failed; xilinx FPGA not programmed	Power-on self test detected this hardware failure.
-335	Self-test failed; NVRAM address lines failure	Power-on self test detected this hardware failure.
-350	Queue overflow	Indicates that there is no room in the error queue and an error occurred but was not recorded.
-400	Query error	This is the generic query error.
-410	Query INTERRUPTED	Indicates that a condition causing an INTERRUPTED Query error occurred. For example, a query followed by any command before a response was completely sent.
-420	Query UNTERMINATED	Indicates that a condition causing an UNTERMINATED Query error occurred. For example, the TIA was addressed to talk and an incomplete program message was received.
-430	Query DEADLOCKED	Indicates that a condition causing a DEADLOCKED Query error occurred. For example, both input buffer and output buffer are full and the TIA cannot continue.
-440	Query UNTERMINATED after indefinite response	Indicates that a query was received in the same program message after a query requesting an indefinite response (for example, *IDN?) was executed.

Table 14-2. Errors (Continued)

Number	Error String	Cause
+2000	Offset calibration on A failed	:CALibration:INP:OFFS:EXEC failed.
+2001	Offset calibration on B failed	:CALibration:INP2:OFFS:EXEC failed.
+2002	Gain calibration on A failed	:CALibration:INP:GAIN:EXEC failed.
+2003	Gain calibration on B failed	:CALibration:INP2:GAIN:EXEC failed.
+2004	Interpolator calibration failed	:CALibration:INTerpolator:EXEC failed.
+2005	Oscillator error	:CALibration:ROSCillator:EXEC failed.
+2012	Format not allowed for measurement type	The format specified upon data retrieval was not appropriate for measurement type.
+2013	Insufficient data between extrapolation levels	Less than two data points were included between the user-designated extrapolation levels. (If two-sided, at least two points must be included on each side.)
+2014	No histogram data included in segments	The enabled segments did not include any histogram data (the data was all zeros).
+2015	Segment did not entirely lie within histogram data	At least one user-enabled segment was outside of the range of the histogram data, and was not included in the calculation. This is a warning message only—the results were nevertheless calculated using the remaining segments.
+2016	Margin level not in the range of the data	The user-specified margin level is not in the range of the data. This message should only be generated if extrapolation has been disabled.
+2017	Bin size of raw histogram is zero	The bin size of the raw histogram array is zero.
+2018	Noise or margin could not be calculated	Based on the data and setup provided, the noise or margin could not be calculated. This can occur, for example, if the apparent noise is infinite.
+2019	Side mismatch	The called function requirres two-sided data when analysis is set to one-sided, or vice versa.

15

Programming Examples

Chapter Contents

This chapter provides information and examples that will show you how to program the HP E1740A to make many common measurements. This chapter is organized as follows:

Introduction

In this chapter, you will see how to program the HP E1740A to make many common measurements. Examples are provided in the following programming languages:

- HP BASIC
- Borland® Turbo C*

Using HP BASIC

This guide uses doubles quotes to enclose string parameters in syntax descriptions, but uses single quotes in the HP BASIC programming examples for readability.

The TIA allows string parameters to be enclosed by either double or single quotes. Each method is discussed in the following sub-sections.

To Send a Double-Quoted String

For the HP BASIC OUTPUT statements, remember that strings enclosed in double quotes need special consideration. For example, send the FUNC "XTIM:TINT" command with the following:

```
OUTPUT 70906;"FUNC ""XTIM:TINT"""
```

Note that a pair of double quotes (as shown in bold) is required by HP BASIC to embed a quoted string within an HP BASIC string.

To Send a Single-Quoted String

For more readable HP BASIC OUTPUT statements, you may send, for example, the following:

```
OUTPUT 70906;"FUNC 'XTIM:TINT'"
```

Note the pair of single quotes (as shown in bold) is more readable.

* Turbo C is a product of Borland International, Inc.

Using Turbo C

The Turbo C examples assume you have an HP 82335A HP-IB Interface card inside your IBM PC or compatible.

List of the Programming Examples

The following examples are provided:

NOTE

All programming examples use the ASCII format to transfer data from the TIA to the computer. The ASCII format is the default format when *RST is used.

Simple Programming Examples

1. Programs Using ASCII Format

2. Programs Using REAL Format

3. Programs Using INTEGER Format

Advanced Programming Examples

Programming Examples

Simple Programming Examples

The eight programs, written in HP BASIC, in this section demonstrate the simplest ways to make and read sequential interval, histogram, and sequential timestamp measurements in ASCII, REAL, and INTEGER formats. The focus is on simplicity rather than advanced programming techniques.

A couple of general comments may be helpful:

- REAL or INTEGER formats will provide faster data transfer than ASCII.
- The HP E1740A is shown to be addressed at 70906. The 7 identifies the interface card select code, the 09 is the address of the VXI Slot 0 module (for HP E1406A, 9 is the factory setting), and 06 is the address of the HP E1740A TIA (6 is the factory setting).

Programming Examples

Programs Using ASCII Format

Sequential Time Interval Example

```
10   !
20   ! Easiest way to make a sequential time interval measurement
30   ! and read the data from the E1740A in ASCII mode. Program
40   ! also shows how to get a computed result, in this case the
50   ! mean of the time interval values.  You need to provide an
60   ! input to channel 1 of the E1740A (something in the 1-80 MHz
70   ! range would be fine).
80   !
90   REAL Tia_data(1:1000)  ! Define an array for the data
100    OUTPUT 70906;"*RST;*CLS" ! Put TIA in known state
110    !
120    !  The next line explicitly sets the meas resolution
130    !
140    OUTPUT 70906;":TINT:RANG:RES 50E-12"
150    !
160    !  Set the number of msmts to acquire. Set it to whatever
170    !  you want.
180    !
190    OUTPUT 70906;":ACQ:MCO 15"  !15 msmts will be taken
200    !
210    ! Now ready to measure
220    !
230  Take_a_meas:!  Repeat this loop each time you want to meas
240    OUTPUT 70906;":INIT"  ! Take the measurement
250    !
260    ! Msmt has completed. Ask TIA for msmt length.
270    !
280    OUTPUT 70906;":ACQ:LENG?"
290    ENTER 70906;Length    ! Read in the length
300    !
310    ! Fetch the time interval data.
320    !
330    OUTPUT 70906;":FETC?"   ! Request the interval data
340    !
350    ! Two methods for reading the data are shown. The 3 lines
360    ! that follow show reading in the data one value at a time.
370    ! Line 440, which is commented, shows a way to read all the
380    !data in a single line.  The USING "%,K" specifies that the
390    ! entry will terminate on EOI, which is what we want.
400    !
410    FOR I=1 TO Length
420    ENTER 70906 USING "%,K";Tia_data(I)
430    NEXT I
440  ! ENTER 70906 USING "%,K";Tia_data(*)
450    !
460    ! The interval data is in the Tia_data array.  Now, ask for
470    ! the mean value of this data, which the TIA will compute
480    ! for you.
490    OUTPUT 70906;":FETC:TINT:MEAN?"  ! try SDEV, MIN, MAX
                                        ! instead of MEAN
500    ENTER 70906;Mean_val  ! Read in the mean
510    !
520    ! Print out the results
530    !
540    PRINT "Mean is:",Mean_val
550    FOR I=1 TO Length
560    PRINT "TI value ",I,":",Tia_data(I)
```

Programming Examples

Sequential Time Interval Example (Continued)

```
570    NEXT I
580 ! GOTO Take_a_meas   ! Uncomment to endlessly repeat msmt loop
590    END
```

Histogram Time Interval Example

```
10   !
20   ! Easiest way to make a time interval histogram measurement
30   ! and read the data from the E1740A in ASCII mode. Program
40   ! also shows how to get a computed result, in this case the
50   ! mean of the Histogram values.  You need to provide an
60   ! input to channel 1 of the E1740A (something in the 1-80 MHz
70   ! range would be fine).
80   !
90   !Define an array for the histogram data. There are 2048 bins.
100  !
110   REAL Hist_data(1:2048)    ! Define an array for the data
120   OUTPUT 70906;"*RST;*CLS" ! Put TIA in known state
130   !
140   !  Configure the TIA for a fast histogram measurement
150   !
160   OUTPUT 70906;":CONF:XTIN:HIST"
170   OUTPUT 70906;":HIST:RANG:RES 50E-12"  ! set the resolution
180   !
190   !  Set the number of msmts to acquire. Set it to whatever
200   !  you want.
210   !
220   OUTPUT 70906;":ACQ:MCO 1E6"  ! 1 million msmts will be taken
230   !
240   ! Now ready to measure
250   !
260  Take_a_meas:!  Repeat this loop each time you want to meas
270   OUTPUT 70906;":INIT"     ! Take the measurement
280   !
290   ! Msmt has completed, fetch the data.
300   !
310   OUTPUT 70906;":FETC?"    ! Request the histogram data
320   !
330   ! Two methods for reading the data are shown. The 3 lines
340   ! that follow show reading in the data one value at a time.
350   ! Line 430, which is commented, shows a way to read all the
360   ! data in a single line.  The USING "%,K" specifies that the
370   ! entry will terminate on EOI, which is what we want.  Be
380   ! patient, ASCII transfers are slow!
390   !
400   FOR I=1 TO 2048
410     ENTER 70906 USING "%,K";Hist_data(I)
420   NEXT I
430   !ENTER 70906 USING "%,K";Hist_data(*)
440   !
450   ! The histogram data is in the Hist_data array. Now, ask for
460   ! the mean value of this data, which the TIA will compute
      ! for you.
470   !
480   OUTPUT 70906;":FETC:TINT:MEAN?"   ! try SDEV, MIN, MAX
                                        ! instead of MEAN
490   ENTER 70906;Mean_val   ! Read in the mean
500   !
510   ! Print out the results. For the histogram, print only
      ! non-zero values.
520   !
530   PRINT "Mean is:",Mean_val
540   FOR I=1 TO 2046
550     IF Hist_data(I)<>0 THEN
560       PRINT "TI value ",I,":",Hist_data(I)
570     END IF
```

Histogram Time Interval Example (Continued)

```
580   NEXT I
590   !GOTO Take_a_meas  ! Uncomment to endlessly repeat msmt loop
600   END
```

Sequential Timestamp Example

```
10   !
20   ! Easiest way to make a sequential timestamp measurement
30   ! and read the data from the E1740A in ASCII mode. A
     ! timestamp
40   ! measurement records the value of an internal 16-bit
     ! counter
50   ! each time a data edge occurs. The data read back contains
     ! the
60   ! counter values. This example shows how this data could be
70   ! converted to produce a timeline of when each edge
     ! occurred.
80   ! You need to provide an input to channel 1 of the E1740A.
90   ! Something in the 1-80 MHz range would be fine.
100  !
110  ! Tst_data will contain the timestamp values,
     ! Cumulative_time
120  ! will get the accumulated timeline.
130  !
140  REAL Tst_data(1:1000),Cumulative_time(1:1000)
150  OUTPUT 70906;"*RST;*CLS" ! Put TIA in known state
160  !
170  ! Configure the E1740A for a timestamp measurement
180  !
190  OUTPUT 70906;":CONF:XTIM:TST"
200  !
210  ! Query the resolution, it will be used to scale the result
220  !
230  OUTPUT 70906;":TINT:RANG:RES?"
240  ENTER 70906;Resolution
250  !
260  !  Set the number of msmts to whatever you want
270  !
280  OUTPUT 70906;":ACQ:MCO 20"  ! 20 msmts will be taken
290  !
300  ! Now ready to measure
310  !
320 Take_a_meas:!  Repeat this loop each time you want to meas
330  OUTPUT 70906;":INIT"     ! Take the measurement
340  !
350  ! Msmt has completed. Ask TIA for msmt length.
360  !
370  OUTPUT 70906;":ACQ:LENG?"
380  ENTER 70906;Length       ! Read in the length
390  !
400  ! Fetch the time interval data.
410  !
420  OUTPUT 70906;":FETC?"    ! Request the timestamp data
430  !
440  ! Two methods for reading the data are shown. The 3 lines
450  ! that follow show reading in the data one value at a time.
460  ! Line 530, which is commented, shows a way to read all the
470  ! data in a single line.  The USING "%,K" specifies that the
480  ! entry will terminate on EOI, which is what we want.
490  !
500  FOR I=1 TO Length
510    ENTER 70906 USING "%,K";Tst_data(I)
520  NEXT I
530  !ENTER 70906 USING "%,K";Tst_data(*)
540  !
```

Programming Examples

Sequential Timestamp Example (Continued)

```
550  ! The timestamp data is in the Tst_data array. Now, use this
560  ! data to determine a cumulative time array that contains
     ! the
570  ! time-of-occurrence of each data edge.
580  !
590  Cumulative_time(1)=0.   ! first sample defines reference
                             !time=0
600  FOR I=2 TO Length
610    IF Tst_data(I)<Tst_data(I-1) THEN
620      !
630      ! If this IF branch taken, the counter overflowed, so an
         ! extra
640      ! cycle must be added. This_edge represents the number
         ! of counter
650      ! clicks that occurred from the last edge to this edge
         ! (counter
660      ! runs at 80 MHz rate).
670      !
680      This_edge=Tst_data(I)-Tst_data(I-1)+65536
690    ELSE
700      This_edge=Tst_data(I)-Tst_data(I-1)
710    END IF
720    !
730    ! cumulative time to this edge is cumulative time to prior
       ! edge +
740    ! incremental time to this edge.
750    !
760    Cumulative_time(I)=Cumulative_time(I-1)+This_edge
770  NEXT I
780  !
790  ! convert the Cumulative_time array from counter ticks to
     ! actual time
800  !
810  MAT Cumulative_time= Cumulative_time*(Resolution)
820  !
830  ! Print out the results
840  !
850  FOR I=1 TO Length
860    PRINT "Time to edge",I,":",Cumulative_time(I)
870  NEXT I
880 !GOTO Take_a_meas  ! Uncomment to endlessly repeat msmt loop
890  END
```

Programs Using REAL Format

Sequential Time Interval Example

```
10   !
20   ! Easiest way to make a sequential time interval measurement
30   ! and read the data from the E1740A in REAL mode. REAL and
     ! INT
40   ! will always be more efficient than ASCII for getting the
     ! data.
50   ! You should provide an input to channel 1 of the E1740A
60   ! (something in the 1-80 MHz range would be fine).
70   !
80   REAL Tia_data(1:1000)     ! Define an array for the data
90   !
100  ! When the data is read in, the numeric structure of the
     ! output
110  ! will match the structure of the array (Tia_data(*)) that
120  ! the data will be read into. Hence, no formatting should be
     ! used
130  ! during the transfer.
140  !
150  ASSIGN @Tia_no_format TO 70906;FORMAT OFF
160  OUTPUT 70906;"*RST;*CLS" ! Put TIA in known state
170  !
180  ! Set the format to output 64-bit floating point values
190  !
200  OUTPUT 70906;":FORM REAL"
210  !
220  ! The next line explicitly sets the meas resolution
230  !
240  OUTPUT 70906;":TINT:RANG:RES 50E-12"
250  !
260  !  Set the number of msmts to acquire. Set it to whatever
270  !  you want.
280  !
290  OUTPUT 70906;":ACQ:MCO 10"  !10 msmts will be taken
300  !
310  ! Now ready to measure
320  !
330 Take_a_meas:!  Repeat this loop each time you want to meas
340  OUTPUT 70906;":INIT"     ! Take the measurement
350  !
360  ! Msmt has completed, fetch the data.
370  !
380  OUTPUT 70906;":FETC?"    ! Request the interval data
390  !
400  ! The first part of the output will be "#X", where X
410  ! identifies the number of bytes of length info that
420  !follows
430  ENTER 70906 USING "#,2A";Header$
440  !
450  ! Use the length info from the prior enter to read in the
460  ! correct number of bytes for the length field.
470  !
480  ENTER 70906 USING "#,"&Header$[2,2]&"A";Data_bytes$
490  !
500  ! Data_bytes$ is a string containing the number of data
510  ! bytes that follow, convert it to a number.
520  !
530  Data_bytes=VAL(Data_bytes$)
540  !
```

Programming Examples

Sequential Time Interval Example (Continued)

```
550   ! For REAL mode, there are 8 bytes per measurement
560   !
570   Meas_count=Data_bytes/8
580   !
590   ! Read in the data. The @Tia_no_format just means that the
600   ! transfer will use the unformatted I/O path that we set up
610   ! at the beginning of the program. The transfer is efficient
620   ! because there is no data conversion needed. For even more
630   ! efficient transfer, a TRANSFER statement can be used.
640   !
650   FOR I=1 TO Meas_count
660     ENTER @Tia_no_format;Tia_data(I)
670   NEXT I
680   ENTER 70906 USING "#,A";Last_byte$
690   !
700   ! The interval data is in the Tia_data array, print out the
      ! results.
710   !
720   FOR I=1 TO Meas_count
730     PRINT "TI value ",I,":",Tia_data(I)
740   NEXT I
750  !GOTO Take_a_meas  ! Uncomment to endlessly repeat msmt loop
760   END
```

Histogram Time Interval Example (Accumulate ON)

```
10   !
20   ! Easiest way to make a time interval histogram measurement
30   ! and read the data from the E1740A in REAL mode.  The
40   ! E1740A outputs histogram data in this 64-bit real format
50   ! when the accumulation feature is on. You need to provide
60   ! an input to channel 1 of the E1740A (something in the
70   ! 1-80 MHz range would be fine).  REAL and INT will always
80   ! be much faster than ASCII for reading in data.
90   !
100  ! Define an array for the histogram data. There are 2048
     ! bins.
110  !
120  REAL Hist_data(1:2048)    ! Define an array for the data
130  !
140  ! When the data is read in, it doesn't need to be formatted,
150  ! so define an unformatted I/O path to the E1740A.
160  !
170  ASSIGN @Tia_no_format TO 70906;FORMAT OFF
180  OUTPUT 70906;"*RST;*CLS" ! Put TIA in known state
190  !
200  !  Configure the TIA for a fast histogram measurement
210  !
220  OUTPUT 70906;":CONF:XTIN:HIST"
230  !
240  ! Set accumulation to ON, which supports REAL format.
250  !
260  OUTPUT 70906;":HIST:ACC ON"  ! clear with :HIST:CLEAR
270  OUTPUT 70906;":FORM REAL" ! set the format
280  OUTPUT 70906;":HIST:RANG:RES 50E-12"  ! set the resolution
290  !
300  !  Set the number of msmts to acquire.
310  !
320  OUTPUT 70906;":ACQ:MCO 1E6"  ! 1 million msmts will be taken
330  !
340  ! Now ready to measure
350  !
360 Take_a_meas:!  Repeat this loop each time you want to meas
370  OUTPUT 70906;":INIT"     ! Take the measurement
380  !
390  ! Msmt has completed, fetch the data.
400   !
410  OUTPUT 70906;":FETC?"    ! Request the histogram data
420  !
430  ! The first part of the output will be "#X", where X
440  ! identifies the number of bytes of length info that
450  !follows
460  ENTER 70906 USING "#,2A";Header$
470  !
480  ! Use the length info from the prior enter to read in the
490  ! correct number of bytes for the length field.
500  !
510  ENTER 70906 USING "#,"&Header$[2,2]&"A";Data_bytes$
520  !
530  ! Data_bytes$ is a string containing the number of data
     ! bytes
540  ! that follow. Convert it to a number.
550  !
560  Data_bytes=VAL(Data_bytes$)
570  !
580  ! There are 8 bytes per histogram bin (64-bits = 8 bytes)
```

Programming Examples

Histogram Time Interval Example (Accumulate ON) (Continued)

```
590  !
600  Hist_bins=Data_bytes/8
610  !
620  ! Read in the data. In the most general form, you could use
     ! the
630  ! FOR-NEXT loop that is commented. It will react dynamically
     ! to
640  ! a change in the number of bins being output. However, for
     ! this
650  ! example, we know that there will always be 2048 values
     ! output,
660  ! so a quicker way to get the data is via line 720. Even
     ! faster
670  ! would be to use a TRANSFER statement.
680  !
690 ! FOR I=1 TO Hist_bins
700 !   ENTER @Tia_no_format;Hist_data(I)
710 ! NEXT I
720  ENTER @Tia_no_format;Hist_data(*)
730  ENTER 70906 USING "#,A";Last_byte$
740  !
750  ! The histogram data is in the Hist_data array. Print out
     ! any
760  ! non-zero bins.
770  !
780  FOR I=1 TO 2048
790    IF Hist_data(I)<>0 THEN
800      PRINT "TI value ",I,":",Hist_data(I)
810    END IF
820  NEXT I
830  !
840  ! Print out the total occurrences in the histogram. If you
     ! loop the
850  ! measurements, you'll observe the accumulation of the
     ! histogram.
860  !
870  PRINT "Total occurrences in histogram:",SUM(Hist_data)
880  !
890  ! Comment the next line if you only want a single pass. If
     ! you loop back,
900  ! the updates will accumulate, since :HIST:ACC is ON.
910  !
920  GOTO Take_a_meas
930  END
```

Programs Using INTEGER Format

Sequential Time Interval Example

```
10   !
20   ! Easiest way to make a sequential time interval measurement
30   ! and read the data from the E1740A in INTEGER mode. REAL
     ! and INT
40   ! will always be more efficient than ASCII for getting the
     ! data.
50   ! You should provide an input to channel 1 of the E1740A
60   ! (something in the 1-80 MHz range would be fine).
70   !
80   INTEGER Tia_data(1:1000)     ! Define an array for the data
90   !
100  ! When the data is read in, the numeric structure of the
     ! output
110  ! will match the structure of the array (Tia_data(*)) that
     ! the
120  ! data will be read into. Hence, no formatting should be
     ! used
130  ! during the transfer.
140  !
150  ASSIGN @Tia_no_format TO 70906;FORMAT OFF
160  OUTPUT 70906;"*RST;*CLS" ! Put TIA in known state
170  !
180  ! Set the format to output 16-bit integer values
190  !
200  OUTPUT 70906;":FORM INT"
210  !
220  ! The next line explicitly sets the meas resolution
230  !
240  OUTPUT 70906;":TINT:RANG:RES 50E-12"
250  !
260  ! Query the exact resolution. This will be used to scale
270  ! the data.
280  !
290  OUTPUT 70906;":TINT:RANG:RES?"
300  ENTER 70906;Resolution
310  !
320  !  Set the number of msmts to acquire. Set it to whatever
     ! you want.
330  !
340  OUTPUT 70906;":ACQ:MCO 10"  !10 msmts will be taken
350  !
360  ! Now ready to measure
370  !
380 Take_a_meas:!  Repeat this loop each time you want to meas
390  OUTPUT 70906;":INIT"     ! Take the measurement
400  !
410  ! Msmt has completed, fetch the data.
420  !
430  OUTPUT 70906;":FETC?"    ! Request the interval data
440  !
450  ! The first part of the output will be "#X", where X
460  ! identifies the number of bytes of length info that
     ! follows.
470  !
480  ENTER 70906 USING "#,2A";Header$
490  !
500  ! Use the length info from the prior enter to read in the
```

Programming Examples

Sequential Time Interval Example (Continued)

```
510   ! correct number of bytes for the length field.
520   !
530   ENTER 70906 USING "#,"&Header$[2,2]&"A";Data_bytes$
540   !
550   ! Data_bytes$ is a string containing the number of data
560   ! bytes that follow, convert it to a number.
570   !
580   Data_bytes=VAL(Data_bytes$)
590   !
600   ! For INTEGER mode, there are 2 bytes per measurement
610   !
620   Meas_count=Data_bytes/2
630   !
640   ! Read in the data. The @Tia_no_format just means that the
650   ! transfer will use the unformatted I/O path that we set up
660   ! at the beginning of the program. The transfer is efficient
670   ! because there is no data conversion needed. For even more
680   ! efficient transfer, a TRANSFER statement can be used.
690   !
700    FOR I=1 TO Meas_count
710     ENTER @Tia_no_format;Tia_data(I)
720   NEXT I
730   ENTER 70906 USING "#,A";Last_byte$
740   !
750   ! The interval data is in the Tia_data array, print out the
760   ! results. For INTEGER format, the intervals are in counter
770   ! tic units. To get time intervals, multiply by the
      ! resolution,
780   ! which we queried earlier in the program.
790   !
800   FOR I=1 TO Meas_count
810     PRINT "TI value ",I,":",Tia_data(I)*Resolution
820   NEXT I
830  !GOTO Take_a_meas  ! Uncomment to endlessly repeat msmt loop
840   END
```

Histogram Time Interval Example (Accumulate OFF)

```
10   !
20   ! Easiest way to make a time interval histogram measurement
30   ! and read the data from the E1740A in INTEGER mode.  The
40   ! E1740A outputs histogram data in this 32-bit integer
     ! format
50   ! when the accumulation feature is off. You need to provide
60   ! an input to channel 1 of the E1740A (something in the
70   ! 1-80 MHz range would be fine).
80   !
90   ! Define an array for the histogram data. There are 2048
     ! bins.
100  !
110  REAL Hist_data(1:2048)    ! Define an array for the data
120  !
130  ! When the data is read in, it doesn't need to be formatted,
140  ! so define an unformatted I/O path to the E1740A.
150  !
160  ASSIGN @Tia_no_format TO 70906;FORMAT OFF
170  OUTPUT 70906;"*RST;*CLS" ! Put TIA in known state
180  !
190  !  Configure the TIA for a fast histogram measurement
200  !
210  OUTPUT 70906;":CONF:XTIN:HIST"
220  !
230  ! Set accumulation to OFF, which supports INT format.
240  !
250  OUTPUT 70906;":HIST:ACC OFF"
260  OUTPUT 70906;":FORM INT" ! set the format
270  OUTPUT 70906;":HIST:RANG:RES 50E-12"  ! set the resolution
280  !
290  !  Set the number of msmts to acquire.
300  !
310  OUTPUT 70906;":ACQ:MCO 1E6"  ! 1 million msmts will be taken
320  !
330  ! Now ready to measure
340  !
350 Take_a_meas:!  Repeat this loop each time you want to meas
360  OUTPUT 70906;":INIT"    ! Take the measurement
370  !
380  ! Msmt has completed, fetch the data.
390  !
400  OUTPUT 70906;":FETC?"   ! Request the histogram data
410  !
420  ! The first part of the output will be "#X", where X
430  ! identifies the number of bytes of length info that
440  !follows.
450  ENTER 70906 USING "#,2A";Header$
460  !
470  ! Use the length info from the prior enter to read in the
480  ! correct number of bytes for the length field.
490  !
500  ENTER 70906 USING "#,"&Header$[2,2]&"A";Data_bytes$
510  !
520  ! Data_bytes$ is a string containing the number of data
     ! bytes
530  ! that follow. Convert it to a number.
540  !
```

Programming Examples

Histogram Time Interval Example (Accumulate OFF) (Continued)

```
550  Data_bytes=VAL(Data_bytes$)
560  !
570  ! There are 4 bytes per histogram bin (32-bits = 4 bytes)
580  !
590  Hist_bins=Data_bytes/4
600  !
610  ! Read in the data. Because HP Basic doesn't support a
     ! 32-bit
620  ! integer type, the loop is much more complicated that it
     ! would
630  ! be in many other languages. Here, each 32-bit value is
     ! read
640  ! into two 16-bit integers. They are then combined to
650  ! produce thedesired result.
660  !
670  INTEGER Upper_byte,Lower_byte
680  FOR I=1 TO Hist_bins
690    !
700    ! Read the 32-bit integer histogram data into two 16-bit
       ! integers.
710    !
720    ENTER @Tia_no_format;Upper_byte,Lower_byte
730    !
740    ! This test is needed because the lower byte will be
       ! interpretted
750    ! as a negative value if the most significant bit is 1.
       ! When this
760    ! happens the needed correction is as shown.
770    !
780    IF Lower_byte<0 THEN
790      Hist_data(I)=Upper_byte*65536+Lower_byte+65536
800    ELSE
810      Hist_data(I)=Upper_byte*65536+Lower_byte
820    END IF
830  NEXT I
840  ENTER 70906 USING "#,A";Last_byte$
850  !
860  ! The histogram data is in the Hist_data array. Print out
     ! any
870  ! non-zero bins.
880  !
890  FOR I=1 TO 2048
900    IF Hist_data(I)<>0 THEN
910      PRINT "TI value ",I,":",Hist_data(I)
920    END IF
930  NEXT I
940  GOTO Take_a_meas  ! Uncomment to endlessly repeat msmt loop
950  END
```

Sequential Timestamp Example

```
10   !
20   ! Easiest way to make a sequential timestamp measurement
30   ! and read the data from the E1740A in INTEGER mode. A
40   ! timestamp measurement records the value of an internal
50   ! 16-bit counter each time a data edge occurs. The data
60   ! read back contains the counter values. This example shows
70   ! how this data could be converted to produce a timeline of
80   ! when each edge occurred.  You need to provide an input to
90   ! channel 1 of the E1740A. Something in the 1-80 MHz range
100  ! would be fine. INT format will be much faster than ASCII.
110  !
120  ! Tst_data will contain the timestamp values,
130  ! Cumulative_time will get the accumulated timeline.
140  !
150  INTEGER Tst_data(1:1000)
160  REAL Cumulative_time(1:1000)
170  !
180  ! When the data is read in, the numeric structure of the
190  ! output will match the structure of the array
200  ! (Tst_data(*)) that the data will be read into. Hence, no
210  ! formatting should be used during the transfer.
220  !
230  ASSIGN @Tia_no_format TO 70906;FORMAT OFF
240  OUTPUT 70906;"*RST;*CLS" ! Put TIA in known state
250  OUTPUT 70906;":FORM INT" ! Set the format to INT
260  !
270  ! Configure the E1740A for a timestamp measurement
280  !
290  OUTPUT 70906;":CONF:XTIM:TST"
300  !
310  ! Query the resolution, it will be used to scale the result
320  !
330  OUTPUT 70906;":TINT:RANG:RES?"
340  ENTER 70906;Resolution
350  !
360  !  Set the number of msmts to whatever you want
370  !
380  OUTPUT 70906;":ACQ:MCO 20"  ! 20 msmts will be taken
390  !
400  ! Now ready to measure
410  !
420 Take_a_meas:!  Repeat this loop each time you want to meas
430  OUTPUT 70906;":INIT"    ! Take the measurement
440  !
450  ! Msmt has completed. Ask TIA for msmt length.
460  !
```

Programming Examples

Sequential Timestamp Example (Continued)

```
470  OUTPUT 70906;":FETC?"   ! Request the timestamp data
480  !
490  ! The first part of the output will be "#X", where X
500  ! identifies the number of bytes of length info that
510  ! follows.
520  ENTER 70906 USING "#,2A";Header$
530  !
540  ! Use the length info from the prior enter to read in the
550  ! correct number of bytes for the length field.
560  !
570  ENTER 70906 USING "#,"&Header$[2,2]&"A";Data_bytes$
580  !
590  ! Data_bytes$ is a string containing the number of data
600  ! bytes that follow, convert it to a number.
610  !
620  Data_bytes=VAL(Data_bytes$)
630  !
640  ! For INTEGER format, there are 2 bytes per timestamp
650  ! value
660  Meas_count=Data_bytes/2
670  !
680  ! Read in the data. The @Tia_no_format just means that
690  ! the transfer will use the unformatted I/O path that
700  ! we set up at the beginning of the program. The
710  ! transfer is efficient because there is no data
720  ! conversion needed.
730  FOR I=1 TO Meas_count
740    ENTER @Tia_no_format;Tst_data(I)
750  NEXT I
760  ENTER 70906 USING "#,A";Last_byte$
770  !
780  ! The timestamp data is in the Tst_data array. Now, use
790  ! this data to determine a cumulative time array that
800  ! contains the time-of-occurrence of each data edge.
810  !
820  Cumulative_time(1)=0. !first sample defines reference
                           ! time=0
830   FOR I=2 TO Meas_count
840    IF Tst_data(I)<Tst_data(I-1) THEN
850     !
860     ! If this IF branch taken, the counter overflowed,
870     ! so an extra cycle must be added. This_edge
880     ! represents the number of counter clicks that
890     ! occurred from the last edge to this edge (counter
900     ! runs at 80 MHz rate). The0. is included in the
910     ! computation to keep the 16-bit integer array from
920     ! possibly overflowing when the difference is taken.
930      This_edge=Tst_data(I)-(0.+Tst_data(I-1))+65536
940    ELSE
950      This_edge=Tst_data(I)-Tst_data(I-1)
960    END IF
970    !
980    ! cumulative time to this edge is cumulative time to
990    ! prior edge + incremental time to this edge.
1000   !
1010   Cumulative_time(I)=Cumulative_time(I-1)+This_edge
1020 NEXT I
1030 !
```

Sequential Timestamp Example (Continued)

```
1040  ! convert the Cumulative_time array from counter ticks
1050  ! to actual time
1060  MAT Cumulative_time= Cumulative_time*(Resolution)
1070  !
1080  ! Print out the results
1090  !
1100 FOR I=1 TO Meas_count
1110   PRINT "Time to edge",I,":",Cumulative_time(I)
1120  NEXT I
1130 !GOTO Take_a_meas  ! Uncomment to endlessly repeat msmt
     ! loop
1140  END
```

Programming Examples

Advanced Programming Examples

To Query and Print Out the Error Queue of the HP E1740A (HP BASIC)

```
10  !This program is an example of a subroutine to query
20  !and print out the error queue of the HP E1740A Time
30  !Interval Analyzer.
40  !
50  !
60  !
70  ASSIGN @Tia TO 70906
80  !
90  !
100 !...........BODY OF PROGRAM.........
110 !
120 !
130 CALL Read_error(@Tia)
140 !
150 END
160 !
170 SUB Read_error(@Tia)  !Subroutine to read the error queue
180   DIM Error_msg$[256]
190   REPEAT
200   OUTPUT @Tia;"SYST:ERR?"
210   ENTER @Tia;Error_code,Error_msg$
220   PRINT Error_code,Error_msg$
230   UNTIL NOT (Error_code)
240 SUBEND
```

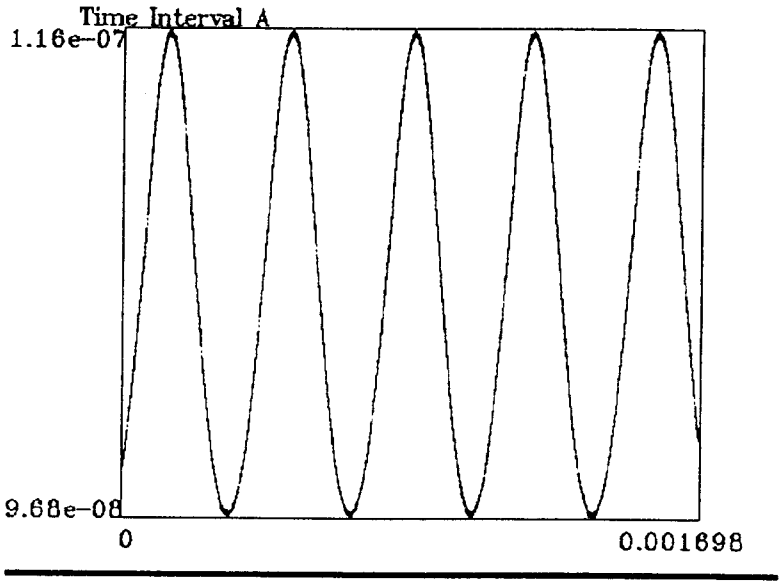


Figure 15-1. Print Out of Time Interval Data From the HP E1740A (HP BASIC)

Programming Examples

To Make Time Interval Histogram Measurements on Channel 1 (HP BASIC)

```
10   ! This program sets up the HP E1740A Time Interval Analyzer
20   ! to make time interval histogram measurements on channel 1.
30   ! The number of measurements to make is set by the variable
40   ! Num_samples. The histogram range is set by three variables.
50   ! The minimum value is set by Tint_offset.  The maximum value
60   ! is equal to (Tint_offset+Tint_res*Num_bins).  After the
70   ! measurement is complete, the data is read using FORMAT OFF and
80   ! displayed as a histogram.  The signal used for this example
90   ! is a 1 MHz sine wave with 5 ns rms jitter.
100  !
110  !
120  CLEAR SCREEN
130  INTEGER Histogram(1:4096),I
140  REAL Results(1:2048)
150  DIM Header$[256]
160  Total_sum=0
170  Tint_offset=9.50E-7     !Set offset (minimum TI value)
180  Tint_res=1.25E-8/256    !Set resolution (appx 48.8ps)
190  Num_bins=2048       !Set number of histogram bins
200  Num_samples=1.E+6     !Set number of samples
210 !
220  ASSIGN @Tia TO 70906
230  ASSIGN @Tia2 TO 70906;FORMAT OFF  !Used for transfer of data
240  !
250  INPUT "Connect a 1 MHz signal to channel 1. Press RETURN.",A$
260  !
270  !Setup a Time Interval Histogram measuremnt on channel A.
280  CLEAR @Tia
290  OUTPUT @Tia;"*RST"      !Reset the HP E1740A
300  OUTPUT @Tia;"*CLS"      !Clear event registers and error queue
310  OUTPUT @Tia;"*SRE 0"    !Clear service request enable register
320  OUTPUT @Tia;"*ESE 0"    !Clear event status enable register
330  !
340  !
350  !
360  OUTPUT @Tia;":FORMAT INTEGER"  !Set data format to integer
370  !
380  OUTPUT @Tia;":INP1:COUP DC"  !Channel 1 DC coupled
390  OUTPUT @Tia;":INP1:IMP 50"   !Channel 1 50 ohms
400  OUTPUT @Tia;":EVENT1:LEVEL 0"  !Channel 1 0 volt trigger level
410  OUTPUT @Tia;":EVENT1:SLOP POS" !Channel 1 trigger slope positive
420  !
430  OUTPUT @Tia;"CONF:XTIN:HIST"   !Single source time interval histogram
440  OUTPUT @Tia;":ACQ:SOUR MCOUNT"
450  OUTPUT @Tia;":ACQ:MCOUNT ";Num_samples !Set measurement length to
                      !Num_samples
460  OUTPUT @Tia;":TRIG:COUNT 1"   !Set number of aquisitions to 1
470  OUTPUT @Tia;":ACQ:PAC IMM"  !Set measurement pacing to immediate
480  OUTPUT @Tia;":TRIG:SOUR IMM"  !Set start trigger to free run
490  !
500  !The next two lines set the histogram measurement range to 950ns to
510  !1.05us.  The minimum value is set by the variable Tint_offset. The
520  !maximum value equals (minimum value + (resolution * 2048)).    2048
530  !is the number of histogram bins.
540  OUTPUT @Tia;":HIST:RANGE:RES ";Tint_res  !Set histogram resolution
550  OUTPUT @Tia;":HIST:RANGE:OFFSET ";Tint_offset  !Set histogram offset
560  !
```

To Make Time Interval Histogram Measurements on Channel 1 (HP BASIC) (Continued)

```
570  OUTPUT @Tia;"*SRE 16"  !Set service request enable mask to pull
580               ! SRQ on the measurement available bit
590  OUTPUT @Tia;":INIT"       !Initiate measurement
600  OUTPUT @Tia;"FETCH? 0,2048"   !Request measurement results
610  ON INTR 7 GOTO Done       !Setup interrupt
620  ENABLE INTR 7;2
630  PRINT "Waiting for measurement available SRQ"
640 Here:GOTO Here   !Could do some other processing while waiting  for
                       !the SRQ
650 Done:OFF INTR 7  !Disable interrupt
660  Test=SPOLL(@Tia)
670  PRINT "Measurement complete, serial poll = ";Test
680  !
690  ENTER @Tia2 USING "#,6A";Header$  !Get information about measurement
700  ENTER @Tia2;Histogram(*)      !Enter data
710  ENTER @Tia2 USING "X"
720  !
730  !
740  CALL Get_4byte_val(Histogram(*),Results(*))
750  FOR I=1 TO 2048
760  Total_sum=Total_sum+Results(I)
770  NEXT I
780  CALL
Hist_label(@Tia,Tint_offset,Tint_res,Num_bins,Num_samples,Results(*))
790  END
800  !
810  !
820  SUB Get_4byte_val(INTEGER Histogram(*),REAL Results(*))
830 Get_4byte_val: ! Converts two BASIC INTEGER types into
840        ! an unsigned 32 bit number.
850  REDIM Histogram(0:4095)
860  INTEGER I,J
870  Two_exp16=65536
880  FOR I=0 TO 4095 STEP 2
890    J=I/2+1
900
Results(J)=(Histogram(I)+(Histogram(I)<0)*Two_exp16+(Histogram(I+1)<0))*Tw
c_exp16+Histogram(I+1)
910  NEXT I
920  SUBEND   !Get_4byte_val
930  !
940  !
950  !Subroutine to plot the histogram to the display
960  SUB
Hist_label(@Tia,Tint_offset,Tint_res,Num_bins,Num_samples,Results(*))
970  GINIT
980  GCLEAR
990  PLOTTER IS CRT,"INTERNAL"
1000   CLEAR SCREEN
1010   DIM Offset$[24],Resolution$[24]
1020   VIEWPORT
.15*(100*MAX(1,RATIO)),.75*(100*MAX(1,RATIO)),.30*(100*MAX(1,1/RATIO)),.95
*(100*MAX(1,1/RATIO))
1030   FRAME
1040   Max_y=1.1*MAX(Results(*))
1050   Max_y=MAX(1,Max_y)
1060   WINDOW 1,Num_bins,0,Max_y
```

Programming Examples

To Make Time Interval Histogram Measurements on Channel 1 (HP BASIC) (Continued)

```
1070    PEN 1
1080    CLIP OFF
1090    CSIZE 3
1100    MOVE 1,0
1110    LORG 3
1120    OUTPUT @Tia;":HIST:RANGE:OFFS?"
1130    ENTER @Tia;Offset$
1140    Tint_offset=VAL(Offset$)
1150    LABEL USING "3D.DE,X,A";Tint_offset;"s"
1160    MOVE Num_bins,0
1170    LORG 9
1180    OUTPUT @Tia;":HIST:RANGE:RES?"
1190    ENTER @Tia;Resolution$
1200    Tint_res=VAL(Resolution$)
1210    LABEL USING "D.4DE,A";Tint_offset+Num_bins*Tint_res;"s"
1220    MOVE 1,Max_y
1230    LORG 8
1240    LABEL USING "D.3DE";Max_y
1250    MOVE Num_bins/2,-Max_y/15
1260    LORG 6
1270    LABEL USING "9D,X,7A";Num_samples,"Samples"
1280    MOVE 0,0
1290    FOR I=1 TO Num_bins
1300    DRAW I,Results(I)
1310    NEXT I
1320    MAT SEARCH Results,LOC MAX;Maximum
1330    LORG 2
1340    MOVE Maximum,Results(Maximum)
1350    LABEL ((Maximum*Tint_res)+Tint_offset);"s"
1360 SUBEND
```

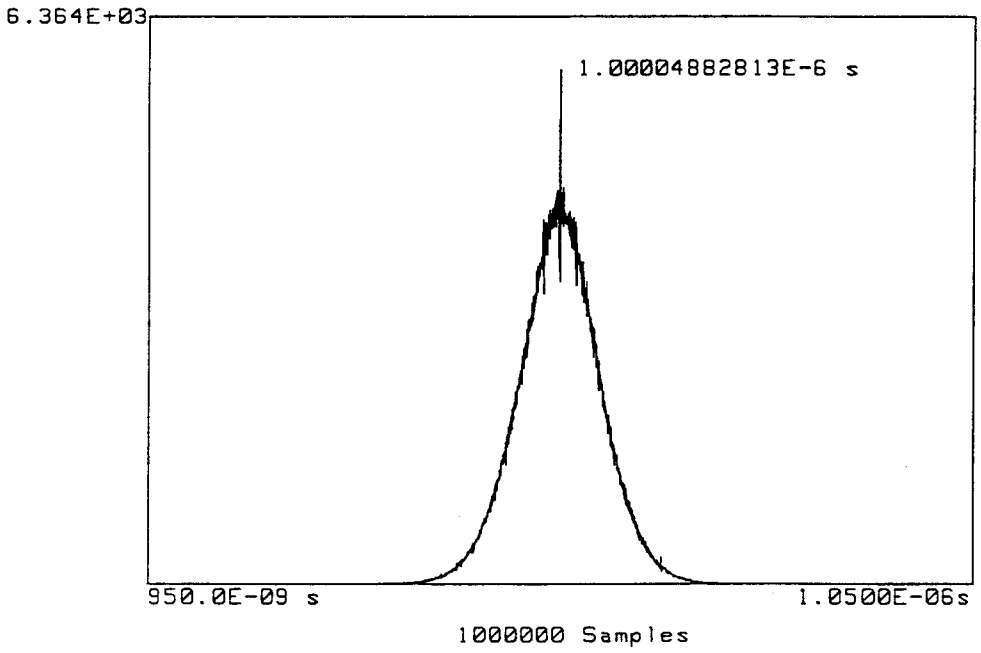


Figure 15-2. ASCII Transfer of Time Interval Histogram Data (HP BASIC)

Programming Examples

To Make Single-Channel Continuous Time Interval Measurements (HP BASIC)

```
10   ! This program sets up the HP E1740A Time Interval Analyzer to
20   ! make single channel continuous time interval measurements.
30   ! The measurements are made using the finest resolution (48.8ps).
40   ! The resolution is set by the variable Tint_res.  The number of
50   ! measurements to make is set by the variable Num_samples. After the
60   ! measurement is complete, the data is read and plotted versus time.
70   ! The measurement data is stored in sequential memory.  The example
80   ! shows how to handle large sample sizes (>32k).  TRANSFER is used
90   ! for I/O, and SRQ on operation complete is used.
100  !
110  !
120  COM INTEGER Rows_res,Rows_int,Col_res,Col_int,Over_flow
130  CLEAR SCREEN
140  INTEGER I
150  INTEGER Intervals(1:2,1:32000) BUFFER
160  DIM
Results(1:2,1:32000),Time_intervals(1:2,1:32000),Time_stamps(1:2,1:32000),
Header$[256],Info$[2]
170  !Would need larger dimensions for sample sizes greater than 64k.
180  !
190  ASSIGN @Int_buff TO BUFFER Intervals(*)
200  Total_sum=0
210  Tint_offset=0
220  Over_flow=0
230  Tint_res=1.25E-8/256     !Set resolution (appx 48.8ps)
240  Num_samples=1000       !Set number of samples
250  !
260  !Dimensioning for large or small sample sizes
270  IF Num_samples<32768 THEN
280  REDIM
Results(1:1,1:Num_samples),Time_intervals(1:1,1:Num_samples),Time_stamps(1
:1,1:Num_samples),Intervals(1:1,1:Num_samples)
290  ELSE
300  Cols=Num_samples/32
310  IF FRACT(Cols)<>0 THEN Cols=INT(Cols)+1
320  Rows=32
330  REDIM
Results(1:Rows,1:Cols),Intervals(1:Rows,1:Cols),Time_intervals(1:Rows,1:Co
ls),Time_stamps(1:Rows,1:Cols)
340  END IF
350  !
360  Rows_res=SIZE(Results,1)
370  Rows_int=SIZE(Intervals,1)
380  Col_res=SIZE(Results,2)
390  Col_int=SIZE(Intervals,2)
400  !
410  ASSIGN @Tia TO 70906
420  ASSIGN @Tia2 TO 70906;FORMAT OFF  !Used for entry of data
430  ASSIGN @Path TO 70906         !Used for data TRANSFER
440  !
450  INPUT "Connect signal to Input 1 (3.2 us max period).  Press
Return",A$
460  !
470  CLEAR 70906
480  OUTPUT @Tia;"*RST"      !Reset the HP E1740A
490  OUTPUT @Tia;"*CLS"      !Clear event registers and error queue
500  OUTPUT @Tia;"*SRE 0"    !Clear service request enable register
```

Programming Examples

To Make Single-Channel Continuous Time Interval Measurements (HP BASIC) (Continued)

```
510  OUTPUT @Tia;"*ESE 0"    !Clear event status enable register
520  !
530  !
540  !
550  OUTPUT @Tia;":FORMAT INTEGER" !Set data format to integer
560  !
570  OUTPUT @Tia;":INP1:COUP DC"   !Channel 1 DC coupled
580  OUTPUT @Tia;":INP1:IMP 50"    !Channel 1 50 ohms
590  OUTPUT @Tia;":EVENT1:LEVEL .5" !Channel 1 .5 volt trigger level
600  OUTPUT @Tia;":EVENT1:SLOP POS" !Channel 1 trigger slope positive
610  !
620  OUTPUT @Tia;"CONF:XTIM:TST DEF,DEF,(@1)" !Single source time stamp
630  OUTPUT @Tia;":ACQ:SOUR MCOUNT"
640  OUTPUT @Tia;":ACQ:MCOUNT ";Num_samples !Set measurement length to
                    !Num_samples
650  OUTPUT @Tia;":TRIG:COUNT 1"    !Set number of aquisitions to 1
660  OUTPUT @Tia;":ACQ:PAC IMM"     !Set measurement pacing to immediate
670                    !  (measure every edge)
680  OUTPUT @Tia;":TRIG:SOUR IMM"  !Set start trigger to free run
690  OUTPUT @Tia;":TINT:RANGE:RES ";Tint_res   !Set resolution
700  !
710  OUTPUT @Tia;"*ESE 1"     !Set event status enable register to 1
720                !   (operation complete)
730  OUTPUT @Tia;"*SRE 32"  !Enable standard event status register
740  ON INTR 7 GOTO Take_data     !Setup interrupt
750  ENABLE INTR 7;2
760  OUTPUT @Tia;"INIT"         !Initiate measurement
770  OUTPUT @Tia;"*OPC"         !Enable operation complete bit
780  !
790  DISP "Waiting for measurement to complete"
800 Loop_here:GOTO Loop_here !Could do some other processing while waiting
810               ! for interrupt
820   !
830 Take_data:OFF INTR 7    !Disable interrupt
840  S=SPOLL(70906)
850  DISP "Measurement complete - serial poll = ";S
860  OUTPUT @Tia;"FETCH? 0,";Num_samples !Request measurement results
870  !
880  !Transfer measurement results
890  ENTER @Tia2 USING "#,2A";Info$
900  ENTER @Tia2 USING "#,K";Header$[1;VAL(Info$[2])]
910  Num_bytes=VAL(Header$[1;VAL(Info$[2])])
920  TRANSFER @Path TO @Int_buff;COUNT Num_bytes,WAIT
930  ENTER @Tia USING "X"
940  !
950  OUTPUT @Tia;":HIST:RANGE:RES?"  !Query resolution
960  ENTER @Tia;Tint_res
970  !
980  CALL Get_4byte_val(Intervals(*),Results(*),Num_samples,Tint_res)
990  CALL
Convert_it(Results(*),Time_stamps(*),Time_intervals(*),Num_samples,Tint_re
s)
1000  !
1010  CALL
Plot_it(@Tia,Tint_offset,Tint_res,Num_samples,Time_intervals(*),Time_stamp
s(*))
1020  Average=SUM(Time_intervals)/Num_samples
```

Programming Examples

To Make Single-Channel Continuous Time Interval Measurements (HP BASIC) (Continued)

```
1030  DISP "Average value  = ";Average
1040  STOP
1050  END
1060  !
1070  !
1080  !
1090  SUB Get_4byte_val(INTEGER Intervals(*),REAL
Results(*),Num_samples,Tint_res)
1100 Get_4byte_val:! Converts a pair of unsigned BASIC INTEGERs into a
1110         ! 32 bit number.
1120   COM INTEGER Rows_res,Rows_int,Col_res,Col_int,Over_flow
1130   REDIM Intervals(1:Rows_int,0:Col_int-1)
1140   INTEGER I,J
1150   Two_exp16=65536
1160   IF MIN(Intervals(*))<0 THEN
1170   FOR J=1 TO Rows_int
1180     FOR I=0 TO Col_int-1
1190     Results(J,I+1)=(Intervals(J,I)+(Intervals(J,I)<0)*Two_exp16)
1200     NEXT I
1210   NEXT J
1220   ELSE
1230   REDIM Intervals(1:Rows_int,1:Col_int)
1240   MAT Results= Intervals
1250   END IF
1260  SUBEND! Get_4byte_val
1270  !
1280  !
1290  !Subroutine to plot the measurement results
1300  SUB Plot_it(@Tia,Tint_offset,Tint_res,Num_samples,REAL
Results(*),Time_stamps(*))
1310  GINIT
1320  GCLEAR
1330 !PLOTTER IS "test","HPGL"
1340  PLOTTER IS CRT,"INTERNAL"
1350   COM INTEGER Rows_res,Rows_int,Col_res,Col_int,Over_flow
1360   DIM Offset$[24],Resolution$[24]
1370   Total_time=SUM(Results)
1380   VIEWPORT
.15*(100*MAX(1,RATIO)),.75*(100*MAX(1,RATIO)),.30*(100*MAX(1,1/RATIO)),.95
*(100*MAX(1,1/RATIO))
1390   FRAME
1400   MAT Time_stamps= Time_stamps-(Time_stamps(1,1)) !Normalize to 0
seconds
1410   Start_time=0
1420   Total_time=Time_stamps(Rows_int,Col_int-Over_flow-1)
1430   Max_y=MAX(Results(*))+.1*(MAX(Results(*))-MIN(Results(*)))
1440   Min_y=MIN(Results(*))-.1*(MAX(Results(*))-MIN(Results(*)))
1450   MOVE 0,0
1460   WINDOW Start_time,Total_time,Min_y,Max_y
1470   PEN 1
1480   CLIP OFF
1490   CSIZE 3
1500   MOVE Start_time,Min_y
1510   LORG 3
1520   OUTPUT @Tia;":HIST:RANGE:OFFS?"  !Query offset
1530   ENTER @Tia;Offset$
1540   Tint_offset=VAL(Offset$)
1550   LABEL USING "3D.DE,X,A";Start_time;"s"
```

To Make Single-Channel Continuous Time Interval Measurements (HP BASIC) (Continued)

```
1560    MOVE Total_time,Min_y
1570    LORG 9
1580    LABEL USING "D.4DE,A";Total_time,"s"
1590    MOVE Start_time,Max_y
1600    LORG 8
1610    LABEL USING "D.3DE,A";Max_y;"s"
1620    MOVE Start_time,Min_y
1630    LABEL USING "D.3DE,A";Min_y;"s"
1640    MOVE (Total_time-Start_time)/2,-Max_y/15
1650    LORG 6
1660    LABEL USING "9D,X,7A";Num_samples,"Samples"
1670    MOVE 0,0
1680    Columns=Col_res-1
1690    FOR J=0 TO Rows_res-1
1700    IF J=Rows_res-1 THEN Columns=Columns-Over_flow
1710    FOR I=1 TO Columns
1720      PLOT Time_stamps(J+1,I+1),Results(J+1,I)
1730    NEXT I
1740    NEXT J
1750 SUBEND
1760 !
1770 !Convert to time intervals
1780 SUB
Convert_it(Results(*),Time_stamps(*),Time_intervals(*),Num_samples,
Tint_res)
1790    INTEGER I,J
1800    COM INTEGER Rows_res,Rows_int,Col_res,Col_int,Over_flow
1810    Time_overflow=0
1820    Two_exp16=65536
1830    REDIM Time_intervals(1:Rows_int,1:Col_int-1)
1840    FOR J=1 TO Rows_int
1850    Time_stamps(J,1)=(Results(J,1)+Time_overflow)*Tint_res
1860    FOR I=2 TO Col_int
1870      IF Results(J,I)<Results(J,I-1) THEN
Time_overflow=Time_overflow+Two_exp16
1880      Time_stamps(J,I)=(Results(J,I)+Time_overflow)*Tint_res
1890      Time_intervals(J,I-1)=Time_stamps(J,I)-Time_stamps(J,I-1)
1900    NEXT I
1910    NEXT J
1920    IF REAL(1.0*Rows_int*Col_int)>Num_samples THEN
1930    Over_flow=REAL(1.0*Rows_int*Col_int)-Num_samples
1940    Dummy=Time_stamps(1,1)
1950    Dummy_int=Time_intervals(1,1)
1960    FOR K=Col_int-Over_flow TO Col_int
1970      Time_stamps(Rows_int,K)=Dummy
1980      IF K=Col_int THEN GOTO Next_k
1990      Time_intervals(Rows_int,K)=Dummy_int
2000 Next_k:      !
2010    NEXT K
2020    END IF
2030 SUBEND
```

Programming Examples

To Make Continuous Timestamps on Channel 1 (Turbo C)

```
/*
 *                  DEMO PROGRAM
 *            Uses Radysis EPConnect library calls
 *
 * This program instructs the HP E1740A to take continuous time stamps
 * on Channel A. The number of timestamps to capture is one greater than
 * the number of measurements specified by the constant NUM_MEAS.
 * After the measurements are collected, they are plotted on the
 * computer CRT.
 *
 * To conserve memeory, all of the data is kept in unsigned integer
 * format in array results. To convert to time intervals, consecutive
 * time stamps are subtracted and then multiplied by the value tint_res.
 *
 * This program is written in Borland C and uses the Large memory model.
 */

/*
 ****************************************************************
 * Include files.
 ****************************************************************
 */
#include <conio.h>
#include <string.h>    /* Used for strcpy() and strcat() */
#include <stdio.h>     /* Used for printf() */
#include <dos.h>       /* Used for delay() */
#include <stdlib.h>    /* Used for exit() */
#include <graphics.h> /* Used for graphics */
#include <busmgr.h>    /* Used for EPC i/o */

/*
 ****************************************************************
 * Constants
 ****************************************************************
 */
#define ULA         (short)48
#define NOCLIP      0
#define NUM_MEAS    16000     /* Number of Time Interval measurements */

/*
 ****************************************************************
 * Local function declarations
 ****************************************************************
 */
static void  init_TIA(void);
static void  setup_TIA(void);
static void  measure(void);
static void  get_timestamps(void);
static void  calc_ti(void);
static void  plot_results(void);
/*
 * Utilities
 */
static void  sendTIA(char *hpib_cmd);
static void  chkTIA(void);
static void  wait_cmpl(void);
static float getTIAf(void);
static short getTIAs(char *reply, short length);
```

To Make Continuous Timestamps on Channel 1 (Turbo C) (Continued)

```
static void  swap_bytes(void);
static void  write_crt(int x,int y,int numdigits,char *text,float result);

/*
 ****************************************************************
 * Global data
 ****************************************************************
 */
unsigned int   *results;     /* Array of time stamps & time intervals */
int      numb_ts;     /* Total number of timestamps */
int      numb_ti;         /* Number of time intervals measured */
double     tint_res;    /* Time interval resolution */

/*
 ****************************************************************
 * Global function definitions
 ****************************************************************
 */
void
main(void)
{
  char a;

  clrscr();          /* Clear the computer CRT */
  init_TIA();             /* Initialize the measurement hardware */
  while (a != 'q')
  {
     setup_TIA();          /* Send measurement setup to hardware */
     measure();     /* Make measurement and wait to complete */
     get_timestamps();     /* Reads back the results */
     calc_ti();       /* Calculate time intervals */
     plot_results();       /* Scale and plot results */
     free(results);     /* Free memory */
     a = getch();       /* Press a key to end the program */
     closegraph();      /* Clear graphics */
     sendTIA("*rst");      /* Reset the TIA */
  }
}

/*
 ****************************************************************
 * Local function definitions.
 ****************************************************************
 */

/*
 * init_TIA(void)
 *
 * Initializes the TIA hardware.
 */
static void
init_TIA(void)
{
  sendTIA("*rst");       /* Reset the TIA */
  sendTIA("*cls");       /* Clear event registers and error queue */
  sendTIA("*sre 0");     /* Clear service request enable register */
```

To Make Continuous Timestamps on Channel 1 (Turbo C) (Continued)

```
  sendTIA("*ese 0");     /* Clear event status enable register */
}

/*
 * setup_TIA(void)
 *
 * Sends measurement setup to measurement hardware.
 */
static void
setup_TIA(void)
{
  char to_TIA[80];        /*String to send to TIA */

  sendTIA(":format int");
  sprintf(to_TIA, "conf:xtim:tst 0,%d,(@1)", NUM_MEAS+1); /* Timestamps */
  sendTIA(to_TIA);
  sendTIA(":inp1:coup dc");      /* Channel 1 DC coupled */
  sendTIA(":inp2:coup dc");      /* Channel 2 DC coupled */
  sendTIA(":inp1:imp 50");       /* Channel 1 50 ohm impedance */
  sendTIA(":inp2:imp 50");       /* Channel 2 50 ohm impedance */
  sendTIA(":event1:level 0");      /* Channel 1 0 volts trigger level */
  sendTIA(":event2:level 0");      /* Channel 2 0 volts trigger level */
  sendTIA(":event1:slope pos");    /* Channel 1 positive slope */
  sendTIA(":event2:slope pos");    /* Channel 2 positive slope */
  sendTIA(":acq:sour mcount");     /* Use meas count to terminate */
  sprintf(to_TIA, ":acq:mcount %d", NUM_MEAS+1); /* Number of meas */
  sendTIA(to_TIA);
  sendTIA(":trig:count 1");      /* Only make one set of meas */
  sendTIA(":acq:pac imm");       /* Start measurement immediately */
  sendTIA(":trig:sour imm");     /* Trigger immediately */
  sendTIA(":tint:range:res 50E-12"); /* Set resolution to 50 ps (min) */
}

/*
 * measure(void)
 *
 * Make measurement, and wait for completion.
 */
static void
measure(void)
{
  sendTIA("*ESE 1");     /* Use bit one of Event Status Register */
  sendTIA("*SRE 32");      /* Summarize Event Status in Status byte  */
  sendTIA("*OPC");       /* Use *OPC  */
  sendTIA("INIT");       /* Start the whole process */
  wait_cmpl();       /* Wait for measurement to complete */
}

/*
 * get_timestamps(void)
 *
 * Gets measurement results.
 */
static void
get_timestamps(void)
```

To Make Continuous Timestamps on Channel 1 (Turbo C) (Continued)

```
{
  int   dig_bytes;       /* Number of bytes in header */
  int   all_bytes;       /* Number of bytes of timestamp data */
  char  reply_head[3];  /* First two characters of header + lf */
  char  reply_bytes[8];   /* Field for number of measurements */
  short length;        /* Number of bytes to transfer */
  char  to_TIA[80];     /*String to send to TIA */
  char  dummy[1];
  char  *p;

  sendTIA(":tint:range:res?");  /* Get the true time resolution */
  tint_res = getTIAf();
  puts("Transferring and processing data");
  sprintf(to_TIA, "fetc? 0,%d", NUM_MEAS+1); /* Ask for results */
  sendTIA(to_TIA);
  length = 2;       /* Get first two bytes (#A) into reply_head */
  (void)getTIAs(reply_head, length);
  p = reply_head;
  p++;            /*Now, point to number of meas bytes */
  sscanf(p, "%1d", &dig_bytes);       /*Ask for a number of bytes */
  dig_bytes = getTIAs(reply_bytes, dig_bytes); /* dig_bytes is number
bytes */
  reply_bytes[dig_bytes] = '\0';
  sscanf(reply_bytes, "%u", &all_bytes);
  results = malloc(all_bytes+1);    /* Save a little extra space */
  printf("Bytes requested: %d\n", all_bytes);
  all_bytes = getTIAs((char *)results, all_bytes);  /* Get all data */
  printf("Bytes received: %d\n", all_bytes);
  getTIAs(dummy, 1);       /* Throw away line feed at end */
  numb_ts = all_bytes / 2;
  numb_ti = numb_ts - 1;
  swap_bytes();
}

/*
 * calc_ti(void)
 *
 * Calculates time intervals from raw time stamp data.
 */
static void
calc_ti(void)
{
  unsigned int *res_ptr;     /* Pointer to location in results array */
  unsigned int *res_max_ptr;  /* Pointer just past results data */
  unsigned int *res_next_ptr; /* Pointer to next timestamp */

  res_max_ptr = results + numb_ti;
  for (res_ptr=results, res_next_ptr=results+1;
    res_ptr < res_max_ptr;
    res_ptr++, res_next_ptr++)
  {
     if (*res_next_ptr > *res_ptr)    /* No overflow */
     {
     *res_ptr = *res_next_ptr - *res_ptr;
     }
     else                /* Correct for overflow */
```

To Make Continuous Timestamps on Channel 1 (Turbo C) (Continued)

```
      {
      *res_ptr = ~(*res_ptr - *res_next_ptr) + 1;
      }
   }
}

/*
 * plot_results(void)
 *
 * Scales and plots results.
 */
static void
plot_results(void)
{
  unsigned int array_max = 0;    /* For maximum time interval */
  unsigned int array_min = 65535; /* For minimum time interval */
  int       driver = DETECT;    /* Detect video system */
  int       mode;         /* Graphics mode */
  int       i;
  float     maxx,maxy,tempmaxx,tempmaxy;
  float     left,right,top,bot;
  float     temp_0,temp_1;
  float     total_time;
  float     yval,xval,xmin,xmax,ymin,ymax;
  char      textout[30];      /* String to place on graphics */

  /*
   * Add up time and find maximum and minimum values of data.
   */
  total_time=0;                 /* Start time = 0 */
  for (i=0; i<numb_ti; i++)
  {
     if (results[i] > array_max)
     {
     array_max=results[i];
     }
     if (results[i]<=array_min)
     {
      array_min=results[i];
     }
     total_time = total_time + results[i];
  }
  total_time = total_time * tint_res; /*Scale by measurement resolution */
  if (array_max == array_min)
  {
     printf("Array max and min are equal\n");
     return;
  }

  /*
   * Beginning of plot routine
   */
  initgraph(&driver,&mode,"c:\\borlandc\\bgi");
  maxx = getmaxx();       /* Get the max and min CRT coordinates */
  maxy = getmaxy();
  left = 0.15 * maxx;     /* Set up a nice coordinate system */
  top  = 0.05 * maxy;
```

To Make Continuous Timestamps on Channel 1 (Turbo C) (Continued)

```
   right = 0.9 * maxx;
   bot   = 0.9 * maxy;
   rectangle(left,top,right,bot);
   setviewport(left,top,right,bot,NOCLIP);

   /*
    * Convert the data so it can be easily plotted
    */
   temp_0=((top-bot)/(array_max-array_min))*(signed)(results[0]-array_max);
   for (i=1;i<NUM_MEAS;i++)
   {
      temp_1 = (   (top-bot)
            / (array_max-array_min)) * (signed)(results[i]-array_max);
      line((i)*(right-left)/numb_ti,
      temp_0,
      (i+1)*(right-left)/numb_ti,
      temp_1);
      temp_0 = temp_1;
   }
   settextstyle(TRIPLEX_FONT,HORIZ_DIR,2);
   strcpy(textout,"Time Interval A");
   settextjustify(2,2);
   setviewport(0,0,maxx,maxy,NOCLIP);
   settextjustify(0,2);
   moveto (.09*maxx,1);
   outtext (textout);
   write_crt(.15*maxx,.91*maxy,1,"",0);
   write_crt(.80*maxx,.91*maxy,4,"",total_time);
   write_crt(0,.04*maxy,3,"",array_max*tint_res);
   write_crt(0,.86*maxy,3,"",array_min*tint_res
   );
}
/*
 * Utility definitions
 */

/*
 * sendTIA (char *hpib_cmd)
 *
 * Sends command string at hpib_cmd to TIA.
 */
static void
sendTIA(char *hpib_cmd)
{
  short         length;             /* Length of message to send */
  short       sent_len;      /* Length actually sent */
  unsigned short      wait = 2000; /* Wait in milliseconds */
  short         error;

  length = strlen(hpib_cmd);      /*Get length of command */
  error = EpcWsSndStr(ULA,hpib_cmd,length,&sent_len,wait); /* To TIA */
  if (error != EPC_SUCCESS)
  {
   printf("Error: %Fs\n  sending %Fs\n",EpcErrStr(error),hpib_cmd);
  }
}
```

Programming Examples

To Make Continuous Timestamps on Channel 1 (Turbo C) (Continued)

```
#if 0  /*  REVISIT */
/*
 * chkTIA(void)
 *
 * Reads from TIA to ensure that previous commands have
 * been acted upon before proceeding.
 */
static void
chkTIA(void)
{
   char message[80];
   int  length;

   sendTIA(":SYST:ERR?");
   length = getTIAs(message, 79);
   message[length] = '\0';
   printf("chkTIA: %Fs\n", message);
}
#endif

/*
 * wait_cmpl(void)
 *
 * Waits for the TIA to complete
 */
static void
wait_cmpl(void)
{
   char temp[80];
   int  result;

   puts("Waiting for measurement to complete");
   do
   {
    sendTIA(":STAT:OPER:ENAB 16");
    delay(100);
    sendTIA(":STAT:OPER:COND?");
    delay(100);
    (void)getTIAs(temp, 79);
    result = atoi(temp);
    delay(300);
   }
   while (result & 16);     /* if =1 then measurement complete */
}

/*
 * getTIAf(void)
 *
 * Gets a single result of type float
 */
static float
getTIAf(void)
{
  char    reply[24];     /* Buffer for reply */
  short   rcd_len;     /* Length received */
  short   length = 23;
  unsigned short wait = 2000;       /* Wait in milliseconds */
  short     error;
```

To Make Continuous Timestamps on Channel 1 (Turbo C) (Continued)

```
  double  answer;     /*Return a double */

  error = EpcWsRcvStr(ULA, reply, length, &rcd_len, wait);
  if (error != EPC_SUCCESS)
  {
   printf("Error: %Fs\n  during TIA query\n",EpcErrStr(error));
  }
  sscanf(reply,"%lf",&answer);     /*Convert to double precision */
  return (answer);           /*Return answer as double */
}

/*
 * getTIAs(char *reply, short length)
 *
 * Reads a string of length characters from TIA into reply.
 * Returns the number of bytes received.
 */
static short
getTIAs(char *reply, short length)
{
  short      rcd_len;     /* Length received */
  unsigned short wait = 2000;          /* Wait in milliseconds */
  short      error;

  error = EpcWsRcvStr(ULA, reply, length, &rcd_len, wait);
  if ((error != EPC_SUCCESS) && (error != ERR_BUFFER_FULL))
  {
   printf("Error: %Fs\n  during TIA query\n",EpcErrStr(error));
  }
  return (rcd_len);
}

/*
 * swap_bytes(void)
 *
 * The two bytes making up each integer are swapped when initially
 * read back.  This function swaps them back into the correct order.
 */
static void
swap_bytes(void)
{
  unsigned int *res_ptr;     /* Pointer to location in results array */
  unsigned int *res_max_ptr;  /* Pointer just past results data */

  res_max_ptr = results + numb_ts;
  for (res_ptr=results; res_ptr < res_max_ptr; res_ptr++)
  {
     *res_ptr = ((*res_ptr & 0xff00) >> 8) | ((*res_ptr & 0xff) << 8);
  }
}

/*
 * write_crt (int x, int y, int numdigits, char *text, float result)
 *
 * Function to put text and numbers onto CRT
 */
```

Programming Examples

To Make Continuous Timestamps on Channel 1 (Turbo C) (Continued)

```
static void
write_crt(int x, int y, int numdigits, char *text, float result)
{
   char textstring[30], newtext[30];

   moveto(x,y);
   strcpy(textstring,text);
   if (numdigits==0)
   {
    outtext(textstring);
   }
   else
   {
    gcvt(result,numdigits,newtext);
    strcat(textstring,newtext);
    outtext(textstring);
   }
}
```

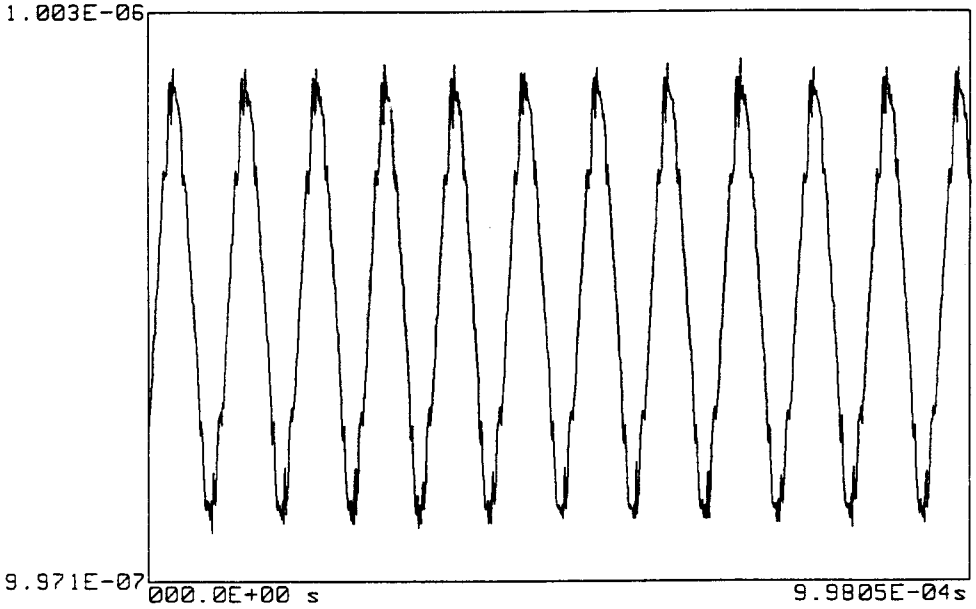


Figure 15-3. ASCII Transfer of Versus Time Data (Turbo C)

16

Maintenance

Service Information

Introduction

This chapter provides the information needed to properly maintain the HP E1740A. The following information and test procedures are included:

- HP E1740A Overview
- Performance tests
- Calibration Procedures
- Troubleshooting and Diagnostic tests
- Instrument Disassembly and Reassembly Procedures
- Replaceable Parts

HP E1740A Overview

The overview describes the configurations of the HP E1740A and the servicing strategies used for each. This section also provides a list of the equipment required for performance tests, calibration, and troubleshooting procedures included in this chapter.

Performance Tests

The performance tests verify warranted specifications. All tests can be performed without access to the inside of the instrument.

Calibration Procedures

The calibration procedures allow fine adjustment of the instrument. Depending on the use and environment conditions, the HP E1740A performance tests and calibration procedures should be completed at least once every year.

Troubleshooting and Diagnostic Tests

In case of an instrument fault, the troubleshooting and diagnostic tests contain the steps necessary to isolate the problem down to the board level. Three boards make up the HP E1740A product.

Instrument Disassembly and Reassembly Procedures

Once a fault has been located, the instrument disassembly and reassembly procedures give instructions on how to extract the suspected module.

Replaceable Parts

Lists all replaceable items in the HP E1740A. These are orderable from the HP parts sources listed.

HP E1740A Overview

The HP E1740A can be used as part of any VXI system for precision time interval measurements. The HP E1740A is also furnished as a component of the HP E1725A Time Interval Analyzer. The HP E1725A also includes the HP E1741A software that provides a graphical user interface and analysis features. The HP E1741A software also provides easy access to the maintenance and service elements.

The Hewlett-Packard E1725A can be returned to Hewlett-Packard for all service work, including troubleshooting, adjusting, calibrating, or verifying specifications. Contact your nearest HP Sales and Service Office for more details.

However, you can perform some simple system-level troubleshooting and isolate the faulty HP E1740A module, and return just that module to Hewlett-Packard for service work. The receipt of the entire HP E1725A will give the service technician the ability of accessing the service and maintenance items via the HP E1741A software. Receipt of only the HP E1740A will require the use of a locally owned PC loaded with the required software, or another computer setup to execute SCPI commands in a VXI environment at the service site.

This chapter provides instructions for using SCPI commands and the HP E1741A software when performing the test procedures to meet the requirements of both the HP E1740A standalone and HP E1725A configurations. The HP E1741A software provides for a much friendlier user interface, and requires a PC equipped with DOS 5.0 (or later) and Windows 3.1. Other computer environments are possible as well, but are not documented in this chapter. HP 9000 machines equipped to run Rocky Mountain Basic can also be used to run the HP E1740A using SCPI commands.

Assumptions

The HP E1740A is VXI compatible and SCPI programmable. It is assumed the service technician is familiar with this type of product. Also, knowledge of PCs using DOS and Windows is assumed, as well as knowledge of the HP 9000 series machine that is used with Rocky

Mountain Basic (RMB). If a PC equipped with IBASIC and associated hardware is used, the service technician will also need knowledge in its operation. VXI and HP-IB interfacing knowledge is also assumed, since the programs enclosed here may require addressing changes to run, depending on the computer and interface setup.

For addressing information, refer to the Chapter 1,"Getting Started," in this guide.

The following program allows an easy SCPI interface to the HP E1740A using something other than the HP E1741A software.

```
10    Dim send$[256], return$[256]
20    assign @HPE1740A to 70906
30    input "SCPI control string",send$
40    disp send$
50    output @HPE1740A; send$
60    if pos(send$,"?") then
70    enter @HPE1740A;return$
80    print return$
90    end if
100   goto 30
110   end
```

Test Record

The results of the performance and calibration test should be recorded on the Performance Test Record, located at the end of the "Performance Tests of Warranted Specifications" section in this chapter. Make copies of this form as needed.

Required Equipment

Equipment required for the performance tests in this chapter is listed in Table 16-1. Any equipment that satisfies the critical specification listed in this table may be substituted for the recommended model(s).

Table 16-1. Required Equipment

Item	Characteristics	Use	Recommended Model
Computer **OR**	Embedded VXI module DOS 5.0 (or higher) Windows 3.1 HP E1741A software	P, C, T	Radisys EPC7
Computer **OR**	PC, or compatible, 386 with co-processor, or 486. 4 MB RAM (minimum) Hard disk Keyboard, Mouse, Display DOS 5.0 (or higher) Windows 3.1 IBasic * E1741A software HPIB interface	P, C, T	HP Vectra HP 82231
Computer	HP 9000 Series Basic HPIB interface	P, C, T	
VXI Card Cage	Five slot minimum	P, C	HP 75000 Series C or Mac Panel 12470-02 (HP-E1725-00001)
VXI Card Cage	Open sided for troubleshooting access	T	HP E1400T
Slot 0 Card	Only required if EPC-7 is not used	P, C, T	HP E1406A
Synthesized Signal Generator	10 MHz to 2.5 GHz Very low noise output	P, C, T	HP 8663A
Pulse Generator	100 KHz TTL output	T	HP 5359A HP 8161A HP 8130A /8131A HP 3325A

P = Performance Tests
C = Calibration
T = Troubleshooting/Diagnostics

* Used for direct SCPI control if the HP E1741 software is not available.

Table 16-1. Required Equipment (Continued)

Item	Characteristics	Use	Recommended Model
Function Generator	1 KHz Sinewave @ 1 V, Offset from ground @ 1 V	T	HP 3325A
Oscilloscope	100 MHz Frequency response	T	HP54600
Digital Multimeter	DC, AC, Ohms	C, T	HP 3458A
DC Power Supply	Adjustable to 5.000 VDC	C	HP 6216C
Frequency Standard	Absolute accuracy $<1x10^{-10}$	C	HP 5061B or HP 5071A
Power Splitter	50 Ohms	P, C	HP 11667A
N-BNC adapter Qty 3	50 Ohms	P, C	HP 1250-0071
BNC 9" (23 cm), Cable Qty 2	50 Ohms	P, C	HP 10502A
P = Performance Tests C = Calibration T = Troubleshooting/Diagnostics			

Operational Verification

Apply power to the HP E1740A.

If power-up completes with no **error** or **failed** LEDs on, then operational verification is complete.

If **error** or **failed** LEDs are on after power-up, a hardware problem is indicated. Refer to the "Troubleshooting and Diagnostics" section in this chapter.

Performance Tests of Warranted Specifications

This section includes the tests for verifying the specifications of the HP E1740A. The specifications of the HP E1740A are as follows:

- Hardware resolution, or least significant bit: 50 ps (Test 1)
- RMS timing jitter: <100 ps (Test 2)
- Minimum Time Interval 1->2 ≤ 0 (Test 3)
- Minimum Time Interval 1->1 or 2->2 (Test 4A)
- Minimum Time Interval Ch 1, <12.5 ns (Test 4B)
- Maximum Time Interval 1->1 or 2->2 (Test 5)
- Latency After Time Interval Measurement (Input 1-> 2) (Test 6)

Test Instructions

In all of the following test procedures, two methods are listed. One method assumes that the HP E1741A software is available. The other method assumes that the software is not available and the tests are performed via SCPI program listings.

The HP E1741A setup is explained from a "cold start." That is, the instrument has been turned on. Windows has been loaded, and the HP E1741A TIA software has been started.

Test 1—Hardware resolution, or least significant bit: 50 ps

Hardware resolution is set by design of the HP E1740A. It is related to the clock frequency and the interpolation circuits and equals a value of 12.5 ns / 256 bins, or approximately 48.8 ps. On the HP E1741 screens, a value of 50 ps is shown, and the specification also relates this as 50 ps. There is no test for this specification.

Test 2—RMS timing jitter: < 100 ps

This test provides the best overall performance view of the HP E1740A.

Perform Powerup and Complete External Connections

1 **Turn on the equipment under test and the source(s).**

Key source specification is very low noise. The HP 8663A specified has extremely low output noise. Be sure all equipment has at least 30 minutes to warm-up.

2 **Connect the source to the HP E1740A Input 1.**

3 **Connect the source 10 MHz reference out to the HP E1740A ref input.**

This locks the time base references, which is important to get correct test results.

4 **Set the source output as follows: 67.890123 MHz, +13 dbm output.**

Instrument Setup

NOTE

When running the SCPI programs set the HPIB/TIA address as required.

HP E1740 Setup:

Send the SCPI commands listed in the program at the end of this test specification.

HP E1741 Setup:

1 **From the menu bar, select Control>TIA Setup (expanded) and select from the Time Interval Measurement box:**

Source: Input 1 only
Record mode: fast histogram

2 **In the Acquire area of the TIA Setup window set the number of measurements to 60,000,000.**

3 **From the menu bar, select Timebase>Lock to external.**

4 **From the TIA Setup window, select the Input button.**

5 **Set both inputs to 50 ohms input impedance. (Even though only Input 1 is being used here, Input 2 will be used in subsequent tests.)**

6 **Close the Input window and select the Span & Resolution button on the TIA Setup window.**

7 **Set the Span and Resolution (Histogram) to 100 ns, 50 ps. Click OK.**

8 **From the menu bar, select Display>Histogram.**

9 **Adjust the size and position of the display as desired for easy viewing.**

Run the Test

1 **On the toolbar press the Run button.**

Display window should show a histogram.

2 **On the toolbar press the Stop button to halt the measurement process.**

3 **In the Histogram window, move the markers so they bracket the light blue area in the narrow bar just below the statistics (Otherwise, no statistics will be calculated) and read the results.**

Verifying the Results

1 **For the HP E1740A alone, the following reading should be visible:**

Std Dev: < 100 ps

2 **For the HP E1741A in use, the following readings should be visible:**

Std Dev: < 100 ps

3 **Record results on test record card.**

Action on Failure

1 **Go to the "Calibration" section of this chapter and calibrate the instrument.**

2 **Repeat the failed performance test.**

3 **If failure continues, a hardware problem is indicated. Perform the tests in the "Troubleshooting and Diagnostics" section in this chapter.**

SCPI/IBASIC Program Listing for Test 2 RMS Jitter

```
10     !TEST 2 RMS jitter
20     !File name: SPEC2
30     !Rev date: 7/01/93
40     !
50     !1. OVERALL SETUP
60     !
70     !Set up interface, reset, clear status, enable
80     !service requests, event status enable, format integer
90     !Note: Set interface assignments as required for setup in
use.
100     ASSIGN @Tia TO 70906
110     OUTPUT @Tia;"*RST"
120     OUTPUT @Tia;"*CLS"
130     OUTPUT @Tia;"*SRE 0"
140     OUTPUT @Tia;"*ESE 0"
150     OUTPUT @Tia;"format int"
160    !
170    !Set time base reference to external
180     OUTPUT @Tia;"ROSC:SOUR EXT"
190    !
200    !Set up for fast histogram time interval
210     OUTPUT @Tia;"CONF:XTIN:HIST DEF,DEF,(@1),(@1)"
220     OUTPUT @Tia;"SENS:EVENT1:HYST:REL
       50;:EVENT1:LEV 0"
230     OUTPUT @Tia;"SENS:EVENT2:HYST:REL
       50;:EVENT2:LEV 0"
240     OUTPUT @Tia;"TRIG:START:SOUR IMM"
250     OUTPUT @Tia;"SENS:ACQ:PAC:SOUR IMM"
260     OUTPUT @Tia;"SENS:ACQ:MCOUNT 6e7"
270    !
280    !Set the input conditions
290     OUTPUT @Tia;"INP1:IMP 50;:INP2:IMP
       50;:INP:ROUT SEP"
300     OUTPUT @Tia;"EVENT1:SLOPE POS;:EVENT2:SLOPE POS"
310    !
320    !Specify the histogram resolution
330     OUTPUT @Tia;"SENSE:HIST:RANG:RES 50pS"
340    !
350    !2. MAKE THE MEASUREMENT
360    !
370     OUTPUT @Tia;"init"
380    !
390    !3. FETCH, ENTER, AND PRINT THE RESULTS
400    !
410     OUTPUT @Tia;"FETCH:TINT:SDEV?"
420     WAIT 10
430     ENTER @Tia;Sdev
440     PRINT "SDEV = ",Sdev
450     GOTO 370
460     END
```

Test 3—Minimum Time Interval 1->2 ≤ 0

This test checks for the ability of the HP E1740A to resolve the time interval between two signals down to zero time.

Perform Powerup and Complete External Connections

1 **Turn on the equipment under test and the source(s).**

Key source specification is very low noise. The HP 8663A specified has extremely low output noise. Be sure all equipment has at least 30 minutes to warm-up.

2 **Connect the source to the HP E1740A Inputs 1 and 2, as follows:**

a. **Connect the source to the input of a power splitter (as recommended in the test equipment list, or an equivalent unit).**

b. **Connect the splitter outputs to the Inputs 1 and 2 using short equal length cables.**

The cables recommended in the test equipment list are 9" (23 cm) long. If the recommended splitter is used, the listed N/BNC adapters will be needed for the BNC cables.

3 **Connect the source 10 MHz reference out to the HP E1740A ref input.**

This locks the time base references, which is important to get correct test results.

4 **Set the source output as follows: 30 MHz, +13 dbm output.**

Instrument Setup

HP E1740 Setup:

Send the SCPI commands listed in the program at the end of this section.

HP E1741 Setup:

1 **From the menu bar, select Control>TIA setup (expanded) and select from the Time Interval Measurement box:**

Source: Input 1 to 2
Record mode: fast histogram

2 **In the Acquire area of the TIA Setup window set the number of measurements to 2,000.**

3 **From the menu bar, select Timebase>Lock to external.**

4 **From the TIA Setup window, select the Input button.**

5 **Set both inputs to 50 ohms input impedance.**

6 **Close the Input window and select the Span & Resolution button on the TIA Setup window.**

7 **Set the Min TI to -590 ps. This requires use of both the up and down arrows.**

8 **Set the Span and Resolution (Histogram) to 100 ns, 50 ps. Click OK.**

9 **From the menu bar, select Display>Histogram.**

10 **Adjust the size and position of the display as desired for easy viewing.**

Run the Test

1 **On the toolbar press the Run button.**

Display window should show a histogram.

2 **On the toolbar press the Stop button to halt the measurement process.**

3 **In the Histogram window, move the markers so they bracket the light blue area in the narrow bar just below the statistics (Otherwise, no statistics will be calculated) and read the results.**

Verifying the Results

1 **For the HP E1740A alone, the following readings should be visible:**

Min: ≤ 0

2 **For the HP E1741A in use, the following readings should be visible:**

Min: ≤ 0

3 **Record results on test record card.**

Action on Failure

1 **Go to the "Calibration" section of this chapter and calibrate the instrument.**

2 **Repeat the failed performance test.**

3 **If failure continues, a hardware problem is indicated. Perform the tests in the "Troubleshooting and Diagnostics" section in this chapter.**

SCPI/IBASIC Program Listing for Test 3 Minimum Time Interval 1->2

```
10    !TEST of Spec 3 Minimum Time Interval 1->2
20    !File Name: SPEC3
30    !Rev date: 7/01/93
40    !
50    !1. OVERALL SETUP
60    !
70    !Setup interface; reset and clear status, enable
80    !service requests, event status enable, format integer
90    !Note: Set interface assignments as required for setup in
use.
100    ASSIGN @Tia TO 70906
110    OUTPUT @Tia;"*RST"
120    OUTPUT @Tia;"*CLS"
130    OUTPUT @Tia;"*SRE 0"
140    OUTPUT @Tia;"*ESE 0"
150    OUTPUT @Tia;"format int"
160   !
170   !Set time base reference to external
180    OUTPUT @Tia;"ROSC:SOUR EXT"
190   !
200   !Set up for fast histogram time interval
210    OUTPUT @Tia;"CONF:XTIN:HIST DEF,DEF,(@1),(@2)"
220    OUTPUT @Tia;"SENS:EVENT1:HYST:REL 50;:EVENT1:LEV 0"
230    OUTPUT @Tia;"SENS:EVENT2:HYST:REL 50;:EVENT2:LEV 0"
240    OUTPUT @Tia;"TRIG:START:SOUR IMM"
250    OUTPUT @Tia;"SENS:ACQ:PAC:SOUR IMM"
260    OUTPUT @Tia;"SENS:ACQ:MCOUNT 2000"
270   !
280   !Set the input conditions
290    OUTPUT @Tia;"INP1:IMP 50;:INP2:IMP 50;:INP:ROUT SEP"
300    OUTPUT @Tia;"EVENT1:SLOPE POS;:EVENT2:SLOPE POS"
310   !
320   !Specify the histogram resolution
330    OUTPUT @Tia;"SENSE:HIST:RANG:RES 50PS"
340   !
350   !2. MAKE THE MEASUREMENT
360   !
370    OUTPUT @Tia;"init"
380   !
390   !3. FETCH, ENTER, AND PRINT THE RESULTS
400   !
410    OUTPUT @Tia;"FETCH:TINT:MIN?"
420    WAIT 5
430    ENTER @Tia;Min
440    PRINT "MIN = ",Min
450    GOTO 370
460    END
```

Test 4—Minimum Time Interval 1->1 or 2->2

This specification contains two procedures as follows:

Minimum TI 1 ->1 or 2->2

- Test 4A—12.5 ns continuous (every edge on 80 MHz signal) for Inputs 1 and 2.
- Test 4B—8 ns non-continuous (2 adjacent edges followed by a delay on a 125 MHz signal) for Input 1 only.

Test 4A—Continuous 12.5 ns Measurements for Inputs 1 and 2

After you have completed this procedure for Input 1 perform this procedure again to verify Input 2.

Perform Power-up and Complete External Connections

1 **Turn on the equipment under test and the source(s).**

Key source specification is very low noise. The HP 8663A specified has extremely low output noise. Be sure all equipment has at least 30 minutes to warm-up.

2 **Connect the source to the HP E1740A Input 1 (Input 2).**

If splitter and cables are still in place from previous test, just leave it. Both inputs will be tested. The setup changes shown below will cause the input signal to be sensed at the proper input jack.

3 **Connect the source 10 MHz reference out to the HP E1740A *10 MHz Ref. In* input.**

This locks the time base references, which is important to get correct test results.

4 **Set the source output as follows: 79.3 MHz, +13 dbm.**

The period of this signal is 12.6 ns which allows for some source noise on the output signal.

Instrument Setup

HP E1740 Setup:

Send the SCPI commands listed in the program at the end of this section.

HP E1741 Setup:

1 **From the menu bar, select Control>TIA setup (expanded) and select from the Time Interval Measurement box:**

Source: Input 1 only (or Input 2 only)
Record mode: fast histogram

2 **In the Acquire area of the TIA Setup window set the number of measurements to 2,000.**

3 **From the menu bar, select Timebase>Lock to external.**

4 **From the TIA Setup window, select the Input button.**

5 **Set both inputs to 50 ohms input impedance.**

6 **Close the Input window and select the Span & Resolution button on the TIA Setup window.**

7 **Set the Span and Resolution (Histogram) to 100 ns, 50 ps. Click OK.**

8 **From the menu bar, select Display>Histogram.**

9 **Adjust the size and position of the display as desired for easy viewing.**

Run the Test

1 **On the toolbar press the Run button.**

Display window should show a histogram.

2 **On the toolbar press the Stop button to halt the measurement process.**

3 **In the Histogram window, move the markers so they bracket the light blue area in the narrow bar just below the statistics (Otherwise, no statistics will be calculated) and read the results.**

Verifying the Results

1 **For the HP E1740A alone, the following readings should be visible:**

Min: 12.5 ns or lower

2 **For the HP E1741A in use, the following readings should be visible:**

Min: 12.5 ns or lower

3 **Record results on test record card.**

4 **Repeat this test for Input 2.**

Action on Failure

1 **Go to the "Calibration" section of this chapter and calibrate the instrument.**

2 **Repeat the failed performance test.**

3 **If failure continues, a hardware problem is indicated. Perform the tests in the "Troubleshooting and Diagnostics" section in this chapter.**

SCPI/IBASIC Program Listing for Test 4— Minimum Time Interval Ch 1->1 or Ch 2 ->2

```
10     !TEST of Spec 4A: Minimum Time Interval Ch 1->1 or Ch 2 ->2
20     !File name: SPEC4A
30     !Rev date: 7/01/93
31     !
40     !1. OVERALL SETUP
41     !
50     !Setup interface; reset and clear status, enable
60     !service requests, event status enable, format integer
61     !Note: Set interface assignments as required for setup in
use.
70      ASSIGN @Tia TO 70906
80      OUTPUT @Tia;"*RST"
90      OUTPUT @Tia;"*CLS"
100     OUTPUT @Tia;"*SRE 0"
110     OUTPUT @Tia;"*ESE 0"
120     OUTPUT @Tia;"format int"
130    !
140    !Set time base reference to external
150     OUTPUT @Tia;"ROSC:SOUR EXT"
160    !
170    !Set up for fast histogram time interval
180     OUTPUT @Tia;"CONF:XTIN:HIST DEF,DEF,(@1),(@1)"*
190     OUTPUT @Tia;"SENS:EVENT1:HYST:REL 50;:EVENT1:LEV 0"
200     OUTPUT @Tia;"SENS:EVENT2:HYST:REL 50;:EVENT2:LEV 0"
210     OUTPUT @Tia;"TRIG:START:SOUR IMM"
220     OUTPUT @Tia;"SENS:ACQ:PAC:SOUR IMM"
230     OUTPUT @Tia;"SENS:ACQ:MCOUNT 2000"
240    !
250    !Set the input conditions
260     OUTPUT @Tia;"INP1:IMP 50;:INP2:IMP 50;:INP:ROUT SEP"
270     OUTPUT @Tia;"EVENT1:SLOPE POS;:EVENT2:SLOPE POS"
280    !
290    !Specify the histogram resolution
300     OUTPUT @Tia;"SENSE:HIST:RANG:RES 50PS"
310    !
320    !2. MAKE THE MEASUREMENT
321    !
330     OUTPUT @Tia;"init"
340    !
350    !3. FETCH, ENTER, AND PRINT THE RESULTS
351    !
360     OUTPUT @Tia;"FETCH:TINT:MIN?"
370     WAIT 5
380     ENTER @Tia;Min
390     PRINT "MIN = ",Min
400     GOTO 330
410     END
```

* For Ch 2 -> 2 tests, change (@1) to (@2) in two places.

Test 4B—Minimum Time Interval Ch 1, <12.5 ns

This specification checks the ability of the HP E1740A to measure time intervals below 12.5 ns, down to 8 ns. A 125 MHz signal is used to test this capability.

External Connections

1 **Set the source output as follows: 125 MHz, +13 dbm.**

The period of this signal is 8 ns.

2 **Connect source to Input 1.**

Instrument Setup

HP E1740 Setup:

Send the SCPI commands listed in the program at the end of this section.

HP E1741 Setup:

- **From the menu bar, select Control>TIA setup (expanded) and select from the Time Interval Measurement box:**

Source: Input 1 only (<12.5 ns)

Run the Test

1 **On the toolbar press the Run button.**

Display window should show a histogram.

2 **On the toolbar press the Stop button to halt the measurement process.**

3 **In the Histogram window, move the markers so they bracket the light blue area in the narrow bar just below the statistics (Otherwise, no statistics will be calculated) and read the results.**

Verifying the Results

1 **For the HP E1740A alone, the following readings should be visible :**

Minimum: 8.0 ns or less

2 **For the HP E1741A in use, the following readings should be visible if all is set properly:**

Minimum: 8.0 ns or less

3 **Record results on test record card.**

Action on Failure

1 **Go to the "Calibration" section of this chapter and calibrate the instrument.**

2 **Repeat the failed performance test.**

3 **If failure continues, a hardware problem is indicated. Perform the tests in the "Troubleshooting and Diagnostics" section in this chapter.**

SCPI/IBASIC Program Listing for Test 4—Minimum Time Interval Ch 1, <12.5 ns

```
10     !TEST 4B: Minimum Time Interval Ch 1, <12.5ns
20     !File name: SPEC4B
30     !Rev date: 7/01/93
40     !
50     !1. OVERALL SETUP
60     !
70     !Setup interface; reset and clear status, enable
80     !service requests, event status enable, format integer
90     !Note: Set interface assignments as required for setup in
use.
100     ASSIGN @Tia TO 70906
110     OUTPUT @Tia;"*RST"
120     OUTPUT @Tia;"*CLS"
130     OUTPUT @Tia;"*SRE 0"
140     OUTPUT @Tia;"*ESE 0"
150     OUTPUT @Tia;"format int"
160    !
170    !Set time base reference to external
180     OUTPUT @Tia;"ROSC:SOUR EXT"
190    !
200    !Set up for fast histogram time interval
210     OUTPUT @Tia;"CONF:XTIN:HIST DEF,DEF,(@1),(@2)"
220     OUTPUT @Tia;"SENS:EVENT1:HYST:REL 50;:EVENT1:LEV 0"
230     OUTPUT @Tia;"SENS:EVENT2:HYST:REL 50;:EVENT2:LEV 0"
240     OUTPUT @Tia;"SENS:TST:NDEL ON"
250     OUTPUT @Tia;"TRIG:START:SOUR IMM"
260     OUTPUT @Tia;"SENS:ACQ:PAC:SOUR IMM"
270     OUTPUT @Tia;"SENS:ACQ:MCOUNT 2000"
280    !
290    !Set the input conditions
300     OUTPUT @Tia;"INP1:IMP 50;:INP2:IMP 50;:INP:ROUT SEP"
310     OUTPUT @Tia;"EVENT1:SLOPE POS;:EVENT2:SLOPE POS"
320    !
330    !Specify the histogram resolution
340     OUTPUT @Tia;"SENSE:HIST:RANG:RES 50PS"
350    !
360    !2. MAKE THE MEASUREMENT
370    !
380     OUTPUT @Tia;"init"
390    !
400    !3. FETCH, ENTER, AND PRINT THE RESULTS
410    !
420     OUTPUT @Tia;"FETCH:TINT:MIN?"
430     WAIT 5
440     ENTER @Tia;Min
450     PRINT "MIN = ",Min
460     GOTO 380
470     END
```

Test 5—Maximum Time Interval 1->1 or 2->2

There is no test for this specification that verifies the maximum time interval that can be measured. This value is *fixed* by the hardware design, and is a product of the maximum width of the count chain and the resolution selected.

The histogram acquisition circuit is 11 bits wide, while the sequential acquisition circuit is 16 bits wide. The relationships between the resolution and max TI for either mode of acquisition may be viewed by opening the TIA Setup window, and then opening the Span and Resolution window. That window is divided into Histogram and Sequential areas, in which you can view the respective spans and resolutions.

Histogram acquisition: At the finest resolution of 50 ps, the largest time interval that may be recorded 100 ns. At the resolution of 400 ns, the largest time interval capture is 819 µs.

Sequential acquisition: With the wider 16 bit circuit, the sequential mode can measure longer time intervals. At 50 ps resolution, the largest is 3.2 us. At the other end of the range, with 400 ns resolution, a 26.2 ms interval can be measured. Thus, the largest time intervals are captured using the sequential mode of acquisition.

- Time Interval Range–3.2 µs to 26.2 ms

There is no test needed for this specification, as it is fixed by the hardware design.

Test 6—Latency After Time Interval Measurement (Input 1 -> 2)

- Latency after time interval measurement Input 1 -> 2: 30 ns.

This test checks the ability of the HP E1740A to capture the next edge after measuring a time interval input 1 to input 2, as long as that next edge is at least 30 ns later.

Perform Powerup and Complete External Connections

1 **Turn on the equipment under test and the source(s).**

Key source specification is very low noise. The HP 8663A specified has extremely low output noise. Be sure all equipment has at least 30 minutes to warm-up.

2 **Connect the source to Input 1 and 2, as follows:**

a. Connect the source to the input of a power splitter (as recommended in the test equipment list, or an equivalent unit).

b. Connect the splitter outputs to the Input 1 and 2 using short equal length cables.

The cables recommended in the test equipment list are 9" (23 cm) long. If the recommended splitter is used, the listed N/BNC adapters will be needed for the BNC cables.

3 **Connect the source 10 MHz reference out to the HP E1740A ref input.**

This locks the time base references, which is important to get correct test results.

4 **Set the source output as follows: 33.5 MHz, +13 dbm output.**

The period is slightly less than 30 ns, the specification.

Instrument Setup

HP E1740 Setup:

Send the SCPI commands listed in the program at the end of this section.

HP E1741 Setup:

1 **Close the histogram display, if it is open.**

Double click in box at upper left hand corner.

2 **From the menu bar, select Timebase>Lock to External.**

3 **From the menu bar, select Control>TIA Setup (expanded) and select from the Time Interval Measurement box:**

Source: Input 1 to Input 2
Record mode: sequential

4 **In the Acquire area of the TIA Setup window set the number of measurements to 5.**

5 **From the TIA Setup window, select the Input button.**

5 **Set both inputs to 50 ohms input impedance.**

6 **Close the Input window and the TIA Setup window.**

7 **From the menu bar, select Display>Versus-Time.**

8 **Adjust the size and position of the display as desired for easy viewing.**

Run the Test

1 **On the toolbar press the Single button.**

The Versus-Time window might show a trace. Messages at the bottom of the display will flash, indicating activity.

2 **From the menu bar, select Versus-Time>Scale to Full View.**

The display should show a jagged line with several small circles or dots along it. Each circle is a measurement. Using the mouse, position one marker on the left most visible circle, which is probably at the left edge of the display. Position the other marker on the next circle to the right (Place the markers carefully, as mis-placement will affect test results). Read the delta X or time between the measurements, at the bottom of the display.

Verifying the Results

1 **For the HP E1740A alone, the following readings should be visible :**

Delta: 30 ns or less

2 **For the HP E1741A in use, the following readings should be visible:**

Average Latency: 30 ns or less

3 **Record results on test record card.**

Action on Failure

1 **Go to the "Calibration" section of this chapter and calibrate the instrument.**

2 **Repeat the failed performance test.**

3 **If failure continues, a hardware problem is indicated. Perform the tests in the "Troubleshooting and Diagnostics" section in this chapter.**

SCPI/IBASIC Program Listing for Test 6 — Latency after TI 1>2

```
10     !TEST 6:Latency after TI 1>2
20     !File name: SPEC 6
30     !Rev date: 7/01/93
40     !
50     !1. OVERALL SETUP
60     !
70     !Setup interface; reset and clear status, enable
80     !service requests, event status enable, format ASCII.
90     !NOTE: Set interface assignments as required for setup in
use.
100     DIM A$[20]
110     ASSIGN @Tia TO 70906
120     OUTPUT @Tia;"*RST"
130     OUTPUT @Tia;"*CLS"
140     OUTPUT @Tia;"*SRE 0"
150     OUTPUT @Tia;"*ESE 0"
160     OUTPUT @Tia;"format asc"
170    !
180    !Set time base reference to external
190     OUTPUT @Tia;"ROSC:SOUR EXT"
200    !
210    !Set up for sequential time stamp measurement
220     OUTPUT @Tia;"CONF:XTIM:TST DEF,DEF,(@1),(@2)"
230     OUTPUT @Tia;"SENS:EVENT1:HYST:REL 50;:EVENT1:LEV 0"
240     OUTPUT @Tia;"SENS:EVENT2:HYST:REL 50;:EVENT2:LEV 0"
250     OUTPUT @Tia;"TRIG:START:SOUR IMM"
260     OUTPUT @Tia;"SENS:ACQ:PAC:SOUR IMM"
270     OUTPUT @Tia;"SENS:ACQ:SOUR MCOUNT;MCOUNT 50"
280    !
290    !Set the input conditions
300     OUTPUT @Tia;"INP1:IMP 50;:INP2:IMP 50;:INP:ROUT SEP"
310     OUTPUT @Tia;"EVENT1:SLOPE POS;:EVENT2:SLOPE POS"
320    !
330    !Turn accumulate off, specify the resolution
340     OUTPUT @Tia;"SENSE:HIST:ACC OFF;RANG:RES 50PS"
350    !
360    !2. MAKE THE MEASUREMENTS
370    !
380     OUTPUT @Tia;"init"
390     OUTPUT @Tia;"sens:acq:leng? data"
400     ENTER @Tia;Length
410    !
420    !3. PROCESS THE DATA AND PRINT IT OUT
430    !
440    !Get the timestamps read in, along with resolution and
450    !calibration factors.
460    !
470     REAL Timestamp(100),Interval(100)
480     OUTPUT @Tia;"SENS:TINT:RANG:RES?"
490     ENTER @Tia;Seq_res
500     OUTPUT @Tia;"ACQ:LENG? DATA"
510     ENTER @Tia;Size
```

SCPI/IBASIC Program Listing for Test 6 —Latency after TI 1>2 (Continued)

```
520     OUTPUT @Tia;"CAL:INP:TIM?"
530     ENTER @Tia;Channel_diff
540     OUTPUT @Tia;"FETCH?"
550     FOR I=1 TO Size
560       ENTER @Tia USING "%,K";Timestamp(I)
570     NEXT I
580     N=0
590     Interval_sum=0
600     FOR I=1 TO Size-1
610       IF (I MOD 2=0) THEN
620         IF (Timestamp(I+1)>Timestamp(I)) THEN
630           Ovflow_correct=0
640         ELSE
650           Ovflow_correct=65536
660         END IF
670         Interval(I)=(Timestamp(I+1)-
            Timestamp(I)+Ovflow_correct)*Seq_res+Channel_diff
680         N=N+1
690         Interval_sum=Interval(I)+Interval_sum
700       END IF
710     NEXT I
720     !
730     !Print the answer and wait so it can be read
740     !
750     PRINT "Average Latency =",Interval_sum/N
760     WAIT 1
770     !
780     !Go back to make another measurement
790     !
800     GOTO 380
810     END
```

HP E1740A Performance Test Record (Page 1 of 2)

Hewlett-Packard Model E1740A
Time Interval Analyzer

Serial Number:_____ **Repair/Work Order No.**_____
Test Performed By:_____ **Temperature:**_____
Date:_____ **Relative Humidity:**_____
Notes:_____

Test Number	Operational Verification	Test Results Pass	Fail
1	Power-On Self-Tests	_____	_____

Complete Performance Tests

Test Number	Description	Actual Reading	Limits
1	Hardware resolution set by design of instrument and equals 12.5 ns/256 bins or approximately 48.8 ps (50 ps).	N/A	
2	RMS Timing Jitter checks.	_____	Histogram Std Dev: < 100 ps
3	Minimum measurable time interval checks between two signals down to zero time.	_____	Histogram Min: ≤ 0

HP E1740A Performance Test Record (Page 2 of 2)

Test Number	Description	Actual Reading	Limits
4	Minimum measurableTime Interval checks for a single channel measurement. **A.** Ability to measure every edge of a 80 MHz signal for Input 1 and Input 2. **B.** Ability to measure adjacent edges of a 125 MHz signal followed by a delay (Input 1 only).	_____ _____ _____	Histogram Min: 12.5 ns or less for Input 1 and Input 2 Histogram Min: 8.0 ns or less
5	Maximum measurable time interval checks. Set by design of instrument.	N/A	
6	Latency checks after an Input 1 to 2 time interval measurement.	_____	Versus-Time Delta: 30 ns or less.

Calibration

Calibration is needed whenever performance tests results are not satisfactory. In addition, calibration should be performed at least once a year. All calibrations are performed electronically, via software control of various circuits. There are no screwdriver adjustments.

These are the calibration tasks:

- Input [1 | 2] offset and gain
- Internal Timebase
- Timing, Input 1 to 2
- Timing, Input 1 (<12.5 ns)
- Interpolator DAC and Correction RAM
- Save Calibration values to non-volatile storage
- Get Firmware Revision

Calibration Instructions

In all of the following calibrations procedures, two methods of calibration are listed. One method assumes that the HP E1741A software is available and calibration is performed via menu selections. The other method assumes that the software is not available and calibration is performed via SCPI commands.

When using the HP E1741A software to complete calibration, do the following to ensure accurate results:

- Select the **Preset** button on the tool bar to put the HP E1740A in a known state before following any of the calibration procedures.
- In the Input window (**Control>TIA Setup>Input**) set the Input impedance to 50 ohms.
- If any calibrations are executed using the menu selection: **Control>Service>SCPI Control of E1740A**, set the **Time Out** option to **Long**. Otherwise, a software driven time out will occur, resulting in incomplete calibration and an error.

- Insure all equipment used in the calibration process has at least 30 minutes to warm-up.
- For all calibrations that call for input signals, a signal MUST be supplied. Otherwise, an error will occur.

Input Calibration

Input calibration consists of offset and gain adjustments for each channel. These adjustments are all performed electronically.

To Calibrate the Input Offset (Input 1 or 2)

Equipment Setup

- **Sources: Disconnect all sources**

Instrument Setup

HP E1740 Setup:

- **Selecting 1 or 2, send the following SCPI command:**

  CAL:INP[1|2]:OFFSet:EXEC?

E1741A Setup:

1 **From the menu bar, select Control>Service>Calibrate.**

  The Calibration dialog box appears.

2 **In the Operation box, select "Input 1 Offset" or "Input 2 Offset" and press Execute.**

Verifying the Results

HP E1740A Results:

- **The SCPI command previously shown will return a 1 if successful, and 0 if it fails.**

HP E1741A Results:

- **The Result box on the Calibration dialog box will respond Success.**

In Case of Failure

1 **Re-check connections and sources as applicable.**

2 **Refer to the Troubleshooting and Diagnostics section in this chapter.**

To Calibrate the Input Gain (Input 1 or 2)

Equipment Setup

- **Sources: Apply 5.000 Vdc to Input 1 or 2.**

Instrument Setup

HP E1740 Setup:

- **Selecting 1 or 2, send the following SCPI command:**

  CAL:INP[1|2]:GAIN:EXEC?

HP E1741A Setup:

1 **From the menu bar, select Control>Service>Calibrate.**

  The Calibration dialog box appears.

2 **In the Operation box, select "Input 1 Gain" or "Input 2 Gain" and press Execute.**

Verifying Results

HP E1740A Results:

- **The SCPI command previously shown will return a 1 if successful, and 0 if it fails.**

HP E1741A Results:

- **The Result box on the Calibration dialog box will respond Success.**

Action on Failure

1 **Re-check connections and sources as applicable.**

2 **Refer to the Troubleshooting and Diagnostics section in this chapter.**

Timebase Calibration

Timebase calibration provides for electronic adjustment of the internal reference oscillator to precisely 10 MHz. Preferred sources are a cesium standard, or a 10811 equipped product with a 10 MHz reference out. For timebase calibration, the HP E1740A takes a large sample of the input, computes the mean value, then adjusts the DAC in an effort to minimize the difference between 100 ns and the mean, and measures again. This continues until the best DAC setting is found or an error condition occurs.

To Calibrate the Timebase

Equipment Setup

- **Connect the external 10 MHz reference source to Input 1.**

Instrument Setup

E1740 Setup:

- **Send the following SCPI command:**

  CAL:TIMebase:EXEC

HP E1741A Setup:

1 **From the menu bar, select Control>Service>Calibrate.**

  The Calibration dialog box appears.

2 **In the Operation box, select Internal Timebase and press Execute.**

Verifying Results

HP E1740 Results:

- **The SCPI command previously shown will return a 1 if successful, and 0 if it fails.**

HP E1741A Results:

- **The Result box on the Calibration dialog box will respond Success.**

Action on Failure

1 **Check connections and source, as applicable.**

2 **Refer to the Troubleshooting and Diagnostics section in this chapter.**

Timing, Input 1 to 2 Calibration

This calibration is designed to electronically adjust out the differences in the channel 1 and channel 2 signal path lengths. This will allow precise measurements of time interval at the probe tips.

Two separate calibrations are available:

- **Control>Adjust 1 to 2 reference** allows easy calibration of probe path length differences. Think of this as "front panel to probe tip" cal.
- **Control>Service>Calibrate ... Timing, Input 1 to 2** allows calibration of internal path length differences. Think of this as front panel inward. This process is covered below. In reality, this will include the time incurred in the short cables and splitter used to connect the inputs.

The process used in both cases is to take a large number of samples of the Input 1 to Input 2 interval and determine the mean. This value is the Input 1 to Input 2 timing offset.

Equipment Setup

- **Source: 68.90123 MHz, 13 dbm level**
- **Connections: Using short practical equal length BNC cables, connect inputs 1 and 2 to a splitter. Connect the splitter to the source.**

  Use the shortest cables practical. (Any error in the lengths will end up in the calibration.)

Instrument Setup

HP E1740 Setup:

- **Send the following SCPI command:**

  CAL:INPut[1|2]:TIMing:EXEC?

HP E1741A Setup:

1 **From the menu bar, select Control>Service>Calibrate.**

  The Calibration dialog box appears.

2 **In the Operation box, select "Timing, Input 1 to 2" and press Execute.**

Verifying Results

HP E1740A Results:

- **The SCPI command previously shown will return a 1 if successful, and 0 if it fails.**

HP E1741A Results:

- **The Result box on the Calibration dialog box will respond Success.**

Action on Failure

1 **Check connections and source, as applicable.**

2 **Refer to the Troubleshooting and Diagnostics section in this chapter.**

Timing, Input 1 (<12.5 ns) Calibration

This calibration is designed to provide a way to calibrate out channel differences for the special measurement mode (Input 1 <12.5 ns) that allows single channel measurements to obtain consecutive edge timestamps in the range from 8 to 12.5 ns.

The Input 1 only mode can go down to 12.5 ns on every edge, while the Input 1 <12.5 ns mode can go down to 8 ns for two edges.

Internally, the input for Input 1 only mode is measured and then the mean of a large sample is computed. Next, the Input 1 <12.5 ns mode is measured and the mean of this distribution is determined. The difference between these means is the cal value. This value is automatically accounted for when Input 1 <12.5 ns is selected.

Equipment Setup

- **Source: 68.90123 MHz, +13 dbm level**
- **Connections: Apply to Input 1.**

Instrument Setup

HP E1740 Setup:

- **Send the following SCPI command:**

  CAL:INPut[1|2]:NDEL:EXEC

HP E1741A Setup:

1 **From the menu bar, select Control>Service>Calibrate.**

  The Calibration dialog box appears.

2 **In the Operation box, select "Timing, Input 1 (<12.5 ns)" and press Execute.**

Verifying Results

HP E1740A Results:

- **The SCPI command previously shown will return a 1 if successful, and 0 if it fails.**

HP E1741A Results:

- **The Result box on the Calibration dialog box will respond Success.**

In Case of Failure

1 **Check connections and source, as applicable.**

2 **Refer to the Troubleshooting and Diagnostics section in this chapter.**

Interpolator DAC and Correction RAM Calibration

The Interpolators enable the HP E1740A to perform fine resolutions of time intervals. The internal 79.9 MHz oscillator is used as a source. The calibration process performs a two step process; the five DACs are adjusted to obtain a relatively uniform distribution across all interpolator codes, and then this distribution is used to determine a correction curve to linearize the interpolator results. This information is loaded into the correction RAM that follows the interpolator.

To Calibrate the Interpolator DAC and Correction RAM

Equipment Setup

- **Sources: none required.**

Instrument Setup

HP E1740 Setup:

- **Send the following SCPI command:**

  CAL:INTerpolator:EXEC?

HP E1741A Setup:

1 **From the menu bar, select Control>Service>Calibrate.**

  The Calibration dialog box appears.

2 **In the Operation box, select Cal interpolators and press Execute.**

Verifying Results

HP E1740 Results:

- **The SCPI command previously shown will return a 1 if successful, and 0 if it fails.**

HP E1741A Results:

- **The Result box on the Calibration dialog box will respond Success.**

In Case of Failure

1 **Check connections and source, as applicable.**

2 **Refer to the Troubleshooting and Diagnostics section in this chapter.**

Save Calibration Values to Non-Volatile Storage

This selection allows the user to save the calibration factors in non-volatile RAM within the HP E1740A.

Equipment Setup

- **None required**

Instrument Setup

HP E1740 Setup:

- **Send the following SCPI command:**

  CAL:SAVE

HP E1741A Setup:

1 **From the menu bar, select Control>Service>Calibrate.**

  The Calibration dialog box appears.

2 **In the Operation box, select "Save Calibration values to non-volatile storage" and press Execute.**

Verifying Results

HP E1740A Results:

- **No evidence will be seen of any action, other than a flash of the "access" light on the E1740A.**

HP E1741A Results:

- **No evidence will be seen of any action, other than a flash of the "access" light on the E1740A.**

Get Firmware Revision

This selection allows the user to obtain the current firmware revision of the HP E1740A. No sources or connections are required.

Equipment Setup

- **None required**

Instrument Setup

HP E1740 Setup:

- **Send the following SCPI command:**

  ***IDN?**

HP E1741A Setup:

1 **From the menu bar, select Control>Service>Calibrate.**

  The Calibration dialog box appears.

2 **In the Operation box, select "Get firmware revision" and press Execute.**

Verifying Results

HP E1740A Results:

- **The SCPI command previously shown will return HEWLETT-PACKARD, E1740, 0, xxxx**

The last four digits, xxxx, are coded as follows: The first two numbers represent the year of manufacture. Add 60 to the first two figures to get the actual year. The last two digits represent the actual week of manufacture.

HP E1741A Results:

- **The Result box on the Calibration dialog box will respond HEWLETT-PACKARD, E1740, 0, xxxx**

  The last four digits, xxxx, are coded as follows: The first two numbers represent the year of manufacture. Add 60 to the first two figures to get the actual year. The last two digits represent the actual week of manufacture.

Troubleshooting and Diagnostics

This section provides service information for the HP E1740A, and is divided into the following main sections:

- **Returning the Instrument to Hewlett-Packard for Service.** This section provides you with step-by-step instructions on how to return the instrument for service.
- **Pre-troubleshooting Information.** This section provides you with pertinent information such as safety considerations, recommended test equipment, repair and after service considerations, service accessories, and assembly identification and location.
- **Troubleshooting the HP E1740A.** This section provides you with troubleshooting flowcharts, procedures, and diagnostic information that allows isolation of the faulty assembly. (Once you find a faulty assembly, use the "Dis-assembly and Re-assembly" section in this chapter, which provides instructions on removing the defective assembly.)

If the instrument is under warranty, return the instrument to Hewlett-Packard for service. Refer to the following section "Returning the Instrument To Hewlett-Packard for Service." If you decide to troubleshoot the instrument yourself, refer to the Troubleshooting Procedures in this section.

Returning the Instrument to Hewlett-Packard for Service

To Provide Repair Information

If you are shipping the instrument to an HP office for service or repair, call your nearest Hewlett-Packard Sales Office to make the arrangements. Then tag and package the HP E1740A for shipment.

1 Write the following information on a tag:

- Owners name and address.
- Instrument model number.
- Complete serial number.

2 Attach the tag to the instrument

3 Pack the instrument

If you have the original packaging materials, use those materials to pack the instrument for return. If the original packaging materials are not available or usable, you can order new materials through an HP Sales and Service Office. The new materials are identical to those used by the factory when packaging the instrument.

Pre-Troubleshooting Information

This section contains the following pertinent troubleshooting information:

- Safety Considerations
- Recommended Test Equipment
- Repair Considerations
- Dis-assembly and re-assembly specifics

Safety Considerations

Although this instrument has been designed in accordance with international safety standards, this guide contains information, cautions, and warnings which must be followed to ensure safe operation and to retain the instrument in a safe condition. Service instructions, and adjustment procedures requiring removal of the instrument cover, are for use by service-trained personnel only. To avoid dangerous electric shock, do not perform any servicing or make any adjustments with the cover removed, unless qualified to do so.

WARNING

BEFORE APPLYING AC POWER, THE INSTRUMENT AND ALL PROTECTIVE EARTH TERMINALS, EXTENSION CORDS, AUTO TRANSFORMERS, AND DEVICES CONNECTED TO THE INSTRUMENT SHOULD BE CONNECTED TO A PROTECTIVE EARTH GROUNDED SOCKET.

ANY INTERRUPTION OF THE PROTECTIVE GROUNDING CONDUCTOR INSIDE OR OUTSIDE THE INSTRUMENT OR DISCONNECTION OF THE PROTECTIVE EARTH TERMINAL WILL CAUSE A POTENTIAL SHOCK HAZARD THAT COULD RESULT IN PERSONAL INJURY. INTENTIONAL INTERRUPTION IS PROHIBITED.

Any adjustment, maintenance, and repair of the opened instrument under voltage should be avoided as much as possible and, if necessary, should be carried out only by a skilled person who is aware of the hazards involved (for example, fire and electric shock).

Recommended Test Equipment

Test equipment recommended for testing and troubleshooting the HP E1740A is listed in Table 16.1 at the beginning of this chapter. Substitute equipment may be used if it meets or exceeds the required characteristics listed in the table.

Repair Considerations

Electrostatic Discharge

Electronic components and assemblies in the HP E1740A can be permanently degraded or damaged by electrostatic discharge. Use the following precautions when servicing the instrument:

1 **ENSURE that static sensitive devices or assemblies are serviced at static safe work stations providing proper grounding for service personnel.**

2 **ENSURE that static sensitive devices or assemblies are stored in static shielding bags or containers.**

3 **DO NOT wear clothing subject to static charge buildup, such as wool or synthetic materials.**

4 **DO NOT handle components or assemblies in carpeted areas.**

5 **DO NOT remove an assembly or component from its static shielding protection until you are ready to install it.**

6 **AVOID touching component leads. (Handle by packaging only.)**

Disassembly and Reassembly Specifics

Refer to the "Disassembly and Reassembly" section of this chapter for complete dis-assembly and re-assemble details, and for exploded views of the instrument parts.

Troubleshooting and Diagnostic Procedures

This section contains troubleshooting and diagnostic information, in the form of a trouble isolation flowchart and descriptions. The diagnostics applied are described as they are used, and are a subset of the entire suite of diagnostics. The intent of this process is to isolate the problem board. Component level troubleshooting is NOT supported in this guide.

To actually do troubleshooting on the HP E1740A, disassembly is required for access. Refer to the "Disassembly and Reassembly" section in this chapter for that information. To allow troubleshooting and probing while the HP E1740A circuit assemblies are outside their sheet metal housing, a open-sided card cage is required, and is listed in the equipment section on Table 16-1 at the beginning of this chapter. Once the cards are accessible outside the sheet metal, re-install the two ribbon cables W1 and W2, and the coax cable W3 and together plug the two cards A1 and A3 into the card cage. A2 is only supported on A3 by the connectors at its edge. Use care while the cards are so installed, as the only support is the connectors at the rear and the lower set of card guides.

Following the procedure is paramount to isolating the faulty assembly. If the diagnostics are used without reference to this process, misleading results may be obtained, leading to improper repair action. Besides the diagnostics, other means are used to localize and isolate the problem board, such as visual checks of LEDs, scope and multimeter checks, etc.

The diagnostics may be executed via SCPI commands, or from the HP E1741A software. All the information is given in the description of the flowchart.

Troubleshooting Instructions

- If a diagnostic is run from the menu selection: **Control>Service>SCPI Control of E1740A**, set the **Time Out** option to **Long.** Otherwise, errors will occur.

- For all diagnostics that call for input signals, a signal MUST be supplied. Otherwise, an error will occur.

List of Diagnostics

This list is provided for your convenience only. Many of these diagnostics tests are conducted in conjunction with the troubleshooting procedures. More information may be found in the "Definitions of the TIA Subsystem Commands" chapter in this guide. However, not all of these diagnostics are used in the following troubleshooting procedures.

To Activate Diagnostics:

1 **If using the HP E1741A software, from the menu bar select Control>Service>Diagnostics and select the one called for in the procedure.**

2 **If using SCPI directly, type in the DIAG:TEST? ##, where the ## is the number of the diagnostic called for as follows:**

Table 16-2. SCPI Diagnostic Tests

DIAG:TEST?	Diagnostic Name
0	Test All (runs all the diagnostics that do not require user action: 1 thru 10, 12, and 13)
1	SRAM
2	EPROM
3	EEPROM
4	NV RAM
5	XILINIX
6	Histogram
7	Video RAM (VRAM)
8	Correction RAM
9	79.9 MHz
10	Event Pacing counter
11	No Test Assigned
12	Randomizer
13	Input Board
14	Trigger level and event count (Input 1 and 2)
15	Inhibit
16	Trigger input
17	Timebase

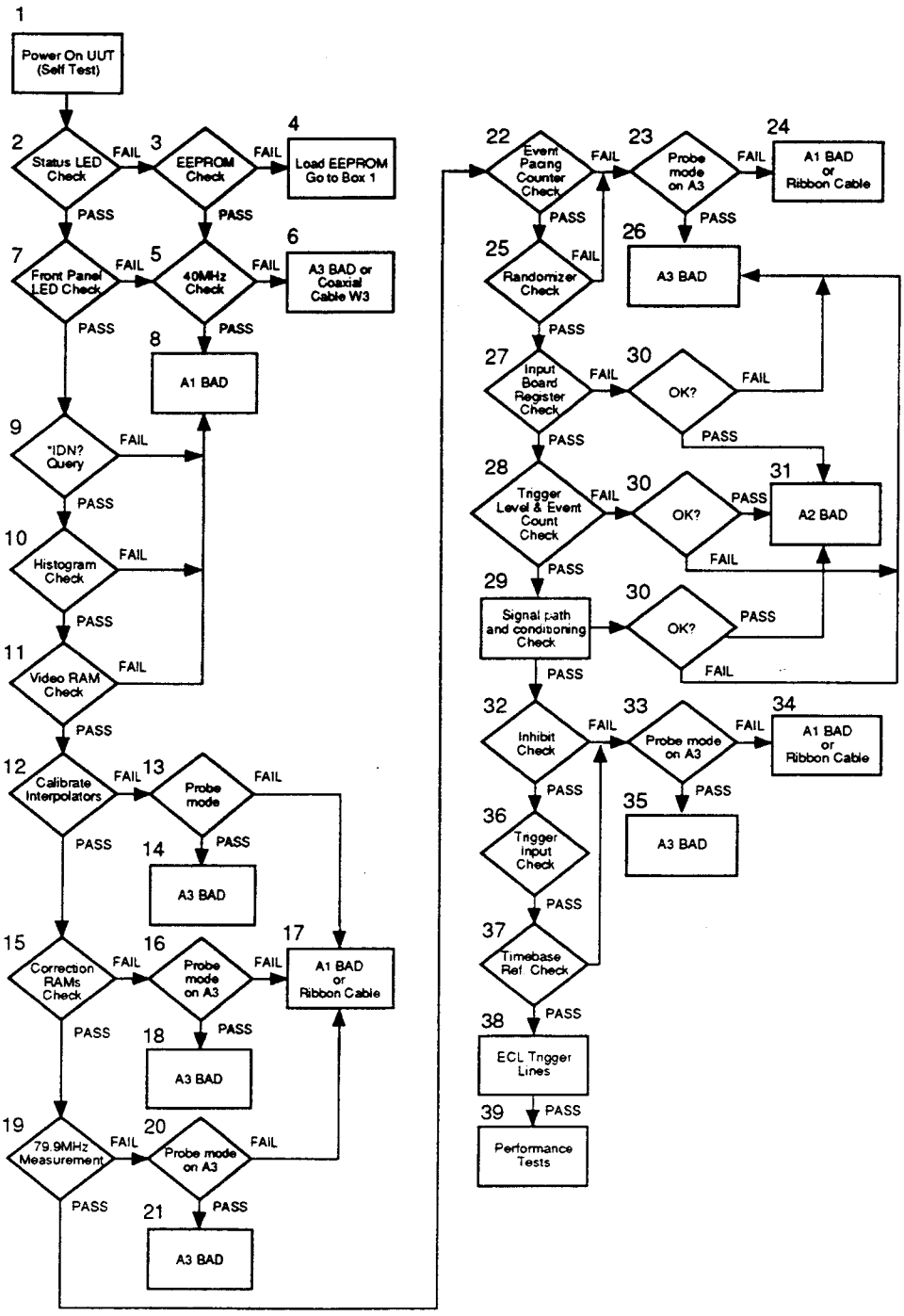


Figure 16-1. E1740A Troubleshooting Flowchart

Troubleshooting Flowchart Description

The following is a description of the numbered boxes in the HP E1740A Troubleshooting Flowchart as seen in Figure 16-1. The remaining figures called out in the flowchart description are located at the end of this section.

1 Power on UUT. [runs diag 1, 2, 3, 4, 5]

These tests will be performed on power up: SRAM, EPROM, EEPROM, NVRAM, XILINX. The internal timebase will be selected. The EPROM and EEPROM checksums will be verified. A visual check of the programmed-done LEDs will be performed.

Action On Failure:
If the CPU does not power up, then no diagnostics can be exercised. Continue to box 2.

2 Status LED Check

The status LEDs will indicate how far the power up self test ran. See Figure 16-2 at the end of this section for the location of the LEDs.

They can be viewed at the top edge of A1 and will all be on when power is applied. As the powerup routine proceeds, they will go out, from right to left as viewed with the VXI connectors at the right.

0	0	0	0	0	0	0
XILINX[1]	XILINX	Lang B	EEPROM	EEPROM	EPROM	EPROM

[1]Xilinx LEDs are approximatetly 2" below the edge of the board.

Pass:
Go to box 7.

Action On Failure:
If any of the status LEDs remain on, the A1 board may have failed.
Go to box 3.

3 EEPROM Check

HP E1741A: 3. EEPROM
SCPI: DIAG:TEST? 3

A checksum is computed for each of the four system EEPROMs. These are compared with the expected checksums that are stored in each EEPROM. If the computed checksum does not match the expected checksum or if the checksum is equal to zero, the test fails.

Setup Procedure:
- No user intervention required.

Expected Results:
- "EEPROM Test PASSED"

Pass:
Go to box 5.

Fail Messages:
"Odd EEPROM 1 checksum FAILED (U93): checksum=2674EB (267335)"
"Even EEPROM 1 checksum FAILED (U21): checksum=2674EB (267335)"
"Odd EEPROM 2 checksum FAILED (U74): checksum=2674EB (267335)"
"Even EEPROM 2 checksum FAILED (U28): checksum=2674EB (267335)"

The returned failure message shows which EEPROM part failed its checksum test, the checksum that was read, and the expected checksum in parenthesis.

Action On Failure:
Go to box 4.

4 Load EEPROM

Re-program EEPROMs and then run DIAG:TEST? 3 again.

Go to box 1 and repeat test. If the test still fails, replace the A1 board.

5 40 MHz Check

This checks the 40 MHz test point to see if the clock is arriving on the A1 board. If the signal is not present, the front panel Error LED will be on too.

Procedure:
Probe the 40 MHz test point on the A1 board.

Signal level expected: TTL square wave.

Pass
Go to box 8.

Action On Failure:
Go to with box 6.

6 A3 BAD or coax cable W3

Procedure
Check cable with ohmmeter. If the cable is OK, the A3 board has failed. Replace A3.

7 Front Panel LED Check

This is a visual test. All 4 LEDs should be off. If not, there is probably a failure of some kind on the A1 board. There is no significance to the actual LED power-on sequence, but it is repeatable.

Pass
Go to box 5.

Action On Failure:
Go to box 9.

8 A1 BAD

Replace the A1 board.

9 Obtain the identification of the HP E1740A

HP E1741A: Open the Control/Service/SCPI control of the HP E1740A and type in the Send block *IDN? and send it.

- SCPI COMMAND: *IDN?

Perform this query to check the VXI interface. Bus grant, select code, ROM revision, and logical address are checked, and response should occur as shown. The Access LED should blink when the command is sent.

Expected Result:s
Response in either case should be:
HEWLETT PACKARD,E1740A,0,3319.

The last four digits will vary depending on revision level of firmware installed.

Pass:
Go to box 16.

Action On Failure:
Go to box 8: replace the A1 board.

10 Histogram Check

HP E1741A: 6. HISTOGRAM
SCPI: DIAG:TEST? 6

Performs a test of the histogram hardware. For each of the four histogram circuits, three tests are performed. A histogram clear operation is performed. This test fails if any histogram RAM locations are not set to zero after the clear operation. Then a register read/write test is performed. Then test patterns are written to and read from each location of the histogram RAM.

Setup Procedure:
No user intervention required.

Expected Results:
"HIST1: PASSED
HIST2: PASSED
HIST3: PASSED
HIST4: PASSED"

Fail Messages:
"HIST1:
FAILED Clear: addr=0 data = fffff
FAILED Register Test...
FAILED RAM Test failures...

A similar message is reported for each failed histogram circuit.

Any data not shown above is unusable at the board replacement level.

Pass:
Go to box 11.

Action On Failure:
Go to box 8. Replace the A1 board.

11 Video RAM Check

E1741A: 7. VRAM
SCPI: DIAG:TEST? 7

A data destructive RAM test is performed on the eight Video RAMs that are used to store the measurement data. An address line test is performed by writing to and reading from selected locations in the Video RAM space.

Setup Procedure:
No user intervention required.

Expected Results:
"VRAM Test PASSED"

Pass:
Go to box 12.

Fail Messages:
"VRAM Test FAILED:

A variety of other information is shown, but is not usable at the board replacement level.

Action On Failure:
Go to box 8: Replace the A1 board.

12 Calibrate interpolators

HP E1741A: From the menu bar, select **Control>Service>Calibrate>Interpolator DAC and Correction RAM**

SCPI: CAL:INT:EXEC?

This calibration exercises the interpolators. A failure of this test indicates a problem on the A3 board.

Setup Procedure:
No user intervention is required.

Expected Results:
HP E1741A: success
SCPI: 1

Pass:
Go to box 15.

Fail Messages:
HP E1741A: failed
SCPI: 0

Action On Failure:
Go to box 13.

13 Probe mode

HP E1741A: From the menu bar, select
Control>Service>SCPI Control of E1740 and then enter: DIAG:PROBE

SCPI: DIAG:PROBE

Checks for communication of control signals between the A1 and A3 boards. All mode bits, as well as address and data lines are toggled to allow probing of various points to check signal integrity.

NOTE

This dianostic is available via SCPI only. When this diagnostic is activated, the E1740A must be power cycled to restore normal operation. Consequently, it is not listed in the E1741A Diagnostic menu.

Setup Procedure:
No user intervention is required, other than dis-assembly required for access to the probing points.

Action: On the A3 board, probe the following ICs (Figure 16-3 and 16-4).

U2: pins 19 and 26
U12: pins 1, 3, 21
U20: pins 1, 2, 3, 4, 5
U26: pins 19, 20, 21, 22, 24, 25, 26, 27, 28
U82: pins 1, 11
U103: pins 22, 27

Expected Results:
An oscilloscope is used to check the signals. TTL levels can be expected.

Pass:
If all readings are at TTL levels, go to box 14.

Fail Messages:
No fail messages occur. The readings on the oscilloscope give the indications of results.

Action On Failure:
Go to box 17.

14 A3 BAD

Replace the A3 board.

15 Correction RAM check

HP E1741A: 8. Correction RAM
SCPI: DIAG:TEST? 8

The correction RAM is part of the interpolator circuits. This test first saves away the current contents of the correction RAM. It then fills it with various values, and then reads them back. The contents of the RAM are then restored.

Setup Procedure:
No user intervention required.

Expected Results:
"Correction RAM PASSED"

Pass:
If the expected results are satisfied, go to box 19.

Fail Messages:
"Correction RAM FAILED: addr=00 data=FF (AA) failures=1"

Action On Failure:
Go to box 16.

16 Probe mode

See number 13 for information.

17 A1 bad or ribbon cable

Check cables with ohmmeter. Replace cables if bad.

If cables are OK, replace the A1 board.

18 A3 BAD

Replace the A3 board.

19 79.9 MHz Measurement

HP E1741A: 9. 79.9 MHz
SCPI: DIAG:TEST? 9

This test performs a continuous histogram measurement on the internal 79.9 MHz oscillator. The measurement takes one block of 1,000,000 samples. A frequency of 79.9 MHz is high enough so that noise might cause time intervals to increase to about 80 MHz. When this happens, measurements are missed and will show up as time intervals that are twice and three times the expected value. These are expected and are ignored for the mean calculation.

Setup Procedure:
No user intervention required.

Expected Results:
"79.9 MHz Test PASSED:
Sdev = 32.304 ps Mean = 12.565 ns Min = 12.402 ns Max = 12.744 ns

Pass
If expected results are statisfied, go to box 21.

Fail Messages:
"79.9 MHz Test FAILED:
Sdev = 32.304 ps Mean = 12.900 ns Min = 12.402 ns Max = 13.044 ns

The statistics shown above may vary slightly. The key observation is the PASS or FAILED message.

The test fails if the measurement mean of the main lobe is more than 1% off of the expected time interval or if more than 10% of the measurements are not in this main lobe. The return message reports the standard deviation, mean, minimum, and maximum of the main lobe.

Action On Failure:
Go to box 22.

20 Probe mode

See number 13 for information.

Pass
Go to box 21.

Action On Failure:
Go to box 17.

21 A3 bad

Replace the A3 board.

22 Event pacing counter check

HP E1741A: 10. Event Pacing Counter
SCPI: DIAG:TEST? 10

This test performs a continuous histogram measurement on every fifth cycle of the internal 79.9 MHz oscillator. The measurement takes one block of 1,000,000 event paced samples.

Setup Procedure:
No user intervention required.

Expected Results:
"Event Pacing Counter Test PASSED:
Sdev = 44.264 ps Mean = 62.627 ns Min = 62.451 ns
Max = 62.793 ns"

Pass:
Go to box 25.

Fail messages:
"Event Pacing Counter Test FAILED:
Sdev = 93.652 ps Mean = 65.534 ns Min = 62.352 ns
Max= 67.423 ns"

The statistics above may vary slightly. The key observation is the PASS or FAIL message.

The test fails if the measurement mean of the main lobe is more than 1% off of the expected time interval or if more than 10% of the measurements are not in this main lobe. Both messages return the standard deviation, mean, minimum, and maximum of the main lobe. It also shows how many measurements fell in the first, second, third, and third lobes.

Action On Failure:
Go to box 23.

23 Probe mode

See number 13 for information.

Pass:
Go to box 24.

Action On Failure:
Go to box 26

24 A1 BAD or Ribbon Cable

Check ribbon cables with ohmmeter. If cables are OK, replace the A1 board.

25 Randomizer Check

HP E1741A: 12. Randomizer
SCPI: DIAG:TEST? 12

This test performs a continuous histogram measurement on every random cycles of the internal 79.9 MHz oscillator. The measurement takes one block of 1,000,000 randomly paced samples. The measurements will clump together into 6 distributions: 6, 7, 8, 9, 10 and 17 times the time interval of the input frequency.

Setup Procedure:
No user intervention required.

Expected Results:
"Randomizer Test PASSED"

Pass
Go to box 27.

Fail Messages:
"Randomizer Test FAILED...

The test fails if the measurements do not distribute in the expected manner. The message returned will contain line for each distribution expected, as well as other information.

Action On Failure:
Go to box 23.

26 A3 BAD

Replace the A3 board.

27 Input Board Register Check.

HP E1741A: 13. Input Board register
SCPI: DIAG:TEST? 13

Performs a clear test, walking ones test, set-all-bit test, 55 and AA pattern tests on the register of the input amplifier board.

Setup Procedure:
No user intervention required.

Expected Results:
"Input Register Test PASSED"

Pass
Go to box 28.

Fail Messages:
"Input Register Test FAILED: Bits = [80]"

Action On Failure:
Go to box 30.

28 Trigger Level and Event Count Check

HP E1741A: 14.Trigger level and event count check
SCPI: DIAG:TEST? 14

Sets the trigger levels of Input 1 and 2 DACs to values: 1.500 V, -1.500 V, 0.015 V and -0.015 V and checks the trigger light value against the expected result for an input of 0.0 V. Then a time stamp measurement is taken where the sign of the trigger level is cycled 100 times.

Setup Procedure:
Set Channel 1 and 2 inputs to ground.

Expected Results:
Trigger Level Test PASSED

Pass
Go to box 29.

Fail Messages:
Trigger Level Test Failed:

Chan 1: 1.500 V failed
Chan 1: 0.015 V failed
Chan 2: 1.500 V failed
Chan 2: 0.015 V failed
Events = 0 [100]"

Action On Failure:
Go to box 30.

29 Signal path and conditioning circuits

Visual Checks:
Apply 10 Hz signal at 1 V peak-to-peak to each input (trigger, Input 1, Input 2) and check trigger and A2 output LEDs. See Figure 16-6 for locations.

Expected Results:
LEDs flash.

Pass:
Go to following manual tests.

Fail Message:
LED does not flash

Action On Failure:
Go to box 30.

MANUAL TESTS:

50 ohm/1 Mohm:

HP E1741A:

Procedure:

1 From the Menu bar, select **Control>TIA Setup** and TIA Setup window appears.

2 Press **Input** and select the 50 ohm button for Input 1 and Input 2.

3 With ohmmeter, probe between Input 1 jack and ground and read 50 ohms. Repeat for Input 2.

SCPI:

Procedure:

1 Issue command INP[1|2]:IMP 50 to set Input 1 to 50 ohms. (Select Input 1 or 2)

2 With ohmmeter, probe between Input 1 jack and ground and read 50 ohms.

AC/DC:

HP E1741A:

Procedure:

1 Apply a sine wave input of 1 V, offset by 1 V from ground to Input 1.

2 From the Menu bar, select **Control>TIA Setup** and TIA Setup window appears.

Procedure:

1 Probe points as indicated on Figure 16-7. Levels are described there also.

2 If all signals are correct , go to box 31.

Action on Failure:
If signals are not seen, carefully inspect the connectors between A2 and A3 for bent or broken pins. If connectors are OK, go to box 26.

B ECL Output Checks

Procedure:

1 Apply a 10 Hz 1 V peak-to-peak sinewave signal to each input in turn.

The LEDs on the A2 board should flash, indicating an output is going to the LEDs. To make sure the signal goes to the A3 board, continue with the next step.

2 Probe pins on the A2 connector as shown in Figure 16-7, checking for an output signal. Note that outputs are ECL balanced lines.

ECL levels: Logic high (1): -0.9 V
Logic low (0): -1.75 V

Action On Failure:
If one or more ECL outputs are not present, the A2 board has failed. Go to box 31.

If these all check OK, and the trigger lights on the front panel are NOT blinking, this can indicate a problem in the A3 trigger light control circuit, the connector between the A2 and A3 boards, or the trigger light drivers on the A2 board.

C. Trigger Light Control Checks

Procedure:

- To locate the trigger light control failure, drive each input in turn with the 10 Hz test input and probe the trigger lines as shown in Figure 16-5.

3 Press **Input** and in the Input window the select the DC coupling for Input 1

4 In DC coupling, input should not trigger.

In the Input window, switch to AC coupling and the HP E1740A should trigger.

5 Repeat the above steps for Input 2.

SCPI:

- Send following commands to change between DC and AC coupling for Input 1 and Input 2.

  INP[1|2]:COUP AC (Select Input 1 or 2)

SEP/COMM relays.

NOTE: This test can only be done via SCPI. No selections exist in E1741A software for this choice.

Procedure:

1 Issue the SCPI command INP:ROUT:COMM to put Input 1 into COMmode.

2 Apply a signal to Input 2. Input 2 should NOT trigger.

All Tests Passed:
Go to box 31.

Action On Failure (any manual test):
Go to box 30.

30 A2/A3 Control, ECL Output, and Trigger Line Checks

A Probe Mode: See number 13 for information on invoking test.

Checks for communication of control signals between A2 and A3. All address and data lines are toggled to allow probing of various points to check signal integrity.

Expected Results:
Active signals indicate that the A3 board is OK, and that the A2 board has failed.

Inactive signals indicate that the A3 board has failed.

Action On Failure:
If active signals are present, go to box 31.
For inactive signals, go to box 26.

31 A2 bad

Replace the A2 board.

32 Inhibit Check

E1741A: 15 Inhibit
SCPI: DIAG:TEST? 15

This test sets up the hardware to make one continuous time stamped measurement of 10,000 samples. The data is then searched to find inhibits.

Setup Procedure:
Connect a 100 KHz TTL trigger signal to the inhibit input.

Expected Results:
"Inhibit Test PASSED: Measurements = 10000 Inhibits = 234

The actual number of inhibits will vary depending on the inhibit input frequency and timing of inhibit occurances. Any number of inhibits OTHER than 0 is OK.

Pass
Go to box 36.

Fail Messages:
"Inhibit Test FAILED: Measurements = 10000 Inhibits = 0
Test fails if no inhibits are found or less than 10,000 measurements are taken.

Action On Failure:
Go to box 33

33 Probe mode:

See Number 13 for information. If test results are satisfied, go to box 36.

Pass
Go to box 35.

Action On Failure:
Go to box 34.

34 A1 or Ribbon Cable bad

Check cable with ohmmeter. If cable is OK, replace the A1 board.

35 A3 bad

Replace the A3 board.

36 Trigger Input check

HP E1741A: 16 Trigger Input
SCPI: DIAG:TEST? 16

This test sets up the hardware to make one externally armed continuous histogram measurement of 1,000 samples. The internal 79.9 MHz signal is used. The external trigger level is set to TTL levels.

Setup Procedure:
Connect a 100 KHz TTL trigger signal to the Trigger input.

Expected Results:
"Trigger Input Test PASSED"
"Trigger Input Test FAILED"

Pass
Go to box 37.

Failed Messages
If the no trigger occurs, the measurement will timeout and return the failed message.

Action On Failure:
Go to box 33.

37 Timebase Reference Checks

HP E1741A: 17. Timebase
SCPI: DIAG:TEST? 17

This test performs three continuous histogram measurements of 1,000,000 samples. The first with the 10 MHz internal timebase, the second from the external 10 MHz oscillator and the third with the 10 MHz "clock10" signal from the VXI backplane.

Setup Procedure:
Connect 10 MHz signal to the 10 MHz Ref In connector. Use 1 to 5 V peak-to-peak as the signal level.

Expected Results:
"Timebase Test PASSED:

Pass
Go to box 38.

Fail Messages:
"Timebase Test FAILED: Information on INTernal, EXTernal, or CLK10 failures will be shown.

The test fails if any of the measurements do not finish or if the mean of any of the measurements are off by more than 1%. The result of the each 79.9 MHz test is reported.

Action On Failure:
Go to box 33.

NOTE

For troubleshooting, if Ext timebase fails, be sure the 10 MHz signal is present. If the CLK 10 fails, check for signal at EPC-7 (or Slot 0) clock out connector.

38 Check ECL trigger lines between the controller and E1740

This test requires an input signal to be measured. No front panel indications on the E1740A can be used to verify proper operation. The Slot 0 card and the E1740A are programmed via SCPI commands to route a TTL trigger signal into the Slot 0 card Trig in connector, thru the VXI back plane, and into the E1740A external trigger lines ECLTRG0 and ECLTRG1. The Slot 0 card receives the TTL signal and creates an ECL signal, which is what the E1740A requires when operated in this mode. This trigger signal will then cause the input signal to be measured with a histogram display for verification. To verify that the external trigger is really causing the measurement, the TTL signal at the Slot 0 front panel jack will be disconnected. The measurement will then stop.

Procedure for first trigger line

Setup Procedure:

1 Apply an input signal of 12 MHz at +10dbm to Input 1.

2 From the menu bar, select **Control>TIA Setup**.

The TIA setup window appears.

3 In the top/left portion of the window, select the "Triggered" button and close the window.

No other settings require modification.

Control setup for External PC and a Slot 0 card:

This makes use of the E1741A software. If direct SCPI control is being used via BASIC or some other language, some programming will need to be done by the user. (The EPC-7 setup will be available at a future time. Call the factory for details.)

NOTE

The Slot 0 card needs to be the HP E1406A called out in the equipment list. Any other card may need a different command to set up the external trigger capability. Consult the manual for the card in use.

Procedure:

1 Send the SCPI commands to set up the E1406A Slot 0 card for external trigger input.

- To do this, the interface settings must be changed to allow communication with the Slot 0 card.

2 From the menu bar, select **Control>E1740 interface** and press **Set Details.**

3 Write down the secondary address for reference

The figure is probably 6. This is the address of the E1740A.

4 Set the Secondary address to 0.

This allows communication with the Slot 0 card.

5 Send the following commands:

OUTput:ECLt0:SOUR EXT;STATe on

6 To be sure this command was executed properly, send the query OUTput:ECLT0:SOUR?

The response should be EXT.

7 Next, send the query
OUTput:ECLt0:STATe?

The response should be 1, indicating that the trigger line is on. Anything else, or no response, indicates that the E1406A did not hear or properly interpret the command. Most likely, the command was improperly formulated. Check it carefully and resend. Re-check it with a query.

To regain control of the E1740A:

1 From the menu bar, select **Control>E1740 interface** and press **Set Details**.

2 Reset the secondary address to what was written down previously probably 6.

Doing this will now allow communication with the E1740A once more.

3 Select **Control>Initialize E1740A** to complete this operation.

E1740A Setup:

You will send the SCPI command to the E1740A to set up for external trigger on line 0:

1 From the menu bar, select **Control>Service>SCPI control of E1740A**.

2 Send the following:
TRIG:SOUR ECLT0

3 To be sure this command was executed properly, send the query TRIG:SOUR?

The proper response is ECLTRG0. Anything else indicates that the E1740A did not hear or properly interpret the command. Check the red ERROR light on the front of the E1740A. If on, the command was improperly formulated. Check it carefully and resend. Re-check it with the query.

4 Select **Display>Histogram**.

5 Apply a 100 KHz trigger signal to "Trig in" jack on Slot 0 card.

6 On the tool bar press the **Run** button.

Expected Results:
A histogram display should be visible.

Pass
A histogram is visible. The following additional checks for a pass condition should also be completed.

As a test of the validity of all the above, assuming that a measurement is being made, disconnect the trigger input from the Slot 0 card. The measurement updating should stop. Reconnect it, and the measurement updating should resume. If it does NOT stop, check two items:

1 From the TIA Setup window (Control>TIA Setup) verify that **Triggered** is selected in the upper-left portion.

2 In the SCPI control screen, query the trigger source with TRIG:SOUR?

The correct response is ECLTRG0. If this is not seen, review the steps of this procedure.

Failed Results:
There is no Histogram display.

If no measurement is visible, check the message in the lower left corner of the display. If the message reports, "Waiting for measurement complete", the external trigger signal is not working.

Action on Failure:
Possible A3 failure. Before replacing any assemblies, review the following:

1 Re-check all the above work

The SCPI programming is unforgiving of any syntax errors. The query commands are the best way to be sure of proper programming.

2 Probe the VXI back plane, checking for the ECL trigger levels.

See the manual for the VXI cardcage manual for further information (This is probably the E1400T openside cage).

If after all this checking, the external trigger will not work, then A3 has failed. Replace the A3 card.

Repeat for the second trigger line:

To test the second trigger line, the previous procedure is basically repeated. However, there are differences in the SCPI commands being sent. Follow these detailed steps:

Procedure:

1 Send the SCPI commands to set up the E1406A Slot 0 card for external trigger input.

- To do this, the interface settings must be changed to allow communication with the Slot 0 card.

2 From the menu bar, select **Control>E1740 interface** and press **Set Details**.

3 Write down the secondary address for reference

The figure is probably 6. This is the address of the E1740A.

4 Set the Secondary address to 0.

This allows communication with the Slot 0 card.

5 Send the following commands:

OUTput:ECLT1:SOUR EXT:STATe on

6 To be sure this command was executed properly, send the query OUTput:ECLT1:SOUR?

The response should be EXT.

7 Next, send the query
OUTput:ECLt1:STATe?

The response should be 1, indicating that the trigger line is on. Anything else, or no response, indicates that the E1406A did not hear or properly interpret the command. Most likely, the command was improperly formulated. Check it carefully and resend. Re-check it with a query.

To regain control of the E1740A:

1 From the menu bar, select **Control>E1740 interface** and press **Set Details**.

2 Reset the secondary address to what was written down previously, probably 6.

Doing this will now allow communication with the E1740A once more.

3 Select **Control>Initialize E1740A** to complete this operation.

E1740A Setup:

You will send the SCPI command to the E1740A to set up for external trigger on line 0:

1 From the menu bar, select **Control>Service>SCPI control of E1740A.**

2 Send the following:
TRIG:SOUR ECLT1

3 To be sure this command was executed properly, send the query TRIG:SOUR?

The proper response is ECLTRG1. Anything else indicates that the E1740A did not hear or properly interpret the command. Check the red ERROR light on the front of the E1740A. If on, the command was improperly formulated. Check it carefully and resend. Re-check it with the query.

4 Select **Display>Histogram**.

5 Apply a 100 KHz trigger signal to "Trig in" jack on Slot 0 card.

6 On the tool bar press the **Run** button.

Expected Results:
A histogram display should be visible.

Pass
A histogram is visible. The following additional checks for a pass condition should also be completed.

As a test of the validity of all the above, assuming that a measurement is being made, disconnect the trigger input from the Slot 0 card. The measurement updating should stop. Reconnect it, and the measurement updating should resume. If it does NOT stop, check two items:

1 From the TIA Setup window (Control>TIA Setup) verify that Triggered is selected in the upper-left portion.

2 In the SCPI control screen, query the trigger source with TRIG:SOUR?

The correct response is ECLTRG0. If this is not seen, review the steps of this procedure.

Failed Results
There is no Histogram display.

If no measurement is visible, check the message in the lower left corner of the display. If the message reports, "Waiting for measurement complete", the external trigger signal is not working.

Action on Failure
Possible A3 failure. Before replacing any assemblies, review the following:

1 Re-check all the above work

The SCPI programming is unforgiving of any syntax errors. The query commands are the best way to be sure of proper programming.

2 Probe the VXI back plane, checking for the ECL trigger levels.

See the manual for the VXI cardcage for more information (This is probably the E1400T openside cage).

If after all this checking, the external trigger will not work, then A3 has failed. Replace the A3 card.

NOTE

Probably the easiest way to restore normal Slot 0 card operation after all this programming is to power cycle the card cage. This will cause the Slot 0 card to re-initialize itself. If the Slot 0 card is not configured as needed after a power cycle, consult the Slot 0 programming information and reset it as needed.

39 Performance Tests

See Performance test section to execute tests.

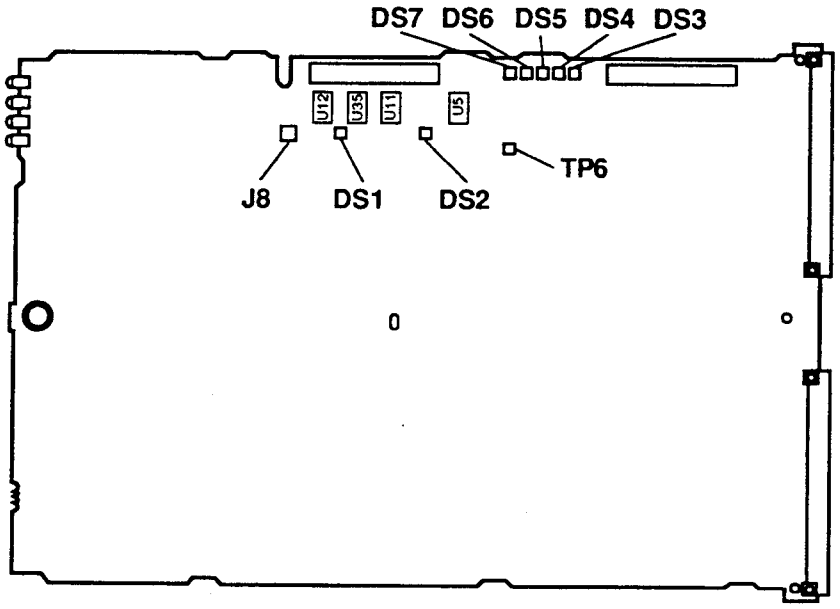


Figure 16-2. Location of LEDs on the A1 Board (DS1-DS7)

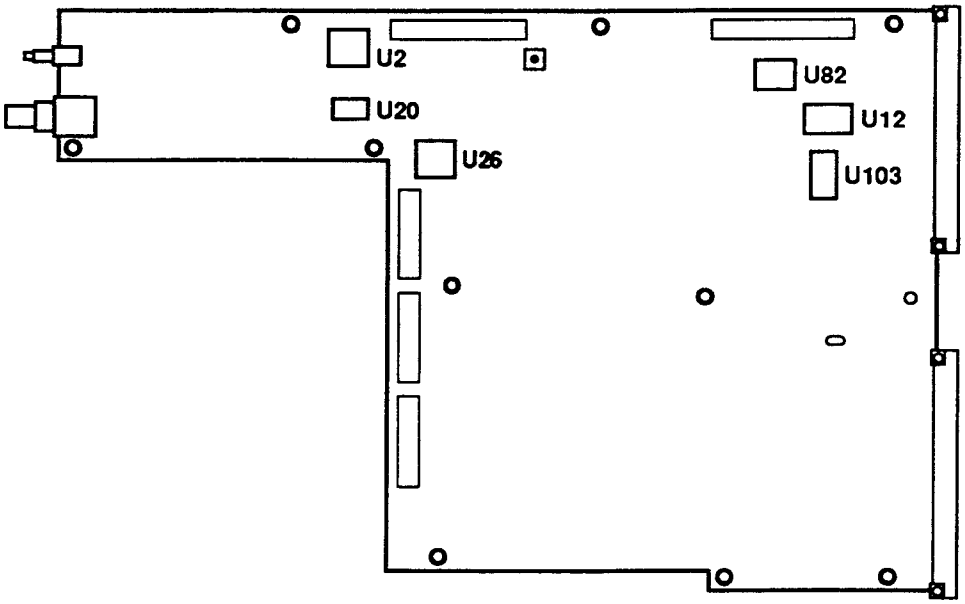


Figure 16-3. Component Locator for the A3 Board

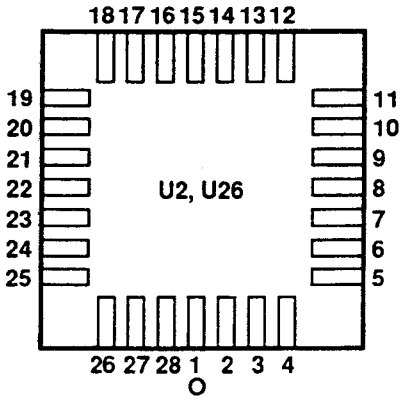


Figure 16-4A. Pin Layout for U2 and U26

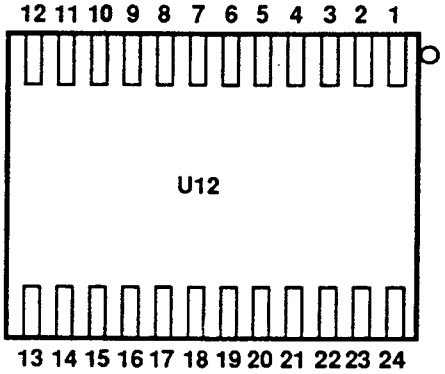


Figure 16-4B. Pin Layout for U12

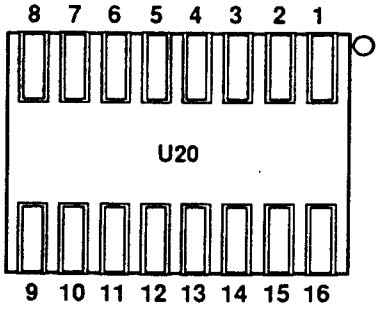


Figure 16-4C. Pin Layout for U20

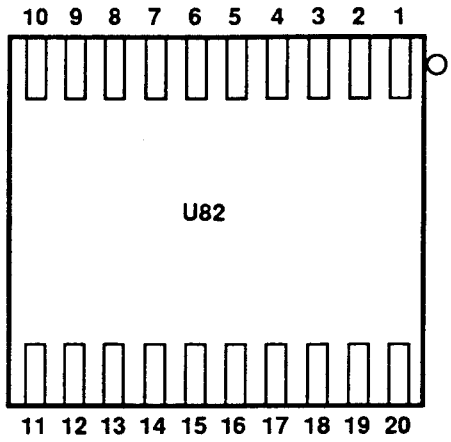


Figure 16-4D. Pin Layout for U82

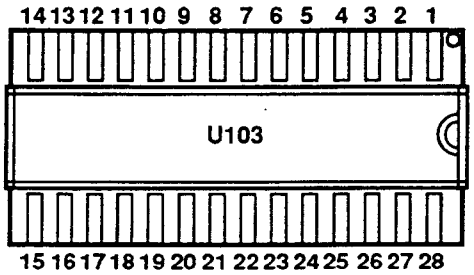


Figure 16-4E. Pin Layout for U103

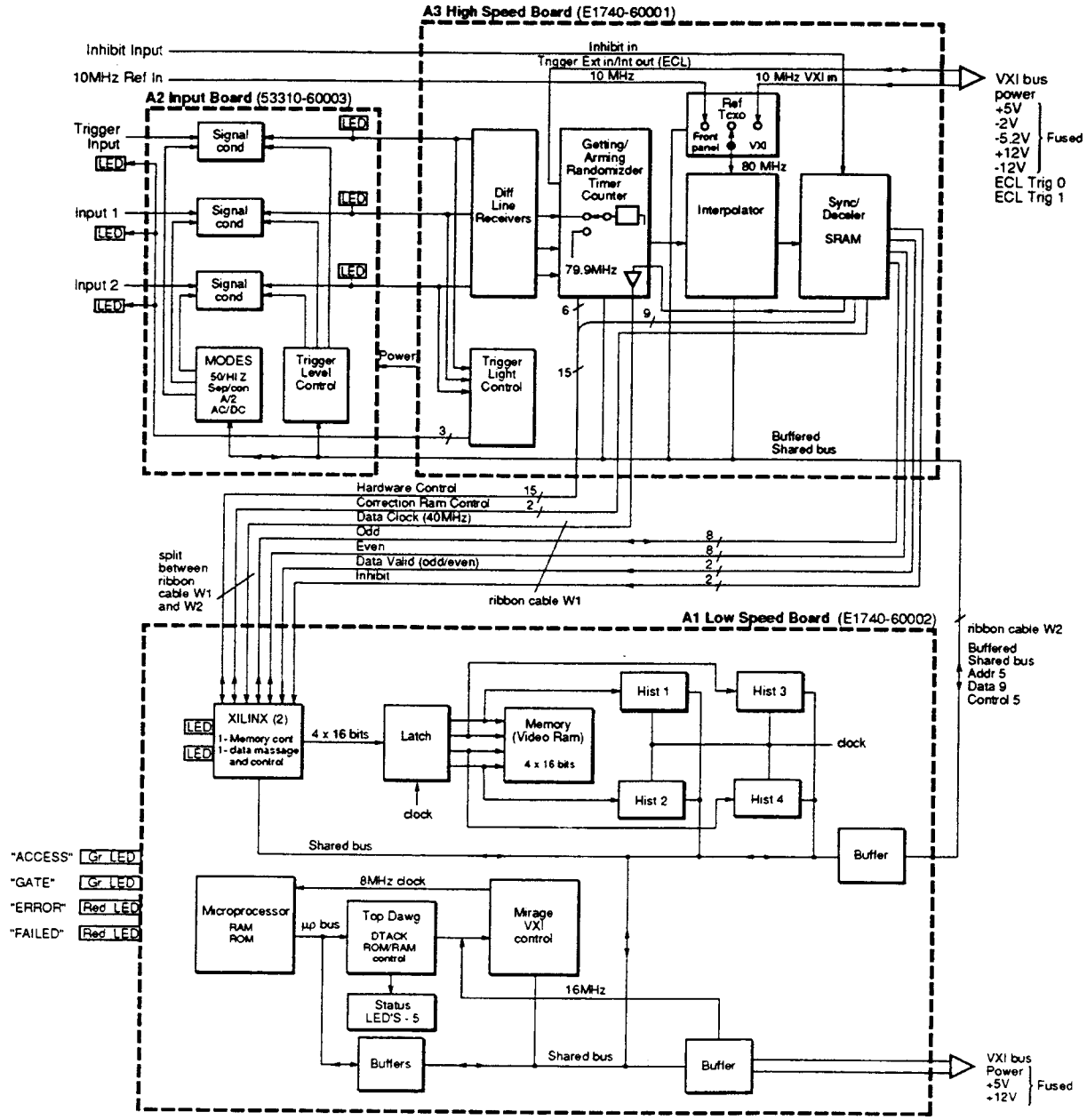


Figure 16-5. E1740A Block Diagram

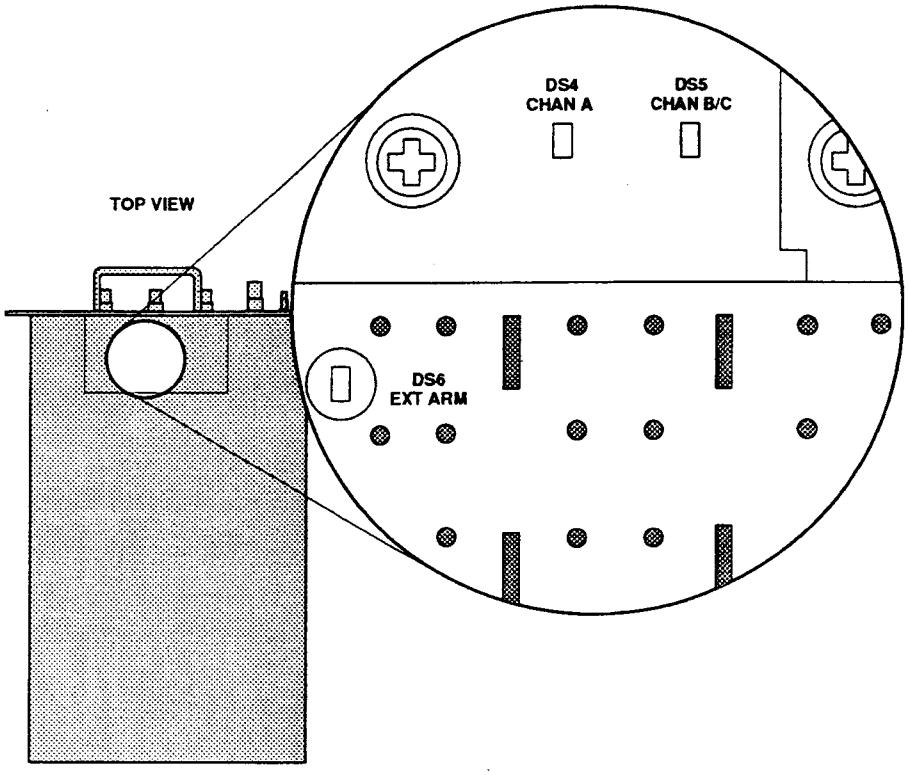


Figure 16-6. A2 Board Trigger LEDs

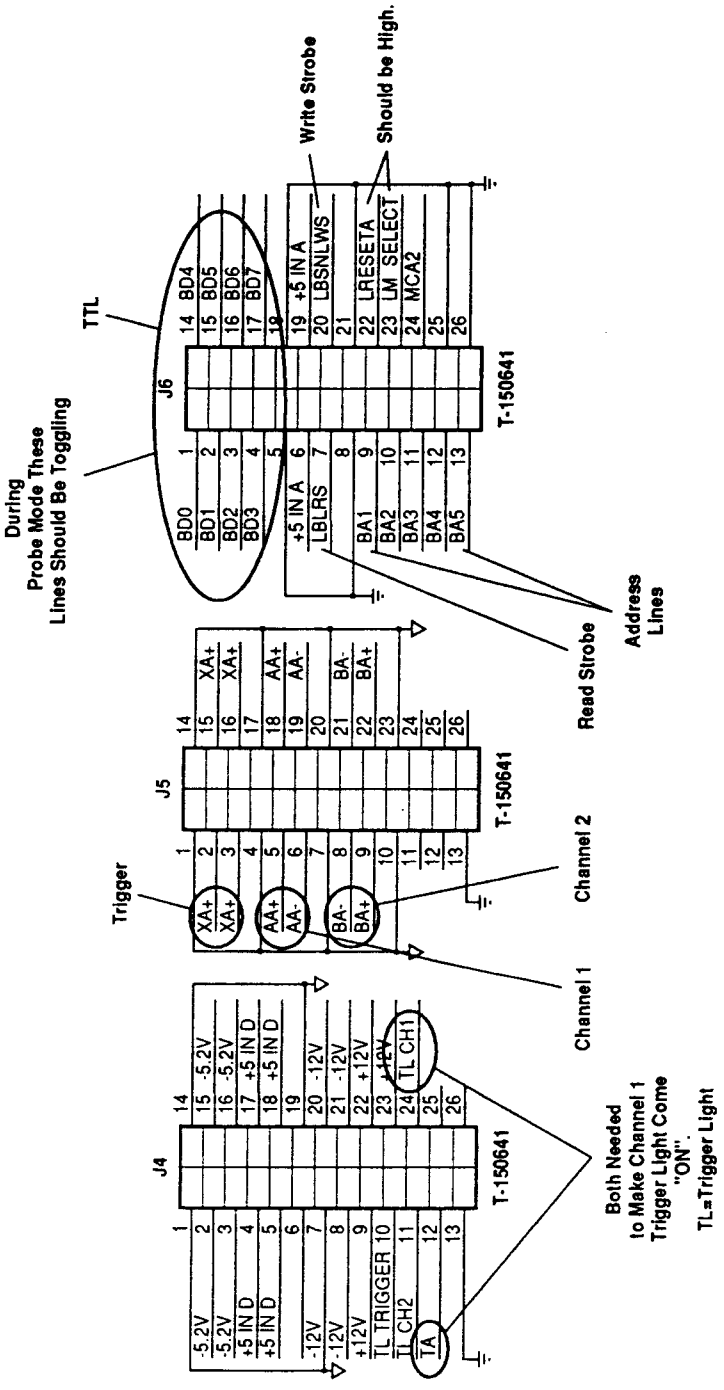


Figure 16-7. Probe Points on the A2 Board

Disassembly and Reassembly of the HP E1740A

NOTES

1. Refer to Figures 16-8A through 16-8D for locating the items referred to in the disassembly and reassebly procedures.
2. [ ] indicate locations of hardware referenced in the disassembly and reassembly procedures.
3. Refer to Table 16-4 (on page 16-91) for descriptions of the reference designators (H, MP, W, etc.) that are used in the disassembly and reassembly procedures.

Tools Required

- Hand TORX® 10 screwdriver (T10) (see note below)
- 5/8-inch spin tight—for the **Inhibit Input** BNC connector.
- 9/16-inch spin tight—for the three **Trigger Input** BNCs.
- 1/4-inch spin tight—for the **10 MHz Ref.** In SMC connector.

NOTE

The TORX® 10 screwdriver is required for removing all the other screws mentioned in the procedures of this section.

Disassembly Procedures

Top Cover (MP2) Removal

1 [A-1] Remove the top cover screw (H4) as shown in Figure 16-8A.

2 [A-2] Remove the front panel screw (H2) as shown in Figure 16-8B.

3 [A-3] On the non-labeled side of the instrumentt, remove screw H9 as shown in Figure 16-8C.

CAUTION

On reassembly, be sure to use this H9 screw. Use of the similar appearing H2 screw will cause a short to a board trace due to the length.

4 [A-4] Remove six top cover screws (H2) as shown in Figure 16-8D.

5 Slide top cover (MP2) back so it is clear of the rear connectors and lift off as shown in Figure 16-8D.

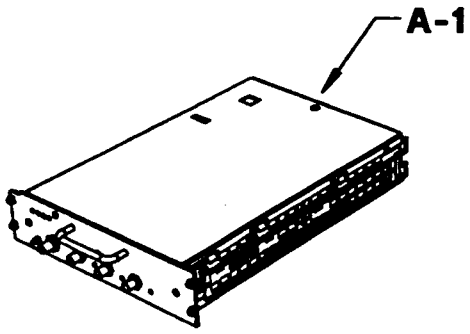


Figure 16-8A. HP E1740A Cover Removal

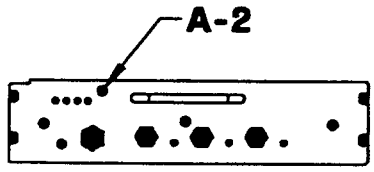


Figure 16-8B. HP E1740A Cover Removal

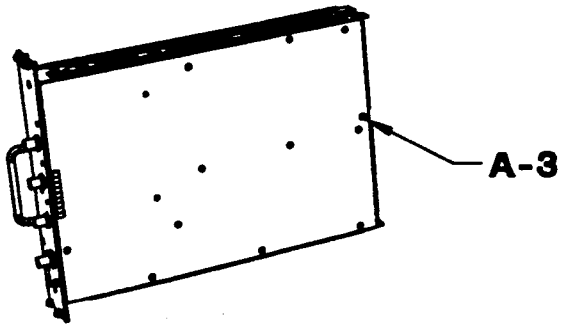


Figure 16-8C. HP E1740A Cover Removal

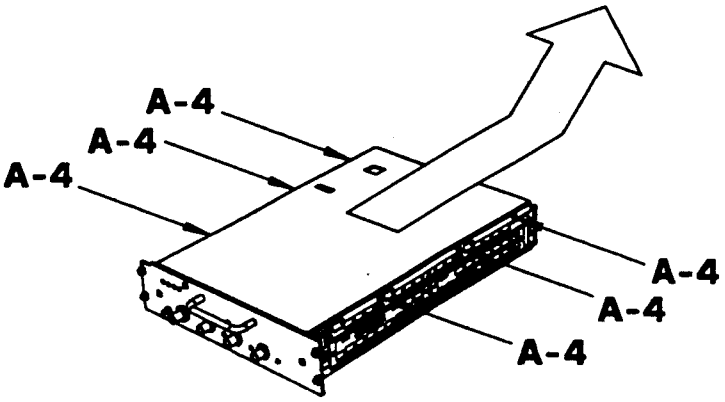


Figure 16-8D. HP E1740A Cover Removal

A1 Low Speed Board Removal

1 [B]: Remove single mounting screw (H3) at the front center of A1 board as shown in Figure 16-9.

2 Remove cables W1, W2, and W3 from A1 board. See exploded view (Figure 16-12) at the end of this chapter.
The ribbon cables W1 and W2 may be tight, and can be removed by "working" them from side to side. The coax cable W3 pulls off its mating connector.

3 Slide the A1 board away from the front panel about 1/2" so notches in the sides of the board align with the lanced card guides in the sides of the chassis. Lift the up the edge opposite that of the ribbon cables, and then lift out the board.

Front Panel Removal (MP3)

1 [C1]: Remove the five hex nuts (H6, H7— in three places, H8,) on the BNC and SMC connectors. See Figure 16-12.

2 [C2]: Remove the three screws (H2, in 3 places) as shown in Figure 16-12.

3 Slide panel off the connectors.

A2 Input Board Removal

1 [D]: Remove two screws (H3) as shown in Figure 16-10.

2 Gently pull up on the rear edge of the board, "working" the connectors, loose. Slide the board out of the chassis front panel holes.

NOTE Installation of flat washers (H10) and lock washers (H11) are required for proper grounding. Be sure the hardware is reinstalled when returning the A2 board.

A3 High Speed Board Removal

1 [E]: Remove hex stand-off (H1) as shown in the exploded view (Figure 16-12) at the end of this chapter.

2 [F]: Remove the ten screws (H3) as shown Figure 6-11.

3 Lift out the board.

NOTE Installation of flat washers (H10) is required for proper grounding. Be sure the hardware is reinstalled when re-installing the A3 board.

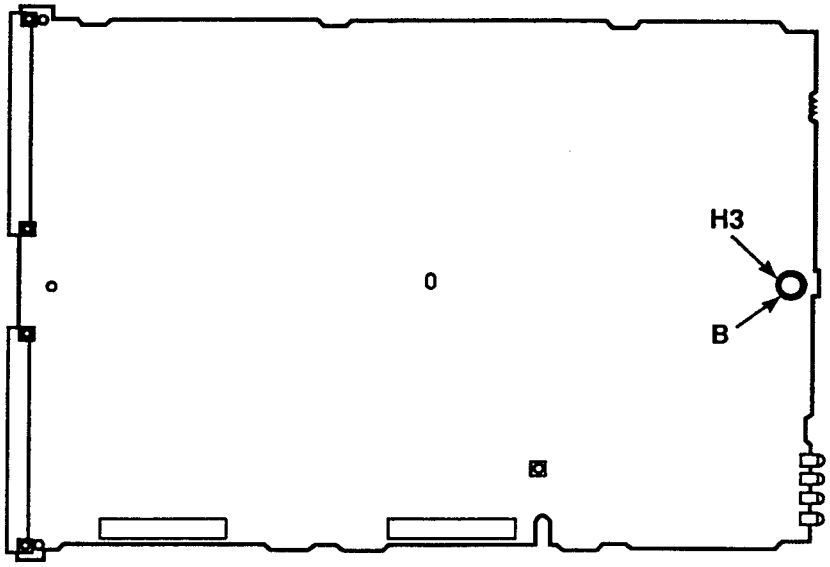


Figure 16-9. A1 Low Speed Board Removal

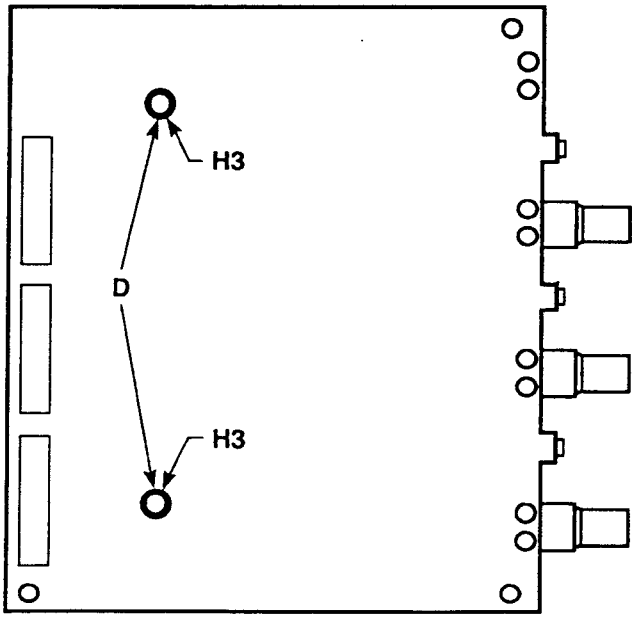


Figure 16-10. A2 Input Board Removal

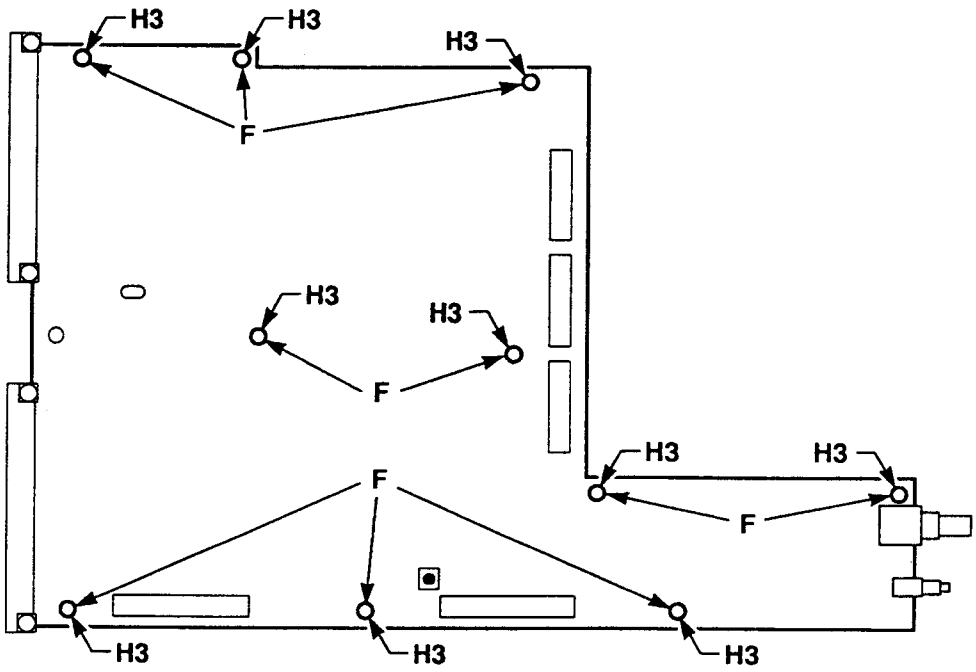


Figure 16-11. A3 High Speed Board Removal

Reassembly Procedures

1 Reassembly is essentially in reverse order.

Use care with the cables and connectors to insure that no cables are pinched or pins misaligned during re-insertion.

2 Use care with tools so as not to scratch finished surfaces.

3 When re-installing A2 onto A3, use care that all pins of the mating connectors are properly aligned. Be sure that washers H10 and H11 are replaced in proper position.

4 Observe caution for screw H9. If the similar appearing H2 screw is used, a short will be created when the longer H2 screw contacts a trace on the PC board inside.

Changing the RFI Fingers (MP4):

1 Pry out the remaining black plastic buttons if they are still present.

2 Before installing the new fingers, slide the finger stock off the mounting strip. The new fingers CANNOT be installed if this step is not done.

3 Place the mounting strip in place over the holes. Using the handle end of a screwdriver, press the plastic buttons into the holes.

4 Slide the finger stock onto the mounting strip.

HP E1740A Replaceable Parts

Introduction

This chapter contains information for ordering parts. Table 16-3 lists the exchange assemblies, and Table 16-4 is a list of reference designations used in the parts list and throughout this chapter. Table 16-5 lists all replaceable assemblies and parts for the HP E1740A Time Interval Analyzer.

Exchange Assemblies

Table 16-3 lists assemblies within the HP E1740A that may be replaced on an exchange basis. Factory repaired and tested exchanges assemblies are available only on a trade-in basis. Defective assemblies must be returned for credit. (Note that the part numbers for the new and exchange assemblies are also listed in Table 16-3.)

Table 16-3. Exchange Assemblies

Assembly	New Assembly HP Part No.	Exchange Assembly HP Part No.
A1 Low Speed Board	E1740-60002	E1740-69002
A2 Input Board	E1740-60004	E1740-69004
A3 High Speed Board	E1740-60001	E1740-69001

Reference Designations

Table 16-4 lists the reference designations used in the parts lists, block diagrams, and through this chapter.

Table 16-4. Reference Designations

A = assembly	DS = annunciator; signaling device (audible or visual) lamp; LED	K = relay	TB = terminal board
AT = attenuator; isolator; termination	E = miscellaneous electrical part	L = coil; inductor	TC = thermocouple
B = fan; motor	F = fuse	M = metre	TP = test point
BT = battery	FL = filter	MP = miscellaneous mechanical part	U = integrated circuit; microcircuit
C = capacitor	H = hardware	P = electrical connector (movable portion); plug	V = electron tube
CP = coupler	HY = circulator	Q = transistor; SCR; triode thyristor	VR = voltage regulator; breakdown diode
CR = diode; diode thyristor; varactor	J = electrical connector (stationary portion); jack	R = resistor	W = cable; transmission path; wire
DC = directional coupler		RT = thermistor	X = socket
DL = delay line		S = switch	Y = crystal unit — piezo-electric
		T = transformer	Z = tuned cavity; tuned circuit

Replaceable Parts Table

Table 16-5 is a list of replaceable parts and is organized as follows:

Electrical assemblies in alphanumerical order by reference designation.

1. Electrical assemblies in alphanumerical order by reference designation.
2. Chassis-mounted electrical parts in alphanumerical order by reference designation.
3. Chassis-mounted mechanical parts in alphanumerical order by reference designation.

The information given for each part consists of the following:

1. Reference designation.
2. Hewlett-Packard part number.
3. Part number check digit (CD).

4. Total quantity (QTY) in instrument. The total quantity is given once and at the first appearance of the part number in the list.
5. Description of the part.
6. Typical manufacturer's part number for the part.

How To Order A Part

Hewlett-Packard wants to keep your parts ordering process as simple and efficient as possible. To order parts perform the following steps:

1 Identify the part and the quantity you want.

2 Determine the ordering method to be used and contact Hewlett-Packard.

Parts Identification

To identify the part(s) you want, first refer to the exploded views (Figure 16-12) at the back of this chapter.

When ordering from Hewlett-Packard, the important numbers to note from the Parts List are the HP Part Number and part-number check digit (in the "CD" column), and the quantity of the part you want.

If the part you want is NOT identified in the manual, you can call on Hewlett-Packard for help (see the following section titled "Contacting Hewlett-Packard"). Please have the following information at hand when you contact HP for help:

- Instrument Model Number (example, "HP E1740").
- Complete instrument Serial Number (example, "1234A56789").
- Description of the part and its use.
- Quantity of the part required.

Contacting Hewlett-Packard

Depending on where you are in the world, there are one or more ways in which you can get parts or parts information from Hewlett-Packard.

- Outside the United States, contact your local HP sales office.

- Within the United States, we encourage you to order replacement parts or request parts information directly by telephone or mail from the HP Support Materials Organization, using the telephone numbers or address listed on the next page. (You can also contact your local HP sales office. HP sales offices are listed at the back of this package.)

  By Telephone:

  - For Parts Ordering use our toll-free number, (800) 227-8164, Monday through Friday (except Holidays), 6 a.m. to 5 p.m. (Pacific Time).

    If you need a part in a hurry, an extra-cost Hotline phone ordering service is available 24 hours a day. Use the toll-free number above at the times indicated; at other times, use (415) 968-2347.

  - For Parts Identification Assistance, call us at 9916) 783-0804. Our Parts Identification hours are from Monday through Friday, 6 a.m. to 5 p.m. (Pacific Time).

  For mail correspondence, use the address below:

  Hewlett-Packard
  Support Materials Roseville
  P. O. Box 1145
  Roseville, CA 95661-1145

Cabinet Parts and Hardware

To locate and identify miscellaneous cabinet and chassis parts and instrument hardware, refer to Figure 16-12. This figure provides an exploded view of the instrument, with the parts identified by reference designations; the reference designations correspond with the ones in Table 16-5.

Table 16-5. Replaceable Parts

Reference Designation	HP Part Number	C D	Qty	Description	Mfr Code	Mfr Part Number
EXCHANGE ASSEMBLIES						
					28480	E1740-60002
A1	E1740-60002	8	1	LOW SPEED BOARD	28480	E1740-69002
	E1740-69002	6	1	LOW SPEED BOARD (RESTORED)		
					28480	E1740-60004
A2	E1740-60004	0	1	INPUT BOARD	28480	E1740-69004
	E1740-69004	8	1	INPUT BOARD (RESTORED)		
					28480	E1740-60001
A3	E1740-60001	7	1	HIGH SPEED BOARD	28480	E1740-69001
	E1740-69001	1	1	HIGH SPEED BOARD (RESTORED)		
CHASSIS PARTS						
H1	0380-3026	2	1	STANDOFF-HEX M3x0.5, 29 MM	00000	ORDER BY DESCRIPTION
H2	0515-1031	2	10	SCREW-MACH M3x0.5, 6 MM, 90 deg FLT HD	00000	ORDER BY DESCRIPTION
H3	0515-0430	3	13	SCREW-MACH M3X0.5, 6 MM	00000	ORDER BY DESCRIPTION
H4	0515-1135	7	1	SCREW-MACH M3X0.5, 25 MM	00000	ORDER BY DESCRIPTION
H5	0515-1968	4	4	SCREW-MACH M2.5X0.045, 11 MM, PAN HEAD	00000	ORDER BY DESCRIPTION
H6	0590-0038	5	1	NUT-HEX-DBL CHAMFER	00000	ORDER BY DESCRIPTION
H7	2950-0035	8	3	NUT-HEX-DBL CHAMFER	00000	ORDER BY DESCRIPTION
H8	2950-0078	9	1	NUT-HEX-DBL CHAMFER, 10-32 THD	00000	ORDER BY DESCRIPTION
H9	0515-2146	2	1	SCREW-MACH M3X0.5, 4MM 90 DEG FLT HD	00000	ORDER BY DESCRIPTION
H10	3050-0962	3	4	WASHER, .503 IN I.D.X .08 IN THK 18-8 STAINLESS STLI	00000	ORDER BY DESCRIPTION
H11	2190-0068	5	3	WASHER, LOCK, INT TOOTH, .505 IN I.D. X .63 IN O.D. x.022 IN THK	00000	ORDER BY DESCRIPTION
MP1	E1740-00001	1	1	CHASSIS	28480	E1740-00001
MP2	E1740-00002	2	1	COVER	28480	E1740-00002
MP3	E1740-00003	3	1	PANEL, FRONT W/HANDLE	28480	E1740-00003
MP4	8160-0686	6	1	RFI STRIP-FINGERS	28480	8160-0686
W1	53310-60201	1	2	CABLE AY-CPU	28480	53310-60201
W2	53310-60201	1		CABLE AY-CPU	28480	53310-60201
W3	8120-5020	8	1	CABLE AY-COAX	28480	8120-5020

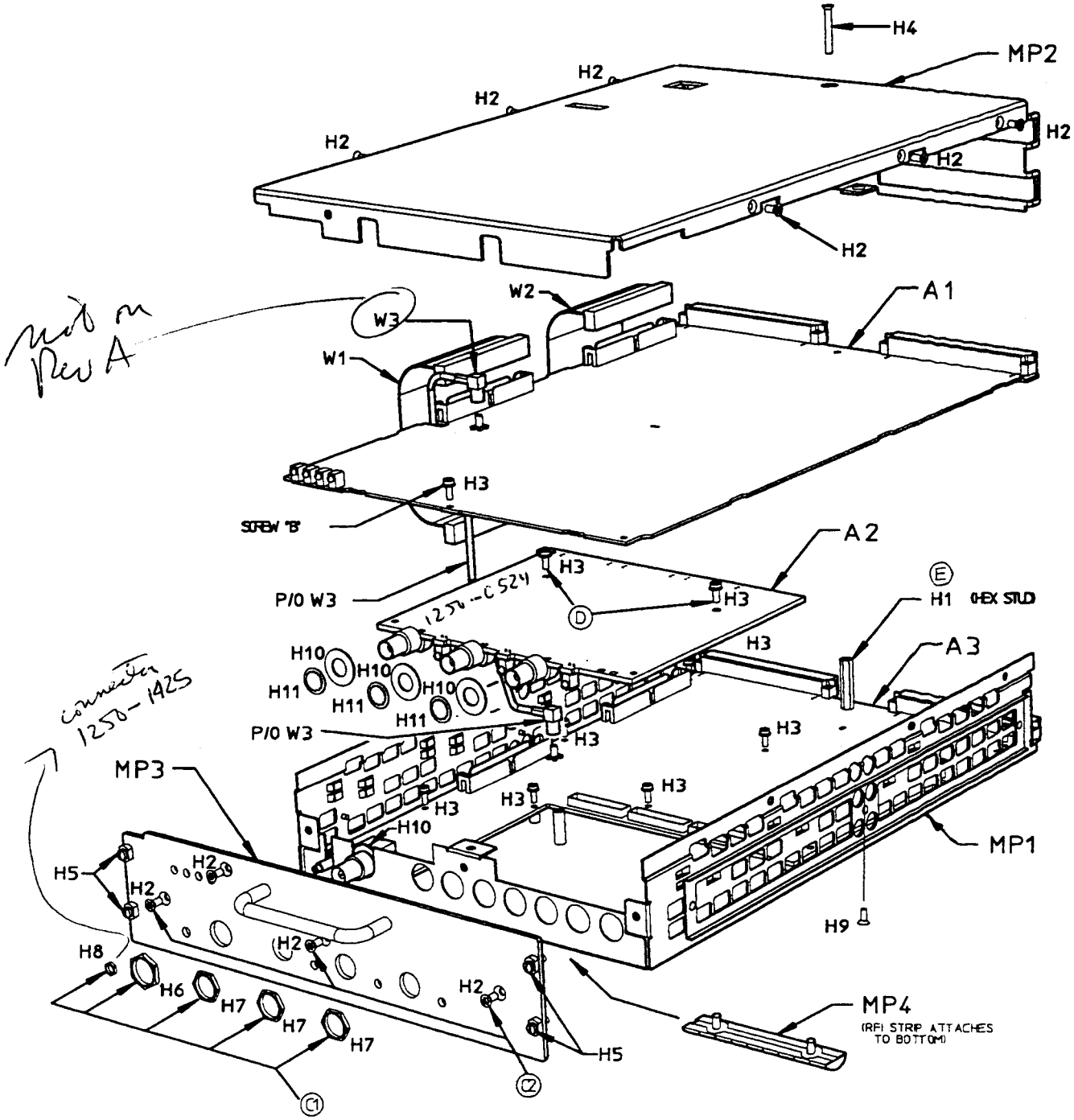


Figure 16-12. Parts & Assembly Relationship

17

Specifications

Specifications & Operating Charactistics

Warranted specifications are indicated in *italics*. All other operating characteristics are typical.

Measurements

Time Interval
1→1, 1→2, 2→2
Frequency
Period (not HP E1740A)

Timing characteristics

Minimum TI 1→2: 0.0 ns
Minimum TI 1→1 or 2→2:
>12.5 ns continuous (every edge on a <80 MHz signal)
8 ns non-continuous (TI 1→1 only)
(2 adjacent edges followed by a ≥30 ns delay on a <125 MHz signal)
Maximum TI: 3.2 µs to 26.2 ms depending on selected resolution (LSB)
Latency after TI 1→2: 30 ns
Resolution (LSB): 50 ps
(variable in binary steps up to 400 ns to increase maximum TI)
RMS jitter: <100 ps
Peak-peak jitter (for 10^{10} measurements): 500 ps

Memory (number of measurements)

Fast Histogram (single or dual): 10^{12}, accumulation to 10^{15}
Single channel sequential (1→1 or 2→2): 512K
Dual channel sequential (1→2): 256K

Arming (acquiring the right edges)

Delay after trigger (holdoff):
Time (100 ns-26 ms),
Edges (1-1 million), or
None
Delay resolution:
25 ns or 1 edge
Measurement duration:
Number of measurements or
Time (25 ns to 26 ms, settable in 25 ns steps)
Re-arm time: <500 ns
Pacing:
Edges (1-1 million),
Random number of edges
Inhibit: Yes

Input characteristics

Inputs 1 & 2
Bandwidth: 200 MHz
Maximum input frequency: 150 MHz
Impedance: 50Ω or 1 MΩ switchable
Slope: ± (1 only), +, –
Sensitivity:
60 mV p-p @ 0% hysteresis
100 mV p-p @ 50% hysteresis
Noise: < 600 µV rms
Trigger Input
Impedance: 1MΩ
Slope: +,-
[1] Threshold levels: ECL, ECL/10, 0V, TTL, TTL/10
Inhibit input
Impedance: 10kΩ
Level: ECL, 0V, TTL
Active: high or low

General

Configuration

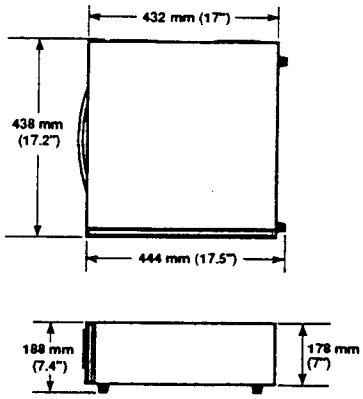
HP E1725A: Standalone instrument
HP E1740A: VXI Card, C-size
Power requirements
HP E1725A, excluding display: 100/120/220/240 VAC (±10%, autoranging), 50/60 Hz, <360 VA, <240 W
HP D1194A display: 100-240 VAC (autoranging), 50/60 Hz, 110 VA, 70 W
HP E1740A (VXI cage supplies)

+5	Vdc:	<5A
+12	Vdc:	<.16A
-12	Vdc:	<.16A
+24	Vdc:	0
-24	Vdc:	0
- 5.2	Vdc:	<6.0A
-2	Vdc:	<1.6A

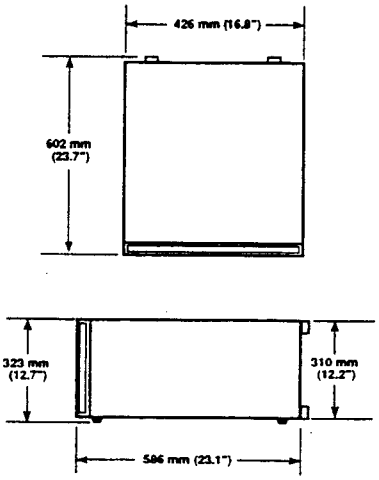
Temperature: 0°-50° C
Weight
HP E1725A (including display and accessories): 34 kg (75 pounds)
Option 001: 18 kg (40 pounds)
Option 002 (including display and accessories): 43 kg (95 pounds)
Option 001 + 002: 28 kg (61 pounds)
HP E1740A: 4 kg (8 pounds)
Frequency Reference
Internal 10 MHz Oscillator, phase-lockable to VXI CLK10 or front panel 10 MHz Reference Input.
Warranty: 3 Years

Dimensions

HP E1725A standard (without display and accessories), or Option 001



HP E1725A Option 002

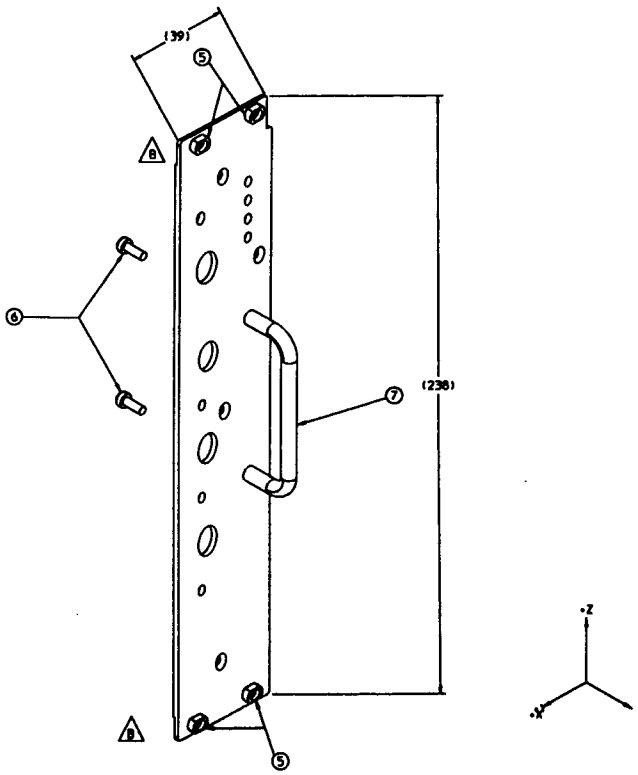


[1] Note: VXI ECLTRIG lines can also be used to trigger the HP E1740A.

REFERENCE DRAWINGS:

a_E1740-00003-0	ME30 SOLID MODEL ASSEMBLY REV. B
1_E1740-00003-2	MARKING DRAWING
s_E1740-00003-0	PESD SOLID MODEL - SHEETMETAL REV. B

DWG RESPONSIBILITY: 02				D-E1740-00003-1
DWG	REVISIONS		APPROVED BY	DATE
A	AS ISSUED		G. TSAI	08-05-93
B	WAS 2.5 mm NOW MILL OFF .79 mm X 10.5 mm BOTH ENDS FAR SIDE. PER PCO 2-22653		JA SATOH	04-17-96
	ADDED REFERENCE DIMENSIONS			



NOTES:

1) SAND BOTH SIDES.
2) CLEAR CHROMATE CONVERSION PER A-5951-1522-1 (606H)
3) FINISH:
- -PAINT NEARSIDE PARCHMENT WHITE PER HP VISUAL STD. 6009-0130.
- -SILKSCREEN CHARCOAL BLACK PER HP VISUAL STD. 6009-0158
- -PAINT NEARSIDE CLEAR BAKE TEXTURE ENAMEL PER HP VISUAL TEXTURE PANEL FINISH STD. 6009-0156.
- -OVERSPRAY PERMITTED ON FARSIDE AND ALL EDGES.

4) COSMETIC REQUIREMENTS IIB PER A-5951-1515-1 (SECTION 741.100G)
5) CONTROL DRAWING FOR PART GEOMETRY IS THE PESD SOLID MODEL. ALL GEOMETRY IN THIS DRAWING IS FOR REFERENCE ONLY.

ITEM	QTY	MATERIAL DESCRIPTION	MATL PART NO.	MATL DWG NO	MATL SPEC
07	01	HANDLE	RAF#8110-632-A-7		
06	02	SCREW, 6-32	2360-0193		
05	04	FASTENER CAPTIVE SCREW	1390-0867		
04	A/R	INK, CHARCOAL BLACK	9260-0902		EPOXY
03	A/R	PAINT, HI-TEMP BAKE WATER BORNE	6010-1277		ENAMEL
02	A/R	PAINT, HI-TEMP BAKE WATER BORNE	6010-1311		ENAMEL
01	A/R	ALUMINUM 2.5 THICK	7204-0026		5052-H32

METRIC — DO NOT SCALE THIS DRAWING

UNLESS OTHERWISE SPECIFIED, DIMENSIONS ARE IN MILLIMETRES
HOLE-TO-HOLE ±0.25mm FOLD-TO-HOLE ±0.5mm
EDGE-TO-HOLE ±0.3 mm FOLD-TO-EDGE ±0.4mm
EDGE-TO-EDGE ±0.4 mm FOLD-TO-FOLD ±0.6mm
HOLE SIZE ±0.08mm FOLD ANGLE ±1.5°
ADHERENCE TO DMS A-5951-1341-1 (OLSTD SECT.008) REQUIRED

THIRD ANGLE PROJECTION

DRAWN	GEORGE TSAI	08-05-93
CHECKED	GEORGE TSAI	08-05-93
APPROVED	BOB PECK	08-05-93
ENGINEERING CONTROL	-	
RELEASED TO PROD.	-	

PANEL FRONT

HEWLETT PACKARD

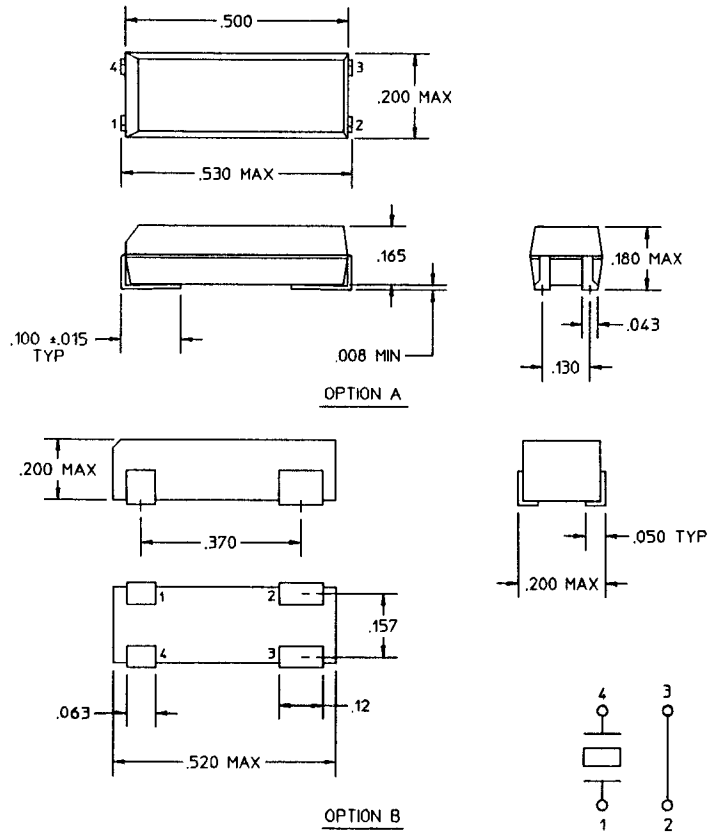
E1740-00003

D-E1740-00003-1

SCALE 1:1 SHEET 1 OF 1

CAD SYSTEM USED: MCAD REV 4.1

HEWLETT PACKARD | SPECIFICATION CONTROL DRAWING



NOTE: ALL DIMENSIONS ARE IN INCHES, TOLERANCES AS SHOWN.

Date this issue:	CRYSTAL - QUARTZ	Page 1 of 2
Changes since previous issue noted by:		Dwg. No. Rev.
Approval:	HP Part Number 0410-2161	A-0410-2161-1 C

MADE ON HE10

HEWLETT PACKARD | SPECIFICATION CONTROL DRAWING

1. GENERAL

1.1 This specification establishes the requirements for a surface mount quartz crystal for a series mode of operation.

2. ELECTRICAL

2.1 Frequency: 20.000 MHz.

2.2 Operating Temperature Range: -20°C to +70°C.

2.3 Calibration Tolerance (at +25°C): ±0.01%.

2.4 Tolerance Across Temperature Range: ±0.01%.

2.5 Drive Level: 100 µW, 0.5 mW maximum.

2.6 Equivalent Series Resistance: 50 Ω maximum.

3. MECHANICAL

3.1 Marking: Parts shall be marked with frequency, date code, and manufacturer's name or symbol.

Date this issue:	CRYSTAL - QUARTZ	Page 2 of 2
Changes since previous issue noted by:		Dwg. No. Rev.
Approval:	HP Part Number 0410-2161	A-0410-2161-1 C

MADE ON HE10

Index

R

S

T

V

W

X

Continued from front matter . . .

Warranty (Cont'd.)

For warranty service or repair, this product must be returned to a service facility designed by HP. Buyer shall prepay shipping charges to HP and HP shall pay shipping charges to return the product to Buyer. However, Buyer shall pay all shipping charges, duties, and taxes for products returned to HP from another country.

HP warrants that its software and firmware designed by HP for use with an instrument will execute its programming instructions when properly installed on that instrument. HP 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 the environmental specifications for the product, or improper site preparation or maintenance.

NO OTHER WARRANTY IS EXPRESSED OR IMPLIED. HP SPECIFICALLY DISCLAIMS THAT IMPLIED WARRANTIES OR MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Exclusive Remedies

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

Assistance

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

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

Safety Considerations (Cont'd.)

WARNING

INSTRUCTIONS FOR ADJUSTMENTS WHILE COVERS ARE REMOVED AND FOR SERVICING ARE FOR USE BY SERVICE-TRAINED PERSONNEL ONLY. TO AVOID DANGEROUS ELECTRIC SHOCK, DO NOT PERFORM SUCH ADJUSTMENTS OR SERVICING UNLESS QUALIFIED TO DO SO.

Acoustic Noise Emissions

LpA<47 dB at operator position, at normal operation, tested per EN 27779. All data are the results from type test.

GERAeUSCHEMISSION

LpA<47 dB am Arbeits platz, normaler Betrieb, geprueft nach EN 27779. DieAnbagen beruhen auf Ergebnissen von Typpruefungen.