Agilent Technologies E4480-90001
SCPI Programmer's Manual

156MTS SONNET
Maintenance Test Set

Agilent Technologies

Publication part number: E4480-90001
Version 4.2
May, 2000

Printed in USA.

Warranty

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

Agilent Technologies makes no warranty of any kind with regard to this material, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose.

Agilent Technologies shall not be liable for errors contained herein or for incidental or consequential damage in connection with the furnishing, performance, or use of this material.

Trademark Credits

Windows® is a trademark of Microsoft Corporation. SLC®-96 is a registered trademark of AT&T. All other trademarks are the property of their respective owners.

ISO Certification

Agilent Technologies Service Test Division (STD) is an International Standards Organization (ISO) 9001 registered company, recognizing that the quality system operated by STD meets the requirements established in accordance with ISO 9001. The CERJAC 156MTS test sets are manufactured STD in strict compliance with this quality system.

STD received ISO 9001 certification (no. 6969) from National Quality Assurance (NQA) on August 30, 1995.

Certificate No. 6969

Communications Measurements Division
2 Robbins Road
Westford, Massachusetts 01886 USA

About This Book

This manual is your guide to programming with SCPI commands to control CERJAC test instruments.

Chapter 1 introduces the SCPI command language and describes how to enter commands.

Chapter 2 describes how to connect a CERJAC test set to a controller and send messages from the controller to the test set.

Chapter 3 outlines how to plan and program a test using SCPI.

Chapter 4 lists the SCPI commands to use for duplicating the front-panel operation test modes.

Chapter 5 describes the “common” SCPI commands.

Chapters 6 through 14 list all the SCPI commands and associated parameters.

Chapter 15 describes error codes you may see after issuing a SCPI command or command.

Chapter 16 contains several sample SCPI programs and a corresponding program written in C programming language.

Chapter 17 provides general background information on the SCPI/HP-IB interfaces and standards.

Chapter 18 contains block diagrams that illustrate how the SCPI commands interact to control the CERJAC instrument. You can use these diagrams to help you determine the SCPI commands to enter to perform a particular test.

Appendix A contains the Software Release Notes from 6.41 to 7.0 that are incorporated in this manual.

Index Section list three level of SCPI reference catagories set in alphabetical order and keyed to the appearing page numbers.

About this Version

SCPI/HP-IB Programmer's Manual printing history

Version	Release date	Notes
2.0	*November, 1993*	
3.0	*September, 1995*	
3.1	*March, 1996*	
3.2	*October, 1996*	
4.0	*November, 1997*	
4.1	*March, 1999*	
4.2	*May, 2000*	*Update to include latest commands*

This version of the *Programmer's Reference Guide* applies to CERJAC 156MTS test sets that have option UHR and are running Host software version **7.00**. These test sets include:

- HP CERJAC 156 (E4280A)
- HP CERJAC 156MTS (E4480A)
- HP CERJAC 31XE (E4487A)

Some instruments running earlier software versions may not support all functions described in this manual; some functions of instruments running later software versions may not be covered by this manual (refer to any user supplements or contact Agilent Technologies at 1-800-923-7522).

Conventions

The following conventions are used in this manual:

SCPI Common Commands

SCPI common commands always begin with an asterisk. For example, *RST. For more information, see Chapter 5, *Common SCPI Commands Reference*.

SCPI Commands

SCPI commands have both a long and abbreviated version. In this manual, the long and abbreviated versions are shown together. The abbreviated version is in ALL CAPS while the remaining letters that comprise the long version are in lower case. For example, :FETCh where **:FETC** is the abbreviated version and **:FETCH** is the long version.

The SCPI command interpreter in the test set is **not** case sensitive, but you must enter either the long or the abbreviated version of a command not a combination. You can enter the :SYSTem:ERRor? command *only* as :SYST:ERR? or :SYSTEM:ERROR?.

Parameters

Parameters that apply to a command are shown in angle brackets < >. For example :FETC:DMOD:ATM? <*results*> — where <*results*> represents the parameter associated with the command.

Multiple parameters are separated by a comma. If multiple parameter choices are listed in the brackets, they are separated by a vertical bar. For example, :SOUR:DM:ALARM <*alarm type*>**,**<ON|OFF>.

When you enter a SCPI command, do not type the angle brackets or the vertical bar.

Introduction

This Agilent CERJAC SCPI/HP-IB Programmer's Manual (Part Number 09-0600-0005) consists of 18 Chapters organized to best present programming information to you in the SCPI (Standard Command for Programmable Instruments) command language to control the CERJAC 156MTS test set.

The SCPI commands allow you to program and monitor the test set through an RS-232 or HP-IB interface remote connection.

Besides acting as a SCPI interface, the RS-232 interface provides an alternate remote control capability. When attached to a terminal or PC running as a terminal emulator the port maintains a Remote Front Panel which mirrors the test set display. Front panel key presses are simulated by key strokes at the terminal emulator keyboard.

The field portable Agilent CERJAC 156MTS test set provides SONET, ATM, and T-carrier testing with flexible configuration for network testing requirements for installation, qualification and maintenance testing of OC12 to DS0.

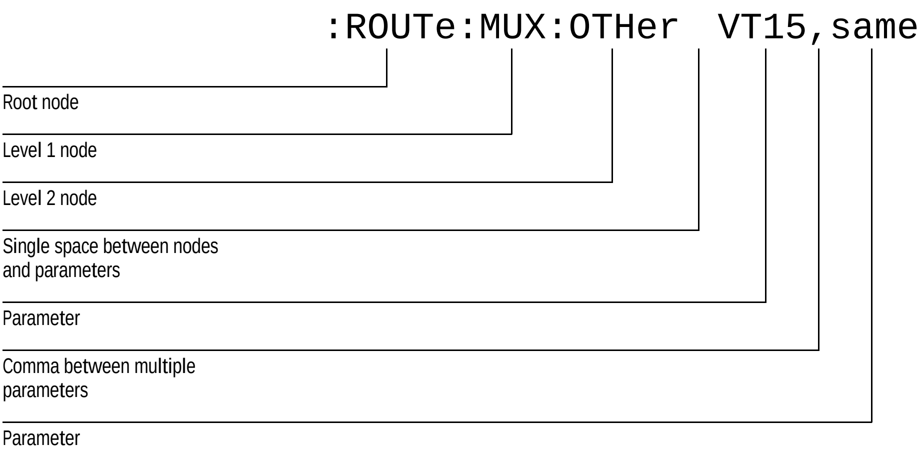
Contents

1

SCPI Basics

SCPI Commands at a Glance

Each SCPI command consists of a root node, one or more lower level nodes, followed by applicable parameters (see figure).



The test set uses this structure to interpret the command. Generally, each root and lower level node is proceeded by a colon (:). This helps the instrument correctly parse the command's component parts.

For example, you can enter the following command:

:INPUT:FILTER:DS3 DSX3

The root node is :INPUT, the level 1 node is :FILTER, the level 2 node is :DS3, and the associated parameter is DSX3.

Root Nodes

SCPI commands are used to make measurements, retrieve data, and query the status of an instrument. SCPI commands are hierarchical in that a command *root node* often has several additional node levels that complete the command.

There are ten groups, or root nodes, of SCPI commands for controlling CERJAC test sets:

:FETCh: Retrieve test results values from the output queue.

:INITiate: Clear the status registers, start a test, and begin accumulating results in the status registers.

:ABORt: Stop a test and freeze the final results.

:INPut: Control the characteristics of the input ports on the test set.

:OUTPut: Control the characteristics of the output ports on the test set.

:ROUTe: Set up signal paths in the test instrument.

:SENSe: Configure the receive functions of the test set.

:SOURce: Configure the transmitter functions of the instrument.

:STATus: Retrieve values from the various SCPI registers.

:SYSTem: Retrieve error codes, software version, and exit SCPI mode, STORE configurations.

Common Commands

A subset of the SCPI commands are the *common commands*. Common commands begin with an asterisk (*) and are used to manage macros, status registers, synchronization, and data storage. The common command you will use most often is the *RST (reset) command. *RST activates SCPI mode and automatically sets all parameters to their default settings. See Chapter 5, *Common SCPI Commands Reference*, for a list of supported SCPI common commands.

Entering Commands

Long Form and Short Form

SCPI commands have both a long and short version; for example :SOURCE and :SOUR. The instrument responds to either version, but will not respond to variations of either version.

The SCPI interface does not differentiate between upper-case and lower-case letters, but only the long or short form of a command is valid.

SCPI Command Entry Format—Example command = ":SOURce"

	Command Entry	
Correct Entry	`:SOURCE`	`:sour`
	`:SOURce`	`:SOUR`
Incorrect Entry	`:SOU`	`:sourc`

Using Parameters

Parameters follow the nodes of commands and are specified throughout this reference in angle brackets (< >). If more than one parameter is available, they are listed in this manual separated by a vertical bar (|). If a command requires more than one parameter, it is entered with the parameters separated by a comma (**,**).

There are five types of parameters that are used with SCPI commands as described in the following table.

Types of Parameters

Parameter Types	Description
Numeric	*All commonly used decimal numbers including optional signs, decimal points, and scientific notation. You can also specify a numeric parameter in hex, octal, and /or binary. Special cases also include MAXimum and MINimum as values.*
Discrete	*Values are represented using a keywords instead of numbers. For example, INTernal and EXTernal.*

Types of Parameters, continued

Parameter Types	Description
Boolean	*A single binary condition that is either true of false. For example, 1 and 0.*
String	*Any set of ASCII characters enclosed in single or double quotes. For example, '10101010' or "DQUOTE;10101010".*
Block	*Used to transfer large quantities of related data. You send blocks as definite length blocks (#<numeric><numeric>) or indefinite length blocks (#0).*

Separating Commands and Parameters

The following table lists the separators to use between commands and parameter.

SCPI Command Separators

To Separate...	Use...	Example
A root node from a lower level node	*Colon (:)*	*SOURce:DM*
A node from another node	*Colon (:)*	*:SOURce:DM:DATA*
Multiple commands entered in a program message	*Semicolon (;)*	*:INPut:FILTer:DS3 3.5;:OUTPut:FILTer:DS3 7.2*
A parameter from a command	*Space*	*:SENSe:DM:FRAMe:TYPE1 esf*
Multiple parameters	*Comma (,)*	*ROUT:MATR STS#1, VT15*

Sending Multiple Commands in a Command String

You can send multiple commands in the same command string. The commands execute one at a time, but this method saves time when you are entering many SCPI commands. To separate the commands, use a semicolon (;).

Here's an example of a command string that contains more than one command:

```
:ROUT:SEL DS3_Tx,Source;:INIT;FETC:DMON:DS3? signal
```

When the test set receives this command string, it executes the three commands separately, and returns the appropriate value in response to the :FETCh command. The number of commands you can enter in a single string is limited by the number of characters you can enter in the command line provided by the HP-IB interface or the terminal emulation package you are using.

Note: *You cannot use the *RST command when stringing multiple SCPI commands together. The *RST command resets the unit to the factory default values. Any commands earlier in the string would be undone; any commands later in the string would be lost.*

Using Query Commands to Retrieve Results

A subset of the SCPI commands, known as Query commands, informs the instrument to prepare information to be transmitted back to the system controller. Query commands all end with a question mark. The question mark comes after the last keyword in the command, and before the first parameter. For example:

```
:FETCh:DMOD:VT15? pntr_value
```

Query commands instruct an instrument to retrieve the current value of the specified parameter, and place it in the output buffer in preparation for transmission back to the controller. The controller can then address the test instrument to Talk, and retrieve the response over the IEEE-488 bus. (For RS-232 applications, the response is not placed in the output queue. Instead it is transmitted immediately to the controller over the serial link.)

:FETCh is the most commonly-used query command. All test results, error, and alarm information are retrieved using the :FETCh command. The test instrument also supports other query commands, including :SYST:ERR? which retrieves messages from the error queue, and :SYST:VERS? which retrieves the SCPI compatibility date.

Some :SOURce, :ROUTe, and :SENSe commands can also be used as query commands by adding a question mark (?) after the level 2 node and omitting the second parameter. The test set returns the currently programmed value for that parameter. For example, the command `:SOUR:DM:OH? DS2_XBIT` would return the current state of the transmit DS2 X-bit (either 1 or 0).

Note: *Query commands can only detect parameters that are changed using SCPI. If you exit SCPI mode and make changes using the front-panel controls, the query commands may return incorrect parameter values.*

Format of Responses

Instruments provide responses to query commands in one of four SCPI **response data types**. The types are numeric response type 1 (NR1), numeric response type 2 (NR2), numeric response type 3 (NR3), and ASCII string messages.

Numeric Response Type 1 (NR1)

Integer quantities are returned in NR1 format, which consists of a positive or negative sign followed by a variable number of decimal digits. No decimal point is sent. This format is used, for example, for counts of seconds (which never have fractional values). In addition, if a value is retrieved for a SONET overhead byte (F1, F2, K1, K2, etc.), the value is returned as a decimal integer in the range of 0 to 255.

In general, responses which are Boolean in nature (on/off, true/false, etc.) are given in NR1, with +1 indicating ON or TRUE, and +0 indicating OFF or FALSE. Most :FETCh commands will return a value of –1 if the requested result is not valid. The: SYST:ERR? query command returns NR1 format also. In this case +0 indicates "No errors, queue empty." Negative values are error codes (see *Error Code Reference*, page 15–1).

Numeric Response Type 2 (NR2)

Fixed point fractional quantities are sent in NR2 format, which consists of a positive or negative sign followed by a variable number of decimal digits. A decimal point is sent as part of the string. Quantities using this format include calculated percentages such as % Error Free Seconds.

Numeric Response Type 3 (NR3)

Exponential quantities are returned in NR3 format, which consists of:

- a positive or negative sign
- followed by a variable number of decimal digits
- followed by the letter E (to indicate Exponent, power of 10)
- followed by a positive or negative sign for the exponent
- and finally a variable number of decimal digits for the exponent.

Quantities returned in this format include calculated bit error ratios.

ASCII String Messages

A test set can also return ASCII string messages. These include, for example, APS state messages derived from the K1/K2 overhead bytes.

2

Getting Started

Connection Requirements

You can connect a test set to a controller using either an GPIB (HP-IB) interface or an RS-232 interface.

Requirements for GPIB

- An GPIB interface card installed in your PC. There are several GPIB/HP-IB (IEEE-488.1) interface cards available from Hewlett-Packard, including the HP 82340A, 82341B, and 82335B.
- GPIB interface software installed on your PC. This software typically comes with the GPIB interface card.
- The controller should not be more than 20 meters (65 feet) away from the test instrument.
- If you are connecting many test sets to a controller, do not exceed 2 meters (6 feet) times the number of devices connected to the controller.
- To increase the operating distance by 1250 meters, use an GPIB extender (order number HP 37204A).

Requirements for RS-232

- The controller should not be more than 10 meters (33 feet) away from the test instrument.
- A terminal or a terminal emulation package installed on your PC, or a control program that can transmit and receive SCPI commands.

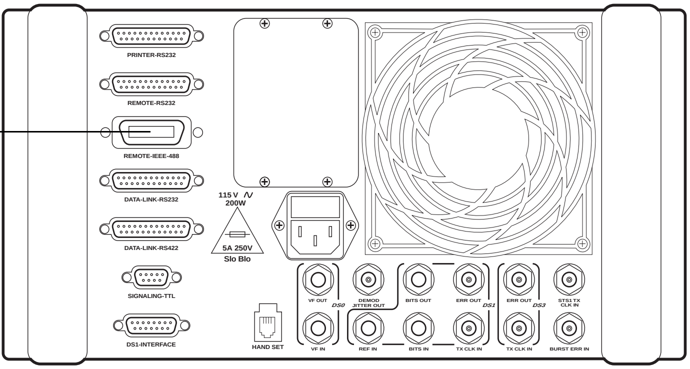
Using an GPIB Interface

To connect a test set to a controller using an GPIB interface cable, follow these steps:

1. Verify that your controller has an GPIB interface. For example, you should have installed an GPIB interface card in your PC and the corresponding software that came with the card.

2. On the rear panel of your CERJAC test set, locate the REMOTE-IEEE-488 port.

GPIB Interface Port



3. Plug one end the GPIB cable, using the pin side of the connector into the REMOTE-IEEE-488 port on the CERJAC test set.

4. Plug the other end of the cable into the connector on the GPIB interface card.

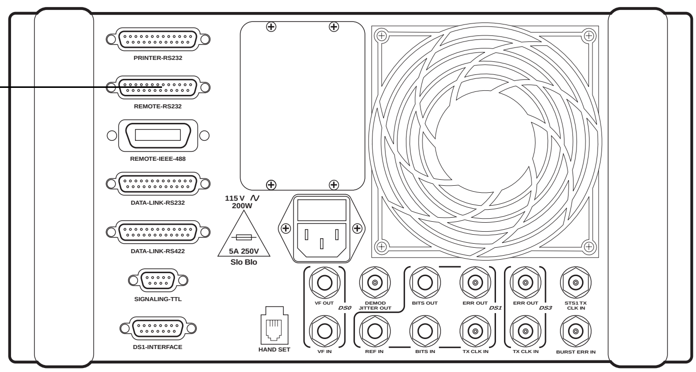
5. If you want to connect additional instruments to a controller, plug one end of the GPIB cable into the REMOTE-IEEE-488 port on the test set, and the other end of the cable into the socket side of a connector plugged into another set. In this manner, you can create a daisy-chain connection from the controller.

Using an RS-232 Interface

To connect a test instrument to a controller using an RS-232 interface cable, follow these steps:

1. On the rear panel of your CERJAC test set, locate the REMOTE-RS232 port.

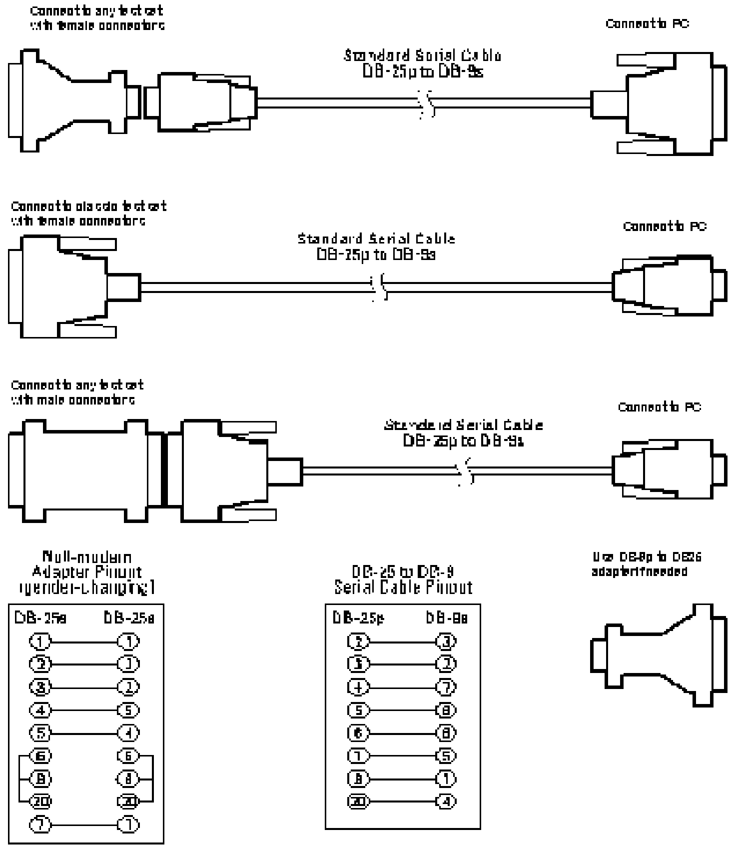
RS-232 Remote Interface Port



2. Plug one end of the serial cable into the REMOTE-RS232 data port.
3. Plug the other end of the cable into the serial port on your PC.

Note: *You may need to attach a DB-25 to DB-9 adaptor to the serial cable if the serial port on your PC requires a DB-9 connector.*

Also see RS-232 cable interface information on page 2-5



RS-232 cable and interface infomation.

Configuring the Port Settings

After you connect the test set to the controller, you must configure the test set for the port you are using. To enter port settings on the test set, follow these steps:

1. Make sure your test instrument is powered on and the Main Menu is shown on the display. (To return to the Main Menu, press the MENU-up key several times.)

```
          MODEL 156 MAIN MENU
Auto Setup
Terminal Testing
Monitor Testing
Drop & Insert Testing
DS3/DS1/ATM Scans & Pointer Sequences
Setup System Parameters
Store and Recall Configurations

----------------------------------------
Press FIELD to highlight item, then
Press MENU-Dn to select item.
```

2. Use FIELD to select the **Setup System Parameters** item and press MENU-down. The Setup System Parameters menu is displayed.

```
          Setup System Parameters

Auxiliary Test Setups
Administrative Setups
Event Logging Setups
System Software Configuration
System Hardware Configuration
Restore Factory Default Settings
Download Menu Screen
----------------------------------------
Press FIELD to highlight item, then
Press MENU-dn to select item.
```

3. Use FIELD to select **Administrative Setups** and press MENU-down.

```
Administrative Setups




Printer:      9600,8-1-None
Remote:       9600,8-1-None
Date & Time: 01/07/95  03:09:59
IEEE-488 Addr: 29
SCPI Via:      REMOTE-IEEE-488
```

4. If you are using an GPIB connection, enter information in the following fields on the Administrative Setup screen:

   - In the **IEEE-488 Addr:** field select a value from **00** through **31**. The GPIB address identifies a test instrument. This address is used in messages sent between a controller and an instrument.
   - In the **SCPI Via:** field, select **IEEE-488**.
   - Continue to step 6.

5. If you are using an RS-232 connection, enter information in the following fields on the Administrative Setup screen:

   - In the **Remote:** field, set the communications parameters. For example: **9600,8-1-None** sets 9600 baud, 8 data bits, 1 stop bit, and no parity.
   - In the **SCPI Via:** field, select **REMOTE-RS232**.
   - Continue to step 6.

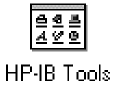
6. Now you are ready to make sure that the connection between the devices is operating properly. Continue to *Testing Your Connection* on page 2–8.

Testing the Connection

After connecting the controller to the test set and configuring the test set interface, follow this procedure to verify the connection between the devices is working properly.

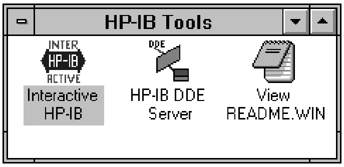
Note: *This procedure assumes you are using HP's Windows tools for HP-IB. Make sure the interface software is properly installed on you controller (refer to the software documentation). If you are using an interface from another vendor, refer to their documentation.*

1. Open the HP-IB Tools window.



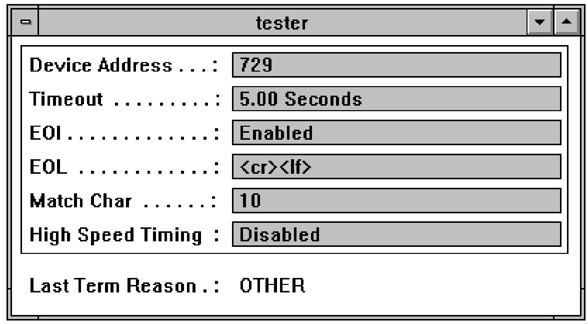
If you are using a serial connection, start the terminal emulation package on the controller. If Windows is installed on the controller, you can use the Terminal item in the Accessories window. See the documentation that came with the emulation package.

2. To launch the HP-IB interface software, click the Interactive HP-IB icon.

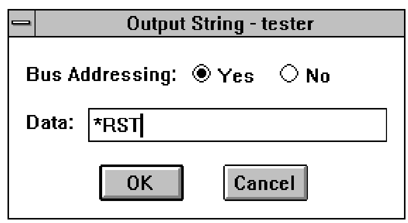


3. Select the **Add Device** option from the Setup menu.
4. When the Add Device window appears, enter a device name and click OK.

5. When the Setup Device Address screen appears, enter the GPIB Address that you assigned to the instrument (see *Using an GPIB Interface*, page 2–3) in the Primary Address field and click OK. A window similar to the following is displayed:



6. To enter a SCPI command, select **String…** from the Output menu.



7. To activate SCPI mode on the test set, type ***RST** or **:SYST:REM** in the Data field and click OK.
   - The *RST command puts the test set is stop mode (if necessary), resets all configuration parameters to their default values, and activates the SCPI interface. See **RST*, page 5–2.
   - The :SYST:REM command puts test set in stop mode (if necessary) and activates the SCPI interface. However, the test set's configuration is not altered. See *SYSTem:REMote*, page 14–3. To use the SCPI, the test set must be in a front panel operating mode.

When the set is in SCPI mode, the instrument's display shows the following:

```
 1 GPIB Remote Mode Final:  00:00:00.00
Last/Current Test Status #2
Pwr Loss Sec:      --
Last Pwr On:       --
Last Pwr Off:      --
Date & Time:       --/--/--  --:--:--
Enable Display of Results and Alarms

     SONET>Off               ATM>Off
       DS3>Off      PLCP>Off
 T-Carrier>Off               VT1.5>Off
 Results Level>Detail    Trouble Scan>Off
```

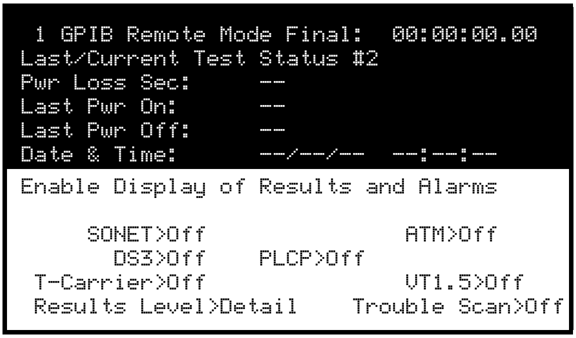
Now you can begin entering SCPI commands from the controller. See Chapter 3 for an introduction to entering SCPI commands, or refer Chapters 5 through 14 if you are familiar with using SCPI commands.

For information controlling the test set's display, see *Local Test Set Control in SCPI Mode*, page 2–11.

8. To exit from GPIB mode, type **:SYST:PRES** in the Data field and click OK.

Local Test Set Control in SCPI Mode

While the test set is in SCPI mode only a few front-panel controls are operational. You can, however, view measurements and choose what information is shown on the display.



Note: *When the test set is in SCPI mode, most front-panel controls are inoperative. Setup operations must be done using SCPI commands from the controller.*

- Use the FIELD and VALUE keys to configure which results are shown on the test set display and front-panel LEDs while the unit is in SCPI mode.
- Use FIELD to select Results Level and then use VALUE to set the results level for the level of measurement detail you want.
- Use the RESULT keys to scroll through measurement screens in the upper half of the display. The measurement screens available depend on the results enabled in the lower half of the display.
- For more information on viewing measurements, refer to the documentation that came with your test set.

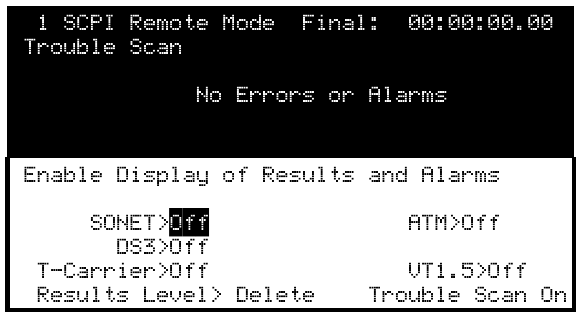
Local Mode - Store/Recall Command Feature

SCPI mode has “new” Store/Recall feature enhancement. Release 7.0 and higher software provides support for test set Store and Recall functions. In SCPI mode, the test set only has a limited number of operational front panel controls. Release 7.0 and higher software syntax requires the use of the leading colon “**:**” where earlier releases do not use it in all instances. Table below lists the various Local mode commands used with the 156 MTS test set.

SCPI Command	Context	Command Description
***RST**	SCPI	*Activates SCPI mode, and sets all parameter to their defaults. Also* See *SYSTem:REMote*, page 14–3
:SYSTEM:ERR?	Local	Query test set for local SCPI mode status.
:SYSTEM:STORE:1[:name]	Local	Store Configuration to position 1.
:SYSTEM:REMote	Local	Enter SCPI control, leave test set mode unchanged.
:SYSTEM: SCPI:IEEE488	Local	Commands SCPI interface connection with IEEE488 connection port.

The test set configurations are saved when you execute the SCPI command :**SYSTEM:STORE:n[:name]**. The “n” is a number location (between 1 and 10) that specifies the save slot of the configuration data. Using the “name” (optional parameter) specifies the name tag under which the test set stores the configuration data. This replaces the default configuration name (the mode title name). Note that to store test set modes, the unit must be operating in the desired mode via local (front panel) mode**.** Prior to entering SCPI, it is possible to query the status of the test set with :**SYSTEM:ERR**. The test set will return to the a textual string indicating the status of the local “SCPI” mode

When using version 7.0 released software, to recall test set configuration, you had to place a test set in non-menu mode prior to entering SCPI mode. The reason being that the test set cannot exit SCPI mode to enter terminal mode unless the test set was placed in terminal mode prior to entering SCPI control. As of Host SW V7.0, the **:SYSTEM:REMOTE** command will be rejected if the test set is in a menu. Enter SCPI control with **:SYST:REM**.



Enter a few selected System Commands in Local SCPI Mode using the store and recall commands. The command syntax , (listed below) require use of the complete (upper-case) mnemonic and not the abbreviated version that is used in SCPI mode. Table lists the Store/Recall command set.

Local Mode STORE/RECALL Command Table

SCPI Command	Context	Command Description
:SYSTEM:ERR?	Local	Query test set for local SCPI mode status.
:SYSTEM:STORE:1[:name]	Local	Store Configuration to position 1.
:SYSTEM:REM	Local	Enter SCPI control, leave test set mode unchanged.
:SENS:AU:STORE?	SCPI	Query the position number last stored.

Local Mode STORE/RECALL Command Table

SCPI Command	Context	Command Description
:SYSTEM: STORE? n	SCPI	Query name of configuration stored in position “n”.
:SYSTEM:CLEAR n	SCPI	Erase configuration stored in position “n”.
:SYSTEM:RECALL n	SCPI	Exit SCPI mode, bring test set to stored configuration “n”.

3

Programming A Test

Planning the Test

Before you begin, carefully consider your application. What rates will you be testing? What subrates will you be testing? Which channels will mapped to and from the multiplexers and demultiplexers? It may help to make notes about your desired configuration before programming the test set.

Command Sequence

As you set up your program, remember that the minimum command execution time is 250 milliseconds (one-quarter second).

The sequence in which you program the test set can affect the outcome of the test. Follow this general sequence for best results:

1. Program the highest signal rates first, and then work your way down through the lower rates.

   For example if you are testing DS1 signals carried on an STS-1, first configure the STS-1 signal, do the DS3 signal next, and finally configure the DS1 signal.

2. Configure the matrix switch first using the :ROUTe:MATRix commands.

3. Configure the signal path (higher rates first) using :ROUTe:SELect commands.

4. Select the channels to demultiplex using the :ROUTe:DMUX commands.

5. Select the channels to be inserted using the :ROUTe:MUX commands.

6. For DS1 or E1 signals, set the test mode using the :SOURce:DM:MODe command. You must use this command before setting the DS1 or E1 signal parameters.

7. Set the other signal parameters as follows:

   - Set the line code using :SOURce:DM:CODE*n*
   - Set the framing format using :SOURce:DM:FRAMe:TYPE*n*
   - Set the timing source using :SOURce:DM:CLOCKn.

8. Configure the next lower rate signal using steps 3 through 7.

9. When you have finished configuring your test set, begin the test using the :INIT command. See *Running a Test*, page 3–4.

Setting Up a Test

1. Connect input and output signals to your test set as appropriate for your application.
2. Put the set in SCPI mode using the *RST or :SYST:REM command (see below).

First make sure your test set and controller are properly connected and configured. See *Getting Started*, page 2–1.

Using the Reset Command

Each SCPI command has a reset (default) value for each of its parameters. The default values reflect the highest transmit and receive rates possible for the options purchased with your instrument.

The SCPI Reset command (*RST) performs the following actions:

- Places a test set into HP-IB mode.
- Sets all parameters to a defined default setting.
- Sets each instrument to a state where it is waiting for a configuration or a measurement command.

When you become familiar with the *RST default settings you will notice that, in many cases, you can enter just a few commands to perform a test or make a measurement.

After you become familiar with SCPI commands, see *Front Panel Emulation Using SCPI*, page 4–1, which lists the SCPI commands you use after a reset command to duplicate any selectable front panel setups.

Using the System Remote Command

The :SYST:REM command places the test set in SCPI mode without altering its current configuration. You can configure the test set using the front panel controls and then use :SYST:REM to put the unit in SCPI mode. Next you must place the test set in test mode, run the test, and retrieve measurements over the SCPI connection.

Running a Test

This section describes a sample SCPI test. The example shows the sequence of SCPI commands to enter if you want to transmit a test signal at OC3 and OC12 with all the STS-1s containing DS3 patterns, and then receive and measure the DS3 which is dropped from the SONET signal by the multiplexer under test.

Each section of the test has a brief overview and an explanation for each SCPI command.

Starting the Test

If you want to gather measurements or test results in the status registers, you should first issue the SCPI commands that set up the appropriate signal paths (see *Command Sequence*, page 3–2). Then you use the :INITiate command to start running the test.

Starting a Test

SCPI Command	Description
**RST*	*Access SCPI mode and set the parameters to their default settings.*
:ROUT:SEL DS3_TX,SOURCE	*Selects the internal DS3 pattern generator for the DS3 transmit payloads.*
:ROUT:SEL DS1_RX,BIPOLAR	*Directs the DS1 bipolar receive port to the DS1 measurement circuits.*
:INIT	*Starts running the test. Turns on the green RUN LED on the test set.*

Note: *Remember that the minimum execution time for each SCPI command is 250 milliseconds (one quarter second).*

Requesting Signal Status

After you start a test, you can request the accumulated signal status from the various SCPI status registers.

Requesting Signal Status

SCPI Command	Description
:FETC:DMOD:DS3? SIGNAL	*Request the DS3 signal status.*
:FETC:DMOD:DS3? M13FRAME	*Request DS3 M13 frame detect status.*
:FETC:DMOD:DS3? PATTERN	*Request DS3 pattern sync status.*

If you receive a response of +1 for each of the status indicators, the receive signal is present and the dropped DS3 signal has the expected framing type. A response of +0 for the STS-1 Loss Of Pointer alarm indicates that the STS-1 pointer is valid (no alarm condition).

Injecting Errors

Now you can inject some errors in the signal and request the error count from the status registers.

Injecting Errors and Requesting Error Status

SCPI Command	Description
:SOUR:DM:EINJ:DS3 DATA,1E-3	*Injects DS3 errors into the DS3 signal.*
(Wait a few moments)	*Wait for errors to accumulate. This will give you more significant measurements.*
:FETC:DMOD:DS3? BIT	*Requests the DS3 bit error count.*
:FETC:DMOD:DS3? BIT_ARATIO	*Requests the DS3 bit error average ratio.*
:FETC:DMOD:DS3? BIT_CRATIO	*Requests the DS3 bit error current ratio.*
:FETC:DMOD:DS3? BIT_SES	*Requests the DS3 severely errored seconds.*

Stopping the Test

When you have gathered all your measurements and test results, you can stop the test and return control to the test set.

Stopping the Test

SCPI Command	Description
:ABOR	*Stops the test. You will notice the green RUN LED turn off.*
:SYST:PRES	*Takes the test set out of SCPI mode and returns control to the test set.*

Returning Control to the Test Set

To exit SCPI mode and return control back to the test instrument, follow these steps:

1. On the controller, issue the :SYST:PRES command. Or you can turn the test instrument off and then on.
2. The main menu appears on the test instrument when it is out of SCPI mode.

```
          MODEL 156 MAIN MENU
Auto Setup
Terminal Testing
Monitor Testing
Drop & Insert Testing
DS3/DS1/ATM Scans & Pointer Sequences
Setup System Parameters
Store and Recall Configurations

________________________________________
Press FIELD to highlight item, then
Press MENU-Dn to select item.
```

Timed Test Duration in SCPI Mode

Release 7.0 Software and higher supports time limited test duration setting under SCPI mode. Use of this feature is accomplished using the commands listed in the following table:

Timed Test Duration Command Table

SCPI Command	Command Description
***RST**	Access SCPI mode and set parameters to default settings.
:SENS:AU:TESTDURMODE TIMER	Put the test set in" Timed Test" duration mode.
:SENS:AU:TIMERDURHOURS 1	Set test duration for 1 hour,
:SENS:AU:TTIMERDURMINS 22	22 minutes, and
:SENS:AU:TTIMERDURSECS 15	15 seconds.
:INIT	Start running the test. Turn on run indicators on test set

Path- and Section-Trace Strings

Host version 6.60 lets you display and edit the J1 and J2 path-trace strings, and the J0 section-trace string. New fields have been added to the J1 & J2 Path Trace Control and J0 Section Trace Control screens.

- The **Fill on ACTION** (or **Fill on INJECT**) and **Format** fields define the transmit trace string to transmit when you press the ACTION (INJECT) button (see the table below).
- **Results Display in** sets the display mode for the path-trace byte and section-trace byte values (below). This field has two options:
  - ASCII (the default)–displays values in ASCII.
  - Hex & ASCII–displays values in hex and ASCII.

Fill on Action	Format - for Fill on Action (inject button) Selections		
	1-Byte	**16-Byte**	**64-Byte**
NULL	*0x00 in all bytes.*	*0x00 in all bytes.*	*0x00 in all bytes.*
HEX	*0x01 in all bytes.*	*0x41 – 0x4F with CRC7, copied 4 times.*	*0x41 to 0x7F (ASCII A to DEL), followed by a carriage return (0x0D) & line feed (0x0A).*
ASCII	*0x41 in all bytes.*	*“nnnnnn” serial number with CRC7, copied 4 times.*	*“Agilent Technologies 156MTS Test Set Serial No. nnnnnn” followed by carriage return (0x0D) and line feed (0x0A).*
USER	*First byte copied to all 64 bytes.*	*Calculate and insert CRC7 of first 16 bytes; copy first 16 bytes 4 times.*	*No action.*

4

Front PanelEmulation Using SCPI

Recreating Front Panel Setups

Although the SCPI command set provides more robust, in-depth control of the CERJAC test set for the programmer, you can use the SCPI command set to duplicate any setup configured using the instrument's front panel controls.

The following sections list some selected front-panel test setups, and the corresponding SCPI commands you can use to emulate them.

Each example begins with the *RST command to reset the test set to its factory defaults. From this state, the examples will configure the test set for the indicated setup. You may need to do further programming to meet the specific needs of your application. As you program, be sure to follow the proper command sequence — see *Command Sequence,* page 3–2.

Terminal Testing Modes

SCPI Commands to Emulate Terminal Testing Modes

Transmit Rate	Receive Rate	Payload	SCPI Command Sequence
OC-12	*OC-12*	*STS-12c*	**RST* *:ROUT:SEL STS12C,CLRCH* *:ROUT:SEL STS12PYLD,STS12C*
OC-12	*OC-12*	*STS-3c*	**RST* *:ROUT:MATR STS3C,CLRCH*
OC-12	*OC-12*	*DS3*	**RST* *:ROUT:SEL DS3_TX,SOURCE*
OC-12	*OC-12*	*DS3/1*	**RST*
OC-12	*OC-12*	*VT/DS1*	**RST* *:ROUT:MATR STS#3,VT15* *:ROUT:MATR STS#2,VT15* *:ROUT:MATR STS#1,VT15* *:ROUT:MATR DS3DROP,STS#1* *:ROUT:MATR VT15DROP,STS#1* *:ROUT:SEL DS1_RX,VT15*
OC-12	*OC-12*	*STS-1/ATM*	**RST* *:ROUT:SEL STS1_TX,ATM* *:SOUR:DM: ATMMUX STS1*
OC-12	*OC-12*	*STS-12c w/ATM*	**RST* *:ROUT:SEL STS12C,ATM* *:ROUT:SEL STS12PYLD,STS12C* *:SENS:DM:ATMDMUX STS12C*
OC-12	*OC-12*	*STS-3c w/ATM*	**RST* *:ROUT:MATR STS3C,ATM* *:SOUR:DM:ATMMUX STS3C*

SCPI Commands to Emulate Terminal Testing Modes, continued

Transmit Rate	Receive Rate	Payload	SCPI Command Sequence
OC-12	*OC-3*	*DS3*	**RST* *:ROUT:SEL DS3_TX,SOURCE* *:ROUT:SEL STS3_RX,OC3*
OC-12	*STS-1*	*DS3*	**RST* *:ROUT:SEL DS3_TX,SOURCE* *:ROUT:MATR STS1DROP,EXT* *:ROUT:MATR DS3DROP,EXT*
OC-12	*DS3*	*DS3*	**RST* *:ROUT:SEL DS3_TX,SOURCE* *:ROUT:SEL DS3_RX,BIPOLAR*
OC-12	*DS1*	*DS3/1*	**RST* *:ROUT:SEL DS1_RX,BIPOLAR* *:ROUT:SEL DS3_RX,BIPOLAR*
OC-12	*DS1*	*VT/DS1*	**RST* *:ROUT:MATR STS#1,VT15* *:ROUT:MATR STS#2,VT15* *:ROUT:MATR STS#3,VT15* *:ROUT:MATR DS3DROP,STS#1* *:ROUT:SEL DS1_RX,BIPOLAR*
OC-3	*OC-12*	*DS3*	**RST* *:ROUT:SEL DS3_TX,SOURCE* *:ROUT:SEL OPT_TX,OC3*
OC-3	*OC-3*	*STS-1/ATM*	**RST* *:ROUT:SEL OPT_TX,OC3* *:ROUT:SEL STS1_TX,ATM* *:SOUR:DM:ATMMUX STS1* *:ROUT:SEL STS3_RX,OC3*

SCPI Commands to Emulate Terminal Testing Modes, continued

Transmit Rate	Receive Rate	Payload	SCPI Command Sequence
OC-3	*OC-3*	*STS-3c*	**RST :ROUT:SEL OPT_TX,OC3 :ROUT:SEL STS3_RX,OC3 :ROUT:MATR STS3C,CLRCH*
OC-3	*OC-3*	*DS3*	**RST :ROUT:SEL DS3_TX,SOURCE :ROUT:SEL OPT_TX,OC3 :ROUT:SEL STS3_RX,OC3*
OC-3	*OC-3*	*DS3/1*	**RST :ROUT:SEL OPT_TX,OC3 :ROUT:SEL STS3_RX,OC3*
OC-3	*OC-3*	*VT/DS1*	**RST :ROUT:SEL OPT_TX,OC3 :ROUT:SEL STS3_RX,OC3 :ROUT:MATR STS#1,VT15 :ROUT:MATR STS#2,VT15 :ROUT:MATR STS#3,VT15 :ROUT:MATR DS3DROP,STS#1 :ROUT:MATR VT15DROP,STS#1 :ROUT:SEL DS1_RX,VT15*
OC-3	*OC-3*	*ATM*	**RST :ROUT:SEL OPT_TX,OC3 :ROUT:SEL STS3_RX,OC3 :ROUT:MATR STS3C,ATM :SOUR:DM:ATMMUX STS3C*

SCPI Commands to Emulate Terminal Testing Modes, continued

Transmit Rate	Receive Rate	Payload	SCPI Command Sequence
OC-3	*STS-1*	*DS3*	**RST* *:ROUT:SEL DS3_TX,SOURCE* *:ROUT:SEL OPT_TX,OC3* *:ROUT:MATR STS1DROP,EXT* *:ROUT:MATR DS3DROP,EXT*
OC-3	*DS3*	*DS3*	**RST* *:ROUT:SEL DS3_TX,SOURCE* *:ROUT:SEL OPT_TX,OC3* *:ROUT:SEL DS3_RX,BIPOLAR*
OC-3	*DS3*	*VT-DS3/1*	**RST* *:ROUT:MATR STS#1,VT15* *:ROUT:MATR STS#2,VT15* *:ROUT:MATR STS#3,VT15* *:ROUT:MATR DS3DROP,STS#1* *:ROUT:MATR VTDROP,EXT* *:ROUT:SEL OPT_TX,OC3* *:ROUT:SEL DS3_RX,BIPOLAR*
OC-3	*DS1*	*DS3/1*	**RST* *:ROUT:SEL OPT_TX,OC3* *:ROUT:SEL DS1_RX,BIPOLAR*
OC-1	*OC-12*	*DS3*	**RST* *:ROUT:SEL OPT_TX, OC1* *:ROUT:SEL DS3_TX, SOURCE*
OC-1	*OC-3*	*OC-3*	**RST* *:ROUT:SEL OPT_TX, OC1* *:ROUT:SEL DS3_TX, SOURCE* *:ROUT:SEL STS3_RX, OC3*

SCPI Commands to Emulate Terminal Testing Modes, continued

Transmit Rate	Receive Rate	Payload	SCPI Command Sequence
OC-1	*OC-1*	*DS3*	**RST* *:ROUT:SEL OPT_TX, OC1* *:ROUT:SEL DS3_TX,SOURCE* *:ROUT:MATR STS1DROP, EXT* *:ROUT:MATR DS3DROP, RXT* *:ROUT:SEL STS1_RX, OC1*
OC-1	*OC-1*	*DS3/DS1*	**RST* *:ROUT:SEL OPT_TX, OC1* *:ROUT:MATR STS1DROP, EXT* *:ROUT:MATR DS3DROP, EXT* *:ROUT:SEL STS1_RX, OC1* *:SENS:DM::DATA3BITERR_OFF*
OC-1	*OC1*	*DS3/E1*	**RST* *ROUT:SEL OPT_TX, OC1* *:SOUS:DM:MODE E1* *:ROUT:MATR STS1DROP, EXT* *:ROUT:MATR DS3DROP, EXT* *:ROUT:SEL STS3_RX,OC1* *:SENS:DM::DATA3BITERR_OFF*
OC-1	*OC-1*	*VT/DS1*	**RST* *:ROUT:SEL OPT_TX, OC1* *:ROUT:MATR STS#, VT15* *:ROUT:MATR DS3DROP, EXT* *:ROUT:MATR DS3DROP, EXT* *:ROUT:MATR VT15DROP, EXT* *:ROUT:SEL STS1_RX, OC1* *:ROUT:SEL DS1_RX,VT15*

SCPI Commands to Emulate Terminal Testing Modes, continued

Transmit Rate	Receive Rate	Payload	SCPI Command Sequence
OC-1	*OC-1*	*ATM*	**RST :ROUT:SEL OPT_TX, OC1 :ROUT:MATR STS1DROP,EX :ROUT:MATR VT15DROP, EXT :ROUT:MATR DS3DROP, EXT :ROUT:SEL STS1_TX, ATM :SOUR:DM:ATMMUX STS1 :ROUT:SEL STS1_RX, OC1*
OC-1	*STS-1*	*DS3*	**RST :ROUT:SEL OPT_TX, OC1 :ROUT:MATR STS1DROP,EXT :ROUT:MATR VT15DROP, EXT :ROUT:SEL STS1_RX, BIPOLAR*
OC-1	*STS-1*	*DS3/DS1*	**RST :ROUT:SEL OPT_TX, OC1 :ROUT:MATR STS1DROP,EXT :ROUT:MATR DS3DROP,EXT :ROUT:SEL STS1_RX, BIPOLAR :SENS:DM::DATA3BITERR_OFF*
OC-1	*STS-1*	*DS3/E1*	**RST :ROUT:SEL OPT_TX, OC1 :SOUR:DM:MODE E1 :ROUT:MATR STS1DROP, EXT :ROUT:MATR DS3DROP, EXT :ROUT:SEL STS1_RX, BIPOLAR :SENS:DM::DATA3BITERR_OFF*

SCPI Commands to Emulate Terminal Testing Modes, continued

Transmit Rate	Receive Rate	Payload	SCPI Command Sequence
OC-1	*STS1*	*VT/DS1*	**RST* *:ROUT:SEL OPT_TX, OC1* *:ROUT:MATR STS1#1, VT15* *:ROUT:MATR STS1DROP, EXT* *:ROUT:MATR DS3DROP, EXT* *:ROUT:MATR VT15DROP, EXT* *:ROUT:SEL DS3_RX, BIPOLAR* *:ROUT:SEL DS1_RX, VT15*
OC-1	*STS-1*	*ATM*	**RST* *:ROUT:SEL OPT_TX, OC1* *:ROUT:MATR STS1DROP, EXT* *:ROUT:MATR V15DROP, EXT* *:ROUT:MATR DS3DROP, EXT* *:ROUT:SEL STS1_TX, ATM* *:SOUR:DM:ATMMUX STS1* *:ROUT:SEL DS3_RX, BIPOLAR*
OC-1	*STS-1*	*DS3*	**RST* *:ROUT:SEL OPT_TX, OC1* *:ROUT:SEL DS3_TX, SOURCE* *:ROUT:MATR STS1DROP, EXT* *:ROUT:MATR DS3DROP, EXT* *:ROUT:MATR DS3_RX, BIPOLAR*
STS-1	*OC12*	*DS3*	**RST* *:ROUT:MATR DS3_TX,SOURCE*
STS-1	*OC-3*	*DS3*	**RST* *:ROUT:MATR DS3_TX,SOURCE* *:ROUT:SEL STS3_RX, OC3*

SCPI Commands to Emulate Terminal Testing Modes, continued

Transmit Rate	Receive Rate	Payload	SCPI Command Sequence
STS-1	*STS-1*	*DS3*	**RST :ROUT:MATR DS3_TX,SOURCE :POUT:MATR STS1DROP, EXT :ROUT:MATR DS3DROP, EXT*
STS-1	*OC-1*	*DS3*	**RST :ROUT:SEL DS3_TX,SOURCE :ROUT:MATR STS1DROP,EXT :ROUT:MATR DS3DROP,EXT :ROUT:SEL STS1_RX, OC1*
STS-1	*STS-1*	*DS3/1*	**RST :ROUT:MATR STS1DROP,EXT :ROUT:MATR DS3DROP,EXT :SENS:DM::DATA3BITERR_OFF*
STS-1	*STS-1*	*VT/DS1*	**RST :ROUT:MATR STS#1,VT15 :ROUT:MATR STS1DROP,EXT :ROUT:MATR DS3DROP,EXT :ROUT:MATR VT15DROP,EXT :ROUT:SEL DS1_RX,VT15*
STS-1	*STS-1*	*ATM*	**RST :ROUT:MATR STS1DROP,EXT :ROUT:MATR VT15DROP,EXT :ROUT:MATR DS3DROP,EXT :ROUT:SEL STS1_TX,ATM :ROUT:SEL STS1_RX,BIPOLAR :SOUR:DM:ATMMUX STS1*

SCPI Commands to Emulate Terminal Testing Modes, continued

Transmit Rate	Receive Rate	Payload	SCPI Command Sequence
STS-1	*DS3*	*DS3*	**RST* *:ROUT:SEL DS3_TX,SOURCE* *:ROUT:MATR STS1DROP,EXT* *:ROUT:MATR DS3DROP,EXT*
STS-1	*DS1*	*DS3/1*	**RST* *:ROUT:SEL DS1_RX,BIPOLAR*
STS-1	*DS1*	*VT/DS1*	**RST* *:ROUT:MATR STS#1,VT15* *:ROUT:MATR DS3DROP,STS#1* *:ROUT:SEL DS1_RX,BIPOLAR*
DS3	*OC-12*	*DS3*	**RST* *:ROUT:SEL DS3_TX,SOURCE*
DS3	*OC-12*	*DS3/1*	**RST*
DS3	*OC-3*	*DS3*	**RST* *:ROUT:SEL DS3_TX,SOURCE* *:ROUT:SEL STS3_RX,OC3*
DS3	*OC-3*	*DS3/1-VT*	**RST* *:ROUT:MATR DS3DROP,STS#1* *:ROUT:MATR VT15DROP,STS#1* *:ROUT:SEL DS1_RX,VT15*
DS3	*STS-1*	*DS3*	**RST* *:ROUT:SEL DS3_TX,SOURCE* *:ROUT:MATR STS1DROP,EXT* *:ROUT:MATR DS3DROP,EXT*
DS3	*DS3*	*DS3*	**RST* *:ROUT:SEL DS3_TX,SOURCE* *:ROUT:SEL DS3_RX,BIPOLAR*

SCPI Commands to Emulate Terminal Testing Modes, continued

Transmit Rate	Receive Rate	Payload	SCPI Command Sequence
DS3	*DS3*	*DS1*	**RST* *:ROUT:SEL DS3_RX,BIPOLAR* *:SENS:DM::DATA3BITERR_OFF*
DS3	*DS3*	*E1*	**RST* *:SOUR:DM:MODE E1* *:ROUT:SEL DS3_RX_BIPOLAR* *:SENS:DM::DATA3BITERR_OFF*
DS3	*DS3*	*ATM (PLCP)*	**RST* *:ROUT:SEL DS3_TX,ATM* *:ROUT:SEL DS3_RX,BIPOLAR* *:SOUR:DM:ATMMUX DS3*
DS3	*DS3*	*ATM (HEC)*	**RST* *:ROUT:SEL DS3_TX,ATM* *:ROUT:SEL DS3_RX,BIPOLAR* *:SOUR:DM:ATMMUX DS3_HEC*
DS3	*DS1*	*DS1*	**RST* *:ROUT:SEL DS1_RX,BIPOLAR*
DS1	*OC-12*	*DS3/1*	**RST*
DS1	*OC-12*	*VT/DS1*	**RST* *:ROUT:MATR DS3DROP,STS#1* *:ROUT:MATR VT15DROP,STS#1* *:ROUT:SEL STS3_RX,VT15*
DS1	*OC-3*	*DS3/1*	**RST* *:ROUT:SEL STS3_RX,OC3*

SCPI Commands to Emulate Terminal Testing Modes, continued

Transmit Rate	Receive Rate	Payload	SCPI Command Sequence
DS1	*OC-3*	*VT/DS1*	**RST* *:ROUT:SEL STS3_RX,OC3* *:ROUT:MATR DS3DROP,STS#1* *:ROUT:MATR VT15DROP,STS#1* *:ROUT:SEL DS1_RX,VT15*
DS1	*STS-1*	*VT/DS1*	**RST* *:ROUT:MATR STS1DROP,EXT* *:ROUT:MATR DS3DROP,STS#1* *:ROUT:MATR VT15DROP,EXT* *:ROUT:SEL DS1_RX,VT15*
DS1	*DS3*	*DS1*	**RST* *:ROUT:SEL DS3_RX,BIPOLAR*
DS1	*DS1*	*DS1*	**RST* *:ROUT:SEL DS1_RX,BIPOLAR*
DS1	*DS1*	*ATM*	**RST* *:ROUT:SEL DS1_TX,ATM* *:ROUT:SEL DS1_RX,BIPOLAR* *:SOUR:DM:ATMMUX DS1*
E1	*E1*	*E1*	**RST* *:SOUR:DM:MODE E1* *:ROUT:SEL E1_RX,BIPOLAR*
E1	*E1*	*ATM*	**RST* *:SOUR:DM:MODE E1* *:ROUT:SEL E1_TX,ATM* *:ROUT:SEL E1_RX,BIPOLAR* *:SOUR:DM:ATMMUX E1*

Monitor Testing Modes

SCPI Commands to Emulate Monitor Testing Modes

Tx/Rx Rate	Payload	SCPI Command Sequence
OC-12	*DS3*	**RST* *:ROUT:SEL STS12_TX,PASS*
OC-12	*DS3/1*	**RST* *:ROUT:SEL STS12_TX,PASS*
OC-12	*VT/DS1*	**RST* *:ROUT:MATR VT15DROP,STS#1* *:ROUT:MATR DS3DROP,STS#1* *:ROUT:SEL DS1_RX,VT15* *:ROUT:SEL STS12_TX,PASS*
OC-3	*DS3*	**RST* *:ROUT:SEL STS3_RX,OC3* *:ROUT:SEL STS3_TX,PASS* *:ROUT:SEL OPT_TX,OC3*
OC-3	*DS3/1*	**RST* *:ROUT:SEL STS3_RX,OC3* *:ROUT:SEL STS3_TX,PASS* *:ROUT:SEL OPT_TX,OC3*
OC-3	*VT/DS1*	**RST* *:ROUT:SEL STS3_RX,OC3* *:ROUT:MATR DS3DROP,STS#1* *:ROUT:MATR VT15DROP,STS#1* *:ROUT:SEL DS1_RX,VT15* *:ROUT:SEL STS3_TX,PASS* *:ROUT:SEL OPT_TX,OC3*

SCPI Commands to Emulate Monitor Testing Modes, continued

Tx/Rx Rate	Payload	SCPI Command Sequence
STS-1	*DS3*	**RST* *:ROUT:MATR STS#1,EXT* *:ROUT:MATR STS1DROP,EXT* *:ROUT:MATR DS3DROP,EXT*
STS-1	*DS3/1*	**RST* *:ROUT:MATR STS#1,EXT* *:ROUT:MATR STS1DROP,EXT* *:ROUT:MATR DS3DROP,EXT*
STS-1	*VT/DS1*	**RST* *:ROUT:MATR STS#1,EXT* *:ROUT:MATR STS1DROP,EXT* *:ROUT:MATR DS3DROP,EXT* *:ROUT:MATR VT15DROP,EXT* *:ROUT:SEL DS1_RX,VT15*
DS3	*DS3*	**RST* *:ROUT:SEL DS3_RX,BIPOLAR* *:ROUT:SEL DS3_TX,LOOP*
DS3	*DS1*	**RST* *:ROUT:SEL DS3_RX,BIPOLAR* *:ROUT:SEL DS3_TX,LOOP*
DS1	*DS1*	**RST* *:ROUT:SEL DS1_RX,BIPOLAR* *:ROUT:SEL DS1_TX,LOOP*

Drop and Insert Testing Modes

SCPI Commands to Emulate Drop and Insert Testing Modes

Tx/Rx Rate	Payload	SCPI Command Sequence
OC-12	*STS-1*	**RST* *:ROUT:MATR STS#1,EXT* *:ROUT:MATR STS#2,STS#2* *:ROUT:MATR STS#3,STS#3* *:ROUT:MUX:OTH STS12,PASS*
OC-12	*DS3*	**RST* *:ROUT:SEL DS3_TX,SOURCE* *:ROUT:MATR STS#2,STS#2* *:ROUT:MATR STS#3,STS#3* *:ROUT:MUX:OTH STS12,PASS*
OC-12	*DS3/1*	**RST* *:ROUT:SEL DS3_TX,SOURCE* *:ROUT:MATR STS#2,STS#2* *:ROUT:MATR STS#3,STS#3* *:ROUT:MUX:OTH STS12,PASS*
OC-3	*STS-1*	**RST* *:ROUT:SEL STS3_RX,OC3* *:ROUT:MATR STS#1,EXT* *:ROUT:MATR STS#2,STS#2* *:ROUT:MATR STS#3,STS#3* *:ROUT:SEL OPT_TX,OC3*

SCPI Commands to Emulate Drop and Insert Testing Modes, continued

Tx/Rx Rate	Payload	SCPI Command Sequence
OC-3	*DS3*	**RST* *:ROUT:SEL STS3_RX,OC3* *:ROUT:SEL DS3_TX,SOURCE* *:ROUT:MATR STS#2,STS#2* *:ROUT:MATR STS#3,STS#3* *:ROUT:SEL OPT_TX,OC3*
OC-3	*DS3/1*	**RST* *:ROUT:SEL STS3_RX,OC3* *:ROUT:SEL DS3_TX,DS31DI* *:ROUT:MATR STS#2,STS#2* *:ROUT:MATR STS#3,STS#3* *:ROUT:SEL OPT_TX,OC3*
STS-1	*DS3/1*	**RST* *:ROUT:MATR STS1DROP,EXT* *:ROUT:MATR DS3DROP,EXT* *:ROUT:SEL DS3_TX,DS31DI*
DS3	*DS1*	**RST* *:ROUT:SEL DS3_RX,BIPOLAR* *:ROUT:SEL DS3_TX,DS31DI*

5

Common SCPI Commands Reference

Test Set Commands

Common SCPI "Test Set" Commands

Command	Description
**IDN?*	*Returns the Model and Serial numbers of the test set, along with the software version number and release date.*
**RST*	*Activates SCPI mode, and sets all parameter to their defaults. See also SYSTem:REMote, page 14–3.*
**TST?*	*The response to *TST? (Self Test Query) is 0, which means "OK".*
**WAI*	*The response to *WAI is NO-OP (no operation) since there are no "over-lapped" commands.*

Register Commands

Common SCPI Register Commands

Command	Description
**CLS*	*Clears the Standard Event Status Register to all 0’s.*
**ESE <x>*	*Sets the Standard Event Status Enable Register bits, where x is a decimal integer whose binary equivalent is the desired 8 bit mask.*
**ESE?*	*Returns the decimal integer value of the standard event status enable register.*
**ESR?*	*Returns the decimal integer value of the Standard Event Status Register. Reading the Standard Event Status Register clears it.*
**OPC*	*After receipt of the *OPC command, Bit 0 of the Standard Event Status Register is set to confirm that all processing of commands received* **before** *receipt of the *OPC have been completed (and results, where appropriate, are in the output queue.)*
**OPC?*	*After receipt of the OPC? command, an ASCII character “1” is put into the output queue to confirm that all processing of commands received* **before** *receipt of the *OPC? have been completed (and results, where appropriate, are in the output queue.)*
**SRE*	*Set the bits in the Service Request Enable Register. Syntax is *SRE <x> where x is a decimal integer whose binary equivalent is the desired 8 bit mask. Note, however, that Bit 6 is set to 0, regardless of the *SRE command. This bit is not programmable.*
**SRE?*	*Returns the decimal value of the Service Request Enable Register. Reading the Register* **does not** *clear it. Note that the retrieved value will always be in the range 0 through 63 or 128 through 191 (Bit 6 = 0).*

Common SCPI Register Commands, continued

Command	Description
**STB?*	*Returns the decimal integer value of the Status Byte Register. Note that in this version of the register, Bit 6 is “MSS” rather than “RQS”.*

:FETCh Command Reference

:FETCh Command Reference

Use the :FETCh commands to retrieve test result values.

:FETCh Command Structure

Root node	Level 1 node	Level 2 node	Level 3 node	Parameter
:FETCh	*:TRBLSCAN?*			
	:DMOD	*:ATM?*		*<result>*
		:DS0?		*<result>*
		:DS1?		*<result>*
		:DS2?		*<result>*
		:DS3? | DS3B?		*<result>*
		:DS3	*:CBIT?*	*<row, cbit>*
		:DS3		*<FEAC code>*
		:E1?		*<result>*
		:VT15?		*<result>*
		:STS1?		*<result>*
			:PATH?	*<result>*
			:LINE?	*<result>*
			:SECTion?	*<result>*
		:STS3C?		*<result>*
		STS12C?		*<result>*

Example

:FETCh commands are used as follows:

```
:FETC:DMOD:STS1:PATH? CV_ES
```

- Retrieves the STS-1 Path code violations errored-seconds count.

Summary_1 Field Descriptions (Block Errors)

Response Fields

The :FETCH:DMOD:DS1? SUMMARY_1 and E1? SUMMARY_1 commands return a data record that consists of eight fields. The fields are as follows:

Field 1 — **Running bit error count:** The number of bit errors.

Field 2 — **Running block error count:** The number of block errors counted.

Field 3 — **Current loss of pattern alarm status:** Loss of pattern (LOP) alarm status (Boolean).

Field 4 — **Recent change of LOP alarm status:** Indicates whether the LOP alarm condition changed in the most recent 250 ms period (Boolean).

Field 5 — **Current DS1 loss of frame alarm status:** Loss of frame (LOF) alarm status (Boolean).

Field 6 — **Recent change of LOF alarm status:** Indicates whether the LOF alarm condition changed in the most recent 250 ms period (Boolean).

Field 7 — **Recent COFA occurrence:** Indicates whether a change of frame alignment (COFA) occurred in the most recent 250 ms period (Boolean).

Field 8 — **Cycle 250 ms counter:** Displays a count from 0 through 15 that is incremented every 250 ms. When the value reaches 15, it wraps back to 0. A value of –1 indicates that the counter is not valid.

Example Summary_1 Response

```
+33 +2 +0 +0 +0 +0 +0 +12
```

Where:

- `+33` 33 bit errors were detected.
- `+2` Two DS1 block errors were detected.
- `+0` The LOP alarm is off.
- `+0` The LOP alarm did not change status in the 250 ms.
- `+0` The LOF alarm is off.
- `+0` The LOF alarm did not change status in the 250 ms
- `+0` The COFA did not occur in the last 250 ms.
- `+12` The current value of the cyclic 250 ms counter.

:FETCh:TRBLSCAN?

This command retrieves Trouble Scan results from the test set. Note that Trouble Scan must first be activated using the :SENS:DM:DATA:TRBL On command
(see *:SENSe:DM:DATA:TRBLscan <ON|OFF>*, page 11–8).

Example

`:FETCh:TRBLSCAN?`

- Retrieves whatever results would have been shown on the test set Trouble Scan display. A typical example might look similar to the following:

```
DS3 LOS: HIS  DS3 OOF: HIS  DS3 BIT: 12
```

- indicating that the DS3 LOS and OOF alarms are in their history state and that 12 DS3 bit errors were detected.

Refer to your test set's manual for more information on how Trouble Scan works.

:FETCh:DMOD:ATM? <*result*>

This command retrieves ATM measurement results. Make sure you have set the ATM DMUX to the appropriate mapping before using this command (see *:SENSe:DM:ATMDMUX <mapping>*, page 11–3).

Selected VP/VC Results

The following table lists the selected cell stream measurements available for this command:

:FETCh:DMOD:ATM? <*selected VP/VC result*>

Parameter	Retrieved result
SEL_BW_HZ_AVG	*Selected VP/VC received bandwidth frequency (Hz) average.*
SEL_BW_PCT_AVG	*Selected VP/VC received bandwidth percentage average.*
SEL_BW_HZ_CUR	*Selected VP/VC received bandwidth frequency (Hz) current.*
SEL_BW_PCT_CUR	*Selected VP/VC received bandwidth percentage current.*
SEL_CELL_CNT	*Selected VP/VC cell count.*
SEL_CELL_SEC_AVG	*Selected VP/VC cells per second, average.*
SEL_CELL_SEC_CUR	*Selected VP/VC cells per second, current.*
SCNR	*Selected cell not received alarm.*
SCNR_SEC	*Selected cell not received alarm seconds.*

Total Cell Stream Results

The following table lists the total cell stream measurements available for this command.

:FETCh:DMOD:ATM? <*total cell stream result*>

Parameter	Retrieved result
TOT_BW_HZ_AVG	*Total active bandwidth frequency (Hz) average.*
TOT_BW_PCT_AVG	*Total active bandwidth percentage average.*
TOT_BW_HZ_CUR	*Total active bandwidth frequency (Hz) current.*
TOT_BW_PCT_CUR	*Total active bandwidth percentage current.*
HCS_ERR_CNT	*Total cell stream HEC error count.*
HCS_ERR_RAT	*Total cell stream HEC error/cell ratio.*
CHCS_ERR_CNT	*The total number of correctable HEC errors. Note this result is available for STS-12c ATM only.*
UHCS_ERR_CNT	*The total number of uncorrectable HEC errors. Note this result is available for STS-12c ATM only.*
TOT_CELL_CNT	*Total cell stream cell count.*
TOT_CELL_SEC_AVG	*Total cell stream cells per second, average.*
TOT_CELL_SEC_CUR	*Total cell stream cells per second, current.*

Payload Bit Error Results

The following table lists the ATM payload bit error measurements available using this command. Note that the ATM receive channel type must be set to AAL-0, AAL-1, or test cell (see *:SOURce:DM:ATMGEN <ATM param>,<setting>*, page 12–9).

:FETCh:DMOD:ATM? *<payload bit error result>*

Parameter	Retrieved result
BIT_ERR_CNT	*ATM payload bit error count.*
BIT_ERR_RAT_AVG	*ATM payload bit error ratio, average.*
BIT_ERR_RAT_CUR	*ATM payload bit error ratio, current.*
PYLD_PAT_SYNC	*ATM payload pattern synchronization status.*
PYLD_ES	*ATM payload bit error seconds.*
PYLD_EFS	*ATM payload bit percent errored free seconds.*
PYLD_LOP	*Payload loss of pattern alarm.*
PYLD_LOP_SEC	*Payload loss of pattern alarm seconds.*

ATM AAL-1 Results

The following parameters are used to retrieve ATM AAL-1 measurement results. Before you use these parameters, you must set the ATM foreground channel type to AAL-1 (see *:SOURce:DM:ATMGEN <ATM param>,<setting>*, page 12–9).

:FETCh:DMOD:ATM? <*AAL-1 result*>

Parameter	Retrieved result
CELL_LOSS_CNT	*AAL-1 cell loss count.*
CELL_LOSS_RAT_AVG	*AAL-1 cell loss ratio, average.*
CELL_LOSS_RAT_CUR	*AAL-1 cell loss ratio, current.*
CELL_LOSS_ES	*AAL-1 cell loss error seconds.*
CELL_LOSS_EFS	*AAL-1 cell loss percent errored free seconds.*
SN_ERR_CNT	*AAL-1 sequence number CRC/parity error count.*
SN_ERR_RAT_AVG	*AAL-1 sequence number CRC/parity error ratio average.*
SN_ERR_RAT_CUR	*AAL-1 sequence number CRC/parity error ratio current.*
SN_ERR_ES	*AAL-1 sequence number CRC/parity error seconds.*
SN_ERR_EFS	*AAL-1 sequence number CRC/parity error percent errored free seconds.*

Cell Delay Results

The following parameters retrieve ATM cell delay measurement results. Before using these commands, you must set the test mode to cell transfer delay or inter-arrival time (:SOURC:DM:ATM tst_mode,xver|inter).

:FETCh:DMOD:ATM? <*cell delay result*>

Parameter	Retrieved result
DLY_MIN_CUR	*Current minimum delay.*
DLY_MAX_CUR	*Current maximum delay.*
DLY_MIN_AVG	*Average minimum delay.*
DLY_MAX_AVG	*Average maximum delay.*
DLY_MIN_PEK	*Peak minimum delay.*
DLY_MAX_PEK	*Peak maximum delay.*
DLY_TYP	*Typical delay.*

OAM Results

The following parameters are used to retrieve ATM OAM cell measurement results.

:FETCh:DMOD:ATM? <*OAM result*>

Parameter	Retrieved result
OAM_F4_AIS	*OAM F4 (VC flow) AIS (alarm indication signal) alarm.*
OAM_F4_AIS_SEC	*OAM F4 AIS alarm seconds.*
OAM_F4_RDI	*OAM F4 RDI (remote defect indication) alarm.*
OAM_F4_RDI_SEC	*OAM F4 RDI alarm seconds.*
OAM_F5_AIS	*OAM F5 (VP flow) AIS alarm.*
OAM_F5_AIS_SEC	*OAM F5 AIS alarm seconds.*
OAM_F5_RDI	*OAM F5 RDI alarm.*
OAM_F5_RDI_SEC	*OAM F5 RDI alarm seconds.*

ATM STS-3c Results

The following parameters are used to retrieve ATM STS-3c measurement results. Before you use these parameters, you must set the ATM DMux to STS3c mapping using the :SENS:DM:ATMDMUX STS3c command (see page 11–3).

:FETCh:DMOD:ATM? <*STS-3c result*>

Parameter	Retrieved result
VAL_CELL_DEL	*Valid cell delineation.*
LOCS	*Loss of cell synchronization alarm.*
LOCS_SEC	*Loss of cell synchronization alarm seconds.*

ATM DS3/HEC Results

The following parameters retrieve ATM measurement results on DS3 HEC-based signals. Before using these commands you must set the ATM DMux to DS3/HEC mapping using :SENS:DM:ATMUX DS3_HEC.

:FETCh:DMOD:ATM? <*DS3 HEC result*>

Parameter	Retrieved result
DS3_VAL_CELL_DEL	*Valid cell delineation.*
DS3_LOCS	*Loss of cell synchronization alarm.*
DS3_LOCS_SEC	*Loss of cell synchronization alarm seconds.*

ATM Misinserted Cell Results

The following parameters retrieve ATM misinserted cell measurements. These commands are only valid when the receive channel type is set for AAL-1 (:SOUR:DM:ATMGEN RX_CELL_TYPE,AAL1, see page 12–14).

:FETCh:DMOD:ATM? <*misinsert result*>

Parameter	Retrieved result
MISINS_CELL_CNT	*Misinserted cell error count.*
MISINS_CELL_SEC	*Misinserted cell error seconds.*
MISINS_CELL_RAT_AVG	*Misinserted cell error average ratio.*
MISINS_CELL_RAT_CUR	*Misinserted cell error current ratio.*
MISINS_CELL_ES	*Misinserted cell error seconds.*
MISINS_CELL_EFS	*Misinserted cell error-free seconds.*

ATM PLCP Results

The following parameters are used to retrieve PLCP (physical layer convergence protocol) measurement results. Before you use these parameters, you must set the ATM DMUX to DS3/PLCP mapping using the :SENS:DM:ATMDMUX ds3 command (see page 11–3).

:FETCh:DMOD:ATM? <*PLCP result*>

Parameter	Retrieved result
VAL_PLCP_SYNC	*PLCP valid PLCP synchronization.*
PLCP_B1_BIP_CNT	*PLCP B1 BIP error count.*
PLCP_B1_BIP_RAT_AVG	*PLCP B1 BIP error ratio average.*
PLCP_B1_BIP_RAT_CUR	*PLCP B1 BIP error ratio current.*
PLCP_B1_BIP_ES	*PLCP B1 BIP errored seconds.*
PLCP_B1_BIP_EFS	*PLCP B1 BIP errored free seconds.*
PLCP_FRM_POI_CNT	*PLCP frame and path overhead error count.*
PLCP_FRM_FEBE_CNT	*PLCP FEBE error count.*

:FETCh:DMOD:ATM? <*PLCP result*>, continued

Parameter	Retrieved result
PLCP_FRM_POI_RAT_AVG	*PLCP frame and path overhead error ratio average.*
PLCP_FRM_POI_RAT_CUR	*PLCP frame and path overhead error ratio current.*
PLCP_FRM_POI_ES	*PLCP frame and path overhead errored seconds.*
PLCP_FRM_POI_EFS	*PLCP frame and path overhead percent errored free seconds.*
PLCP_FEBE_RAT_AVG	*PLCP FEBE error ratio average.*
PLCP_FEBE_RAT_CUR	*PLCP FEBE error ratio current.*
PLCP_FEBE_ES	*PLCP FEBE errored seconds.*
PLCP_FEBE_EFS	*PLCP FEBE percent errored free seconds.*
PLCP_LOF	*PLCP loss of frame alarm.*
PLCP_YEL	*PLCP Yellow alarm.*
LPCL_LOF_SEC	*PLCP loss of frame alarm seconds.*
PLCP_YEL_SEC	*PLCP Yellow alarm seconds.*

:FETCh:DMOD:DS0? <*result*>

This command retrieves DS0 measurement results. The following table lists the parameters to use and the retrieved results:

:FETCh:DMOD:DS0? <*result*>

Parameter	Retrieved result
BIT	*Number of bit errors detected.*
BIT_ARATIO	*Average bit error ratio.*
BIT_CRATIO	*Current bit error ratio.*
DATA	*Decimal value of the 8-bit data.*
FREQ	*Frequency of the received tone.*
SIGNALA	*Received A signaling bit status.*
SIGNALB	*Received B signaling bit status.*
SIGNALC	*Received C signaling bit status.*
SIGNALD	*Received D signaling bit status.*
VF_DBM	*RMS power of the received tone, in dBm.*
VF_3DBM_SEC	*Seconds with received tone greater than 3dBm.*

:FETCh:DMOD:DS1? <*result*>

This command retrieves DS1 measurement results. The following table lists the parameters to use with this command.

:FETCh:DMOD:DS1? <*DS1 result*>

	Parameter	Retrieved result
Pattern Bit Errors	*AVAIL_SEC*	*Number of available seconds.*
	UAS	*Number of unavailable seconds.*
	BIT	*DS1 bit error count.*
	BIT_ARATIO	*DS1 average bit error ratio.*
	BIT_CRATIO	*DS1 current bit error ratio.*
	BIT_CSES3	*DS1 consecutive SES count.*
	BIT_EFS	*DS1 bit error free seconds.*
	BIT_EFS_PC	*DS1 % bit error free seconds.*
	BIT_ES	*DS1 bit errored seconds.*
	BIT_SES	*DS1 bit severely errored seconds.*
	BIT_SYNC_ES	*DS1 bit synchronous ES.*
	DEGMIN	*DS1 degraded minute count.*
	TES3	*DS1 10-3 threshold ES.*
	TES4	*DS1 10-4 threshold ES.*
	TES5	*DS1 10-5 threshold ES.*
	TES6	*DS1 10-6 threshold ES.*
ESF Data Link	*DL_SYNC_SEC*	*Number of ESF datalink synchronous seconds.*
	DL_PAT	*The currently detected ESF datalink pattern.*

:FETCh:DMOD:DS1? <*DS1 result*>, continued

	Parameter	Retrieved result
Block Errors	*BLOCK*	*Number of block errors counted.*
	ESTBLOCK	*Estimated total number of blocks received.*
	BIT_BES	*Number of pattern burst error seconds.*
	CATV_UAS	*Number of Cable TV unavailable seconds.*
	SEE	*Number of Severe Error Events.*
	SUMMARY_1	*An eight-field data record. See Summary_1 Field Descriptions (Block Errors), page 6–30 for an explanation for each field.*
Signal	*RXFREQ*	*DS1 input frequency.*
	PEAKV	*DS1 peak voltage.*
	DBDSX	*DS1 signal level in dBdsx.*
Status	*SIGNAL*	*DS1 signal status.*
	SFSYNC	*DS1 SF (D4) detection status.*
	ESFSYNC	*DS1 ESF detection status.*
	PATTERN	*DS1 pattern detection status.*
	B8ZS	*DS1 B8ZS detection status.*
Alarms	*LOS*	*DS1 Loss of Signal alarm.*
	LOS_SEC	*DS1 LOS Seconds count.*
	LOF	*DS1 Loss of Frame alarm.*
	LOF_SEC	*DS1 LOF seconds.*
	LOP	*DS1 loss of pattern alarm.*
	LOP_SEC	*DS1 LOP seconds.*
	ONESDEN	*DS1 1s density alarm state (1=alarm tripped; 0=not tripped).*

:FETCh:DMOD:DS1? <*DS1 result*>, continued

	Parameter	Retrieved result
Alarms	*AIS*	*DS1 AIS alarm.*
	AIS_SEC	*DS1 AIS seconds.*
	YELLOW	*DS1 yellow alarm.*
	YELLOW_SEC	*DS1 yellow alarm seconds.*
	COFA	*DS1 change of frame alignment.*
	EX0	*DS1 excess 0s alarm.*
	IDLE	*DS1 idle signal status.*
	IDLE_SEC	*DS1 idle signal seconds.*
BPV	*BPV*	*DS1 bipolar violation count.*
	BPV_ARATIO	*DS1 BPV average error ratio.*
	BPV_CRATIO	*DS1 BPV current error ratio.*
	BPV_EFS	*DS1 BPV error free seconds.*
	BPV_EFS_PC	*DS1 BPV % bit error free seconds.*
	BPV_ES	*DS1 BPV errored seconds.*
CRC	*CRC*	*DS1 CRC error count.*
	CRC_ARATIO	*DS1 CRC average error ratio.*
	CRC_CRATIO	*DS1 CRC current error ratio.*
	CRC_EFS	*DS1 CRC error free seconds.*
	CRC_EFS_PC	*DS1 CRC % bit error free seconds.*
	CRC_ES	*DS1 CRC errored seconds.*

:FETCh:DMOD:DS1? <*DS1 result*>, continued

	Parameter	Retrieved result
Framing	*FRM*	*DS1 frame error count.*
	FRM_ARATIO	*DS1 frame average error ratio.*
	FRM_CRATIO	*DS1 frame current error ratio.*
	FRM_EFS	*DS1 frame error free seconds.*
	FRM_EFS_PC	*DS1 frame % bit error free seconds.*
	FRM_ES	*DS1 frame errored seconds.*
	FRM_SLIP	*DS1 frame slip count.*
Framing	*FRM_SLIP_SEC*	*DS1 frame slip seconds.*
	FRMCRC_ES	*DS1 frame/CRC errored seconds.*
	FRMCRC_ESA	*DS1 frm/CRC threshold A ES count.*
	FRMCRC_ESB	*DS1 frm/CRC threshold B ES count.*
	FRMCRC_SES	*DS1 frm/CRC severely ES.*
	FRMCRC_SES	*DS1 frm/CRC consecutive SES.*
Loopback	*LPUP_INB_SYNC*	*Inband loop-up synchronization.*
	LPUP_INB_SYNC_SEC	*Inband loop-up synchronization seconds.*
	LPUP_OUTB_SYNC	*Out-of-band loop-up synchronization.*
	LPUP_OUTB_SYNC_SEC	*Out-of-band loop-up synchronization seconds.*
	LPDN_INB_SYNC	*Inband loop-down synchronization.*
	LPDN_INB_SYNC_SEC	*Inband loop-down synchronization seconds.*
	LPDN_OUTB_SYNC	*Out-of-band loop-down synchronization.*
	LPDN_OUTB_SYNC_SEC	*Out-of-band loop-down sync seconds.*

:FETCh:DMOD:DS1? <*DS1 result*>, continued

	Parameter	Retrieved result
Jitter	*JITTER_CUR_PTOP_H*	*DS1 high band current p-to-p jitter.*
	JITTER_MAX_PTOP_H	*DS1 high band max p-to-p jitter.*
	JITTER_POS_PEAK_H	*DS1 high band positive peak jitter.*
	JITTER_NEG_PEAK_H	*DS1 high band negative peak jitter.*
	JITTER_MASK_PC_H	*DS1 high band max % jitter mask.*
	JITTER_HITS_H	*DS1 high band jitter hits count.*
	JITTER_HITS_SEC_H	*DS1 high band jitter hits time.*
	JITTER_CUR_PTOP_W	*DS1 wide band current p-to-p jitter.*
	JITTER_MAX_PTOP_W	*DS1 wide band max p-to-p jitter.*
	JITTER_POS_PEAK_W	*DS1 wide band positive peak jitter.*
	JITTER_NEG_PEAK_W	*DS1 wide band negative peak jitter.*
	JITTER_MASK_PC_W	*DS1 wide band max % jitter mask.*
	JITTER_HITS_W	*DS1 wide band jitter hits count.*
	JITTER_HITS_SEC_W	*DS1 wide band jitter hits time.*

:FETCh:DMOD:DS2? <*result*>

This command retrieves DS2 measurement results. The following table lists the applicable parameters and the retrieved results:

:FETCh:DMOD:DS2? <*DS2 result*>

Parameter	Retrieved result
LOS	*DS2 loss of signal alarm.*
AIS	*DS2 AIS alarm.*
LOF	*DS2 loss of frame alarm.*
XBIT	*DS2 X-bit status.*

:FETCh:DMOD:<DS3?|DS3B?> <*result*>

This command retrieves DS3 measurement results from the DS3 error, alarm, overhead, and frequency sensor (including pulse mask); and from the DS3 BPV, jitter, and voltage sensor.

For dual DS3 applications the level 2 node :DS3? retrieves measurements from the DS3-A sensor; :DS3B? retrieves measurements from the DS3-B sensor. For both commands, the result selections are the same. The following table lists the applicable parameters.

:FETCh:DMOD:<DS3?|DS3B?> <*DS3 result*>

	Parameter	Retrieved result
Signal	*PEAKV*	*DS3 peak voltage.*
	RXFREQ	*DS3 frequency.*
	DBDSX	*DS3 signal level in dBdsx.*
Status	*SIGNAL*	*DS3 Signal status.*
	CBITFRAME	*DS3 C-bit frame detect status.*
	M13FRAME	*DS3 M13 frame detect status.*
	PATTERN	*DS3 pattern detection status.*
	IDLE	*DS3 Idle status.*
	IDLE_SEC	*DS3 Idle status seconds.*
	XBIT	*DS3 X-bit status.*
	X1 | X2	*Current binary value of the DS3 X1 or X2 bit.*
	OBITS	*DS3 O-bits (binary value).*
Alarms	*LOS*	*DS3 Loss of Signal alarm.*
	LOS_SEC	*DS3 LOS Seconds count.*
	LOF	*DS3 LOF alarm.*
	LOF_SEC	*DS3 LOF seconds.*

:FETCh:DMOD:<DS3?|DS3B?> <*DS3 result*>, continued

	Parameter	Retrieved result
Alarms	*LOP*	*DS3 Loss of Pattern alarm.*
	LOP_SEC	*DS3 LOP seconds.*
	EX0	*DS3 excess zeros alarm.*
	AIS	*DS3 AIS alarm.*
	AIS_SEC	*DS3 AIS seconds.*
	LCVA	*DS3 Line Code Viol. alarm.*
	LCVA_SEC	*DS3 LCVA seconds.*
	BLUE	*DS3 Blue alarm.*
BPV	*BPV*	*DS3 bipolar violation count.*
	BPV_ARATIO	*DS3 BPV average error ratio.*
	BPV_CRATIO	*DS3 BPV current error ratio.*
	BPV_EFS	*DS3 BPV error free seconds.*
	BPV_EFS_PC	*DS3 BPV percentage error free seconds.*
	BPV_ES	*DS3 BPV errored seconds.*
Pattern Bit Errors	*BIT*	*DS3 bit error count.*
	BIT_ARATIO	*DS3 average bit error ratio.*
	BIT_CRATIO	*DS3 current bit error ratio.*
	BIT_CSES3	*DS3 consecutive 3 SES count.*
	BIT_EFS	*DS3 bit error free seconds.*
	BIT_EFS_PC	*DS3 % bit error free seconds.*
	BIT_ES	*DS3 bit errored seconds.*
	BIT_SES	*DS3 bit severely errored sec.*
	BIT_SYNC_ES	*DS3 bit synchronous ES.*

:FETCh:DMOD:<DS3?|DS3B?> <*DS3 result*>, continued

	Parameter	Retrieved result
Pattern Bit Errors	*TES3*	*DS3 10^{-3} Threshold ES.*
	TES4	*DS3 10^{-4} Threshold ES.*
	TES5	*DS3 10^{-5} Threshold ES.*
	TES6	*DS3 10^{-6} Threshold ES.*
C/P-Bit Errors	*CBIT*	*DS3 C-bit error count.*
	CBIT_ARATIO	*DS3 C-bit average error ratio.*
	CBIT_CRATIO	*DS3 C-bit current error ratio.*
	CBIT_EFS	*DS3 C-bit error free seconds.*
	CBIT_EFS_PC	*DS3 C-bit % error free seconds.*
	CBIT_ES_CNT	*DS3 C-bit errored seconds.*
	P1 | P2	*Current binary value of the DS3 P1 or P2 bit.*
	PBIT	*DS3 P-bit error count.*
	PBIT_ARATIO	*DS3 P-bit average error ratio.*
	PBIT_CRATIO	*DS3 P-bit current error ratio.*
	PBIT_EFS	*DS3 P-bit error free seconds.*
	PBIT_EFS_PC	*DS3 P-bit % error free seconds.*
	PBIT_ES_CNT	*DS3 P-bit errored seconds.*

:FETCh:DMOD:<DS3?|DS3B?> <*DS3 result*>, continued

	Parameter	Retrieved result
Frame/FFCV Errors	*FRM*	*DS3 frame error count.*
	FRM_ARATIO	*DS3 frame average error ratio.*
	FRM_CRATIO	*DS3 frame current error ratio.*
	FRM_EFS	*DS3 frame error free seconds.*
	FRM_EFS_PC	*DS3 frame % error free seconds.*
	FRM_ES	*DS3 frame errored seconds.*
	FFCV_FE	*DS3 frame format CV, far end.*
Frame/FFCV Errors	*FFCV_FESA*	*DS3 FFCV far end seconds A.*
	FFCV_FESB	*DS3 FFCV far end seconds B.*
	FFCV_FESC	*DS3 FFCV far end seconds c.*
	FFCV_NE	*DS3 frame format CV, near end.*
	FFCV_NESA	*DS3 FFCV near end seconds A.*
	FCV_NESB	*DS3 FFCV near end seconds B.*
	FFCV_NESC	*DS3 FFCV near end seconds c.*

:FETCh:DMOD:<DS3?|DS3B?> <*DS3 result*>, continued

	Parameter	Retrieved result
Jitter	*JITTER_CUR_PTOP_H*	*DS3 high band current p-to-p jitter.*
	JITTER_MAX_PTOP_H	*DS3 high band max p-to-p jitter.*
	JITTER_POS_PEAK_H	*DS3 high band positive peak jitter.*
	JITTER_NEG_PEAK_H	*DS3 high band negative peak jitter.*
	JITTER_MASK_PC_H	*DS3 high band max % jitter mask.*
	JITTER_HITS_H	*DS3 high band jitter hits count.*
	JITTER_HITS_SEC_H	*DS3 high band jitter hits time.*
	JITTER_CUR_PTOP_W	*DS3 wide band current p-to-p jitter.*
	JITTER_MAX_PTOP_W	*DS3 wide band max p-to-p jitter.*
	JITTER_POS_PEAK_W	*DS3 wide band positive peak jitter.*
	JITTER_NEG_PEAK_W	*DS3 wide band negative peak jitter.*
	JITTER_MASK_PC_W	*DS3 wide band max % jitter mask.*
	JITTER_HITS_W	*DS3 wide band jitter hits count.*
	JITTER_HITS_SEC_W	*DS3 wide band jitter hits time.*

:FETCh:DMOD:<DS3?|DS3B?> <*DS3 result*>, continued

	Parameter	Retrieved result
Pulse Mask (Available for DS3-A only)	*MSK_PSHAPE*	*Positive pulse shape results (pass, fail, or unavailable).*
	MSK_NSHAPE	*Negative pulse shape results (pass fail, unavailable).*
	MSK_PRISE	*Positive pulse rise time in nanoseconds.*
	MSK_NRISE	*Negative pulse rise time in nanoseconds.*
	MSK_PFALL	*Positive pulse fall time in nanoseconds.*
	MSK_NFALL	*Negative pulse fall time in nanoseconds.*
	MSK_PWIDTH	*Positive pulse width in nanoseconds.*
	MSK_NWIDTH	*Negative pulse width in nanoseconds.*
	MSK_PAMP	*Positive pulse amplitude (pass, fail, unavailable).*
	MSK_NAMP	*Negative pulse amplitude (pass, fail, unavailable).*
	MSK_PAMP_MV	*Positive pulse amplitude in millivolts.*
	MSK_NAMP_MV	*Negative pulse amplitude in millivolts.*
	MSK_IMB	*Pulse imbalance status (pass, fail, not applicable).*
	MSK_IMB_RATIO	*Pulse imbalance ratio.*
	MSK_PSTAT	*Status of positive pulse acquisition (text).*
	MSK_NSTAT	*Status of negative pulse acquisition (text).*
	MSK_UPPER, # *MSK_LOWER, #*	*These two commands return the standardized mask curve data points for the selected mask (see :SENSe:DM:MSKSEL <pulse mask>, page 11–15). The upper and lower boundaries of the mask are each described by 576 values.* *"#" can be set from 1 through 576, to retrieve a specific data point, or to "A" to retrieve all data points at once.*

:FETCh:DMOD:DS3:CBIT? <*row, cbit*>

This command retrieves the last C-bit values received by the test set.

row: The M3 subframe (1–7).

cbit: The C-bit whose value you want to retrieve (1, 2, or 3).

Example

```
:FETCh:DMOD:DS3:CBIT? 1, 3
```

- Retrieves the C3 bit in row 1 (the FEAC bit).

:FETCh:DMOD:DS3? <*FEAC code*>

This command retrieves DS3 FEAC codes. The following table lists the applicable parameters and the retrieved results:

:FETCh:DMOD:DS3? <*FEAC code*>

Parameter	Retrieved value
FEAC_CODE	*Last received FEAC loop code or alarm/status code (text and six-bit sequence).*
FEAC_CODE_SEC	*Number of seconds since last FEAC code received.*
FEAC_SEQ	*Type of FEAC control-code sequence (activate or deactivate) last received.*
FEAC_SEQ_SEC	*Number of seconds since last FEAC control-code sequence received.*
FEAC_LINE	*Line affected by most recently received FEAC loop-back control sequence (for example, DS1 #7).*

:FETCh:DMOD:E1? <*result*>

This command retrieves E1 measurement results. The following table lists the applicable parameters and the retrieved results.

:FETCh:DMOD:E1? <*E1 result*>

	Parameter	Retrieved result
Signal	*PEAKV*	*E1 peak voltage.*
	RXFREQ	*E1 frequency.*
	SIGNAL	*E1 Signal status.*
	REF_FREQ	*Reference signal frequency.*
	DELTA_FREQ	*Difference between E1 and reference signals.*
	DBDSX	*E1 signal level in dBdsx.*
Status	*MA*	*E1 signal level in milliamperes.*
	TSOSYNC	*TS0 synchronization status.*
	TS16SYNC	*TS16 synchronization status.*
	CRC4SYNC	*CRC-4 frame synchronization status.*
	PATTERN	*E1 pattern detection status.*
	HDB3	*Valid HDB3 signal status.*
Alarms	*LOS*	*E1 Loss of Signal alarm.*
	LOS_SEC	*E1 LOS Seconds count.*
	OOF	*Out of Frame events.*
	OOF_SEC	*Out of Frame seconds.*
	LOP	*E1 loss of pattern alarm.*
	LOP_SEE	*E1 LOP seconds.*
	AIS	*E1 Path AIS alarm*

:FETCh:DMOD:E1? <*E1 result*>, continued

	Parameter	Retrieved result
Alarms	*AIS_SEC*	*E1 Path AIS alarm seconds*
	RAI	*Remote alarm indication.*
	RAI_SEC	*Number of remote alarm indication seconds.*
	DISTANT	*Distant multiframe alarm.*
	DISTANT_SEC	*Distant multiframe alarm seconds.*
	EX0	*Excess zeros alarm status.*
BPV	*BPV*	*E1 bipolar violation count.*
	BPV_ARATIO	*E1 BPV average error ratio.*
	BPV_CRATIO	*E1 BPV current error ratio.*
	BPV_EFS	*E1 BPV error free seconds.*
	BPV_EFS_PC	*E1 BPV % error free seconds.*
	BPV_ES	*E1 BPV errored seconds.*
Pattern Bit Errors	*BIT*	*E1 bit error count.*
	BIT_ARATIO	*E1 average bit error ratio.*
	BIT_CRATIO	*E1 current bit error ratio.*
	BIT_CSES3	*E1 consecutive 3 SES count.*
	BIT_EFS	*E1 bit error free seconds.*
	BIT_EFS_PC	*E1 % bit error free seconds.*
	BIT_ES	*E1 bit errored seconds.*
	BIT_SES	*E1 bit severely errored sec.*
	BIT_SYNC_ES	*E1 bit synchronous ES.*
	BIT_DRIBBLE	*E1 dribbling error seconds.*
	AVAIL_SEC	*Available seconds.*

:FETCh:DMOD:E1? <*E1 result*>, continued

	Parameter	Retrieved result
Pattern Bit Errors	*UAS*	*Unavailable seconds.*
	DEGMIN	*Degraded minutes.*
	TES3	*E1 10^{-3} Threshold ES.*
	TES4	*E1 10^{-4} Threshold ES.*
	TES5	*E1 10^{-5} Threshold ES.*
	TES6	*E1 10^{-6} Threshold ES.*
Framing/CRC	*CRC4*	*Number of CRC-4 errors.*
	CRC4_ARATIO	*CRC-4 error ratio, average.*
	CRC4_CRATIO	*CRC-4 error ratio, current.*
	CRC4_EFS	*CRC-4 error-free seconds.*
	CRC4_EFS_PC	*CRC-4 percentage error-free seconds.*
	CRC4_ES	*CRC-4 errored seconds.*
	TS0_FRM	*TS0 frame errors.*
	TS0_FRM_ARATIO	*TS0 frame error ratio, average.*
	TS0_FRM_CRATIO	*TS0 frame error ratio, current.*
	TS0_FRM_EFS	*TS0 frame error-free seconds.*
	TS0_FRM_EFS_PC	*TS0 percentage frame error-free seconds.*
	TS0_FRM_ES	*TS0 frame errored seconds.*
	CAS_FRM	*CAS frame errors.*
	CAS_FRM_ARATIO	*CAS frame error ratio, average.*
	CAS_FRM_CRATIO	*CAS frame error ratio, current.*
	CAS_FRM_EFS	*CAS frame error-free seconds.*
	CAS_FRM_EFS_PC	*CAS percentage frame error-free seconds.*

:FETCh:DMOD:E1? <*E1 result*>, continued

	Parameter	Retrieved result
Framing/CRC	*CAS_FRM_ES*	*CAS frame errored seconds.*
Slips	*FRM_SLIP*	*Number of E1 frame slips.*
	FRM_SLIP_SEC	*Number of E1 frame slip seconds.*
	BIT_SLIP	*Number of E1 bit slips.*
Block Errors	*BLOCK*	*Number of block errors counted.*
	BIT_BES	*Number of burst error seconds.*
	CATV_UAS	*Number of Cable TV unavailable seconds.*
	SEE	*Number of Severe Error Events.*
	SUMMARY_1	*An eight-field data record (see pg 6–30).*
Jitter	*JITTER_CUR_PTOP_H*	*E1 high band current p-to-p jitter.*
	JITTER_MAX_PTOP_H	*E1 high band max p-to-p jitter.*
	JITTER_POS_PEAK_H	*E1 high band positive peak jitter.*
	JITTER_NEG_PEAK_H	*E1 high band negative peak jitter.*
	JITTER_MASK_PC_H	*E1 high band max % jitter mask.*
	JITTER_HITS_H	*E1 high band jitter hits count.*
	JITTER_HITS_SEC_H	*E1 high band jitter hits time.*
	JITTER_CUR_PTOP_W	*E1 wide band current p-to-p jitter.*
	JITTER_MAX_PTOP_W	*E1 wide band max p-to-p jitter.*
	JITTER_POS_PEAK_W	*E1 wide band positive peak jitter.*
	JITTER_NEG_PEAK_W	*E1 wide band negative peak jitter.*
	JITTER_MASK_PC_W	*E1 wide band max % jitter mask.*
	JITTER_HITS_W	*E1 wide band jitter hits count.*
	JITTER_HITS_SEC_W	*E1 wide band jitter hits time.*

:FETCh:DMOD:OPTions?

This command returns the 8-byte configuration code in the same format that is displayed on the system software configuration screen. The configuration code consists of a 24_character ASCII string, with 8-bites of hexadecimal data (each separated by a space), and null terminated.

Example

OPTions?

Retrieved data will typically reflect the following format:

07 2D 05 FF FD D8 FE F3

:FETCh:DMOD:VT15? <*result*>

This command retrieves VT1.5 measurement results. The following table lists the applicable parameters and the retrieved results:

:FETCh:DMOD:VT15? <*VT1.5 result*>

	Parameter	Retrieved result
Alarms	*AIS_V*	*VT path AIS alarm events.* *(formerly :FETC:DMOD:VT15? AIS)*
	AIS_V_SEC	*VT path AIS alarm seconds.* *(formerly :FETC:DMOD:VT15? AIS_SEC)*
	LOP_V	*VT loss of pointer alarm.* *(formerly :FETC:DMOD:VT15? LOPNTR)*
	LOP_V_SEC	*VT loss of pointer alarm seconds.* *(formerly :FETC:DMOD:VT15? LOPNTR_SEC)*
	RDI	*VT path remote defect indication alarm.* *(formerly :FETC:DMOD:VT15? YELLOW)*
	RDI_SEC	*VT path remote defect indication alarm seconds.* *(formerly :FETC:DMOD:VT15? YELLOW_SEC)*
	PSYNC	*VT P1/P0 sync loss alarm.*
	PSYNC_SEC	*VT P1/P0 sync loss seconds.*
	RDI_V	*VT remote defect indicator (RDI-V).*
	RDI_VSEC	*RDI-V seconds.*
	PDI_V	*VT Path defect indicator (PDI-V).*
	PDI_VSEC	*PDI-V seconds.*
	RFI_V	*VT remote failure indicator (RFI-V).*
	RFI_VSEC	*RFI-V seconds.*

:FETCh:DMOD:VT15? <*VT1.5 result*>, continued

	Parameter	Retrieved result
Pointer	*PNTR_VALUE*	*VT pointer value.*
	PNTR_PJSEC	*VT pointer justification seconds.*
	POS_PNTR_CNT	*Count of positive VT pointer adjustments.*
	NEG_PNTR_CNT	*Count of negative VT pointer adjustments.*
	V1	*VT V1 byte value.*
	V2	*VT V2 byte value.*
Code Violations	*CV*	*VT CV (parity error) count.*
	CV_ARATIO	*VT CV average bit error ratio.*
	CV_ES	*VT CV errored seconds.*
	CV_SES	*VT CV severely errored sec.*
	CV_UAS	*VT CV unavailable seconds.*
	CV_EFS	*VT CV error free seconds.*
	CV_PC_EFS	*VT CV % error free seconds.*
Overhead	*V4*	*The V4 byte value.*
	SIGLBL	*The VT signal label (bits 5–7 of the V5 byte). Returns bit value and text message.*
	Z6	*The Z6 (growth byte) value.*
	Z7	*The Z7 byte value. Bit 8 is used for RFI-V.*
	J2	*The VT path trace string. Returns 64 ASCII character string.*

:FETCh:DMOD:VT15? <*VT1.5 result*>, continued

	Parameter	Retrieved result
Remote Event Indication *(formerly FEBE)*	*REI_V*	*VT remote event indication (REI) count. (formerly :FETC:DMOD:VT15? FEBE)*
	REI_V_ARATIO	*VT REI average error ratio (formerly :FETC:DMOD:VT15? FEBE_ARATIO)*
	REI_V_CRATIO	*VT REI current error ratio (formerly :FETC:DMOD:VT15? FEBE_CRATIO)*
	REI_V_ES	*VT REI errored seconds. (formerly :FETC:DMOD:VT15? FEBE_ES)*
	REI_V_SES	*VT REI severely err sec. (formerly :FETC:DMOD:VT15? FEBE_SES)*
	REI_V_UAS	*VT REI unavailable seconds. (formerly :FETC:DMOD:VT15? FEBE_UAS)*
	REI_V_EFS	*VT REI error free seconds. (formerly :FETC:DMOD:VT15? FEBE_EFS)*
	REI_V_PC_EFS	*VT REI percentage of error-free seconds. (formerly :FETC:DMOD:VT15? FEBE_PC_EFS)*

:FETCh:DMOD:STS1? <*result*>

STS-1 Electrical Measurements

These commands retrieve STS-1 electrical measurement results from the STS1_Rx selector. The following table lists the applicable parameters and the retrieved results:

:FETCh:DMOD:STS1? <*STS-1 electrical result*>

Parameter	Retrieved result
PEAKV	*STS-1 peak voltage.*
BPV	*STS-1 bipolar violation count.*
BPV_ARATIO	*STS-1 BPV average error ratio.*
BPV_CRATIO	*STS-1 BPV current error ratio.*
BPV_EFS	*STS-1 BPV error free seconds.*
BPV_EFS_PC	*STS-1 BPV % error free seconds.*
BPV_ES	*STS-1 BPV errored seconds.*
RXFREQ	*STS-1 frequency. Note that if the receiver is set for OC-12 or OC-3 you must multiply the returned value by 12 or 3, respectively, to calculate the STS-N frequency.*
OPT_DBM	*Receive signal optical power in decibels relative to a milliwatt (dBm).*
DBDSX	*STS-1 electrical power, in decibels relative to the DSX level (dBdsx).*

STS-1 Overhead Byte Values

These commands retrieve STS-1 overhead byte values from the DS3Drop output of the matrix switch (before DS3 de-synchronization).

The following table lists the applicable parameters and the retrieved results. Some parameters are grouped together by function in the table, but you can only :FETCh the value of one parameter at a time.

:FETCh:DMOD:STS1? <*result*>

Parameter	Retrieved result
C2	*Decimal value of the path signal label.*
D1 | D2 | D3	*Decimal value of the section DCC.*
D4 | D5 | … | D12	*Decimal value of the line DCC.*
E1 | E2	*Decimal value of the local (E1) and express (E2) orderwires.*
F1	*Decimal value for the section (F1) user channel.*
G1	*Decimal value for the path status byte.*
H4	*Decimal value for the VT multiframe phase indicator.*
J0	*For OC-12 only. Retrieves the received 64-byte J0 byte string.*
J1	*Returns a 64-character ASCII string for the path trace signal. Each character represents one byte. Bytes without an ASCII equivalent are not displayed.*
K1 | K2	*Decimal value for the APS channel.*
Z1 | Z2	*Decimal value for the synchronization status (STS-1 #1) and growth bytes.*
Z3 | Z4 | Z5	*Decimal value for the growth bytes.*

STS-1 Path Measurements

These commands retrieve STS-1 Path overhead measurement results from the DS3Drop output of the matrix switch (before DS3 desynchronization). Additional Path measurements can be retrieved using the :STS1:PATH? commands (see *:FETCh:DMOD:STS1:PATH? <results>*, page 6–43).

The following table lists the applicable parameters and the retrieved results:

:FETCh:DMOD:STS1? <*path result*>

Parameter	Retrieved result
AIS_P	*STS-1 Path alarm indication signal alarm (formerly :FETC:DMOD:STS1:PATH? AIS).*
AIS_P_SEC	*STS-1 Path alarm indication signal alarm seconds (formerly :FETC:DMOD:STS1:PATH? AIS_SEC).*
RDI_P	*STS-1 Path remote defect indication alarm (formerly :FETC:DMOD:STS1? YELLOW).*
RDI_P_SEC	*STS-1 Path remote defect indication alarm seconds (formerly :FETC:DMOD:STS1? YELLOW_SEC).*
REI_P	*STS-1 Path remote event indication (REI) alarm count (formerly :FETC:DMOD:STS1:PATH? FEBE).*
REI_P_ARATIO	*STS-1 Path REI alarm average ratio (formerly :FETC:DMOD:STS1:PATH? FEBE_ARATIO).*
REI_P_CRATIO	*STS-1 Path REI alarm current ratio (formerly :FETC:DMOD:STS1:PATH? FEBE_CRATIO).*
REI_P_ES	*STS-1 Path REI errored seconds count (formerly :FETC:DMOD:STS1:PATH? FEBE_ES).*
REI_P_SES	*STS-1 Path REI severely errored seconds count (formerly :FETC:DMOD:STS1:PATH? FEBE_SES).*
REI_P_UAS	*STS-1 Path REI unavailable seconds count (formerly :FETC:DMOD:STS1:PATH? FEBE_UAS).*
REI_P_EFS	*STS-1 Path REI error-free seconds count (formerly :FETC:DMOD:STS1:PATH? FEBE_EFS).*

:FETCh:DMOD:STS1? <*path result*>, continued

Parameter	**Retrieved result**
REI_P_PC_EFS	*STS-1 Path REI percentage of error-free seconds (formerly :FETC:DMOD:STS1:PATH? FEBE_PC_EFS).*
LOP_P	*STS-1 Path loss of pointer alarm (formerly :FETC:DMOD:STS1? LOPNTR).*
LOP_P_SEC	*STS-1 Path loss of pointer alarm seconds (formerly :FETC:DMOD:STS1? LOPNTR_SEC).*

STS-1 Line Measurements

These commands retrieve STS-1 Line overhead measurement results from the DS3Drop output of the matrix switch (before DS3 desynchronization). Additional Line measurements can be retrieved using the :STS1:LINE? commands (see *:FETCh:DMOD:STS1:LINE? <result>*, page 6–44).

The following table lists the applicable parameters and the retrieved results:

:FETCh:DMOD:STS1? <*line result*>

Parameter	Retrieved result
AIS_L	*STS-1 Line alarm indication signal alarm (formerly :FETC:DMOD:STS1:LINE? AIS).*
AIS_L_SEC	*STS-1 Line alarm indication signal alarm seconds (formerly :FETC:DMOD:STS1:LINE? AIS_SEC).*
RDI_L	*STS-1 Line remote defect indication alarm (formerly :FETC:DMOD:STS1? LFERF).*
RDI_L_SEC	*STS-1 Line remote defect indication alarm seconds (formerly :FETC:DMOD:STS1? LFERF_SEC).*
REI_L	*STS-1 Line remote event indication (REI) alarm count (formerly :FETC:DMOD:STS1:LINE? FEBE).*
REI_L_ARATIO	*STS-1 Line REI alarm average ratio (formerly :FETC:DMOD:STS1:LINE? FEBE_ARATIO).*
REI_L_CRATIO	*STS-1 Line REI alarm current ratio (formerly :FETC:DMOD:STS1:LINE? FEBE_CRATIO).*
REI_L_ES	*STS-1 Line REI errored seconds count (formerly :FETC:DMOD:STS1:LINE? FEBE_ES).*
REI_L_SES	*STS-1 Line REI severely errored seconds count (formerly :FETC:DMOD:STS1:LINE? FEBE_SES).*
REI_L_EFS	*STS-1 Line REI error-free seconds count (formerly :FETC:DMOD:STS1:LINE? FEBE_EFS).*
REI_L_PC_EFS	*STS-1 Line REI percentage of error-free seconds (formerly :FETC:DMOD:STS1:LINE? FEBE_PC_EFS).*

STS-1 Pointer and Frequency Results

This commands retrieve STS-1 pointer and frequency measurement results from the STS1Drop output on the matrix switch. The following table lists the applicable parameters and the retrieved results:

:FETCh:DMOD:STS1? <*pointer/frequency result*>

Parameter	Retrieved result
POINTER	*STS-1 Valid pointer status.*
PNTR_VALUE	*STS-1 pointer value.*
H1	*STS-1 OH H1 byte decimal value.*
H2	*STS-1 OH H2 byte decimal value.*
PNTR_JUST_CNT	*STS-1 pointer justification count.*
PNTR_PJSEC	*STS-1 pointer justification seconds.*
PNTR_LAST_PJ_DIR	*STS-1 pointer new data flag seconds.*
PNTR_NDF_SEC	*STS-1 pointer new data flag seconds.*
DROP_FREQ	*STS-1 drop frequency.*
JITTER_CUR_PTOP_H	*STS-1 high band current p-to-p jitter.*
JITTER_MAX_PTOP_H	*STS-1 high band max p-to-p jitter.*
JITTER_POS_PEAK_H	*STS-1 high band positive peak jitter.*
JITTER_NEG_PEAK_H	*STS-1 high band negative peak jitter.*
JITTER_MASK_PC_H	*STS-1 high band max % jitter mask.*
JITTER_HITS_H	*STS-1 high band jitter hits count.*
JITTER_HITS_SEC_H	*STS-1 high band jitter hits time.*
JITTER_CUR_PTOP_W	*STS-1 wide band current p-to-p jitter.*
JITTER_MAX_PTOP_W	*STS-1 wide band max p-to-p jitter.*
JITTER_POS_PEAK_W	*STS-1 wide band positive peak jitter.*
JITTER_NEG_PEAK_W	*STS-1 wide band negative peak jitter.*
JITTER_MASK_PC_W	*STS-1 wide band max % jitter mask.*

:FETCh:DMOD:STS1? <*pointer/frequency result*>, continued	
Parameter	**Retrieved result**
JITTER_HITS_W	*STS-1 wide band jitter hits count.*
JITTER_HITS_SEC_W	*STS-1 wide band jitter hits time.*

STS-N Transport OH Results

These commands retrieve SONET transport overhead measurement results. The signal from which the measurement is retrieved depends on the configuration of the test set and certain :ROUTe:MATRix commands (see *STS-N Measurement Receive Signal Selection*, page 6–46). Other transport overhead measurements are retrieved using the :STS1:LINE? and :STS1:SECTion? commands (see *:FETCh:DMOD:STS1:LINE? <result>*, page 6–44, and *:FETCh:DMOD:STS1:SECTion? <result>*, page 6–45).

The following table lists the applicable parameters and the retrieved results.

FETCh:DMOD:STS1? <*STS-N result*>	
Parameter	**Retrieved result**
SIGNAL	*STS-1 signal status.*
FRAME	*STS-1 frame status.*
LOS	*STS-1 Loss of Signal alarm.*
LOS_SEC	*STS-1 Loss of Signal seconds count.*
LOF	*STS-1 Loss of Frame alarm.*
LOF_SEC	*STS-1 Loss of Frame seconds.*
LFERF	*STS-1 Loss of Far-end Receive Failure alarm.*
LFERF_SEC	*STS-1 LFERF seconds.*
LOCLK	*STS-1 Loss of Clock alarm.*
LOCLK_SEC	*STS-1 LOClk seconds.*
F1	*OH byte F1 value.*

FETCh:DMOD:STS1? <*STS-N result*>, continued

Parameter	Retrieved result
K1	*PH byte K1 value.*
K2	*OH byte K2 value.*
OOF_EV	*Out of Frame events. Note: not valid for OC-12.*
OOF_SEC	*Out of Frame seconds.*
OOF_EV_EFS	*Out of Frame error free seconds.*
OOF_EV_PC_EFS	*Out of Frame % error free seconds.*

:FETCh:DMOD:STS1:PATH? <*results*>

For SONET Path FEBE measurements, see the REI_P commands on page 6–36.

These commands retrieve STS-1 path overhead measurement results from the DS3Drop output of the matrix switch. Additional Path overhead results can be retrieved using some :STS1? commands (see *STS-1 Path Measurements*, page 6–37). The following table lists the applicable parameters and the retrieved results:

:FETCh:DMOD:STS1:PATH? <*result*>

Parameter	Retrieved result
F2	*STS-1 Path OH byte F2 value.*
CV	*STS-1 Path code violation (CV) count.*
CV_ARATIO	*STS-1 Path CV average BER.*
CV_ES	*STS-1 Path CV errored seconds.*
CV_SES	*STS-1 Path CV severely errored seconds.*
CV_UAS	*STS-1 Path CV unavailable seconds.*
CV_EFS	*STS-1 Path CV error-free seconds.*
CV_PC_EFS	*STS-1 Path CV percentage error-free seconds.*

:FETCh:DMOD:STS1:LINE? <*result*>

These commands retrieve STS-N Line overhead measurements. The signal from which the measurement is retrieved depends on the test set and :ROUTe:MATRix commands (see *STS-N Measurement Receive Signal Selection*, page 6–46). Additional Line overhead results can be retrieved using some :STS1? commands (see *STS-1 Line Measurements*, page 6–39).

The following table lists the parameters and the retrieved results:

:FETCh:DMOD:STS1:LINE? <*STS-N line overhead result*>

Parameter	Retrieved result
APS_BDG_CHAN	*APS bridge channel.*
APS_BYTE_FAIL_SEC	*APS Byte fail seconds.*
APS_MODE	*APS mode.*
APS_REQUEST_CHAN	*APS request channel.*
APS_STATE	*APS state.*
APS_STATE_CHNG_SEC	*APS state change seconds.*
APS_TYPE	*APS type.*
CV	*Line code violation (CV) count.*
CV_ARATIO	*Line CV average BER.*
CV_ES	*Line CV errored seconds.*
CV_SES	*Line CV severely errored seconds.*
CV_EFS	*Line CV error free seconds.*
CV_PC_EFS	*Line CV percentage error free seconds.*

:FETCh:DMOD:STS1:SECTion? <*result*>

These commands retrieve STS-N section overhead measurement results. The signal from which the measurement is retrieved depends on the configuration of the test set and certain ROUTe:MATRix commands (see *STS-N Measurement Receive Signal Selection*, page 6–46). Other transport overhead measurements are retrieved using the :STS1? and :STS1:PATH? commands (see *STS-N Transport OH Results*, page 6–41, and *:FETCh:DMOD:STS1:PATH? <results>*, page 6–43).

The following table lists the applicable parameters and the retrieved results:

:FETCh:DMOD:STS1:SECTion? <*STS-N section overhead result*>

Parameter	Retrieved result
CV	*Section CV count.*
CV_ARATIO	*Section CV average BER.*
CV_ES	*Section CV errored seconds.*
CV_SES	*Section CV severely errored seconds.*
CV_EFS	*Section CV error free seconds.*
CV_PC_EFS	*Section CV % error free seconds.*

STS-N Measurement Receive Signal Selection

STS-N transport measurement commands (STS1?, STS1:LINE?, and STS1:SECTion?) retrieve measurements based on only one received SONET signal. The measured SONET signal is determined as follows:

- After *RST is received, the test set retrieves the result values from the highest-level signal for which it is configured (OC-12, OC-3, STS-1 electrical)
- After ROUTe:MATRix <*XDrop*>,Ext is received where *XDrop* is STS1Drop, VT15Drop, or DS3Drop, the test set retrieves the result values from the received STS-1 electrical signal. If the STS1_Rx selector is then switched to OC1, the results are retrieved from the received OC-1 optical signal.
- After receiving any ROUTe:MATRix command not listed above, the test set retrieves the result values from the received OC-12 or OC-3 depending on the setting of the STS3_Rx selector.

Note: *The commands always specify FETCh:DMOD:STS1? <result>, even though the retrieved result may be measured from the received OC-12, OC-3, or OC-1 signal rather than the STS-1.*

:FETCh:DMOD:STS3C? <*result*>

This command retrieves STS-3c error and alarm measurements. The following table describes the applicable parameters to use with this command.

:FETCh:DMOD:STS3C? <*result*>

Parameter	Retrieved result
CONCAT	*The signal's concatenation status*
RXFREQ	*The STS-3 signal frequency.*
PATTERN	*The receive STS-3c data pattern*
LOP	*Loss of pointer events*
LOP_SEC	*Loss of pointer seconds*
BIT	*STS-3c bit errors*
BIT_ARATIO	*Average BER for bit errors*
BIT_CRATIO	*Current BER for bit errors*
BIT_ES	*Bit errored seconds*
BIT_SES	*Bit severely-errored seconds*
BIT_EFS	*Bit error-free seconds*
BIT_EFS_PC	*Percentage of bit error-free seconds*
H1H2_1	*Decimal value of the H1/H2 pointer bytes for STS-1 #1 in the STS-3c.*
H1H2_2	*Decimal value of the H1/H2 pointer bytes for STS-1 #2 in the STS-3c.*
H1H2_3	*Decimal value of the H1/H2 pointer bytes for STS-1 #3 in the STS-3c.*

:FETCh:DMOD:STS12C? <*result*>

This command retrieves STS-12c error and alarm measurements. The following table describes the applicable parameters to use with this command.

:FETCh:DMOD:STS12C? <*result*>

Parameter	Retrieved result
CONCAT	*The signal's concatenation status*
RXFREQ	*The STS-12 signal frequency.*
PATTERN	*The receive STS-12c data pattern*
LOP	*Loss of pattern events*
LOP_SEC	*Loss of pattern seconds*
BIT	*STS-12c bit errors*
BIT_ARATIO	*Average BER for bit errors*
BIT_CRATIO	*Current BER for bit errors*
BIT_ES	*Bit errored seconds*
BIT_SES	*Bit severely errored seconds*
BIT_EFS	*Bit error-free seconds*
BIT_EFS_PC	*Percentage of bit error-free seconds*
H1H2_1 through H1H2_12	*Decimal value of the H1/H2 pointer bytes.* *For example: :FETC:DMOD:STS12C? H1H2_7 returns the h1/H2 byte value for STS-1 #7 in the STS-12c.* *Note that h1/H2 #1 contains the pointer value, and H1/H2 #2 through #12 contain the concatenation indicator value.*

:INITiate and :ABORt Command Reference

:INITiate and :ABORt Command Reference

Use the :INITiate commands to clear results in the status registers, start a test, and begin accumulating results in the status registers. To stop a test and freeze the final results, use the :ABORt command.

:INITiate

After you enter the Reset command and set up signal paths, you use the :INITiate command to begin running the test.

Example

You might enter the following sequence of commands if you want to transmit a test signal containing DS3 patterns:

```
*RST

:ROUT:SEL DS3_TX,SOURCE

:ROUT:MATR DS3DROP,STS#2

:INIT
```

:ABORt

When you want to stop a test and freeze the accumulated results, you use the :ABORt command.

Example

After you initiate the test and request the status of some of the signal variables, you can abort the test:

```
*RST

:ROUT:SEL DS3_TX,SOURCE

:ROUT:MATR DS3DROP,STS#2

:INIT

:FETC:DMOD:DS3? SIGNAL

:FETC:DMOD:DS3? PATTERN

:ABOR
```

8

:INPut Command Reference

:INPut Command Reference

Use the :INPut command to control the characteristics of the input ports on the test set.

:INPut Command Structure

Root node	Level 1 node	Level 2 node	Parameter
:INPut	*:FILTer*	*:DS1*†	*<signal level>*
		:DS3†	*<signal level>*
		:DS3B†	*<signal level>*
		:STS1†	*<signal level>*
		:E1†	*<signal level>*
	:IMpedance†		*Returns setting for DS1 input impedance.*
	:TYPE†		*<termination>*

† These commands can also be used as quary commands by adding a question mark (?) after the level 2 node andomitting the second parameter. The test set will return the currently programmed value for that parameter.

Note: *The DS1 line code is set using the :SENS:DM:CODE1 command. See :SENSe:DM:CODE1 <line code>, page 11–4.*

:INPut:FILTer:DS1 <*signal level*>

This command sets the receive level filter for the DS1 input.

***RST default:** DSX1

Example

`:INPUT:FILTER:DS1 DSX1-MON`

- Sets the DS1 receive port to DSX-1 monitor level.

The following table lists the parameters to use with this command:

:INPut:FILTer:DS1 <*signal level*> Parameters

Parameter	Description
DSX1	*DS1 cross-connect level; equalized for 0–655 feet of cable.*
DSX1-MON	*DSX-1 monitor level; 10–25 dBdsx flat loss.*
ALBO	*Automatic Line Build Out; automatic equalization for 400–4000 feet of cable.*

Also, executing the SCPI query command `:INPut:FILTer:DS1?` returns the input filter settings for DS1 (digital signal level1).

:INPut:FILTer:DS3<B> <*signal level*>

This command sets the receive level filter for the DS3 input. For dual DS3, the level 2 node :DS3 sets the level for DS3-A; :DS3B sets the level for DS3-B. For both commands, the level selections are the same.

***RST default:** DSX3

Example

`:INPut:FILTer:DS3 HIGH`

- Sets the DS3 receive port to high level.

The following table lists the parameters to use with this command:

:INPut:FILTer:DS3 <*signal level*> Parameters

Parameters	Description
DSX3	*Automatic equalizer for 0–900 feet of cable.*
DSX3-MON	*DS3 monitor level; up to 26 dBdsx flat loss.*
HIGH	*High level signal; nominal 0.91 Vpk.*
LOW	*Low level signal; nominal 0.186 Vpk.*

Also, executing the SCPI query command `:INPut:FILTer:DS3?` returns the input filter settingsm for DS3 (digital signal level3).

:INPut:FILTer:STS1 <*signal level*>

This command sets the receive level filter for the STS-1 input.

***RST default:** STSX1

Example

`:INPut:FILTer:STS1 STSX1-MON`

- Sets the STS-1 receive port to STS-1 monitor level.

The following table lists the parameters to use with this command:

:INPut:FILTer:STS1 <*signal level*> Parameters

Parameters	Description
STSX1	*Automatic equalizer for 0–900 feet of cable.*
STSX1-MON	*Monitor level; up to 26 dBstsx flat loss.*
HIGH	*High level; nominal 1.110 Vpk input signal.*
LOW	*Low level; nominal 0.206 Vpk input.*

Also, executing the SCPI query command `:INPut:FILTer:STS1?` returns the input filter settings for STS1 (basic synchronous transport signal level1).

:INPut:FILTer:E1 <*signal level*>

This command sets the receive level filter for the E1 input.

***RST default:** DSX

Example

`:INPut:FILTer:E1 DSX`

- Sets the E1 receive port to the cross-connect level.

This parameter can be set to either **DSX** (cross-connect level) or **MONITOR** (monitor level).

Also, executing the SCPI query command `:INPut:FILTer:E1?` returns the input filter settings for E1 (CCITT digital signal).

:INPut:IMPedance? <Query *Command*>

This SCPI query command returns the settings of DS1 input impedance.

:INPut:TYPE <*termination mode*>

This command sets the termination mode for the DS1/E1 input port.

***RST default:** TERM

Example

`:INPut:TYPE BRIDGE`

- Sets the termination mode for the front-panel DS1 receive port to bridge.

This parameter can be set to either **TERM** (terminate) or **BRIDGE** (bridged).

Also, executing the SCPI query command `:INPut:TYPE?` returns the input settings of the DS1/E1 port imput termination.

9

:OUTPut Command Reference

:OUTPut Command Reference

The :OUTPut commands control the characteristics of the output ports on the test set.

:OUTPut Command Structure

Root node	Level 1 node	Level 2 node	Parameter
:OUTPut	*:FILTer*	*:DS1*†*/E1*	*<signal level>*
		:DS3†	*<signal level>*
		:STS1†	*<signal level>*
	:STATeN†		*<on/off>*

† These commands can also be used as quary commands by adding a question mark (?) after the level 2 node andomitting the second parameter. The test set will return the currently programmed value for that parameter.

:OUTPut:FILTer:DS1 <*signal level*>

This command sets the transmit signal level for the DS1 output.

***RST default:** DSX1

Example

```
:OUTP:FILT:DS1 DSX1
```

- Sets the DS1 transmit port to DSX-1 level.

The following table lists the parameters to use with this command:

:OUTPut:FILTer:DS1 <*signal level*> Parameters

Parameters	Description
DSX1	*DSX-1 level E1.*
LBO7.5	*Line Build Out of –7.5 dB.*
LBO15	*Line Build Out of –15 dB.*
LBO22	*Line Build Out of –22 dB.*

Also, executing the SCPI query command `:OUTPut:FILTer:DS1?` returns the setting of the transmit output port for DS1.

:OUTPut:FILTer:DS3 <*signal level*>

This command sets the transmit signal level for the DS3 output.

***RST default:** DSX3

Example

`:OUTP:FILT:DS3 HIGH`

- Sets the DS3 transmit signal level to high.

The following table lists the parameters to use with this command:

:OUTPut:FILTer:DS3 <*signal level*> Parameters

Parameters	Description
DSX3	*DS3 cross-connect level (LBO of 450 feet of cable).*
HIGH	*High level; no LBO.*
LOW	*Low level; flat loss from High level.*
900	*LBO simulating 900 feet of cable added.*

Also, executing the SCPI query command `:OUTPut:FILTer:DS3?` returns the setting of the transmit output port for DS3.

:OUTPut:FILTer:STS1 <*signal level*>

This command sets the transmit signal level for the STS-1 output.

***RST default:** STSX1

Example

`:OUTP:FILT:STS1 STSX1`

- Sets the STS-1 transmit signal level to STSX-1 level.

The following table lists the parameters to use with this command:

:OUTPut:FILTer:STS1 <*signal level*> Parameters

Parameters	Description
STSX1	*STSX-1 level; LBO simulating 450 feet of cable.*
HIGH	*High level; no LBO.*
LOW	*Low level; Flat loss from High level.*
900	*LBO simulating 900 feet of cable added.*

Also, executing the SCPI query command `:OUTPut:FILTer:STS1?` returns the setting of the transmit output port for STS1.

:OUTPut:STATe*N* <ON|OFF>

This command activates and deactivates the four transmit signals. Each source can be set **ON** (activated) or **OFF** (deactivated).

***RST default:** ON for all sources.

Example 1

```
:OUTP:STATe2 ON
```

- Activates the STS-1 transmit signal.

Example 2

```
:OUTP:STATe3 OFF
```

- Deactivates the DS3 transmit signal.

The following table lists the parameters to use with this command:

:OUTPut:STATe*N* Parameters

Parameter (*N*)	Description
:STATe1	*DS1/E1 transmit signal.*
:STATe2	*STS-1 transmit signal.*
:STATe3	*DS3 transmit signal.*
:STATe4	*Optical transmit signal.*

Also, executing the SCPI query command `:OUTPut:STATe1?` returns the state of DS1 output.

Executing the SCPI query command `:OUTPut:STATe2?` returns the state of STS1 output.

Executing the SCPI query command `:OUTPut:STATe3?` returns the state of DS3 output.

Executing the SCPI query command `:OUTPut:STATe4?` returns the state of Optical Output.

10

:ROUTe Command Reference

:ROUTe Command Reference

The :ROUTe commands set up signal paths in the test instrument.

:ROUTe Command Structure

Root node	Level 1 node	Level 2 node	Parameters	
:ROUTe	*:DMUX*†		*<demux>*	*<drop channel>*
	:MATRix		*<output>*	*<input>*
	:MUX	*:INS*†	*<mux>*	*<insert channel>*
		:OTHer†	*<mux>*	*<payload>*
	:SELect		*<signal>*	*<source/clock>*

These commands can also be used as query commands by adding a question mark (?) after the level 2 node and omitting the second parameter. The test set will return the currently programmed value for that parameter.

:ROUTe:DMUX *<demux>,<drop channel>*

This command selects the channel to be dropped from a demultiplexer.

***RST default:** 1 for all sources.

Example 1

```
:ROUT:DMUX VTGRP,2
```

```
:ROUT:DMUX VT15,4
```

- The first command drops VT group 2 from the VTGRP DMux; the second command drops VT 4 from the dropped group.

Example 2

```
:ROUT:DMUX STS12,3
```

- Drops the third STS-3 from the STS-12 demultiplexer (STS12 DMux).

The following table lists the parameters to use with this command:

:ROUTe:DMUX <*demux*>,<*drop channel*> Parameters

Demux	Channel	Description
DS0	*1–24*	*DS0 dropped from the DS0/TS demultiplexer (DS0/TS DMux).*
DS31	*1–28*	*DS1 dropped from the DS3 demultiplexer (DS31 DMux).* *In dual DS3 modes, the DS3 signal is selected by :ROUT:SEL DS1_Drop (see page 10–10).*
VTGRP	*1–7*	*VT group dropped from the VTGRP DMux*
VT15	*1–4*	*VT1.5 dropped from the VT15 DMux.*
STS12	*1–4*	*STS-3 dropped from the STS12 DMux.*

Also, executing the SCPI query command `:ROUTe:DMUX? DS0|DS31|VTGRP|VT15|STS12`returns the channel to be dropped from the demultiplexer.

:ROUTe:MATRix <*output*>,<*input*>

The matrix switch provides great flexibility in loading and unloading the STS-1 and STS-3 payloads.

:ROUTe:MATRix commands control the matrix switch by first identifying an output and then specifying the input to be connected to it.

*RST Defaults

For STS#1, STS#2, and STS#3 Outputs: DS3

For DS3Drop and STS1Drop Outputs: EXT

For test sets with a maximum rate of OC-3 or 0C-12: STS#1

For test sets with a maximum rate of STS-1: EXT

Example 1

```
:ROUT:MATR STS#1,STS#2
```

- Connects the STS#1 output of the matrix switch to the STS#2 input.

Example 2

```
:ROUT:MATR DS3DROP,STS#3
```

- Connects the DS3Drop output on the switch to the STS#3 input.

The following table lists the parameters to use with the :ROUT:MATR command. For a description of each input, see *Matrix Input Descriptions*, page 10–6.

:ROUTe:MATRix <*output*>,<*input*> Parameters

Output	Input	Description
STS#1	*STS#1 | STS#2 | STS#3 | VT15 | DS3 | EXT*	*STS#1 output of the matrix switch.*
STS#2	*STS#1 | STS#2 | STS#3 | VT15 | DS3 | EXT*	*STS#2 output of the matrix switch.*
STS#3	*STS#1 | STS#2 | STS#3 | VT15 | DS3 | EXT*	*STS#3 output of the matrix switch.*
STS3C	*CLRCH*	*Sets the matrix switch for STS-3c clear-channel transmit & receive.* *Note: This command overrides the setting of the DS3Drop output of the matrix switch.*
	ATM	*Sets the matrix switch for STS-3c ATM transmit and receive. The ATM Mux should be set to STS-3c mapping also.* *Do not use this command if you are using the :SOUR:DM:DATA4 ATM command (see page 12–27).*
DS3DROP	*STS#1 | STS#2 | STS#3 | VT15 | DS3 | Ext*	*Lets you control the DS3Drop output of the matrix switch.*
VT15DROP	*STS#1 | STS#2 | STS#3 | VT15 | DS3 | Ext*	*Lets you control the VT15Drop output of the matrix switch.*
STS1DROP	*STS#1 | STS#2 | STS#3 | VT15 | DS3 | Ext*	*Lets you control the STS1Drop output of the matrix switch.*

Note: *You can use these commands as query commands by adding a question mark (?) and omitting the Input parameter. The unit will return the current value. For example* `:ROUTe:MATRix? STS#1` *would retrieve the current input assigned to the STS-1 #1 output.*

Matrix Input Descriptions

The following table describes the input parameters to use with the :ROUT:MATR command.

:ROUTe:MATRix Input Parameters

Input	Description
STS#1	*Selects the STS#1 input.*
STS#2	*Selects the STS#2 input.*
STS#3	*Selects the STS#3 input.*
VT15	*Selects the internally generated VT1.5 payload, mapped onto an STS-1.*
DS3	*Selects the internally generated DS3 payload, mapped onto an STS-1.*
EXT	*External—Selects the output of the STS1_Rx selector.*
ATM	*Selects the internally generated ATM payload.*
CLRCH	*Clear channel—Selects the internally generated, non-channelized STS-3c signal.*

:ROUTe:MUX:INS <*mux*>,<*insert channel*>

This route command configures the channel to be inserted on a multiplexer.

***RST default:** 1 for all sources.

Example

```
:ROUT:MUX:INS DS0,2
```

- Inserts the signal from the DS0 Tx selector onto channel 2 of the DS0 Mux.

The following table lists the parameters to use with this command:

:ROUTe:MUX:INS <*mux*>,<*insert channel*> Parameters

Mux	Channel	Description
DS0	*1–24*	*Inserts the output of the DS0_Tx selector into the selected channel of the DS0 Mux.*
M13	*1–28*	*Inserts the output from the DS31_Ins selector on the selected channel of the M13 Mux.*
VT15	*1–4*	*Inserts the output of the VT_Ins selector on the selected channel of the VT15 Mux. (The VT15 Mux generates a VT Group signal containing 4 DS1 signals.)*
VTGRP	*1–7*	*Inserts the output of the VT15 Mux on the selected channel (VT group position) of the VTGrp Mux. (The VTGrp Mux generates an STS1 signal containing seven VT group signals.)*
STS12	*1–4*	*Inserts the output from the STS3 Mux on the selected STS-3 position of the STS12 Mux.*

Also, executing the SCPI query command `:ROUTe:MUX:INS? DS0|DS31|VTGRP|VT15|STS12` returns the channel inserted on the multiplexer.

:ROUTe:MUX:OTH *<mux>*,*<payload>*

This :ROUTe command configures the remaining channels (not specified in the :INS command) on the multiplexer.

***RST default for DS0:** ONES

***RST default for M13, VT15, and STS12:** SAME

***RST default for VTGrp and STS12:** 1

Example

```
:ROUT:MUX:OTH DS0,SAME
```

- The remaining DS0s are the same as the selected insert channel.

The following table lists the parameters to use with this command:

:ROUTe:MUX:OTH <*mux*>,<*payload*> Parameters

Mux	Payload	Description
DS0	*SAME*	*Uses the same pattern as the insert channel.*
	ONES	*Inserts an all-ones pattern on the other channels.*
	PASS	*Pass the other channels through as they are.*
M13	*AIS*	*Fills the other channels with DS1 AIS.*
	INV	*Uses an inverted copy of the insert channel for the others.*
	SAME	*Uses the same pattern as the insert channel.*

:ROUTe:MUX:OTH <*mux*>,<*payload*> Parameters, continued

Mux	Payload	Description
VT15	*AIS*	*Fills the other channels with AIS. Asynchronous VT only.*
	ZERO	*All zeros pattern.*
	INV	*Inverted copy of the primary insert channel.*
	SAME	*Uses the same pattern as the insert channel.*
	EXT	*External input from the DS1 Bipolar Receive port. Async only.*
	AUX	*Inserts the signal from the Auxiliary DS1 port on the other channels. Asynchronous VT only.*
VTGrp	*AIS*	*Fills the other channels with VT Path AIS.*
	SAME	*Uses the same pattern as the insert group.*
	UNEQ	*VT Unequipped.*
STS	*SAME*	*Uses the same pattern as the insert channel.*
	UNEQ	*STS Unequipped*
	EXT	*External input from STS-n Bipolar Port*
STS12	*UNEQ*	*STS Unequipped.*
	SAME	*Uses the same pattern as the insert channel.*
	PASS	*Pass–through of the three unused STS-3 signals, for drop and insert operations.*

Also, executing the SCPI query command `:ROUTe:MUX:OTH? DS0|DS31|VTGRP|VT15|STS12` returns the setting of the remaining channel on the demultiplexer.

:ROUTe:SELect <*selector*>,<*input source*>

Add a question mark (?) and omit the second parameter to *query* the unit.

These commands configure the transmit and receive selectors for the various signals.

Example

```
:ROUT:SEL DS1_Tx,LOOP
```

- Sets the DS1_Tx selector to the output from the DS1_Rx selector.

DS1 and DS0 Selectors

The following table lists the DS1 and DS0 selector commands.

:ROUTe:SELect <*selector*>,<*input source*> DS1/DS0 Parameters

Selector	Input	Description
DS0_TX	*SOURCE*	*Internal VF tone generator.*
	*EXT_VF**	*Rear-panel VF input.*
	EXT_RS232	*Rear-panel RS-232 input.*
DS1_RX	*BIPOLAR*	*Front-panel input.*
	VT15	*DS1 dropped from a VT1.5 mapped SONET signal.*
	*DS3**	*DS1 dropped from a DS3 signal.*
DS1_TX	*LOOP*	*Output of the DS1_Rx selector.*
	*SOURCE**	*DS1 internal source.*
	DS0	*Output of the DS0 Mux.*
	ATM	*Output of the ATM generator.* *Do not use this command if you are using the :SOUR:DM:DATA1 ATM command (see page 12–24).*

:ROUTe:SELect <*selector*>,<*input source*> DS1/DS0 Parameters, continued

Selector	Input	Description
DS1_DROP	*DS3_RXA* *DS3_RXB*	*(Dual DS3 modes). Selects the DS3 (A or B) from which the DS1 is dropped, and from which DS1 and DS2 measurements are made.*

DS3 and E1 Selectors

The following table lists the DS3 and E1 selector commands.

:ROUTe:SELect <*selector*>,<*input source*> DS3/E1 Parameters

Selector	Input	Description
E1_RX	*BIPOLAR*	*Front-panel input.*
	DS3	*DS1 dropped from a DS3 signal.*
E1_TX	*LOOP*	*Output of the E1_Rx selector.*
	*SOURCE**	*E1 internal source.*
	ATM	*Output of the ATM generator.* *Do not use this command if you are using the :SOUR:DM:DATA1 ATM command (see page 12–24).*
DS3_RX	*BIPOLAR*	*Front-panel input port.* *Note: This is the reset default for instruments configured for a maximum rate of DS3.*
	DROP	*DS3 drop from the matrix switch.* *Note: This is the reset default for instruments configured with STS-1 or higher.*
DS31_INS	*BIPOLAR*	*Front-panel DS1 bipolar receive port.*
	*DS1_TX**	*Output of the DS1_Tx selector.*

:ROUTe:SELect <*selector*>,<*input source*> DS3/E1 Parameters, continued

Selector	Input	Description
DS3_TX	*LOOP*	*DS3 dropped on the receive programming diagram (see page 18–4).*
	SOURCE	*DS3 internal source.*
	DS31DI	*Output of the DS3/1 drop & insert circuitry.*
	*DS31**	*Output of the M13 Mux.*
	ATM	*Output of the ATM Mux.* *Note: The ATM Mux should be set to DS3 mapping (see :SOUR:DM:ATMMUX command on page 12–8).* *Do not use this command if you are using the :SOUR:DM:DATA3 ATM command (see page 12–26).*
DS3_INS	*BIPOLAR*	*Front-panel DS3 Bipolar Receive port.*
	*DS3_TX**	*Output of the DS3_Tx selector.*

STS-N and VT Selectors

The following table lists the STS-N and VT selector commands.

:ROUT:SEL <*selector*>,<*input source*> STS-N/VT Parameters

Selector	Input	Description
VT_INS	*BIPOLAR*	*Front-panel DS1 bipolar receive port.*
	AUX	*Rear-panel DS1 input port.*
	*DS1_TX**	*Output of the DS1_Tx selector.*
STS1_RX	*OC1*	*Front-panel optical receive port.*
	NRZ	*Rear-panel STS-1 input.*
	*BIPOLAR**	*Front-panel STS-1 input.*
STS1_TX	*PASS*	*STS-1 pass-through from STS1_Rx selector.*
	*TX**	*STS-1 #1 from the matrix switch.*
	ATM	*Output of the ATM generator.*
STS_BPTX	*PASS*	*STS-1 pass-through from the front-panel STS-1 port.*
	*TX**	*STS-1 #1 from the matrix switch.*
STS3_RX	*OC3*	*Front-panel optical receive port.* *Note: This is the reset default for OC-3 instruments.*
	NRZ	*Rear-panel STS-3 input.*
	DROP	*STS-3 dropped from the OC12 DMux.* *Note: This is the reset default for OC-12 equipped instruments.*
STS3_TX	*PASS*	*STS-3 pass-through from STS3_Rx selector.*
	*TX**	*STS-3 from the STS3 Mux.*
STS12_RX	*OC12*	*Front-panel optical receive port.*
	NRZ	*Rear-panel STS-12 input.*

:ROUT:SEL <*selector*>,<*input source*> STS-N/VT Parameters, continued

Selector	Input	Description
STS12_TX	*PASS*	*STS-12 pass-through from front-panel optical rx port.*
	TX	*STS-12 from the STS12 Mux.*
STS12C	*CLRCH*	*STS-12c set to clear channel.*
	ATM	*STS-12c set to ATM.*
STS12PYLD	*STS12C*	*STS-12 payload set to concatenated (STS-12c).*
	STS3	*STS-12 payload set to channelized (four STS-3s).*
OPT_TX	*OC12*	*Output of the STS12 Tx selector.*
	OC3	*Output of the STS3_Tx selector.* *Note: This is the reset default for OC-3 equipped instruments.*
	OC1	*Output of the STS1_Tx selector.*

RS-232 Datalink Selector

The following table lists the RS-232 datalink selector commands. These commands affect the routing of signals to and from the rear-panel RS-232 datalink port. Port input and output functions are controlled simultaneously.

:ROUTe:SELect <*selector*>,<*input source*> Parameters

Selector	Input	Description
RS232	*NONE*	*The RS-232 datalink port is disabled.*
	F1	*SONET section user channel. Port input is routed to STS-12, STS-3, and STS-1 Tx signals. Port output is routed from optical receive, STS3_Rx selector, or STS1_Rx selector.*
	E1	*SONET section orderwire. See routing for F1 above.*
	E2	*SONET line orderwire. See routing for F1 above.*
	CBIT_R5	*DS3 C-bit PMDL (row 5). Port input is routed to DS3_Tx and DS3 Bipolar transmit. Port output is routed from DS3_Rx selector.*
	CBIT_R67	*DS3 C-bit rows 6 and 7. See routing for CBit_R5 above.*
	CBIT_R267	*DS3 C-bit rows 2, 6, and 7. See routing for CBit_R5 above.*
	ESF_SLC	*DS1 ESF or SLC-96 data link channel. Port input is routed to DS1_Tx and DS1 bipolar transmit. Port output is routed from DS1/E1_Rx selector.*
	DS0	*DS0 data. Port input is routed to the DS0_Tx selector. Port output is routed from DS0/TS DMux.*

:ROUTe:SELect <selector>,<input source>

11

:SENSe Command Reference

:SENSe Command Reference

Use :SENSe commands to control test set measurement functions.

:SENSe Command Structure

Root node	Level 1 node	Level 2 node	Level 3 node	Parameter
SENSe	*:AU*	*:STORE?*		*<query>*
		:TESTDURMODE	*TIMER*	
		:TESTDURHOURS		
		:TESTDURMINUTES		
		:TESTDURSECONDS		

:SENSe Command Structure

Root node	Level 1 node	Level 2 node	Level 3 node	Parameter
:SENSe	*:DM*	*:ATMDMUX*†		*<mapping>*
		:CODE1†		*<line code>*
		:DATA	*:IERRor*†	*<on/off>*
			TRBLscan†	*<on/off>*
		:DATA1		*<pattern>*
		:DATA3		*<pattern>*
		:DATA3B		*<pattern>*
		:DS3B†		*LEDS,<share>*
		:FRAMe	*:TYPE1*†	*<format>*
			:TYPE3†	*<format>*
			:TYPE3B†	*<format>*
		:PROG1	*BLOCK_SIZE*	*<size>*
		:JITTer†	*RATE*	*<rate>*
			WB_THRESH	*<threshold>*
			HB_THRESH	*<threshold>*
		:MSKSEL†		*<pulse mask>*
		:MSKPOL†		*<pulse polarity>*
		:MSKTOL†		*<mask tolerance>*
		:MSK†		*<start/stop>*

† *These commands can also be used as query commands by adding a question mark (?) after the level 2 node and omitting the second parameter. The test set will return the currently programmed value for that parameter.*

:SENSe:AU:STORE?

Use this :SENSE:AU:STORE? command in SCPI mode to quary the position number that was last stored.

```
:SENSe:AU:STORE?
```

:SENSe:AU:TESTDURMODE TIMER

Use this :SENSe:AU:TESTDURMODE TIMER command in SCPI mode to set test mode duration

```
:SENSe:AU:TESTDURMODE TIMER
```

:SENSe:AU:TIMERDURHOURS<00-99>

Use this :SENSe:AU:TESTDURHOURS 1 command in SCPI mode to set test duration to one hour.

```
:SENSe:AU:TESTDURHOURS 1
```

:SENSe:AU:TIMERDURMINUTES <00-59>

Use this :SENSe:AU:TESTDURMINUTES 22 command in SCPI mode to set test duration to 22 minutes.

```
:SENSe:AU:TESTDURMINUTES 22
```

:SENSe:AU:TIMERDURSECONDS <00-59>

Use this :SENSe:AU:TESTDURSECONDS 15 command in SCPI mode to set test duration to 15 seconds.

```
:SENSe:AU:TESTDURSECONDS 15
```

:SENSe:DM:ATMDMUX <*mapping*>

Use this :SENSe command to set the ATM DMUX mapping before you try to retrieve ATM measurement results. (See the ATM :FETCh commands on page 6–4). If you do not have the ATM option installed on your instrument, the instrument returns a value of -1 after you enter this command.

Note: *Do not use this command if you are also using a :SOUR:DM:DATAn ATM command.*

***RST default:** STS3C

Example

`:SENS:DM:ATMDMUX DS3`

- Sets PLCP-based ATM mapping on the DS3 signal.

Note: *When you set the mapping with this command, the :SOUR:DM:ATMMUX automatically gets set to the same setting.*

The following table lists the parameters to use with this command:

:SENSe:DM:ATMMUX <*mapping*> Parameters

Mapping	Description
DS3	*Sets the ATM demux to DS3 PLCP-based mapping.*
DS3_HEC	*Sets the ATM demux to DS3 HEC-based mapping.*
STS3C	*Sets the ATM demux to STS-3c mapping.*
STS12C	*Sets the ATM demux to STS-12c mapping.*
STS1	*Sets the ATM demux to STS-1 mapping.*
E1	*Sets the ATM demux to E1 mapping.*
DS1	*Sets the ATM demux to DS1 mapping.*

Also, executing SCPI query command :`SENSe:DM:ATMMUX` returns the setting of the ATM physical layer mapping.

:SENSe:DM:CODE1 <*line code*>

Use this :SENSe command to simultaneously switch the line code for the DS1 Bipolar Receive, the DS1 Bipolar Transmit, and the DS1 Bipolar Drop ports.

***RST default:** AMI

Example

```
:SENS:DM:CODE1 AMI
```

- Sets the DS1 line code to AMI.

The following table lists the parameters to use with this command:

:SENSe:DM:CODE1 <*line code*> Parameters

Parameters	Description
AMI	*Alternate Mark Inversion.*
B8ZS	*Bipolar with 8 zero substitution.*
HDB3	*High-density bipolar with 3-zero substitution (E1).*

Also, executing the SCPI query command `:SENSe:DM:CODE1? DS0|DS31|VTGRP|VT15|STS12` returns the setting of the DS1 line code.

:SENSe:DM:DATA1 <*pattern*>

Add a question mark (?) and omit the parameter to *query* the unit for the current value. For example `:SENS:DM:DATA1?`.

The DS1 Error, Alarm, Freq Measure sensor can be set to look for a particular data pattern. Status and alarm information is then based on whether these selected framing and data patterns have been found in the measured signal. The selected data pattern is also used for detection of bit errors.

This :SENSe command identifies the transmit pattern when DS1 is the source.

***RST default:** QRSS

Example

`:SENS:DM:DATA1 AIS`

- Sets the DS1 transmit pattern to AIS.

Note: *This :SENSe command also sets up the DS1 internal source, and can therefore be used interchangeably with the corresponding SOURce commands.*

The following table lists the parameters to use with this command:

:SENSe:DM:DATA1 <*pattern*> Parameters

Parameter	Description
AIS	*Alarm Indication Signal.*
PROG	*User-Programmable pattern.*
QRSS | QRSSINV	*Quasi-random signal.*
2^23-1 | 2^23-1INV *2^20-1 | 2^20-1INV* *2^15-1 | 2^15-1INV* *2^11-1 | 2^11-1INV* *2^9-1 | 2^9-1INV* *2^6-1 | 2^6-1INV*	*PRBS (pseudo-random bit sequence) patterns, for example: 2^23-1 =* 2^{23}*-1 PRNS.* *The "INV" selections invert the PRBS.*
ONES | ZEROS | ALT	*All-ones, all-zeros, or alternating ones and zeros.*
1IN8 | 2IN8 | 3IN24	*One-in-eight pattern, and so forth.*
55OCTET	*The 55-octet (or Daly) pattern.*

:SENSe:DM:DATA1 HOLDOFF,<*interval*>

This :SENSe command sets the programmable interval for DS1 LOP and OOF hold-off.

***RST default:** 0.0

Example

```
:SENSe:DM:DATA1 HOLDOFF,2.0
```

- Sets the programmable interval T to two seconds.

The following table lists the parameters to use with this command:

:SENSe:DM:DATA1 HOLDOFF,<*interval*> Parameters

Parameter	Description
0.0	*0 seconds.*
0.5	*0.5 seconds.*
1.0	*1 second.*
1.5	*1.5 seconds.*
2.0	*2 seconds.*
2.5	*2.5 seconds.*
3.0	*3 seconds.*
3.5	*3.5 seconds.*
4.0	*4 seconds.*
4.5	*4.5 seconds.*

:SENSe:DM:DATA3<B> <*pattern*>

Add a question mark (?) and omit the parameter to *query* the unit for the current value. For example: `:SENS:DM:DATA3?`.

This command sets the data pattern for the DS3 error, alarm, overhead, and frequency measure sensor. The pattern selection is used to compare to the receive pattern for bit errors, LOP, and so forth.

For dual DS3 applications the level 2 node :DATA3 sets the pattern for the DS3-A sensor; :DATA3B sets the pattern for the DS3-B sensor. For both commands, the pattern selections are the same.

***RST default:** 2^15-1

Example

`:SENS:DM:DATA3 IDLE`

- Sets the DS3 transmit pattern to idle.

The following table lists the parameters to use with this command:

NOTE:

This :SENSe command also sets up the DS3 internal source (Except for BITERR_ON and BITERR_OFF) (DS3-A only), and can therefore be used interchangeably with the corresponding :SOURce commands.

:SENSe:DM:DATA3 <*pattern*> Parameters

Parameter	Description
AIS	*Alarm Indication Signal.*
BITERR_ON	*Enable Display of DS3 bit errors*
BITERR_OFF	*Disable Display of DS3 bit errors*
PROG	*User-Programmable pattern.* *Note: If you select this pattern, the user-programmable pattern #1 as configured from the front panel is used.*
QRSS	*Quasi-random signal.*
2^23-1 *2^23-1INV*	*2^{23}-1 PRBS (pseudo-random bit sequence).* *"INV" inverts the PRBS.*
2^20-1 *2^20-1INV*	*2^{20}-1 PRBS*
2^15-1 *2^15-1INV*	*2^{15}-1 PRBS*

:SENSe:DM:DATA:IERRor <ON|OFF>

This :SENSe command, Inhibit Errors on alarm, tells the test set to inhibit counting of errors during related alarm conditions. This prevents error counts from running during Loss Of Signal, or Loss Of Frame conditions.

***RST default:** ON

Example

`:SENS:DM:DATA:IEER ON`

- Errors are not counted during alarm conditions.

This command can be set to ON or OFF.

Also, the SCPI query command `:SENSe:DM:DATA:IERRor?` returns the inhibit error on the alarm setting during the related alarm condition.

:SENSe:DM:DATA:TRBLscan <ON|OFF>

This :SENSe command switches the Trouble Scan feature On or Off. The configuration set by this command will be retained when the unit is returned to front-panel control.

***RST default:** ON

Example

`:SENS:DM:DATA:TRBL ON`

- Turns Trouble Scan on.

Also, executing the SCPI query command `:SENSe:DM:DATA:TRBLscan`? returns the on/off setting of the trouble scan.

:SENSe:DM:FRAMe:TYPE1 <*frame format*>

This :SENSe command sets the DS1 frame type.

***RST default:** ESF

Example

`:SENS:DM:DATA:TYPE1 SF`

- This sets the DS1 frame type to SuperFrame format.

The following table lists the parameters to use with this command:

:SENSe:DM:FRAMe:TYPE1 <*framing format*> Parameters

Parameter	Description
SF	*SuperFrame format.*
ESF	*Extended SuperFrame format.*
SLC-96	*SLC-96 frame format.*
NONE	*Unframed.*

Also, executing the SCPI query command `:INput:FILTer:DS1?` returns the framing type as specified by the command.

:SENSe:DM:DS3B LEDS,<*sharing mode*>

This command sets the LED sharing mode for the test set's front-panel DS3 ALARMS and STATUS indicator LEDs in dual DS3 modes.

***RST default:** NONE

Example

`:SENS:DM:DS3B LEDS,ALM_STAT`

- ALARMS and STATUS indicator are shared between DS3-A and DS3-B.

The sharing mode selections for this command are as follows:

NONE: LEDs are not shared. The front-panel DS3 ALARMS and STATUS indicators show only DS3-A information.

ALARM: The ALARMS indicators are shared, indicating when there is a current or historical alarm condition on either DS3-A or DS3-B.

ALM_STAT: ALARMS and STATUS indicators are shared, indicating when there is alarm, history, and signal status conditions on either DS3-A or DS3-B.

Also, executing the SCPI query command `:SENSe:DM:DS3B? LEDS` returns the DS3B LED to sharing mode.

:SENS:DM:FRAM:TYPE3<B> <*frame format*>

This command sets the framing format for the DS3 error, alarm, overhead, and frequency measure sensor. The framing selection is used to compare to the receive signal for framing mismatch, framing errors, and so forth.

For dual DS3 applications the level 3 node :TYPE3 sets the framing for the DS3-A sensor; :TYPE3B sets the pattern for the DS3-B sensor. For both commands, the framing selections are the same.

***RST default:** M13

Example

`:SENS:DM:FRAM:TYPE3 M13`

- This sets the DS3 frame type to M13 format.

The following table lists the parameters to use with this command:

:SENSe:DM:FRAMe:TYPE3 <*framing format*> Parameters

Parameter	Description
CBIT	*DS3 C-bit frame format.*
M13	*M13 framing format.*
NONE	*Unframed.*

Also, executing the SCPI query command `:SENSe:DM:FRAMe:TYPE3?` returns the framing type as specified by the command.

Also, executing the SCPI query command `:SENSe:DM:FRAMe:TYPE3B?` returns the framing type as specified by the command.

:SENSe:DM:PROG1 BLOCK_SIZE <*size*>

This :SENSe command lets you program the size of the DS1 blocks.

***RST default:** 2K

Example

`:SENS:DM:PROG1 BLOCK_SIZE 3K`

- Sets the block size for the instrument to 3 kilobits.

Add a question mark (?) and omit the parameter to *query* the unit for the current value. For example `:SENS:DM:PROG1?`.

The block size can be set from 2K through 8K.

:SENSe:DM:JITTer RATE,<*operating rate*>

This :SENSe command lets you select the jitter measurement rate.

***RST default:** Highest installed rate.

Example

`:SENS:DM:JITT RATE,DS1`

- This sets the jitter measurement rate to DS1.

The following table lists the parameters to use with this command:

:SENSe:DM:JITTer <*operating rate*> Parameters

Parameter	Description
DS1	*DS1 jitter measurement.*
E1	*E1 jitter measurement.*
DS3	*DS3 jitter measurement.*
STS1	*STS1 jitter measurement.*

Also, executing the SCPI query command `:SENSe:DM:JITTer? RATE` returns the rate jitter measurement.

:SENSe:DM:JITTer WB_THRESH,<*threshold*>

This :SENSe command sets the wide band jitter hit threshold.

***RST default:** 5.0

Example

`:SENS:DM:JITT WB_THRESH,3`

- This sets the hit threshold to 3.0.

Where the threshold can be a number from 0 through 6.9.

Also, executing the SCPI query command `:SENSe:DM:JITTer? WB_THRESHold` returns the wide band threshold jitter measurement.

:SENSe:DM:JITTer HB_THRESH,<*threshold*>

This :SENSe command sets the high band jitter hit threshold.

***RST default:** 0.5

Example

```
:SENS:DM:JITT HB_THRESH,1
```

- This sets the hit threshold to 1.

Where the threshold can be a number from 0 through 1.9.

Also, executing the SCPI query command `:SENSe:DM:JITTer? HB_THRESHold` returns the high band threshold jitter measurement.

:SENSe:DM:MSKSEL <*pulse mask*>

This :SENSe command sets the pulse mask.

***RST Default:** none

Example

`:SENS:DM:MSKSEL T1_102`

- This selects the T1.104 DS3 pulse mask.

The pulse mask can be set to **NONE**, **G_703** (G.703), **T1_102** (T1.102), **T1_404** (T1.404), or **TR499** (TR-NWT-000499).

Also, executing the SCPI query command `:SENSe:DM:MSKSEL?` returns the pulse mask setting.

:SENSe:DM:MSKPOL <*pulse polarity*>

This :SENSe command selects the pulse polarity to be acquired.

***RST Default:** POS

Example

`:SENS:DM:MSKPOL POS+NEG`

- This selects both positive and negative pulses to be acquired.

The polarity can be set to **POS** (positive), **NEG** (negative), or **POS+NEG** (both polarities).

Also, executing the SCPI query command `:SENSe:DM:MSKPOL?` returns the mask polarity setting.

:SENSe:DM:MSKTOL <*mask tolerance*>

This :SENSe command sets the adjustment to the pulse mask tolerance.

***RST Default:** spec

Example

`:SENS:DM:MSKTOL 3PC`

- This adjusts the mask tolerance by 3 percent.

The mask tolerance can be set to **SPEC** (no adjustment), **3PC** (3 percent), or **6PC** (6 percent).

Also, executing the SCPI query command `:SENSe:DM:MSKTOL?` returns the mask tolerance setting.

:SENSe:DM:MSK <*start|stop*>

This :SENSe command starts or stops a pulse mask test.

***RST Default:** pulse mask test not active.

Example

`:SENS:DM:MSK START`

- This begins the pulse mask test.

The pulse mask test can be activated (**START**) or halted (**STOP**) using this command.

Also, executing the SCPI query command `:SENSe:DM:MSK?` returns the mask start stop switch.

12

:SOURce Command Reference

:SOURce Command Reference

The :SOURce commands affect the transmitter side of the instrument.

:SOURce Command Structure

<table>
<tr><th>Root node</th><th>Level 1 node</th><th>Level 2 node</th><th>Level 3 node</th><th>Level 4 node</th><th>Parameters</th></tr>
<tr><td>*:SOURce*</td><td>*:LOCAL*†</td><td></td><td></td><td></td><td>*<LOCK | UNLOCK>*</td></tr>
<tr><td></td><td>*:DM*</td><td>*:ALARm*†</td><td></td><td></td><td>*<alarm type>,<on|off>*</td></tr>
<tr><td></td><td></td><td>*:ATMMUX*†</td><td></td><td></td><td>*<mapping>*</td></tr>
<tr><td></td><td></td><td>*:ATMgen*†</td><td></td><td></td><td>*<parameter>,<setting>*</td></tr>
<tr><td></td><td></td><td>*:CLOCkn*</td><td></td><td></td><td>*<signal> <timing source>*</td></tr>
<tr><td></td><td></td><td>*:CODE1*</td><td></td><td></td><td>*<line code>*</td></tr>
<tr><td></td><td></td><td>*:DATAn*†</td><td></td><td></td><td>*<signal> <payload>*</td></tr>
<tr><td></td><td></td><td>*:EINJect*†</td><td>*:DS1|DS2|DS3|DS3B|E1*</td><td></td><td>*<error type>,<rate>*</td></tr>
<tr><td></td><td></td><td></td><td>*:VT15*†</td><td></td><td>*<error type>,<rate>*</td></tr>
<tr><td></td><td></td><td></td><td>*:STS1|STS1#n*</td><td></td><td>*<error type>,<rate>*</td></tr>
<tr><td></td><td></td><td></td><td>*:STS3*</td><td>*:STS1PATH*</td><td>*<source>,<status>*</td></tr>
<tr><td></td><td></td><td></td><td>*STS3C | STS12C*</td><td></td><td>*<error type>,<rate>*</td></tr>
<tr><td></td><td></td><td></td><td>*:STS12*</td><td></td><td>*<error type>,<rate>*</td></tr>
<tr><td></td><td></td><td>*:FEAC*</td><td>*:CBIT*</td><td></td><td>*<row, cbit, 0 | 1>*</td></tr>
<tr><td></td><td></td><td></td><td>*:CONT_ALM_STAT*</td><td></td><td>*<ON | OFF>*</td></tr>
<tr><td></td><td></td><td></td><td>*:BURST_SETTING*</td><td></td><td>*<setting>*</td></tr>
<tr><td></td><td></td><td></td><td>*:LOOPBACK_LINE*</td><td></td><td>*<line>*</td></tr>
<tr><td></td><td></td><td></td><td>*:BURST_SIZE*</td><td></td><td>*<size>*</td></tr>
<tr><td></td><td></td><td></td><td>*:ALM_STAT*</td><td></td><td>*<alarm>*</td></tr>
<tr><td></td><td></td><td></td><td>*:ACTION*</td><td></td><td></td></tr>
</table>

† *These commands can also be used as query commands by adding a question mark (?) after the level 2 node and omitting the second parameter. The test set will return the currently programmed value for that parameter. For example, the command* `:SOUR:DM:OH? DS2_XBIT` *would return the current state of the transmit DS2 X-bit (either 1 or 0).*

:SOURce Command Structure, continued

Root node	Level 1 node	Level 2 node	Level 3 node	Level 4 node	Parameters
:SOURce	*:DM*	*:FRAMe*	*:TYPE*†		*<signal> <frame format>*
		SCRAMBLE			*<ON | OFF>*
		:MODe			*<mode>*
		NX			*<64K | 56K>*
		:OH†			*<bit/byte> <value>*
		PROGn† *PATTERN*			*<user pattern bits>*

† *These commands can also be used as query commands by adding a question mark (?) after the level 2 node and omitting the second parameter. The test set will return the currently programmed value for that parameter. For example, the command* `:SOUR:DM:OH? DS2_XBIT` *would return the current state of the transmit DS2 X-bit (either 1 or 0).*

:SOURce:LOCAL <LOCK | UNLOCK>

This command enables or disables the LOCAL key on the CERJAC test set's front panel. When the key is locked, it cannot be used to take the test set out of SCPI (remote) mode. When the key is unlocked, the remote mode can be disabled by pressing LOCAL on the front panel.

***RST Default:** UNLOCK

Example

```
:SOUR:LOCAL LOCK
```

- Locks out the front-panel LOCAL key.

Example

```
:SOUR:LOCAL?
```

- Queries the test set for the current status of the LOCAL key.

:SOURce:DM:ALARM <*alarm type*>,<*On|Off*>

This :SOURce command activates or deactivates transmitted alarms.

***RST default:** OFF for all alarm types.

Example

Add a question mark (?) and omit the second parameter to *query* the unit for the current value. For example `:SOUR:DM:ALARm?` AIS _V.

`:SOUR:DM:ALARM AIS_V,ON`

- Sets the VT AIS alarm on.

`:SOUR:DM:ALARM RDI_P,ON`

- Sets the SONET Path remote defect indication alarm on.

The following tables list the parameters to use with this command.

SONET Alarms

:SOURce:DM:ALARM <*SONET alarm type*>,<ON|OFF>

Alarm Type	Description
AIS_L	*Line alarm indication signal (formerly :SOUR:DM:ALARM LAIS).*
RDI_L	*Line remote defect indication (formerly :SOUR:DM:ALARM LFERF).*
AIS_P	*Path alarm indication signal (formerly :SOUR:DM:ALARM PAIS).*
RDI_P	*Line remote defect indication (formerly :SOUR:DM:ALARM PYEL).*
LOP_P	*Path loss of pointer.*
STSN_LOS	*Loss of signal.*
STSN_LOF	*Loss of framing.*

VT1.5 Alarms

:SOURce:DM:ALARM <*VT alarm type*>,<ON|OFF>	
Alarm Type	**Description**
AIS_V	*VT alarm indication signal. This alarm is injected on all three STS-1s leaving the matrix switch. To inject this alarm selectively, see :SOURce:DM:EINJect:STS3:STS1PATH <source>,<status>, page 12–41. (Formerly :SOUR:DM:ALARM VT_PAIS.)*
RDI_V	*VT remote defect indication. This alarm is inserted simultaneously on both the DS3-mapped and VT-mapped STS-1s, and any fan-outs in the matrix switch. It is not set on STS-1 through-data. (Formerly :SOUR:DM:ALARM VT_PYEL.)*
VT_PLM	*Payload Label Mismatch alarm. Set simultaneously on both the DS3-mapped and VT-mapped STS-1s, and any fan-outs in the matrix switch. It is not set on STS-1 through-data.*
VT_UNEQ	*Path Unequipped alarm is simultaneously on both the DS3-mapped and VT-mapped STS-1s, and any fan-outs in the matrix switch. It is not set on STS-1 through-data.*
LOP_V	*VT loss of pointer.*

DS3 Alarms

:SOURce:DM:ALARM <*DS3 alarm type*>,<ON|OFF>	
Alarm Type	**Description**
DS3_LOS	*DS3 loss of signal.*

DS1 Alarms

:SOURce:DM:ALARM <*DS1 alarm type*>,<ON|OFF>	
Alarm Type	**Description**
DS1_YEL	*DS1 Yellow alarm.*

:SOURce:DM:ALARM <*DS1 alarm type*>,<ON|OFF>, continued	
Alarm Type	**Description**
DS1_IDLE	*DS1 Idle code.*

E1 Alarms

:SOURce:DM:ALARM <*E1 alarm type*>,<ON|OFF>	
Alarm Type	**Description**
E1_REMOTE	*E1 remote alarm indication (RAI).*
E1_MFRAME	*E1 distant multiframe alarm.*

:SOURce:DM:ATMMUX <*mapping*>

This :SOURce command sets the ATM physical layer mapping. You must set the mapping with this command before you enter the :SOUR:DM:ATMGEN command.

Note: *Do not use this command if you are also using a :SOUR:DM:DATAn ATM command.*

***RST default:** STS3C

Example

`:SOUR:DM:ATMMUX DS3`

- Sets the ATM physical layer mapping to DS3 PLCP-based mapping.

Note: *When you set the mapping with this command, the :SENS:DM:ATMDMUX automatically gets set to the same setting.*

The following table lists the parameters to use with this command:

:SOURce:DM:ATMMUX <*mapping*>

Mapping	Description
DS1	*Sets the ATM Mux to DS1 ATM mapping.*
E1	*Sets the ATM Mux for E1 ATM mapping.*
DS3	*Sets the ATM Mux to DS3 PLCP-based mapping.*
DS3_HEC	*Sets the ATM Mux to DS3 HEC-based mapping.*
STS1	*Sets the ATM Mux for STS-1 mapping.*
STS3c	*Sets the ATM Mux to STS-3c mapping.*

Also, executing the SCPI query command `:SOURce:DM:ATMMUX?` returns the setting of the ATM physical layer mapping.

:SOURce:DM:ATMGEN <*ATM param*>,<*setting*>

This :SOURce command sets the ATM Layer Generator. Before you use this command, the ATM Mux must be set to the appropriate mapping (see the :SOUR:DM:ATMMUX command on page 12–8).

Example

```
:SOUR:DM:ATMgen fgnd_distrib,continuous
```

- Sets the foreground channel distribution to continuous.

The tables on the following pages list the parameters to use with the :SOUR:DM:ATMgen command.

General ATM Setup

:SOURce:DM:ATMgen <*ATM parameter*>,<*setting*>

ATM Parameter	Setting	Description
TST_MODE	*STND**	*Sets the ATM test mode to standard.*
	XVER	*Sets the ATM test mode to cell transfer delay.*
	INTER	*Sets the ATM test mode to cell inter-arrival time.*
CELL_SCRAM	*ON* | OFF*	*Sets cell scrambinge on or off.*
FGND_BW_STEP	*10**	*Sets the foreground channel bandwidth step size to 10% with background AALs.* *Note: You must use this command along with the :SOUR:DM:ATMgen FGND_BW,<xxx> (see the next parameter description in this table).*
	2	*Sets the foreground channel bandwidth step size to 2% with no background AALs.* *Note: You must use this command along with the :SOUR:DM:ATMgen FGND_BW,<xxx> (see the next parameter description in this table).*
TRAFFIC_SHAPING	*OFF* | BASIC*	*Enables or disables ATM traffic shaping.*

** Indicates the *RST default for each source.*

ATM Foreground Channel Setup

:SOURce:DM:ATMgen <*foreground parameter*>,<*setting*>

Foreground Parameter	Setting	Description
FGND_BW	*xxx*	*Sets the foreground channel percent bandwidth control.* *If you entered the :SOUR:DM:ATMgen fgnd_bw_step,10 command, xxx can be 0, 10, 20, 30, …, 80, 90, 100.* *If you entered the :SOUR:DM:ATMgen fgnd_bw_step,2 command, xxx can be 0, 2, 4, 6, 8, 10, …, 94, 98, 100.* *For STS-12c ATM, the xxx can be 0, 10, 15, …, 90, 95, 100.*
LINE_RATE	*STS1 | DS3HEC | E1TS0 | E1TS0TS16 | DS1 | 64K | 56K*	*Sets the foreground bandwidth to a nominal line rate. Note that error code -221 is returned if your selected line rate cannot be supported by the physical setup.*
LINE_RATE_INC	*+5 | +4 | +3 | +2 | +1 | NOMINAL | -1 | -2 | -3 | -4 | -5*	*Adjusts the nominal line rate up or down in five steps.*
FGND_TYPE	*AAL0*	*Sets the foreground channel AAL type to AALO.*
	*AAL1**	*Sets the foreground channel AAL type to AAL1.*
	TST	*Sets the foreground channel type to test cell.*
FGND_SRV_TYPE	*CBR | VBR*	*Sets the service type for the foreground channel.*
FGND_PCR	*xxx*	*VBR foreground PCR percentage;* *xxx is 0, 2, 4, … 100. *RST default is 100.*
FGND_SCR	*xxx*	*VBR foreground SCR percentage;* *xxx is 0, 2, 4, … 100. *RST default is 10.*
FGND_MBS	*xxx*	*VBR foreground MBS percentage;* *xxx is 0, 1, 3, … 9. *RST default is 9.*

* *Indicates the *RST default for each source.*

:SOURce:DM:ATMgen <*foreground parameter*>,<*setting*>, continued

Foreground Parameter	Setting	Description
FGND_DISTRIB	*OFF*	*Sets the foreground channel distribution to off.*
	*CONTINUOUS**	*Sets the foreground channel distribution to continuous.*
	SINGLE	*Sets the foreground channel distribution to single.*
	PERIODIC	*Sets the foreground channel distribution to periodic.*
	ON	*For STS-12c ATM only. Activates the foreground channel.*
FGND_CELL_BURST	*xxx*	*Sets the foreground channel cells per burst. xxx can be 0 through 512 decimal.* **RST default is 100.*
FGND_BURST_PERIOD	*x.x*	*Sets the foreground channel burst period. x.x can be 0.0, 0.1, 0.2, 0.3, …, 9.7, 9.8, 9.9 seconds.* **RST default is 0.1.*
FGND_SINGLE_BURST	*(none)*	*Enable a foreground channel single burst by the ATM layer generator.* *Note: Before you use this command, you must set the ATM generator foreground channel distribution to single by using this command, :SOUR:DM:STMgen fgnd_distrib,single.*

* *Indicates the *RST default for each source.*

:SOURce:DM:ATMgen <*foreground parameter*>,<*setting*>, continued

Foreground Parameter	Setting	Description
FGND_DATA	*1'S*	*Sets the foreground channel payload data to all ones.*
	0'S	*Sets the foreground channel payload data to all zeros.*
	32BIT	*Sets the foreground channel payload to programmable.*
	2^15-1 | 2^15-1INV*	*Sets the foreground channel payload data to 2^{15}-1 PRBS (pseudo-random bit sequence)*
	2^20-1 | 2^20-1INV	*Sets the foreground channel payload to 2^{20}-1 PRBS. "INV" indicates inverted PRBS.*
	2^23-1 | 2^23-1INV	*Sets the foreground channel payload data to 2^{23}-1 PRBS*
	1010	*Sets the foreground channel payload data to 1010.*
	1100	*Sets the foreground channel payload data to1100.*
	LIVE	*"Live" traffic (no pattern).*
FGND_HDR_GFC	*x*	*Sets the foreground channel header GFC to 0 through F hex.* **RST default is 0.*
FGND_HDR_VPI	*xx*	*Sets the foreground channel header VPI to 00–FF hex.* **RST default is 01.*
FGND_HDR_VCI	*xxxx*	*Sets the foreground channel header VCI to 0000–FFFF hex.* **RST default is 0001.*
FGND_HDR_PT	*x*	*Sets the foreground channel header PT to 0 through 7.* **RST default is 0.*
FGND_HDR_CLP	*0**	*Sets the foreground channel header CLP to 0.*
	1	*Sets the foreground channel header CLP to 1.*

** Indicates the *RST default for each source.*

:SOURce:DM:ATMgen <*foreground parameter*>,<*setting*>, continued

Foreground Parameter	Setting	Description
BIT_PATT32	*xxxxxxxx*	*Sets the 32-bit pattern foreground cell payload pattern to 00000000 through FFFFFFFF hex. *RST default is 12345678.*
FGND_OAM_FLOW	*F4*	*Sets the foreground OAM flow to F4 (VPs).*
	*F5**	*Sets the foreground OAM flow to F5 (VCs).*
	F4&F5	*Sets the foreground OAM flow to both F4 and F5.*
FGND_OAM_ALARM	*AIS**	*Sets the generated OAM alarm function to AIS.*
	RDI	*Sets the generated OAM alarm function to RDI.*
	AIS&RDI	*Sets the generated OAM alarm function to both AIS & RDI.*
FGND_OAM_TYPE	*SEG*	*Sets the foreground OAM payload type to segment.*
	*END**	*Sets the foreground OAM payload type to end-to-end.*
FGND_OAM_EN	*OFF* | ON*	*Enables or disables foreground OAM generation.*

* *Indicates the *RST default for each source.*

Also, executing the SCPI query command `:SOURCe:DM:ATMGEN?< ATM PARM>` returns the setting of the ATM layer generator.

ATM Idle Cell Setup

:SOURce:DM:ATMgen <*idle cell parameter*>,<*setting*>

Idle Cell Parameter	Setting	Description
IDLE_HDR_GFC	*x*	*Sets the idle cell header GFC to 0 through F hex. *RST default is 0.*
IDLE_HDR_PT	*x*	*Sets the idle cell header PT to 0 through 7. *RST default is 0.*
IDLE_HDR_CLP	*0**	*Sets the idle cell header CLP to 0.*
	1	*Sets the idle cell header CLP to 1.*
IDLE_CELL_DATA	*xx*	*Sets the idle cell payload data to 00 through FF hex.* **RST default is 00.*

** Indicates the *RST default for each source.*

ATM Receive Channel Setup

:SOURce:DM:ATMgen <*rx channel parameter*>,<*setting*>

Rx Channel Parameter	Setting	Description
RX_HDR_VPI	*xx*	*Sets the receive channel header VPI to 00 through FF hex. *RST default is 01.*
RX_HDR_VCI	*xxxx*	*Sets the receive channel header VCI to 0000–FFFF hex. *RST default is 0001.*
RX_CELL_TYPE	*AAL0*	*Sets the receive channel AAL type to AAL0.*
	*AAL1**	*Sets the receive channel AAL type to AAL1.*
	AAL3/4	*Sets the receive channel AAL type to AAL3/4.*
	AAL5	*Sets the receive channel AAL type to AAL5.*
	TST	*Sets the receive channel type to test cell.*

** Indicates the *RST default for each source.*

ATM Background Channel Setup

:SOURce:DM:ATMgen <*background parameter*>,<*setting*>

Background Parameter	Setting	Description
BKGND_TYPE	*IDLE*	*Sets the background channels cell type to idle.*
	AAL3/4	*Sets the background channels cell type to AAL3/4.*
	*AAL5**	*Sets the background channels cell type to AAL5.*
BKGND_DATA *BKGND_DATA1* *BKGND_DATA2* *BKGND_DATA3* *BKGND_DATA4*	*1'S | 0'S | 32BIT**	*Sets the background channels 1–4 cell payloads data to all ones, all zeros, or to the 32-bit programmable pattern.* *The "DATA" command sets all channels to the same pattern; the "DATAn" commands set a single channel each.*
BKGND_BIT_PATT32 *BKGND_CH1_BIT_PATT32* *BKGND_CH2_BIT_PATT32* *BKGND_CH3_BIT_PATT32* *BKGND_CH4_BIT_PATT32*	*xxxxxxxx*	*Sets the background channels 1–4 cell payload 32-bit patterns to 00000000 through FFFFFFFF hex. *RST default is 9ABCDEF0.* *The "BIT_PATT32" command sets all channels to the same pattern; the "CHn" commands set a single channel each.*
BKGND_HDR1_GFC *BKGND_HDR2_GFC* *BKGND_HDR3_GFC* *BKGND_HDR4_GFC*	*x*	*Sets the background channel 1–4 header GFC to 0–F hex. *RST default is 0.*
BKGND_HDR1_VPI *BKGND_HDR2_VPI* *BKGND_HDR3_VPI* *BKGND_HDR4_VPI*	*xx*	*Sets the background channel 1–4 header VPI to 00–FF hex. *RST defaults are HDR1 = 02, HDR2 = 03, HDR3 = 04, HDR4 = 05.*
BKGND_HDR1_VCI *BKGND_HDR2_VCI* *BKGND_HDR3_VCI* *BKGND_HDR4_VCI*	*xxxx*	*Sets background channel 1–4 header VCI to 0000–FFFF hex. *RST default is 0001 for all channels.*
BKGND_HDR1_PT *BKGND_HDR2_PT* *BKGND_HDR3_PT* *BKGND_HDR4_PT*	*x*	*Sets the background channel 1–4 header PT to 0 through 7. *RST default is 0.*

** Indicates the *RST default for each source.*

:SOURce:DM:ATMgen <*background parameter*>,<*setting*>, continued

Background Parameter	Setting	Description
BKGND_HDR1_CLP *BKGND_HDR2_CLP* *BKGND_HDR3_CLP* *BKGND_HDR4_CLP*	*0* | 1*	*Sets the background channel 1–4 header CLP to 0 or 1.*
BKGND_SEND_EN1 *BKGND_SEND_EN2* *BKGND_SEND_EN3* *BKGND_SEND_EN4*	*OFF* | ON*	*Disables or enables ATM background channel 1–4.* *Note: You must set the ATM foreground channel bandwidth step size to 10% with background AALs using the :SOUR:DM:ATMgen FGND_BW_STEP,10 command.*
BKGND_SRV_TYPE1 *BKGND_SRV_TYPE2* *BKGND_SRV_TYPE3* *BKGND_SRV_TYPE4*	*CBR* | VBR*	*Sets the background channel 1–4 service type to either CBR or VBR.*
BKGND_AAL_TYPE1 *BKGND_AAL_TYPE2* *BKGND_AAL_TYPE3* *BKGND_AAL_TYPE4*	*AAL3/4 | AAL5**	*Sets the background channel 1–4 AAL type to either AAL-3/4 or AAL-5.*
BKGND_OAM_FLOW	*F4*	*Sets the background OAM flow to F4 (VPs).*
	*F5**	*Sets the background OAM flow to F5 (VCs).*
	F4&F5	*Sets the background OAM flow to both F4 and F5.*
BKGND_OAM_ALARM	*AIS**	*Sets the generated OAM alarm function to AIS.*
	RDI	*Sets the generated OAM alarm function to RDI.*
	AIS&RDI	*Sets the generated OAM alarm function to both AIS & RDI.*
BKGND_OAM_TYPE	*SEG*	*Sets the background OAM payload type to segment.*
	*END**	*Sets the background OAM payload type to end-to-end.*
BKGND_OAM_EN	*OFF* | ON*	*Enables or disables background OAM generation.*

** Indicates the *RST default for each source.*

ATM Error Injection

:SOURce:DM:ATMgen <*ATM error parameter*>,<*setting*>

ATM Error Parameter	Setting	Description
HCS_BYTE_CONT	*OFF* | ON*	*Sets continuous HEC error injection off or on.* *Note: You must set the ATM Mux to STS3C using the :SENS:DM:ATMDMUX STS3C command (see page 11–3).*
HCS_BYTE	*OFF* | SINGLE | 6_CONSEC | 7_CONSEC | 8_CONSEC*	*For STS-12c ATM only. Injects HEC byte errors.*
HCS_BIT_CONT	*OFF* | ON*	*Sets continuous HEC error injection off or on (DS3 only).*
HCS_BIT	*OFF* | SINGLE | 6_CONSEC | 7_CONSEC | 8_CONSEC*	*For STS-12c ATM only. Injects HEC bit errors.*
PLCP_B1_CONT	*OFF* | ON*	*Sets continuous PLCP B1 error injection off or on.* *Note: You must set the ATM Mux to DS3 mapping using the :SOUR:DM:ATMDMUX DS3 command (see page 12–8).*
PLCP_FEBE_CONT	*OFF* | ON*	*Sets continuous PLCP FEBE error injection off or on.* *Note: You must set the ATM Mux to DS3 mapping using the :SENS:DM:ATMDMUX DS3 command (see page 11–3).*
PLCP_A1A2_CONT	*OFF* | ON*	*Sets continuous PLCP A1 or A2 error injection off or on.* *Note: You must set the ATM Mux to DS3 mapping using the :SOUR:DM:ATMDMUX DS3 command (see page 11–3).*
PLCP_POI_CONT	*OFF* | ON*	*Sets continuous PLCP POI error injection off or on.* *Note: You must set the ATM Mux to DS3 mapping using the :SOUR:DM:ATMDMUX DS3 command (see page 11–3).*
PLCP_YEL_ALM	*OFF* | ON*	*Turns on or off continuous PLCP yellow alarm injection.*

** Indicates the *RST default for each source.*

:SOURce:DM:ATMgen <*ATM error parameter*>,<*setting*>, continued

ATM Error Parameter	Setting	Description
LOSS_OF_CELL_SINGLE	*(none)*	*Forces a single dropped cell.* *Note: You must set the ATMgen foreground channel type to AAL1 using :SOUR:DM:ATMgen fgnd_type,AAL1.*
SN_CRC_PAR_SINGLE	*(none)*	*Injects a single sequence number CRC/parity error.* *Note: You must set the ATMgen foreground channel type to AAL1 using :SOUR:DM:ATMgen fgnd_type,AAL1.*
PYLD_SINGLE	*(none)*	*Injects a single payload bit error.*
PYLD_10E_3	*OFF* | ON*	*Sets payload bit error injection at the rate of 10E-3 off or on.*
PYLD_10E_6	*OFF* | ON*	*Sets payload bit error injection at the rate of 10E-6 off or on.*
PYLD_10E_N	*OFF* | ON*	*STS-12c ATM only. Injects a payload bit error at a rate of 10E-n where n is 2 through 9.*
MISINSERT_PERIOD	*x.x*	*Defines the cell misinsert period.Each x can be set from 0 through 9. Default is 0.1*
MISINSERT_DATA	*1'S | 0'S | 32BIT*	*Sets the misinsert cell payload to all-ones, all-zeros, or the user-programmed 32-bit misinsert pattern.*
MISINSERT_PATT32	*xxxxxxxx*	*The 32-bit user-programmed misinsert pattern. Each x can be set from 0 through F (hex). Default is ABCDEF01.*
MISINSERT_CELL_SINGLE	*(none)*	*Inserts a single misinserted cell as defined by the MISINSERT commands above.*
MISINSERT_CELL_PERIODIC	*ON | OFF*	*Injects misinserted cells periodically.*

** Indicates the *RST default for each source.*

:SOURce:DM:CLOCK<*n*> <*timing source*>

:SOURCE <1, 2, 3>

This :SOURce command sets the source for the transmit timing (STS-N, DS3, or DS1) and the type of timing.

***RST default:** INT

Example

```
:SOUR:DM:CLOCK1 INT
```

- Sets the DS1 timing source to internal.

The following tables lists the parameters to use with this command:

:SOURce:DM:CLOCK*n* (*n = Signal Type*) Parameters

Signal Type	Timing Source	Description
:CLOCK1	*INT | EXT | LOOP | LOOP_DS3 | LOOP_RX | LOOP_VT | REF*	*DS1*
:CLOCK2	*INT | EXT | LOOP | BITS*	*STS-N*
:CLOCK3	*INT | EXT | sonet LOOP | DS3 LOOP_RX | DS3 LOOP_DROP*	*DS3*

:SOURce:DM:CLOCK*n* <*timing source*> Parameters

Timing Source	Description
INT	*Internal*
EXT	*External*
REF	*Reference input.*
LOOP	*Loop timed for receive signal.*
BITS	*Derived from the input BITS signal at the rear panel.*

:SOURce:DM:CODE1 <*line code*>

This :SOURce command sets the line code for the DS1 Bipolar Receive, the DS1 Bipolar Transmit, and the DS1 Bipolar Drop ports to either AMI or B8ZS.

***RST default:** AMI

Example

```
:SOUR:DM:CODE1 AMI
```

- Sets the DS1 line code to AMI.

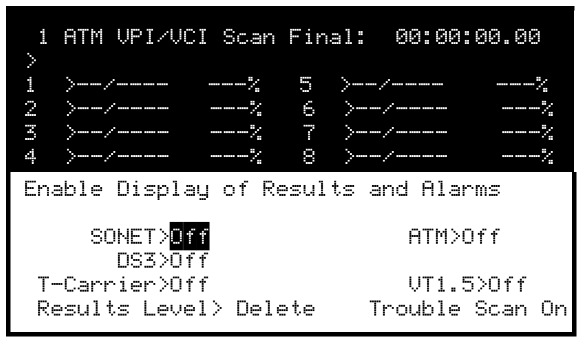
The following table lists the parameters to use with this command:

:SOURce:DM:CODE1 <*line code*> Parameters

Line Code	Description
AMI	*Alternate mark inversion.*
B8ZS	*Bipolar with eight-zero substitution.*
HDB3	*High-density with three-zero substitution.*

TM VPI/CPI Scan Capability Enhancement

With Release 7.0 or higher, SCPI mode now supports an ATM VPI/CPI (Virtual Path Identifier/Virtual Channel Identifier) scan. RST. Issue the appropriate SCPI commane the set the desired rate and with an ATM payload.



Perform the VPI/CPI scan using the following sequence of commands from the table below:

ATM VPI/CPI Scan CapabilityTable

SCPI Command	Command Description
:SYSTEM:SCAN:MODE CHANNEL	Select VPI/VCI scan channel search mode.
:SYSTEM:SCAN:STATUS	Return channel search scan status.
:SYSTEM:SCAN:START	Start the channel search scan, observe front panel.
:ABORT	Stop scan when all channel are recognized.

ATM VPI/CPI Scan CapabilityTable

SCPI Command	Command Description
:SOUR:DM:ATMGEN DECILAM	Set format of VPI/VCI channel results to decimal.
:SOUR:DM:ATMGEN HEX	Set format of VPI/VCI channel results to hexa-decimal.
:SYSTEM:SCAN:CHANNEL? n	Fetch VPI/VCI scan results for channel n.
:SYSTEM:SCAN:MODE BANDWIDTH	Select VPI/VCI scan bandwidth measurement mode.
:SYSTEM:SCAN:START	Start the bandwidth scan, observe the front panel.
:SYSTEM:SCAN:STATUS	Return bandwidth scan status.
SYSTEM:SCAN:BANDWIDTH? n	Fetch VPI/VCI bandwidth results for channel “n”.
:SYSTEM:SCAN:SELECT n	Select Channel “n”for further measurement.
:SYSTEM:SCAN:SELECT?	Quarry last channel selected for further measurement.
:SYSTEM:SCAN:MODE EXIT	Exit VPI/VCI scan mode and return to ordinary SCPI mode

:SOURce:DM:DATA0 <*pattern*>

This :SOURce command identifies the transmit payload when DS0 is the source.

***RST default:** 1004HZ

Example

Add a question mark (?) to *query* the unit for the current value. For example `:SOUR:DM:DATA0?`.

```
:SOUR:DM:DATA0 1012HZ
```

- Sets the DS0 transmit tone to 1012 Hertz.

The following table lists the parameters to use with this command:

:SOURce:DM:DATA0 <*pattern*> Parameters

Pattern	Description
1004HZ | 1012HZ | 1020HZ | 2010HZ	*1004, 1012, 1020, or 2010 Hertz tones.*
PROG	*User-Programmable pattern.*
QRSS | QRSSINV	*Quasi-random signal.*
2^6-1 | 2^6-1INV	2^{6}*-1 PRBS (pseudo-random bit sequence); INV = inverted*
2^9-1 | 2^9-1INV	2^{9}*-1 PRBS; INV = inverted PRBS.*
2^11-1 | 2^11-1INV	2^{11}*-1 PRBS (pseudo-random bit sequence)*
2^15-1 | 2^15-1INV	2^{15}*-1 PRBS*
2^20-1 | 2^20-1INV	2^{20}*-1 PRBS*
2^23-1 | 2^23-1INV	2^{23}*-1 PRBS*
55OCTET	*55-octet pattern; also called Daly pattern.*
ONES | ZEROS | ALT	*All-ones, all-zeros, or alternating ones and zeros pattern.*
1IN8 | 2IN8 | 3IN24	*1-in-8 pattern, and so forth.*
EXTVF	*External VF signal.*

:SOURce:DM:DATA1 <*pattern*>

This :SOURce command sets the DS1 transmit pattern.

***RST default:** QRSS

Example

Add a question mark (?) to *query* the unit for the current value. For example `:SOUR:DM:DATA1?`.

```
:SOUR:DM:DATA1 AIS
```

- Sets the DS1 transmit pattern to AIS.

The following table lists the parameters to use with this command:

:SOURce:DM:DATA1 <*pattern*> Parameters

Pattern	Description
AIS	*Alarm Indication Signal.*
PROG	*User-Programmable pattern.*
QRSS I QRSSINV	*Quasi-random signal; "INV" = inverted.*
2^6-1 | 2^6-1INV	2^6*-1 PRBS (pseudo-random bit sequence).*
2^9-1 | 2^9-1INV	2^9*-1 PRBS.*
2^11-1 | 2^11-1INV	2^{11}*-1 PRBS (pseudo-random bit sequence).*
2^15-1 | 2^15-1INV	2^{15}*-1 PRBS.*
2^20-1 | 2^20-1INV	2^{20}*-1 PRBS.*
2^23-1 | 2^23-1INV	2^{23}*-1 PRBS.*
55OCTET	*55 octet pattern.*
ONES | ZEROS | ALT	*All-ones, all-zeros, or alternating ones and zeros.*
1IN8 | 2IN8 | 3IN24	*1-in-8 pattern, an so on.*
LIVE	*"Live" traffic (no pattern).*
PASSTHRU	*Data pattern pass-through.*
ATM	*ATM traffic. Do not use this command if you are using the :ROUT:SEL:DS1_TX,ATM or :SOUR:DM:ATMMUX DS1 command.*

:SOURce:DM:DATA1 <*pattern*> Parameters

Pattern	Description
EXT	

:SOURce:DM:DATA3 <*pattern*>

This :SOURce command sets the DS3 data pattern.

***RST default:** 2^15-1

Example

Add a question mark (?) to *query* the unit for the current value. For example `:SOUR:DM:DATA3?`.

`:SOUR:DM:DATA3 IDLE`

- Sets the DS3 transmit pattern to idle.

The following table lists the parameters to use with this command:

:SOURce:DM:DATA3 <*pattern*> Parameters

Pattern	Description
IDLE	*Idle.*
AIS	*Alarm Indication Signal.*
PROG	*User-Programmable pattern.*
EXT	*External.*
LIVE	*:Live” traffic (no pattern).*
ONES	*All ones pattern.*
2^23-1 | 2^23-1INV	2^{23}*-1 PRBS (pseudo-random bit sequence).* *The “INV” voice inverts the PRBS pattern.*
2^20-1 | 2^20-1INV	2^{20}*-1 PRBS.*
2^15-1 | 2^15-1INV	2^{15}*-1 PRBS.*
ATM_HEC *ATM_PLCP*	*ATM traffic. Do not use this command if you are using the :ROUT:SEL:DS3_TX,ATM or :SOUR:DM:ATMMUX DS3 command.*

:SOURce:DM:DATA4 <*pattern*>

This :SOURce command sets the STS-3c data pattern.

***RST default:** 2^15-1

Example

Add a question mark (?) to *query* the unit for the current value. For example `:SOUR:DM:DATA4?`.

```
:SOUR:DM:DATA4 2^20-1INV
```

- Sets the STS-3c data pattern for an inverted 2^{20}–1 PRBS.

The following table lists the patterns to use with this command.

:SOURce:DM:DATA4 <*pattern*> Parameters

Pattern	Description
2^15-1	*2^{15}-1 PRBS (pseudo-random bit sequence)*
2^20-1	*2^{20}-1 PRBS*
2^23-1	*2^{23}-1 PRBS*
2^15-1INV	*Inverted 2^{15}-1 PRBS*
2^20-1INV	*Inverted 2^{20}-1 PRBS*
2^23-1INV	*Inverted 2^{23}-1 PRBS*
ONES	*All-ones pattern*
ZEROS	*All-zeros pattern*
LIVE	*"Live" traffic (no pattern)*
ATM	*ATM traffic. Do not use this command if you are using the :ROUT:MATRIX STS3C,ATM or :SOUR:DM:ATMMUX STS3C command.*

:SOURce:DM:DATA5 <*pattern*>

This :SOURce command sets the STS-12c payload pattern.

***RST default:** 2^15-1

Example

Add a question mark (?) to *query* the unit for the current value. For example `:SOUR:DM:DATA5?`.

`:SOUR:DM:DATA5 2^20-1INV`

- Sets the STS-12c data pattern for an inverted 2^{20}–1 PRBS.

The following table lists the patterns to use with this command.

:SOURce:DM:DATA5 <*pattern*> Parameters

Pattern	Description
2^15-1	*2^{15}-1 PRBS (pseudo-random bit sequence)*
2^20-1	*2^{20}-1 PRBS*
2^23-1	*2^{23}-1 PRBS*
2^31-1	*2^{31}-1 PRBS*
2^15-1INV	*Inverted 2^{15}-1 PRBS*
2^20-1INV	*Inverted 2^{20}-1 PRBS*
2^23-1INV	*Inverted 2^{23}-1 PRBS*
2^31-1INV	*Inverted 2^{31}-1 PRBS*
ONES	*All-ones pattern*
ZEROS	*All-zeros pattern*
PROG32	*The user-programmable pattern defined by the :SOURce:DM:PROG32 PATTERN command (see page 12–63).*
LIVE	*“Live” traffic (no pattern)*

:SOURce:DM:DL_DATA <*FDL message*>

This :SOURce command programs the transmitted DS1 extended superframe (ESF) facility datalink (FDL) message.

***RST default:** 0111 1110 0111 1110

Example

Add a question mark (?) to *query* the unit for the current value. For example `:SOUR:DM:DL_DATA?`.

`:SOUR:DM:DL_DATA 101101`

- Sets the transmitted DS1 ESF FDL message to the following bit sequence: 0**101 101**0 0111 1110.
  (Note that only bits 2 through 7 are user-programmable.)

The following table lists the patterns to use with this command.

:SOURce:DM:DL_DATA <*FDL message*> Parameters

Pattern	Description
xxxxxx	*xxxxxx is the user-programmable bits sequence. Each "x" can be either a binary 1 or 0. The code is transmitted in the format "0xxxxxx0 11111111".*
ONES	*Set the FDL message to all-ones.*
IDLE	*Sets the FDL message to "idle."*

:SOURce:DM:<*DS1 loopback*> <*value*>

Add a question mark (?) to *query* the unit for the current value. For example `:SOUR:DM:LPUP_CODE?`.

These :SOURce commands configure the DS1 loopback codes. To transmit the loop codes after you have configured them, use :SOUR:DM:EINJ:DS1 LOOPUP or LOOPDN (see page 12–34).

***RST defaults:** See table.

Example

`:SOUR:DM:LPUP_CODE PROG`

- Sets the loop up code type to user-programmed in-band codes.

Example

`:SOUR:DM:LPUP_PROG_TX 111000`

- Sets the user-programmable in-band loop up code to “111000”.

The following table lists the parameters to use for these commands.

:SOURce:DM:<*DS1 loopback*> <*value*>

Loopback	Value	Description
LP_FRM_OVR	*ON | OFF*	*This command turns on/off the framing overwrite. Note, for in-band only. Default is OFF*
LPUP_CODE *LPDN_CODE*	*These commands select the loop code type. Default is LINE. An error code -221 will return if code is not set properly. Note: “LPUP” = loop up; “LPDN” = loop down.*	
	LINE	*In-band line loopback codes.*
	NTWK	*In-band network loopback codes.*
	PROG	*In-band user-programmable loopback code.*
	4BIT	*In-band (only) four-bit loopback codes.*
	OUTB_LINE	*Out-of-band line loopback codes.*
	OUTB_NTWK	*Out-of-band network loopback codes.*
	OUTB_PROG	*Out-of-band user-programmable loopback codes.*
	OUTB_PYLD	*Out-of-band payload loopback codes.*

:SOURce:DM:<*DS1 loopback* > <*value*> , continued

Loopback	**Value**	**Description**
LPUP_PROG_TX *LPDN_PROG_TX*	*xxxxxxxxxxxxxxxx* *(16-bit binary value)*	*Set the transmit in-band user-programmable loop-back codes. Each “x” can be set to 1 or 0. Default is 10000.*
LPUP_PROG_RX *LPDN_PROG_RX*	*xxxxxxxxxxxxxxxx* *(16-bit binary value)*	*Set the receive in-band user-programmable loop-back codes. Each “x” can be set to 1 or 0. Default is 10000.*
LPUP_DL_OUTB_PROG *LPDN_DL_OUTB_PROG*	*xxxxxx* *(6-bit binary value)*	*Set the user-programmed out-of-band loopback codes. Each “x” can be set to 1 or 0. Default is 111111*

:SOURce:DM:EINJect:DS0 DATA,<*rate*>

This :SOURce command allows you to insert logic errors in the DS0 signal.

***RST default:** Rate = OFF

Example

Add a question mark (?) and omit the second parameter to *query* the unit for the current value. For example `:SOUR:DM:EINJ`**?** `DS0`.

```
:SOUR:DM:EINJ:DS0 DATA,SINGLE
```

- Injects a BPV error into a DS0 signal.

The following table lists the parameters to use with this command:

:SOURce:DM:EINJect:DS0 DATA,<*rate*> Parameters

Rate	Description
SINGLE	*A single error is injected.*
1E-2 through 1E-9	*A steady error rate is started at the indicated rate.*
BURST	*Enables control via the rear panel Burst input. The error is sent for all valid bit periods when the burst input is at a logic high level.*
OFF	*Disables error injection.*

:SOURce:DM:EINJect:DS1 *<err type>,<rate>*

This :SOURce command allows you to insert logic errors in the DS1 signal.

***RST default:** Rate = OFF

Example

`:SOUR:DM:EINJ:DS1 BPV,SINGLE`

- Injects a BPV error into a DS1 signal.

Add a question mark (?) and omit the parameters to *query* the unit for the current value. For example `:SOUR:DM:EINJ? DS1`.

The following table lists the parameters to use with this command:

:SOURce:DM:DS1 <*error type*>,<*rate*> Parameters

Error Type	Applicable Rates	Description
BPV	*SINGLE | 1E-2 to 1E-9 | BURST | OFF*	*Bipolar Violation.*
DATA	*SINGLE | 1E-2 to 1E-9 | BURST | OFF*	*Data errors cause the signal to be errored prior to calculation of the CRC.*
DAT_CRC	*SINGLE | 1E-2 to 1E-9 | BURST | OFF*	*For combined Data and CRC errors, the signal is errored after calculation of the CRC.*
FRAME	*SINGLE | 1E-2 to 1E-9 | BURST | OFF*	*Selection of DS1 Frame errors causes the frame bits to be errored.*
LOOPUP	*OFF | 1S | 2S | 4S | 8S | 16S | CONTINUOUS | UNTIL_RCVD*	*Sends the currently configured loop up code (see page 12–31).*
LOOPDN	*OFF | 1S | 2S | 4S | 8S | 16S | CONTINUOUS | UNTIL_RCVD*	*Sends the currently configured loop down code (see page 12–31).*

:SOURce:DM:EINJect:DS2 <*err type*>,<*rate*>

This :SOURce command allows you to inject DS2 logic errors when the DS3 signal is composed of DS1 sub-channels.

***RST default:** Rate = OFF

Example

Add a question mark (?) and omit the parameters to *query* the unit for the current value. For example `:SOUR:DM:EINJ? DS2`.

`:SOUR:DM:EINJ:D21 C2,SINGLE`

- Injects a C2 bit error into a DS2 signal.

The following table lists the parameters to use with this command:

:SOURce:DM:EINJ:DS2 <*error type*>,<*rate*> Parameters

Error Type	Applicable Rates	Description
C1	*SINGLE* *1E-2 through 1E-9* *BURST* *OFF*	*The DS2 C1 bits are errored at the selected rate.*
C2	*SINGLE* *1E-2 through 1E-9* *BURST* *OFF*	*The DS2 C2 bits are errored at the selected rate.*
C3	*SINGLE* *1E-2 through 1E-9* *BURST* *OFF*	*The DS2 C3 bits are errored at the selected rate.*
C_ALL	*SINGLE* *1E-2 through 1E-9* *BURST* *OFF*	*Errors at the selected rate are injected into any or all of the DS2 C-bits.*

:SOUR:DM:EINJ:DS3<B> *<err type>*,*<rate>*

This :SOURce command inserts errors on the DS3 signal. For dual DS3 applications the level 3 node :DS3 inserts errors on DS3-A; :DS3B inserts errors on DS3-B. For both commands, error types and rates are the same.

***RST default:** Rate = OFF

Example

Add a question mark (?) and omit the parameters to *query* the unit for the current value. For example `:SOUR:DM:EINJ?` `DS3`.

`:SOUR:DM:EINJ:DS3 DATA,BURST`

- Injects a data error into the DS3 signal when triggered by a high level at the rear-panel BURST ERR IN jack.

`:SOUR:DM:EINJ:DS3B FRAME,SINGLE`

- Injects a single frame error on DS3-B.

The following table lists the parameters to use with this command:

:SOURce:DM:EINJect:DS3 <*error type.*,<*rate*> and
:SOURce:DM:EINJect:DS3B <*error type.*,<*rate*>

Error Type	Rates	Description
LOGIC	*SINGLE, 1E-2 to 1E-9, BURST, OFF*	*Logic error.*
BPV	*SINGLE, 1E-2 to 1E-9, BURST, OFF*	*Bipolar violation.*
DATA	*SINGLE, 1E-2 to 1E-9, BURST, OFF*	*Data error.*
DAT_PBIT	*SINGLE, 1E-2 to 1E-9, BURST, OFF*	*Data Parity Bit error.*
FRAME	*SINGLE, 1E-2 to 1E-9, BURST, OFF*	*Frame error (F1/F0 bit errors).*
C1	*SINGLE, 1E-2 to 1E-9, BURST, OFF*	*C1 bit error.*
C2	*SINGLE, 1E-2 to 1E-9, BURST, OFF*	*C2 bit error.*
C3	*SINGLE, 1E-2 to 1E-9, BURST, OFF*	*C3 bit error.*
C_ALL	*SINGLE, 1E-2 to 1E-9, BURST, OFF*	*Errors are injected in all of the C-bits.*

:SOURce:DM:EINJect:E1 *<error type>*,*<rate>*

This :SOURce command allows you to insert logic errors in the E1 signal.

***RST default:** Rate = OFF

Example

```
:SOUR:DM:EINJ:E1 BPV,SINGLE
```

- Injects a BPV error into a E1 signal.

Add a question mark (?) and omit the parameters to *query* the unit for the current value. For example `:SOUR:DM:EINJ?E1`.

The following table lists the parameters to use with this command:

:SOURce:DM:EINJect:E1 <*error type*>,<*rate*>

Error Type	Applicable Rates	Description
BPV	*SINGLE, 1E-2 to 1E-9, BURST, OFF*	*Bipolar Violation.*
DATA	*SINGLE, 1E-2 to 1E-9, BURST, OFF*	*Data errors. The signal is errored before calculation of the CRC.*
DAT_CRC	*SINGLE, 1E-2 to 1E-9, BURST, OFF*	*Data and CRC errors. The signal is errored after calculation of the CRC.*
FRAME_CRC	*SINGLE, 1E-2 to 1E-9, BURST, OFF*	*Frame and CRC errors.*
MFRAME_CRC	*SINGLE, 1E-2 to 1E-9, BURST, OFF*	*Multiframe and CRC errors.*

:SOUR:DM:EINJect:VT15 <*error type*>,<*rate*>

These :SOURce command allow the insertion of VT1.5 errors and alarms, and return error and alarms results.

***RST default:** Rate = OFF

Example

`:SOUR:DM:EINJ:VT15 BER,SINGLE`

- Injects a bit error in the VT1.5 payload.

Example

`:SOUR:DM:EINJ:VT15 PNTR,8_CONSEC`

- Injects an invalid pointer into eight consecutive VT super frames.

Add a question mark (?) and omit the parameters to *query* the unit for the current values. For example `:SOUR:DM:EINJ? VT15`.

The following table lists the parameters to use with these commands:

:SOURce:DM:EINJect:VT15 <*error type*>,<*rate*>

Error Type	Applicable Rates	Description
REI_V	*SINGLE | 1E-2 through 1E-4 | OFF*	*VT1.5 remote event indication alarms.*
BER	*SINGLE | 1E-2 through 1E- | BURST, OFF*	*Bit errors.*
PNTR	*7_CONSEC | 8_CONSEC | 9_CONSEC | CONTINUOUS OFF*	*VT pointer errors.*
FEBE	*SINGLE | 1E-2 through 1E-8 | OFF*	*VT1.5 Far-end block errors.*

:SOUR:DM:EINJ:STS1*<#n> <err type>*,*<rate>*

This command injects STS-1 errors. The error type and rate are selectable. For STS-3 signals, the individual STS-1 can be identified for path BER and pointer error injection.

SONET errors and alarms listed below are injected on the outgoing STS-1 electrical signal, and at the same time (when appropriate) on the active optical transmit signal, whether it be OC-1, OC-3, or OC-12.

***RST default:** Rate = OFF

Example

```
:SOUR:DM:EINJ:STS1#1 BPV,SINGLE
```

- Injects a single BPV error in the first STS-1 of an STS-3 signal.

Add a question mark (?) and omit the parameters to *query* the unit for the current values. For example `:SOUR:DM:EINJ? STS1#2`.

The STS-1 parameter can be entered as follows:

STS1: Errors are injected on all STS-1 transmit paths.

STS1#*n*: Where *n* can be **1**, **2**, or **3**. Errors are injected on the specified STS-1 of an STS-3 signal. This applies for path BER and pointer errors only.

The following table lists the error type and rate parameters :

:SOURce:DM:EINJect:STS1<#*n*> <*error type*>,<*rate*>

Error Type	Applicable Rates	Description
BPV	*SINGLE | 1E-2 through 1E-9 | BURST | OFF*	*Inserts a Bipolar Violation in the outgoing STS-1 electrical signal.*
SECT_BER	*SINGLE | 1E-2 through 1E-9 | BURST | OFF*	*Inserts a Section bit error.*
LINE_BER	*SINGLE | 1E-2 through 1E-9 | BURST | OFF*	*Inserts a Line bit error.*
PATH_BER	*SINGLE | 1E-2 through 1E-9 | BURST | OFF*	*Inserts an STS-1 path bit error in all three transmit paths (unless STS1#n parameter is used) leaving the STS-1 matrix switch, including through paths.*

:SOURce:DM:EINJect:STS1<#*n*> <*error type*>,<*rate*> , continued

Error Type	Applicable Rates	Description
REI_P	*SINGLE | 1E-4 through 1E-8 | OFF*	*Path remote event indication (formerly :SOUR:DM:EINJ:STS1 PATH_FEBE).*
REI_L	*SINGLE | 1E-4 through 1E-8 | OFF*	*Line remote event indication (formerly :SOUR:DM:EINJ:STS1 LINE_FEBE).*
B1	*SINGLE or OFF*	*Results in an inversion of the value of the B1 byte in one frame.* *Note: B1 byte inversion is not available at OC-12-OC-12 C work.*
B2	*SINGLE or OFF*	*Results in an inversion of the value of the B2 byte in one frame.*
B3	*SINGLE or OFF*	*Results in an inversion of the value of the B3 byte in one frame. It is inserted on the DS3-mapped STS-1 unless the matrix switch has been set for VT15Drop.*
A1A2	*3_CONSEC | 4_CONSEC | 5_CONSEC | 23_CONSEC | 24_CONSEC through 30_CONSEC | CONTINUOUS | BURST | OFF*	*Results in an invalid frame word being transmitted in a selectable number of consecutive frames.*
PNTR	*7_CONSEC | 8_CONSEC | 9_CONSEC | CONTINUOUS | BURST | OFF*	*Results in an invalid STS1 pointer value being inserted in a selectable number of consecutive frames. Pointer errors are inserted on all three STS-1 transmit paths (unless STS1#n parameter is used) leaving the matrix switch, including through paths.*

Note: *STS-1 pointer errors (PNTR) and frame errors (A1A2) cannot be inserted simultaneously. If pointer errors are active, selecting frame errors disables the pointer errors and starts the frame errors. Similarly, selecting pointer errors while frame errors are active stops the A1/A2 errors and starts the pointer errors. Setting either type "off" disables whichever is error type is currently active.*

:SOURce:DM:EINJect:STS3:STS1PATH <*source*>,<*status*>

This :SOURce command sets selected STS-1 channels leaving the matrix switch to Unequipped or Path AIS.

***RST default:** Rate = Norm

Example

Add a question mark (?) and omit the parameters to *query* the unit for the current values. `:SOUR:DM:EINJ?` `STS3STS1PATH`.

`:SOUR:DM:EINJ:STS3:STS1PATH 1,UNEQ`

- Sets STS-1 #1in the STS-3 to unequipped.

The following table lists the parameters to use with this command:

:SOURce:DM:EINJect:STS3:STS1PATH <*source*>,<*status*>

Sources		Status	Description
1	*STS-1 #1*	*NORM*	*Normal.*
2	*STS-1 #2*	*UNEQ*	*Unequipped.*
3	*STS-1 #3*	*PAIS*	*Path Alarm Indication Signal*

:SOUR:DM:EINJ:STS3C <*error type*>,<*rate*> :SOUR:DM:EINJ:STS12C <*error type*>,<*rate*>

These two :SOURce commands inject errors on the transmit STS-3c or STS-12c signals.

***RST default:** Rate = OFF

Example

Add a question mark (?) and omit the parameters to *query* the unit for the current values. For example `:SOUR:DM:EINJ?STS3C`.

```
:SOUR:DM:EINJ:STS3C FEBE,SINGLE
```

- Injects a single FEBE error on the transmit STS-3c signal.

```
:SOUR:DM:EINJ:STS12C DATA,SINGLE
```

- Injects a single payload pattern bit error on the transmit STS-12c signal.

The following table lists the error type and rate parameters to use with these commands.

:SOURce:DM:EINJect:STS3C <*error type*>,<*rate*> and :SOURce:DM:EINJect:STS12C <*error type*>,<*rate*>

Error Type	Applicable Rates	Description
DATA	*SINGLE,1E-2 through 1E-9 | BURST | OFF*	*Injects bit errors in the STS-3c data pattern.*
REI_P	*SINGLE | 1E-4 through 1E-8 | OFF*	*Injects Path remote event indication alarms using the STS-3c or STS-12c G1 byte (formerly :SOUR:DM:EINJ:STS3C FEBE).*

:SOURce:DM:EINJect:STS12 *<OH bit>*,*<rate>*

This :SOURce command injects errors into the OC-12 signal.

***RST default:** Rate = OFF

Example

Add a question mark (?) and omit the parameters to *query* the unit for the current values. For example `:SOUR:DM:EINJ?STS12`.

```
:SOUR:DM:EINJ:STS12 B1_BIT,CONTINUOUS
```

- Injects B1 byte bit errors continuously on the OC-12 signal.

The following table lists the parameters to use with this command:

:SOURce:DM:EINJect:STS12 <*overhead bit*>,<*error rate*>

Overhead Bit	Description
B1_BIT	*Forces an inversion of the least significant bit of B1.*
B2_BIT	*Forces an inversion of the least significant bit of B2.*
A1_BIT	*Inverts the least significant bit of the A1 framing byte.*
Rate	
Each overhead bit above can be set to either of the following:	
CONTINUOUS	*Activates continuous injection of the overhead bit error.*
OFF	*Deactivates injection of the overhead bit error.*

:SOUR:DM:EINJ:STS1 *xxx*_PNTR_ADJ,<*rate*>

Pointer adjustments are injected simultaneously on the STS-1s exiting the STS-1 matrix switch (at the “Path, BER, H-pointer, and Error Insertion” element; see *Transmit Programming Diagram*, page 18–5). Pointer adjustments do not affect drop or pass channels.

***RST default:** Rate = OFF

Positive Pointer Adjustments

Positive pointer adjustments are injected as follows:

`SOURce:DM:EINJect:STS1 POS_PNTR_ADJ,`<*rate*>

Where <*rate*> can be any value listed in the table on the next page.

Negative Pointer Adjustments

Negative pointer adjustments are introduced as follows:

`SOURce:DM:EINJect:STS1 NEG_PNTR_ADJ,`<*rate*>

Where <*rate*> can be any value listed in the table on the next page.

STS-1 Pointer New Data Flag

STS-1 pointer New Data Flag events are created as follows:

`SOURce:DM:EINJect:STS1 NDF_PNTR_ADJ,SINGLE`

Only one NDF event can be injected at a time.

INIT Selection Rate

Running the INIT, preconditions the system for pointer adjustments.

Pointer Adjustment rates

The value for <rate> indicates the repetition frequency of the pointer adjustments, and may be set as described in the following table:

:SOURce:DM:EINJect:STS1 *xxx*_PNTR_ADJ,<*rate*>
(pointer adjustment rate parameters)

Select/Rate	Description
INIT	*Init preconditions the system's elastic store to minimize any delays in responding to subsequent pointer adjustment commands. When the init rate is used with these commands, a short burst of pointer adjustments occurs.*
OFF	*Terminates pointer adjustment activity.*
SINGLE	*A single pointer adjustment.*
11MS	*11 milliseconds.*
32MS	*32 milliseconds.*
40MS	*40 milliseconds.*
53MS	*53 milliseconds.*
80MS	*80 milliseconds.*
91MS	*91 milliseconds.*
106MS	*106 milliseconds.*
128MS	*128 milliseconds.*
1S	*1 second.*
3S	*3 seconds.*
4S	*4 seconds.*
5S	*5 seconds.*
8S	*8 seconds.*
9S	*9 seconds.*

:SOURce:DM:EINJect:STS1 *xxx*_PNTR_ADJ,<*rate*> **(pointer adjustment rate parameters), continued**	
Select/Rate	**Description**
10.6S	*10.6 seconds.*
12.8S	*12.8 seconds.*

:SOURce:DM:DS3:FEAC:CBIT *<row, cbit, 0 | 1>*

This command sets the specified C-bit to a 0 or 1.

row: The M3 subframe (1–7).

cbit: The C-bit (1, 2, or 3) whose value you want to set to 0 or 1.

Example

```
:SOURce:DM:DS3:FEAC:CBIT 1, 3, 0
```

- Assigns the value 0 to the C3 bit in row 1 (the FEAC bit).

:SOURce:DM:DS3:FEAC:CONT_ALM_STAT *<ON | OFF>*

This command controls whether the test set sends FEAC codes continuously (ON), or not (off).

The :SOURce:DM:DS3:FEAC:ALM_STAT command specifies the FEAC code to send. When off, the FEAC bit is set by the DS3 C-Bit Control screen: Row 1, C3 program bit.

Example

```
:SOURce:DM:DS3:FEAC:CONT_ALM_STAT ON
```

- Configures the test set to continuously send the specified FEAC code.

:SOURce:DM:DS3:FEAC:BURST_SETTING <*setting*>

This command selects the type of FEAC code to transmit when you press the test set's ACTION (INJECT) key, or issue the command :SOURce:DM:DS3:FEAC:ACTION.

The :SOURce:DM:DS3:FEAC:BURST_SIZE command specifies how many times the FEAC code is sent.

The following table lists command parameters.

:SOURce:DM:DS3:FEAC:BURST_SETTING <setting>

Setting	Code Transmitted
NONE	*No codes transmitted.*
ACTIVATE	*Loop-up code for :SOURce:DM:DS3:FEAC:LOOPBACK_LINE.*
DEACTIVATE	*Loop-down code for :SOURce:DM:DS3:FEAC:LOOPBACK_LINE.*
ALM_STAT	*FEAC code specified by :SOURce:DM:DS3:FEAC:CONT_ALM_STAT.*

Example

```
:SOURce:DM:DS3:FEAC:BURST_SETTING ACTIVATE
```

- Configures the test set to send loop-up code to the line specified by the :SOURce:DM:DS3:FEAC:LOOPBACK_LINE command.

:SOURce:DM:DS3:FEAC:LOOPBACK_LINE *<line>*

This command selects the DS3 line to send loop code to when :SOURce:DM:DS3:FEAC:BURST_SETTING is set to ACTIVATE or DEACTIVATE. For **line**, specify the decimal equivalent of the line ID code listed in the following table.

Loopback Line> Field FEAC Code Selections

Line	Line ID Code	Decimal	Line	Line ID Code	Decimal
DS3 Line	*011011*	*27*	*DS1 Line—#15*	*101111*	*47*
DS1 Line—#1	*100001*	*33*	*DS1 Line—#16*	*110000*	*48*
DS1 Line—#2	*100010*	*34*	*DS1 Line—#17*	*110001*	*49*
DS1 Line—#3	*100011*	*35*	*DS1 Line—#18*	*110010*	*50*
DS1 Line—#4	*100100*	*36*	*DS1 Line—#19*	*110011*	*51*
DS1 Line—#5	*100101*	*37*	*DS1 Line—#20*	*110100*	*52*
DS1 Line—#6	*100110*	*38*	*DS1 Line—#21*	*110101*	*53*
DS1 Line—#7	*100111*	*39*	*DS1 Line—#22*	*110110*	*54*
DS1 Line—#8	*101000*	*40*	*DS1 Line—#23*	*110111*	*55*
DS1 Line—#9	*101001*	*41*	*DS1 Line—#24*	*111000*	*56*
DS1 Line—#10	*101010*	*42*	*DS1 Line—#25*	*111001*	*57*
DS1 Line—#11	*101011*	*43*	*DS1 Line—#26*	*111010*	*58*
DS1 Line—#12	*101100*	*44*	*DS1 Line—#27*	*111011*	*59*
DS1 Line—#13	*101101*	*45*	*DS1 Line—#28*	*111100*	*60*
DS1 Line—#14	*101110*	*46*	*DS1 Line—All*	*010011*	*19*

Example

```
:SOURce:DM:DS3:FEAC:LOOPBACK_LINE 47
```

- Sends loop-up (activate) or loop-down (deactivate) code to DS1 line #15.

:SOURce:DM:DS3:FEAC:BURST_SIZE <*size*>

This command specifies the number of times to transmit the selected FEAC code when you press the test set's ACTION (INJECT) key, or issue the :SOURce:DM:DS3:FEAC:ACTION command.

Use :SOURce:DM:DS3:FEAC:BURST_SETTING to select the FEAC code.

size: The number of times (3–15) to transmit the FEAC code.

Example

```
:SOURce:DM:DS3:FEAC:BURST_SIZE 10
```

- Configures the test set to send the selected FEAC code 10 times.

:SOURce:DM:DS3:FEAC:ALM_STAT <*alarm*>

This command selects the FEAC code to transmit when :SOURce:DM:DS3:FEAC:CONT_ALM_STAT is set to ON, or :SOURce:DM:DS3:FEAC:BURST_SETTING is set to ALM_STAT.

For **alarm**, specify the decimal equivalent of the FEAC code bit sequence listed in the following table.

Alarm/Status> Field FEAC Code Selections

FEAC Code	Bit Sequence	Decimal	Description
DS3 Eqpt Fail (SA)	*011001*	*25*	*DS3 service-affecting equipment failure*
DS3 LOS/HBER	*001110*	*14*	*Loss of signal/High bit error ratio*
DS3 Out-of-Frame	*000000*	*0*	
DS3 AIS Received	*010110*	*22*	*Alarm indication signal*
DS3 IDLE Received	*011010*	*26*	
DS3 Eqpt Fail (NSA)	*001111*	*15*	*DS3 non-service-affecting equipment failure*
Com Eqpt Fail (NSA)	*011101*	*29*	*Non-service-affecting common equip. failure*
Multi DS1 LOS/HBER	*010101*	*21*	*Multiple DS1 loss of signal/high bit error ratio*
DS1 Eqpt Fail (SA)	*000101*	*5*	*Service-affecting DS1 equipment failure*
Single DS1 LOS/HBER	*011110*	*30*	*Single DS1 loss of signal/high bit error ratio*
DS1 Eqpt Fail (NSA)	*000011*	*3*	*Non-service-affecting DS1 equipment failure*
DS3 NIU Loop Up	*001001*	*9*	*Network interface unit loop activation*
DS3 NIU Loop Down	*010010*	*18*	*Network interface unit loop deactivation*
—	—		*All other bit sequences are unassigned.*

Example

```
:SOURce:DM:DS3:FEAC:ALM_STAT 25
```

- Configures the test set to send a DS3 Equipment Failure (SA) alarm (FEAC code bit sequence = 011001).

:SOURce:DM:DS3:FEAC:ACTION

This command is equivalent to pressing the test set's ACTION (INJECT) key from the DS3 FEAC Control screen.

:SOURce:DM:<FT1 | FE1> <*channels*>

This :SOURce command selects which channels are used to make up the fractional T1 (FT1) or fractional E1 (FE1) signal.

***RST default:** (All channels)

Example

```
:SOUR:DM:FT1 1,2,3,7,9,21
```

- Sets the FT1 signal to be made up of DS0 channels 1, 2, 3, 7, 8, and 21.

Example

```
:SOUR:DM:FE1 7,16
```

- Sets the FE1 signal to be made up of timeslots 7 and 16. Note that if the E1 framing format is TS0/16 or TS0/16CRC then timeslot 16 cannot be used.

For FT1, the highest channel number is 24; for FE1, the highest channel number is 31.

:SOURce:DM:FRAMe:TYPE1 <*frame format*>

This :SOURce command sets the DS1 and E1 framing formats.

***RST default:** ESF

Example

`:SOUR:DM:FRAM:TYPE1 ESF`

- Sets the frame type for a DS1 signal to Superframe format.

Note: *For E1 testing, you must set the DS1/E1 mode to E1 before setting the other signal parameters. Use the :SOURce:DM:MODE E1 or E1&TS commands (see page 12–55).*

The following table lists the framing types to use with this command:

:SOURce:DM:FRAMe:TYPE1 <*framing format*>

Framing Format	Description
SF	*DS1 SuperFrame format.*
ESF	*DS1 Extended Super Frame format.*
SLC_96	*DS1 SLC-96 frame format.*
TS0	*E1 TS0 framing (:SOUR:DM:MODE:E1)*
TS0&CRC	*E1 TS0/CRC framing (:SOUR:DM:MODE:E1)*
TS0&TS16	*E1 TS0/TS16 framing (:SOUR:DM:MODE:E1)*
TS0&TS16&CRC	*E1 TS0/TS16/CRC framing (:SOUR:DM:MODE:E1)*
NONE	*DS1 or E1 unframed.*

Also, executing the query command `:SOURce:DM:FRAMe:TYPE1?` returns DS1 frame setting.

:SOURce:DM:FRAMe:TYPE3 <*framing type*>

This :SOURce command sets the DS3 framing format.

***RST default:** M13

Example

```
:SOUR:DM:FRAM:TYPE3 M13
```

- Sets the frame type for a DS3 signal to M13.

The following table lists the framing types to use with this command:

:SOURce:DM:FRAMe:TYPE3 <*framing type*>

Framing Type	Description
CBIT	*The third, fifth, and seventh bits in a DS3 signals' M-subframe.*
M13	*The multiplex between the DS1 and DS3 levels: up to 28 DSI facilities are combined into a single DS3 facility.*
NONE	*No framing type.*

Also, executing the query command `:SOURce:DM:FRAMe:TYPE3?` returns DS3 frame setting.

:SOURce:DM:MODE?

This :SOURce command lets you determine the current DS1/E1 mode (see below).

:SOURce:DM:MODE <*DS1/E1 mode*>

This :SOURce command lets you set the DS1/E1 mode. The test set must be set to the appropriate mode for the test you want to run (DS1 or E1).

Note: *You must set the DS1/E1 mode before you set any of the other DS1 or E1 parameters.*

***RST default:** ?DS1

Example

```
:SOUR:DM:MODE E1
```

- Sets the DS1/E1 mode to E1.

The following table lists the framing types to use with this command:

:SOURce:DM:MODE <*DS1/E1 mode*>

Source	Description
DS1	*DS1 mode.*
DS1&DS0	*DS1 with DS0 subrate.*
FT1	*Fractional T1 mode.*
E1	*E1 (2,048 Mbs).*
E1&TS	*E1 with TS subrate.*
FE1	*Fractional E1 mode.*

:SOURce:DM:NX <64K | 56K>

This :SOURce command sets the DS0 base rate to either 64 kHz or 56 kHz. Use a question mark without the parameter to query the test set for the current setting.

***RST default:** 64K

Example

`:SOUR:DM:NX 56K`

- Sets the DS0 base rate to 56 kilohertz.

:SOURce:DM:OH <*DS0 signaling bit*>,<*value*>

This :SOURce command lets you control the signaling bits for a DS0 signal.

***RST default:** 0.

Example

Add a question mark (?) and omit the second parameter to *query* the unit for the current value. For example `:SOUR:DM:OH?FEBE`.

`:SOUR:DM:OH SIGA,0`

- Sets signaling bit A bits to zero.

The following table lists the parameters to use with this command:

:SOURce:DM:OH <*DS0 signaling bit*>,<*value*>

OH Bit	Valid Values	Description
SIGA	*1 or 0*	*Signaling bit A*
SIGB	*1 or 0*	*Signaling bit B*
SIGC	*1 or 0*	*Signaling bit C*
SIGD	*1 or 0*	*Signaling bit D*

:SOURce:DM:OH *<DS3 overhead bit>,<value>*

This :SOURce command lets you control the overhead bits for a DS3 signal as well as the DS2 X-bit state.

***RST default:** FEBE = 111 or XBIT = 1.

Example

Add a question mark (?) and omit the second parameter to *query* the unit for the current value. For example `:SOUR:DM:OH?FEBE`.

`:SOUR:DM:OH FEBE,111`

- Sets the FEBE bits to all ones.

The following table lists the parameters to use with this command:

:SOURce:DM:OH *<DS3 overhead bit>*,*<value>*

OH Bit	Valid Values	Description
FEBE	*000, 001, 010—111*	*Far-End Block Error.*
XBIT	*1 and 0*	*The first bit in the first and second M-subframes in a DS3 M-frame.*
OBIT	*1 through 262143*	*Programs the 18 overhead bits in the unused bandwidth created when DS3 is mapped into STS-1 (O-bits).* *Enter the decimal equivalent of the 18-bit binary value to be encoded into the O-bits.*
DS2_XBIT	*1 and 0*	*Sets the transmit DS2 X-bit state.*

:SOURce:DM:OH <*overhead byte*>,<*value*>

This :SOURce command lets you control the SONET overhead.

***RST default: 0** (zero) or ASCII Null

Example

`:SOUR:DM:OH C2,1`

- Sets the C2 byte to the binary equivalent of 1 (00000001).

Add a question mark (?) and omit the second parameter to *query* the unit for the current value. For example `:SOUR:DM:OH? C2`.

The following table lists the parameters to use with this command. For each byte (except J1 and J0, see table), enter the decimal equivalent for the desired 8-bit value (0–255). Some bytes in the table are listed in groups by function. However, you can only set one byte at a time.

:SOURce:DM:OH SONET <*overhead byte*>,<*value*>

OH Byte	Description
C2	*Path signal label.*
D1 | D2 | D3	*Section DCC.*
D4 | D5 | … | D12	*Line DCC.*
E1 | E2	*Local (E1) or express (E2) orderwire.*
F1 | F2	*Section user channel (F1) and path user channel (F2).*
G1	*Path status byte.*
H1	*H1ss: Field set bit 5 of the H1 byte. Set bit 5 to 1 to indicate SDH signal.*
H4	*VT multiframe phase indicator.*
J0 or J1	*Enter an ASCII string, as many as 64 characters long, for the Path (J1) or Section (J0; OC-12 only) trace. Each character is one byte.*
K1 | K2	*APS channel.*
Z1 | Z2	*Synchronization status (STS-1 #1) and growth bytes.*
Z3 | Z4 | Z5	*Growth bytes.*

Note:
Also see fill selections on format table on page12-59.

:SOURce:DM:OH *<VT OH byte>,<value>*

These :SOURce commands let you control the VT path overhead bytes.

***RST default:** 0 (zero) or ASCII null.

Example

Add a question mark (?) and omit the second parameter to *query* the unit for the current value. For example `:SOUR:DM:OH? J2`.

```
:SOUR:DM:OH Z6,1
```

- Sets the Z6 byte to the binary equivalent of 1 (00000001).

The following table lists the parameters to use with this command.

:SOURce:DM:OH SONET<*VT overhead byte*>,<*value*>

VT OH Byte	Description
J2	*Enter an ASCII string, as many as 64 characters long, for the VT path trace signal. Each character is one byte. Valid only in BSYNC mode.*
Z6	*Enter the decimal value for the growth byte. Byte-sync modes only.*
Z7	*Enter the decimal for this byte: Bit 8 is the RFI-V; other bits are undefined. Not applicable for asynchronous VT modes.*
SIGLBL	*Enter the decimal value for the VT signal label (bits 5 through 7 of the V5 byte). Value can be from 0 through 7.*

Also see fill selections on format table on page12-59

Fill Action Reference

- The **Fill on ACTION** (or **Fill on INJECT**) and **Format** fields define the transmit trace string to transmit when you press the ACTION (INJECT) button (see the table below).
- **Results Display in** sets the display mode for the path-trace byte and section-trace byte values (below). This field has two options:
  - ASCII (the default)–displays values in ASCII.
  - Hex & ASCII–displays values in hex and ASCII.

Fill on Action	Format - for Fill on Action (inject button) Selections		
	1-Byte	**16-Byte**	**64-Byte**
NULL	*0x00 in all bytes.*	*0x00 in all bytes.*	*0x00 in all bytes.*
HEX	*0x01 in all bytes.*	*0x41 – 0x4F with CRC7, copied 4 times.*	*0x41 to 0x7F (ASCII A to DEL), followed by a carriage return (0x0D) & line feed (0x0A).*
ASCII	*0x41 in all bytes.*	*“nnnnnn” serial number with CRC7, copied 4 times.*	*“Agilent Technologies 156MTS Test Set Serial No. nnnnnn” followed by carriage return (0x0D) and line feed (0x0A).*
USER	*First byte copied to all 64 bytes.*	*Calculate and insert CRC7 of first 16 bytes; copy first 16 bytes 4 times.*	*No action.*

:SOURce:DM:OH FRM_REGEN,<PASS|REGEN>

This command affects the framing of the DS1 dropped from the VT1.5 (available for byte-synchronous VT modes only).

***RST default:** pass

Example

`:SOUR:DM:OH FRM_REGEN,REGEN`

- Sets the DS1 drop framing to be regenerated by the test set.

Add a question mark (?) and omit the second parameter to *query* the unit for the current value. For example `:SOUR:DM:OH`**`?`** `FRM_REGEN`.

PASS: The dropped DS1 data and framing are both derived from the VT1.5.

REGEN: The DS1 data and signaling are derived from the VT1.5; the DS1 framing is internally generated by the test set.

:SOURce:DM:SCRAMBLE <ON | OFF>

This command enables and disables the STS-1 signal scrambler.

***RST default:** ON

Example

`:SOUR:DM:SCRAMBLE OFF`

- Disables the STS-1 scrambler.

:SOURce:DM:PROG1 PATTERN,<*pattern*>

This :SOURce command configures the DS1 user-programmable pattern.

***RST default:** 101010101010101010101010

Example

`:SOUR:DM:PROG1 PATTERN,10110111`

- Transmits the user defined pattern 10110111 repeating data pattern.

Where pattern is the 2 to 24-character pattern. Enter 1s and 0s for the pattern bits.

Note: *This command overrides the :SOUR:DM:DATA command.*

:SOURce:DM:PROG3 PATTERN,<*pattern*>

This :SOURce command configures the DS3 user-programmable pattern.

***RST default:** 101010101010101010101010

Example

`:SOUR:DM:PROG3 PATTERN,10110110`

- Transmits the user defined pattern 10110110 repeating data pattern.

Where pattern is the 2 to 24-character pattern. Enter 1s and 0s for the pattern bits.

Note: *This command overrides the :SOUR:DM:DATA command.*

:SOURce:DM:PROG32 PATTERN,<*pattern*>

This :SOURce command configures the STS-12c user-programmable pattern.

Each digit of the eight-digit hexadecimal value can be set from 0 through F. The binary equivalent of the hex value forms the 32-bit pattern when the :SOURce:DM:DATA5 PROG32 command is used.

***RST default:** 12345678

Example

```
:SOUR:DM:PROG3 PATTERN,1234CDEF
```

- Transmits the repeating, user-defined pattern:

```
00010010001101001100110111101111
```

:SOURce:DM:VT15 <*sync mode*>

This :SOURce command selects the VT mode (either asynchronous or byte-synchronous).

***RST default:** ASYNC

Example

`:SOURce:DM:VT15 bsync`

- Sets the VT1.5 mode to byte-synchronous.

The sync mode can be set to either of the following:

ASYNC: Asynchronous VT1.5 mode.

BSYNC: Byte-synchronous VT1.5 mode.

Also, executing the query command `:SOURce:DM:VT15?` returns VT1.5 mode setting in either asynchronous or byte-synchronous.

:SOURce:DM:VT15 <sync mode>

13

:STATus Command Reference

:STATus Command Reference

The :STATus commands let you retrieve the value from the various SCPI registers. The :STATus command tree is:

:STATus Command Structure

Root node	Level 1 node	Level 2 node	Parameter
:STATus	*:OPERation*	*:EVENt?*	
		:CONDition?	
		:ENABLe	*<decimal value>*
		:ENABLe?	
	:QUEStionable	*:EVENt?*	
		:CONDition?	
		:ENABLe	*<decimal value>*
		:ENABLe?	
	:PRESet		

:STATus:OPERation:EVENt?

The SCPI standard requires that the following two commands be included for reading the Operation Event register and its corresponding Operation Condition register (see *Using Status Registers*, page 3–10).

Use the following command to return and clear the value in the Operation Event register:

`:STATus:OPERation:EVENt?`

:STATus:OPERation:CONDition?

Use the following command to retrieve the value of the Operation Condition register:

`:STATus:OPERation:CONDition?`

:STATus:OPERation:ENABle <*value*>

Use the following command to set the Operation Event Enable register:

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

Where value is a decimal integer representing the desired 16-bit mask.

:STATus:OPERation:ENABle?

Use the following command to return the decimal value of the 16-bit mask:

`:STATus:OPERation:ENABle?`

:STATus:QUEStionable:EVENt?

The SCPI standard requires the following two commands for reading the Questionable Event register and its corresponding Questionable Condition register. In addition, a Questionable Event Enable register can be set and cleared (see *Using Status Registers*, page 3–10).

Use the following command to return and clear the value in the Questionable Event register:

`:STATus:QUEStionable:EVENt?`

:STATus:QUEStionable:CONDition?

Use the following command to retrieve the value of the Questionable Condition register:

`:STATus:QUEStionable:CONDition?`

:STATus:QUEStionable:ENABle <*value*>

Use this command to set the Questionable Event Enable register:

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

Where value is a decimal integer representing the desired 16-bit mask.

:STATus:QUEStionable:ENABle?

Use this command to return the decimal value of the 16-bit mask:

`:STATus:QUEStionable:ENABle?`

:STATus:PRESet

Use the following command to clear both the Operation Event Enable register and the Questionable Event Enable register to zero:

```
:STATus:PRESet
```

14

:SYSTem Command Reference

:SYSTem Command Reference

The SYSTem and :SYSTem commands let you collect the functions that are not related to instrument performance. The command tree is:

SYSTem and :SYSTem Command Structure

Root node	Level 1 node
:SYSTem	*:REMote (Only works from front panel operation mode.)*
:SYSTem	*:ERRor?*
	:VERSion?
	:OPTionS?
	:PRESet
	:SCPI:
	:STORE

:SYSTem:REMote

This command can only be used if the instruction is running from a Front Panel Mode.

The :SYSTem:REM command works from a front panel operation and activates SCPI mode without resetting the test set's configuration. This can be useful if you want to set up the test set using the front panel controls, or by using the recall function; but run the test remotely with the SCPI interface.

`:SYSTem:REMote`

- This puts the test set into SCPI mode, stopping any active test but maintaining the instrument's configuration.

Note: *Compare this command to *RST. See *RST, page 5–2.*

:SYSTem:ERRor?

Use the following query command to return the next error code in the error queue. If the queue is empty, you will see the +0 value.

```
:SYSTem:ERRor?
```

:SYSTem:VERSion?

Use the following query command to see the SCPI revision level installed in your test set:

```
:SYSTem:VERSion?
```

:SYSTem:PRESet

Use the following command to take the test set out of HP-IB mode and restore the set to normal front panel operation:

```
:SYSTem:PRESet
```

:SYSTem:OPTionS?

This query command retrieves the test set’s software configuration code. The configuration code is a 24-character ASCII string, with 8 bytes of hexadecimal data (each separated by a space), and NULL terminated.

Example

```
:SYSTem:OPTS?
```

Retrieves the information in the Config. Code field of the System Software Configuration screen (see your test set manual for more information about the screen.) Following is an example of a typical display:

```
01 01 00 FF FE D1 FF F7
```

:SYSTem:SCPI:

In Local Mode, Use following command to select communications interface connection that must be specified as part of this command as example: IEEE-488-{-1 or -2}. Refer to page 17-2 for the SCPI Interface information and description. Example:

```
:SYSTem:SCPI:IEE488
```

SYSTem:STORE:

Local Mode

:SYSTEM:STORE:n[:name] In **Local (or front panel) Mode**, this command stores configuration to a position stored as specified by the number position 1 through 10 and the name specified as “name”.

Test set configurations are saved using SCPI command **`:SYSTEM:STORE:n[:name]`** where “n” is a number 1 to 10 specifying which slot the configuration is to be saved.

The “name” is an optional parameter, when specified is stored as the name of the configuration thus replacing the default configuration name (the mode title name). Note that to store test set modes, the unit must be operating in the desired mode via local (front panel) mode.

SCPI Mode

:SYSTEM:STORE:? n In **SCPI mode**, this command queries the name of configuration stored in position name “n”. **`:SYSTEM:STORE:? n`**

The “name” is not permitted for s query in SCPI mode.

Error Code Reference

Error Code Reference

The CERJAC test sets save errors that may occur when you send a SCPI command that does not follow the correct syntax or when processing a SCPI command in the Error Queue. You can use the :SYST:ERR? query command to read these errors, one at a time. When you issue this query command, the test set sends the next error in the queue.

An error response consists of an error number followed by a message. The errors that you can receive from as CERJAC test set are listed in the following table.

SCPI Error Messages

Error Code	Error Message	Description
0	*No error.*	*None.*
-100	*Command error.*	*You did not enter the command correctly. Check command syntax and spelling.*
-109	*Missing parameter.*	*You are missing a parameter in the command.*
-112	*Parameter mnemonic too long (>12 characters).*	*The parameter you entered is greater that 12 characters. Check command syntax and spelling.*
-120	*Numeric data error (out of range).*	*You entered an invalid number in a command. Check your command and the valid ranges that apply.*
-221	*Setting conflict.*	*Command conflicts with the current settings.*
-223	*Too much data.*	*You entered too many commands or parameters in the command string. Break up the commands into two string.*
-350	*Queue overflow.*	*Indicates that the error queue is full and some errors may have been lost.*

16

SCPI Programming Examples

About the Examples

This chapter contains programming examples that address many applications, and can be modified to perform additional required functions. Each program example has two versions which show you how to issue SCPI commands:

- One at a time in HP-IB mode
- Inside a C program

Use these program examples to help you create your own custom test programs. See Chapters 5 through 14 for information on a SCPI command used in an example.

Conventions Used in Examples

The following conventions are used in the programming examples in this chapter:

Responses

When an inquiry command is sent over the IEEE-488 bus to the test set, the test set will return a generated response to the output queue. When a response results from an inquiry command, it is noted in the example in the following format:

{read bus> *xxx*}

Where *xxx* indicate the expected result. When you see the “read bus” notation, you should select the HP-IB String... option under the Enter menu to view the results. If you are running a program, the output may be saved to a file or printed on the controller’s screen.

Note: *For RS-232 applications, the controller does not have to read the bus, since the response to a query is transmitted immediately by the test set over the RS-232 connection.*

Wait Command

The minimum command execution time is 250 milliseconds (one quarter second). Some commands may require a wait command to allow for test set processing time, or to accumulate results. When a wait command is required, it is noted in an example in the following format:

{wait *xS*}

Where *x*S indicates the number of seconds (S) to wait before continuing.

Program Comments

The program comments explain each command or a group of commands.

SCPI program commands are listed in the example table next to each command. C program comments begin with a /* and end with a */. These comments provide you with additional information about the commands or applications.

Test a DS3 Signal with DS1 Payload

This example shows how to transmit a test signal containing M13 format DS3 signals. The transmit and receive signal rate is the *RST default (OC-12, OC-3, STS-1, or DS3 depending on test set configuration.) Receive the signal, drop out the first STS1, from it drop the DS3, and from it drop the first DS1. Then request testing results using this configuration.

Using SCPI Commands

The following example shows how to complete this test using SCPI commands issued in SCPI mode:

M13 Mapped DS3/DS1 Payload Using SCPI Commands

SCPI Command	Description
`*RST`	*Accesses HP-IB mode and sets the parameters to their default settings.*
`:INIT`	*Starts the test. (Notice the green RUN LED on the test set.)*
`:FETC:DMOD:STS1? SIGNAL`	*Requests a SONET signal status.* *NOTE: A response of +1 for each of the status indicators tells you that the receive signal is present and the dropped DS3 and DS1 signals have the expected framing types. A response of +0 for the STS-1 Loss Of Pointer alarm indicates that the STS-1 pointer is valid (no alarm condition).*
{read bus> STS1 signal status}	*Sends the signal status to the output queue.*
`:FETC:DMOD:STS1? LOPNTR`	*Requests an STS1 Loss of Pointer alarm indication.*
{read bus> STS1 Loss of Pointer alarm}	*Sends the alarm indicator to the output queue.*
`:FETC:DMOD:DS3? M13FRAME`	*Requests a DS3 M13 frame detect status.*
{read bus> DS3 m13 frame detect status}	*Sends the detect status to the output queue.*
`:FETC:DMOD:DS1? SFSYNC`	*Requests a DS1 SF frame detect status.*
{read bus> DS1 SF frame detect status}	*Sends the detect status to the output queue.*
`:SOUR:DM:EINJ:DS1 DATA,1E-3`	*Injects DS1 errors.*
{wait 1S}	*Waits 1 second to allow errors to accumulate.*

M13 Mapped DS3/DS1 Payload Using SCPI Commands, continued

SCPI Command	Description
`:FETC:DMOD:DS1? bit`	*Requests the DS1 bit error count.*
{read bus> DS1 bit error count}	*Sends the error bit count to the output queue.*
`:FETC:DMOD:DS1? bit_aratio`	*Requests a DS1 bit error average ratio.*
{read bus> DS1 bit error average ratio}	*Sends the ratio to the output queue.*
`:FETC:DMOD:DS1? bit_cratio`	*Requests a DS1 bit error current ratio.*
{read bus> DS1 bit error current ratio}	*Sends the ratio to the output queue.*
`:FETC:DMOD:DS1? bit_es`	*Requests the number of DS1 bit errored seconds.*
{read bus> DS1 bit errored seconds}	*Sends the number of bit errored seconds to the output queue.*
`:FETC:DMOD:DS1? bit_ses`	*Requests the number of DS1 severely errored seconds.*
{read bus> DS1 severely errored seconds}	*Sends the number of DS1 severely errored seconds to the output queue.*
`:ABOR`	*Ends the test. (Notice the green RUN LED turn off on the test set.)*
`:SYST:PRES`	*Takes the test set out of HP-IB mode and returns control to the test set.*

Using C

The following example shows how to complete this test using SCPI commands within a C program:

C Program listing for Testing M13 Format DS3/DS1 Payload

```
/******************************************************************************
This example shows how to transmit a test signal containing M13-mapped DS3 signals.
******************************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define GPIB_WRITE_ERROR -1
#define GPIB_READ_ERROR -2
#define SYSTEM_QUERY_CMD"SYST:ERR?"

/* ** External Functions ** */

/*      Note, the following are vendor specific GPIB I/O functions. The functions are
        generic. Therefore, the parameters and/or function name may not match the
        functions provided by specific vendors.
*/

/* Function:    open
Purpose:        This function establishes a session with the desired interface or deice and returns a session
                identifier. The session identifier should only be passed as a parameter to other vendor specific GPIB
                functions.
Return:         0 if Success or error code.
*/
extern int iopen( char *address, int *Id);

/* Function:    write
Purpose:        This is a vendor specific function that writes SCPI commands to the system via GPIB interface.
Return:         0 if Success or error code.
*/
extern int iwrite( int id, char *data, unsigned int datalen );

/* Function:    iread
Purpose:        This is a vendor specific function that reads results from the system via GPIB interface.
Return:          0 if Success or error code.
*/
extern int iread( int id, char *buffer, unsigned int bufsize );

/* ** Local Functions ** */

static int GPIBSendCommand( int instId, char *command );
static int GPIBReadResult( int instId, char *result, unsigned int cnt );
static void ProcessErrors( int error_code );

/* ** Main Program ** */
/*
```

C Program listing for Testing M13 Format DS3/DS1 Payload, continued

```
        Note, This example will exit the program when error occurs.
*/
void main ()

{
        int instId;/*session identifier */
        int status;/* GPIB I/O and system status */
        char result[50];/* result from instrument */

        /* establish session between GPIB interface and instrument */
        status = iopen( "HPIB, 29", &instId );
        if (status != 0)
        {
              ProcessErrors( status );
        return;
        }

        /* Initialize the instrument */
        status = GPIBSendCommand( instId, "*RST" );
        if (status != 0)
        {
              ProcessErrors( status );
              return;
        }

        /* Start the test */
        status = GPIBSendCommand( instId, "INIT" );
        if (status != 0)
        {
        ProcessErrors( status );
        return;
        }

        /* Request a SONET signal status */
        status = GPIBSendCommand( instId, "FETC:DMOD:STS1? SIGNAL" );
        if (status != 0)
        {
              ProcessErrors( status );
              return;
        }

        /* Read STS1 signal status */
        status = GPIBReadResult( instId, result, 50 );
        if (status != 0)
        {
              ProcessErrors( status );
              return;
        }
```

C Program listing for Testing M13 Format DS3/DS1 Payload, continued

```
/* Request STS1 Loss of Pointer alarm indication */
status = GPIBSendCommand( instId, "FETC:DMOD:STS1? LOPNTR" );
if (status != 0)
{
    ProcessErrors( status );
    return;
}

/* Read STS1 Loss of Pointer alarm */
status = GPIBReadResult( instId, result, 50 );
if (status != 0)
{
    ProcessErrors( status );
    return;
}

/* Request a DS3 M13 frame detect status */
status = GPIBSendCommand( instId, "FETC:DMOD:DS3? M13FRAME" );
if (status != 0)
{
    ProcessErrors( status );
    return;
}

/* Read a DS3 M13 frame detect status */
status = GPIBReadResult( instId, result, 50 );
if (status != 0)
{
    ProcessErrors( status );
    return;
}

/* Request a DS1 SF frame detect status */
status = GPIBSendCommand( instId, "FETC:DMOD:DS1? SFSYNC" );
if (status != 0)
{
    ProcessErrors( status );
    return;
}

/* Read a a DS1 SF frame detect status */
status = GPIBReadResult( instId, result, 50 );
if (status != 0)
{
    ProcessErrors( status );
    return;
}
```

C Program listing for Testing M13 Format DS3/DS1 Payload, continued

```
    /* Inject DS1 errors */
    status = GPIBSendCommand( instId, "SOUR:DM:EINJ:DS1 data, 1E-3" );
    if (status != 0)
    {
        ProcessErrors( status );
        return;
    }

    /* wait 1 second */

    /* Request DS1 bit error count */
    status = GPIBSendCommand( instId, "FETC:DMOD:DS1? BIT" );
    if (status != 0)
    {
        ProcessErrors( status );
        return;
    }

    /* Read DS1 bit error count */
    status = GPIBReadResult( instId, result, 50 );
    if (status != 0)
{
        ProcessErrors( status );
        return;
    }

    /* Request DS1 bit error average ratio */
    status = GPIBSendCommand( instId, "FETC:DMOD:DS1? BIT_ARATIO" );
    if (status != 0)
    {
        ProcessErrors( status );
        return;
    }

    /* Read DS1 bit error average ratio */
    status = GPIBReadResult( instId, result, 50 );
    if (status != 0)
    {
        ProcessErrors( status );
        return;
    }

    /* Request DS1 bit error current ratio */
    status = GPIBSendCommand( instId, "FETC:DMOD:DS1? BIT_CRATIO" );
    if (status != 0)
    {
        ProcessErrors( status );
        return;
}
```

C Program listing for Testing M13 Format DS3/DS1 Payload, continued

```
/* Read DS1 bit error current ratio */
status = GPIBReadResult( instId, result, 50 );
if (status != 0)
{
      ProcessErrors( status );
      return;
}

/* Request DS1 bit error seconds */
status = GPIBSendCommand( instId, "FETC:DMOD:DS1? BIT_ES" );
if (status != 0)
{
      ProcessErrors( status );
      return;
}

/* Read DS1 severely error seconds */
status = GPIBReadResult( instId, result, 50 );
if (status != 0)
{
      ProcessErrors( status );
      return;
}

/* Request DS1 severely error seconds */
status = GPIBSendCommand( instId, "FETC:DMOD:DS1? BIT_SES" );
if (status != 0)
{
      ProcessErrors( status );
      return;
}

/* Read DS1 bit error seconds */
status = GPIBReadResult( instId, result, 50 );
if (status != 0)
{
      ProcessErrors( status );
      return;
}

/* End Test */
status = GPIBSendCommand( instId, "ABORT" );
if (status != 0)
{
      ProcessErrors( status );
      return;
}
```

C Program listing for Testing M13 Format DS3/DS1 Payload, continued

```
    /* Take test set out of HP-IB mode */
    status = GPIBSendCommand( instId, "SYST:PRES" );
    if (status != 0)
    {
        ProcessErrors( status );
        return;
    }
}

/* Function:   GPIBSendCommand
Purpose:       This function writes SCPI commands to the system and verifies the command is acceptable by the
               system.
Return:        0 if Success or error code.
*/
static int GPIBSendCommand( int instId, char *command )
{
    char system_result[5];      /* SYST ERROR query buffer */
    int status;                 /*GPIB I/O and system status */

    /* ** send msg string to instrument and verify GPIB status ** */

    /* send command to instrument */
    status = iwrite( instId, command, strlen(command) );
    if (status != 0)
    {
    return GPIB_WRITE_ERROR;
    }
    /* send SYSTEM ERROR QUERY command in order to verify valid commands */
    status = iwrite( instId, SYSTEM_QUERY_CMD, strlen(SYSTEM_QUERY_CMD) );
    if (status != 0)
    {
        return GPIB_WRITE_ERROR;
    }

    /* Read system error result from instrument */
    status = iread(instId, system_result, 5);
    if (status != 0)
    {
    return GPIB_READ_ERROR;
    }

    /* convert system result into integer for error code (use as return value) */
    status = atoi( system_result );

    return status;
}
```

C Program listing for Testing M13 Format DS3/DS1 Payload, continued

```
/* Function:   GPIBReadResult
Purpose:       This function reads results from the system via GPIB interface.
Return:        0 if Success or error code.
*/
static int GPIBReadResult( int instId, char *result, unsigned int cnt )
{
        int status;           /* GPIB I/O and system status */
        /* Read system error result from instrument */
        status = iread(instId, result, cnt);
        if (status != 0)
        {
              return GPIB_READ_ERROR;
        }

        return status;
}

/* Function:   ProcessErrors
Purpose:       This is a generic function that process GPIB command or I/O errors.
Return:        void
*/
static void ProcessErrors( int error_code )
{
        /* Generic routines to process GPIB Command or I/O errors */
}
```

Test VT1.5 Signals

This example shows how to transmit a test signal containing VT1.5-mapped DS1 signals. The transmit and receive signal rate is the *RST default (OC-12, OC-3, STS-1 depending on test set configuration.) Receive the signal, drop out the first STS-1, from it drop the first VT1.5 with its DS1 payload. Then request testing results using this configuration.

Using SCPI Commands

The following example shows how to complete this test using SCPI commands issued in HP-IB mode:

Test VT1.5 Signals Using SCPI Commands

SCPI Command	Description
`*RST`	*Accesses HP-IB mode and sets the parameters to their default settings.*
`:ROUT:MATR STS#1,VT15`	*Sets the first transmit STS-1 to VT1.5 mapping.*
`:ROUT:MATR STS#2,VT15`	*Sets the second transmit STS-1 to VT1.5 mapping.*
`:ROUT:MATR STS#3,VT15`	*Sets the third transmit STS-1 to VT1.5 mapping.*
`:ROUT:MATR VT15DROP,STS#1`	*Drops the first receive STS-1 to the VT1.5 demultiplexing circuitry.*
`:ROUT:SEL DS1_RX,VT15`	*Directs the DS1 dropped from the VT1.5 to the DS1 error measurement sense block.*
`:INIT`	*Starts the test. (Notice the green RUN LED on the test set.)*
`:FETC:DMOD:STS1? SIGNAL`	*Requests a SONET signal status.* *NOTE: A response of +1 for each of the status indicators tells you that the receive signal is present and the dropped DS1 signal has the expected framing type. A response of +0 for the STS-1 and VT1.5 Loss Of Pointer alarms indicate that the STS-1 and VT1.5 pointer are valid (no alarm conditions).*
{read bus> STS1 signal status}	*Sends the signal status to the output queue.*
`:FETC:DMOD:STS1? LOPNTR`	*Requests an STS1 Loss of Pointer alarm indication.*
{read bus> STS1 Loss of Pointer alarm}	*Sends the alarm indicator to the output queue.*

Test VT1.5 Signals Using SCPI Commands, continued

SCPI Command	Description
`:FETC:DMOD:VT15? LOPNTR`	*Requests the VT1.5 Loss of Pointer alarm indication.*
{read bus> VT15 Loss of Pointer alarm}	*Sends the detect status to the output queue.*
`:FETC:DMOD:DS1? SFSYNC`	*Requests a DS1 SF frame detect status.*
{read bus> DS1 SF frame detect status}	*Sends the detect status to the output queue.*
`:SOUR:DM:EINJ:VT15 BER,1E-3`	*Injects VT1.5 errors.*
{wait 1S}	*Waits 1 second to allow errors to accumulate.*
`:FETC:DMOD:VT15? CV`	*Request VT1.5 Code Violation (parity error) count.*
{read bus> VT1.5 cv error count}	*Sends the error count to the output queue.*
`:FETC:DMOD:VT15? CV_ARATIO`	*Request VT1.5 Code Violation average error ratio (bit error rate).*
{read bus> VT1.5 cv average ratio}	*Sends the average ratio to the output queue.*
`:FETC:DMOD:DS1? BIT`	*Requests the DS1 bit error count.*
{read bus> DS1 bit error count}	*Sends the bit error count to the output queue.*
`:FETC:DMOD:DS1? BIT_CRATIO`	*Requests a DS1 bit error current ratio.*
{read bus> DS1 bit error current ratio}	*Sends the ratio to the output queue.*
`:FETC:DMOD:DS1? BIT_ES`	*Requests the number of DS1 bit errored seconds.*
{read bus> DS1 bit errored seconds}	*Sends the number of bit errored seconds to the output queue.*
`:ABOR`	*Ends the test. (Notice the green RUN LED turn off on the test set.)*
`:SYST:PRES`	*Takes the test set out of HP-IB mode and returns control to the test set.*

Using C

The following example shows how to complete this test using SCPI commands within a C program:

C Program Listing for Testing VT1.5 Signals Using SCPI Commands

```
/*****************************************************************************
     This example shows how to transmit a test signal containing VT1.5-mapped DS1
     signals.
******************************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define GPIB_WRITE_ERROR -1
#define GPIB_READ_ERROR -2
#define SYSTEM_QUERY_CMD"SYST:ERR?"

/* ** External Functions ** */

/* Note, the following are vendor specific GPIB I/O functions.  The functions are generic.  Therefore, the parameters
                 and/or function name may not match the functions provided by specific vendors.
*/

/* Function:i    open
  Purpose:       This function establishes a session with the desired interface or deice and returns a session
                 identifier.  The session identifier should only be passed as a parameter to other vendor-specific GPIB
                 functions.
  Return:        0 if Success or error code.
*/
extern int iopen( char *address, int *Id);

/* Function:   iwrite
   Purpose:      This is a vendor specific function that writes SCPI commands to the system via GPIB interface.
  Return:        0 if Success or error code.
*/
extern int iwrite( int id, char *data, unsigned int datalen );

/* Function:   iread
  Purpose:       This is a vendor specific function that reads results from the system via GPIB interface.
  Return:        0 if Success or error code.
*/
extern int iread( int id, char *buffer, unsigned int bufsize );


/* ** Local Functions ** */

static int GPIBSendCommand( int instId, char *command );
static int GPIBReadResult( int instId, char *result, unsigned int cnt );
static void ProcessErrors( int error_code );
```

C Program Listing for Testing VT1.5 Signals Using SCPI Commands, continued

```
/* ** Main Program  ** */
/*  Note, This example will exit the program when error occurs.
*/
void main ()
{
    int instId;/* session identifier */
    int status;/* GPIB I/O and system status */
    char result[50];/* result from instrument */

    /* establish session between GPIB interface and instrument */
    status = iopen( "HPIB, 29", &instId );
    if (status != 0)
    {
        ProcessErrors( status );
        return;
    }

    /* Initialize the instrument */
    status = GPIBSendCommand( instId, "*RST" );
    if (status != 0)
    {
        ProcessErrors( status );
        return;
    }

    /* Set the first transmit STS-1 to VT1.5 mapping */
    status = GPIBSendCommand( instId, "ROUT:MATR STS#1, VT15" );
    if (status != 0)
    {
        ProcessErrors( status );
        return;
    }

    /* Set the second transmit STS-1 to VT1.5 mapping */
    status = GPIBSendCommand( instId, "ROUT:MATR STS#2, VT15" );
    if (status != 0)
    {
        ProcessErrors( status );
        return;
    }

    /* Set the third transmit STS-1 to VT1.5 mapping */
    status = GPIBSendCommand( instId, "ROUT:MATR STS#3, VT15" );
    if (status != 0)
    {
        ProcessErrors( status );
        return;
    }
```

C Program Listing for Testing VT1.5 Signals Using SCPI Commands, continued

```
/* Drop the first receive STS-1 to the VT1.5 demux  */
status = GPIBSendCommand( instId, "ROUT:MATR VT15DROP, STS#1" );
if (status != 0)
{
    ProcessErrors( status );
    return;
}

/* Direct the DS1 dropped from the VT1.5 to DS1 error measurement */
status = GPIBSendCommand( instId, "ROUT:SEL DS1_RX, VT15" );
if (status != 0)
{
    ProcessErrors( status );
    return;
}

/* Start the tests */
status = GPIBSendCommand( instId, "INIT" );
if (status != 0)
{
    ProcessErrors( status );
    return;
}

/* Request a SONET signal status */
status = GPIBSendCommand( instId, "FETC:DMOD:STS1? SIGNAL" );
if (status != 0)
{
    ProcessErrors( status );
    return;
}

/* Read STS1 signal status */
status = GPIBReadResult( instId, result, 50 );
if (status != 0)
{
    ProcessErrors( status );
    return;
}

/* Request STS1 Loss of Pointer alarm indication */
status = GPIBSendCommand( instId, "FETC:DMOD:STS1? LOPNTR" );
if (status != 0)
{
    ProcessErrors( status );
    return;
}
```

C Program Listing for Testing VT1.5 Signals Using SCPI Commands, continued

```
/* Read STS1 Loss of Pointer alarm */
status = GPIBReadResult( instId, result, 50 );
if (status != 0)
{
      ProcessErrors( status );
      return;
}

/* Request VT15 Loss of Pointer alarm indication */
status = GPIBSendCommand( instId, "FETC:DMOD:VT15? LOPNTR" );
if (status != 0)
{
      ProcessErrors( status );
      return;
}

/* Read VT15 Loss of Pointer alarm */
status = GPIBReadResult( instId, result, 50 );
if (status != 0)
{
      ProcessErrors( status );
      return;
}

/* Request a DS1 SF frame detect status */
status = GPIBSendCommand( instId, "FETC:DMOD:DS1? SFSYNC" );
if (status != 0)
{
      ProcessErrors( status );
      return;
}

/* Read a a DS1 SF frame detect status */
status = GPIBReadResult( instId, result, 50 );
if (status != 0)
{
      ProcessErrors( status );
      return;
}

/* Inject VT15 errors */
status = GPIBSendCommand( instId, "SOUR:DM:EINJ:VT15 BER, 1E-3" );
if (status != 0)
{
      ProcessErrors( status );
      return;
}

/* wait 1 second */
```

C Program Listing for Testing VT1.5 Signals Using SCPI Commands, continued

```
/* Request VT15 code violation count */
status = GPIBSendCommand( instId, "FETC:DMOD:VT15? CV" );
if (status != 0)
{
    ProcessErrors( status );
    return;
}

/* Read VT15 code violation count */
status = GPIBReadResult( instId, result, 50 );
if (status != 0)
{
    ProcessErrors( status );
    return;
}

/* Request VT15 code violation average error ratio */
status = GPIBSendCommand( instId, "FETC:DMOD:VT15? CV_ARATIO" );
if (status != 0)
{
    ProcessErrors( status );
    return;
}

/* Read VT15 code violation average error ratio */
status = GPIBReadResult( instId, result, 50 );
if (status != 0)
{
    ProcessErrors( status );
    return;
}

/* Request DS1 bit error count */
status = GPIBSendCommand( instId, "FETC:DMOD:DS1? BIT" );
if (status != 0)
{
    ProcessErrors( status );
    return;
}

/* Read DS1 bit error count */
status = GPIBReadResult( instId, result, 50 );
if (status != 0)
{
    ProcessErrors( status );
    return;
}
```

C Program Listing for Testing VT1.5 Signals Using SCPI Commands, continued

```
/* Request DS1 bit error average ratio */
status = GPIBSendCommand( instId, "FETC:DMOD:DS1? BIT_ARATIO" );
if (status != 0)
{
    ProcessErrors( status );
    return;
}

/* Read DS1 bit error average ratio */
status = GPIBReadResult( instId, result, 50 );
if (status != 0)
{
    ProcessErrors( status );
    return;
}

/* Request DS1 bit error current ratio */
status = GPIBSendCommand( instId, "FETC:DMOD:DS1? BIT_CRATIO" );
if (status != 0)
{
    ProcessErrors( status );
    return;
}

/* Read DS1 bit error current ratio */
status = GPIBReadResult( instId, result, 50 );
if (status != 0)
{
    ProcessErrors( status );
    return;
}

/* Request DS1 bit error seconds */
status = GPIBSendCommand( instId, "FETC:DMOD:DS1? BIT_ES" );
if (status != 0)
{
    ProcessErrors( status );
    return;
}

/* Read DS1 severely error seconds */
status = GPIBReadResult( instId, result, 50 );
if (status != 0)
{
    ProcessErrors( status );
    return;
}
```

C Program Listing for Testing VT1.5 Signals Using SCPI Commands, continued

```
    /* End Test */
    status = GPIBSendCommand( instId, "ABORT" );
    if (status != 0)
    {
        ProcessErrors( status );
        return;
    }

    /* Take test set out of HP-IB mode */
    status = GPIBSendCommand( instId, "SYST:PRES" );
    if (status != 0)
    {
        ProcessErrors( status );
        return;
    }
}

/* Function:   GPIBSendCommand
   Purpose:     This function writes SCPI commands to the system and verifies the command is acceptable by the
                system.
   Return:       0 if Success or error code.
*/
static int GPIBSendCommand( int instId, char *command )
{
    char system_result[5];      /* SYST ERROR query buffer */
    int status;/* GPIB I/O and system status */

    /* ** send msg string to instrument and verify GPIB status ** */

    /* send command to instrument */
    status = iwrite( instId, command, strlen(command) );
    if (status != 0)
    {
        return GPIB_WRITE_ERROR;
    }

    /* send SYSTEM ERROR QUERY command in order to verify valid commands */
    status = iwrite( instId, SYSTEM_QUERY_CMD, strlen(SYSTEM_QUERY_CMD) );
    if (status != 0)
    {
        return GPIB_WRITE_ERROR;
    }

    /* Read system error result from instrument */
    status = iread(instId, system_result, 5);
    if (status != 0)
    {
        return GPIB_READ_ERROR;
    }
```

C Program Listing for Testing VT1.5 Signals Using SCPI Commands, continued

```
    /* convert system result into integer for error code (use as return value) */
    status = atoi( system_result );

    return status;
}

/* Function:   GPIBReadResult
   Purpose:    This function reads results from the system via GPIB interface.
   Return:     0 if Success or error code.
*/
static int GPIBReadResult( int instId, char *result, unsigned int cnt )
{
    int status;/* GPIB I/O and system status */

    /* Read system error result from instrument */
    status = iread(instId, result, cnt);
    if (status != 0)
    {
        return GPIB_READ_ERROR;
    }

    return status;
}

/* Function:   ProcessErrors
   Purpose:    This is a generic function that process GPIB command or I/O errors.
   Return:     void
*/
static void ProcessErrors( int error_code )
{
    /* Generic routines to process GPIB Command or I/O errors */
}
```

Testing a DS3 Dropped from a SONET Signal

This example shows how to transmit a SONET test signal with each STS1 containing DS3 patterns. The transmit signal rate is the *RST default (OC-12 or OC-3, depending on test set configuration.) Receive and measure the DS3 signal, which is dropped from the SONET signal by the multiplexer under test.

Using SCPI Commands

The following example shows how to complete this test using SCPI commands issued in HP-IB mode:

Testing a Dropped DS3 Signal Using SCPI Commands

SCPI Command	Description
`*RST`	*Accesses HP-IB mode and sets the parameters to their default settings.*
`:ROUT:SEL DS3_TX,SOURCE`	*Selects the internal pattern generator for the DS3 payload.*
`:ROUT:SEL DS3_RX,BIPOLAR`	*Sets the DS3 Rx selector to take its signal from the bipolar DS3 receive port.*
`:INIT`	*Starts the test. (Notice the green RUN LED on the test set.)*
`:FETC:DMOD:DS3? SIGNAL`	*Requests the signal status.* *NOTE: A response of +1 for each of the status indicators tells you that the receive signal is present and the dropped DS3 signals has the expected framing type. A response of +0 for the STS-1 Loss Of Pointer alarm indicates that the STS-1 pointer is valid (no alarm condition).*
{read bus> DS3 signal status}	*Sends the signal status to the output queue.*
`:FETC:DMOD:DS3? M13FRAME`	*Requests a DS3 M13 frame detect status.*
{read bus> DS3 M13 frame detect status}	*Sends the frame detect status to the output queue.*
`:FETC:DMOD:DS3? PATTERN`	*Requests the DS3 pattern sync status.*
{read bus> STS1 pattern status}	*Sends the pattern status to the output queue.*
`:SOUR:DM:EINJ:DS3 DATA,1E-3`	*Injects DS3 errors.*
{wait 1S}	*Waits 1 second to allow errors to accumulate.*

Testing a Dropped DS3 Signal Using SCPI Commands, continued

SCPI Command	Description
`:FETC:DMOD:DS3? BIT`	*Requests the DS3 bit error count.*
{read bus> DS3 bit error count}	*Sends the bit error count to the output queue.*
`:FETC:DMOD:DS3? BIT_ARATIO`	*Requests a DS3 bit error average ratio.*
{read bus> DS3 bit error average ratio}	*Sends the ratio to the output queue.*
`:FETC:DMOD:DS3? BIT_CRATIO`	*Requests a DS3 bit error current ratio.*
{read bus> DS3 bit error current ratio}	*Sends the ratio to the output queue.*
`:FETC:DMOD:DS3? BIT_SES`	*Requests the number of DS3 severely errored seconds.*
{read bus> DS3 severely errored seconds}	*Sends the number of severely errored seconds to the output queue.*
`:ABOR`	*Ends the test. (Notice the green RUN LED turn off on the test set.)*
`:SYST:PRES`	*Takes the test set out of HP-IB mode and returns local control.*

Using C

The following example shows how to complete this test using SCPI commands within a C program:

Testing a Dropped DS3 Signal Using C Commands

```
/*****************************************************************************
        This example shows how to transmit a SONET test signal with each STS1
        containing DS3 patterns.
******************************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define GPIB_WRITE_ERROR -1
#define GPIB_READ_ERROR -2
#define SYSTEM_QUERY_CMD"SYST:ERR?"

/* ** External Functions ** */

/* Note, the following are vendor specific GPIB I/O functions.  The functions are generic.  Therefore, the parameters
                and/or function name may not match the functions provided by specific vendors.
*/

/* Function:    iopen
   Purpose:     This function establishes a session with the desired interface or deice and returns a session
                identifier.  The session identifier should only be passed as a parameter to other vendor specific GPIB
                functions.
   Return:      0 if Success or error code.
*/
extern int iopen( char *address, int *Id);
/* Function:    iwrite
   Purpose:     This is a vendor specific function that writes SCPI commands to the system via GPIB interface.
   Return:      0 if Success or error code.
*/
extern int iwrite( int id, char *data, unsigned int datalen );

/* Function:   iread
   Purpose:     This is a vendor specific function that reads results from the system via GPIB interface.
   Return:      0 if Success or error code.
*/
extern int iread( int id, char *buffer, unsigned int bufsize );

/* ** Local Functions ** */

static int GPIBSendCommand( int instId, char *command );
static int GPIBReadResult( int instId, char *result, unsigned int cnt );
static void ProcessErrors( int error_code );
```

Testing a Dropped DS3 Signal Using C Commands, continued

```
/* ** Main Program  ** */
/*
        Note, This example will exit the program when error occurs.
*/
void main ()
{
        int instId;/* session identifier */
        int status;/* GPIB I/O and system status */
        char result[50];/* result from instrument */

        /* establish session between GPIB interface and instrument */
        status = iopen( "HPIB, 29", &instId );
        if (status != 0)
        {
              ProcessErrors( status );
              return;
        }

        /* Initialize the instrument */
        status = GPIBSendCommand( instId, "*RST" );
        if (status != 0)
        {
              ProcessErrors( status );
              return;
        }

        /*  selects the internal pattern generator for the DS3 payload*/
        status = GPIBSendCommand( instId, "ROUT:SEL DS3_TX, SOURCE" );
        if (status != 0)
        {
              ProcessErrors( status );
              return;
        }
        /* select the DS3 Rx selector to take its signal from the bipolar DS3 port */
        status = GPIBSendCommand( instId, "ROUT:SEL DS3_RX, BIPOLAR" );
        if (status != 0)
        {
              ProcessErrors( status );
              return;
        }

        /* start the test */
        status = GPIBSendCommand( instId, "INIT" );
        if (status != 0)
        {
              ProcessErrors( status );
              return;
        }
```

Testing a Dropped DS3 Signal Using C Commands, continued

```
/*  request signal status*/
status = GPIBSendCommand( instId, "FETC:DMOD:DS3? SIGNAL" );
if (status != 0)
{
      ProcessErrors( status );
      return;
}

/* Read  signal status*/
status = GPIBReadResult( instId, result, 50 );
if (status != 0)
{
      ProcessErrors( status );
      return;
}

/*  request DS3 M13 frame detect status */
status = GPIBSendCommand( instId, "FETC:DMOD:DS3? M13FRAME" );
if (status != 0)
{
      ProcessErrors( status );
      return;
}

/* Read  DS3 M13 frame detect status*/
status = GPIBReadResult( instId, result, 50 );
if (status != 0)
{
      ProcessErrors( status );
      return;
}

/*  request DS3 pattern sync status */
status = GPIBSendCommand( instId, "FETC:DMOD:DS3? PATTERN" );
if (status != 0)
{
      ProcessErrors( status );
      return;
}

/* Read  DS3 pattern status */
status = GPIBReadResult( instId, result, 50 );
if (status != 0)
{
      ProcessErrors( status );
      return;
}
```

Testing a Dropped DS3 Signal Using C Commands, continued

```
/* Inject DS3 errors */
status = GPIBSendCommand( instId, "SOUR:DM:EINJ:DS3 data, 1E-3" );
if (status != 0)
{
    ProcessErrors( status );
    return;
}

/* wait 1 second */

/* request  DS3 bit error count */
status = GPIBSendCommand( instId, "FETC:DMOD:DS3? BIT" );
if (status != 0)
{
    ProcessErrors( status );
    return;
}

/* Read  DS3 bit error count */
status = GPIBReadResult( instId, result, 50 );
if (status != 0)
{
    ProcessErrors( status );
    return;
}

/* Request DS3 bit error average ratio */
status = GPIBSendCommand( instId, "FETC:DMOD:DS3? BIT_ARATIO" );
if (status != 0)
{
    ProcessErrors( status );
    return;
}

/* Read DS3 bit error average ratio */
status = GPIBReadResult( instId, result, 50 );
if (status != 0)
{
    ProcessErrors( status );
    return;
}

/* Request DS3 bit error current ratio */
status = GPIBSendCommand( instId, "FETC:DMOD:DS3? BIT_CRATIO" );
if (status != 0)
{
    ProcessErrors( status );
    return;
}
```

Testing a Dropped DS3 Signal Using C Commands, continued

```
    /* Read DS3 bit error current ratio */
    status = GPIBReadResult( instId, result, 50 );
    if (status != 0)
    {
        ProcessErrors( status );
        return;
    }

    /* Request DS3 severely error seconds */
    status = GPIBSendCommand( instId, "FETC:DMOD:DS3? BIT_SES" );
    if (status != 0)
    {
        ProcessErrors( status );
        return;
    }

    /* Read DS3 bit error seconds */
    status = GPIBReadResult( instId, result, 50 );
    if (status != 0)
    {
        ProcessErrors( status );
        return;
    }

    /* End Test */
    status = GPIBSendCommand( instId, "ABORT" );
    if (status != 0)
    {
        ProcessErrors( status );
        return;
    }

    /* Take test set out of HP-IB mode */
    status = GPIBSendCommand( instId, "SYST:PRES" );
    if (status != 0)
    {
        ProcessErrors( status );
        return;
    }
}

/* Function:   GPIBSendCommand
   Purpose:    This function writes SCPI commands to the system and verifies the command is acceptable by the
               system.
   Return:     0 if Success or error code.
*/
static int GPIBSendCommand( int instId, char *command )
{
    char system_result[5];      /* SYST ERROR query buffer */
    int status;/* GPIB I/O and system status */
```

Testing a Dropped DS3 Signal Using C Commands, continued

```
    /* ** send msg string to instrument and verify GPIB status ** */

    /* send command to instrument */
    status = iwrite( instId, command, strlen(command) );
    if (status != 0)
    {
        return GPIB_WRITE_ERROR;
    }
    /* send SYSTEM ERROR QUERY command in order to verify valid commands */
    status = iwrite( instId, SYSTEM_QUERY_CMD, strlen(SYSTEM_QUERY_CMD) );
    if (status != 0)
    {
        return GPIB_WRITE_ERROR;
    }

    /* Read system error result from instrument */
    status = iread(instId, system_result, 5);
    if (status != 0)
    {
        return GPIB_READ_ERROR;
    }

    /* convert system result into integer for error code (use as return value) */
    status = atoi( system_result );

    return status;
}

/* Function:   GPIBReadResult
   Purpose:     This function reads results from the system via GPIB interface.
   Return:       0 if Success or error code.
*/
static int GPIBReadResult( int instId, char *result, unsigned int cnt )
{
    int status;/* GPIB I/O and system status */

    /* Read system error result from instrument */
    status = iread(instId, result, cnt);
    if (status != 0)
    {
        return GPIB_READ_ERROR;
    }

    return status;
}
```

Testing a Dropped DS3 Signal Using C Commands, continued

```
/* Function:   ProcessErrors
   Purpose:     This is a generic function that process GPIB command or I/O errors.
   Return:      void
*/
static void ProcessErrors( int error_code )
{
        /* Generic routines to process GPIB Command or I/O errors */
}
```

Simultaneous Independent DS1 and SONET Transmission Testing

This example shows how to transmit a test signal at OC-3 or OC-12 (depending on the reset default), with all STS-1s containing DS3 patterns. Receive the signal and drop and measure the DS3 in the first STS-1. Simultaneously, run an independent DS1 test using the bipolar DS1 transmit and receive ports.

Using SCPI Commands

The following example shows how to complete this test using SCPI commands issued in HP-IB mode:

DS1 and SONET Transmission Testing Using SCPI Commands

SCPI Command	Description
`*RST`	*Accesses HP-IB mode and sets the parameters to their defaults.*
`:ROUT:SEL DS3_TX,SOURCE`	*Selects the internal DS3 pattern generator for the DS3 transmit payload.*
`:ROUT:SEL DS1_RX,BIPOLAR`	*Directs the DS1 bipolar receive port to the DS1 measurement circuits.*
`:INIT`	*Starts the test. (Notice the green RUN LED on the test set.)*
`:FETC:DMOD:STS1? SIGNAL`	*Requests the SONET signal status.* *NOTE: A response of +1 for each of the status indicators tells you that the receive signal is present and the dropped DS3 and DS1 signals have the expected framing types. A response of +0 for the STS-1 Loss Of Pointer alarm indicates that the STS-1 pointer is valid (no alarm condition).*
{read bus> STS-1 signal status}	*Sends the signal status to the output queue.*
`:FETC:DMOD:STS1? LOPNTR`	*Requests an STS-1 Loss of Pointer alarm indication.*
{read bus> STS-1 Loss of Pointer alarm}	*Sends the alarm indication to the output queue.*
`:FETC:DMOD:DS3? M13FRAME`	*Requests the DS3 M13 frame detect status.*
{read bus> DS3 m13bit frame detect status}	*Sends the status to the output queue.*
`:FETC:DMOD:DS1? SFSYNC`	*Requests the DS1 SF frame detect status.*

DS1 and SONET Transmission Testing Using SCPI Commands, continued

SCPI Command	Description
{read bus> DS1 SF frame detect status}	*Sends the status to the output queue.*
`:SOUR:DM:EINJ:DS3 DATA,1E-3`	*Injects DS3 errors.*
{wait 1S}	*Waits 1 second to allow errors to accumulate.*
`:FETC:DMOD:DS3? BIT`	*Requests the DS3 bit error count.*
{read bus> DS3 bit error count}	*Sends the bit error count to the output queue.*
`:FETC:DMOD:DS3? BIT_ARATIO`	*Requests a DS3 bit error average ratio.*
{read bus> DS3 bit error average ratio}	*Sends the ratio to the output queue.*
`:FETC:DMOD:DS3? BIT_CRATIO`	*Requests a DS3 bit error current ratio.*
{read bus> DS3 bit error current ratio}	*Sends the ratio to the output queue.*
`:FETC:DMOD:DS3? BIT_SES`	*Requests the number of DS3 severely errored seconds.*
{read bus> DS3 severely errored seconds}	*Sends the number of severely errored seconds to the output queue.*
`:ABOR`	*Ends the test. (Notice the green RUN LED turn off on the test set.)*
`:SYST:PRES`	*Takes the test set out of HP-IB mode and returns local control.*

Using C

The following example shows how to complete this test using SCPI commands within a C program:

DS1 and SONET Transmission Testing Using C

```
/******************************************************************************
	This function shows how to transmit a test signal at OC3 or OC12, with all
	STS1s containing DS3 patterns.
******************************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define GPIB_WRITE_ERROR -1
#define GPIB_READ_ERROR -2
#define SYSTEM_QUERY_CMD"SYST:ERR?"

/* ** External Functions ** */

/* Note, the following are vendor specific GPIB I/O functions.  The functions are generic.  Therefore, the parameters
		and/or function name may not match the functions provided by specific vendors.
*/

/* Function:	open
   Purpose:	This function establishes a session with the desired interface or deice and returns a session
		identifier.  The session identifier should only be passed as a parameter to other vendor specific GPIB
		functions.
   Return:	0 if Success or error code.
*/
extern int iopen( char *address, int *Id);
/* Function:	iwrite
   Purpose:	This is a vendor specific function that writes SCPI commands to the system via GPIB interface.
   Return:	0 if Success or error code.
*/
extern int iwrite( int id, char *data, unsigned int datalen );

/* Function:	iread
   Purpose:	This is a vendor specific function that reads results from the system via GPIB interface.
   Return:	0 if Success or error code.
*/
extern int iread( int id, char *buffer, unsigned int bufsize );

/* ** Local Functions ** */

static int GPIBSendCommand( int instId, char *command );
static int GPIBReadResult( int instId, char *result, unsigned int cnt );
static void ProcessErrors( int error_code );
```

DS1 and SONET Transmission Testing Using C, continued

```
/* ** Main Program  ** */
/*
      Note, This example will exit the program when error occurs.
*/
void main ()
{
      int instId;/* session identifier */
      int status;/* GPIB I/O and system status */
      char result[50];/* result from instrument */

      /* establish session between GPIB interface and instrument */
      status = iopen( "HPIB, 29", &instId );
      if (status != 0)
      {
            ProcessErrors( status );
            return;
      }

      /* Initialize the instrument */
      status = GPIBSendCommand( instId, "*RST" );
      if (status != 0)
      {
            ProcessErrors( status );
            return;
      }
      /*  selects the internal pattern generator for the DS3 payload*/
      status = GPIBSendCommand( instId, "ROUT:SEL DS3_TX, SOURCE" );
      if (status != 0)
      {
            ProcessErrors( status );
            return;
      }

      /* select the DS3 Rx selector to take its signal from the bipolar DS3 port */
      status = GPIBSendCommand( instId, "ROUT:SEL DS3_RX, BIPOLAR" );
      if (status != 0)
      {
            ProcessErrors( status );
            return;
      }

      /* start the test */
      status = GPIBSendCommand( instId, "INIT" );
      if (status != 0)
      {
            ProcessErrors( status );
            return;
      }
```

DS1 and SONET Transmission Testing Using C, continued

```
/* Request a SONET signal status */
status = GPIBSendCommand( instId, "FETC:DMOD:STS1? SIGNAL" );
if (status != 0)
{
    ProcessErrors( status );
    return;
}

/* Read STS1 signal status */
status = GPIBReadResult( instId, result, 50 );
if (status != 0)
{
    ProcessErrors( status );
    return;
}

/* Request STS1 Loss of Pointer alarm indication */
status = GPIBSendCommand( instId, "FETC:DMOD:STS1? LOPNTR" );
if (status != 0)
{
    ProcessErrors( status );
    return;
}

/* Read STS1 Loss of Pointer alarm */
status = GPIBReadResult( instId, result, 50 );
if (status != 0)
{
    ProcessErrors( status );
    return;
}

/* Request a DS3 M13 frame detect status */
status = GPIBSendCommand( instId, "FETC:DMOD:DS3? M13FRAME" );
if (status != 0)
{
    ProcessErrors( status );
    return;
}

/* Read a DS3 M13 frame detect status */
status = GPIBReadResult( instId, result, 50 );
if (status != 0)
{
    ProcessErrors( status );
    return;
}
```

DS1 and SONET Transmission Testing Using C, continued

```
/* Request a DS1 SF frame detect status */
status = GPIBSendCommand( instId, "FETC:DMOD:DS1? SFSYNC" );
if (status != 0)
{
      ProcessErrors( status );
      return;
}

/* Read a a DS1 SF frame detect status */
status = GPIBReadResult( instId, result, 50 );
if (status != 0)
{
      ProcessErrors( status );
      return;
}

/* Inject DS3 errors */
status = GPIBSendCommand( instId, "SOUR:DM:EINJ:DS3 data, 1E-3" );
if (status != 0)
{
      ProcessErrors( status );
      return;
}

/* wait 1 second */

/* request  DS3 bit error count */
status = GPIBSendCommand( instId, "FETC:DMOD:DS3? BIT" );
if (status != 0)
{
      ProcessErrors( status );
      return;
}

/* Read  DS3 bit error count */
status = GPIBReadResult( instId, result, 50 );
if (status != 0)
{
      ProcessErrors( status );
      return;
}

/* Request DS3 bit error average ratio */
status = GPIBSendCommand( instId, "FETC:DMOD:DS3? BIT_ARATIO" );
if (status != 0)
{
      ProcessErrors( status );
      return;
}
```

DS1 and SONET Transmission Testing Using C, continued

```
/* Read DS3 bit error average ratio */
status = GPIBReadResult( instId, result, 50 );
if (status != 0)
{
    ProcessErrors( status );
    return;
}

/* Request DS3 bit error current ratio */
status = GPIBSendCommand( instId, "FETC:DMOD:DS3? BIT_CRATIO" );
if (status != 0)
{
    ProcessErrors( status );
    return;
}

/* Read DS3 bit error current ratio */
status = GPIBReadResult( instId, result, 50 );
if (status != 0)
{
    ProcessErrors( status );
    return;
}

/* Request DS3 severely error seconds */
status = GPIBSendCommand( instId, "FETC:DMOD:DS3? BIT_SES" );
if (status != 0)
{
    ProcessErrors( status );
    return;
}

/* Read DS3 bit error seconds */
status = GPIBReadResult( instId, result, 50 );
if (status != 0)
{
    ProcessErrors( status );
    return;
}

/* End Test */
status = GPIBSendCommand( instId, "ABORT" );
if (status != 0)
{
    ProcessErrors( status );
    return;
}
```

DS1 and SONET Transmission Testing Using C, continued

```
    /* Take test set out of HP-IB mode */
    status = GPIBSendCommand( instId, "SYST:PRES" );
    if (status != 0)
    {
        ProcessErrors( status );
        return;
    }
}

/* Function:  GPIBSendCommand
   Purpose:   This function writes SCPI commands to the system and verifies the command is acceptable by the
              system.
   Return:    0 if Success or error code.
*/
static int GPIBSendCommand( int instId, char *command )
{
    char system_result[5];      /* SYST ERROR query buffer */
    int status;/* GPIB I/O and system status */

    /* ** send msg string to instrument and verify GPIB status ** */

    /* send command to instrument */
    status = iwrite( instId, command, strlen(command) );
    if (status != 0)
    {
        return GPIB_WRITE_ERROR;
    }

    /* send SYSTEM ERROR QUERY command in order to verify valid commands */
    status = iwrite( instId, SYSTEM_QUERY_CMD, strlen(SYSTEM_QUERY_CMD) );
    if (status != 0)
    {
        return GPIB_WRITE_ERROR;
    }

    /* Read system error result from instrument */
    status = iread(instId, system_result, 5);
    if (status != 0)
    {
        return GPIB_READ_ERROR;
    }

    /* convert system result into integer for error code (use as return value) */
    status = atoi( system_result );

    return status;
}
```

DS1 and SONET Transmission Testing Using C, continued

```
/* Function:   GPIBReadResult
   Purpose:     This function reads results from the system via GPIB interface.
   Return:      0 if Success or error code.
*/
static int GPIBReadResult( int instId, char *result, unsigned int cnt )
{
        int status;/* GPIB I/O and system status */

        /* Read system error result from instrument */
        status = iread(instId, result, cnt);
        if (status != 0)
        {
              return GPIB_READ_ERROR;
        }
        return status;
}

/* Function:   ProcessErrors
   Purpose:     This is a generic function that process GPIB command or I/O errors.
   Return:      void
*/
static void ProcessErrors( int error_code )
{
        /* Generic routines to process GPIB Command or I/O errors */
}
```

17

About SCPI

The SCPI Interface

Controller refers to a PC or terminal that is connected to a test instrument.

Standard Commands for Programmable Instruments (SCPI) is a command language used to control electronic test and measurement instruments. SCPI commands are sent from a controller to an instrument to configure the instrument, perform tests, and gather data. The functions available from the instrument's front panel controls are available using SCPI. In addition SCPI goes beyond the front panel controls, allowing more flexible configuration options by providing complete access to the set's functionality.

IEEE-488.2 and SCPI

The IEEE-488.2 standard describes how to send commands to an instrument and send responses to a controller. It also defines a group of frequently used "housekeeping" commands to be accepted by most SCPI-compliant instruments, and specifies how some types of features should be implemented in SCPI firmware.

Although the IEEE-488.2 standard covers many aspects of communication between a controller and an instrument, it does not specify which commands or features should be supported by each type of instrument. Test sets from different manufacturers that conform to the IEEE-488.2 standard may not be able to communicate with each other because they do not support the same command set.

SCPI promotes a uniform and consistent programming environment for instrument control and data usage over a IEEE-488 bus. Consistency is achieved by defining the program messages, instrument responses, and data formats to use when communicating with SCPI instruments.

HP-IP Standard

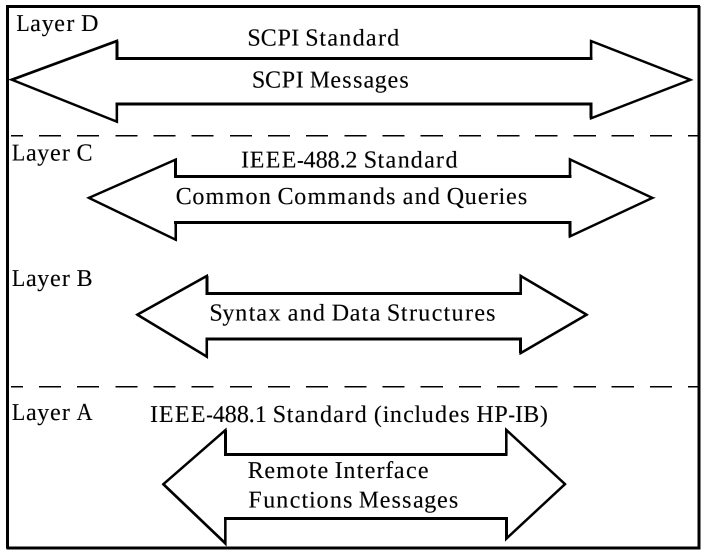
The Hewlett-Packard Interface Bus (HP-IB) is based on the IEEE-488.1 standard and defines the type of physical and electrical connection to use when connecting a test instrument to a controller, how the devices talk and listen to each other, and how a byte of data is transmitted back and forth between a controller and an instrument.

The HP-IP cable connected between a controller and an instrument, and the HP-IP interface card that you install in a controller adhere to this interface standard.

Note: *The terms GPIB (General Purpose Interface Bus), IEEE-488 bus, and HP-IB are used interchangeably in this book to refer to the communication between a controller and a CERJAC test set.*

Layering of the Standards

The SCPI/HP-IB interface is based on four successive, independent levels of communication protocol. Each layer supports it own set of common messages. These messages form the independent protocols used to communicate within the layer and with the corresponding layers. The following diagram shows the succession of layers:



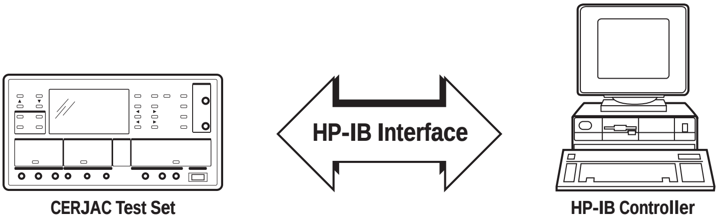
The following table summarizes the protocol layers.

SCPI/HP-IB Communications Protocol Layers

Layer	Standard	Specifies
A	*SCPI*	*The commands that control each programmable function in an instrument.* *The types of parameters that you can use with these commands.* *The types of values that these commands return.*
B & C	*IEEE-488.2*	*The syntax to use to send data between devices.* *The way the devices format and interpret data.* *The registers that should exist inside an instrument for the purpose of recording its internal state.* *The commands to use to do common tasks.*
D	*HP-IB (IEEE-488.1) (GPIB)*	*The physical connection between devices.* *The electrical connection between devices.* *How a byte of data is transmitted.* *How devices are instructed to talk and listed.*

Interface Connections

The CERJAC test sets accept SCPI commands through the HP-IB or RS-232 port. When you connect an instrument to a controller you can use an HP-IP interface card and an HP-IB cable, or an RS-232 serial cable and a terminal or terminal emulation package.



HP-IB Interface

HP-IB, a parallel interface, can be used to connect one or many SCPI instruments to a controller. To use this interface, your controller must also have an HP-IB interface (for example, an HP-IB interface card installed in a PC). An HP-IP interface cable is then used to connect the instrument to the controller.

If you want to connect multiple test sets to a controller, connect one instrument to the controller and daisy-chain the remaining sets together (as many as 30). See *Using an GPIB Interface*, page 2–3 for connection procedures.

RS-232 Interface

You can use an RS-232 serial interface to connect a single instrument to a controller. To communicate with the instrument, you will also need a terminal, or a PC running a terminal emulator or other SCPI interface application.

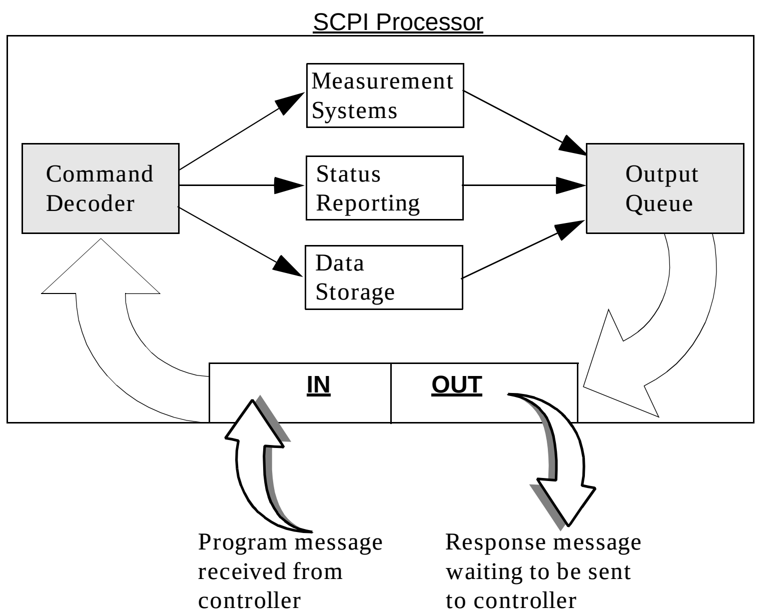
The commands and responses are identical whether you use this interface or the HP-IB interface. The primary difference is that the RS-232 interface transmits responses as soon as they are ready, rather than placing them in an output queue. See *Using an RS-232 Interface*, page 2–4 for connection procedures.

Also, see test set interface cable information on Page 27-6 of the 156MTS SONET User Manual.

Sending SCPI Messages

When you send a program message from a controller to an instrument, the instrument's SCPI processor parses (decodes) the message to determine which command it received. The instrument then carries out the command. If the command requires data to be sent back to the controller, the data is buffered in the output queue until it is requested by the controller.

The following diagram shows the process a command follows through the SCPI processor.



18

Programming Diagrams

Programming Diagrams

This chapter presents a graphic model of your CERJAC test set as it is controlled by the SCPI command set. The programming diagrams provide a visual reference to help you configure the instrument. Each element in the Receive Programming Diagram on page 18–4, and in the Transmit Programming Diagram on page 18–5, represents a SCPI command and its applicable parameters.

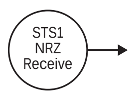
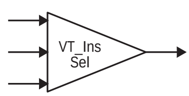
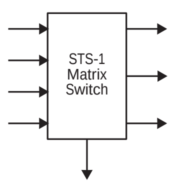
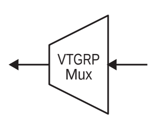
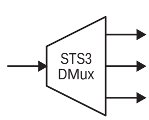
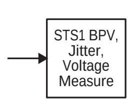
Using the Programming Diagrams

The following steps explain how to use the programming diagrams to help you generate SCPI programs.

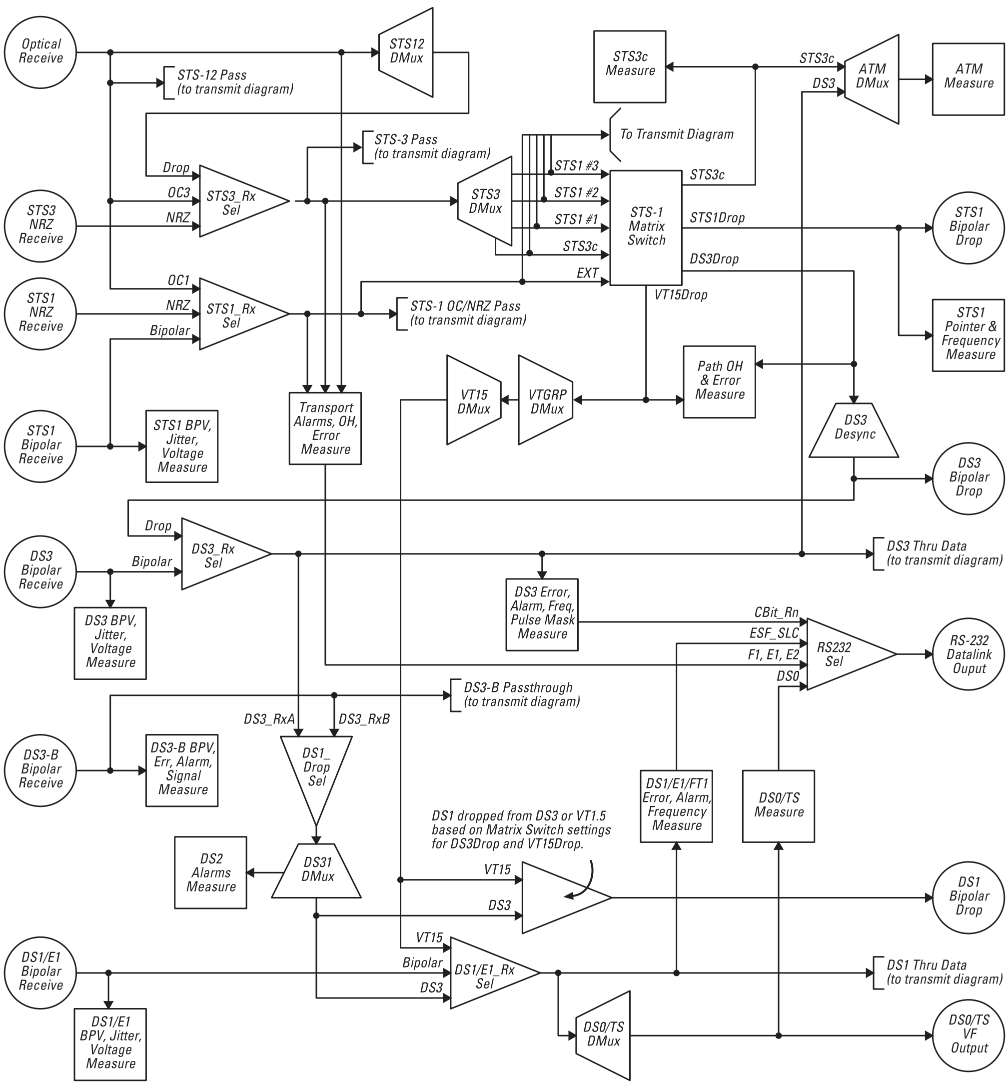
1. Make copies of the transmit and receive programming diagrams.
2. On each diagram, highlight the desired signal routes, and circle the appropriate :SOURce choices and the :SENSe boxes you want to use.
3. Confirm that the highlighted paths connect the sources to the desired outputs, and the inputs to the desired :SENSe boxes.
4. With these annotated diagrams in hand, refer to the appropriate chapters of this manual to determine the specific :ROUTe, :SENSe, and :SOURce commands required to implement the system setup.

Key to Diagrams

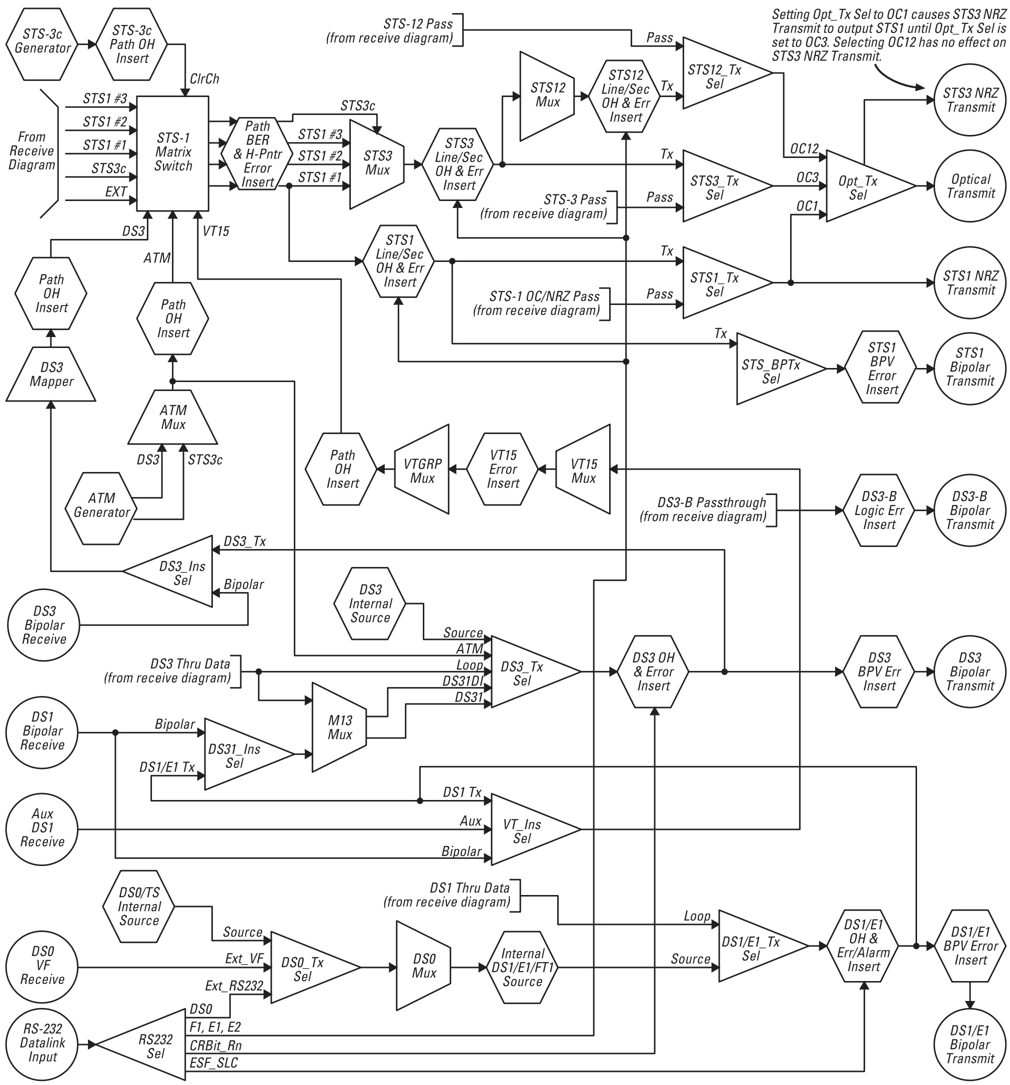
The programming diagrams are made up of the following elements:

Element	Symbol	Description
Sources		*Internal signal or pattern generators and error injection elements. Sources are controlled using :SOURce commands.*
I/O Ports		*Signal input or output connections on the test set, such as front-panel jacks, and so on. Ports are controlled using :OUTPut and :INPut commands.*
Selectors		*Switching elements that select a signal path from various choices. Selectors are controlled using :ROUTe:SELect commands.*
Matrix Switch		*A unique switching element with multiple inputs and outputs. Some functions of the STS-1 matrix switch are shown on the receive diagram and some are shown on the transmit diagram. The matrix switch is controlled using :ROUTe:MATRix commands.*
Multiplexers		*Circuitry that combines low rate signals into a composite higher-rate signal. Multiplexers are controlled using :ROUTe:MUX commands.*
Demultiplexers		*Circuitry that drops a selected lower-rate channel from a higher-rate signal. Demultiplexers are controlled using :ROUTe:DMUX commands.*
Sensors		*Elements which perform one or more measurements. Sensors are controlled using :SENSe commands. Results of measurements performed by sensors are retrieved using :FETCh commands.*

Receive Programming Diagram



Transmit Programming Diagram



A

Appendix A - Software Release Notes

Software Release Notes

156MTS HOST CODE V7.00 Release Notes

2. Added Monitor mode OC3-Mon (STS1/ATM).
3. Added Drop and Insert mode OC3-D&I (STS1/ATM).
4. Added Terminal mode OC12-OC12 (STS1/ATM).
5. Expanded Auto Setup to include SDH, OC3-OC3 (STS1/ATM), and OC12-OC12 (STS1/ATM) detection.
6. Store/Recall Feature added to SCPI Mode
7. ATM VPI/VCI Scan Capability added to SCPI Mode
8. Timed Test Duration in SCPI Mode

Fixes:

1. Corrected problem with ESF FDL data intepretation.
2. SCPI commands returns invalid data for functions not previously set has been corrected by initializing all SCPI queries structures to zero.
3. SCPI command **:SYST:REMOTE** was corrected.
4. E1 TS mode with drop/insert set to 29 to 31, then switching to a DS1 mode, the DS0 setup screens drop and insert field settings were corrected.
5. Removed the “Aux” pick from field “Others>” in all modes with a VT1.5A payload.
6. Fixed problem in the H pointer increment test for mode OC12-OC12 (STS12); where error inject and local action key appears to be locked.
7. DS2 OOF (out of frame) error while recovering from a power interruption.
8. Fixed SCPI commands :**`SOUR:DM:EINJ:STS1 A1A2 | B1 | SECT_BER | LINE_BER`**
9. Fixed H Pointer operation in STS1-STS1 (ATM). Added OC3-OC3 (STS1/ATM) and OC12-OC12 (STS1/ATM).
10. Fixed mode with VT1.5b payload following factory defaults.
11. Corrected OC12-D&I (DS3/1) and OC3-D&I (DS3/1) modes.
12. Changed J0, J1, and J2 from Hewlett Packard to Agilent Technologies.
13. Fixed SCPI command **`:SOUR:DM:ALARM DS1_YEL.`**
14. Fixed SCPI commands **`:SOUR:DM:EINJ:DS3 FRAME, BURST`** and, :**`SOUR:DM:EINJ:DS3 FRAME,OFF.`**

15. Fixed STS frame measurements (A1/A2) results screen.

16. Corrected A1/A2 SEFS detection for OC3 and STS.

17. Added the SCPI command **`:SENSe:DM:DATA3 BITER_ON`** to display DS3 error bits. Also, SCPI command **`:SENSe:DM:DATA3 BITER_OFF`** inhibits error display.

18. Upon entering SCPI mode, all display results get set to off and must be reenabled.

156MTS Host Software Version 6.8 Release Notes

Product Number E4480A

The 156MTS SONET Maintenance Test Set Host Code Version 6.80 is a new release of the application software for the E4480A 156MTS SONET Maintenance Test Set.

Release Overview

This new version of the 156MTS SONET Maintenance Test Set software incorporates several significant improvements.

SCPI Command Fixes and Additions

For each new query command shown below, the SCPI/HPIB User's Manual documentation has been updated to incorporate a question mark (?). The question mark has been added to the command to query the test set for its current parameter values.

1. Two SCPI commands have been added to return the DS3 and DS1 frame type settings. The commands are:

   `:SOURce:DM:FRAMe:TYPE1?` Returns DS1 frame setting, and

   `:SOURce:DM:FRAMe:TYPE3?` Returns DS3 frame setting.

2. A new SCPI command has been added to return the VT1.5 mode setting (either asynchronous or byte-synchronous). The command syntax is:

   `:SOURce:DM:VT15?`

3. Five new SCPI commands have been added to return the input filter settings, as follows:

   `:INput:FILTer:DS1?`

   `:INput:FILTer:DS3?`

   `:INput:FILTer:DS3B?`

   `:INput:FILTer:STS1?`

   `:INput:FILTer:E1?`

4. A following SCPI command has been added to return the setting of the DS1/E1 input port termination:

   `:INput:TYPE?`

5. Three new SCPI commands have been added to return the setting of the transmit port output:

   `:OUTPut:FILTer:DS1?`

:OUTPut:FILTer:DS3?

:OUTPut:FILTer:STS1?

6. Four new SCPI commands have been added to return the ON or OFF setting of the transmit outputs:

   :OUTPut:STATe1? Returns state of the DS1 output.

   :OUTPut:STATe2? Returns state of the STS1 output.

   :OUTPut:STATe3? Returns state of the DS3 output.

   :OUTPut:STATe4? Returns state of the optical output.

7. A new SCPI command returns the identity of the channel to be dropped from a demultiplexer:

   :ROUTe:DMUX? DS0|DS31|VTGRP|VT15|STS12

8. A SCPI command has been added to return the identity of the channel inserted in a multiplexer:

   :ROUTe:MUX:INS? DS0|M13|VT15|VTGRP|STS12

9. A new SCPI command returns the setting of the remaining channels in the multiplexer:

   :ROUTe:MUX:OTH? DS0|M13|VT15|VTGRP|STS12

10. A SCPI command has been added to return the setting of the DS1 line code:

    :SENSe:DM:CODE1?

11. A SCPI command has been added to return the inhibit error on alarm setting:

    :SENSe:DM:DATA:IERR?

12. Another new SCPI command to return the on/off setting of Trouble Scan function:

    :SENSe:DM:DATA:TRBLscan?

13. Three new SCPI commands were added to return framing type specified by the :**SENSe:DM:FRAMe** command:

    :SENSe:DM:FRAMe:TYPE1?

    :SENSe:DM:FRAMe:TYPE3?

:SENSe:DM:FRAMe:TYPE3B?

14. The following new SCPI command returns the DS3B LED sharing mode:

    :SENSe:DM:DS3B? LEDS

15. A new SCPI command to return the ATM DMUX mapping setting:

    :SENSe:DM:ATMDMUX?

16. A new SCPI command has been added to return the selected jitter measurement rate:

    :SENSe:DM:JITTer? RATE|WB_THRESH|HB_THRESH

17. A new SCPI command has been added to return the DS3 pulse mask setting:

    :SENSe:DM:MSKSEL?

18. A new SCPI command was added to return the pulse mask polarity setting:

    :SENSe:DM:MSKPOL?

19. A SCPI command was added to return the pulse mask tolerance setting:

    :SENSe:DM:MSKTOL?

20. A SCPI command was added to return the setting of the pulse mask stop/start switch:

    :SENSe:DM:MSK?

21. A new SCPI command was added to return the setting of the ATM physical layer mapping:

    :SOURce:DM:ATMMUX?

22. A new SCPI command returns the settings of the ATM Layer Generator:

    :SOURce:DM:ATMGEN? <ATM param>

23. A new SCPI command was added to return the setting of the DS1 input impedance:

    :INPUT:IMPedance?

SCPI Manual, Text Changes Only

The following H1 byte overhead command exists but were not previously documented in this manual. These commands set and query SONET overhead H1 byte. The commands are:

:SOURce:DM:OHH1, <value>

where value can be 0 or 1, and

:SOURce:DM:OHH1?

The following query commands (?) exist but were not previously documented in this manual. these commands have been added to the SCPI documentation

:SOURce:DM:PROG1 PATTERN,<pattern>?

:SOURce:DM:PROG3 PATTERN,<pattern>?

:SOURce:DM:PROG32 PATTERN,<pattern>?

:ROUTe:MATRix <output>,<input>?

:SOURce:DM:DS3:FEAC:CBIT?

:SOURce:DM:DS3:FEAC:CONT_ALM_STAT?

:SOURce:DM:DS3:FEAC:BURST_SETTING?

:SOURce:DM:DS3:FEAC:LOOPBACK_LINE?

:SOURce:DM:DS3:FEAC:BURST_SIZE?

:SOURce:DM:DS3:FEAC:ALM_STAT?

:SOURce:DM:FT1?

:SOURce:DM:FE1?

:SOURce:DM:NX?

The following SCPI command for setting the input impedance exists but was not documented in this manual.

:INPUT:IMPedance <value>

Terminal mode is indicated when the value is less than or equal to 100. Bridge mode is indicated when the value is more than 100.

Miscellaneous. Fixes

1. ATM functionality failed following the restoration of default settings using a Setup Menu pick. The problem was caused when the ATM board lost synchronization with the host CPU when the test set was warm started. This problem was corrected by restoring test set default settings on the fly and not warm starting the unit.
2. When the test set was configured for DS1 rate with a DS1 payload in monitor mode, it would not display the DS0 VF Measurement screen. This short coming has been corrected.
3. When the test set is configured for terminal mode STS1/OC1-STS1/OC1 (VT1.5A) and 1 to 28 counting mode with other VT's set to AIS, drop and insert fields set to 7, then 8, and back to 7; the DS1 signal became lost. This problem was corrected
4. A SCPI command was fixed to set the terminal mode state for DS1-DS1 (DS1) with DS0 data. This command changes the state of the ABCD signaling bits.
5. For terminal mode DS3-DS3 (DS3), the test set loses sync when TX clock is set to LOOP. This problem has been corrected.
6. For terminal mode E1-E1 (E1), an unframed data lights the yellow alarm. This problem has Been corrected.

Additions

Support has been added for the new OC12-4 board. which includes board recognition, and loading.

Firmware.

Firmware code was added to recognize and provide electrical STS1 LOS signal detection. This capability is provided only when the OC12-4 board is present. It is not supported by the OC12+ or earlier boards.

Installation

Install only software with a higher version number than the software already in your test set. The distribution diskette includes the appropriate software for your test set.

For More Information

For more information, call Agilent's TNTD at 800-923-7522.

Product Number E4480A

The 156MTS SONET Maintenance Test Set Host Software Version 6.70 is a new release of the software for the E4480A 156MTS SONET Maintenance Test Set.

Release Overview

This new version of the 156MTS SONET Maintenance Test Set

156MTS Host Software Version 6.7 Release

Software Improvements

An Auto Setup problem was fixed in which the unit found an STS1/OC1 with an STS-1/ATM payload but reported “This mapping is supported for OC3C & DS3 transport only".

1. An inability of the Auto Setup function to detect E1 mapping on a single STS in OC3-OC3 (DS3/E1) mode, has been fixed.
2. A key was added to the Auto Setup MENU on the DS1 map screen to allow an escape from Auto Setup.
3. Additional code was added to set STS drop & insert fields for the OC12-OC12 (DS3), & OC3-OC3 (DS3) modes.

The Auto Setup function has been enhanced for modes OC12, OC3, & STS1 and a VT1.5A/B payload.

The new software adds a new menu layer. After VT structured STS1 is chosen, the test set scans and displays 28 DS1 channels. In terminal mode, the unit operator can then pick a DS1 channel to be tested. The insert fields for the STS1, VT group, & VT position have been modified to match the drops and now supports VT group and 1 to 28 counting modes.

An enhanced Auto Setup capability was added for modes that support DS3, DS1, & E1 payloads. The new software adds a new menu layer. Now after choosing DS3 structured STS1, the test set displays 28 DS1 or 21 E1 channels, respectively. In terminal mode the operator can pick a DS1/E1 channel to test. The insert fields were also modified to match the drops.

A tributary frame scan function was added. This function allows the test set to scan each DS1 channel to identify the framing of each DS1, or E1 channel.

The Auto Setup feature was also enhanced by the addition of a data pattern scan capability along with the framing scan function.

Now the operator can choose a channel in the low-level tributary menu, perform a frame scan, or a data scan sequence.

The Auto-Setup function was changed to force the test set to search for signals in descending order from the highest optical rates (OC3) down to the slower electrical rates (STS1).

Additional Auto Setup features and improvements include:

A frame that scan can now be performed at each of the DS1, E1, and VT Auto Setup menu screens. The scan can be aborted by pressing the MENU key once. The operator can then choose to rescan, pick a channel, or go back to the main menu.

Trouble Scan Improvements

Trouble Scan performance has been improved; reducing total scan time from 4.5 to 1.0 seconds.

An ability to filter errors and alarms issued to the front panel has been added through the use of the screen selector fields.

SCPI Command Improvements

Nine new SCPI commands have been added to return errors and alarms by payload category. For example:

:FETCh:TRBL:DS0?
[DS1?|DS2?|DS3?|DS3B?|E1?|VT?|ATM?|SONET?]

This manual has been updated to indicate that the host software supports a SCPI command that enables the DS3 error out signal. The SCPI command is as follows:

:ROUTe:SELect BIT_ERROUT OFF|DS3|STS3C|STS12C

Additionally, the DS1 error out jack is active, both on the front panel and in SCPI command mode.

This signal is always active when DS1 bit errors are present. There is no corresponding SCPI command to control the DS1 error out so the above SCPI commands has been added to this manual.

The SCPI commands have been reworked to match SCPI **:SOURCE:DM:CLOCKn** command set in the documentation in this manual. The updated SCPI command structure is as follows:

:SOURce:DM:CLOCK1 <parameter> (sets DS1 clock source)

:SOURce:DM:CLOCK2 <parameter> (sets STS1 clock source)

:SOURce:DM:CLOCK3 <parameter> (sets DS3 clock source)

New following SCPI commands have been added:

:SOURce:DM:CLOCK1? (queries DS1 clock source

:SOURce:DM:CLOCK2? (queries STS1 clock source)

The new SCPI commands have been added to query test set for its clock source settings. These commands have been added to this SCPI Manual.

The following SCPI commands has been added to query DS3 FFM status:

:FETCh:DMOD:DS3? FFM

The test set has been modified to detect DS1 yellow alarm when transmit data pattern is set to AIS.

Miscellaneous Fixes

The SCPI command used to inject errors has been improved. Previously, the software required the error inject command to be issued twice before the test set responded. This has been corrected. The error injection command format is as follows: **:SOURce:DM:EINJect:<payload> <error><rate>** where the parameter for rate is CONTINUOUS.

Product Numbers E4480A & E4487A

Host software version 6.60 is a software maintenance release for the HP 156MTS and 31XE test sets.

Agilent recommends you install this software in your test set.

Applicability

Host version 6.60 supports these test sets:

- E4480A 156MTS SONET/T-Carrier Test Set

- E4487A 31XE SONET Electrical Test Set

Host Software Version 6.60 Release Notes

Requirement

Before installing the new host version 6.60 software, you must upgrade the test set's software with boot-code version 6.2. In the past, you had to send your 156MTS or 31XE test set back to HP to have new boot-code software installed. Starting with boot-code version 6.2, you can install the software yourself. See the *Boot-code Software Version 6.2 Release Notes* for installation instructions.

Enhancements

Host version 6.60 includes the following enhancements:

- Path-trace string (J1 and J2 byte) and section-trace string (J0 byte) display and editing features
- Performance improvements

SCPI Commands

Host version 6.60 introduces several new SCPI commands. also look up command descriptions in this manual.

:SYSTem:OPTionS ?

:FETCh:DMON:OPTionS ?

:FETCh:DMOD:DS3:CBIT ? <*row,cbit*>

:FETCh:DMOD:STS3C ? RXFREQ

:FETCh:DMOD:STS12C ? RXFREQ

Use the following new SCPI command to retrieve DS1 ones density alarm violations:

:FETCh:DMOD:DS1 ? ONESDEN

Installation

You can install the new host software (see –10) after you install the boot-code software version 6.2 on your test set.

Install only software with a higher version number than the software already in your test set. The distribution diskette includes the appropriate software for your test set.

For more information,
call Telecomms Networks Test Division (TNTD) at 800-923-7522.

Path- and Section-Trace Strings

Host version 6.60 lets you display and edit the J1 and J2 path-trace strings, and the J0 section-trace string. (See Chapters 3 and 4 of this manual, for information about the screens.)

- New fields have been added to the J1 & J2 Path Trace Control and J0 Section Trace Control screens.
- The **Fill on ACTION** (or **Fill on INJECT**) and **Format** fields define the transmit trace string to transmit when you press the ACTION (INJECT) button (see the table below).
- **Results Display in** sets the display mode for the path-trace byte and section-trace byte values (below). This field has two options:
  - **ASCII** (the default)–displays values in ASCII.
  - **Hex & ASCII**–displays values in hex and ASCII.

Fill on Action	**Format**		
	1-Byte	**16-Byte**	**64-Byte**
NULL	*0x00 in all bytes.*	*0x00 in all bytes.*	*0x00 in all bytes.*
HEX	*0x01 in all bytes.*	*0x41 – 0x4F with CRC7, copied 4 times.*	*0x41 to 0x7F (ASCII A to DEL), followed by a carriage return (0x0D) and line feed (0x0A).*
ASCII	*0x41 in all bytes.*	*“nnnnnn” serial number with CRC7, copied 4 times.*	*“Agilent Technologies 156MTS Test Set Serial No. nnnnnn” followed by carriage return (0x0D) and line feed (0x0A).*

Fill on Action	Format		
	1-Byte	**16-Byte**	**64-Byte**
USER	*First byte copied to all 64 bytes.*	*Calculate and insert CRC7 of first 16 bytes; copy first 16 bytes 4 times.*	*No action.*

Product Numbers E4480A & E4487A

Host software version 6.50 is a software maintenance release for the 156MTS and 31XE test sets.

Agilent recommends you install this software in your test set.

Applicability

Host version 6.50 supports these test sets:

- E4480A 156MTS SONET/T-Carrier Test Set
- E4487A 31XE SONET Electrical test Set

New Features

Host version 6.50 includes the following new features:

156 MTS Host Software Version 6.50 Release

- [illegible] display and editing features
- several new SCPI commands for controlling DS3 FEAC bits
- performance improvements

This release also fixes problems with SCPI performance, and results store/recall functions.

Installation

If the software described in this release note is not already installed in your test set, follow the instructions on –26 to download the software.

Install only software with a higher version number than the software already in your test set. The distribution diskette includes the appropriate software for your test set.

For more information, call Telecomms Networks Test Division (TNTD) at 800-923-7522.

Path- and Section-Trace Strings

Host version 6.50 introduces enhancements for displaying and editing the J1 path-trace string and the J0 section-trace string. (See Chapters 3 and 4 of the user manual for the screens.)

J1 Path Trace Control and J0 Section Trace Control Screens

Two new fields have been added to these screens.

Fill on ACTION (or **Fill on INJECT** for older models) automatically fills the transmit trace string when you press the ACTION (INJECT) button. The field has three options:

- **NULL** fills the string with 00.
- **HEX** fills the string with incremental hex values from 42 to 7D (ASCII characters B to }), followed by a carriage return (0x0D) and line feed (0x0A). For example, byte 1 is set to 42, byte 2 to 43, and so on, up to byte 64 which is 7D.
- **ASCII** fills the string with the text 156MTS Test Set Serial No

nnnnnn' (where *nnnnnn* is the serial number of the test set). The string is followed by a carriage return (0x0D) and line feed (0x0A). The remaining bytes are set to 00.

Results Display in sets the display mode for the path-trace byte and section-trace byte values (below). This field has two options:

- **ASCII** (the default) displays values in ASCII.
- **Hex & ASCII** displays values in hex and ASCII.

STS-Path Trace Byte (J1) Values and OC-12 Section Trace Byte (J0) Values Screens

These screens display the 64-byte message carried in the path-trace (J1) or section-trace (J0) bytes. The message begins at the first non-zero byte following a **NULL** (0x00), carriage return (0x0D), or line feed (0x0A) byte.

The **Results Display in** field (above) controls how the message is displayed:

- When set to **ASCII,** only characters in the range 20 through 7E hex are shown.
- When set to **Hex & ASCII**, the message takes up two screens: the first screen shows bytes 1–32, and the second shows bytes 33–64.

DS3 FEAC Control

The DS3 FEAC Control screen provides access to DS3 FEAC bits. This section describes several screen features previously not documented. (See “DS3 FEAC Channel Parameters” in Chapter 7 of the user manual for information about this screen.)

When the **Continuous Alarm/Status** field is set to ON, the test set continuously transmits the FEAC code specified in the **Alarm/Status** field. When the field is set to off, the FEAC bit is set to the value (0 or 1) specified in the DS3 C-Bit Control screen: Row 1, C3 program bit.

The **FEAC Burst** and **Burst of** fields are used to transmit a

FEAC code several times in a row. The code burst is sent when the ACTION (INJECT) key is pressed. When the burst sequence is complete, the FEAC setting returns to its previous value.

- **FEAC Burst** specifies the code to send.
- **Burst of** specifies how many times to send it.

When **FEAC Burst** is set to Loopback Activate or Loopback Deactivate, the test set sends loopback codes (loop-up or loop-down) to the line specified in **Loopback Line**, as follows:

- The loopback code is sent the number of times specified by **Burst of**.
- The value of **Loopback Line** is sent the number of times specified by **Burst of**.

When **FEAC Burst** is set to Alarm/Status, the FEAC code specified by the **Alarm/Status** field is sent **Burst of** number of times.

See the following section for information about how to control DS3 FEAC codes through the SCPI interface.

New DS3 FEAC SCPI Commands

Host version 6.50 introduces several commands for controlling DS3 FEAC codes through the SCPI interface. Currently, the commands are available only for FEAC codes received on the DS3-A input.

For more information about FEAC codes and their functionality, ANSI T1.107, section 9.3.3 (1995).

`:FETCh:DMOD:DS3? <parameter>`
retrieves information about the FEAC codes received by the test set. The following table lists command parameters.

Parameter	Retrieved Value
FEAC_CODE	*Last received FEAC alarm/status code (text and six-bit sequence)*

Parameter	Retrieved Value
FEAC_CODE_SEC	*Number of seconds since last FEAC code received*
FEAC_SEQ	*Type of FEAC control-code sequence (activate or deactivate) last received*
FEAC_SEQ_SEC	*Number of seconds since last FEAC control-code sequence received*
FEAC_LINE	*Line affected by most recently received FEAC loopback control sequence (for example, DS1 #7)*

For example, **`:FETCh:DMOD:DS3? FEAC_CODE`** retrieves the last FEAC alarm/status code received.

`:SOURce:DM:DS3:FEAC:CBIT < row, cbit, 0|1 >` sets the specified C-bit to a 1 or 0 (where *row* is the M3 subframe 1–7, and *cbit* is 1, 2, or 3). For example, **`:SOURce:DM:DS3:FEAC:CBIT 1, 3, 0`** assigns the value 0 to the C3 bit in row 1 (the FEAC bit). See "DS3 C-Bit Configuration" in Chapter 7 of the user manual for more information about rows and C-bits.

`:SOURce:DM:DS3:FEAC:CONT_ALM_STAT <ON|OFF>`

controls whether FEAC codes are sent continuously (ON), or not (off). When off, the FEAC bit is set by the DS3 C-Bit Control screen: Row 1, C3 program bit.

:SOURce:DM:DS3:FEAC:BURST_SETTING <parameter> selects the type of FEAC code to transmit several times in a row (**BURST_SIZE** specifies how often). The codes are transmitted when the ACTION (INJECT) key is pressed.

Parameter	Code Transmitted
NONE	*No codes transmitted*
ACTIVATE	*Loop-up code for LOOPBACK_LINE*
DEACTIVATE	*Loop-down code for LOOPBACK_LINE*

Parameter	Code Transmitted
ALM_STAT	*FEAC code for ALM_STAT (below)*

`:SOURce:DM:DS3:FEAC:LOOPBACK_LINE < 0...63 >`
selects the DS3 line to send the loop code to. Specify the decimal equivalent (0 – 63) of the line identifier code listed in the user manual (the left-most bit is the MSB). For example, **`:SOURce:DM:DS3:FEAC:LOOPBACK_LINE 47`** sends the loop code to DS1 line #15.

`:SOURce:DM:DS3:FEAC:BURST_SIZE < 3...15 >`
specifies the number of times (3–15) to transmit the selected FEAC code (**BURST_SETTING**) when the ACTION (INJECT) key is pressed.

`:SOURce:DM:DS3:FEAC:ALM_STAT < 0...63 >`
selects the FEAC code to transmit when **CONT_ALM_STAT** is set to ON, or **BURST_SETTING** is set to ALM_STAT. Specify the decimal equivalent (0 – 63) of the FEAC code bit sequence listed in the user manual (the left-most bit is the MSB). For example, **`:SOURce:DM:DS3:FEAC:ALM_STAT 25`** sends a DS3 Equipment Failure (SA) alarm.

`:SOURce:DM:DS3:FEAC:ACTION`
is equivalent to pressing the ACTION (INJECT) key in the DS3 FEAC Control screen.

Product Numbers E4480A & E4487A

Host software version 6.41 is a minor maintenance upgrade to version 6.40.

Agilent recommends you install this software in your test set.

Installation

If release 6.41/3.40 is not already installed in your test set, please follow the attached instructions to download the software.

Applicability

Host version 6.41 and ATM software version 3.40 supports these

Host Software Version 6.41 and ATM Software Version 3.40 Release Notes

- E4480A 156Mb/s SONET/T-Carrier Test Set
- E4487A 31XX SONET Electrical Test Set

Older test sets may require a hardware upgrade to support some features. Please contact the factory.

New ATM Features

Host software version 6.41 combined with ATM version 3.40 adds new ATM test capabilities.

Option URZ is required for ATM.

ATM Data Selection

“**ATM**” is a payload selection for the **Data**> field in OC-3c, DS3, DS1, and E1 signals. This replaces the ATM selections for the **Payload** field on the test setup screen. The DS3, DS1, and E1 signals can be dropped from higher-rate signals.

When **Data**> is set to ATM, the **ATM Setup** selection is enabled on the Control Screens menu.

The **Data**> ATM selection does not apply for STS-12c and STS-1 ATM testing. These modes are selected by setting **Payload** to ATM on the Test Setup screen.

ATM Setup Menu

An ATM Setup menu has been added to select the ATM test mode. Selecting **ATM Setup** from the Control Screens menu calls this menu, which allows the selection of standard, VPI/VCI scan, transfer delay, inter-arrival time, or cell capture modes.

Selecting an ATM mode automatically activates the appropriate ATM setup screens and corresponding ATM results screens.

Use with older ATM hardware

To take full advantage of the new ATM features in the 6.41/3.40 release, older test sets may require a hardware upgrade performed at the factory. Please consult the factory for more information.

Other New Features

Other features in the 6.41/3.40 release include:

Auto Setup Enhancements

- Enhanced E1 detection
- Improved “demo” when test set is looped back on itself
- Animated activity indicator during Auto Setup test

SCPI Commands

The following SCPI commands have been added to support the ATM **Data**> selections:

- **`:SOUR:DM:DATA1 <ATM>`**
- **`:SOUR:DM:DATA3 <ATM_HEC | ATM_PLCP>`**
- **`:SOUR:DM:DATA4 <ATM>`**

Index

T

U

V

X

Manual Part Number
E4480-9001

Version 4-2
Printed in USA
May 2000

Agilent Technologies