Remote Operation Getting Started Guide

This guide describes remote operation and control of a T-BERD/MTS 5800 Family, MSAM, CSAM, DMC, Transport Module, MAP-2100, SC 4800/4800P, ONA-800, and ONA-1000, using SCPI commands.

SCPI command syntax and format requirements are presented along with instructions on how to establish a connection to a module and initiate example applications. A list of commands and queries that apply to all applications concludes this section as a reference for basic remote control applications.

This guide addresses several test platforms. As these platforms are physically different, only those physical features that apply directly to remote operations will be discussed when necessary.

The sections contained in this document are as follows:

- "About the T-BERD/MTS 5800 Family, MSAM, CSAM, DMC, Transport Module, MAP-2100, SC 4800/4800P, ONA-800, and ONA-1000 " on page 2
- "Remote control flow" on page 4
- "Command syntax and format requirements" on page 5
- "Getting the module port number" on page 6
- "Getting the remote control port number" on page 8
- "Working with applications" on page 13
- "Using setups, actions, and results" on page 17
- "Rebooting an instrument" on page 22
- "Example remote control procedures" on page 23
- "Demo Remote Control Scripts" on page 30
- "Automation using SCPI Commands - Steps to Connect" on page 30
- "MAP-2100 Optical Switch Automation" on page 37
- "SCPI command reference" on page 38
- "Automation with ONA-800 and ONA-1000" on page 38
- "Appendix: Special Setup Types" on page 44

About the T-BERD/MTS 5800 Family, MSAM, CSAM, DMC, Transport Module, MAP-2100, SC 4800/4800P, ONA-800, and ONA-1000

Before beginning a remote session, you should be familiar with the products and modules, found in the *T-BERD/MTS/SC/MAP-2100 Getting Started Guide, ONA-1000 Getting Started Guide,* and *ONA-800 Getting Started Guide*.

The following terms have a specific meaning when referring to the T-BERD/MTS 5800 Family, MSAM, CSAM, DMC, Transport Module, MAP-2100, SC 4800/4800P, ONA-800, and ONA-1000:

- **10/100/1000 Ethernet** — Used to represent 10/100/1000 Mbps Ethernet. The connector panel also uses 10/100/1000M to identify the connector used for 10/100/1000 Mbps Ethernet testing.
- **1GigE** — Used to represent 1 Gigabit Ethernet.
- **10GigE** — Used to represent 10 Gigabit Ethernet.
- **40GigE** — Used to represent 40 Gigabit Ethernet.
- **100GigE** — Used to represent 100 Gigabit Ethernet.
- **Action** — Refers to an item that triggers some change to the test interface, such as inserting an alarm or enabling the laser. In the user interface, these are located in the Actions toolbar at the bottom of the main results screen.
- **Application module** — Used to refer to the component that provides test functionality to the assembled instrument. This manual supports application modules: the ***MSAM*** and ***CSAM***.
- **Assembly** — Used to refer to a complete *set of components* assembled as an instrument and used for testing. This manual supports several assemblies: the ***Transport Module assembly***, consisting of an T-BERD ⁄ MTS 8000 base unit and Transport Module, the ***MSAM assembly***, consisting of a MSAM, Physical Interface Modules (PIMs), and a T-BERD ⁄ MTS 6000A base unit, the CSAM, and a ***DMC assembly***, consisting of up to two MSAMs, up to four PIMs, and/or up to two CSAMs, a Dual Module Carrier (DMC), a T-BERD ⁄ MTS 8000 base unit, and an ONA-1000/ONA-800 base unit.
- **Base unit** — The unit which connects to the application module and power adapter, providing the user interface and a variety of connectivity and work flow tools. If optioned to do so, the base unit also allows you to measure emitted power, received power, and optical link loss on fiber optic networks.
- **Battery Module —** The module connected to the back of the base unit, which supplies power whenever it is not provided using the power adapter.
- **Component**— Used to refer to an individual hardware *component* which is connected to the other components to build a test instrument (assembly). This manual supports the following components: the Transport Module, the MSAM, the CSAM, and the DMC. The base units are documented in separate manuals.
- **DMC** — Dual Module Carrier. The DMC is a two slot chassis which you can connect to the T-BERD / MTS 8000 base unit to test using up to two MSAM and/or up to two CSAM application modules and four Physical Interface Modules (PIMs).

- **FC** — Used to represent Fibre Channel on the Transport Module connector panel and the Transport Module and the MSAM user interface to identify the optical connectors used for Fibre Channel testing.
- **Viavi Ethernet test set** — A test set marketed by Viavi and designed to transmit an Acterna Test Packet (ATP) payload. These packets carry a time stamp used to calculate a variety of test results. The FST-2802 TestPad, the SmartClass Ethernet tester, the HST with an Ethernet SIM, the T-BERD/MTS 8000 Transport Module, and the T-BERD/MTS 5800 Family, MSAM, CSAM, DMC, Transport Module, MAP-2100, SC 4800/4800P, ONA-800, and ONA-1000 can all be configured to transmit and analyze ATP payloads, and can be used in end-to-end and loopback configurations during testing.
- **MSAM (Multiple Services Application Module)** — Referred to generically as "the instrument" when inserted in the T-BERD / MTS 6000A base unit or the DMC with a PIM. The MSAM provides testing functionality for the base unit.
- **CSAM (100G Services Application Module)** — Provides testing functionality up to 100G for the base unit.
- **MTS 8000 and MTS 6000A** — *See* T-BERD/MTS 8000 and T-BERT 6000A.
- **OC-n** — Used to refer to each of the optical SONET rates supported by the Transport Module or MSAM (OC-3, OC-12, OC-48, and OC-192), where *"n"* represents the user-selected line rate.
- **OTN** — Optical Transport Network.
- **OTU1** — Optical Transport Unit 1. A 2.7G OTN signal designed to carry a SONET OC-48 or SDH STM-16 client signal.
- **OTU2** — Optical Transport Unit 2. A 10.7G, 11.05G (OTU1e), or 11.1G (OTU2e) OTN signal designed to carry SONET OC-192, SDH STM-64, or 10GigE Ethernet WAN and LAN client signals.
- **OTU3** — Optical Transport Unit 3. A 43G OTN signal designed to carry a SONET OC-768 or SDH STM-256 client signal or 43GE signal using transcoding.
- **OTU4** — Optical Transport Unit 4. A 111.8G signal designed to carry 100GE client signals.
- **PIM —** The physical interface module (PIM) inserted into one of up to two ports provided on the MSAM chassis. PIMs supply the physical connectors (interfaces) required to connect the MSAM to the circuit under test. A variety of cables, SFPs, and XFPs are offered as options, and can used to connect the PIMs to the circuit.
- **Result** — Refers to a measurement made by the test application. In the user interface, these are displayed in the result windows and in the software LED panel on the main Results screen.
- **Setup** — Refers to a setting of a test application that is usually changeable by the user. In the user interface, they are located on the Setup screen and on the main Result screen in the horizontal toolbars.
- **SFP** — Small form-factor pluggable module. Used on the connector panel and throughout this manual to represent pluggable optical modules.
- **Slice** — Used as a generic term for an application module or battery module. Typical use is referring to the Transport Module, as the "BERT slice". After assembling a T-BERD/MTS 8000 base unit to a battery module and then to the Transport Module, the instrument resembles a sandwich; thus, the individual components are slices.

- **STM-n** — Used to refer to each of the optical SDH rates supported by the Transport Module or MSAM (STM-1, STM-4, STM-16, and STM-64), where *"n"* represents the user-selected line rate.
- **STS-1** — Used to refer to the electrical equivalent of OC-1 (51.84 Mbps) supported by the Transport Module or MSAM.
- **STM-1e** — Used to refer to the electrical equivalent of STM-1 (155.52 Mbps) supported by the Transport Module or MSAM.
- **T-BERD/MTS 5800 Family, MSAM, CSAM, DMC, Transport Module, MAP-2100, SC 4800/ 4800P, ONA-800, and ONA-1000** — The family of products, typically a combination of a base unit, a battery module, and one or more application modules. The Dual Module Carrier (DMC) can be used on the T-BERD / MTS 8000 platform to test using two MSAMs.

  The products are branded T-BERD in North America and branded MTS (Media Test Set) in Europe, and are interchangeably used throughout supporting documentation. (T-BERD 8000, MTS 8000, MTS-6000A, T-BERD 6000A, MTS5800, T-BERD 5800). Smart Class 4800 and 4800P are also referred to as SC-4800 in supporting documentation.
- **XFP** — 10 Gigabit Small Form Factor Pluggable optical transceiver. A variety of optional XFPs are available for testing 10 Gigabit fiber circuits.
- **SFP** — Small Form Factor Pluggable optical transceiver.
- **SFP+** — 10 Gigabit Small Form Factor Pluggable optical transceiver.
- **QSFP28** — Quad Small Form Factor Pluggable optical transceiver with up to 28 Gbps per lane for 100GE/OTU4.

Remote control flow

When running a remote control session, the basic flow is as follows:

1 Obtain the remote control port number

2 Start remote control

3 Interacting with the applications

- **a** query for running applications
- **b** launching an application
- **c** using setup, action, and results
- **d** restarting a test
- **e** configuring a timed test

4 Exiting the application(s)

5 Leaving remote control

Command syntax and format requirements

Before starting the remote session you should become familiar with the syntax and format requirements.

Command case-sensitivity

SCPI command headers are not case sensitive, so both commands below are correct and return the Signal Present test result.

```
:SENSE:DATA? CSTatus:PHYSical:SIGNal
:sense:data? CSTatus:PHYSical:SIGNal
```

Result names are not case sensitive, so both commands below return the Signal Present test result.

```
:SENSE:DATA? CSTatus:PHYSical:SIGNal
:SENSE:DATA? cstatus:physical:signal
```

Setup and action values *are* case-sensitive.

`:SENSe:PAYLoad:BERT:PATTern ALL_ONES`	Correct
`:SENSe:PAYLoad:BERT:PATTern all_ones`	Incorrect

Sockets and messages

All remote control communications are socket-based TCP/IP. Any message passed via remote control must be terminated with a newline character ("\n") as shown (or ENTER in direct entry mode).

```
:INITiate\n
```

Without the following \n, a message may be interpreted incorrectly or cause an error condition.

Issuing multiple commands

You can issue multiple commands on a single line by separating each of the commands using a semicolon. The following examples show correct and incorrect presentation of the multiple commands.

`:INITiate; :SYSTem:ERRor:NEXT?\n`	Correct
`:INITiate :SYSTem:ERRor:NEXT?\n`	Incorrect

Serial command processing

The remote control system processes commands sequentially in the order they are received. It does not process multiple commands simultaneously. If a command is sent while another

command is still being processed, the second command will not be processed until the first command has been fully executed.

Short-form v. Long-form commands

Commands may be abbreviated using the short form (the capital letters in the command). Each word in the command may be either the long or short-form.

If long-form is used, the entire word must be specified; “partial” long-form is not recognized.

The following examples show the complete long-form version followed by various, short-form versions of the same command. The final example shows an example of “partial” long-form that is not acceptable.

:SENSe:DATA? CSTatus:PHYSical:SIGNal	Correct
:SENSe:DATA? CST:PHYS:SIGN	Correct
:SENS:DATA? CSTatus:PHYS:SIGNal	Correct
:SENS:DATA? CST:PHYS:SIGN	Correct
:SENS:DATA? CST:PHYS:SIGNa	Incorrect

Getting the module port number

NOTE:

If you are using the firewall feature on 5800 models or MAP-2100, you will first need to establish a tunnel over SSH (port 22) with the unit. All commands can be executed on the provided port numbers local to the unit or over an SSH tunnel.

Starting remote operation

The first step is to start remote operation

1. Open a TCP/IP socket connection to the unit on port 8000.
2. Start remote operation.

   ***REM**

IMPORTANT:

The ***REM** command must always be the first command sent after opening a socket (to port 8000, the module port, or the RC port). On the ANT platforms this was not necessary - they allowed you to connect, access applications, disconnect, then reconnect and continue without having to send *REM again; on the MTS platforms, it is required.

Obtaining the module port number

Before issuing any SCPI commands, a socket connection to the module must be established.

Module identification and location

In order to establish a connection, the module must first be identified using the <**side**> and <**slice**> parameters. The MTS-5800 and MTS-6000 platforms can only hold a single module so these always use the same side/slice parameters. The MTS-8000 can potentially hold multiple modules, and these parameters specify which one to establish the remote control connection with. Refer to the table below to determine which parameter values to use.

Platform	<side>	<slice>
MTS-5800v1, MTS-5800v2 and MTS-5800-100G. MAP-2100	**BOTH**	**BASE** Note: This was formerly “SLIC1” but was updated with the release of the MTS-5800v2 (V6.0 for the MTS-5800v1).
MTS-6000	**PWRS**	**SLIC1**
MTS-8000 with Transport Module	**BOTH**	Specify the “slice” number of the desired module, starting in the front and counting back. – SLIC1 – SLIC2 – SLIC3 If only one module is present, this parameter should be SLIC1
MTS-8000 with Dual Module Carrier	This parameter specifies which MSAM and/or CSAM (left or right) to connect to. – **OPPS** – Module on the left side (opposite side from the power connector) – **PWRS**–– Module on the right side (same side as the power connector)	Same as above: – SLIC1 – SLIC2 – SLIC3

Verify the module location:

`MOD:FUNC:LIST?` `<side>,<slice>`

replacing <**side**> and <**slice**> as detailed in “Module identification and location”. If the module is located where queried, the system will respond with “BERT”.

Example procedure

1 Verify whether the module is powered.

MOD:FUNC:SEL? <side>,<slice>,"BERT"

example (MTS-5800v1 and MTS-5800v2):

MOD:FUNC:SEL? BOTH,BASE,"BERT"

The module will return “OFF” or “ON”.

2 If the module state response was OFF, turn ON the module:

MOD:FUNC:SEL <side>,<slice>,"BERT",ON

3 Query the module’s port number.

MOD:FUNC:PORT? <side>,<slice>,"BERT"

The module will return its port number.If the module is in a ready state, the response to the port query will be the port number, for example “8002”.

If the port is not in a ready state, the response will be **-1** .

This port number will be needed in the next step.

Rebooting the Instrument

To reboot the unit, establish a socket connection to port 8000. Follow the procedure in “Rebooting an instrument”.

Getting the remote control port number

1 Open a TCP/IP socket to the module port obtained in “Module identification and location” on page 7.

2 Enable remote operations on the module:

***REM**

3 Verify the BERT module is ready

:SYST:FUNC:READY? <side>,<slice>,"BERT"

Returns 1 if the BERT module is in a ready state (i.e. powered on and finished booting up), 0 otherwise.

4 Enable the remote control socket and return its port number

:SYST:FUNC:PORT? <side>,<slice>,"BERT"

Returns this module's remote control port number. If the module is not in a ready state, the response will be -1.

Remote modes

Remote operations are often conducted when there is no physical access to the base unit where applications can be directly implemented using the Graphical User Interface (GUI). Alternate situ-

ations can exist where remote operations need to be monitored by a technician at the base unit or even supplemented with input from the base unit.

Default Mode

The default remote operational mode disables the Graphical User Interface (GUI) on the base unit. This mode is activated by the implementation of the following command with no optional values:

`*REM`

In this mode, the GUI is disabled and a message is displayed indicating the unit is under remote control. At any time the GUI is enabled or disabled, all currently running applications will be shut down.

GUI Mode

Alternatively, remote control may be used without deactivating the GUI. This mode can be used to debug remote control tests and to verify that commands have the expected behavior; the default mode is recommended for routine automated tasks. To activate GUI mode, two optional parameters may be added to the *REM command on the remote control port: <GUI-mode> and <access-mode>.

<GUI-mode> parameter

The **`<GUI-mode>`** parameter has three variables that can be defined:

- *HIDDEN* - when specified, the GUI is disabled (default if not specified).
- *VISIBLE* - when specified, the GUI is enabled.
- *CURRENT* - when specified, whatever is currently set will continue to be applied to the new application or command.

<access-mode> parameter

The **`<access-mode>`** parameter has three variables that can be defined.

- *READ_ONLY* - when specified, a GUI user may navigate the UI but not change any settings (default, if not specified).
- *FULL* - when specified, a GUI user may change settings.
- *CURRENT* - when specified, whatever is currently set will continue to be applied to the new application or command.

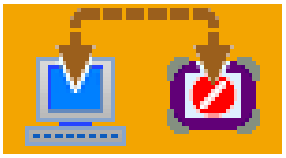
These modes may be activated by the implementation of the REM command and specifying the alternate parameters as follows:

`*REM <GUI-mode> <access-mode>`

Remote Mode Indicators

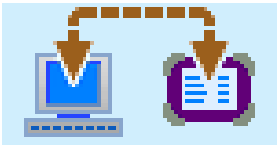
To inform the local user of the base unit of the GUI mode specified by the remote operations, one of two icons will display on the GUI.

Read-Only



The Read-Only icon will display when the *REM <access-mode> has been set to READ_ONLY.

Full-Access



The Full-Access icon will display when the *REM <access-mode> has been set to FULL.

Once remote control has been started with the GUI active, an "Access Mode" entry will become available in the Tools menu, which allows the GUI user to switch between read-only or full-access mode.

Global Commands

The following commands are recognized by remote control. The *RST, *IDN? and *OPC? commands are loosely based on the SCPI standard, but may not have the same behavior. Other common commands from the SCPI standard starting with “*” are not supported.

*REM

The *REM command is usually the first command issued once connected to the remote control port. This puts the unit into remote control mode. By default, this will shut down any running applications, disable the graphical user interface (GUI), and display a message indicating the unit is under remote control.

***REM** may be given additional parameters to keep the GUI active, and possibly prevent another user from modifying any settings while the GUI-enabled remote control mode is active.

*RST

The *RST command shuts down all running applications and clears any saved application settings. This is useful for closing multiple applications at one time, or to ensure that a newly launched application isn't affected by any settings made in a previous test.

*IDN?

The *IDN? command returns identifying information about the test unit in the form of a comma separated list. The manufacturer, software name, module id, and software version number are provided.

```
> *IDN?
JDSU,BERT,A1-4690290,V11.1
```

*OPC?

The operation complete query from the SCPI standard is recognized, but does not have the same behavior the standard describes. This due to the fact that in remote control, commands are processed serially; a command will not be processed until all previous commands are completed. This means that *OPC? will not be processed until all previous operations are complete, at which point it will always return 1.

It is recommended that the :SYSTem:ERRor? query be used instead of *OPC?, as it also provides error checking on the previous command.

*GUI

This command terminates remote control mode and restores the GUI interface. All applications that were running under remote control will be shut down, and first application in the test menu will be launched. Alternatively, the application to launch may be specified as an optional parameter.

The example below shows this command with a parameter to launch 10/100/1000M Ethernet Layer 2 Traffic Terminate. Keep in mind the *GUI command may take some time to process. The 0, “No error” message may not show up for a minute or two.

```
> *GUI TermEth10ML2Traffic
> :SYSTem:ERRor?
0, "No error"
```

Switching to Remote Control mode

Prepare BERT Application

Using the information discovered in "Getting the remote control port number" on page 8, establish a socket connection to the BERT application port so the BERT application can be set for operation.

To prepare the BERT application port for use, it must be reset to its default configuration, shutting down all running applications.

Start Remote Operation

Set the BERT application for remote operations:

***REM**

Exits all running applications and shows a message on the screen indicating the unit is under remote control.

Reset BERT

Optional. Exit all currently running applications and restore the default configurations:

***RST**

Query for Error Status

To verify that the system has been reset, the system must be checked for errors. There will be no return from this query until the reset sequence has been completed. This may take a little while but a 90 second time-out on this command should be sufficient.

Check the system for errors:

:SYSTem:ERRor?

The return from this command will be `0, "No error"`. There will be no return until the ***RST** command is finished.

When the `0, "No error"` response is received, the desired module connection has been completed. The system is now ready to have testing applications started on that module.

Working with applications

System Commands

Check for Errors

:SYSTem:ERRor?

If a remote control command has an error, no message will be returned unless requested with "`:SYSTem:ERRor?`". The response to this command is an error code number and a brief description in quotes. The description will be one of the messages in the table below, possibly followed by more information.

Error code	Error message	Error code	Error message
0	No error	-141	Invalid character data
-100	Command Error	-144	Character data too long
-102	Syntax Error	-150	String data error
-103	Invalid separator	-151	Invalid string data
-108	Parameter not allowed	-200	Execution error
-109	Missing parameter	-220	Parameter error
-113	Undefined header	-221	Settings conflict
-120	Numeric data error	-222	Data out of range
-121	Invalid character in number	-224	Illegal parameter value
-138	Suffix not allowed	-231	Data questionable

It is recommended that this command is sent automatically after any query command (ending with a '?'). If a command is errored, checking "`:SYSTem:ERRor?`" will immediately report the mistake, while not checking it will result in the mistake being silently ignored.

Examples:

```
> *REM
> :INITiate
> :SYSTem:ERRor?
-221, "Settings conflict; no application has been selected"

> :SENSe:MISSpelled:COMMand?
> :SYSTem:ERRor?
```

```
-113, "Undefined header; could not find
':sense:misspelled:command'"
```

List Available Applications

:SYSTem:APPLication:APPLications?

This command returns a comma-separated list of all applications that can be launched in addition to any applications currently running.

If an existing application is using a particular interface, then all other applications that use that interface type will be unavailable unless another suitable interface is available. For example, if a 10/100/1000M electrical Ethernet application is using the only Ethernet port, all other 10/100/1000M Ethernet applications will be unavailable and therefore will not show up in this query. To determine the full list of applications a unit supports, query this command when no other applications are running.

Example:

```
> :SYSTem:APPLication:APPLications?
TermDs1Bert,ThruDs1Bert,SingleMonDs1Bert,...
```

For units that have multiple ports, this query can also be given additional parameters to request the applications that are available to a specific port. The first parameter is reserved for internal use; use the value “none”. The second parameter specifies the port to query: 1 or 2.

Example:

```
> :SYSTem:APPLication:APPLications? none 2
MonEth10ML2Traffic,TermEth10ML2Loopback,...
```

List Currently Running Applications

:SYSTem:APPLication:CAPPlications?

This command returns a comma-separated list of the unique identifiers for all currently running applications. The return values returned from this query may be used to select an application (using the `:SYSTem:APPLication:SELect` command).

Example:

```
> :SYSTem:APPLication:CAPPlications?
TermStm16Au4Vc4E4Bert_121,TermStm64Au4Vc4E4Bert_121
```

Launch an Application

:SYSTem:APPLication:LAUNch <application name> [port]

This command will launch the specified application. This operation is different from normal GUI interaction in that the application will be launched in addition to any existing applications; it will not replace the currently running application. The port number on which to launch the application (1 or 2) may be specified as an optional parameter if the unit has a second port. If not specified, the default is the first available port. If no open ports are available, an error message will be stored (accessible with the `:SYSTem:ERROR?` command) and no action will be taken.

Examples:

Launch 2G Fibre Channel on port 1 (or port 2 if port 1 is unavailable)

```
> :SYSTem:APPLication:LAUNch TermFc1GL2Traffic
```

Launch 2G Fibre Channel on port 2

```
> :SYSTem:APPLication:LAUNch TermFc1GL2Traffic 2
```

Another key difference from launching the test on the GUI is that in remote control, the test will come up in the "stopped" state, leaving most test results unreadable. The remote control user should usually send the `:INITiate` command after initializing the remote session.

:SYSTem:APPLication:LAUNch?

Adding a question mark (?) to the end of the launch command will query for the application identifier of the most recently launched application. This identifier may be used to select the application using the `:SYSTem:APPLication:SELect` command.

Example:

```
> :SYSTem:APPLication:LAUNch?
TermFc1GL2Traffic_121
```

Select an Application

:SYSTem:APPLication:SELect <application id>

This command selects a specific running instance of an application, which directs all subsequent commands to that application. It must be sent after the application is launched and the unique id for that application instance has been obtained.

The application id will be the application name followed by a 3-digit number. The first two digits represent the "slice" number and the "side" number where the module is located (see the documentation on Connecting to Remote Control for more explanation of these), and will always be the same within a particular remote control connection. The third digit represents the port number: 1

or 2. The example below shows two instances of the same Ethernet application, one running on port 1 and one running on port 2, and the command to select the one running on port 1.

Example:

```
> :SYSTem:APPLication:CAPPlications?
TermEth10ML2Traffic_121,TermEth10ML2Traffic_122
> :SYSTem:APPLication:SELect TermEth10ML2Traffic_121
> :SYSTem:ERRor?
0, "No error"
```

Remote Control Session Commands

A remote control session maintains information about a specific connection to an application. Once an application has been selected, a remote control session must be created and started before the user can interact with the application.

:SESSion:CREate

This command creates a remote control session with the selected application.

:SESSion:STARt

This command starts a previously created remote control session.

:SESSion:END

This command terminates the current remote control session. The session will automatically be ended when an application is shut down or the remote connection is terminated.

Common Application Commands

The following commands apply to all test applications.

:INITiate

The `:INITiate` command starts the test if it is in the “stopped” state.

:ABORt

The `:ABORt` command stops the test if it is in the “running” state.

To perform a test restart and restart all test measurement results, perform a test stop, followed by a test start.

Example:

```
> :ABORt
> :INITiate
> :SYSTem:ERRor?
0, "No error"
```

:EXIT

Shuts down the currently running application.

Using setups, actions, and results

Setup commands

A unique SCPI command exists for every setup.

The setup value should be space-separated from the command, but there should be no space before the port number and/or question mark.

NOTE

Commands do not return any values unless they are a query. Remember to check each command with the **:SYSTem:ERRor?** query. If you do not check each command, and one is incorrect, the error message does not indicate the incorrect command.

Setup values specification

To change a setup, issue the SCPI command for that setup followed by the new value. The command in the following example sets (or changes) the receive pattern on port 2 to all ones:

```
:SENSe:PAYLoad:BERT:PATTern1 ALL_ONES
```

For setups that are arrays, you can specify an index or list indices of the array element(s) that you want. A colon (:) can be used to denote an index range (within the @ specification of the channel) followed by the status value.

The following example disables (0) time slot 4 (first line) and enables (1) time slots 1, 2, 3 and 7 (second line) in fractional E1:

```
:SOURce:PDH:E1:CHANnel (@4) 0
:SOURce:PDH:E1:CHANnel (@1:3,7) 1
```

Setup value format

Setup values do not follow standard SCPI naming conventions and must be stated using the string values and enumerated value formats illustrated in the following examples. As with all values, setup values need to be space-separated from the command.

String values

String values are used for setups where the exact string is not a value defined by the SCPI implementation. String values must be enclosed in double quotes.

Correct format

```
:SOURce:DATA:MAC:IPV6:TX:DESTination:ADDRess "ff00::1234:2345"
```

Incorrect format

```
:SOURce:DATA:MAC:IPV6:TX:DESTination:ADDRess ff00::1234:2345
```

Port Definition

Port definitions are generally not required in setups unless a dual port application is being tested. If no number is specified, 0 (port 1) is assumed.

To specify port numbers in dual port applications, add a zero-based port number (0 for port 1, 1 for port 2) directly after the command (or before the `?` if you are issuing a query).

The setup value should be space-separated from the command, but there should be no space before the port number and/or question mark.

NOTE

Remember to check each command with the `SYST:ERR?` query. If you do not check each command, and one is incorrect, the error message does not indicate the incorrect command.

Enumerated Values

Enumerated values are used for setups that have specified settings defined by the SCPI implementation. Enumerated setup values are case sensitive and should not be enclosed in quotes.

Correct format

```
:SENSe:PAYLoad:BERT:PATTern ALL_ONES
```

Incorrect format

```
:SENSe:PAYLoad:BERT:PATTern "ALL_ONES"
:SENSe:PAYLoad:BERT:PATTern all_ones
```

Investigate the command line specifications, found elsewhere in this document, for a more complete list of string values and enumerated values that apply to various commands.

Query values

To query the current value of the setup, issue the SCPI command for that setup followed by a question mark.

The commands in the following examples both query for the transmit DS1 framing on line 0 (as described in "Port Definition" on page 18):

```
:SOURce:PDH:DS1:FRAMING?
:SOURce:PDH:DS1:FRAMING0?
```

Initiating actions

A unique SCPI command exists for every action. The examples in this section will show how actions can be implemented, and status can be queried and toggled using the same root command with minor variations on its initiation.

NOTE

Remember to check each command with the SYST:ERR? query. If you do not check each command, and one is incorrect, the error message does not indicate the incorrect command.

Initiate command

The command in the following example inserts a single bit error on line 0 (line assumed to be 0 when not indicated):

```
:SOURce:PAYLoad:BERT:INSert:TSE
```

Query state

To discover what is the state of error rate insertion action (on line 0) the **:RATE?** query is added:

```
:SOURce:PAYLoad:BERT:INSert:TSE:RATE?
:SOURce:PAYLoad:BERT:INSert:TSE:RATE0?
```

Initiate command

To turn error rate insertion ON (on line 0):

```
:SOURce:PAYLoad:BERT:INSert:TSE:RATE ON
:SOURce:PAYLoad:BERT:INSert:TSE:RATE0 ON
```

Toggling actions

Many action commands simply toggle the action ON and OFF.

The following command toggles error rate insertion (reverses its state) for line 0.

```
:SOURce:PAYLoad:BERT:INSert:TSE:RATE
```

Executing the same command a second time without a parameter will turn error rate insertion back to its original state (OFF).

Port Definition

As with setups, the zero-based port index (0 for port 1 and 1 for port 2) may optionally be specified at the end of the command (or before the "?" if a query is issued).

If no number is specified, 0 (port 1) is assumed.

Further resources

Investigate the action initiation specifications, found elsewhere in this document, for a more complete list of actions commands.

Querying results

Action commands do not return any value unless they are queries. To obtain the results of the tests initiated, a query must be issued.

All test results are queried using the following command:

```
[:SENSe]:DATA <result name>
```

where

- **`[:SENSe]`** is an [optional] entry.

  `[line]` represents the port number. It is not necessary to enter the port number unless a dual port application is running. For dual port applications, "0" corresponds to port 1; "1" corresponds to port 2.
  Usage: **`[:SENSe]:DATA[line]? <result name>`**

- **`[array indices]`** specifies indices for results that are arrays. For example, `(@1,3:5)` indicates that the query results are for indices 1, 3, 4 and 5. Note that index numbering is zero-based.
  Usage: **`[:SENSe]:DATA[line]? [array indices] <result name>`**
- **`result name`** is the name of the result to be queried.

These examples query for the value of the signal present result on line 1 (the second port), illustrating that [:SENSe] is optional:

```
:SENSe:DATA1? CSTATUS:PHYSICAL:SIGNAL
:DATA1? CSTATUS:PHYSICAL:SIGNAL
```

The following examples query for the value of pattern sync present result on line 0 (the first port), illustrating the optional nature of defining port 0:

```
:SENSE:DATA? CSTatus:PAYLoad:BERT:PATTern:SYNC
:SENS:DATA0? CST:PAYL:BERT:PATT:SYNC
:DATA? CST:PAYL:BERT:PATT:SYNC
:DATA0? CST:PAYL:BERT:PATT:SYNC
```

NOTE

Not-a-Number formats from the SCPI standard are supported. If a result is not applicable or is out of range, a result query will return an appropriate value as shown below.

`9.91e+37`	Not Applicable/Not Ready (cannot be calculated from current information)
`9.90e+37`	Over range
`-9.90e+37`	Under range

Result groups

Result commands may be added to user-defined groups, which allow the simultaneous query and reporting of multiple parameter result values.

Group definition

The format of a command to add a result parameter to a group is:

```
:SENSe:DATA:GROUP [group name] [result parameter]
```

where

- **`[group name]`** specifies the name of the group.
- **`[result parameter]`** is the data parameter being tested that is to be reported when the group results report is queried.

  A result can only be in a group once; no change will be made to the group if it is added again.

Group result report

Once a group is defined, it can be queried with the following command:

:SENSe:DATA:GROUP? [group name]

This will return the list of result values separated by semicolons (;) in the order in which they were added to the group.

Result group examples

The following example creates and queries a small group of Ethernet results:

:SENSe:DATA:GROUP summary CSTatus:PHYSical:SIGNal

:SENSe:DATA:GROUP summary CSTatus:PCS:PHY:SYNC:ACTive

:SENSe:DATA:GROUP summary CSTatus:MAC:ETH:FRAMe:DETect

:SENSe:DATA:GROUP? summary

The last command will return something like "`1; 0; 9.91e+37`", indicating signal is on, sync is off, and frame detect is not available.

Reset group deletion

Result groups are a report of multiple parameter results from a currently running application. Result groups are not retained beyond the current session.

All result groups will be deleted when the remote control session is closed with "**:SESSion:END**".

Rebooting an instrument

To reboot the unit, establish a socket connection to port 8000. The `*REM` command must be sent first. Then send the following command:

:SYSTem:REBoot

The time to reboot will depend on the platform type.

Query the `:SYST:FUNC:READY? <side>,<slice>,"BERT"` command on the module port to determine when the reboot is complete and BERT software is running again (see "Getting the remote control port number" for information on this command). This will bring the unit back into GUI mode. Any remote connections will need to be reestablished.

Example remote control procedures

This section includes a few examples of common procedures you may encounter. Lines that begin with ">" indicate that a command is being sent by the user to the remote control socket connection. If the command returns a value, the response is shown on the next line.

- "Connecting to an instrument and obtaining the remote control port number" on page 23
- "Starting Remote Control" on page 26
- "Application interaction" on page 26

Connecting to an instrument and obtaining the remote control port number

Remote control is accessed by connecting to the test unit on a specific port number. This number is determined by connecting to port 8000 and obtaining a port to access the module, then connecting to that port and querying for its remote control port. Below are a few examples of this process for specific platforms.

Connecting to the MTS-5800 Family and MAP-2100

The MTS-5800 and MAP-2100 instruments do not have a pluggable application module, so the parameters supplied to the commands below should always be `BOTH,BASE,"BERT"`.

1 Open a TCP/IP socket connection to the unit on port 8000.

```
> *REM
```

> MOD:FUNC:SEL? BOTH,BASE,"BERT" (Optional query to verify the module is powered on, returns "OFF" or "ON")

```
ON
```

> MOD:FUNC:PORT? BOTH,BASE,"BERT" (Query for the module's port number)

```
8002
```

2 Open a TCP/IP socket connection to the unit on the port number returned by the previous step.

> *REM > :SYST:FUNC:READY? BOTH,BASE,"BERT" (Optional query to verify the BERT application is running and ready to communicate, returns "0" or "1")

```
1
```

> :SYST:FUNC:PORT? BOTH,BASE,"BERT" (Query for the port number to remote control) 8003

3 The port number returned by the final command can be used to establish a remote control connection, as shown in "Starting Remote Control" on page 26.

Connecting to a MTS-6000A

The MTS-6000A only supports the use of a single module (MSAM or CSAM). Therefore when connecting to that platform, the parameters supplied to the commands below should always be `PWRS,SLIC1,"BERT"`.

1 Open a TCP/IP socket connection to the unit on port 8000.

`> *REM`

`> MOD:FUNC:SEL? PWRS,SLIC1,"BERT"` (Optional query to verify the module is powered on, returns "OFF" or "ON")

`ON`

`> MOD:FUNC:PORT? PWRS,SLIC1,"BERT"` (Query for the module's port number)

`8002`

2 Open a TCP/IP socket connection to the unit on the MSA port number returned by the previous step.

`> *REM`

`> :SYST:FUNC:READY? PWRS,SLIC1,"BERT"` (Optional query to verify the BERT application is running and ready to communicate, returns “0” or “1”)

`1`

`> :SYST:FUNC:PORT? PWRS,SLIC1,"BERT"` (Query for the port number to remote control)

`8003`

The port number returned by the final command can be used to establish a remote control connection, as shown in “Starting Remote Control” on page 26.

Connecting to a MTS-8000 transport module

The MTS-8000 supports the use of multiple transport modules. When connecting to one of these, the “slice id” must be provided to indicate the desired module. Slice ids are specified as SLIC1, SLIC2, etc. with SLIC1 being the closest module to the front display. The example below connects to the first (or only) slice, SLIC1.

1 1. Open a TCP/IP socket connection to the unit on port 8000. Lines starting with ">" indicate commands sent by the user.

`> *REM`

`> MOD:FUNC:SEL? BOTH,SLIC1,"BERT"` (Optional query to verify the module is powered on, returns "OFF" or "ON")

`ON`

`> MOD:FUNC:PORT? BOTH,SLIC1,"BERT"` (Query for the module's port number)

`8002`

2 Open a TCP/IP socket connection to the unit on the port number returned by the previous step.

```
> *REM
> :SYST:FUNC:READY? BOTH,SLIC1,"BERT"
```
(Optional query to verify the BERT application is running and ready to communicate, returns "0" or "1")

```
1
> :SYST:FUNC:PORT? BOTH,SLIC1,"BERT"
```
(Query for the port number to remote control)

```
8003
```

The port number returned by the final command can be used to establish a remote control connection, as shown in "Starting Remote Control" on page 26.

Connecting to an MTS-8000 dual module carrier supporting MSAM or CSAM

The MTS-8000 supports the use of multiple dual Module carriers, each of which may have up to two MSAM and/or CSAM modules. When connecting to one of these, the "side" and the "slice id" must be provided to indicate the desired module. The side parameter is either OPPS - indicating the MSAM or CSAM is on the side opposite the power connector (the left side when viewed from the front), or PWRS - on the same side as the power connector (the right side when viewed from the front). Slice ids are specified as SLIC1, SLIC2, etc. with SLIC1 being the closest module to the front display. The example below connects to an MSAM on left side (OPPS) of the first slice (SLIC1).

1 Open a TCP/IP socket connection to the unit on port 8000.

```
> *REM
> MOD:FUNC:SEL? OPPS,SLIC1,"BERT"
```
(Optional query to verify the module is powered on, returns "OFF" or "ON")

```
ON
> MOD:FUNC:PORT? OPPS,SLIC1,"BERT"
```
(Query for the module's port number)

```
8002
```

2 Open a TCP/IP socket connection to the unit on the port number returned by the previous step.

```
> *REM
> :SYST:FUNC:READY? OPPS,SLIC1,"BERT"
```
(Optional query to verify the BERT application is running and ready to communicate, returns "0" or "1")

```
1
> :SYST:FUNC:PORT? OPPS,SLIC1,"BERT"
```
(Query for the port number to remote control)

```
8003
```

The port number returned by the final command can be used to establish a remote control connection, as shown in "Starting Remote Control" on page 26.

Starting Remote Control

Once the port number for remote control has been obtained (see "Connecting to an instrument and obtaining the remote control port number" on page 23), remote control mode may be activated with the *REM command. It is recommended that the "`:SYSTem:ERRor?`" command be sent after every non-query command, as this is the easiest way to verify that each command is processed successfully. Lines starting with ">" indicate commands sent by the user.

Remote Control with GUI disabled (default)

This command will shut down the GUI and display a message indicate the unit is under remote control. All running applications will be stopped.

`> *REM` (This may take a few seconds if an application is running...)

`> :SYSTem:ERRor?` (This command may be sent in the mean time, but will not be processed until *REM is finished.)

`0, "No error"` (Once this message is received, the switch to remote control mode is complete.)

Remote Control with GUI enabled

This command does not shut down the GUI, or any running applications.

> **`*REM VISIBLE`** (This may take a few seconds.)

`> :SYSTem:ERRor?` (This command may be sent in the mean time, but will not be processed until *REM is finished.)

`0, "No error"` (Once this message is received, the switch to remote control mode is complete.)

Application interaction

Querying for running applications

Once the test unit is in remote control mode, the following commands may be useful to determine what applications can be launched.

`> :SYSTem:APPLication:APPLications?`

Get a list of available application names. The return value for this command depends on what is currently running. To get the list of all applications the unit supports, send this query when no applications are running.

`TermDs1Bert,ThruDs1Bert,SingleMonDs1Bert,...`

`> :SYSTem:APPLication:CAPPlication?`

Get a list of identifiers for applications that are currently running.

TermStm4Au3Vc3Bert_121,TermEth1GL3Traffic_122

> :SYSTem:APPLication:LAUNch?

Get the identifier of the most recently launched application.

TermEth1GL3Traffic_122

Launching an application

Once the test unit is in remote control mode, the following commands may be sent to launch an application (in this example, 1GigE L3 Traffic Terminate).

> :SYSTem:APPLication:LAUNch TermEth1GL3Traffic

Launch the application (same as "Add test" on the GUI; other running applications will not be shut down by this command). This may take a few seconds...

> :SYSTem:ERRor?

This command may be sent in the mean time, but will not be processed until the application launch is finished.

0, "No error"

Once the application is running, this message will be returned.

> :SYSTem:APPLication:LAUNch?

Get the id of the most recently launched application.

TermEth1GL3Traffic_121

> :SYSTem:APPLication:SELect TermEth1GL3Traffic_121

Select that application id.

> :SYSTem:ERRor?

0, "No error"

> :SESSion:CREate

Create a new remote session

> :SESSion:STARt

Start the session

> :SYSTem:ERRor?

0, "No error"

> :INITiate

Start the test. (When applications are launched through remote control, they begin in the "stopped" state).

> :SYSTem:ERRor?

0, "No error"

When using remote control with the GUI enabled and an desired application is already running, a similar process can be followed to connect to it. Use the ":SYSTem:APPLication:CAPPli-

cation?" query to find its application id, select the application, then create and start a remote session. "**:**INITiate" only needs to be sent if the test is in the "stopped" state.

This example also illustrates use of the **:**SYSTem**:**ERRor? query. The user should have this command sent automatically after every non-query command and the 0, "No error" response should be expected. Doing so will provide vital information in the event of a command error or connection issue.

Restarting the test

To restart the test using remote control, send the test stop command followed by a test start.

> :ABORt	Test stop
> :INITiate	Test start

Using setups, actions, and results (inserting and checking an alarm)

The following commands demonstrate the use of application setups, actions and results. In this example, assume we have launched and initialized the DS1 BERT Terminate application (TermDs1Bert). Lines starting with ">" indicate commands sent by the user.

> **:SOURce:ALARm:TYPE?**

Query the current alarm type setup

"Ds1LofAlarm"

Note: some query commands return values with double quotes (") and some do not. Making remote control scripts quote-insensitive when checking returned values is highly recommended.

> :SOURce:ALARm:TYPE Ds1AisAlarm

Set the alarm type setup to AIS.

> :SOURce:PDH:DS1:INSert:AIS ON

Enable the AIS alarm insertion action

> :SOURce:PDH:DS1:INSert:AIS?

Query the current state of the AIS alarm insertion action

ON

> :SENSe:DATA? CSTatus:PDH:DS1:AIS

Query the AIS alarm result

1

If the unit is self-looped, it will detect the AIS.

> :SOURce:PDH:DS1:INSert:AIS OFF

Turn the alarm back off

> :SENSe:DATA? CSTatus:PDH:DS1:AIS

Query the AIS alarm result again

0

Configuring a timed test

There are three test timing modes: not timed, timed test, and delayed-start timed test. This example demonstrates how to configure each of those.

> :SENSe:TEST:ENABle OFF

Sets the test type to “Not Timed”

> :SENSe:TEST:ENABle ON

Sets the test type to “Timed Test”

> :SENSe:TEST:DURation 3600

Set the timed test duration in seconds

> :SENSe:TEST:ENABle DELAYED

Sets the test type to “Delayed-start timed test”

> :SENSe:TEST:DURation 3600

Set the timed test duration in seconds

> :SENSe:TEST:DELay:DATE "10/31/2011"

Set the start test date

> :SENSe:TEST:DELay:TIME "23:00:00"

Set the start test time

Exiting an application

To shut down an application through remote control, it must first be selected. If the application is already selected and a session has been created, that session should be ended.

> :SYSTem:APPLication:SELect TermDs1Bert_121

> :EXIT

or

> :SESSion:END

> :EXIT

Leaving remote control mode

The *GUI command shuts down all applications running under remote control and restores the GUI interface. This command may take some time to finish processing. The “:SYSTem:ERRor?” query will still indicate when *GUI is complete, but most other remote control commands (except *REM, *RST, etc.) will no longer work.

> *GUI

Return to GUI mode, make take a while...

```
> :SYSTem:ERRor?
```

This command will return once the GUI is back up.

```
0, "No error"
```

Demo Remote Control Scripts

The following Perl scripts are provided on the USB stick to demonstrate how to establish and use a remote control session to the various platforms. They are all similar, only differing in the way the script arguments are processed.

RC-Transport.pl - to connect to a Transport Module in a T-BERD/MTS-8000
Usage: **`perl RC-Transport.pl <unit ip> <slice #>`**

RC-6000A.pl - to connect to an MSAM or CSAM module a T-BERD/MTS-6000A
Usage: **`perl RC-6000A.pl <unit ip>`**

RC-DMC.pl - to connect to an MSAM module a T-BERD/MTS-8000 dual Module carrier with MSAM or CSAM
Usage: **`perl RC-DMC.pl <unit ip> <side> <slice #>`**

RC-5800.pl - to connect to an MTS-5800
Usage: **`perl RC-5800.pl <unit ip>`**

Script parameters:

- **`<unit ip>`** - the IP address of the unit
- **`<side>`** - optional, specifies which slot of the DMC to communicate with
  - **`OPPS`** - (default) left side (“opposite side” from the power connector)
  - **`PWRS`** - right side (“power side”, same side as the power connector)
- **`<slice #>`** - optional, specifies which module to communicate with
  - **`1`** - (default) the module closest to the front display
  - **`2`**, **`3`**, etc - second or third modules, counting from the front

Automation using SCPI Commands - Steps to Connect

This section includes procedures for automation using PuTTY and SCPI, and a working example of getting BERT bit error results in an Ethernet layer 2 application.

This section applies to all platforms including SC4800, 5811P, 5822P, 5800-100G, MAP-2100, and the ONA-100.

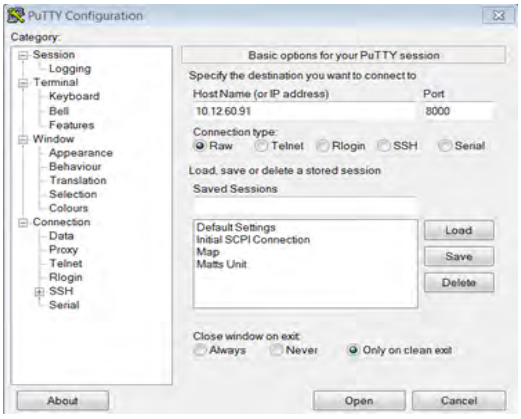
NOTE

PuTTY is available from https://www.putty.org

Setting up Connectivity

1 Launch PuTTY.

2 Connect to the unit's IP address using port 8000 and the connection type **Raw**.



3 From the shell prompt, issue the **`*REM`** command.
The **`*REM`** command must be issued before any other commands.

4 Query identification information about the unit by issuing the *IDN? command.
The unit will return a string such as:

```
Viavi Solutions,"T-BERD5800
100G","WMSE0075160050","BERT","V26.0.0"
```

or

```
Viavi Solutions,"MTS5800V2","WMME0031140091","BERT","V26.0.0"
```

Such a response confirms that the unit is alive, and can be interpreted as follows:

- **The unit type**: `T-BERD5800V2, MTS5800V2, T-BERD5800-100G, MTS5800-100G` or `MAP-2100`.
- **The serial number**, for example `"WMSE0075160050"` or `"WMME0031140091"`
- `"BERT"` indicates that the unit is a Bit Error Rate Tester.
- **The BERT software version**, for example: `"V26.0.0"`

5 Issue the following command:

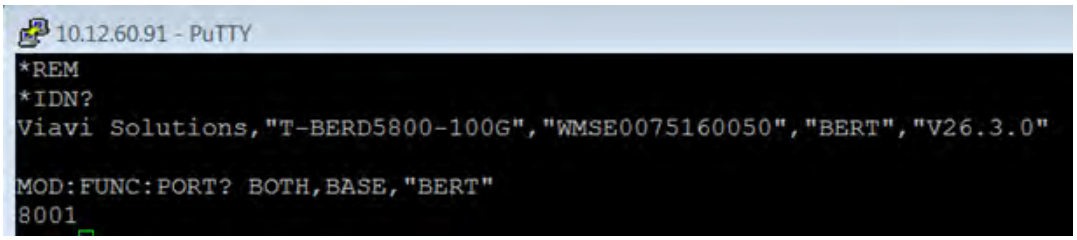
`MOD:FUNC:LIST? BOTH,BASE`

This command will return `"BERT"`

6 Issue the following command to determine the correct port to use.

`MOD:FUNC:PORT? BOTH,BASE,"BERT"`

The command will return a port number such as `8001`.

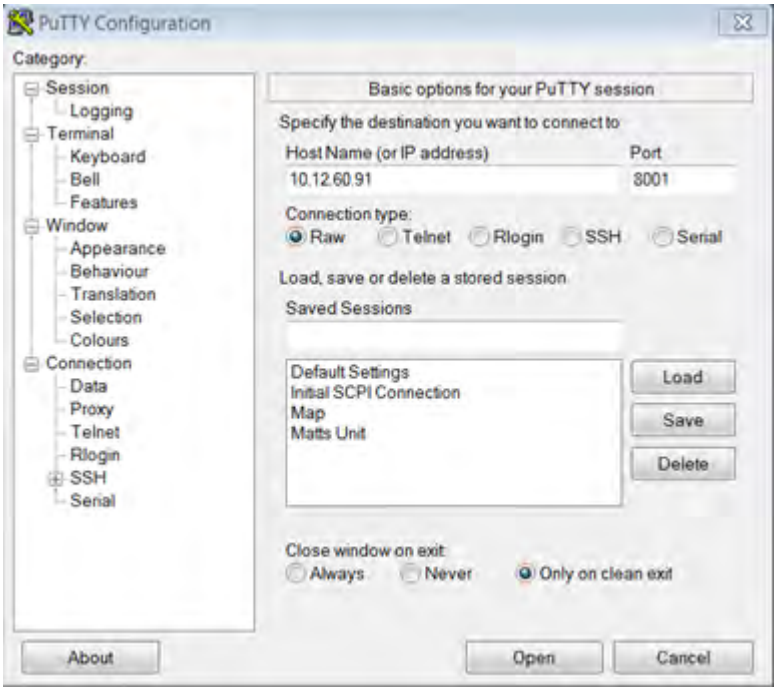


NOTE

If the system is not in a ready state, the response will be `-1`.

7 Close the session from PuTTY.

8 Reconnect to the same IP address using the port returned in step 6, again using a **Raw** connection type.



9 From the shell prompt, issue the **`*REM`** command.
The **`*REM`** command must be issued before any other commands.

10 Query identification information about the unit by issuing the **`*IDN?`** command.

The unit will return a string such as:

```
Viavi Solutions,"T-BERD5800
100G","WMSE0075160050","BERT","V26.0.0"
```

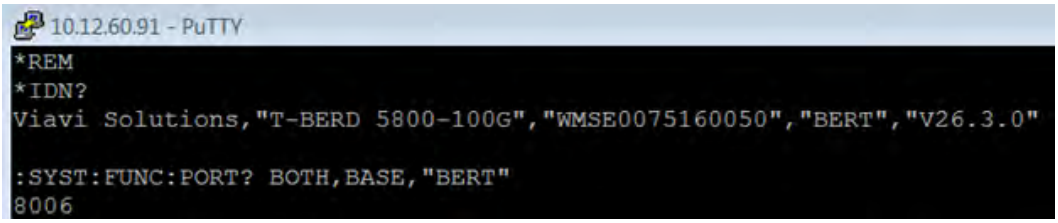
or

```
Viavi Solutions,"MTS5800V2","WMME0031140091","BERT","V26.0.0"
```

11 Issue the following command to determine the correct port to use for automation.

`:SYST:FUNC:PORT? BOTH,BASE,"BERT"`

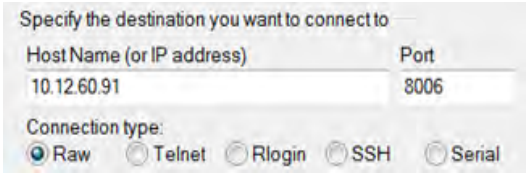
This command will return a port number such as `8006`.



If the system is not in a ready state, the response will be `-1`.

12 Close the session from PuTTY.

13 Reconnect to the same IP address using the port returned in step X, and the **Raw** connection type.



14 Issue the following command. This must be the first command issued after reconnecting to the unit.

`*REM VISIBLE FULL`

`VISIBLE FULL` keeps the UI displayed on the unit. Omit **`VISIBLE FULL`** if no User Interface control is desired. If **`VISIBLE FULL`** is not used, the UI will appear as shown below.

If **`VISIBLE FULL`** is used, then the UI remains available.

15 Issue the following command.

`*IDN?`

The unit will return a string such as:

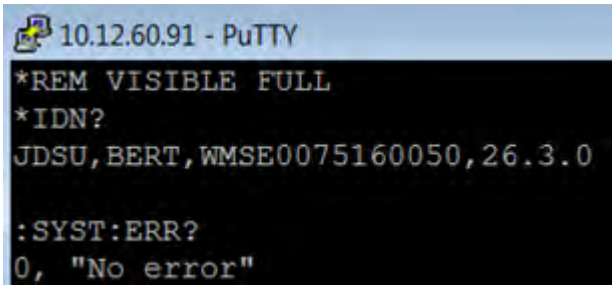
```
"JDSU,BERT, WMSE0075160050,26.0.0"
```

NOTE

If logged into the UI, the UI will now say:

```
"Remote Control is in use for this module and the display has
been disabled"
```

At any point during these steps it is possible to check errors with the command **:SYST:ERR?**



Example: Get BERT bit error results in an Ethernet layer 2 application

For more information on the commands shown below, see the"SCPI command reference" on page 38.

This procedure applies to the map-2100 only.

1. List Currently Running Applications.

   :SYST: APPL:CAPP?

   TermOC3Sts3cBert_101,TermEth10GL2Traffic_102

2. Select the 10G application.

   :SYST:APPL:SEL TermEth10GL2Traffic_102

   :SYST:ERR?

   0, "No error"

3. Create a Remote Control session.

   :SESS :CRE

4. Start a Remote Control session.

   :SESS:STAR Word file has a screen shot here showing :SENS:DATA?

5. Exit from the 10G app.

   :EXIT

   :SYST:ERR?

0, "No error"

6 Launch the 100GE Layer 2 Traffic app on port 2

:SYST:APPL:LAUN TermEth100GL2Traffic_102

:SYST:ERR?

0, "No error"

:SYST: APPL:CAPP?

TermOC3Sts3cBert_101,TermEth100GL2Traffic_102

:SYST:APPL:SEL TermEth100GL2Traffic_102

:SYST:ERR?

0, "No error"

7 Create a Remote Control session.

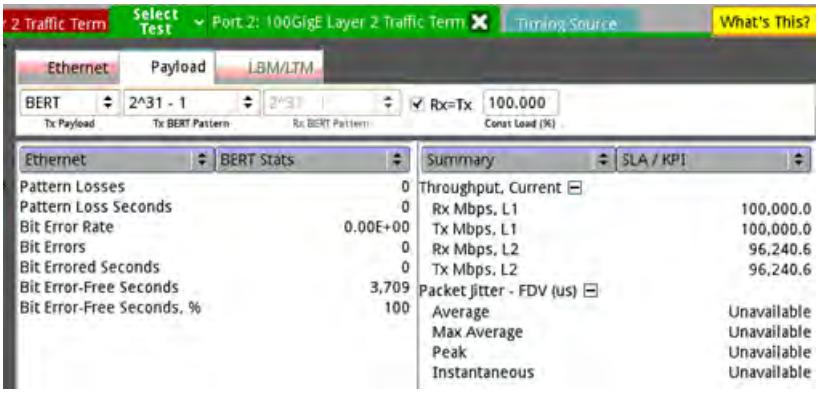
:SESS:CRE

8 Start a Remote Control session.

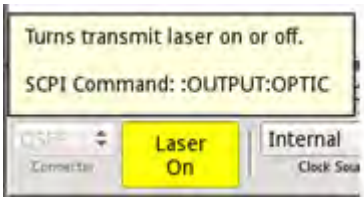
:SESS:STAR

9 On the unit's graphical interface, select the **What's This?** button.

It will turn yellow.

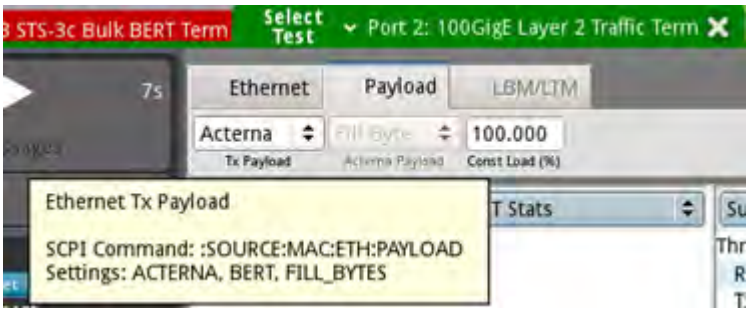


10 Click on a command such as **Laser On**.



The SCPI command issued is automatically added to the clipboard.

11 Change the payload to BERT.

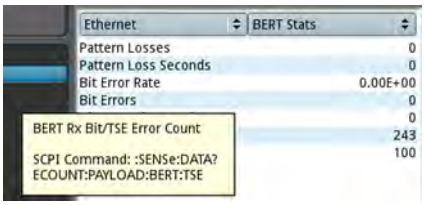



```
:SOURCE:MAC:ETH:PAYLOAD BERT
:SYST:ERR?
0, "No error"
:SOURCE:MAC:ETH:PAYLOAD?
 BERT
```

12 Restart a test.

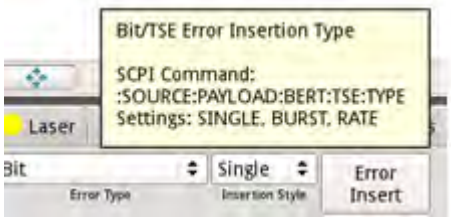
```
:ABOR
:INIT
```

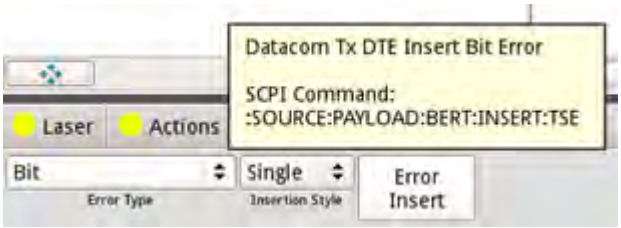
13 Query the number of BERT errors.




```
:SENS:DATA? ECOUNT:PAYLOAD:BERT:TSE
0
```

14 Inject an error.





```
:SOURCE:PAYLOAD:NBERT:TSE:TYPE?
SINGLE
:SOURCE:PAYLOAD:BERT:INSERT:TSE
:SENS:DATA? ECOUNT:PAYLOAD:BERT:TSE
1
:SENS:DATA? ERATE:PAYLOAD:BERT:TSE
1.07692e-13
```

15 End the remote session.

```
:SESS:END
:SYST:ERR?
0, "No error"
```

MAP-2100 Optical Switch Automation

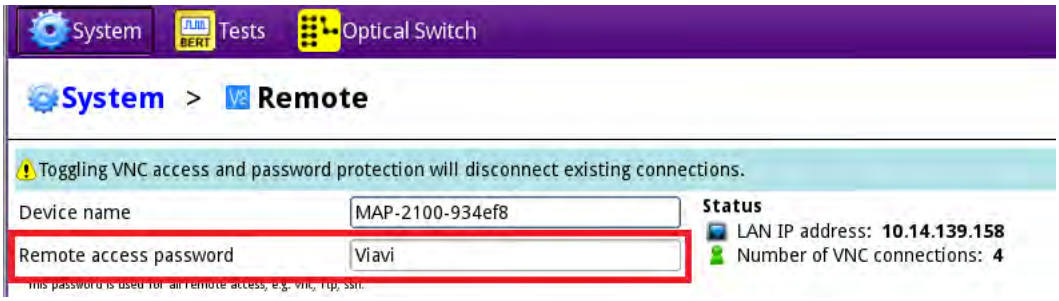
The user can manipulate the use of the optical switch through Telnet or SSH communications.

From the command line, enter **`optswitch`** to view the available commands and functions of the optical switch.

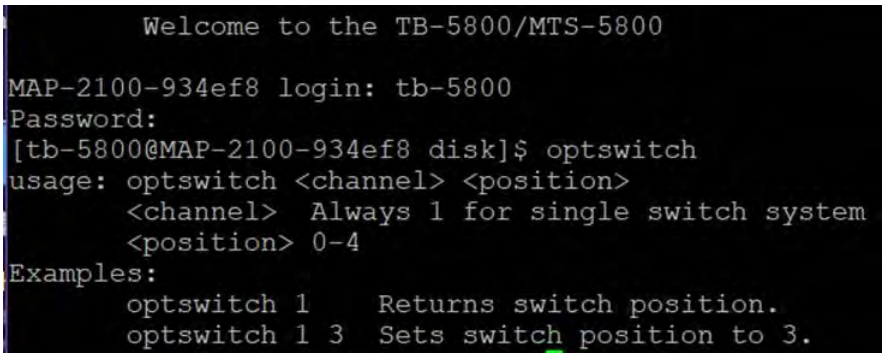
NOTE

The command **`optswitch`** is not a SCPI command.

For a Telnet connection, use port 23; for an SSH connection, use port 22. Use the ID **`tb-5800`** or **`mts-5800`**, and use the remote access password provided in the GUI (**System** > **Remote**).



The following figure shows an example of a telnet session.



NOTE

There is a delay of up to 3 seconds from the command input to execution. The UI will not update in the optical switch application when manipulated in this manner.

SCPI command reference

All SCPI commands are on the unit. Click the **What's This?** button at the top and select the item to view its corresponding SCPI command. Not all items have SCPI commands.

Automation with ONA-800 and ONA-1000

This section indicates how to communicate with the mainframe and obtain module information. To connect to Ethernet/BERT applications on ONA-1000 please refer to section "Automation using SCPI Commands - Steps to Connect" on page 30.

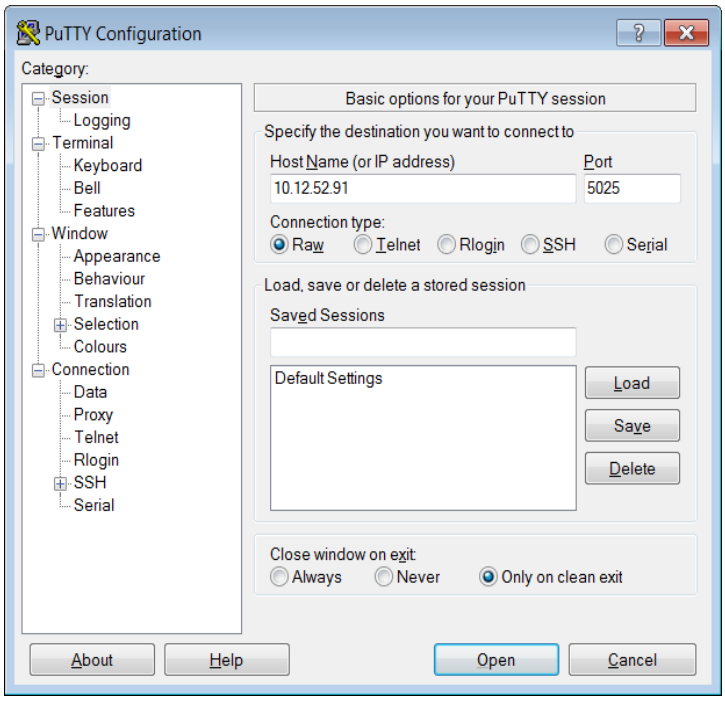
Starting remote operation

1 Open a TCP/IP socket connection to the unit on port 5025.

`telnet <IP> 5025`

Alternatively,

Launch PuTTY and connect to the unit's IP address using port 5025 and the connection type **Raw**.



2 Start remote operation.

`*REM`

The **`*REM`** command must be issued before any other commands.

3 Query identification information about the unit by issuing the *IDN? command.

The unit returns a string based on the platform. Bellow are two examples.

```
"Viavi Solutions","ONA-800","ABCD123","platform-version:
1.1.0, caa-solution: 6.010.002b, fiber-optic: 18.94"
```

```
"Viavi Solutions","ONA-1000"," ABCD123","bert_tester: 1.1.0,
platform-version: 2.1.0, fiber-optic: 18.94"
```

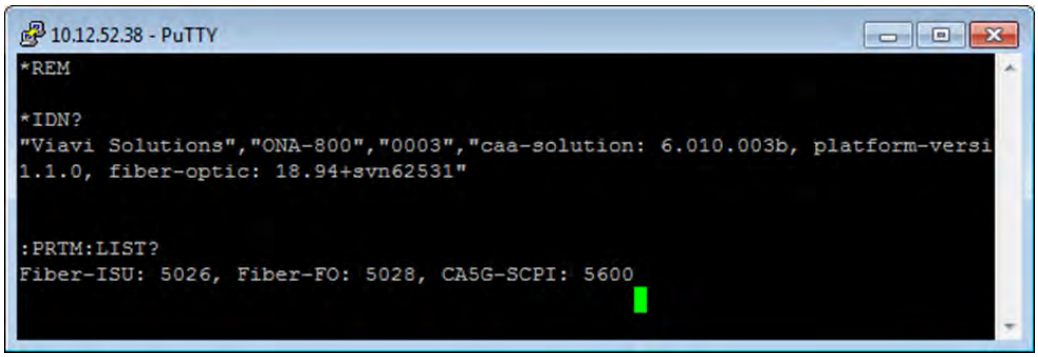
4 Issue the following command to get the port numbers associated with the attached modules.

`:PRTM:LIST?`

This command will return the port numbers, such as below, based on the platform and modules present. Below are two examples.

```
Fiber-ISU: 5026, Fiber-FO: 5028, CA5G-SCPI: 5600
```

```
Fiber-ISU: 5026, Fiber-FO: 5028, BERT: 8002
```



5 Close the session with PuTTY.

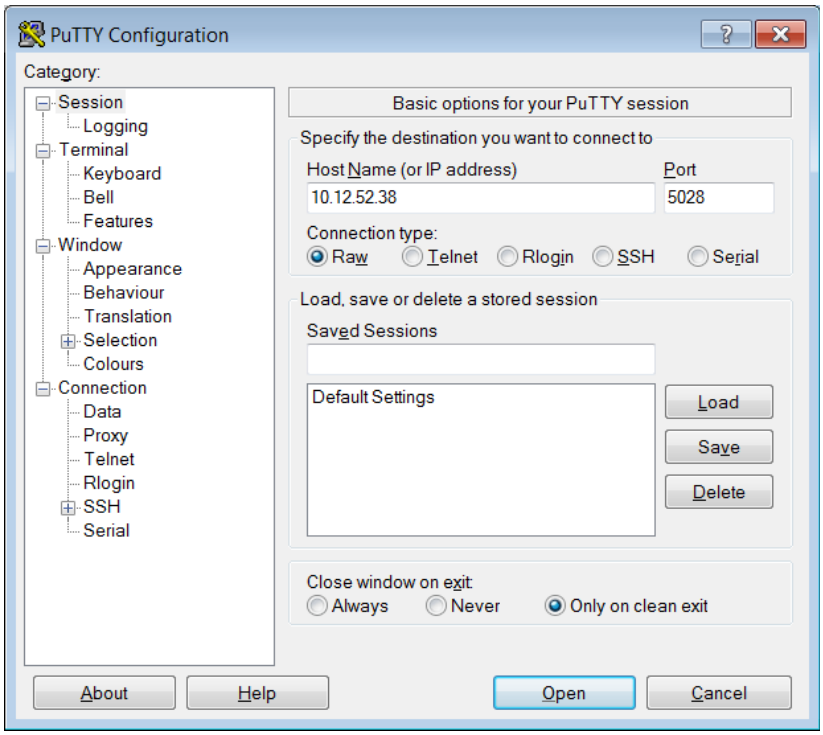
Starting remote operation with module

1 Open a TCP/IP socket connection to the module using the port number obtained in Step 4 in the previous procedure.

`telnet <IP> <Port number>`

Alternatively,

Launch PuTTY and connect to the unit's IP address using the port number obtained in Step 4 in the previous procedure and the connection type **Raw**.

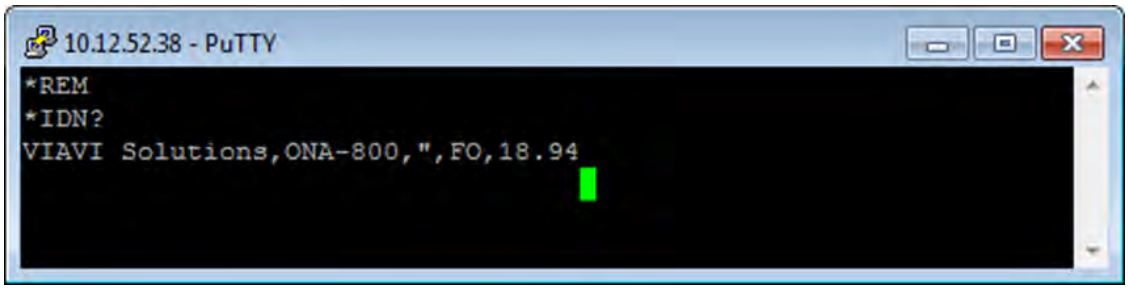


2 Start remote operation.

***REM**

The ***REM** command must be issued before any other commands.

3 Query identification information about the unit by issuing the *IDN? command.



4 Refer to the module's remote control guide for a list of commands that are available.

List of commands for ONA-800 and ONA-1000 Base Unit

esr_response()

***ESR?**

Returns: The current error register value.

- 0 = STATUS_CLEAR
- 16 = EXECUTION_ERROR
- 32 = INSTRUCTION_ERROR

Example:

***ESR?**

```
>> 0
```

idn_response()

***IDN?**

Returns: The instrument identification string.

Example:

***IDN?**

```
>> "Viavi Solutions","ONA-800","ABCD123","platform-version:
1.1.0, caa-solution: 6.010.002b, fiber-optic: 18.94"
```

prtm_list_response()

:PRTM:LIST?

Returns: A list of clients/modules and corresponding remote control ports.

Example:

```
:PRTM:LIST?
>> Client1: 5027, Client2: 5028
```

stsync_acntid_action(param_list)

:STSYnc:ACNTid

Set the StrataSync account ID.

Parameters:param_list – the account ID

Example:

```
:STSY:ACNT my_tech_id
```

stsync_acntid_response()

:STSYnc:ACNTid?

Get the StrataSync account ID.

Example:

```
:STSY:ACNT?
>> my_tech_id
```

stsync_srvname_action(param_list)

:STSYnc:SRVName

Set the StrataSync server name.

Parameters:param_list – the server name

Example:

```
:STSY:SRV http://my_server.com
```

stsync_srvname_response()

:STSYnc:SRVName?

Get the StrataSync server name.

Example:

```
:STSY:SRV?
>> http://my_server.com
```

stsync_status_response()

:STSYnc:STATus?

Get the StrataSync sync status.

Example:

```
:STSY:STAT?
>> TSC_STATUS_SUCCESS
```

stsync_techid_action(param_list)

:STSYnc:TECHid

Set the StrataSync tech ID.

Parameters:param_list – the tech ID

Example:

```
:STSY:TECH 1234
```

stsync_techid_response()

:STSYnc:TECHid?

Get the StrataSync tech ID.

Example:

```
:STSY:TECH?
>> 1234
```

system_reboot_action()

:SYSTem:REBoot

Initiates a system reboot.

system_vncrestart_action()

`:SYST:VNCRestart`

Restarts the VNC server

Appendix: Special Setup Types

MAC Addresses

MAC addresses are set as an array of 6 integers between 0-255. These integers are determined taking 6 pairs of hexadecimal digits from the MAC address and converting them to decimal. For example, to set the destination MAC address to 00-80-16-8A-FF-C2:

```
:SOURce:MAC:ETH:DESTination:ADDRess (@0) 0
:SOURce:MAC:ETH:DESTination:ADDRess (@1) 128
:SOURce:MAC:ETH:DESTination:ADDRess (@2) 22
:SOURce:MAC:ETH:DESTination:ADDRess (@3) 138
:SOURce:MAC:ETH:DESTination:ADDRess (@4) 255
:SOURce:MAC:ETH:DESTination:ADDRess (@5) 194
```

IP Addresses

IP Addresses are specified as a single 32-bit integer given by the following formula:

Setup value = (first IP number) x 2563 + (second IP number) x 2562 + (third IP number) x 256 + (fourth IP number)

For example to set the destination IP address to 192.168.1.3:

```
:SOURce:MAC:IP:DESTination:ADDRess 3232235779
```

IPv6 Addresses

To specify an IPv6 address through remote control, simply provide or read the address as a string.

```
:SOURce:DATA:MAC:IPV6:TX:DESTination:ADDRess "ff00::1234:2345"
```

Arrays

Array setups or results contain multiple values that are accessed using a single command. Querying the value of the setup or result will return all of the values as a comma-separated list. Alternatively, a zero-based index can be provided to query or set a single element from the array using the following syntax:

To query the value of 4th element:

```
:SOURce:PDH:E1:CHANnel? (@3)
```

To set the 8th element to 1:

```
:SOURce:PDH:E1:CHANnel (@7) 1
```

To set the 3rd, 4th, 5th and 8th elements to 1 (note only one setting may be specified):

```
:SOURce:PDH:E1:CHANnel (@2:4,7) 1
```

Character Arrays

Character arrays are used to represent text. Each element of the array is an integer between 0-255 indicating the ASCII value of a character. The end of the text is marked by the first 0 in the array. See the table below for a list of ASCII values.

Example: To set the expected SDH HP trace id to "JDSU 8000":

```
:SENSe:SDH:HP:EXPected:TRACe (@0) 74
:SENSe:SDH:HP:EXPected:TRACe (@1) 68
:SENSe:SDH:HP:EXPected:TRACe (@2) 83
:SENSe:SDH:HP:EXPected:TRACe (@3) 85
:SENSe:SDH:HP:EXPected:TRACe (@4) 32
:SENSe:SDH:HP:EXPected:TRACe (@5) 56
:SENSe:SDH:HP:EXPected:TRACe (@6) 48
:SENSe:SDH:HP:EXPected:TRACe (@7) 48
:SENSe:SDH:HP:EXPected:TRACe (@8) 48
:SENSe:SDH:HP:EXPected:TRACe (@9) 0
```

Table of commonly used ASCII values

32	Space	48	0	64	@	80	P	96	`	112	p
33	!	49	1	65	A	81	Q	97	a	113	q
34	"	50	2	66	B	82	R	98	b	114	r
35	#	51	3	67	C	83	S	99	c	115	s
36	$	52	4	68	D	84	T	100	d	116	t
37	%	53	5	69	E	85	U	101	e	117	u
38	&	54	6	70	F	86	V	102	f	118	v
39	'	55	7	71	G	87	W	103	g	119	w
40	(	56	8	72	H	88	X	104	h	120	x
41	)	57	9	73	I	89	Y	105	i	121	y
42	*	58	:	74	J	90	Z	106	j	122	z
43	+	59	;	75	K	91	[	107	k	123	{
44	,	60	>	76	L	92	\	108	l	124	|
45	-	61	=	77	M	93	]	109	m	125	}
46	.	62	<	78	N	94	^	110	n	126	~
47	/	63	?	79	O	95	_	111	o		

VCG configuration in Nextgen SONET/SDH

Virtual Concatenated Groups (VCGs) are configured using arrays for the channel and sequence numbers, and a setup that indicates the number of members in the VCG. Channel numbers are specified in NKLM format. In SONET, N is the STS-N channel, and L and M are the VT Group and the VT-1.5 channels respectively. In SDH, N is the STM-N channel and K, L and M are the TUG3, TUG2 and TU-12 channels. If any of these settings do not apply to the given mapping, 0 should be used as a placeholder. It is important to set the number of members after setting the channel and sequence numbers because the number of members is used to detect changes to those arrays. If the VCG has the same number of members as the previous VCG, then number of members should first be set to 0, then back to the correct setting to ensure the change is recognized.

Below are the commands to configure a VCG (with matching Tx and Rx settings) in a SONET Vt-1.5-Xv application using the first 3 Vt-1.5 channels on STS-N channel 14.

```
:SOURce:SONet:EXPected:SEQ (@0) 0
:SOURce:SONet:NKLM (@0) 14011
:SOURce:SONet:EXPected:SEQ (@1) 1
:SOURce:SONet:NKLM (@1) 14012
:SOURce:SONet:EXPected:SEQ (@2) 2
:SOURce:SONet:NKLM (@2) 14013
```

```
:SENSe:SONet:EXPected:SEQ (@0) 0
:SENSe:SONet:NKLM (@0) 14011
:SENSe:SONet:EXPected:SEQ (@1) 1
:SENSe:SONet:NKLM (@1) 14012
:SENSe:SONet:EXPected:SEQ (@2) 2
:SENSe:SONet:NKLM (@2) 14013
:SOURce:SONet:VCATlcas:NUM:MEMBers 0
:SENSe:SONet:VCATlcas:NUM:MEMBers 0
:SOURce:SONet:VCATlcas:NUM:MEMBers 3
:SENSe:SONet:VCATlcas:NUM:MEMBers 3
```

Selecting VCG members for error/alarm insertion

Members are selected for error or alarm insertion using an array of flags indicating whether each member will receive an error or not. To insert on one member, set the entire selection array to 0's and set the element for the desired member to 1:

```
:SOURce:SONet:PATH:INS:SELect (@0-63) 0
:SOURce:SONet:PATH:INS:SELect (@10) 1
:SOURce:SONet:VT:INSert:B3
```

To insert the error or alarm on all members, set the entire selection array to 1's:

```
:SOURce:SONet:PATH:INS:SELect (@0-63) 1
:SOURce:SONet:VT:INSert:B3
```