Tektronix®

TekExpress® PCI Express
Transmitter Compliance and Testing Solution Software

Printable Application Help

077-0774-04

Tektronix®

TekExpress® PCI Express
Transmitter Compliance and Testing Solution Software
Printable Application Help

www.tek.com
077-0774-04

Contacting Tektronix

Tektronix, Inc.
14150 SW Karl Braun Drive
P.O. Box 500
Beaverton, OR 97077
USA

For product information, sales, service, and technical support:

- In North America, call 1-800-833-9200.
- Worldwide, visit *www.tek.com* to find contacts in your area.

Table of Contents

Running tests

Saving and recalling test setups

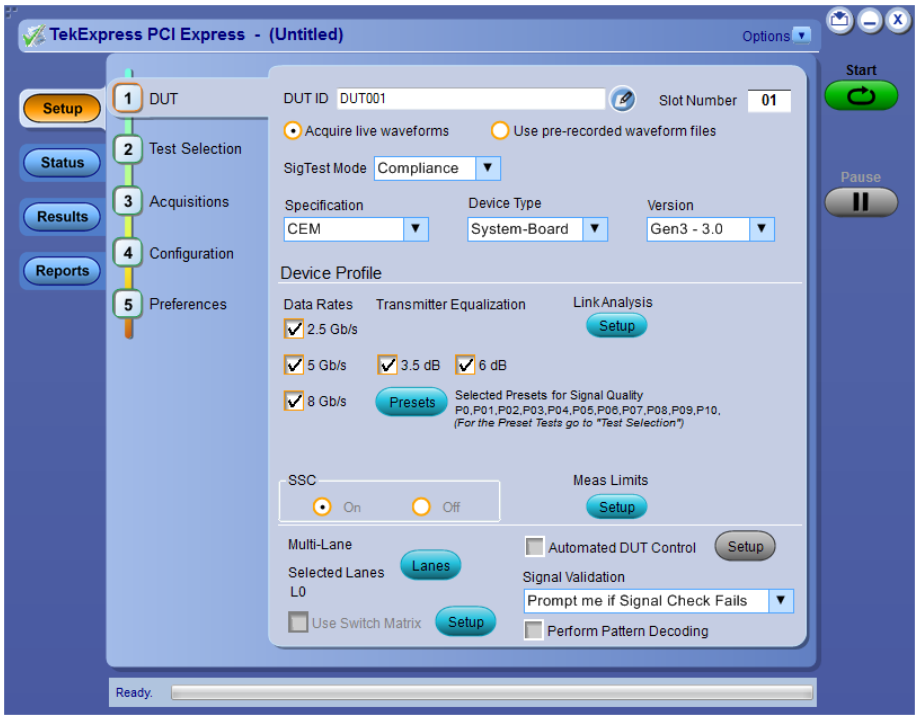
Switch Matrix application

TekExpress programmatic interface

SCPI commands

Reference

Welcome



Welcome to the TekExpress® PCI Express Automated Test Solution Software application (referred to as TekExpress PCIe or PCIe in the rest of the document). TekExpress PCIe provides an automated, simple, and efficient way to test PCI Express interfaces and devices consistent to the requirements of the PCI Express specifications.

Tek Express PCIe key features and benefits

New features from current release

- RF Switch support to test the x12 and x16 lanes using Keithley and Gigatronics switches respectively
- Supports Base specification Gen3 - 3.0 Tx Test Board DUT Type
- Supports P75xx, P76xx, and P77xx Trimode probes
- Simple push button, enabling the users to manually toggle PCIe presets from AWG/AFG
- Clock lane support in differential mode for the System-Board device type of CEM specification and Host device type of U.2 (SFF-8639) specification
- Updated mask co-ordinates for the Add-in-card device type of CEM specification, of Gen2 in DLL mode

Features of previous release

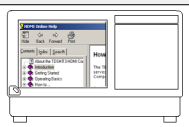
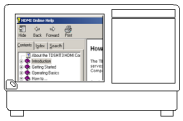
- Support U.2 (SFF-8639) specification, provided the machine has SigTest v3.2.0 with U.2 templates
- Trigger type support for Gen3 (Auto/Width/Edge)
- Support new SigTest v3.2.0 template with pattern check
- TekExpress setup files in-line with PCI-SIG Compliance Workshop
- Faster test execution time with improved Autoset
- Automates compliance measurements for PCI Express 3.0 CEM Specification Rev 0.9. for the following configurations:
  - PCIE_1_0a- PCIEX_TX_ADD_CON_250UI
  - PCIE_1_0a- PCIEX_TX_SYS_CON_250UI
  - PCIE_CEM_CARD_1_1
  - PCIE_CEM_SYS_1_1
  - PCIE_2_0_CARD
  - PCIE_2_0_SYS
  - PCIE_3_0_CARD
  - PCIE_3_0_SYS
- Fully automated General, Jitter, Composite Eye, Transition Eye, and Non Transition Eye measurements
- Provides both an automation solution (for compliance) and DPOJET (for debug)
- The PCI-SIG® PCI Express Compliance Test Library is integrated into the TekExpress framework
- Reduces the time required to conduct testing
- Minimizes user intervention when conducting time-consuming testing
- Enables loading filter files to support system and add-in card measurements
- Performs fully-automated testing for system and add-in card measurements
- Provides individual or group test selection by using a tree-structure menu
- Built-in reporting features:
  - Provides a Pass/Fail summary table
  - Provides margin details on each test
  - Provides a consolidated report for all tests
- Complete programmatic interface enables automation scripts to call PCIe functions

Getting help and support

Related documentation

The following manuals are available as part of the TekExpress PCIe Compliance and Debug Solution documentation set.

Table 1: Product documentation

Item	Purpose	Location
Application Help	In-depth operation and UI help	
PDF of the Application Help	Printable version of the compiled Application help	

See also

Technical support

Conventions used in help

Application Help uses the following conventions:

- The term “DUT” is an abbreviation for Device Under Test.
- The term “select” is a generic term that applies to the two methods of choosing an option: using a mouse or using the touch screen.
- The term "close" refers to normally close the relay signal (select the signal).
- The term "open" refers to normally open the relay signal (unselect the signal).
- The term "channel" refers to the relay signal or oscilloscope channel.

Technical support

Tektronix values your feedback on our products. To help us serve you better, please send us your suggestions, ideas, or comments on your application or oscilloscope. Contact Tektronix through mail, telephone, or the Web site. See *Contacting Tektronix* for more information.

When you contact Tektronix Technical Support, please include the following information (be as specific as possible):

General information

- All instrument model numbers
- Hardware options, if any
- Probes used
- Your name, company, mailing address, phone number, FAX number
- Please indicate if you would like to be contacted by Tektronix about your suggestion or comments.

Application specific information

- Software version number
- Description of the problem such that technical support can duplicate the problem
- If possible, save the setup files for all the instruments used and the application
- If possible, save the TekExpress setup files, log.xml, *.TekX (session files and folders), and status messages text file
- If possible, save the waveform on which you are performing the measurement as a .wfm file

Getting started

Minimum system requirements

The following table shows the minimum system requirements needed for an oscilloscope to run TekExpress PCI Express.

Table 2: System requirements

Component	Requirement
Oscilloscope	See *Instruments and accessories required*
Processor	Same as the oscilloscope
Operating system	Microsoft Windows 7 (64-bit only) *Required Windows 7 user account settings*
Memory	Same as the oscilloscope
Hard disk	Same as the oscilloscope
Display	Same as the oscilloscope [1]
Firmware	Tekscope 10.3.0 or later for MSO/DSA/DPO70000C,D,DX
Software	■ DPOJET, Jitter and Eye Diagram Analysis Tool [2] ■ Microsoft .NET 4.0 Framework ■ Microsoft Internet Explorer 8.0 SP1 or later ■ PyVisa version 1.0.0 ■ IronPython version 2.7.3 ■ Microsoft Photo Editor 3.0 or equivalent software for viewing image files ■ Adobe Reader 7.0 or equivalent software for viewing portable document format (PDF) files

[1] If TekExpress is running on an instrument having a video resolution lower than 800x600 (for example, a sampling oscilloscope), it is recommended that you connect a secondary monitor, which must be enabled before launching the application.

[2] For software version, refer to Readme TekExpress PCI Express.txt file at C:\Program Files (x86)\Tektronix \TekExpress\TekExpress PCI Express

Instruments and accessories required

The following table lists the instruments and accessories required for TekExpress PCI Express application.

Table 3: Instruments and accessories required for PCI Express application

Instrument/Accessory	Model number
Oscilloscope	MSO70604 [3], DPO/MSO70604C (Gen1 testing only) MSO70804[3], DPO/MSO70804C (Gen1 and Gen2 testing only) MSO71254[3], DPO/MSO71254C (Gen1, Gen2, and Gen3 testing) MSO71604[3], DPO/MSO71604C (Gen1, Gen2, and Gen3 testing) MSO72004[3], DPO/MSO72004C (Gen1, Gen2, and Gen3 testing) DPO/DSA72504D (Gen 1, Gen2, and Gen 3 testing) DPO/DSA73304D (Gen 1, Gen 2, and Gen 3 testing) DPO/MSO72304DX (Gen1, Gen2, and Gen3 testing) DPO/MSO72504DX (Gen1, Gen2 and Gen3 testing) DPO/MSO73304DX (Gen1, Gen2 and Gen3 testing) DPO72304SX (Gen1, Gen2 and Gen3 testing) DPO75002SX [Standalone [4] or 2 stack] (Gen1, Gen2 and Gen3 testing) DPO73304SX (Gen1, Gen2 and Gen3 testing) DPO75902SX [Standalone[4] or 2 stack] (Gen1, Gen2 and Gen3 testing) DPO77002SX [Standalone[4] or 2 stack] (Gen1, Gen2 and Gen3 testing)
Arbitrary Function Generator (AFG) [5] (for automatic test pattern toggling)	Tektronix AFG3252, AFG3252C
Arbitrary Waveform Generator (AWG) (for automatic test pattern toggling)	■ Tektronix AWG5002B/C, AWG5012B/C, AWG5014B/C ■ Tektronix AWG7082B/C, AWG7122B/C ■ Tektronix AWG70001A, AWG70002A
Probes	P7313, P7513, P7513A, P7516, P7520A (with P75PST tip), P7520A, P7633, P7720, P7716, P7713

[3] Requires Microsoft Windows 7 (64-bit) operating system. Contact your local Tektronix Customer Service representative for upgrade information.

[4] Standalone can be used only for CEM specification Add-In-Card device type or U.2(SFF8639) specification Module device type

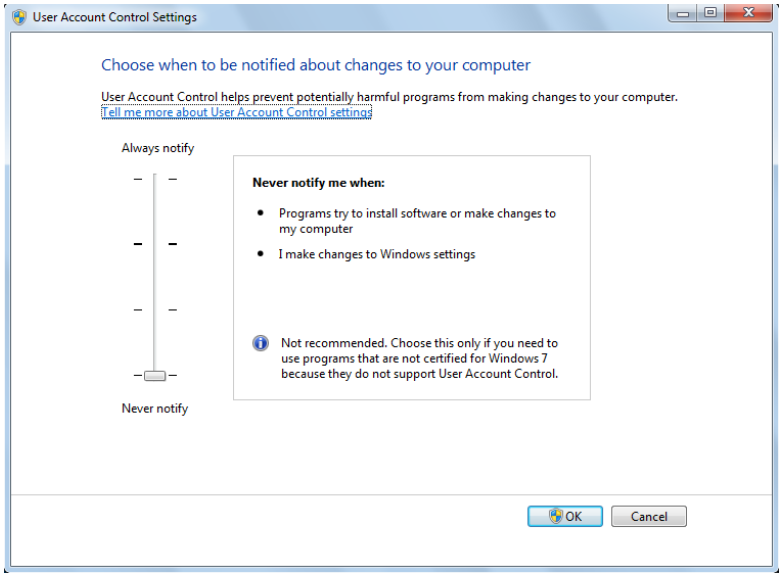
[5] The listed AFG/AWG instruments support both differential inputs (requires 2 channels) and 100 MHz burst mode.

Instrument/Accessory	Model number
RF Switch [6] [7]	■ Keithley System S46 and S46T RF Microwave Switch Systems for x12 PCIe ■ Gigatronics RF Switch 26GHz (8902-L-48TS26) for x16 PCIe
Other devices	■ SMP-SMA cables ■ TCA-SMA connectors ■ Matched pair cables ■ Tektronix P7313 SMA Differential Probe

Windows 7 user account settings

Windows 7 instruments need to have the User Account Control Settings set to **Never Notify**. To set User Account Control Settings:

1. Go to **Control Panel > User Accounts > Change User Account Control settings**.
2. Set it to **Never Notify** as shown in the image.



[6] Use GPIB cable or USB-to-GPIB cable to connect the oscilloscope to switch.
[7] The GPIB address of the RF Switch and toggle instrument (AWG/AFG) must be different.

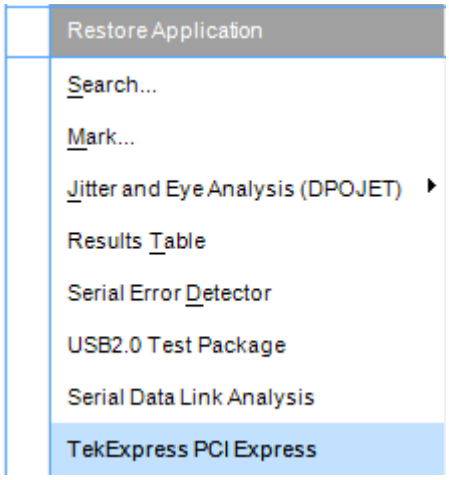
See also *Supported oscilloscopes*

Install the software

Use the following steps to install PCI Express software on any compatible instrument running Microsoft Windows 7 (64-bit). See *Minimum System Requirements* for details.

1. Close all applications (including the TekScope application).
2. Go to the www.tek.com Web site and search for TekExpress PCI Express to locate the installation file. Download the file TekExpress_PCIe_Deployment_Package.exe.
3. Copy or download the PCIe installer file to the oscilloscope.
4. Double-click the installer .exe file to extract the installation files and launch the InstallShield Wizard. Follow the on-screen instructions. The software installs in the following location:

   C:\Program Files (x86)\Tektronix\TekExpress\TekExpress PCI Express
5. The installer updates the TekScope Analyze menu to include the installed options.



See also

Minimum system requirements

Supported oscilloscopes

Set application file permissions

Before you run tests for the first time, do the following:

1. Understand where your test files are stored on the instrument.

After you install and launch TekExpress PCIe, it creates the following folders on the oscilloscope:

- \My Documents\My TekExpress\PCI Express
- \My Documents\My TekExpress\PCI Express\Untitled Session

Every time you launch TekExpress PCIe, an Untitled Session folder is created in the PCIe folder. The Untitled Session folder is automatically deleted when you exit the PCIe application. To preserve your test session files, save the test setup before exiting the TekExpress application.

CAUTION.

Do not modify any of the session files or folders because this may result in loss of data or corrupted session files. Each session has multiple files associated with it. When you save a session, a .TekX file, and a folder named for the session that contains associated files, is created on the oscilloscope X: drive.

2. *Map the shared My TekExpress folder* as **X:** (X drive) on the instruments used in test setups running Microsoft Windows Operating System.

   The My TekExpress folder has the share name format <domain><user ID>My TekExpress. Or, if the instrument is not connected to a domain, the share name format is <instrument name><user ID>My TekExpress. This shared folder is used to save the waveform files and is used during other file transfer operations.

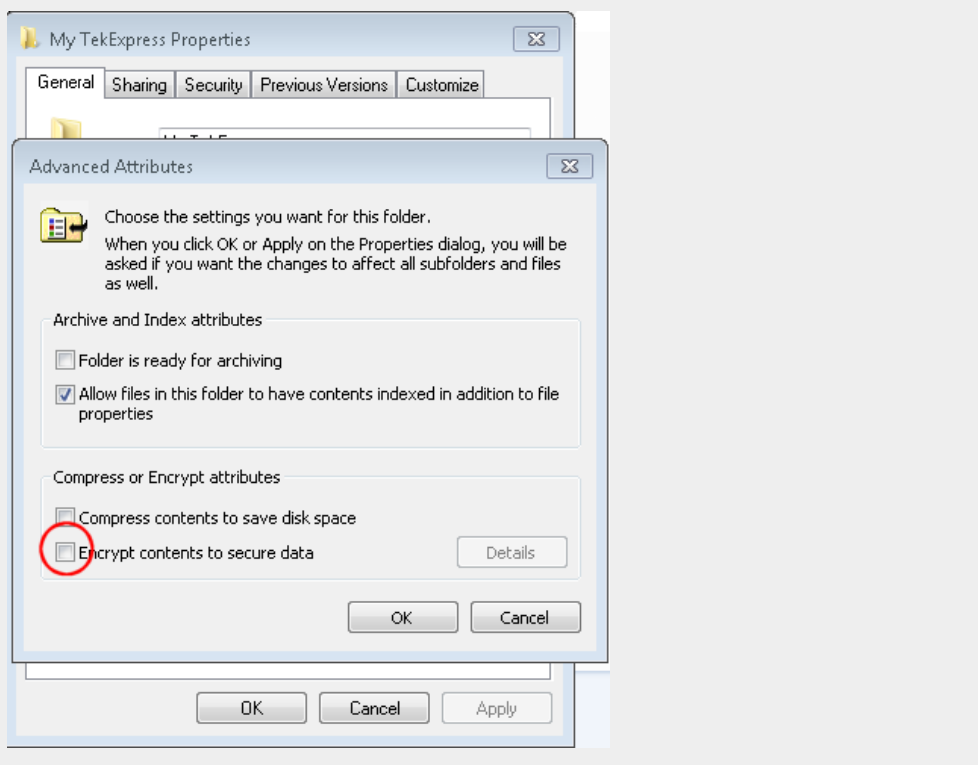
   NOTE. *If the X: drive is mapped to any other shared folder, the application will display a warning message asking you to disconnect the X: drive manually.*

3. Make sure that the My TekExpress folder (Drive X:) has read and write access:

   a. Right-click the folder and select **Properties**.

   b. Select the **General** tab and then click **Advanced**.

   c. In the Advanced Attributes dialog box, make sure that the option **Encrypt contents to secure data** is NOT selected (not checked). Example.



4. See the *prerun checklist* before you run a test.

See also

Configuration test parameters

View test-related files

Application directories and usage

File name extensions

Activate the license

Activate the license using the Option Installation wizard on the oscilloscope. Instructions for using the Options Installation window to activate licenses for installed applications is provided in the oscilloscope online Help:

1. From the oscilloscope menu bar, click **Utilities** > **Option Installation**.

   The TekScope Option Installation wizard opens.

2. Push the **F1** key on the oscilloscope keyboard to open the Option Installation help topic. Follow the directions in the topic to activate the license.

See also

View version and license information

View software version and license information

Use the following instructions to view version information for the application and for the application modules such as the Programmatic Interface and the Programmatic Interface Client.

To view version information for PCIe:

1. In the PCIe application, click the **Options** button and select **About TekExpress**.

   The About Tektronix TekExpress PCI Express dialog box appears, showing the version details.

To view license and option key information:

1. From the TekScope menu, select **Help > About TekScope**.
2. Scroll through the Options section list to locate PCI Express.
3. To view the Option key, look below the **Options** list.

See also

Activate the license

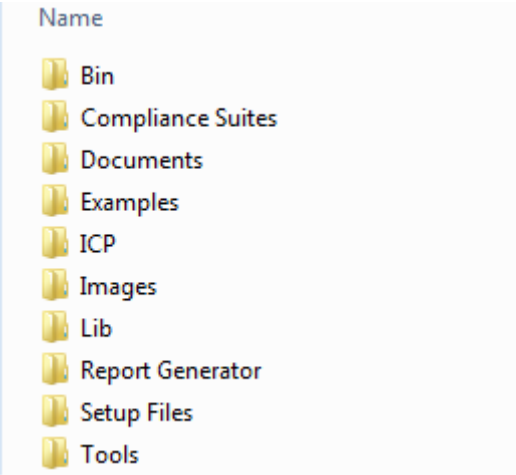
Options menu

Application directories and their contents

TekExpress PCIe application

The TekExpress PCIe application files are installed at *C:\Program Files (x86)\Tektronix\TekExpress\TekExpress PCI Express*.

It contains the following folders:



The following table lists the application directory names and their purpose.

Table 4: Application directories and usage

Directory names	Usage
ACP	Contains instrument and PCIe application-specific interface libraries
Bin	Contains miscellaneous PCIe application libraries
Compliance Suites	Contains compliance-specific files
Data Manager	Contains result management-specific libraries of the PCIe application
Data Storage	Contains libraries needed for storing data
Documents	Contains the technical documentation for the PCIe application
Examples	Contains various support files and example Python and C# test files
ICP	Contains instrument and PCIe application-specific interface libraries
Lib	Contains utility files specific to the PCIe application and Switch Matrix filter files

Directory names	Usage
Report Generator	Contains style sheets for report generation
Setup files	Contains PCI-SIG work shop compliant TekExpress setup files for System-Bard, Add-In-Card of CEM specification and Host, Module of U.2(SFF8639) specification
Tools	Contains instrument and PCIe application-specific files

TekExpress TekSig

The TekExpress TekSig application files are installed at *C:\Program Files (x86)\Tektronix\TekSigPackage*.

See also

View test-related files

File name extensions

File name extensions

The TekExpress PCIe application uses the following file name extensions:

File name extension	Description
.TekX	Application session files (the extensions may not be displayed)
.py	Python test file. See the TekExpress PCI Express\Examples folder for a sample file
.xml	Test-specific configuration information (encrypted) file Application log file
.wfm	Test waveform file
.mht	Test result reports (default). Test reports can also be *saved in HTML format*
.flt	Filter files
.chm, pdf	Help manuals

See also

Select report options

View test-related files

Application directories and their contents

Before you click start

Operating basics

Run the application

To launch the PCIe application, do either of the following:

- Select **Analyze > TekExpress PCI Express** from the TekScope menu.
- Double-click any saved PCIe session file (<file name>.TekX).

When you first run the application after installation, the application checks for a file called Resources.xml located at C:\Users\<username>\My TekExpress\PCI Express. The Resources.xml file gets mapped to the X: drive when the application launches. Session files are then stored inside the X:\PCI Express folder.

The Resources.xml file contains information about available network-connected instruments. If this file is not found, the application runs an instrument discovery program, before launching PCIe, to locate available instruments.

To keep the application window on top, select **Keep On Top** from the PCIe *Options menu*. If the application goes behind the oscilloscope application, click **Analyze > TekExpress PCI Express** to move the application to be in front.

See also

Activate the license

Exit the application

Use the following method to exit the application:

NOTE. *Using other methods to exit the application results in abnormal termination of the application.*

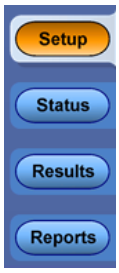
1. Click  on the application title bar.
2. Do one of the following:
   - If you have an unsaved session or test setup, you are asked to save it before exiting. To save it, click **Yes**. Otherwise click **No**. The application closes.
   - A message box appears asking if you really want to exit TekExpress. To exit, click **Yes**.

NOTE. *To minimise the application, click*  *on the application title bar.*

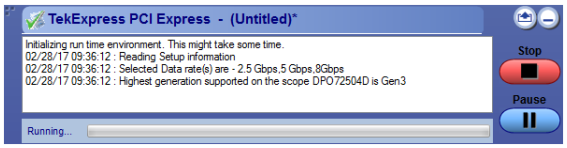
Application controls and menus

Application controls

Table 5: Application controls descriptions

Item	Description
Options menu 	Menu to display global application controls
Panel buttons 	Controls that open panels for configuring test settings and options.

Item	Description
Start/Stop button   	Use the Start button to start the test run of the measurements in the selected order. If prior acquired measurements have not been cleared, the new measurements are added to the existing set. The button toggles to the Stop mode while tests are running. Use the Stop button to abort the test.
Pause \ Continue button   	Use the Pause button to temporarily interrupt the current acquisition. When a test is paused, the button name changes to “Continue.”
Clear button  	Use the Clear button to clear all existing measurement results. Adding or deleting a measurement, or changing a configuration parameter of an existing measurement, also clears measurements. This is to prevent the accumulation of measurement statistics or sets of statistics that are not coherent. This button is available only on the *Results pane*.
Minimize button 	Minimizes the application.
Close button 	Exits the application.

Item	Description
Application window move	Place the cursor over the application window and drag it to the desired location.
Mini view 	Toggles the application to mini view which displays all the run messages with the time stamp, progress bar, Start / Stop button, and Pause / Continue button. The application will navigate to mini view when you start the test run of the measurements. 

Options menu

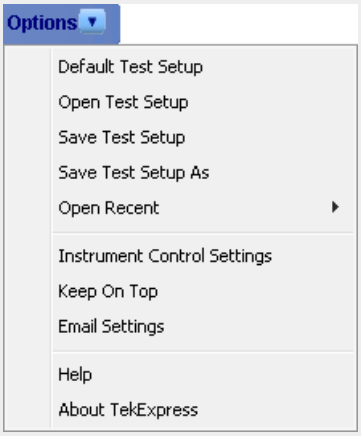
Options menu overview. The Options menu is located in the upper right corner of the application.

The Options menu has the following selections:

Menu	Function
Default Test Setup	Opens an untitled test setup with defaults selected
Open Test Setup	Opens a saved test setup
Save Test Setup	Saves the current test setup selections [1]
Save Test Setup As	Creates a new test setup based on an existing one [1]
Open Recent	Displays a menu of recently opened test setups to select from
Instrument Control Settings	Shows the list of instruments connected to the test setup and allows you to locate and refresh connections to those instruments
Keep On Top	Keeps the TekExpress PCIe utility on top of other open windows on the desktop
Email Settings	Use to configure email options for test run and results notifications
Help	Displays the TekExpress PCIe Online help
About TekExpress	■ Displays application details such as software name, version number, and copyright ■ Provides access to *license information* for your PCIe installation ■ Provides a link to the Tektronix Web site

[1] In pre-recorded mode, waveform recall will not be successful if the session name is lengthy, i.e. more than 10 characters.

Options menu

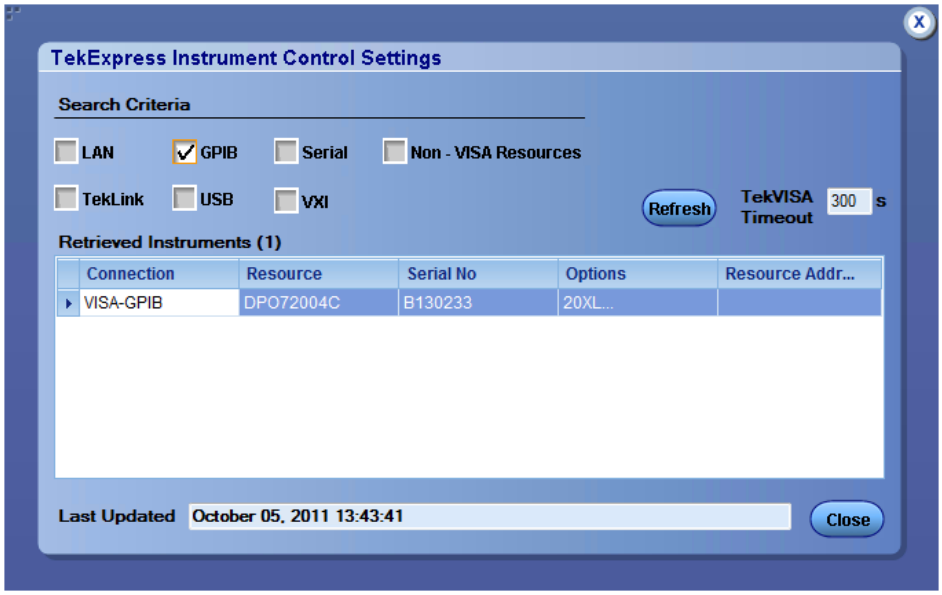


See also.

Application controls

Instrument control settings dialog box. Use the TekExpress Instrument Control Settings dialog box to search for and list the connected resources (instruments) found on specified connections (LAN, GPIB, USB, and so on) and each instruments connection information. You access this dialog box from the Options menu.

Access this dialog box from the **Options** menu.



Use the Instrument Control Settings feature to *search for connected instruments* and view instrument connection details. Connected instruments displayed here can be selected for use under Global Settings in the test configuration section.

See also.

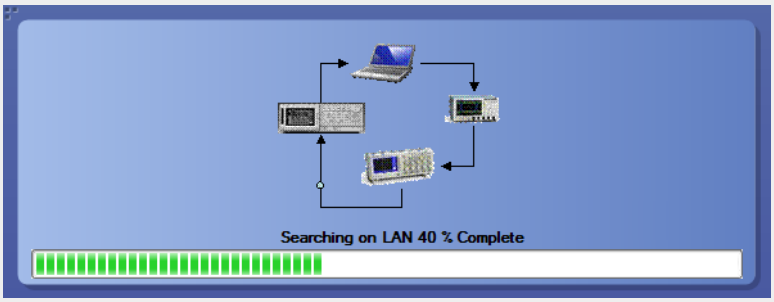
Options menu overview

View connected instruments. Use the Instrument Control Settings dialog box to view or search for connected instruments required for the tests. The application uses TekVISA to discover the connected instruments.

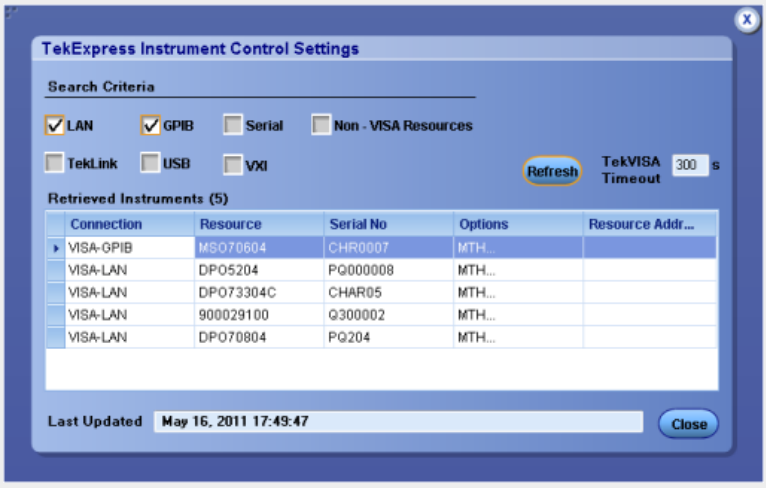
To refresh the list of connected instruments:

1. From the Options menu, select **Instrument Control Settings**.
2. In the Search Criteria section of the Instrument Control Settings dialog box, select the connection types of the instruments for which to search.

   Instrument search is based on the VISA layer, but different connections determine the resource type, such as LAN, GPIB, and USB. For example, if you choose LAN, the search will include all the instruments supported by TekExpress that are communicating over the LAN. If the search does not find any instruments that match a selected resource type, a message appears telling you that no such instruments were found.
3. Click **Refresh**. TekExpress searches for connected instruments.



4. After discovery, the dialog box lists the instrument-related details based on the search criteria you selected. For example, if you selected LAN and GPIB as the search criteria, the application checks for the availability of instruments over LAN, then GPIB.



The details of the instruments are displayed in the Retrieved Instruments table. The time and date of instrument refresh is displayed in the Last Updated field.

See also.

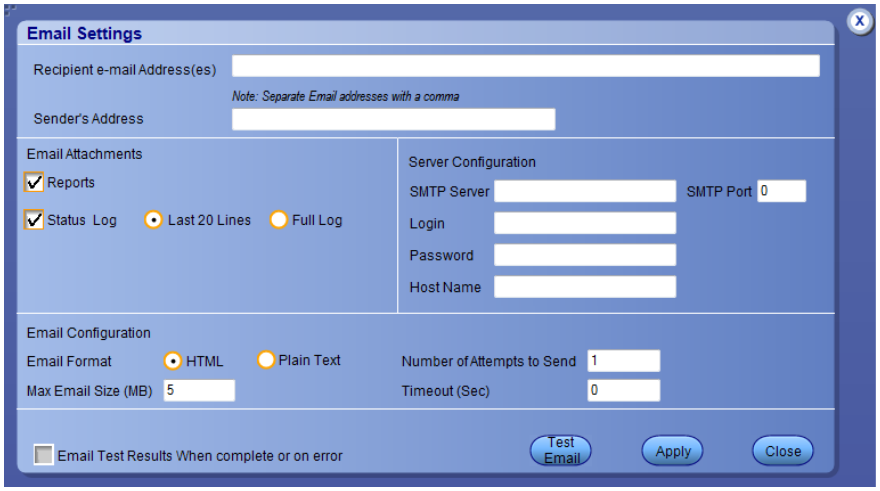
Configuration test parameters

Equipment connection setup

Email settings. Use the Email Settings utility to *configure email notifications* to receive notifications when a test completes, produces an error, or fails. Select the type of test session information to include with the email, such as test reports and test logs, the email message format, and the email message size limit.

Access this dialog box from the Options menu.

NOTE. *Recipient email address, sender's address, and SMTP Server are mandatory fields.*



See also.

Configure email settings

Options menu

Select test notification preferences

Configure email settings. To be notified by email when a test completes, fails, or produces an error, configure the email settings.

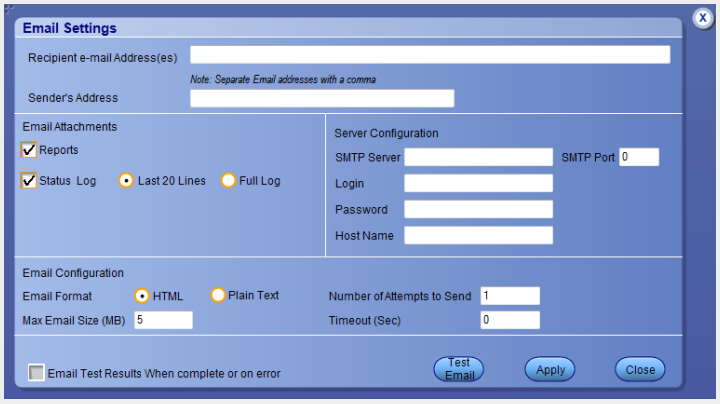
1. **Options > Email Settings** to open the Email Settings dialog box.
2. (Required) For Recipient email Address(es), enter one or more email addresses to which to send the test notification. To include multiple addresses, separate the addresses with commas.
3. (Required) For Sender's Address, enter the email address used by the instrument. This address consists of the instrument name followed by an underscore followed by the instrument serial number, then the @ symbol and the email server used. For example: DPO72016C_B130099@yourcompany.com.
4. (Required) In the Server Configuration section, type the SMTP Server address of the Mail server configured at the client location, and the SMTP Port number, in the corresponding fields.

   If this server requires password authentication, enter a valid login name, password, and host name in the corresponding fields.

   NOTE. *If any of the above required fields are left blank, the settings will not be saved and email notifications will not be sent.*

5. In the Email Attachments section, select from the following options:
   - **Reports**: Select to receive the test report with the notification email.
   - **Status Log**: Select to receive the test status log with the notification email. If you select this option, then also select whether you want to receive the full log or just the last 20 lines.
6. In the Email Configuration section:
   - Select the message file format to send: HTML (the default) or plain text.
   - Enter a maximum file size for the email message. Messages with attachments larger than this limit will not be sent. The default is 5 MB.
   - Enter the number in the Number of Attempts to Send field, to limit the number of attempts that the system makes to send a notification. The default is 1. You can also specify a timeout period.
7. Select the **Email Test Results When complete or on error** check box. Use this check box to quickly enable or disable email notifications.
8. To test your email settings, click **Test Email**.
9. To apply your settings, click **Apply**.
10. Click **Close** when finished.

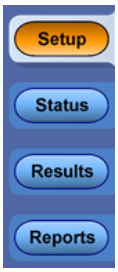
Email settings



Application panels

Application panel overview

Panels group related configuration, test selection, and results settings. Click a button to open the associated panel.



A panel may have one or more tabs that list the selections available in that panel. Controls in a panel can change depending on settings made in that panel or another panel.

The TekExpress PCIe panels are:

Table 6: Application panels

Panel Name	Purpose
Setup	The Setup panel shows the test setup controls. Click the **Setup** button to open this panel. Use this panel to: ■ *Select DUTparameters*. ■ *Select the test(s)*. ■ *Set acquisitions parameters* for selected tests. ■ *Configuration test parameters* ■ *Select test notification preferences*.
Status	View the progress and analysis status of the selected tests, and view test logs.
Results	View a summary of test results and select result viewing preferences.
Reports	Browse for reports, save reports as specific file types, specify report naming conventions, select report content to include (such as summary information, detailed information, user comments, setup configuration, application configuration), and select report viewing options.

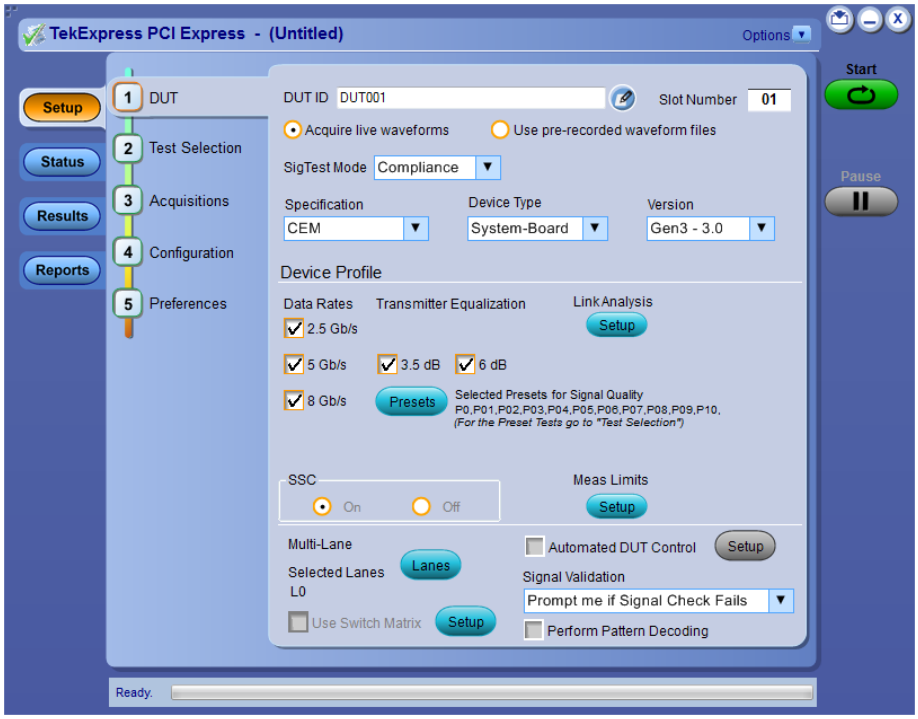
See also.

Application controls

About setting up test

Setup panel

Setup panel overview. The Setup panel contains sequentially ordered tabs that help guide you through a typical test setup process.



Use the tabs on this panel to:

Set the DUT parameters

Select test(s)

Select acquisition parameters

Set configuration parameters

Select test notification preferences

Set DUT parameters. Use the DUT tab to select parameters for the device under test. The settings are global and apply to all tests for the current session. DUT settings also affect the list of available tests in the Test Selection tab.

Click **Setup > DUT** to access the DUT parameters.

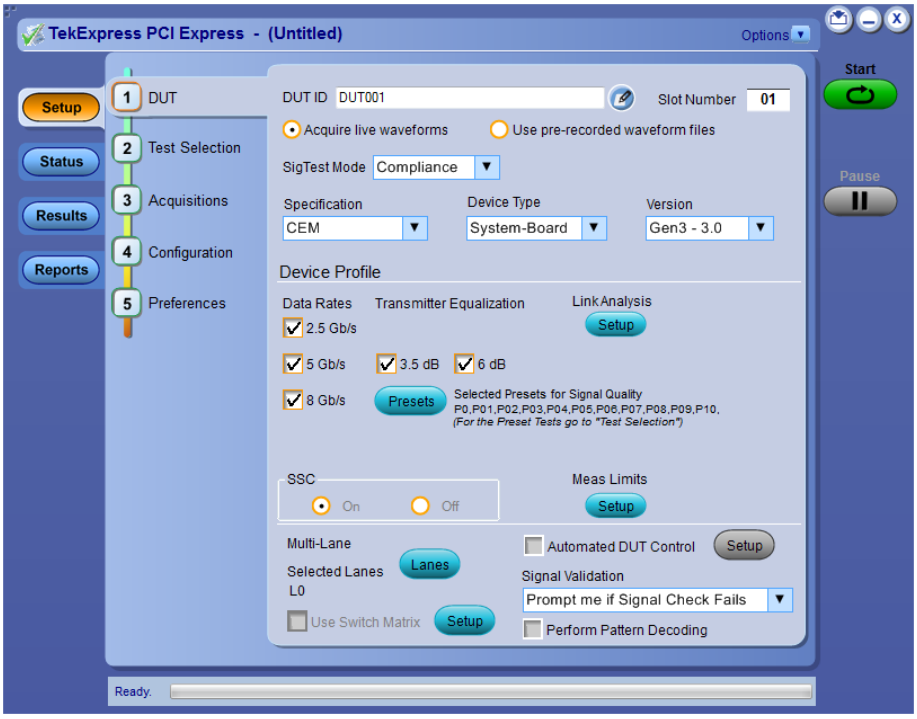


Table 7: DUT tab settings

Setting	Description
DUT ID	Adds an optional text label for the DUT to reports. The default value is DUT001. [1]
Slot Number	The slot parameter (1, 2, 4, 8,16, or 32) of the DUT.
Comments icon (to the right of the DUT ID field)	Opens a Comments dialog box in which to enter optional text to add to a report. The maximum number of characters is 256. To enable or disable comments appearing on the test report, see *Select report options*.)
Acquire live waveforms	Acquire active signals from the DUT for testing.

[1] In pre-recorded mode, waveform recall will not be successful if the session name is lengthy, i.e. more than 10 characters.

Setting	Description
Use prerecorded waveform files	Run tests on a saved waveform. *Open (load) a saved test setup*
SigTest Mode	Sets the overall testing mode. Select Compliance or User Defined: ■ Compliance: Preselects tests and parameters to meet compliance specifications for the selected version, specification, and device type. ■ User Defined: Enables the user to select specific tests and set custom parameters for tests.
Specification	PCIe supports the CEM, BaseSpec, and U.2 (SFF-8639) specification. ***NOTE.*** *U.2 (SFF8639) and Base Specification supports Gen3 (3.0) version only.*
Device Type	Sets the DUT device type (System-Board or Add-in-Card). Sets the DUT device type. The device type available depends on the selected specification. For CEM, the Device Type is System-Board or Add-In-Card For BaseSpec, the Device Type is TX Test Board For U.2 (SFF-8639), the Device Type is Module or Host
Version	Sets the DUT generation version. Available versions are Gen 1 (1.0a and 1.1), Gen2 (2.0), and Gen3 (3.0)
Device Profile	
Data Rates	Sets the data rates to test (2.5 Gb/s, 5 Gb/s, and 8 Gb/s). The data rates available depend on the selected DUT version.
Transmitter Equalization	Sets transmitter preemphasis levels. Available for Gen 2 and Gen 3 devices. The application selects both preemphasis levels by default when in the compliance mode for an Add-in-Card. At least one preemphasis level must be selected.
Link Analysis	Opens the Link Analysis dialog box to select custom filter files with which to perform link analysis on the source waveforms. *Link Analysis dialog box*

Setting		Description
Presets		Opens the Presets dialog box to select the presets (P0-P10) used to perform the signal quality tests. Only available for Gen 3 DUT version.
Meas Limits	Voltage Swing Limits	Sets the lane/link transmitter p-p voltage swing. This affects the limits applied to certain measurements based on the settings and does not change anything on the DUT tab.
	Crosstalk Limits	Sets specific eye test limits depending on if the DUT design uses interleaved or non interleaved routing. This affects the limits applied to certain measurements based on the settings and does not change anything on the DUT tab. This is applicable for Gen2. ■ When the DUT uses noninterleaved routing, select **Crosstalk (noninterleaved routing)**. ■ When the DUT uses interleaved routing, select **No Crosstalk (interleaved routing)**.
SSC (spread spectrum clocking)		Enables or disables SSC clocking. This affects the limits applied to certain measurements based on the settings and does not change anything on the DUT tab.
Lanes		Opens the Test Lane Setup dialog box to select the lanes to test. Lanes required for compliance testing are colored orange. At least one lane must be selected. The Link Width setting determines the number of lanes that can be tested. *Test Lane Setup dialog box*
Automated DUT Control		Enables automatic toggling of the DUT into different test modes (generation/equalization). Requires the use of an AFG or AWG instrument. Click **Setup** to access the *Automated DUT Control dialog box*
Use Switch Matrix		Select to use the switch matrix. This solution allows you to map each of the several transmitter signals and forward the selected input either to another relay or to the oscilloscope channel. Click **Setup** to configure the switch matrix. Refer *Switch Matrix application* for more details on configurations. ***NOTE.*** *Keithley supports maximum of 12 lanes and Gigatronics supports maximum of 16 lanes.*

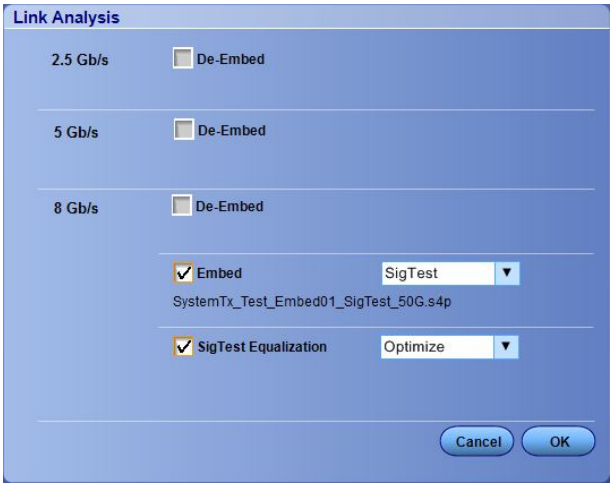
Setting	Description
Signal validation	Sets the application to validate acquisition signals and perform the specified action to take when acquired signals do not meet requirements. Select the action from the list.
Perform Pattern Decoding	Select to validate the pattern according to the respective lane and preset for Gen3.

See also.

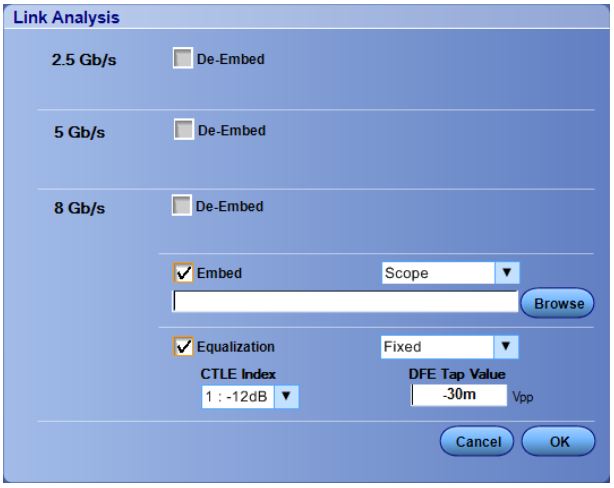
About setting up tests

Select a test

Link analysis dialog box. The Link Analysis dialog box lets you select custom filter files with which to perform link analysis on the source waveforms. The options available change with the each data rate. Selecting a mode enables a file browser with which to select the de-embed file.



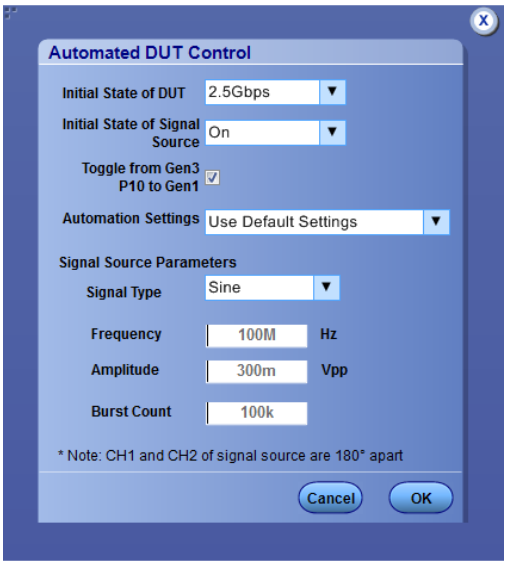
- For 2.5 Gb/s and 5 Gb/s data rates, only the De-embed option is available.
- There are two options for the 8 Gb/s data rate Embed mode: “Scope” and “SigTest.” This selection effects the fields shown in the Equalization selections.
- There are two options for the 8 Gb/s data rate equalization mode: “Optimize” and “Fixed”. “Fixed” mode provides fields with which to set the CTLE Index and DFE Tap Value parameters.



Test lane setup dialog box. The Test Lane Setup dialog box enables setting the link width and specific lanes to test. Lanes required for compliance testing are colored orange. At least one lane must be selected.

- The Link Width parameter sets the lanes that are tested.
- The lanes required for compliance testing are selected by default.
- Click **Select All** to select all the lanes.
- Click **Deselect All** to clear the selected lanes.
- Click **Compliance Lanes** to set all lanes required for compliance testing for the specified link width value.

Automated DUT control setup. The Automated DUT Control dialog box sets the parameters needed for automatic toggling of the DUT into different test modes (generation/equalization). DUT automation requires the use of a signal source (AFG or AWG).



Initial State of DUT: Sets the starting state of the DUT.

Initial State of Signal Source: Sets the AFG/AWG state to **On** (default) or **Off**. The On state enables the AFG/AWG output before the application starts signal acquisition. Some DUTs will toggle to the next signal state when the AFG/AWG initial state is On. Set the initial state to Off for these types of DUTs before running automated tests.

Toggle from Gen3 P10 to Gen1: Sets the DUT to toggle automatically to Gen1 after Gen3 P10 test execution.

Automation Settings: The Automation Settings values are Use Default Settings, Manually Configure Settings, and Custom Settings:

- **Use Default Settings**: The signal source parameters are set to predefined values as recommended by the test specification. The signal source parameter fields are disabled and cannot be edited.
- **Manually Configure Settings**: The signal source parameters are set directly at the AFG or AWG. The signal source parameter fields are disabled and cannot be edited. The PCIe application turns on or off the signal source without changing the settings.
- **Use Custom Settings**: The signal source parameters are set to the values specified in the Signal Source Parameters area. The signal source parameter fields are enabled.

Signal Type: Valid signal types are **Sine** and **Square**.

Frequency, **Amplitude**: Sets the AFG to output the specified frequency and amplitude values.

Burst Count: Sets the AFG to output the specified signal burst count.

NOTE. *Ch 1 and Ch 2 on the AFG source are set to 180° phase difference in all modes except Manually Configure Settings.*

NOTE. *Using DC Caps or Manual toggle, you can eliminate the automatic toggling issues that is due to DC offset.*

Select tests. Use the **Test Selection** tab to select the Signal Test and Preset Test(s) (for Gen3 only).

NOTE. *All tests are selected by default.*

1. Click **Setup > Test Selection**.
2. Select the test(s) to run:
   - Click + to expand a group of commands. Click the check box adjacent to a test group to select all tests in that group. Click check boxes adjacent to individual tests to select those tests.
   - Click **Deselect All** button to deselect all tests. All tests are selected by default.
   - Click **Select All** button to select all tests.
   - Click **Show MOI** button to open the MOI (Methods of Implementation) document for all measurements.
   - Click **Schematic** button to view a diagram that shows the correct DUT and equipment setup for the selected test. Use to verify your test equipment setup before running the test.
3. For Gen3 testing:
   - (Gen3 only) Click the **Preset Test** tab and select the presets tests.
   - (Gen3 only) Click the **Lanes** button in the Preset Test tab to view and select which lanes to use for preset testing. At least one lane must be selected.

See also.

Set acquisition parameters

About setting up tests

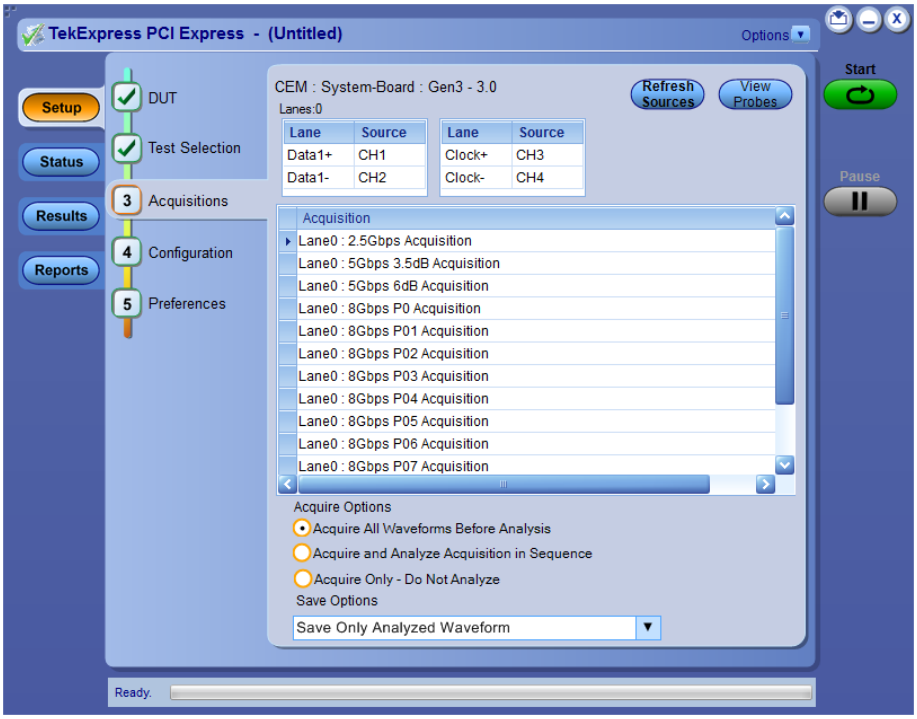
Acquisitions tab.

Set acquisition parameters. Use the **Acquisition** tab in the Setup panel to view and select test acquisition parameters, including the signal source channels, acquisition options, and waveform save options. This panel also shows the signal inputs required for the selected DUT parameters.

Contents displayed on this tab depend on whether you acquire active waveforms or use prerecorded waveform files (as set in the **DUT** tab. Contents displayed on this tab also depend on detected probes and the specified DUT type.

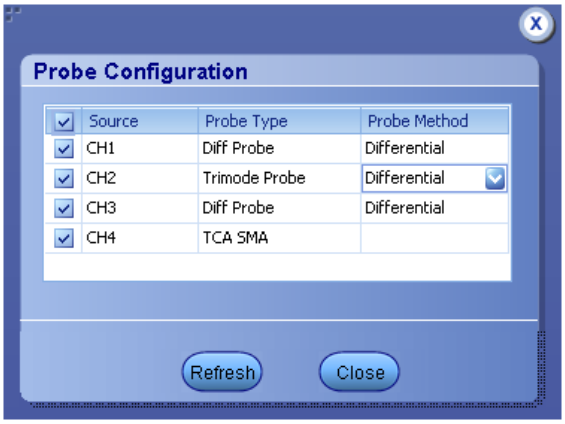
Active waveforms.

Figure 1: Acquisitions tab: using active waveforms



- Click the *Source* fields to select signal sources for the listed lanes. The number of lanes shown depends on the parameters set in the DUT tab.
- Click **Refresh Sources** to refresh the probe configuration after changing any probes. (This button performs the same function as the Refresh button in the Probe Configuration dialog box.)

- Click **View Probes** to view the detected probe configuration. Use the View Probes dialog box to enable or disable probe signal source access in the application.

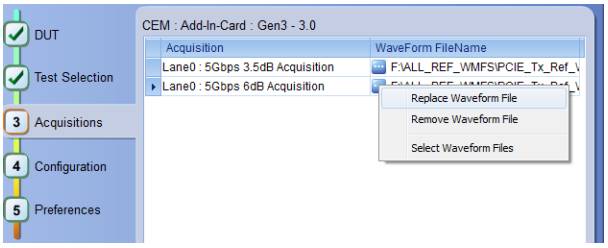


Only Differential option is available under Probe Method for Trimode probes.

- Click the *Acquire Options* controls to set how the application acquires and analyzes signals.
- Click the *Save Options* field to set how the application saves acquired waveforms (save all waveforms, save all waveforms after applying filters, or discard all waveforms after running analysis).

Prerecorded waveforms.

Figure 2: Acquisitions tab: using prerecorded waveforms



When using prerecorded waveform files, this panel lists available prerecorded waveform files. You can only select the source of the prerecorded waveform file for each test. See *Set acquisition waveform source for prerecorded waveform files*.

Set acquisition options. Select an **Acquire Option** to set the order in which waveforms are acquired and analyzed:

- **Acquire All Waveforms Before Analysis**: Acquire all waveforms required by tests before performing analysis. All required user interventions (such as connecting to different lanes) are completed, and waveforms acquired, before the analysis is run. You can turn off the DUT after the acquisitions are completed.
- **Acquire and Analyze Acquisition in Sequence**: Acquire waveforms and analyze for each test before proceeding to the next test. Use this setting to stop the testing when an error occurs, investigate and correct DUT, instrumentation connections, or application settings, then restart testing.
- **Acquire Only – Do Not Analyze**: Acquire all waveforms required by tests, and then stop (do not use waveforms to perform test analysis). Use this setting for testing multiple DUTs once the test and application settings are correct. Acquire all required waveforms and save the session for each DUT, and then recall the waveforms at a later point to analyze in *Prerecorded* mode.

See also.

Set acquisitions signal source

Set acquisition waveform save options

Set acquisition waveform save options. Select a **Save Option** to set how to save acquired test waveforms:

- **Save All the Waveforms**: Save all waveforms that were acquired for tests.
- **Save Only Analyzed Wfms**: Save waveforms that was used for analysis.
- **No Waveforms Saved – Discard after analysis**: Delete all acquired waveform data after analysis is complete.

Waveforms are saved to a folder that is unique to each session (a session starts when you click the Start button). The folder path is X:\PCI Express\Untitled Session\<dutid>\<date>_<time>. Images created for each analysis, CSV files with result values, reports, and other information specific to that particular execution are also saved in this folder. When the session is saved, content is moved to that session folder and the “Untitled Session” gets replaced by the session name.

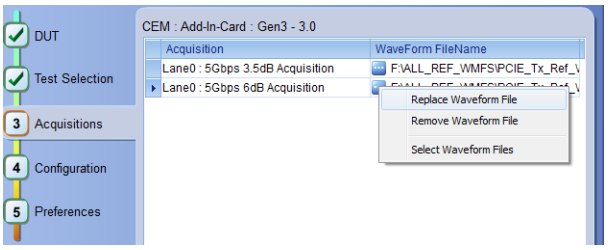
See also.

Set acquisitions signal source

Set acquisition waveform source for prerecorded waveform files

Set acquisition Acquire options

Set acquisition waveform source for prerecorded waveform files. When using prerecorded waveform files, there are no acquisition source selections to make. You can only select the source of the prerecorded waveform files for each test.



If you selected to use a prerecorded waveform file (in the DUT tab), the lane and source fields are not applicable and are not shown. The Acquisition tab instead shows a table of the waveforms used for the required test acquisitions.

You can load a different waveform file for each table item. To load a different waveform file:

1. Click the ellipsis button ( ) of the waveform file to change.
2. Select the waveform task to perform (replace, remove, or select the waveform file).
3. Use the dialog box to navigate to and select the waveform file with which to replace the current file. You need to select all required differential waveforms for analysis. For example, select one data waveform and one clock waveform for each acquisition (except 2.5 Gbps) for testing a system board.

NOTE. *Clock signals are not required for Gen1 (2.5 Gbps data rate) testing.*

See also.

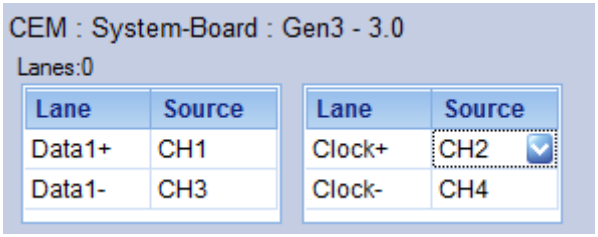
Set acquisition signal source

Set acquisition options

Set acquisition waveform save options

Set acquisition signal source. Use this procedure to set the channel sources for live waveform acquisitions. The number of Lane and Source fields shown depends on the number of lanes selected for testing in the **DUT** tab.

1. Click **Setup > Acquisitions**.
2. Click in the Source column of the field to change.
3. Click the arrow button to list available sources from which to select.



See also.

Set acquisition options

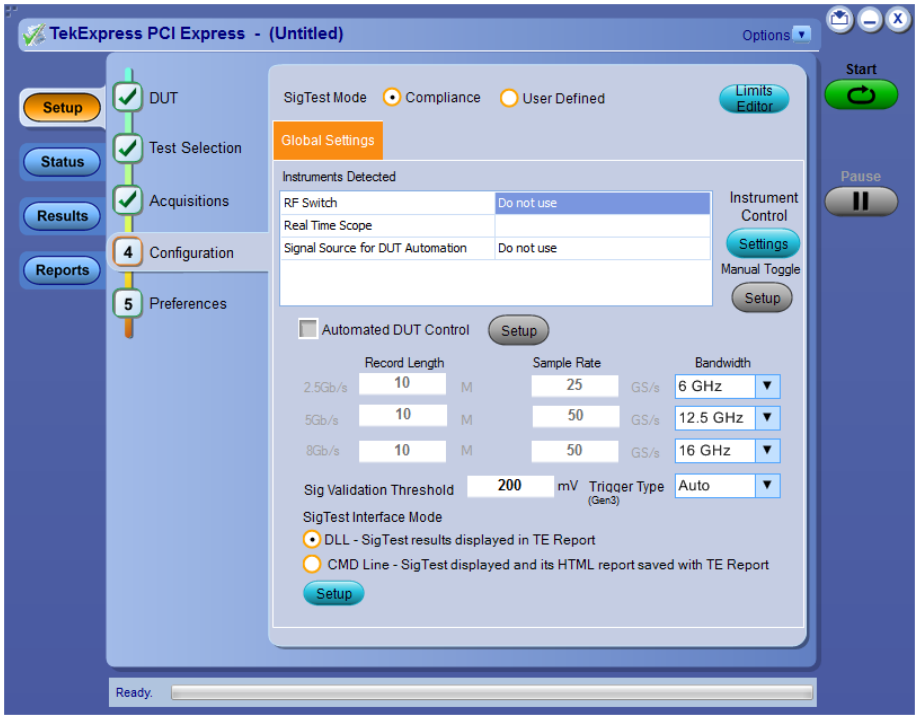
Set acquisition waveform save options

Set acquisition waveform source for prerecorded waveform files

Configure test parameters.

About configuring test parameters. Use the **Configuration** tab to view and set global and individual measurement parameters for the selected tests. Which fields are available to edit depends on the selected Sigtest mode (Compliance or User Defined) as set in this tab or the DUT tab.

NOTE. *You cannot change test parameters that are grayed out.*



See also.

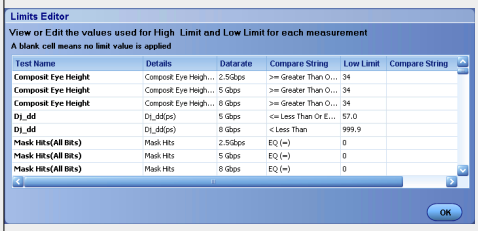
Configuration tab parameters

About setting up tests

About running tests

Configuration tab parameters. The following table lists the Configuration tab settings and parameters.

Table 8: Configuration tab parameters

Parameter	Description
SigTest Mode	Determines whether test parameters are in compliance or can be edited (User Defined Mode). ■ Compliance: Most test parameter values cannot be edited. ■ User Defined: Enables editing of most test parameters.
Limits Editor	Shows the upper and lower limits for the applicable measurement using different types of comparisons. In Compliance Mode, use the Limits Editor to view the measurement high and low limits used for selected tests. In User Defined Mode, use the Limits Editor to edit the limit settings.  To edit a value, click that field and either select from the displayed list or enter a new value. Use the bottom scroll bar to view all available fields.
Instruments Detected	Displays a list of the connected instruments found during the instrument discovery. Instrument types include equipment such as oscilloscopes and signal generators.
Instrument Control	Click **Settings** to *search for connected instruments* and view instrument connection details. Connected instruments displayed here can be selected for use under Global Settings in the test configuration section.
Manual Toggle	Click **Setup** to manually toggle AWG or AFG for PCIe presets. This is enabled when the Signal Source for DUT Automation in Instruments Detected is selected.
Automated DUT Control	Enables automatic toggling of test patterns for DUT tests. Requires an AFG instrument as part of the test setup. Click **Setup** to configure the DUT automation settings.

Parameter	Description
Record Length, Sample Rate, Bandwidth	These settings apply to all tests selected for the indicated data rate. ■ Record Length: Specifies the waveform record length. ■ Sample Rate: Specifies the oscilloscope sample rate to use for all tests. ■ Bandwidth: Specifies the oscilloscope bandwidth to use for all tests.
Sig Validation Threshold	Sets the threshold voltage to use for signal validation.
Trigger Type (Gen3)	■ Edge ■ Width ■ Auto
SigTest Interface Mode	Sets whether to use a Dynamic Link Library (DLL) or command line interface (CMD) for running Sigtest. ***NOTE.*** *DLL is applicable for CEM specification only* Click **Setup** to open the **SigTest Module Settings** dialog box, where you can specify which revision of PCI-SIG Sigtest library to use for running tests.

See also.

About acquisition

De-embed using filter files

Set test notification preferences. Use the Preferences tab to set the application action when a test measurement fails:

1. Click **Setup > Preferences**.
2. Select the measurement failure action:
   - Select **On Test Failure, pause the test and let me investigate** to pause the test when a failure occurs. Click the **Status** and **Results** buttons to explore the failure condition. To resume the test, click **Continue**.
   - Select **On Test Failure, stop and notify me of the failure** to stop the test and send an email when a test fails. Click **Email Settings** to verify that **Email Test Results when complete or on error** is selected, and to verify the address to which the email is sent.

Preferences tab parameters. Use the Preferences tab to set the application action when a test measurement fails, and how the application handles opo-up error, warning, and information messages during test sequences.

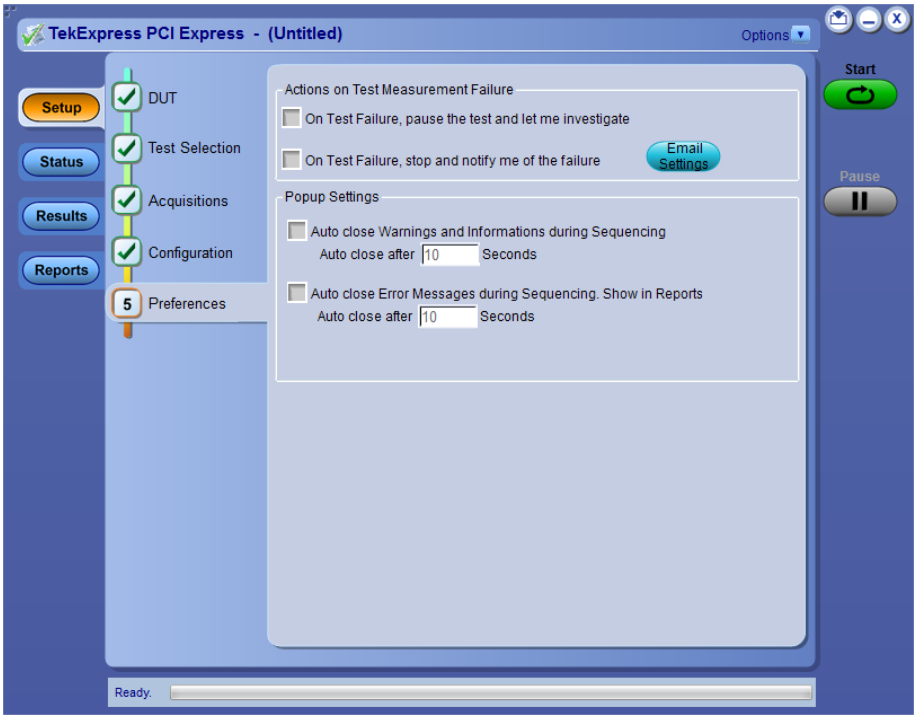


Table 9: Preferences tab parameters

Parameter	Description
Actions on Test Measurement Failure	
On Test Failure, stop and notify me of the failure	Stops the test sequence and sends an email when a test fails.
On Test Failure, pause the test and let me investigate	Pauses the test when a failure occurs. Click the **Status** and **Results** buttons to explore the failure condition. To resume the test, click **Continue**.
Email Settings button	Click **Email Settings** to open the Email Settings dialog box and verify that **Email Test Results when complete or on error** is selected, and verify the address to which the email is sent.
Popup Settings	
Auto close Warnings and Informations during Sequencing	Sets the time for how long the application displays Warning and Information pop-up messages before automatically closing the messages and continues testing by taking the default action.

Parameter	Description
Auto close Error Messages during Sequencing. Show in Reports	Sets the time for how long the application displays Error pop-up messages before automatically closing the messages and continuing with testing. Message content is added to the test report.

See also.

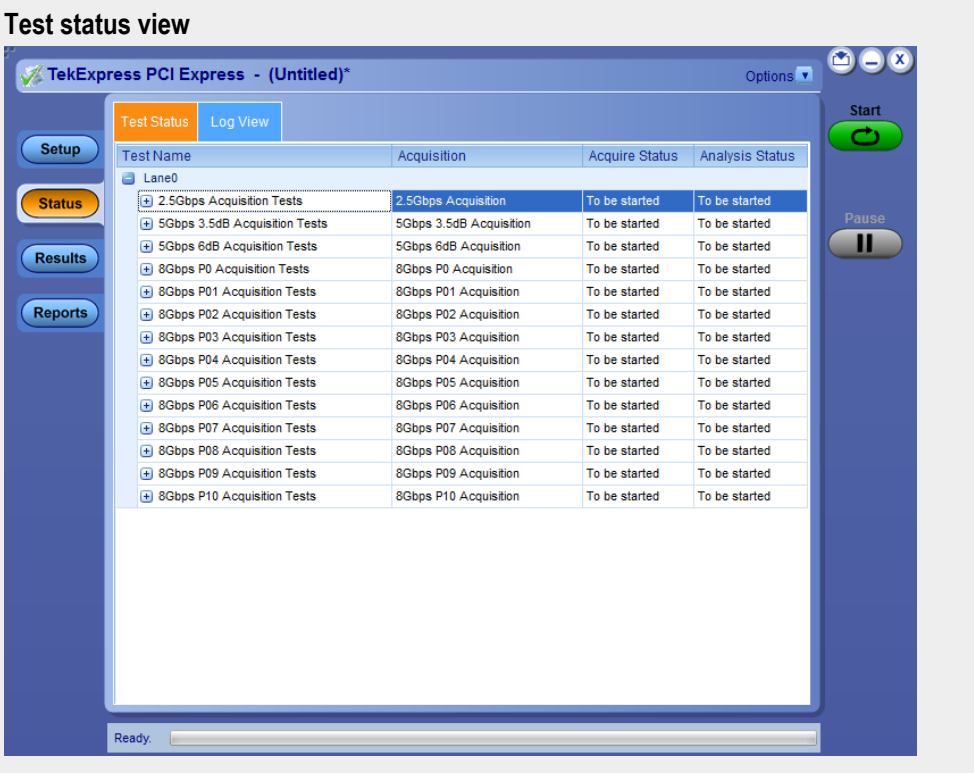
About setting up tests

Select report options

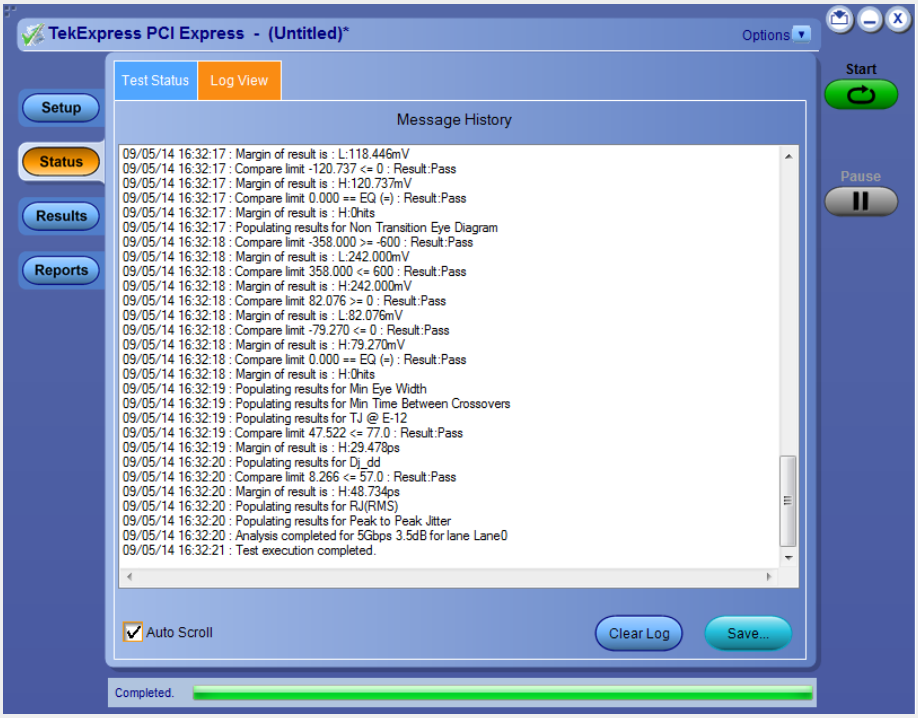
Status panel overview

The Status panel provides status on test acquisition and analysis (Test Status tab) and a listing of test tasks performed (Log View tab). The application opens the Test Status tab when you start a test run. You can select the Test Status or the Log View tab to view these items while tests are running.

Test status view



Log view



The Log View display has several viewing options:

- Message History: This window timestamps and displays all run messages.
- **Auto Scroll** button : Select this check box to have the program automatically scroll down as information is added to the log during the test.
- **Clear Log** button : Click this button to clear all messages from the display.
- **Save** button : Click this button to save the log file as a text file. A standard Save File window is displayed to name and save the file.

See also.

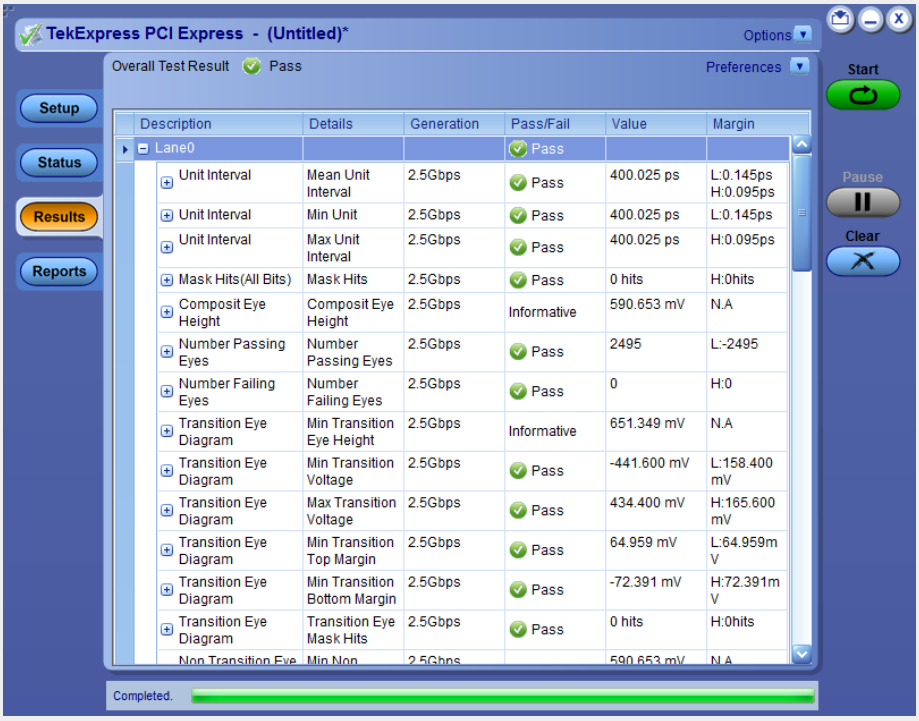
Application panel overview

Results panel

Results panel overview. When a test finishes, the application switches to the Results panel to display a summary of signal and preset test results. The Overall Test Result is displayed at the top left of the Results table. If all of the tests for the session pass, the overall test result is **Pass**. If one or more tests fail, the overall test result is **Fail**.

Set viewing preferences for this panel from the Preferences menu in the upper right corner. Viewing preferences include showing whether a test passed or failed, summary results or detailed results, and enabling wordwrap.

Results panel



Description	Details	Generation	Pass/Fail	Value	Margin
Lane0			Pass		
Unit Interval	Mean Unit Interval	2.5Gbps	Pass	400.025 ps	L:0.145ps H:0.095ps
Unit Interval	Min Unit	2.5Gbps	Pass	400.025 ps	L:0.145ps
Unit Interval	Max Unit Interval	2.5Gbps	Pass	400.025 ps	H:0.095ps
Mask Hits(All Bits)	Mask Hits	2.5Gbps	Pass	0 hits	H:0hits
Composit Eye Height	Composit Eye Height	2.5Gbps	Informative	590.653 mV	N.A
Number Passing Eyes	Number Passing Eyes	2.5Gbps	Pass	2495	L:-2495
Number Failing Eyes	Number Failing Eyes	2.5Gbps	Pass	0	H:0
Transition Eye Diagram	Min Transition Eye Height	2.5Gbps	Informative	651.349 mV	N.A
Transition Eye Diagram	Min Transition Voltage	2.5Gbps	Pass	-441.600 mV	L:158.400 mV
Transition Eye Diagram	Max Transition Voltage	2.5Gbps	Pass	434.400 mV	H:165.600 mV
Transition Eye Diagram	Min Transition Top Margin	2.5Gbps	Pass	64.959 mV	L:64.959m V
Transition Eye Diagram	Min Transition Bottom Margin	2.5Gbps	Pass	-72.391 mV	H:72.391m V
Transition Eye Diagram	Transition Eye Mask Hits	2.5Gbps	Pass	0 hits	H:0hits
Non Transition Eye	Min Non	2.5Gbps		590.653 mV	N.A

When a test finishes, the application switches to the Results panel, which displays a summary of test results.

NOTE. *NAN (Not A Number) is displayed in the test results if an invalid waveform was supplied for the test.*

Each test result occupies a row in the Results table. By default, results are displayed in summary format with the measurement details collapsed and with the Pass/Fail column visible. Change the view in the following ways:

- To expand all tests listed, select **View Results Details** from the Preferences menu in the upper right corner.
- To expand and collapse tests, click the plus and minus buttons.
- To collapse all expanded tests, select **Preferences > View Results Summary**.
- To remove or restore the Pass/Fail column, select **Preferences > Show Pass/Fail**.

- To enable or disable the wordwrap feature, select **Preferences > Enable Wordwrap**.
- To expand the width of a column, place the cursor over the vertical line that separates the column from the column to the right. When the cursor changes to a double-ended arrow, hold down the mouse button and drag the column to the desired width.
- To group and view the tests by Lane, Test, Equalization, Pass/Fail, use the Preferences option in Results Panel.
- To clear all test results displayed, click **Clear**.

See also.

View a report

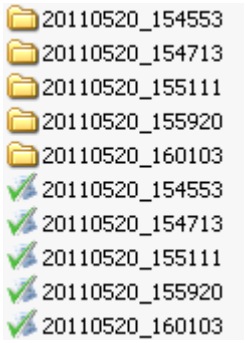
About panels

View test-related files. Files related to tests are stored in the My TekExpress\PCI Express folder. Each test setup in this folder has a test setup file and a test setup folder, both with the test setup name.

The test setup file is preceded by the TekExpress icon and usually has no visible file name extension.

Inside the test setup folder is another folder named for the DUT ID used in the test sessions. The default is DUT001.

Inside the DUT001 folder are the session folders and files. Each session also has a folder and file pair, both named for the test session using the naming convention (date)_(time). Each session file is stored outside its matching session folder:



Each session folder contains image files of any plots generated from running the test session. If you selected to save all waveforms or ran tests using prerecorded waveform files, these are included here.

The first time you run a new, unsaved session, the session files are stored in the Untitled Session folder located at ..\My TekExpress\PCI Express. When you name and save the session, the files are placed in a folder with the name that you specify. A copy of the test files stay in the Untitled Session folder until you run a new test or until you close the PCIe application.

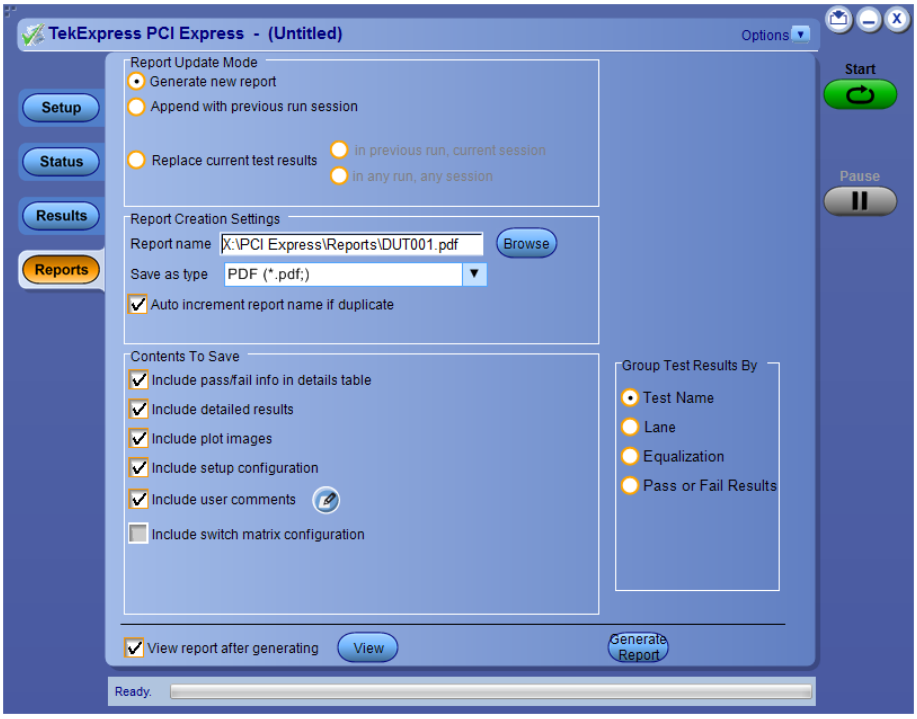
See also.

File name extensions

Before you click start

Reports panel

Reports panel overview. Use the Reports panel to browse for reports, name and save reports, select test content to include in reports, and select report viewing options.



For information on setting up reports, see *Select report options*. For information on viewing reports, see *View a Report*.

See also.

About panels

Select report options. Click the **Reports** button and use the Reports panel controls to select which test result information to include in the report, and the naming conventions to use for the report. For example, always give the report a unique name or select to have the same name increment each time you run a particular test.

Select report options before running a test or when creating and saving test setups. Report settings are included in saved test setups.

In the Reports panel, select from the following report options:

Table 10: Report options

Setting		Description
Report Update Mode		
Generate new report		Creates a new report.
Append with previous run session		Appends the latest test results to the end of the current test results report.
Replace current test results	In previous run, current session	Select to replace current test results in the report with the test result(s) of previous run in current session.
	In previous run, current session	Select to replace current test results in the report with the test result(s) in selected run session's report. Click  and select test result of any other run session from another setup.
Report Creation Settings		

Setting	Description
Report name	Displays the name and location from which to open a report. The default location is at *\My TekExpress\PCI Express\Reports*. The report file in this folder gets overwritten each time you run a test unless you specify a unique name or select to auto increment the report name, this option is selected by default. Change the report name or location. Do one of the following: ■ In the Report Path field, type over the current folder path and name. ■ Double-click in the Report Path field and then make selections from the popup keyboard and click the **Enter** button. Be sure to include the entire folder path, the file name, and the file extension. For example: C:\Documents and Settings\your user name\My Documents\My TekExpress\PCI Express\DUT001_Test_72.7.1.3.mht. Open an existing report. Click **Browse** button to locate and select the report file and then click **View** button at the bottom of the panel.
Save as type	Saves a report in the specified file type. Lists supported file types to choose from. ***NOTE.*** *If you select a file type different from the default, be sure to change the report file name extension in the Report Name field to match. By default, report is generated in PDF format but you can also generate the report in MHT format.*
Auto increment report name if duplicate	Sets the application to automatically increment the name of the report file if the application finds a file with the same name as the one being generated. For example: DUT001, DUT002, DUT003. This option is enabled by default. ***NOTE.*** *If you choose, say 'TID_345', for 'DUT ID', the next report for a new run in the same session will be named 'TID_346' which may contradict or cause confusion. In this case, it is suggested that you choose 'TID_345_001' for DUT ID, since the next runs in the same session will have report names generated with the last suffix auto-incremented as follows – 'TID_345_002' and so on.*

Setting	Description
Contents To Save	
Include pass/fail info in details table	Select to include the column labeled Test Result (indicating whether the test passed or failed) in the report. For details, see Report Contents in *View a report*.
Include detailed results	Sets the application to include parameter limits, execution time, and test-specific comments generated during the test.
Include plot images	Sets the application to include plotted diagrams such as Eye diagrams.
Include setup configuration	Sets the application to include hardware and software information in the summary box at the top of the report. Information includes: the oscilloscope model and serial number, probe model and serial number, the oscilloscope firmware version, SPC and factory calibration status, and software versions for applications used in the measurements.
Include user comments	Select to include any comments about the test that you or another user added in the DUT tab of the Setup panel. Comments appear in the Comments section, under the summary box at the beginning of each report.
Include switch matrix configuration	Select to include switch matrix configuration.
Group Test Results by	
Test Name	Select to display the test results by test name.
Lane	Select to display the test results by lane.
Equalization	Select to display the test results by equalization.
Pass or Fail Results	Select to display the test results by pass or fail results.
Other	
View Report After Generating	Automatically opens the report in a Web browser when the test completes. This option is selected by default.
View button	Click to view the most current report.
Generate Report button	Generates a new report based on the current analysis results.

See also.

View a report

About setting up tests

View a report. The application automatically generates a report when test analysis is completed and displays the report in your default Web browser (unless the **View Report After Generating** check box is not selected). If this check box is not cleared, or to view a different test report, do the following:

1. Click the **Browse** button and locate and select the report file to view.
2. In the Reports panel, click **View**.

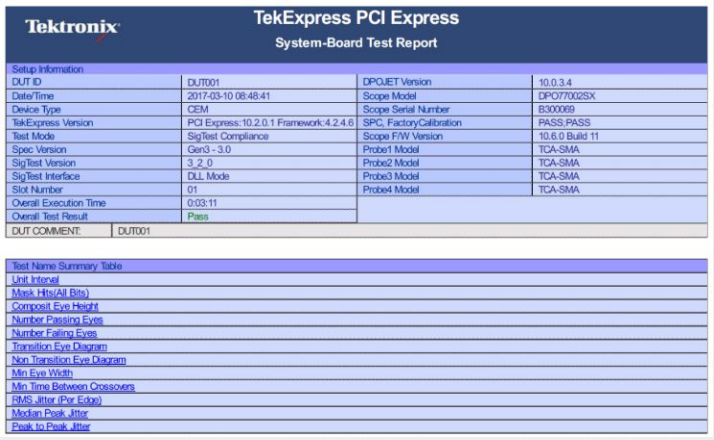
For information on changing the file type, file name, and other report options, see *Select report options*.

Report contents. A report shows specified test details, as defined in the Reports panel.

NOTE. *NAN (Not A Number) is displayed in the report contents if an invalid waveform was supplied for the test. In PDF generated report, hyperlinks do not work and result table for some measurements maybe spread across a page.*

Setup configuration information

Setup configuration information is listed in the summary box at the beginning of the report. This information includes the oscilloscope model and serial number, and software versions. To exclude this information from a report, clear the **Include Setup Configuration** check box in the Reports panel before running the test.



Tektronix

TekExpress PCI Express

System-Board Test Report

Setup Information

DUT ID	DUT001	DPOJET Version	10.0.3.4
Date/Time	2017-03-10 08:48:41	Scope Model	DPO77002SX
Device Type	CEM	Scope Serial Number	B300069
TekExpress Version	PCI Express:10.2.0.1 Framework:4.2.4.6	SPC, FactoryCalibration	PASS;PASS
Test Mode	SigTest Compliance	Scope F/W Version	10.6.0 Build 11
Spec Version	Gen3 - 3.0	Probe1 Model	TCA-SMA
SigTest Version	3_2_0	Probe2 Model	TCA-SMA
SigTest Interface	DLL Mode	Probe3 Model	TCA-SMA
Slot Number	01	Probe4 Model	TCA-SMA
Overall Execution Time	0:03:11		
Overall Test Result	Pass		
DUT COMMENT:	DUT001		

Test Name Summary Table

Unit Interval
Mask Hits(All Bits)
Composit Eye Height
Number Passing Eyes
Number Failing Eyes
Transition Eye Diagram
Non Transition Eye Diagram
Min Eye Width
Min Time Between Crossovers
RMS Jitter (Per Edge)
Median Peak Jitter
Peak to Peak Jitter

Test result summary

The Test Result column indicates whether a test passed or failed. If the test passed, the column cell is green. If the test failed, it is red. To exclude this information from a report, clear the **Include pass/fail info in details table** check box in the Reports panel before running the test.

Unit Interval

Measurement Details	Lane Name	Data Rate	Equalization	Measured Value	Test Result	Margin	Low Limit	High Limit
Mean Unit Interval	Lane0	2.5Gbps	-	399.999 ps	Pass	L:0.119ps H:2.121ps	399.88	402.12
Min Unit Interval	Lane0	2.5Gbps	-	399.998 ps	Pass	L:0.118ps	399.88	N.A
Max Unit Interval	Lane0	2.5Gbps	-	399.999 ps	Pass	H:2.121ps	N.A	402.12
COMMENTS	PCI Express Base Specification Revision 3.1 Table 4-18: Transmitter Specifications							

Back to Summary Table

Mask Hits(All Bits)

Measurement Details	Lane Name	Data Rate	Equalization	Measured Value	Test Result	Margin	Low Limit	High Limit
Mask Hits	Lane0	2.5Gbps	-	0 hits	Pass	N.A	0	0
COMMENTS								

Back to Summary Table

Composit Eye Height

Measurement Details	Lane Name	Data Rate	Equalization	Measured Value	Test Result	Margin	Low Limit	High Limit
Composit Eye Height	Lane0	2.5Gbps	-	808.831 mV	Informative	N.A	N.A	N.A
COMMENTS								

Back to Summary Table

See also.

Results panel overview

View test-related files

Setting up and configuring tests

About setting up tests

Set up tests using the tabs in the *Setup panel*. Settings in the DUT tab use a top-down, left-to-right logic flow, so that any parameter that affects or acts as a filter for other parameters appears either to the top of or to the left of the affected parameters.

Tests are saved when you save a test setup. To avoid overwriting test results, remember to assign a unique name to the test either before running it or immediately after.

See also

Test setup overview

Before you click start

About test setups

About running tests

Equipment connection setup

Click the **Setup > Test Selection > Schematic** button to open a PDF file that shows the compliance test setup diagrams (instrument, DUT, and cabling) for supported testing configurations.

See also

Minimum system requirements

View connected instruments

About setting up tests

Equipment connection setup through switch system

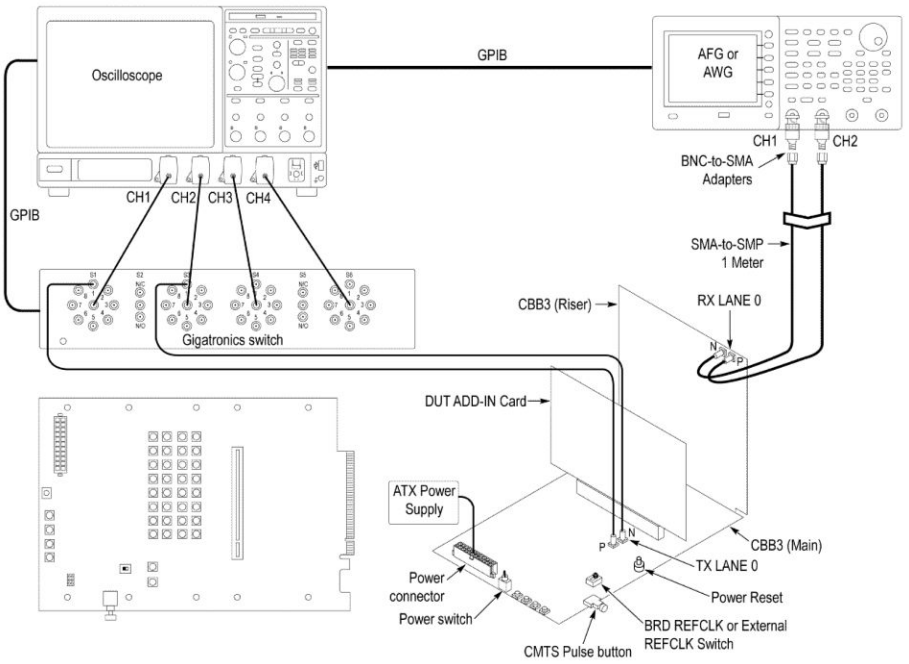


Figure 3: Add-In card connection setup through switch system

Gigatronics Switch: DUT Lane to Signal connection mapping			
TX LANE 0 P	S1 (relay) > Signal 1	TX LANE 0 N	S3 > Signal 1
TX LANE 1 P	S1 > Signal 2	TX LANE 1 N	S3 > Signal 2
TX LANE 2 P	S1 > Signal 3	TX LANE 2 N	S3 > Signal 3
TX LANE 3 P	S1 > Signal 4	TX LANE 3 N	S3 > Signal 4
TX LANE 4 P	S1 > Signal 5	TX LANE 4 N	S3 > Signal 5
TX LANE 5 P	S1 > Signal 6	TX LANE 5 N	S3 > Signal 6
TX LANE 6 P	S1 > Signal 7	TX LANE 6 N	S3 > Signal 7
TX LANE 7 P	S1 > Signal 8	TX LANE 7 N	S3 > Signal 8
TX LANE 8 P	S4 > Signal 1	TX LANE 8 N	S6 > Signal 1
TX LANE 9 P	S4 > Signal 2	TX LANE 9 N	S6 > Signal 2
TX LANE 10 P	S4 > Signal 3	TX LANE 10 N	S6 > Signal 3
TX LANE 11 P	S4 > Signal 4	TX LANE 11 N	S6 > Signal 4
TX LANE 12 P	S4 > Signal 5	TX LANE 12 N	S6 > Signal 5
TX LANE 13 P	S4 > Signal 6	TX LANE 13 N	S6 > Signal 6
TX LANE 14 P	S4 > Signal 7	TX LANE 14 N	S6 > Signal 7
TX LANE 15 P	S4 > Signal 8	TX LANE 15 N	S6 > Signal 8

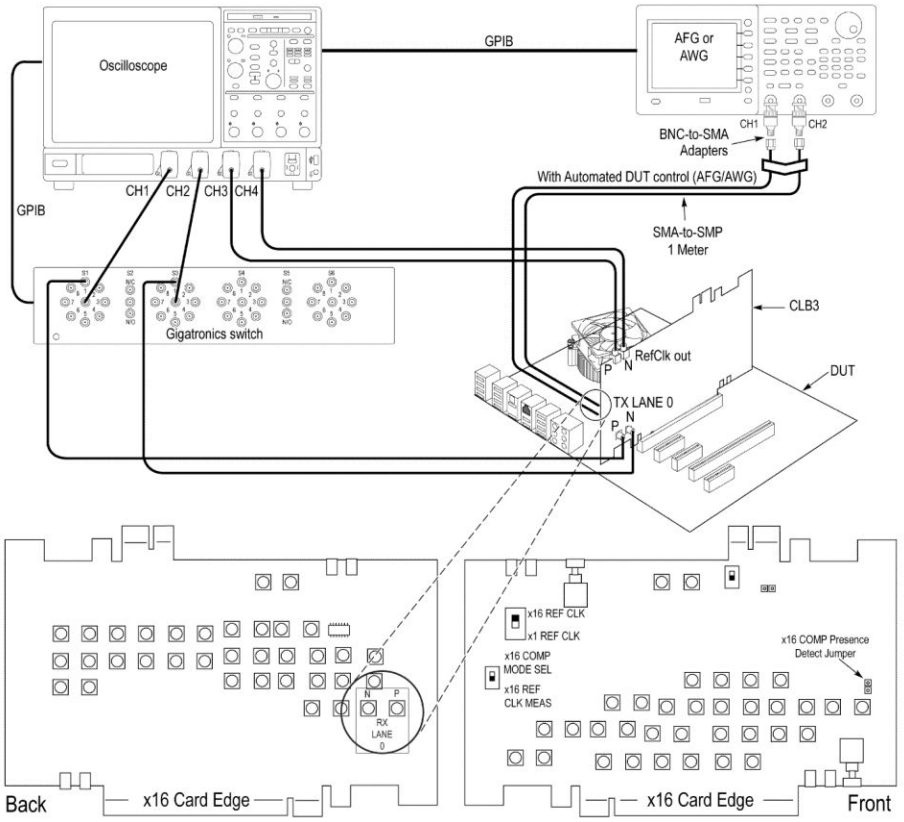


Figure 4: System board (normal) connection setup through switch system

Gigatronics Switch: DUT Lane to Signal connection mapping			
TX LANE 0 P	S1 (relay) > Signal 1	TX LANE 0 N	S3 > Signal 1
TX LANE 1 P	S1 > Signal 2	TX LANE 1 N	S3 > Signal 2
TX LANE 2 P	S1 > Signal 3	TX LANE 2 N	S3 > Signal 3
TX LANE 3 P	S1 > Signal 4	TX LANE 3 N	S3 > Signal 4
TX LANE 4 P	S1 > Signal 5	TX LANE 4 N	S3 > Signal 5
TX LANE 5 P	S1 > Signal 6	TX LANE 5 N	S3 > Signal 6
TX LANE 6 P	S1 > Signal 7	TX LANE 6 N	S3 > Signal 7
TX LANE 7 P	S1 > Signal 8	TX LANE 7 N	S3 > Signal 8

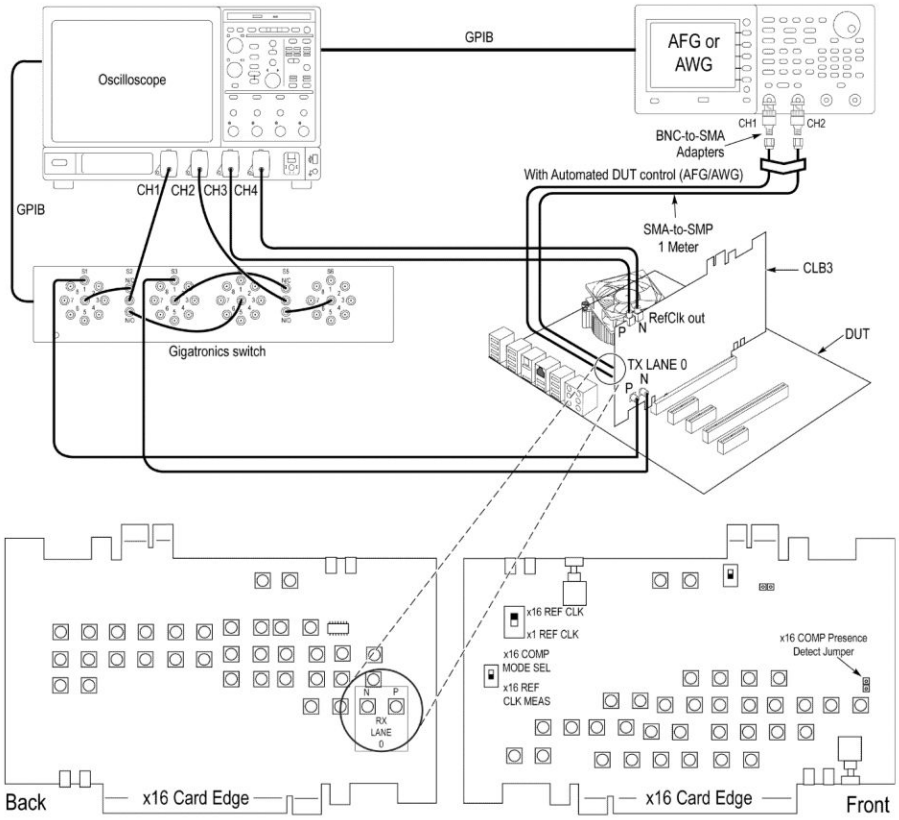


Figure 5: System board (cascade) connection setup through switch system

Gigatronics Switch: DUT Lane to Signal connection mapping			
TX LANE 0 P	S1 (relay) > Signal 1	TX LANE 0 N	S3 > Signal 1
TX LANE 1 P	S1 > Signal 2	TX LANE 1 N	S3 > Signal 2
TX LANE 2 P	S1 > Signal 3	TX LANE 2 N	S3 > Signal 3
TX LANE 3 P	S1 > Signal 4	TX LANE 3 N	S3 > Signal 4
TX LANE 4 P	S1 > Signal 5	TX LANE 4 N	S3 > Signal 5
TX LANE 5 P	S1 > Signal 6	TX LANE 5 N	S3 > Signal 6
TX LANE 6 P	S1 > Signal 7	TX LANE 6 N	S3 > Signal 7
TX LANE 7 P	S1 > Signal 8	TX LANE 7 N	S3 > Signal 8
TX LANE 8 P	S4 > Signal 1	TX LANE 8 N	S6 > Signal 1
TX LANE 9 P	S4 > Signal 2	TX LANE 9 N	S6 > Signal 2
TX LANE 10 P	S4 > Signal 3	TX LANE 10 N	S6 > Signal 3
TX LANE 11 P	S4 > Signal 4	TX LANE 11 N	S6 > Signal 4
TX LANE 12 P	S4 > Signal 5	TX LANE 12 N	S6 > Signal 5
TX LANE 13 P	S4 > Signal 6	TX LANE 13 N	S6 > Signal 6
TX LANE 14 P	S4 > Signal 7	TX LANE 14 N	S6 > Signal 7
TX LANE 15 P	S4 > Signal 8	TX LANE 15 N	S6 > Signal 8

NOTE. *GPIB is the recommended interface to execute the switch matrix commands.*

Running tests

Test setup overview

Test setup includes acquisition and configuration parameters. You can also select report options when setting up tests. Use the options in the *Setup panel* and *Reports panel* to select and configure tests.

1. *Set up equipment*.
2. *Do the prerun checklist*.
3. *Set DUT parameters*.
4. *Select one or more tests*.
5. *Select acquisitions*.
6. *Configuration test parameters*.
7. *Select test measurement notification options*.
8. *Select report options*.

See also

About test setups

Before you click star

About running tests

About running tests

After selecting and configuring tests, review the *Prerun checklist* and then click **Start** to run the tests. While tests are running, you cannot access the Setup or Reports panels. To monitor the test progress, switch back and forth between the Status panel and the Results panel.

The application displays a report when the tests are complete. While the tests are running, other applications may display windows in the background. The TekScope application takes precedence over other applications, but you can switch to other applications by using the **Alt + Tab** key combination. To keep the TekExpress PCIe application on top, select **Keep On Top** from the TekExpress Options menu.

See also

Before you click start

About configuring tests

About setting up tests

Prerun checklist

Do the following before you click Start to run a test. If this is the first time you are running a test on a setup, refer to the information in *Before you click start*.

1. Make sure that all the required instruments are properly warmed up (approximately 20 minutes).
2. Perform Signal Path Compensation (SPC):
   a. On the oscilloscope main menu, select the **Utilities** menu.
   b. Select **Instrument Calibration**.
3. Verify that the application is able to find the DUT. If it cannot, perform a search for connected instruments:
   a. In PCIe, select the **Setup** panel and then click the **Test Selection** tab.
   b. Select any test and then click **Configure**.
   c. n the Configuration section, click **Global Settings**.
   d. In the **Instruments Detected** section, click the drop-down arrow to the right of **Real Time Scope** and make sure that the oscilloscope with the (GPIB8::1::INSTR) designation is in the list.

See also

Equipment connection setup

Saving and recalling test setups

About test setups

TekExpress PCIe opens with the default setup selected. Run a test before or after saving a setup. When you save a setup, the test information, such as the selected oscilloscope, general parameters, acquisition parameters, measurement limits, waveforms (if applicable), and other configuration settings are all saved under the setup name at *X:\PCI Express*.

Use test setups to:

- Run a saved test in prerecorded mode.
- View all the information associated with a saved test, including the log file, the history of the test status as it executed, and the results summary.
- Create a new test setup based on an existing one.
- Run a new session, acquiring live waveforms, using a saved test configuration.

See also

About setting up tests

Save a test setup

Recall a saved test setup

Save a test setup

Save a test setup before or after running a test to save the test configuration. Create a new test setup from any open setup or from the default setup. When you select the default test setup, all parameters are returned to the application's default values.

To save the current setup session to the same setup name, select **Options > Save Test Setup**.

To save the current setup session to a new setup name, select **Options > Save Test Setup As**.

To create and save a new setup from the default test setup:

1. Select **Options > Default Test Setup**.
2. Select **Setup** and set required options and parameters in the tabs (DUT, Test Selection, and so on).
3. Select **Reports** and set your *report options*.
4. Optional: Click **Start** to run the test and verify that it runs correctly and captures the information you want. If it does not, edit the parameters and repeat this step until the test runs to your satisfaction.

   Running the test helps verify that all parameters are set correctly, but it is not a necessary step.
5. Select **Options > Save Test Setup**. Enter the file name for the setup file. The application saves the file to *X:\PCI Express\<session_name*.

See also

About setting up tests

Test setup overview

View test-related files

About configuring tests

Open (load) a saved test setup

These instructions are for recalling saved test setups.

1. Select **Options > Open Test Setup**.
2. Select the setup from the list and click **Open**. Setup files must be located at *X:\PCI Express*.

See also

About test setups

Create a new test setup based on an existing one

Test setups overview

Create a new test setup based on an existing one

Use this method to create a variation on a test setup without having to create the setup from the beginning.

1. Select **Options > Open Test Setup**.
2. Select a setup from the list and then click **Open**.
3. Use the **Setup** and **Reports** panels to modify the parameters to meet your testing requirements.
4. Select **Options > Save Test Setup As**.
5. Enter a test setup name and click **Save**.

NOTE.

- *Select **Default Test Setup** before execution of every test and save the session with a valid name; this will prevent overwriting/losing data.*
- *When you save session using the option **Save Test Setup**, the "Untitled Session" folder and the "Untitled Session.Tekx" file will get renamed with the session name provided by you. It takes few seconds to store the data in the folder.*
- *When you save session using the option **Save Test Setup As**, the " Untitled Session" folder and the " Untitled Session.Tekx" fie will get named with session name provided by you. It takes few seconds to store the data in the folder. The same is applicable when you use the **Save as** option to save the already saved session files.*

See also

About test setups

Set DUT parameters

Configuration parameters

Select acquisitions

Switch Matrix application

Product description

Switch Matrix application allows to configure and setup automated multi-lane testing using RF switch. The solution allows you to map each of the several transmitter signals and forward the selected input either to another relay or to the oscilloscope channel.

Application overview

This section describes the Switch Matrix application settings.

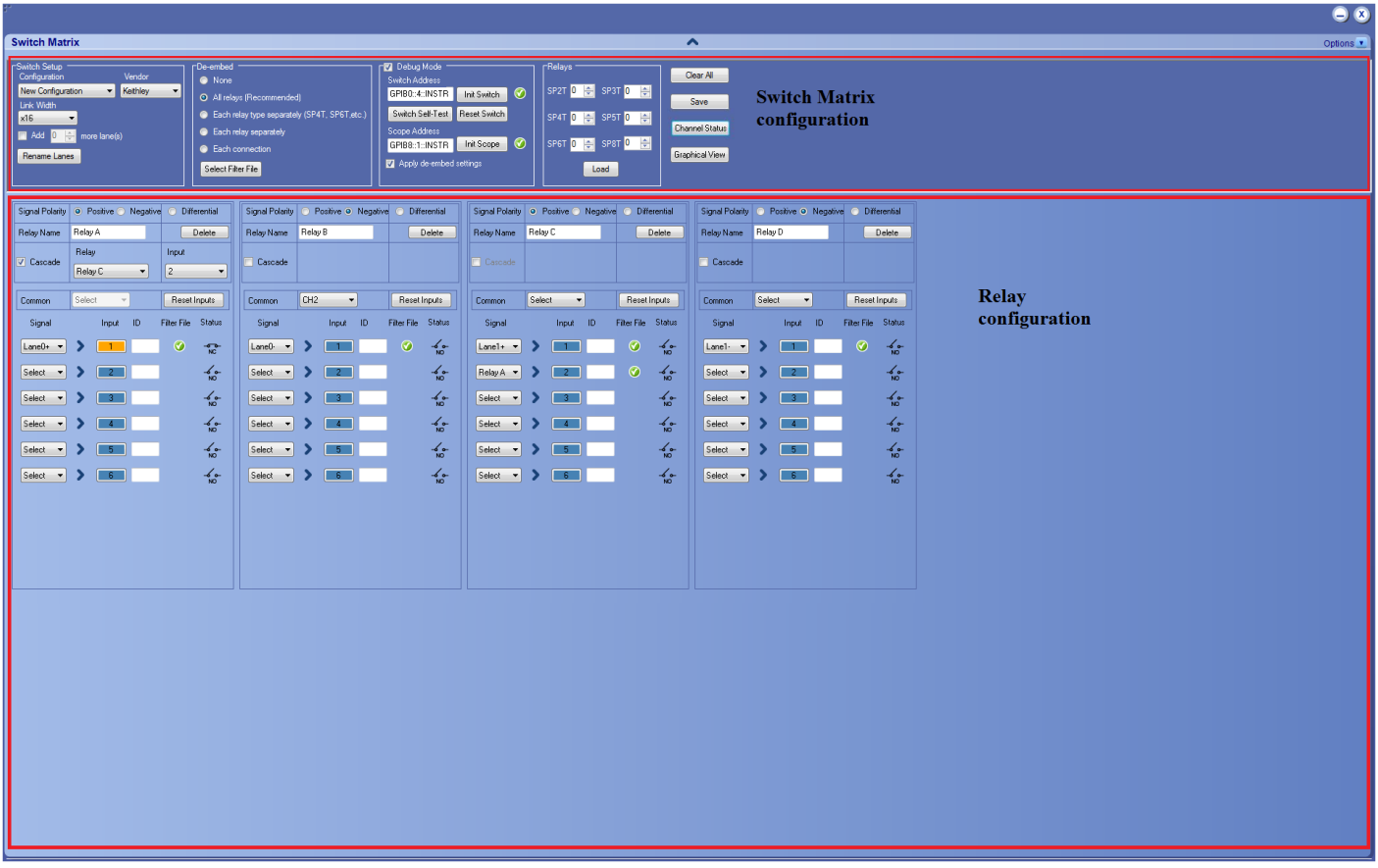
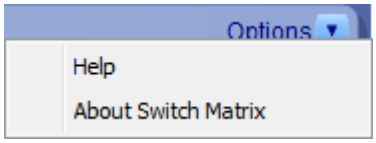
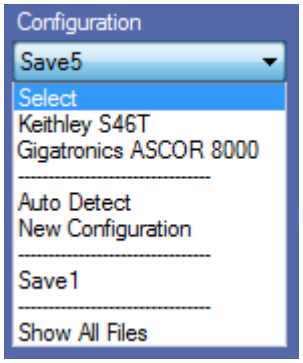
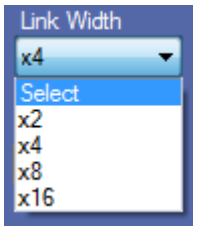
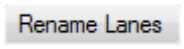
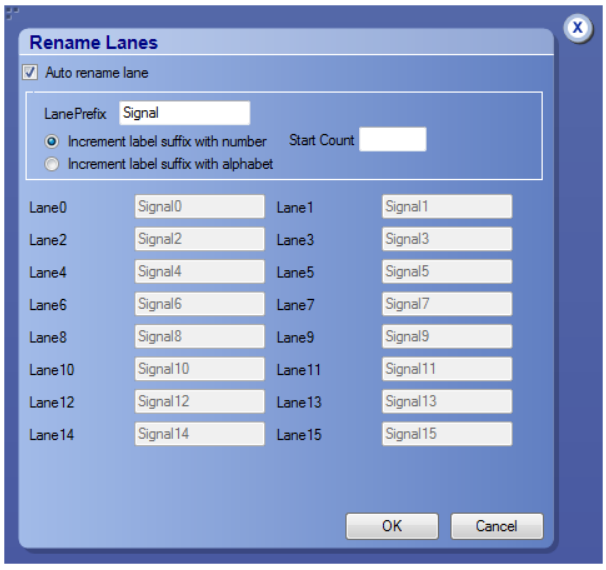
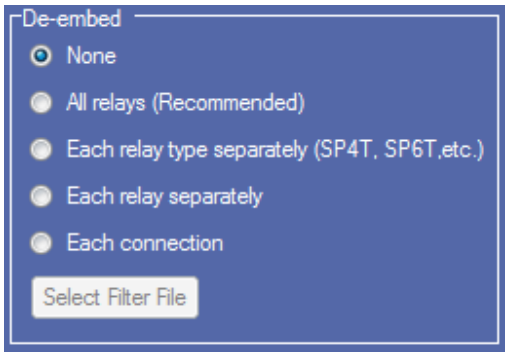


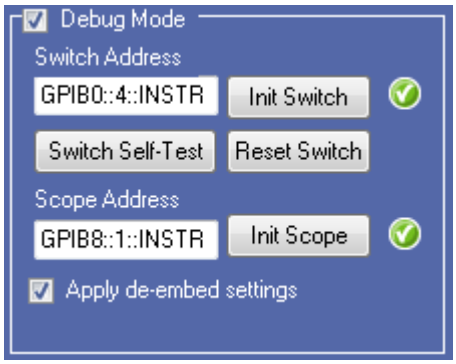
Table 11: Switch Matrix configuration settings

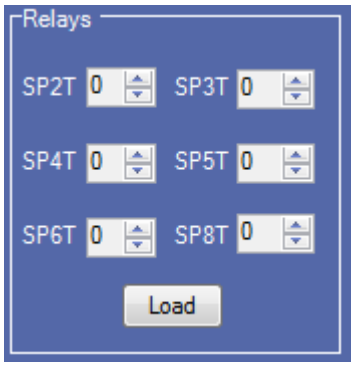
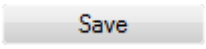
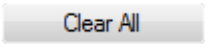
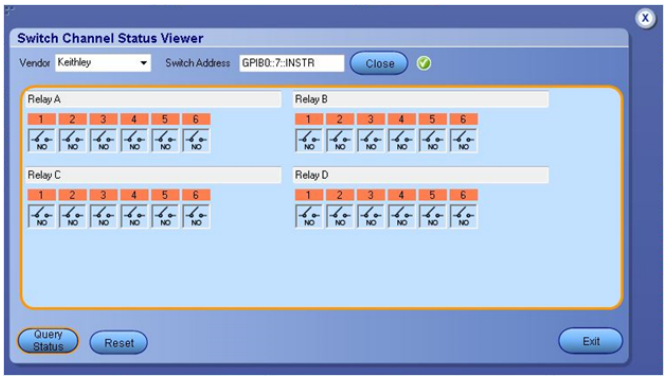
Item	Description
	Click to expand/collapse the switch matrix configuration.
Options 	Click **Help** to view the software help document and **About Switch Matrix** for software version.
Switch Matrix configuration	
Configuration 	Select the configuration option: ■ **Keithley S46T**: 6-input-to-1-output switch configuration ■ **Gigatronics ASCOR 8000**: 8-input-to-1-output switch configuration ■ **Auto Detect**: Select to autodetect the switch. ■ **New Configuration**: Select to manually configure the switch. ■ **Saved file names**: Saved configuration file name(s) are displayed in the drop-down list. Select to recall the configuration. ■ ***Show All Files***: Select to view the list of all saved files.
Vendor 	Select the vendor from the drop-down list. This field is displayed: ■ When you select **Configuration** > **New Configuration** to create afresh configuration. ■ When you open a saved configuration. The displayed vendor name is not editable. ■ When Auto Detect is selected. The displayed vendor name is not editable.

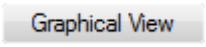
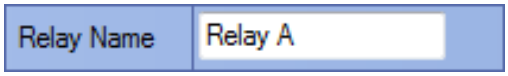
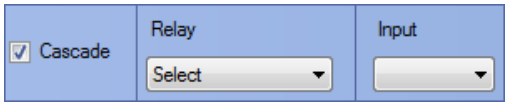
Item	Description
Link Width 	Select the Link Width from the drop-down list. This determines the maximum number of lanes supported by the DUT.
Add <X> more lane(s) 	Select to add extra lanes (Additional1, Additional2,....) to the lanes list. The extra lanes added are displayed in the relay signals. You can add a maximum of 10 lanes.
Rename Lanes 	Click to rename the lanes. Enter the LanePrefix and select the increment label type to suffix by either number or alphabet. The number of lanes depends on the Link Width selected. Clear the Auto rename lane check box to set unique names for the lanes. 
De-embed	

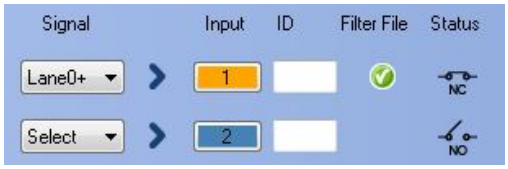
Item	Description
De-embed [1] 	Select the De-embed option: ■ None ■ *All relays (Recommended)* ■ *Each relay type separately (SP4T, SP6T, etc.)* ■ *Each relay separately* ■ *Each connection* Select the de-embed option and click **Select Filter File** to browse and select the filter file(s).
Debug Mode	

[1] Configure at least one relay before configuring the de-embed settings.

Item	Description
Debug Mode 	Select Debug Mode to manually configure the switch. **Switch Address** Enter the Switch Address in the GPIB or TCPIP format. GPIB format: GPIB0:X:INSTR TCPIP format: TCPIP::IPADDR::INSTR **Init Switch** This will synchronize the configuration of relay(s) in the application with the relay(s) in the switch. Synchronization will be successful only for the relays whose configuration matches with the physical switch. Pass/Fail status is displayed next to the button. ***NOTE.*** *Relay configurations (number of relays, number of relay inputs, and name of relays) in the application should match the physical switch, for successful synchronization.* **Switch Self-Test** This will close and open all switch channels one-by-one and displays the pass/fail status of the channel next to the ID. A self-test report (CSV) is generated at the end of the process. You cannot abort this process. ***NOTE.*** *Initialize the switch before performing the self-test.* **Reset Switch** Click **Reset Switch** to reset the switch. This will open all channels. **Scope Address** Enter the oscilloscope address in the GPIB or TCPIP format. GPIB format: GPIB0:X:INSTR TCPIP format: TCPIP::IPADDR::INSTR **Init Scope** Enter the oscilloscope address in the Scope Address field and click **Init Scope** to initialize the oscilloscope. This will establish the connection with the oscilloscope. The pass/fail status is displayed next to the button. **Apply De-embed settings** Select to apply de-embed settings to the channels. When the oscilloscope is initialized and de-embed settings are configured, closing a connection will apply the de-embed settings and then close the connection.
Relays	

Item	Description
Relays 	Select the relay(s). In SPnT, *n* represents the number of connection signals for the relay. For example, SP4T is a four signal connection relay. This field displays only for a new configuration. By default, zero relays are selected. Enter the total number of relays to be loaded in their respective input box and click **Load**. You can also click or to increase or decrease the number.
Save 	Click to save the configuration at *C:\ProgramData\Tektronix\Switch Matrix Configurations*.xml*. This operation checks whether all the required configurations are done. If any of the required configurations are not selected, then error popup is displayed, which prompts you to complete the configuration(s).
Clear All 	Click to clear all configurations. The application will be loaded with Configuration drop-down (default).
Channel Status 	Click to view the relays and status of channels of Keithley or Gigatronics switch. This updates the channel status dynamically. In Switch Channel Status Viewer, select the Vendor type, enter the Switch Address and click **Init** to initialize the switch. This will establish the connection with the switch. Click **Query Status** to get the details of the relays of the switch and the status of the channels. Click **Reset** to reset the status viewer. 

Item	Description
Graphical View 	Click to view the graphical representation of the configured relays. If the relays are cascaded, then they are also displayed in the graphical representation.
Relay configuration	
Signal Polarity 	Select the signal polarity of DUT: ■ **Positive**: populates Lane0+, Lane1+, …. connection signals. ■ **Negative**: populates Lane0-, Lane1-, …. connection signals. ■ **Differential**: populates Lane0, Lane1, …. connection signals.
Relay Name 	Enter the relay name. This name should match the relay name of the connected switch.
Delete 	Click to delete the relay. This configuration is only available for the configured (loaded) relays, when Configuration > New Configuration is selected.
Cascade 	Select to cascade the relay by connecting the common channel as the input signal for another relay. Select the Relay and the Input of the relay. Check that the selected relay signal displays the appropriate relay name. The cascade settings is also displayed in the graphical view. *Click here* to get details about Cascade. ***NOTE.*** *Select the cascade settings before you save the configuration.*

Item	Description
Common 	Select the oscilloscope channel for Common. If cascaded, it displays the name of the relay. Click **Reset Inputs** to clear all connection signal settings. ***NOTE.*** *Select the common settings for all the relays, before you save the configuration.*
	**Signal** Select the DUT connection signal. This drop-down list shows the lanes based on *Link Width* and *Signal Polarity* settings. If the link width is x8 and signal polarity is Positive, then the Signal drop-down list will have Lane0+ to Lane7+ options. **Input** This button is enabled only in debug mode and if a valid signal is configured for the channel. Click to close or open the channel. **ID** Enter the three character alias name for the channel. This is shown in the *graphical view of switch matrix configuration*. **Filter File** This column shows  or  indicating the status of the filter file configuration for the channel. If no de-embed option is selected, then this column remains blank. **Status** This column displays the status of the channel. Channel closed (normally closed) :  Channel opened (normally opened) : 

File name extensions

This application uses the following file name extensions:

File name extension	Description
.html	Saved configuration and Graphical view file formats
.xml	Switch Matrix configuration files

Status indicators

Status indicators	Description
	Success indicator
	Failure indicator
	Closed channel indicator (NC = Normally Closed)
	Opened channel indicator (NO = Normally Opened)

Saved configurations

Click **Configuration** > **Show All Files** to view the list of all saved files.

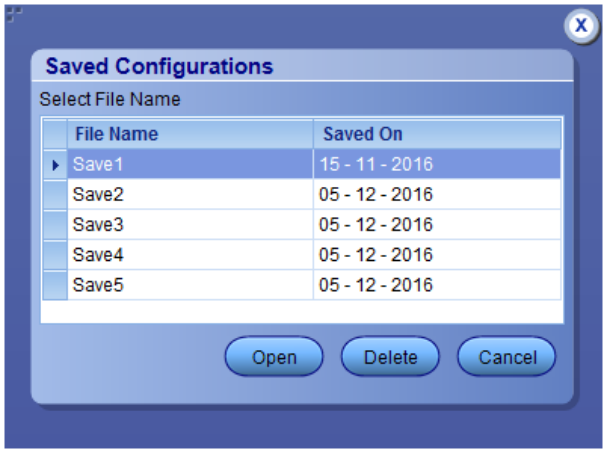


Table 12: Saved configurations

Item	Description
Open	Opens the selected file.
Delete	Deletes the selected file.
Cancel	Closes the Saved Configurations window.

De-embed settings

De-embed allows you to apply filter file(s) for relay(s). Select the De-embed option and click **Select Filter File** to browse and select the filter file(s).

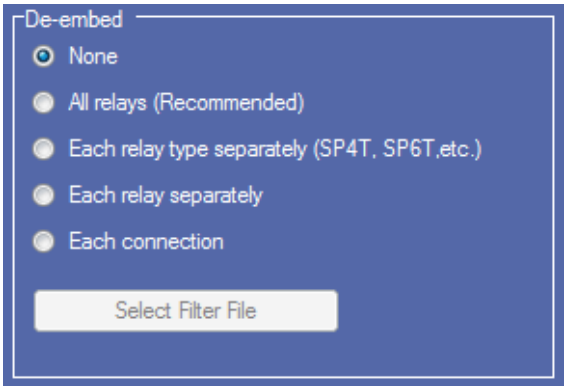
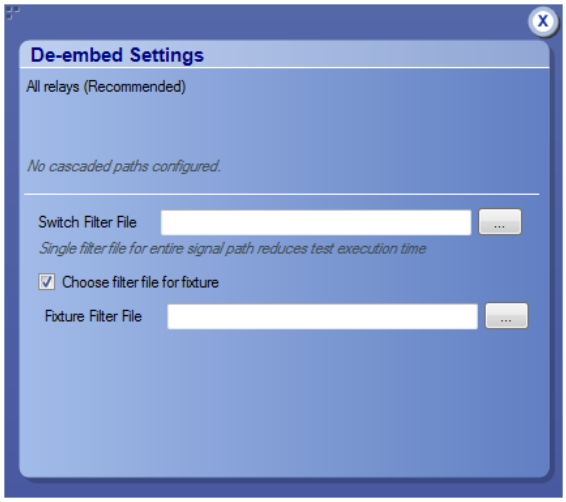


Figure 6: De-embed options

Apply a filter file for all relays

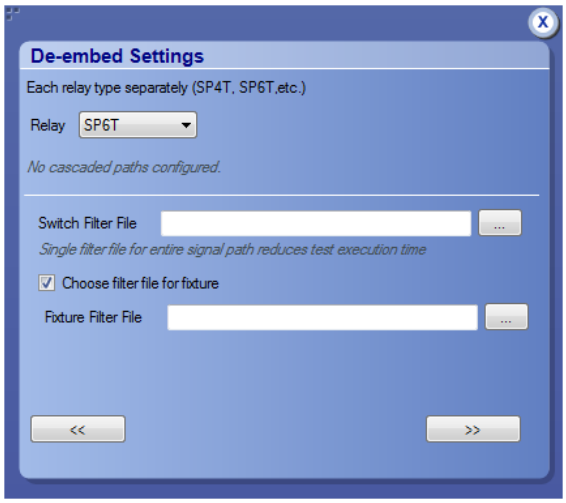
Select to apply a single filter file for all relays.



Click ... to browse and select the filter file for the switch. To apply the filter file for the fixture, select **Choose filter file for fixture** and browse the filter file.

Apply a filter file for each relay type separately

Select to separately apply a single filter file for each relay type.



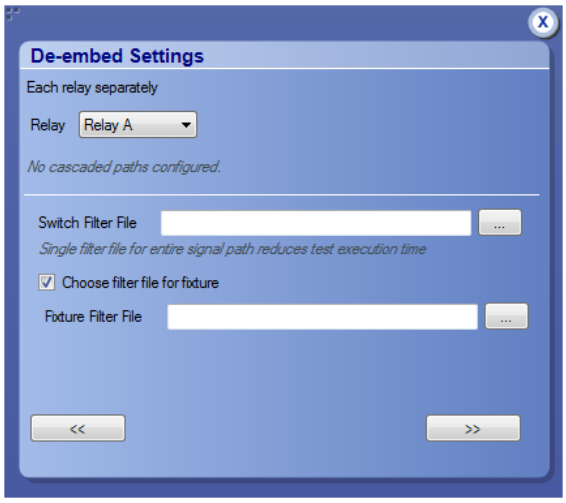
Select the Relay type from the drop-down list; click ... to browse and select the filter file for the switch. To apply the filter file for the fixture, select **Choose filter file for fixture** and browse the filter file.

TIP. *Click* << *or* >> *to select the previous or next relay type.*

TIP. *The selected relay types are highlighted in dark blue in the application.*

Apply a filter file for each relay separately

Select to separately apply a filter file for each relay.



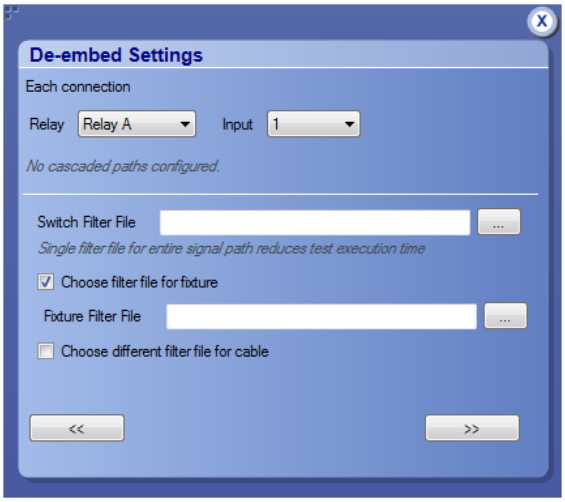
Select the Relay from the drop-down list; click ... to browse and select the filter file for the switch. To apply the filter file for the fixture, select **Choose filter file for fixture** and browse the filter file.

TIP. *Click << or >> to select the previous or next relay.*

TIP. *The selected relay is highlighted in dark blue in the application.*

Apply a filter file for each connection separately

Select to apply a filter file for each connection.



Select the Relay and the Input from the drop-down list; click ... to browse and select the filter file for the switch. To apply the filter file for the fixture, select **Choose filter file for fixture** and browse the filter file. Select Choose different filter file for cable to browse and select the filter file for cable.

TIP. *Click << or >> to select the previous or next channel.*

TIP. *The selected relay signal is highlighted in dark blue in the application.*

NOTE. *Clicking >> on the last input of a relay selects the first input of the next relay; clicking << on the first input of a relay selects the last input of the previous relay.*

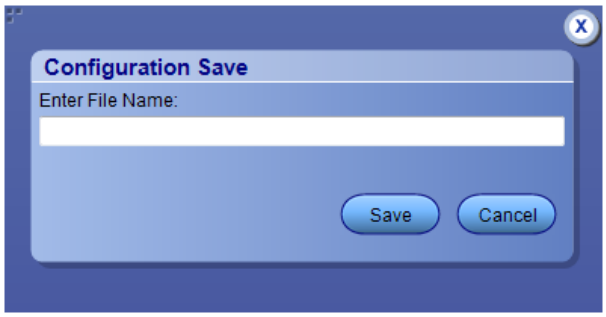
Graphical view of switch matrix configuration

The Graphical view displays the pictorial representation of the switch configuration.

The following figure displays the graphical view of a Keithley switch configuration.

Saving the configuration

Click **Save** in the configuration panel; in the Configuration Save dialog box, enter the file name and click **Save**. The default save path is *C:\ProgramData\Tektronix\Switch Matrix Configurations*.



NOTE. *Save configuration checks whether all the required configurations are done. If any of the required configurations are not selected, then error message is displayed, which prompts you to complete the configuration(s).*

Feature description

Link width

Link width determines the number of DUT signals. For example, x8 represents an eight lane DUT.

This works in conjunction with the signal polarity selected for each relay. For example, if the link width is x8, and:

- If the signal polarity is **Positive**, then the signal drop-down list will have selections from Lane0+ to Lane7+.
- If the signal polarity is **Negative**, then the signal drop-down list will have selections from Lane0- to Lane7-.
- If the signal polarity is **Differential**, then the signal drop-down list will have selections from Lane0 to Lane7.

Debug mode

Init Switch. Enter the Switch Address and click **Init Switch** to initialize the switch. This will synchronize the configuration of relay(s) in the application with the relay(s) in the switch. Synchronization will only be successful for those relays that match the physical switch. Pass/Fail status is displayed next to the button.

The factory default GPIB address for Keithley (GPIB0::7::INSTR) and Gigatronics (GPIB0::4::INSTR) is populated in the switch address based on the configured vendor. You can enter the address in GPIB (GPIB0:X:INSTR) or TCPIP (TCPIP::IPADDR::INSTR) format.

NOTE. *Relay configurations (number of relays, number of relay inputs, and name of relays) in the application should match the physical switch, for successful synchronization.*

Switch Self-Test. This will close and open all switch channels one-by-one. A selftest report (CSV) is generated at the end of the process. You cannot abort this process.

NOTE. *Initialize the switch before performing the Switch Self-Test.*

Reset Switch. Click **Reset Switch** to reset the switch. This will open all channels.

Init Scope. Enter the oscilloscope address in the Scope Address field and click **Init Scope** to initialize the oscilloscope. This will establish the connection with the oscilloscope. The pass/fail status is displayed next to the button.

You can enter the address in GPIB (GPIB0:X:INSTR) or TCPIP (TCPIP::IPADDR::INSTR) format.

When the oscilloscope is initialized and de-embed settings are configured, closing a connection will apply the de-embed settings and then close the connection.

NOTE. *The virtual GPIB address of the oscilloscope is GPIB8::1::INSTR.*

NOTE. *If oscilloscope fails to respond to the *IDN? query during initialization, then the connection attempt is considered a failure.*

Cascade (Relay cascade)

This feature allows you to cascade the relay by connecting the common channel as an input signal for another relay.

To cascade, select **Cascade** in the relay and select the Relay and Input of the relay. Check that the selected relay signal displays the relay name, specifying that the lane input signal is the output from that relay.

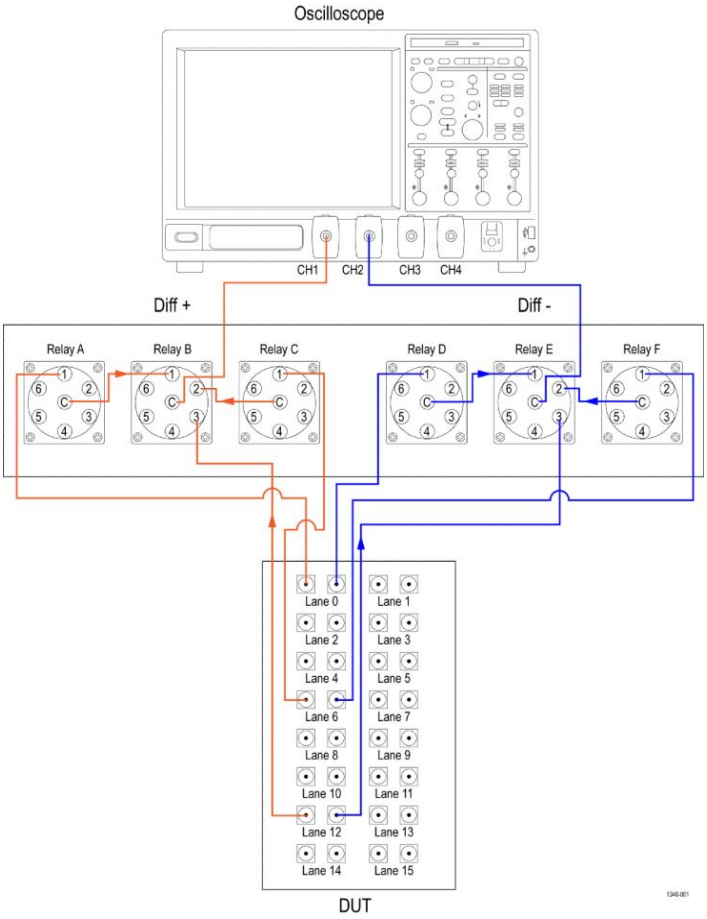


Figure 7: Graphical view of relay cascade configuration

TIP. *Switch Matrix application supports only single-level cascading of the relays. For example, if the Relay A output is cascaded to Relay B, then the Relay B output cannot be cascaded.*

Error messages

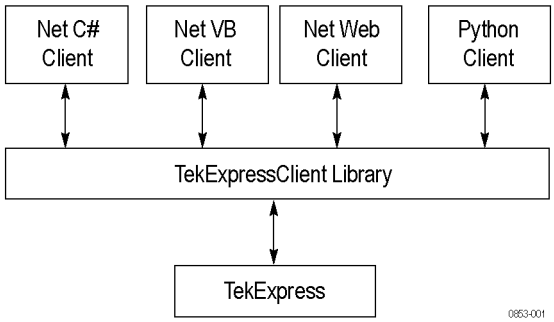
Error message	Possible solution
"A filename cannot be empty and it cannot contain any of the following characters:\n\t. .. \ \ / : ? \" < > | * ! @ # $ % ^ & * ( ) - + ., / \\ \' < > Also, the file name cannot be \"Keithley S46T\", \"Gigatronics ASCOR 8000\",\"Select\", \"New Configuration\", \"Custom\" , \"Auto Detect\" or \"Show All Files\""	
Configure appropriate signals before the de-embed settings.	Select at least one signal for a relay before configuring the de-embed settings.
Either the instrument address is invalid or instrument is not connected.	Check the GPIB connection from oscilloscope to switch and verify the instrument address.
Error occurred while trying to recall the configuration settings. Try re-creating configuration or recalling a different configuration file.	Re-create the configuration file or recall a different configuration file.
Error occurred while trying to access the connection for open/close operation.	
Filter file <FilterFileName> not found.	Reselect the de-embed filter file and try again.
Graphical view is not generated or does not exist.	
Initialize the switch	Initialize the switch and then perform the switch operations.
Instrument address doesn't belong to any supported switch.	Verify the switch address.
Instrument address is empty.	Instrument address cannot be empty. Enter a valid instrument address in the GPIB (*GPIB0:X:INSTR*) or TCPIP (*TCPIP::IPADDR::INSTR*) format.
No switch detected. Connect a Keithley or Gigatronics switch and try auto detection by selecting Configuration > Auto Detect.	Check the GPIB connection from the oscilloscope to switch and whether the instrument is detected in TekVisa.
Number of relays cannot be more than 26	
Please ensure that the name(s) of the configured relay(s) match the ones present on the physical switch.	

Error message	Possible solution
Relay name cannot be empty	
Scope initialization failed. Check if the address is valid and ensure that the instrument is switched on and try again.	Validate the oscilloscope address try again.
Switch communication failed...	Ensure that the switch is on. Reset the switch and try again.
Switch initialization failed. Check if the address is valid and ensure that the instrument is switched on and try again.	Validate the switch address and ensure that the instrument is switched on. Try again.
The start count cannot be more than 74	
Timeout Error. Either the command is invalid or instrument is not active.	Check the command syntax and the connection of the instrument by *SWITCH:*IDN* command.
Two or more lanes have same name. The lane names should be unique.	
Two or more relays have same name.	

TekExpress programmatic interface

About the programmatic interface

The Programmatic interface seamlessly integrates the TekExpress test automation application with the high-level automation layer. This also lets you control the state of the TekExpress application running on a local or a remote computer.



The following terminology is used in this section to simplify description text:

- **TekExpress Client:** A high-level automation application that communicates with TekExpress using TekExpress Programmatic Interface.
- **TekExpress Server:** The TekExpress application when being controlled by TekExpress Client.

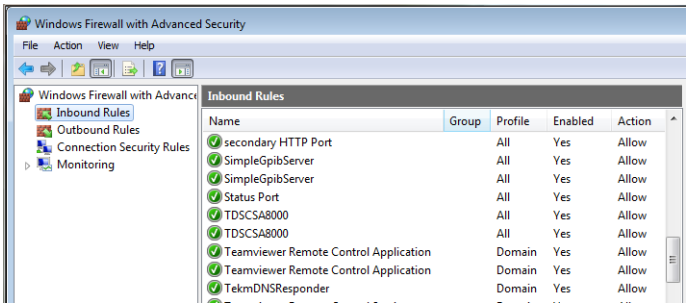
TekExpress leverages .Net Marshalling to enable the Programmatic Interface for TekExpress Client. TekExpress provides a client library for TekExpress clients to use the programmatic interface. The TekExpress client library maintains a reference to the TekExpress Server and this reference allows the client to control the server state.

See also *Requirements for developing TekExpress client*

To enable remote access

To access and remotely control an instrument using the TekExpress programmatic interface, you need to change specific firewall settings as follows:

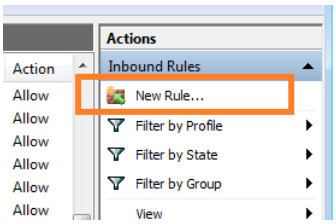
1. Access the Windows Control Panel and open the Windows Firewall tool (**Start > Control Panel > All Control Panel Items > Windows Firewall**).
2. Click **Advance Settings > Inbound Rules**.
3. Scroll through the **Inbound Rules** list to see if the following items (or with a similar name) are shown:
   - TekExpress PCI Express
   - TekExpress



4. If both items are shown, you do not need to set up any rules. Exit the Windows Firewall tool.
5. If one or both are missing, use the following procedure to run the **New Inbound Rule Wizard** and add these executables to the rules to enable remote access to the TekExpress application.
6. On the client side, include controller. exe through which TexExpress PCIe application is remotely controlled. For example, if the application is controlled using python scripts the "ipy64.exe" should be included as part of Inbound rules.

Run the New Inbound Rule Wizard

1. Click **New Rule** (in Actions column) to start the **New Inbound Rule Wizard**.

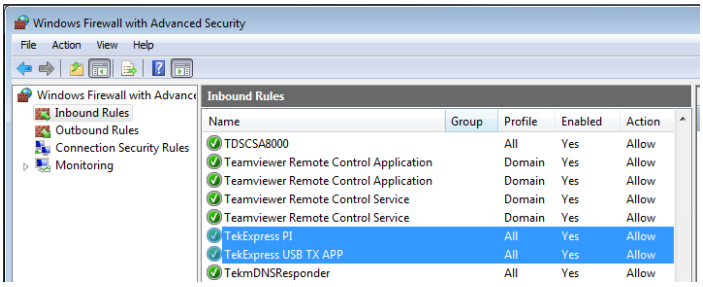


2. Verify that **Program** is selected in the Rule Type panel and click **Next**.
3. Click **Browse** in the Program panel and navigate to and select one of the following TekExpress applications (depending on the one for which you need to create a rule):
4. TekExpress PCI Express.exe
5. TekExpress.exe

NOTE. *See Application directories and content for the path to the application files.*

---

6. Click **Next**.
7. Verify that **Allow the connection** is selected in the Action panel and click **Next**.
8. Verify that all fields are selected (**Domain**, **Private**, and **Public**) in the Profile panel and click **Next**.
9. Use the fields in the Name panel to enter a name and optional description for the rule. For example; **TekExpress PCI Express Application**. Add description text to further identify the rule.
10. Click **Finish** to return to the main Windows Firewall screen.
11. Scroll through the Inbound Rules list and verify that the list shows the rule that you just entered.



12. Repeat steps 1 through 11 to enter the other TekExpress executable if it is missing from the list. Enter **TekExpress PI** as the name.
13. Scroll through the Inbound Rules list and verify that the list shows the rule that you just entered.
14. Exit the Windows Firewall tool.

To use the remote access:

1. Obtain the IP address of the instrument on which you are running TekExpress PCI Express. For example, 134.64.235.198.
2. On the PC from which you are accessing the remote instrument, use the instrument IP address as part of the TekExpress PCI Express PI code to access that instrument. For example:

   object obj = piClient.Connect("134.64.235.198",out clientid);

Requirements for developing TekExpress client

While developing TekExpress Client, use the TekExpressClient.dll. The client can be a VB .Net, C# .Net, TestStand, Python, or Web application. The examples for interfaces are in the TekExpress PCI Express\Examples folder.

NOTE. *The TestStand run time engine is no longer installed as of this release of TekExpress PCI Express. You can continue to use TestStand scripts if you install the TestStand run time engine and set up your environment to use it.*

References required

- TekExpressClient.dll has an internal reference to IIdlglib.dll and IRemoteInterface.dll.
- IIdlglib.dll has a reference to TekDotNetLib.dll.
- IRemoteInterface.dll provides the interfaces required to perform the remote automations. It is an interface that forms the communication line between the server and the client.
- IIdlglib.dll provides the methods to generate and direct the secondary dialog messages at the client-end.

NOTE. *The end-user client application does not need any reference to the above mentioned DLL files. It is essential to have these DLLs (IRemoteInterface.dll, IIdlglib.dll and TekDotNetLib.dll) in the same folder as that of TekExpressClient.dll.*

Required steps for a client

The client uses the following steps to use TekExpressClient.dll to programmatically control the server:

Develop a client UI to access the interfaces exposed through the server. This client loads TekExpressClient.dll to access the interfaces. After TekExpressClient.dll is loaded, the client UI can call the specific functions to run the operations requested by the client. When the client is up and running, it does the following to run a remote operation:

1. To connect to the server, the client provides the IP address of the PC where the server is running.
2. The client locks the server application to avoid conflict with any other Client that may try to control the server simultaneously. "Lock" would also disable

all user controls on the server so that server state cannot be changed by manual operation.

If any other client tries to access a server that is locked, it will receive a notification that the server is locked by another client.

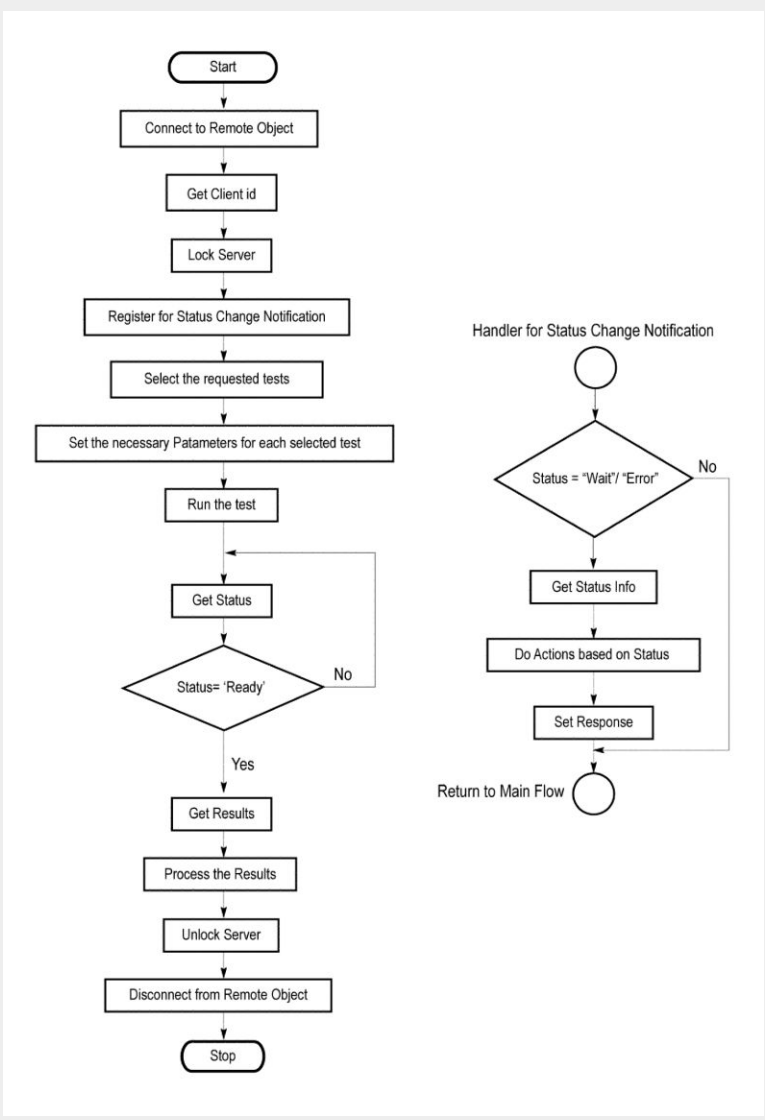
3. When the client has connected to and locked the server, the client can access any of the programmatic controls needed to run the remote automations.
4. After the client operations finish, the client unlocks the server.

See also *PCIe application commands flow*

Client programmatic interface example

An example of the client programmatic interface is described and shown as follows:

Process flowchart



1. Connect to a server or remote object using the programmatic interface provided.
2. Get the client ID that is created when connecting to the remote object. This client ID is one of the required parameters to communicate with the server.

   NOTE. *The server identifies the client with this ID only and rejects any request if the ID is invalid.*

3. Lock the server for further operations. This disables the application interface.

   NOTE. *You can get values from the server or set values from the server to the client only if the application is locked.*

4. Register for receiving notifications on status change events on the server. To register you need to give a handler as a parameter.

   NOTE. *Whenever there is a change in the status of the server, all the clients registered with the server receive a notification from the server.*

5. Select the tests that you want to run through the programmatic interface.
6. Set the necessary parameters for each test.
7. Run the tests.
8. Poll for the status of the application.

   NOTE. *Skip step 8 if you are registered for the status change notification and the status is Ready.*

9. After completing the tests, get the results.
10. Create a report or display the results and verify or process the results.
11. Unlock the server after you complete all the tasks.
12. Disconnect from the remote object.

Handler of status change notification

1. Get the status. If the status is Wait or Error, get the information that contains the title, message description, and the expected responses for the status.
2. Perform the actions based on the status information.
3. Set the response as expected.

See also

PCIe application commands flow

Program remote access code example

Program remote access code example

This code example shows how to communicate between a remote PC and TekExpress PCI Express.

Table 13: Remote access code example

Task	Code
Start the application	
Connect through an IP address.	m_Client.Connect("localhost") 'True or False clientID = m_Client.getClientID
Lock the server	m_Client.LockServer(clientID)
Disable the Popups	m_Client.SetVerboseMode(clientID, false)
Set the DUT ID	m_Client.SetDutId(clientID, "DUT_Name")
Run with set configurations	m_Client.Run(clientID)
Wait for the test to complete.	Do Thread.Sleep(500) m_Client.Application_Status(clientID) Select Case status Case "Wait"
Get the current state information	mClient.GetCurrentStateInfo(clientID, WaitingMsbBxCaption, WaitingMsbBxMessage, WaitingMsbBxButtontexts)
Send the response	mClient.SendResponse(clientID, WaitingMsbBxCaption, WaitingMsbBxMessage, WaitingMsbBxResponse) End Select Loop Until status = "Ready"
Save results	Save all results values from folder for current run m_Client.TransferResult(clientID, logDirname)
Unlock the server	m_Client.UnlockServer(clientID)
Disconnect from server	m_Client.Disconnect()
Exit the application	

Python and C# examples

Example Python and C# test sequence files are located atTekExpress PCI Express\Examples\.

PCIe application commands

PCIe application commands listing

Click a client action link to see the associated command name, description, parameters, return value, and an example.

Connect through an IP address

Lock the server

Disable the popups

Set or get the DUT ID

Select the PCIe device

Select the suite

Set the PCIe test version

Set the test mode parameter

Set the prerecorded waveform execution mode

Set the data rate parameter

Set the 5 Gbs preemphasis parameter

Set the SSC parameter

Set the voltage swing parameter

Set the signal quality preset parameter

Set the preset lanes parameter

Set the lane source parameter

Set the preset parameter

Set the acquisition parameter

Set the analysis mode parameter

Set the Sigtest version parameter

Set the on failure action parameter

Set the report update mode parameter

Set the append report parameter

Set the crosstalk parameter

Set the waveform save parameter

Set the link analysis embed/de embed signal parameter

Set the link analysis embed/de embed filter file parameter

Set the link analysis other filter file parameter

Set the link analysis equalization dropdown parameter

Set the link analysis CTLE index paramete

Set the link analysis DFE parameter

Set the DUT auto toggle parameter

Set the DUT auto toggle options parameter

Set the AFG signal type parameter

Set the AFG signal frequency parameter

Set the AFG signal amplitude parameter

Set the burst count parameter

Set the record length parameter

Set the sample rate parameter

Set the bandwidth parameter

Set the signal validation parameter

Set the slot number parameter

Set SigTest Interface mode

Set Sig Validation Threshold

Set Toggle from Gen3P10 to Gen1

Run with set configurations or stop the run operation

Handle error codes

Get or set the timeout value

Wait for the test to complete

After the test is complete

Save, recall, or query a saved session

Unlock the server

Disconnect from the server

string id

Name	Type	Direction	Description
id	string	IN	Identifier of the client performing the remote function

Ready: Test configured and ready to start

Running: Test running

Paused: Test paused

Wait: A popup that needs your inputs

Error: An error is occurred

string dutName

Name	Type	Direction	Description
dutName	string	IN	The new DUT ID of the setup

out bool saved

Name	Type	Direction	Description
saved	bool	OUT	Boolean representing whether the current session is saved

This parameter is used as a check in SaveSession() and SaveSessionAs() functions.

string ipAddress

Name	Type	Direction	Description
ipAddress	string	IN	The ip address of the server to which the client is trying to connect. This is required to establish the connection between the server and the client.

out string clientID

Name	Type	Direction	Description
clientid	string	OUT	Identifier of the client that is connected to the server clientId = unique number + ipaddress of the client. For example, 1065–192.157.98.70

NOTE. *If the dutName parameter is null, the client is prompted to provide a valid DUT ID.*

NOTE. *The server must be active and running for the client to connect to the server. You can connect any number of clients to the server at a time.*

NOTE. *When the client is disconnected, it is unlocked from the server and then disconnected. The id is reused.*

string dutId

Name	Type	Direction	Description
dutId	string	OUT	The DUT ID of the setup. for example,

The dutId parameter is set after the server processes the request.

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “PCIe”.

string suite

Name	Type	Direction	Description
suite	string	IN	Specifies the name of the suite

string test

Name	Type	Direction	Description
test	string	IN	Specifies the name of the test to obtain the pass or fail status or a test result value. Append spaces at the end of the test name to differentiate the PCIe DUT generation version for which to return measurements: Gen1: no spaces Gen2: One space Gen3: Two spaces **Examples:** Gen1: “Unit Interval” Gen2: “Unit Interval ” Gen3: “Unit Interval ”

string parameterString

Name	Type	Direction	Description
parameterString	string	IN	Selects or deselects a test

int rowNr

Name	Type	Direction	Description
rowNr	int	IN	Specifies the zero based row index of the sub-measurement for obtaining the result value

NOTE. *When the client tries to lock a server that is locked by another client, the client gets a notification that the server is already locked and it must wait until the server is unlocked. If the client locks the server and is idle for a certain amount of time then the server is unlocked automatically from that client.*

out string[] status

Name	Type	Direction	Description
status	string array	OUT	The list of status messages generated during the run

string name

Name	Type	Direction	Description
name	string	IN	The name of the session being recalled

The name parameter cannot be empty. If it is empty, the client is prompted to provide a valid name.

NOTE. *When the run is performed, the status of the run is updated periodically using a timer.*

string name

Name	Type	Direction	Description
name	string	IN	The name of the session being saved

The name parameter cannot be empty. If it is empty, the client is prompted to provide a valid name.

Once the session is saved under 'name' you cannot use this method to save the session in a different name. Use SaveSessionAs instead.

string name

Name	Type	Direction	Description
name	string	IN	The name of the session being recalled

The same session is saved under different names using this method. The name parameter cannot be empty. If it is empty, the client is prompted to provide a valid name.

bool isSelected

Name	Type	Direction	Description
isSelected	bool	IN	Selects or deselects a test

string time

Name	Type	Direction	Description
time	string	IN	The time in seconds that refers to the timeout period

The time parameter gives the timeout period, which is the time the client is allowed to be locked and idle. After the timeout period if the client is still idle, it gets unlocked.

The time parameter should be a positive integer; otherwise, the client is prompted to provide a valid timeout period.

bool_verbose

Name	Type	Direction	Description
_verbose	bool	IN	Specifies whether the verbose mode should be turned ON or OFF

NOTE. *When the session is stopped, the client is prompted to stop the session and is stopped at the consent.*

string filePath

Name	Type	Direction	Description
filePath	string	IN	The location where the report must be saved in the client

NOTE. *If the client does not provide the location to save the report, the report is saved at C:\ProgramFiles.*

NOTE. *When the client is disconnected, the client is unlocked automatically.*

out string WaitingMsbBxCaption

Name	Type	Direction	Description
caption	string	OUT	The wait state or error state message sent to you

out string WaitingMsbBxMessage

Name	Type	Direction	Description
message	string	OUT	The wait state/error state message sent to you

out string[] WaitingMsbBxButtontexts

Name	Type	Direction	Description
buttonTexts	string array	OUT	An array of strings containing the possible response types that you can send

string WaitingMsbBxResponse

Name	Type	Direction	Description
response	string	IN	A string containing the response type that you can select (it must be one of the strings in the string array buttonTexts)

out string clientID

Name	Type	Direction	Description
clientID	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IP address of the client. For example, 1065–192.157.98.70

Connect through an IP address

Command name	Parameters	Description	Return value	Example

Connect()	string ipAddress
	out string clientID
		This method connects the client to the server. Note
		NOTE. *The server must be active and running for the client to connect to the server. You can connect any number of clients to the server at a time.*
		The client provides the IP address to connect to the server.
		The server provides a unique client identification number when connected to it.
			Return value is either True or False	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as boolean
				returnval = m_Client.Connect(ipaddress,m_clientID)

string ipAddress

Name	Type	Direction	Description
ipAddress	string	IN	The ip address of the server to which the client is trying to connect. This is required to establish the connection between the server and the client.

out string clientID

Name	Type	Direction	Description
clientid	string	OUT	Identifier of the client that is connected to the server clientId = unique number + ipaddress of the client. For example, 1065–192.157.98.70

NOTE. *The Fail condition for PI commands occurs in any of the following cases:*

If the server is locked, the application displays "Server is locked by another client."

If the session is unlocked, the application displays "Lock session to execute the command."

If the server is not found, the application displays "Server not found-Disconnect!."

If the fail condition is not one of the above types, the application displays "Failed."

Lock the server

Command name	Parameters	Description	Return value	Example
LockSession()	string clientID	This method locks the server. Note ***NOTE.*** *When the client tries to lock a server that is locked by another client, the client gets a notification that the server is already locked and it must wait until the server is unlocked. If the client locks the server and is idle for a certain amount of time then the server is unlocked automatically from that client* The client must call this method before running any of the remote automations. The server can be locked by only one client.	String value that gives the status of the operation after it was performed The return value is "Session Locked..." on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string returnval = m_Client.LockServer(clientID)

out string clientID

Name	Type	Direction	Description
clientid	string	OUT	Identifier of the client that is connected to the server clientId = unique number + ipaddress of the client. For example, 1065–192.157.98.70

NOTE. *The Fail condition for PI commands occurs in any of the following cases:*

If the server is locked, the application displays "Server is locked by another client."

If the session is unlocked, the application displays "Lock session to execute the command."

If the server is not found, the application displays "Server not found-Disconnect!."

If the fail condition is not one of the above types, the application displays "Failed."

Disable the popups

Use these commands to disable popup messages that require user intervention.

Command name	Parameters	Description	Return value	Example

SetVerboseMode()	string clientID
	bool _verbose
		This method sets the verbose mode to either true or false.
		When the value is set to true, any message boxes that appear during the application are routed to the client machine that is controlling TekExpress.
		When the value is set to false, all the message boxes are shown on the server machine.
			String that gives the status of the operation after it was performed
			When Verbose mode is set to true, the return value is “Verbose mode turned on. All dialog boxes will be shown to client”.
			When Verbose mode is set to false, the return value is “Verbose mode turned off. All dialog boxes will be shown to server”.
				m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string
				Verbose mode is turned on
				return=m_Client.SetVerboseMode(clientID, true)
				Verbose mode is turned off
				returnval=m_Client.SetVerboseMode(clientID, false)

out string clientID

Name	Type	Direction	Description
clientid	string	OUT	Identifier of the client that is connected to the server clientId = unique number + ipaddress of the client. For example, 1065–192.157.98.70

bool_verbose

Name	Type	Direction	Description
_verbose	bool	IN	Specifies whether the verbose mode should be turned ON or OFF

NOTE. *The Fail condition for PI commands occurs in any of the following cases:*

If the server is locked, the application displays "Server is locked by another client."

If the session is unlocked, the application displays "Lock session to execute the command."

If the server is not found, the application displays "Server not found-Disconnect!."

If the fail condition is not one of the above types, the application displays "Failed."

Set or get the DUT ID

Command name	Parameters	Description	Return value	Example
SetDutId()	string clientID string dutName	This method changes the DUT ID of the setup. The client must provide a valid DUT ID.	String that gives the status of the operation after it was performed Return value is “DUT Id Changed” on success	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string return=m_Client.SetDutId(clientID,desiredDutId) Note
GetDutId()	string clientID string dutId	This method gets the DUT ID of the current setup.	String that gives the status of the operation after it was performed	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string return=m_Client.GetDutid(clientID, out DutId)

out string clientID

Name	Type	Direction	Description
clientid	string	OUT	Identifier of the client that is connected to the server clientId = unique number + ipaddress of the client. For example, 1065–192.157.98.70

string dutName

Name	Type	Direction	Description
dutName	string	IN	The new DUT ID of the setup

string dutId

Name	Type	Direction	Description
dutId	string	OUT	The DUT ID of the setup. for example,

The dutId parameter is set after the server processes the request.

NOTE. *If the dutName parameter is null, the client is prompted to provide a valid DUT ID.*

NOTE. *The Fail condition for PI commands occurs in any of the following cases:*

If the server is locked, the application displays "Server is locked by another client."

If the session is unlocked, the application displays "Lock session to execute the command."

If the server is not found, the application displays "Server not found-Disconnect!."

If the fail condition is not one of the above types, the application displays "Failed."

Select the PCIe device

Syntax: mClient.SelectDevice(clientId, **device**, true);

Command name	Parameters	Description	Return value	Example
SelectDevice	clientID device	Selects the PCIe device.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string Select PCIe device example

out string clientID

Name	Type	Direction	Description
clientid	string	OUT	Identifier of the client that is connected to the server clientId = unique number + ipaddress of the client. For example, 1065–192.157.98.70

Select PCIe device example

mClient.SelectDevice("1065-192.157.98.70", **"CEM"**, true);

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U.2(SFF-8639)" or "BaseSpec".

Select the suite

Syntax: mClient.SelectSuite(clientId, device, **devicesuite**, true);

Command name	Parameters	Description	Return value	Example
SelectSuite	clientID device **devicesuite**	Sets the suite parameter.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string Select suite example

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suite, enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2(SFF-8639), valid values are **Module** and **Host**. For BaseSpec, valid value is **TX Test Board**.

Select suite example

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "**Add-In-Card**", true);

out string clientID

Name	Type	Direction	Description
clientid	string	OUT	Identifier of the client that is connected to the server clientId = unique number + ipaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U. 2(SFF-8639)".

Set the PCIe test version

Syntax: mClient.SelectVersion(clientId, device, devicesuite, **testversion**);

Command name	Parameters	Description	Return value	Example

SelectVersion	clientID
	device
	testversion
		Sets the PCIe compliance test suite.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as string
				Select test version example

string testversion

Name	Type	Direction	Description
testversion	string	IN	Specifies the version of the PCIe specification test suite. Valid values are **Gen1-1.0a**, **Gen1-1.1**, **Gen2-2.0,**, and **Gen3-3.0**. Declare each value as a single array variable

out string clientID

Name	Type	Direction	Description
clientid	string	OUT	Identifier of the client that is connected to the server clientId = unique number + ipaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U. 2 (SFF-8639)".

Set test version example

mClient.SelectVersions("1065-192.157.98.70", "CEM", "Add-In-Card", **Gen2–2.0**);

Set the data rate parameter

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example

SetGeneralParameter	clientID
	device
	devicesuite
	parameterString
		Sets the data rate parameter.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as string
				Set data rate example

out string clientID

Name	Type	Direction	Description
clientid	string	OUT	Identifier of the client that is connected to the server clientId = unique number + ipaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U. 2 (SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suite enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2 (SFF-8639), valid values are **Module** and **Host**.

string parameterString (for test rate)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter is the data rate. Valid values are **DataRate2Gb**, **DataRate5Gb**, and **DataRate8Gb**. The second parameter sets whether to include or exclude the data rate. Valid values are **Included** and **Excluded**. String example: "DataRate2Gb $Included".

Set data rate example

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**DataRate2Gb$Included**");

Set the test mode parameter

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

NOTE. *This parameter has to be set before setting the link analysis, sample rate, bandwidth, and record length parameters.*

Command name	Parameters	Description	Return value	Example

SetGeneralParameter	clientID
	device
	devicesuite
	parameterString
		Sets the Sigtest test mode parameter (compliance or use defined).	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as string
				Set test mode example

out string clientID

Name	Type	Direction	Description
clientid	string	OUT	Identifier of the client that is connected to the server clientId = unique number + ipaddress of the client. For example, 1065–192.157.98.70

string parameterString (for test mode)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter is the data rate. Valid values are **TestMode**. The second parameter sets the test mode. Valid values are **SigTest User Defined** and **SigTest Compliance**. String example: "TestMode$SigTest Compliance".

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U. 2 (SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suit, enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2 (SFF-8639), valid values are **Module** and **Host**.

Set test mode example

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**TestMode$SigTest Compliance**");

Set the prerecorded waveform execution mode

Syntax: mClient.SetPreRecorded(clientId, false | true, out error);

Command name	Parameters	Description	Return value	Example

SetPreRecorded	clientID
	device
	devicesuite
		Enables or disables using prerecorded (saved) waveforms for testing.
		NOTE. *There is no command in the PI to specify the prerecorded waveforms. Use this command to run a test session where you have already specified the waveforms files with the application user interface.*
			String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as string
				Set SetPreRecorded mode example

out string clientID

Name	Type	Direction	Description
clientid	string	OUT	Identifier of the client that is connected to the server clientId = unique number + ipaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U. 2 (SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suite, enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2 (SFF-8639), valid values are **Module** and **Host**.

Set SetPreRecorded mode example

mClient.SetPreRecorded(clientId, **false**, out error);

Set the 5 Gb/s preemphasis parameter

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example

SetGeneralParameter	clientID
	device
	devicesuite
	parameterString
		Sets the 5 Gb/s preemphasis parameter.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as string
				Select 5 Gb/s preemphasis example

out string clientID

Name	Type	Direction	Description
clientid	string	OUT	Identifier of the client that is connected to the server clientId = unique number + ipaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U.2(SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suite, enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2(SFF-8639), valid values are **Module** and **Host**.

string parameterString (for 5 Gb/s preemphasis)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter is the preemphasis value. Valid values are **PreEmphasis3dB** and **PreEmphasis6dB**. The second parameter sets whether to include or exclude the preemphasis value. Valid values are **Included** and **Excluded**. String example:"DataRate2Gb $Included".

Set 5 Gb/s preemphasis example

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**PreEmphasis3dB$Included**");

Set the SSC parameter

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example

SetGeneralParameter	clientID
	device
	devicesuite
	parameterString
		Sets the data rate.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as string
				Set SSC parameter example

out string clientID

Name	Type	Direction	Description
clientid	string	OUT	Identifier of the client that is connected to the server clientId = unique number + ipaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U. 2 (SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suite, enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2 (SFF-8639), valid values are **Module** and **Host**.

string parameterString (for SSC)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter is **SSC**. The second parameter sets whether to enable or disable SSC. Valid values are **On** or **Off**. String example:"SSC $Off".

Set SSC example

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**SSC$On**");

Set the voltage swing parameter

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example

SetGeneralParameter	clientID
	device
	devicesuite
	parameterString
		Sets the voltage swing parameter.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as string
				Set voltage swing parameter example

out string clientID

Name	Type	Direction	Description
clientid	string	OUT	Identifier of the client that is connected to the server clientId = unique number + ipaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U. 2 (SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suite, enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2 (SFF-8639), valid values are **Module** and **Host**.

string parameterString (for voltage swing)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter is **VoltageSwing**. The second parameter sets the voltage swing amount. Valid values are **Full** and **Reduced**. String example:"VoltageSwing $Reduced". This parameter affects the Signal Quality Preset tests: ■ **Full** selects all Signal quality Preset tests (P0–P10). ■ **Reduced** selects P01, P03 P04, P05, P06, and P09 (and in 5 Gbps only, 3.5 dB preemphasis is selected).

Set voltage swing example

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**VoltageSwing$Full**");

Set the signal quality preset parameter

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example

SetGeneralParameter	clientID
	device
	devicesuite
	parameterString
		Sets the presets for the signal quality test.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as string
				Set signal quality preset example

out string clientID

Name	Type	Direction	Description
clientid	string	OUT	Identifier of the client that is connected to the server clientId = unique number + ipaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U. 2 (SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suite, enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2 (SFF-8639), valid values are **Module** and **Host**.

string parameterString (for signal quality preset)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter is **SignalPreset**. The second parameter sets the preset(s) to enable. Enter the preset, followed by an underscore character for additional preset selections. String example:"SignalPreset $P0_P4_P5".

Set signal quality preset example

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**SignalPreset$P0_P01_P04_**");

To set all presets at one time:

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**SignalPreset$P0_P01_P02_P03_P04_P05_P06_P07_P08_P09_P10_**");

Set the preset lanes parameter

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example
SetGeneralParameter	clientID device devicesuite **parameterString**	Sets the lanes required to run the preset test. This should be equal to or a subset of the lanes selected by the SelectedLanes parameter or as set on the application DUT panel.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string Set preset lanes example

out string clientID

Name	Type	Direction	Description
clientid	string	OUT	Identifier of the client that is connected to the server clientId = unique number + ipaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U. 2 (SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suite, enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2 (SFF-8639), valid values are **Module** and **Host**.

string parameterString (for preset lanes)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter is **PresetLanes**. The second parameter sets the lane(s) to enable. Enter the lane, followed by an underscore character for additional lane selections. String example:"PresetLanes $Lane0_Lane1".

Set preset lanes example

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**PresetLanes$Lane0_Lane1_Lane4**");

If all lanes required, set link widtth to 16 and then:

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**PresetLanes $Lane0_Lane1_Lane2_Lane3_Lane4_Lane5_Lane6_Lane7_Lane8_Lane9_ Lane10_Lane11_Lane12_Lane13_Lane14_Lane15**");

Set the lane source parameter

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example
SetGeneralParameter	clientID device devicesuite **parameterString**	Sets the signal source and probing mode (single ended or differential) for each lane.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string Set lane source example

out string clientID

Name	Type	Direction	Description
clientid	string	OUT	Identifier of the client that is connected to the server clientId = unique number + ipaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U. 2 (SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suite, enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2 (SFF-8639), valid values are **Module** and **Host**.

string parameterString (for lane source)

Name	Type	Direction	Description
parameterString	string	IN	A string containing four parameters, separated by either a colon or $, enclosed in quotes. ■ The first parameter is **Lane<n> Connected to:**, where n is the lane number (0–15). ■ The second parameter is **Lane<n>:**, where n is the lane number (0–15) to which the first parameter is connected. ■ The third parameter is the probing mode. Valid values are **Differential**, **+ Single Ended**, and **– Single Ended**. ■ The fourth parameter, separated from the first three by a $ symbol is **CH<n>:**, where n is the channel number (1–4) to which the lane parameters are connected. String example:"Lane0 Connected to:Lane0:Differential $CH1".

Set lane source example

For differential probing:

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**Lane0 Connected to:Lane0:Differential$CH1**");

For single ended probing:

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**Lane0 Connected to:Lane0:+ Single Ended$CH1**");

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**Lane0 Connected to:Lane0:– Single Ended$CH1**");

Set the preset parameter

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example

SetGeneralParameter	clientID
	device
	devicesuite
	parameterString
		Sets the parameter for the Preset test.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as string
				Set preset example

out string clientID

Name	Type	Direction	Description
clientiD	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IPaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U. 2 (SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suite, enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2 (SFF-8639), valid values are **Module** and **Host**.

string parameterString (for preset)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter is **Preset**. The second parameter sets the preset(s) to enable. Enter the preset(s), followed by an underscore character. String example:"Preset $P0_P01_P02_P04_".

8 Gbps preset testing dependencies

Select the required dependent preset for 8 Gbps preset testing along with primary presets:

Table 14:

Evaluating preset	Dependent preset
P0	P04
P01	P04
P02	P04
P03	P04
P04	- -
P05	P04
P06	P04
P07	P02, P04, P05
P08	P03, P04, P06

Evaluating preset	Dependent preset
P09	P04
P10	P04

Set preset test example

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**Preset$P0_P02_P04_**");

To set all presets:

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**Preset$P0_P01_P02_P03_P04_P05_P06_P07_P08_P09_P10_**");

Set the acquisition parameter

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example

SetGeneralParameter	clientID
	device
	devicesuite
	parameterString
		Sets the acquisition mode.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as string
				Set acquisition example

out string clientID

Name	Type	Direction	Description
clientiD	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IPaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U. 2(SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suite, enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2(SFF-8639), valid values are **Module** and **Host**.

string parameterString (for acquisition)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter is **Acquisition**. The second parameter sets when acquisitions occur. Valid values are **BeforeAnalysis**, **InSequence**, and **AcquireOnly**. String example:"Acquisition $BeforeAnalysis".

Set acquisition example

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "**Acquisition$BeforeAnalysis**")

Set the analysis mode parameter

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example

SetGeneralParameter	clientID
	device
	devicesuite
	parameterString
		Sets the signal analysis mode (DLL or CLI).	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as string
				Set analysis mode example

out string clientID

Name	Type	Direction	Description
clientiD	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IPaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “PCIe”.

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the PCIe device suite, enclosed in quotes. Valid values are **System-Board** and **Add-In-Card**.

string parameterString (for analysis mode)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter is **AnalysisMode**. The second parameter sets the analysis mode. Valid values are **DLL** and **CLI**. String example:"AnalysisMode$DLL".

Set analysis mode example

mClient.SetGeneralParameter("1065-192.157.98.70", "PCIe", "System-Board", "", "**AnalysisMode$DLL**");

Set the sigtest version parameter

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example
SetGeneralParameter	clientID device devicesuite **parameterString**	Sets the Sigtest version or source file used for testing.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string Set Sigtest version example

out string clientID

Name	Type	Direction	Description
clientiD	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IPaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U. 2 (SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suite, enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2 (SFF-8639), valid values are **Module** and **Host**.

string parameterString (for Sigtest version)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter is either **SigTestVersion** (when AnalysisMode is set to DLL) or **SigTestPath** (when AnalysisMode is set to CLI) . The second parameter sets the SigTest version source. Valid values are the SigTest version installed on the instrument (when AnalysisMode is set to DLL), or the full path to the SigTest executable file (when AnalysisMode is set to CLI). String example: "SigTestVersion $3_2_0". "SigTestPath$C:\Program Files (x86)\SigTest 3.2.6\SigTest.exe".

Set Sigtest version example

For when AnalysisMode is set to DLL:

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**SigTestVersion$3_2_0**");

For when AnalysisMode is set to CLI:

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**SigTestPath$C:\Program Files (x86)\SigTest 3.2.6\SigTest.exe**");

Set the on failure action parameter

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example

SetGeneralParameter	clientID
	device
	devicesuite
	parameterString
		Sets the action taken when the application encounters a test failure.
		Note: Email settings must be entered in the application (with the user interface) before using this command.
			String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as string
				Set on failure action example

out string clientID

Name	Type	Direction	Description
clientiD	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IPaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U. 2 (SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suite, enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2 (SFF-8639), valid values are **Module** and **Host**.

string parameterString (for on failure action)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter sets the failure action. Valid values are **On Failure Stop and Notify** or **On Failure Pause**. The second parameter enables or disables the on failure action. Valid values are **True** and **False**. String example:"On Failure Stop and Notify $True".

Set on failure action example

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**On Failure Stop and Notify$False**");

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**On Failure Pause$True**");

Set the report update mode parameter

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example
SetGeneralParameter	clientID device devicesuite **parameterString**	Sets how the application saves the current test run report. **Note**: Changes to the test report update mode must be made before running a test session.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string Set report update example

out string clientID

Name	Type	Direction	Description
clientiD	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IPaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U. 2 (SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suite, enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2 (SFF-8639), valid values are **Module** and **Host**.

string parameterString (for report update mode)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter is **Report Update Mode**. The second parameter sets how the report is saved in relation to the last report from the session. Valid values are **Append**, **New**, and **Replace**. String example:"Report Update Mode $Append".

Set report update mode example

To create a new report for each test run:

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**AReport Update Mode$New**");

Set the append report parameter

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example
SetGeneralParameter	clientID device devicesuite **parameterString**	Enables or disables appending the current test session report to the previous test report.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string Set append report example

out string clientID

Name	Type	Direction	Description
clientiD	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IPaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “PCIe”.

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the PCIe device suite, enclosed in quotes. Valid values are **System-Board** and **Add-In-Card**.

string parameterString (for append report mode)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter is **Append Report**. The second parameter enables or disables appending the report. Valid values are **True** and **False**. String example:"Append Report$True".

Set append report mode example

mClient.SetGeneralParameter("1065-192.157.98.70", "PCIe", "System-Board", "", "**Append Report$False**");

Set the crosstalk parameter

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example
SetGeneralParameter	clientID device devicesuite **parameterString**	Sets the DUT crosstalk type (interleaved or noninterleved signal routing).	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string Set crosstalk mode example

out string clientID

Name	Type	Direction	Description
clientiD	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IPaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U. 2 (SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suite, enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2 (SFF-8639), valid values are **Module** and **Host**.

string parameterString (for crosstalk mode)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter is **CrossTalk**. The second parameter sets the crosstalk mode. Valid values are **On** and **Off**. String example:"CrossTalk $On.

Set crosstalk mode example

To set Crosstalk mode on (same as selecting the application control **Crosstalk (noninterleaved routing)**:

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**CrossTalk$On**");

To set Crosstalk mode off (same as selecting the application control **No Crosstalk (interleaved routing)**:

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**CrossTalk$Off**");

Set the waveform save parameter

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example
SetGeneralParameter	clientID device devicesuite **parameterString**	Sets the presets for the Preset test.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string Set waveform save parameter example

out string clientID

Name	Type	Direction	Description
clientiD	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IPaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U. 2 (SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suite, enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2 (SFF-8639), valid values are **Module** and **Host**.

string parameterString (for waveform save)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter is **SaveOptions**. The second parameter sets when acquisitions occur. Valid values are**Save All the Waveforms**,**Save Waveforms after applying Filters**,**No Waveforms saved - Discard after analysis**, and **Analyze Immediately - No Waveforms saved**. String example:"SaveOptions $Save All the Waveforms".

Set waveform save parameter example

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**SaveOptions$Save Waveforms after applying Filters**");

Set the link analysis embed-de-embed signal parameter

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example

SetGeneralParameter	clientID
	device
	devicesuite
	parameterString
		Embeds or de-embeds the 2.5, 5, and 8 Gb/s signals.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as string
				Set link analysis embed/de-embed signal example

out string clientID

Name	Type	Direction	Description
clientiD	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IPaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U. 2 (SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suite, enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2 (SFF-8639), valid values are **Module** and **Host**.

string parameterString (for embed/de-embed signal)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter values are **DeEmbed2Gb**, **DeEmbed5Gb**, **DeEmbed8Gb**, **Embed8Gb**, or **Equalization8Gb**. The second parameter sets whether to include or exclude the first parameter. Valid values are **Included** and **Excluded**. String example:"Embed8Gb $Included".

Set embed/de-embed signal example

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**Embed2Gb$Included**");

Set the link analysis embed-de-embed filter file parameter

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example

SetGeneralParameter	clientID
	device
	devicesuite
	parameterString
		Embeds or de-embeds the specified filter file.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as string
				Set link analysis embed/de-embed filter file example

out string clientID

Name	Type	Direction	Description
clientiD	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IPaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U. 2 (SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suite, enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2 (SFF-8639), valid values are **Module** and **Host**.

string parameterString (for embed/de-embed filter file)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter value is **Filterfile2Gb**, **Filterfile5Gb**, **FilterfileEmbed8Gb**, or **FilterfileDeEmbed8Gb**. The second parameter specifies the .flt filter file name to load. String example:"FilterFileEmbed8Gb$C:\PCI_Filters\Gen1.flt". PCIe does not include any filter files.

Set embed/de-embed filter file example

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**FilterFile5Gb$file.flt**");

Set the link analysis other filter file parameter

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example

SetGeneralParameter	clientID
	device
	devicesuite
	parameterString
		Loads the specified Sigtest or user-defined filter file.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as string
				Set load other filter file example

out string clientID

Name	Type	Direction	Description
clientiD	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IPaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U. 2 (SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suite, enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2 (SFF-8639), valid values are **Module** and **Host**.

string parameterString (for other filter file)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter value is **EmbedDropdown**. Valid second parameter values are **SigTest** and **Scope**. String example:"EmbedDropdown$SigTest".

Set load other filter file example

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**EmbedDropdown$Scope**");

Set the link analysis equalization dropdown parameter

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example

SetGeneralParameter	clientID
	device
	devicesuite
	parameterString
		Loads the specified Sigtest or user-defined filter file.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as string
				Set equalization dropdown example

out string clientID

Name	Type	Direction	Description
clientiD	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IPaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U. 2 (SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suite, enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2 (SFF-8639), valid values are **Module** and **Host**.

string parameterString (for equalization dropdown)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter value is **EqualizationDropdown**. Valid second parameter values are **Optimize** and **Fixed**. String example:"EqualizationDropdown$Optimize".

Set equalization dropdown example

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**EqualizationDropdown$Fixed**");

Set the link analysis CTLE index parameter

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example

SetGeneralParameter	clientID
	device
	devicesuite
	parameterString
		Sets the CTLE Index parameter.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as string
				Set CTLE index example

out string clientID

Name	Type	Direction	Description
clientiD	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IPaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U. 2 (SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suite, enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2 (SFF-8639), valid values are **Module** and **Host**.

string parameterString (for CTLE index)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter value is **CTLE Index**. Valid second parameter values are **1: –12db**,**2: –11db**, **3: –10db**, **4: –9db**, **5: –8db**, and **6: –7db**. String example:"CTLE Index$3 : -10dB".

Set CTLE index example

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**CTLE Index$5 : -8dB**");

Set the link analysis DFE parameter

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example
SetGeneralParameter	clientID device devicesuite **parameterString**	Sets the DFE value.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string Set DFE example

out string clientID

Name	Type	Direction	Description
clientiD	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IPaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U. 2 (SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suite, enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2 (SFF-8639), valid values are **Module** and **Host**.

string parameterString (for DFE)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter value is **DFE**. The second parameter specifies the DFE value, using standard scientific notation. String example:"DFE $30e-3".

Set DFE example

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**DFE$30e-3**");

Set the DUT auto toggle parameter

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example
SetGeneralParameter	clientID device devicesuite **parameterString**	Enables DUT Automation if AFG is connected.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string Set DUT auto toggle example

out string clientID

Name	Type	Direction	Description
clientiD	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IPaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U. 2 (SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suite, enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2 (SFF-8639), valid values are **Module** and **Host**.

string parameterString (for DUT auto toggle)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter value is **EnableDUTAutomation**. The second parameter values are **Included** and **Excluded**. String example:"EnableDUTAutomation$Included".

Set DUT auto toggle example

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**EnableDUTAutomation$Included**");

Set the DUT auto toggle options parameter

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example

SetGeneralParameter	clientID
	device
	devicesuite
	parameterString
		Sets DUT Automation settings preference.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as string
				Set DUT auto toggle options example

out string clientID

Name	Type	Direction	Description
clientiD	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IPaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U. 2 (SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suite, enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2 (SFF-8639), valid values are **Module** and **Host**.

string parameterString (for DUT auto toggle options)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter value is **Automation Settings**. The second parameter values are **Use Default Settings**, **Manually Configure Settings**, and **Use Custom Settings**. String example:"Automation Settings$Use Default Settings".

Set DUT auto toggle options example

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**Automation Settings$Use Custom Settings**");

Set AFG signal type parameter

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example

SetGeneralParameter	clientID
	device
	devicesuite
	parameterString
		Sets the signal type (square or sine wave) from AFG or AWG source.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as string
				Set signal source example

out string clientID

Name	Type	Direction	Description
clientiD	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IPaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U.2(SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suite, enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2(SFF-8639), valid values are **Module** and **Host**.

string parameterString (for AFG/AWG signal type)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter value is **Signal Type**. The second parameter values are **Square** and **Sine**. String example:"Signal Type$Sine".

Set AFG/AWG signal type example

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**Signal Type$Square**");

Set the AFG signal frequency parameter

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example
SetGeneralParameter	clientID device devicesuite **parameterString**	Specifies the AFG source signal frequency.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string Set AFG signal frequency example

out string clientID

Name	Type	Direction	Description
clientiD	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IPaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “PCIe”.

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the PCIe device suite, enclosed in quotes. Valid values are **System-Board** and **Add-In-Card**.

string parameterString (for AFG signal frequency)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter value is **AFGFrequency**. The second parameter specifies the frequency of the AFG source, using standard scientific notation. String example:"AFGFrequency$80e6".

Set AFG signal frequency example

mClient.SetGeneralParameter("1065-192.157.98.70", "PCIe", "System-Board", "", "**AFGFrequency$80e6**");

Set the AFG signal amplitude parameter

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example

SetGeneralParameter	clientID
	device
	devicesuite
	parameterString
		Specifies the AFG source signal amplitude.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as string
				Set AFG signal amplitude example

string parameterString (for AFG signal amplitude)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter value is **AFGAmplitude**. The second parameter specifies the amplitude of the AFG source, using standard scientific notation. String example:"AFGAmplitude$**80**".

out string clientID

Name	Type	Direction	Description
clientiD	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IPaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “PCIe”.

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the PCIe device suite, enclosed in quotes. Valid values are **System-Board** and **Add-In-Card**.

Set AFG signal amplitude example

mClient.SetGeneralParameter("1065-192.157.98.70", "PCIe", "System-Board", "", "**AFGAmplitude$60**");

Set the burst count parameter

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example

SetGeneralParameter	clientID
	device
	devicesuite
	parameterString
		Specifies the burst count parameters related to the AFG.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as string
				Set burst count example

out string clientID

Name	Type	Direction	Description
clientiD	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IPaddress of the client. For example, 1065–192.157.98.70

string parameterString (for burst count)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter value is **Burst Count**. The second parameter specifies the size of the burst count. String example:"Burst Count$**100k**".

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U. 2 (SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suite, enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2 (SFF-8639), valid values are **Module** and **Host**.

Set burst count example

mClient.SetGeneralParameter("1065-192.157.98.70", "PCIe", "System-Board", "", "**Burst Count$100k**");

Set the record length parameter

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example

SetGeneralParameter	clientID
	device
	devicesuite
	parameterString
		Sets the acquisition record length of the specified signal rate.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as string
				Set record length example

out string clientID

Name	Type	Direction	Description
clientiD	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IPaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U. 2 (SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suite, enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2 (SFF-8639), valid values are **Module** and **Host**.

string parameterString (for record length)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter value is **RecordLength2Gb**, **RecordLength5Gb**, or **RecordLength8Gb**. The second parameter sets the record length using standard scientific notation. String example:"RecordLength5Gb$10e6".

Set record length example

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**RecordLength2Gb$10e6**");

Set the sample rate parameter

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example
SetGeneralParameter	clientID device devicesuite **parameterString**	Sets the sampling rate of the specified signal type.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string Set sample rate example

out string clientID

Name	Type	Direction	Description
clientiD	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IPaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U. 2 (SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suite, enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2 (SFF-8639), valid values are **Module** and **Host**.

string parameterString (for sampling rate)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter value is **SampleRate2Gb**, **SampleRate5Gb**, or **SampleRate8Gb**. The second parameter sets the sample rate using standard scientific notation. String example:"SampleRate2Gb$25e9".

Set sample rate example

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**SampleRate8Gb$25e9**");

Set the bandwidth parameter

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example

SetGeneralParameter	clientID
	device
	devicesuite
	parameterString
		Sets the bandwidth of the specified signal type.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as string
				Set bandwidth example

out string clientID

Name	Type	Direction	Description
clientiD	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IPaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U. 2(SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suite, enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2 (SFF-8639), valid values are **Module** and **Host**.

string parameterString (for bandwidth)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter value is **Bandwidth2Gb**, **Bandwidth5Gb**, or **Bandwidth8Gb**. The second parameter bandwidth frequency using standard scientific notation. String example:"Bandwidth2Gb$6e9".

Set bandwidth example

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**Bandwidth8Gb$6e9**");

Set the signal validation parameter

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example

SetGeneralParameter	clientID
	device
	devicesuite
	parameterString
		Sets the signal check (validation) action.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as string
				Set signal validation example

out string clientID

Name	Type	Direction	Description
clientiD	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IPaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U. 2 (SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suite, enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2 (SFF-8639), valid values are **Module** and **Host**.

string parameterString (for signal validation)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter value is **Signal Validation**. The second parameter sets the signal check (validation) action. Valid values are **Prompt me if Signal Check Fails**, **Skip Test if Signal Check Fail**s, or **Turn Off Signal Check**. String example:"Signal Validation$Prompt me if Signal Check Fails".

Set signal validation example

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**Signal Validation$Turn Off Signal Check**");

Set the slot number parameter

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example

SetGeneralParameter	clientID
	device
	devicesuite
	parameterString
		Sets the test slot number.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as string
				Set slot number example

out string clientID

Name	Type	Direction	Description
clientiD	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IPaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U. 2 (SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suite, enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2 (SFF-8639), valid values are **Module** and **Host**.

string parameterString (for select slot number)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter value is **SlotNumber**. The second parameter specifies the slot number. Valid values are **01** through **08**. String example:"SlotNumber $02".

Set slot number example

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**SlotNumber$05**");

Set SigTest Interface Mode

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example
SetGeneralParameter	clientID device devicesuite **parameterString**	Sets the test slot number.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string Set SigTest Interface mode example

out string clientID

Name	Type	Direction	Description
clientiD	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IPaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U.2(SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suite, enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2(SFF-8639), valid values are **Module** and **Host**.

string parameterString (to set SigTest interface mode)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter value is **AnalysisMode**. The second parameter specifies the slot number. Valid values are **CLI** through **DLL**. String example:"AnalysisMode$CLI".

Set SigTest interface mode example

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board","**AnalysisMode$CLI**");

Set the trigger type parameters

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example

SetGeneralParameter	clientID
	device
	devicesuite
	parameterString
		Sets the acquisition record length of the specified signal rate.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as string
				Set record length example

out string clientID

Name	Type	Direction	Description
clientiD	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IPaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U. 2 (SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suite, enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2 (SFF-8639), valid values are **Module** and **Host**.

string parameterString (for trigger type)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter value is **Trigger Type**. The second parameter is **Auto**, **Edge**, or **Width**. sets the trigger type using standard scientific notation. String example:"Trigger Type$Auto".

Set trigger type example

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**Trigger Type$Auto**");

NOTE. *Trigger Type is applicable for Version Gen3 (3.0) only.*

Set Sig Validation Threshold

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example
SetGeneralParameter	clientID device devicesuite **parameterString**	Sets the test slot number.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string Set Sig Validation Threshold example

out string clientID

Name	Type	Direction	Description
clientiD	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IPaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U.2(SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suite, enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2(SFF-8639), valid values are **Module** and **Host**.

string parameterString (to set signal validation threshold)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter value is **Signal Validation Threshold(mV)**. The second parameter specifies the slot number. Valid values are **50** through **400**. String example:"Signal Validation Threshold(mV)$325".

Set signal validation threshold example

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board","**Signal Validation Threshold(mV)$325**");

Set Toggle from Gen3P10 to Gen1

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example
SetGeneralParameter	clientID device devicesuite **parameterString**	Sets the test slot number.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string Set Toggle from Gen3P10 to Gen1 example

out string clientID

Name	Type	Direction	Description
clientiD	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IPaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “PCIe”.

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the PCIe device suite, enclosed in quotes. Valid values are **System-Board** and **Add-In-Card**.

string parameterString (for select slot number)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter value is **Toggle from Gen3P10 to Gen1**. The second parameter specifies the slot number. Valid values are **On** through **Off**. String example:"Toggle from Gen3P10 to Gen1$Off".

Set Toggle from Gen3P10 to Gen1

mClient.SetGeneralParameter("1065-192.157.98.70", "PCIe", "System-Board","**Toggle from Gen3P10 to Gen1$Off**");

Set the group test results by parameters

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example

SetGeneralParameter	clientID
	device
	devicesuite
	parameterString
		Sets the acquisition record length of the specified signal rate.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as string
				Set record length example

out string clientID

Name	Type	Direction	Description
clientiD	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IPaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U. 2 (SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suite, enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2 (SFF-8639), valid values are **Module** and **Host**.

string parameterString (for group test results)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter value is **Report Group Mode**. The second parameter is **Test Name**, **Lane Name**, **Equalization**or **Test Result**. sets the group test results using standard scientific notation. String example:"Report Group Mode$Test Result".

Set group test results example

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**Report Group Mode$Test Result**");

Set the report creation path parameter

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example
SetGeneralParameter	clientID device devicesuite **parameterString**	Sets the acquisition record length of the specified signal rate.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string Set record length example

out string clientID

Name	Type	Direction	Description
clientiD	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IPaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U. 2 (SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suite, enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2 (SFF-8639), valid values are **Module** and **Host**.

string parameterString (for report creation path)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter value is **Report Path**. The second parameter is **X:\PCI Express \Reports\DUT001.pdf**. sets the record length using standard scientific notation. String example:"Report Path$X:\PCI Express \Reports\DUT001.pdf".

Set report creation path example

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**Report Path$X:\PCI Express\Reports\DUT001.pdf**");

Set the report contents to save parameters

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example
SetGeneralParameter	clientID device devicesuite **parameterString**	Sets the acquisition record length of the specified signal rate.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string Set record length example

out string clientID

Name	Type	Direction	Description
clientiD	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IPaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U. 2 (SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suite, enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2 (SFF-8639), valid values are **Module** and **Host**.

string parameterString (for report contents to save)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter value are: ■ **Include Pass/Fail Results Summary** ■ **Include Detailed Results** ■ **Include Plot Images** ■ **Include Setup Configuration** ■ **Include User Comment** . The second parameter is **True**, or **False**. sets the report contents to save using standard scientific notation. String example:"Include Pass/Fail Results Summary$True".

Set report contents to save example

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**Include Detailed Results$True**");

Set the report creation type parameter

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example

SetGeneralParameter	clientID
	device
	devicesuite
	parameterString
		Sets the acquisition record length of the specified signal rate.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as string
				Set record length example

out string clientID

Name	Type	Direction	Description
clientiD	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IPaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U. 2 (SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suite, enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2 (SFF-8639), valid values are **Module** and **Host**.

string parameterString (for report creation path)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter value is **Save As Type**. The second parameter is **PDF** or **Web Archive**. sets the record length using standard scientific notation. String example:"Save As Type$Web Archive".

Set report creation path example

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**Save As Type$Web Archive**");

Set the auto increment report name if duplicate parameter

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example
SetGeneralParameter	clientID device devicesuite **parameterString**	Sets the acquisition record length of the specified signal rate.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string Set record length example

out string clientID

Name	Type	Direction	Description
clientiD	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IPaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U. 2 (SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suite, enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2 (SFF-8639), valid values are **Module** and **Host**.

string parameterString (for report creation path)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter value is **Auto increment report name if duplicate**. The second parameter is **True** or **False**. sets the record length using standard scientific notation. String example:"Auto increment report name if duplicate$True".

Set report creation path example

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**Auto increment report name if duplicate$True**");

Set the view report after generating parameter

Syntax: mClient.SetGeneralParameter(clientId, device, devicesuite, "", **parameterString**);

Command name	Parameters	Description	Return value	Example

SetGeneralParameter	clientID
	device
	devicesuite
	parameterString
		Sets the acquisition record length of the specified signal rate.	String that gives the status of the operation after it was performed. The return value is "" (an empty String) on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as string
				Set record length example

out string clientID

Name	Type	Direction	Description
clientiD	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IPaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U. 2 (SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the device suite, enclosed in quotes. For CEM, valid values are **System-Board** and **Add-In-Card**. For U.2 (SFF-8639), valid values are **Module** and **Host**.

string parameterString (for view report after generating)

Name	Type	Direction	Description
parameterString	string	IN	A string containing two parameters separated by a $ symbol, enclosed in quotes. The first parameter value is **View Report After Generating**. The second parameter is **True**, or **False**. sets the record length using standard scientific notation. String example:"View Report After Generating$True".

Set record length example

mClient.SetGeneralParameter("1065-192.157.98.70", "CEM", "System-Board", "", "**View Report After Generating$True**");

Run with set configurations or stop the run operation

Command name	Parameters	Description	Return value	Example
Run()	string clientID	Runs the selected tests Note After the server is set up and configured, run it remotely using this function.	String that gives the status of the operation after it was performed. The return value is "Run started..." on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string returnval=m_Client.Run(clientID)
Stop()	string clientID	Stops the running tests. Note	String that gives the status of the operation after it was performed The return value is "Stopped..." on success.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string returnval=m_Client.Stop(clientID)

out string clientID

Name	Type	Direction	Description
clientiD	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IPaddress of the client. For example, 1065–192.157.98.70

NOTE. *When the run is performed, the status of the run is updated periodically using a timer.*

NOTE. *When the session is stopped, the client is prompted to stop the session and is stopped at the consent.*

NOTE. *The Fail condition for PI commands occurs in any of the following cases:*

If the server is locked, the application displays "Server is locked by another client."

If the session is unlocked, the application displays "Lock session to execute the command."

If the server is not found, the application displays "Server not found-Disconnect!."

If the fail condition is not one of the above types, the application displays "Failed."

Handle error codes

The return value of the remote automations at the server-end is OP_STATUS, which changes to a string value depending on its code, and returned to the client. The values of OP_STATUS are as follows:

Code	Value	Description
-1	FAIL	The operation failed
1	SUCCESS	The operation succeeded
2	NOT FOUND	Server not found
3	LOCKED	The server is locked by another client, so the operation cannot be performed

Code	Value	Description
4	UNLOCK	The server is not locked; lock the server before performing the operation
0	NULL	Nothing

Save recall or query a saved session

Command name	Parameters	Description	Return value	Example

CheckSessionSaved()	string clientID
	out bool saved
		This method checks whether the current session is saved.	Return value is either True or False	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as string
				returnval=m_Client.CheckSessionSaved(m_clientID, out savedStatus)

RecallSession()	string clientID
	string name
		Recalls a saved session. The client provides the session name.	String that gives the status of the operation after it was performed
			The return value is “Session Recalled...”
				m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as string
				returnval=m_Client.RecallSession(clientID, savedSessionName)

SaveSession()	string clientID
	string name
		Saves the current session. The client provides the session name.	String that gives the status of the operation after it was performed
			The return value is "Session Saved..."/"Failed..."
				m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as string
				returnval=m_Client.SaveSession(clientID, desiredSessionName)

SaveSessionAs()	string clientID
	string name
		Saves the current session under a different name every time this method is called. The client provides the session name.	String that gives the status of the operation after it was performed
			The return value is "Session Saved..."
				m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as string
				returnval=m_Client.SaveSessionAs(clientID, desiredSessionName)

out string clientID

Name	Type	Direction	Description
clientid	string	OUT	Identifier of the client that is connected to the server clientId = unique number + ipaddress of the client. For example, 1065–192.157.98.70

out bool saved

Name	Type	Direction	Description
saved	bool	OUT	Boolean representing whether the current session is saved

This parameter is used as a check in SaveSession() and SaveSessionAs() functions.

string name

Name	Type	Direction	Description
name	string	IN	The name of the session being recalled

The name parameter cannot be empty. If it is empty, the client is prompted to provide a valid name.

NOTE. *The Fail condition for PI commands occurs in any of the following cases:*

If the server is locked, the application displays "Server is locked by another client."

If the session is unlocked, the application displays "Lock session to execute the command."

If the server is not found, the application displays "Server not found-Disconnect!."

If the fail condition is not one of the above types, the application displays "Failed."

Get or set the timeout value

Command name	Parameters	Description	Return value	Example
GetTimeOut()	string clientID	Returns the current timeout period set by the client	String that gives the status of the operation after it was performed The default return value is 1800000.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string returnval=m_Client.GetTimeOut()
SetTimeOut()	string clientID string time	Sets a timeout period specified by the client. After this timeout period expires, the server is unlocked automatically.	String that gives the status of the operation after it was performed On success the return value is “TimeOut Period Changed”.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string returnval=m_Client.SetTimeOut(clientID, desiredTimeOut)

out string clientID

Name	Type	Direction	Description
clientiD	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IPaddress of the client. For example, 1065–192.157.98.70

string time

Name	Type	Direction	Description
time	string	IN	The time in seconds that refers to the timeout period

The time parameter gives the timeout period, which is the time the client is allowed to be locked and idle. After the timeout period if the client is still idle, it gets unlocked.

The time parameter should be a positive integer; otherwise, the client is prompted to provide a valid timeout period.

NOTE. *The Fail condition for PI commands occurs in any of the following cases:*

If the server is locked, the application displays "Server is locked by another client."

If the session is unlocked, the application displays "Lock session to execute the command."

If the server is not found, the application displays "Server not found-Disconnect!."

If the fail condition is not one of the above types, the application displays "Failed."

Wait for the test to complete

The commands in this group execute while tests are running. The GetCurrentStateInfo() and SendResponse() commands are executed when the application is running and in the wait state.

Command name	Parameters	Description	Return value	Example
ApplicationStatus()	string clientID	This method gets the status of the server application. The states are Running, Paused, Wait, and Error	String value that gives the status of the server application	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string returnval=m_Client.ApplicationStatus(clientID)
QueryStatus()	string clientID out string[] status	An interface for the user to transfer Analyze panel status messages from the server to the client	String that gives the status of the operation after it was performed On success the return value is “Transferred...”.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string Query status example
GetCurrentStateInfo() ***NOTE.*** *This command is used when the application is running and is in the wait or error state.*	string clientID out string WaitingMsbBxCaption out string WaitingMsbBxMessage out string[] WaitingMsbBxButtontexts	This method gets the additional information of the states when the application is in Wait or Error state. Except client ID, all the others are Out parameters. If the application is in Wait state, and the caption and button texts are retrieved, if the caption is "Unable to Trigger," then Cancel is an expected response. Note: If cancel is sent to skip the acquisition, then the string must have a space at the end (example: “Cancel “). Otherwise OK is considered as the response.	This command does not return any value. This function populates the Out parameters that are passed when invoking this function.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL mClient.GetCurrentStateInfo(clientID, WaitingMsbBxCaption, WaitingMsbBxMessage, WaitingMsbBxButtontexts)

Command name	Parameters	Description	Return value	Example

SendResponse()
NOTE. *This command is used when the application is running and is in the wait or error state.*
	string clientID
	out string WaitingMsbBxCaption
	out string WaitingMsbBxMessage
	string WaitingMsbBxResponse
		After receiving the additional information using the method GetCurrentStateInfo(), the client can decide which response to send and then send the response to the application using this function. The response should be one of the strings that was received earlier as a string array in the GetCurrentStateInfo function. The _caption and _message should match the information received earlier in the GetCurrentStateInfo function.	This command does not return any value.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL
				mClient.SendResponse(clientID, WaitingMsbBxCaption, WaitingMsbBxMessage, WaitingMsbBxResponse)

out string clientID

Name	Type	Direction	Description
clientid	string	OUT	Identifier of the client that is connected to the server clientId = unique number + ipaddress of the client. For example, 1065–192.157.98.70

out string[] status

Name	Type	Direction	Description
status	string array	OUT	The list of status messages generated during the run

out string WaitingMsbBxCaption

Name	Type	Direction	Description
caption	string	OUT	The wait state or error state message sent to you

out string WaitingMsbBxMessage

Name	Type	Direction	Description
message	string	OUT	The wait state/error state message sent to you

out string[] WaitingMsbBxButtontexts

Name	Type	Direction	Description
buttonTexts	string array	OUT	An array of strings containing the possible response types that you can send

string WaitingMsbBxResponse

Name	Type	Direction	Description
response	string	IN	A string containing the response type that you can select (it must be one of the strings in the string array buttonTexts)

Ready: Test configured and ready to start

Running: Test running

Paused: Test paused

Wait: A popup that needs your inputs

Error: An error is occurred

NOTE. *The Fail condition for PI commands occurs in any of the following cases:*

If the server is locked, the application displays "Server is locked by another client."

If the session is unlocked, the application displays "Lock session to execute the command."

If the server is not found, the application displays "Server not found-Disconnect!."

If the fail condition is not one of the above types, the application displays "Failed."

Query status example

```
returnVal=m_Client.QueryStatus(clientID, out statusMessages)
if ((OP_STATUS)returnVal == OP_STATUS.SUCCESS)
return “Status updated...”
else
return CommandFailed(returnVal)
```

After the test is complete

Command name	Parameters	Description	Return value	Example

GetPassFailStatus()	string clientID
	string device
	string suite
	string test
		This method gets the pass or fail status of the measurement after test completion.
		NOTE. *Execute this command after completing the measurement.*
			String that gives the status of the operation after it was performed Returns the pass or fail status in the form of a string	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as string
				returnval=m_Client.GetPassFailStatus(clientID, device, devicesuite, test)
				Get pass/fail status for a measurement example

GetResultsValue()	string clientID
	string device
	string suite
	string test
	string parameterString
		This method gets the result values of the measurement after the run.	String that gives the status of the operation after it was performed Returns the result value in the form of a string	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as srting
				returnval=m_Client.GetResultsValue(clientID, device, devicesuite, test, parameterString)
				Get pass/fail status for a measurement example

	To get all results of a test, first get the value by passing the column name, for example 'Value,' and then pass true value for the last argument. This command will return a value string which contains values for all of the test details in a comma-separated format. See example at end of table.			

Command name	Parameters	Description	Return value	Example

GetReportParameter()	string clientID
	string device
	string suite
	string test
	string parameterString
		This method gets the general report details such as oscilloscope model and TekExpress version.	The return value is the oscilloscope model, TekExpress application version, or PCIe application version.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as string
				Oscilloscope Model
				returnval=m_Client.GetReportParameter(clientID,"Scope Model")
				TekExpress Version
				returnval=m_Client.GetReportParameter(clientID,"TekExpress Version")
				PCIe Version
				returnval=m_Client.GetReportParameter(clientID,"Application Version")

Command name	Parameters	Description	Return value	Example

TransferResult()
NOTE. *The target folder must have write permission when transferring the results and images from the server machine to the client machine. Otherwise the transfer will fail.*
	string clientID
	string filePath
		This method transfers the report generated after the run.
		The report contains the summary of the run.
		The client must provide the location where the report is to be saved at the client-end.
			String that gives the status of the operation after it was performed. Transfers all the result values in the form of a string.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as string
				returnval=m_Client.TransferReport(clientID,"C:\Report")

TransferImages()
NOTE. *The target folder must have write permission when transferring the results and images from the server machine to the client machine. Otherwise the transfer will fail.*
	string clientID
	string filePath
		This method transfers all the images (screen shots) from the specified client and folder for the current run (for a suite or measurement).
		NOTE. *Every time you click Start, a folder is created in the X: drive. Transfer the waveforms before clicking Start.*
			String that gives the status of the operation after it was performed. Transfers all the images in the form of a string.	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as string
				returnval=m_Client.TransferImages(clientID, "C:\Waveforms")

getResultsValue example:

```
string values = mClient.GetResultsValue(clientId, device, devicesuite, test, "Details", true);

  string[] allDetailsList = values.Split(',');

  values = mClient.GetResultsValue(clientId, device, devicesuite, test, "Value", true);

  string[] allValuesList = values.Split(',');

  for(int index=0;index<allDetailsList.Length;index++)

    Console.WriteLine("Value for "+allDetailsList[index].ToString()+ " = "+allValuesList[index].ToString

());
```

NOTE. *The Fail condition for PI commands occurs in any of the following cases:*

If the server is locked, the application displays "Server is locked by another client."

If the session is unlocked, the application displays "Lock session to execute the command."

If the server is not found, the application displays "Server not found-Disconnect!."

If the fail condition is not one of the above types, the application displays "Failed."

out string clientID

Name	Type	Direction	Description
clientid	string	OUT	Identifier of the client that is connected to the server clientId = unique number + ipaddress of the client. For example, 1065–192.157.98.70

string device

Name	Type	Direction	Description
device	string	IN	Specifies the name of the device. For PCIe testing, set this to “CEM” or "U. 2 (SFF-8639)".

string devicesuite

Name	Type	Direction	Description
devicesuite	string	IN	Specifies the name of the PCIe device suite, enclosed in quotes. For "CEM", valid values are **System-Board** and **Add-In-Card**. For "U.2 (SFF-8639)", valid values are**Module** and **Host**.

string test

Name	Type	Direction	Description
test	string	IN	Specifies the name of the test to obtain the pass or fail status or a test result value. Append spaces at the end of the test name to differentiate the PCIe DUT generation version for which to return measurements: Gen1: no spaces Gen2: One space Gen3: Two spaces **Examples:** Gen1: “Unit Interval” Gen2: “Unit Interval ” Gen3: “Unit Interval ”

string filePath

Name	Type	Direction	Description
filePath	string	IN	The location where the report must be saved in the client

NOTE. *If the client does not provide the location to save the report, the report is saved at C:\ProgramFiles.*

string parameterString

Name	Type	Direction	Description
parameterString	string	IN	Specifies the oscilloscope model, TekExpress version, or application version

string parameterString "Value"

Name	Type	Direction	Description
parameterString	string	IN	Specifies to return the measured value for the indicated test. Enter “Value” for this argument

Get pass/fail status for a measurement example

This example returns the pass/fail status for the **Gen1** transition eye diagram measurement:

returnval=m_Client.GetPassFailStatus(clientId, device, devicesuite, "Transition Eye Diagram")

This example returns the results for the **Gen3** unit interval measurement:

returnval=m_Client.GetPassFailStatus(clientId, device, devicesuite, "Unit Interval ")

Note the two blank spaces between the end of the measurement name and the closing quote for that parameter. The test parameter uses blank spaces at the end of the test name to differentiate the PCIe DUT generation version for which to return measurements:

Gen1: no spaces

Gen2: One space

Gen3: Two spaces

Examples:

Gen1: “Unit Interval”

Gen2: “Unit Interval ”

Gen3: “Unit Interval ”

Unlock the server

Command name	Parameters	Description	Return value	Example
UnlockSession()	string clientID	This method unlocks the server from the client. The ID of the client to be unlocked must be provided. Note	String that gives the status of the operation after it was performed The return value is "Session UnLocked..."	m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string returnval=m_Client.UnlockServer(clientID)

NOTE. *When the client is disconnected, the client is unlocked automatically.*

out string clientID

Name	Type	Direction	Description
clientID	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IP address of the client. For example, 1065–192.157.98.70

NOTE. *The Fail condition for PI commands occurs in any of the following cases:*

If the server is locked, the application displays "Server is locked by another client."

If the session is unlocked, the application displays "Lock session to execute the command."

If the server is not found, the application displays "Server not found-Disconnect!."

If the fail condition is not one of the above types, the application displays "Failed."

Disconnect from the server

Command name	Parameters	Description	Return value	Example

Disconnect()	string clientID	This method disconnects the client from the server. Note	Integer value that gives the status of the operation after it was performed
			1 for Success
			–1 for Failure
				m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as string
				returnval=m_Client.Disconnect(m_clientID)

NOTE. *When the client is disconnected, it is unlocked from the server and then disconnected. The Id is reused.*

out string clientID

Name	Type	Direction	Description
clientID	string	OUT	Identifier of the client that is connected to the server clientID = unique number + IP address of the client. For example, 1065–192.157.98.70

NOTE. *The Fail condition for PI commands occurs in any of the following cases:*

If the server is locked, the application displays "Server is locked by another client."

If the session is unlocked, the application displays "Lock session to execute the command."

If the server is not found, the application displays "Server not found-Disconnect!."

If the fail condition is not one of the above types, the application displays "Failed."

SCPI commands

About SCPI command

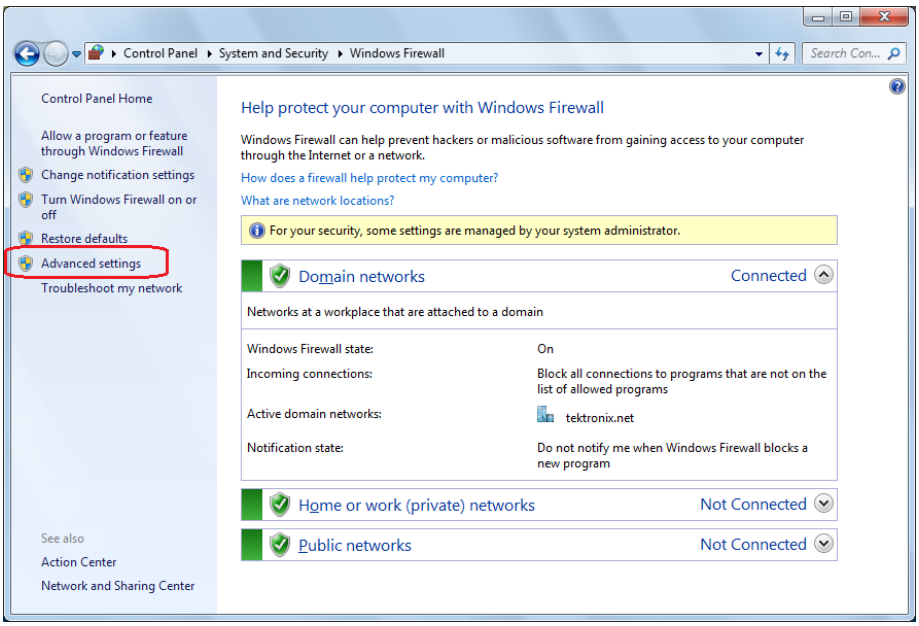
You can use Standard Commands for Programmable Instruments (SCPI) to communicate with the TekExpress application.

Socket configuration for SCPI commands

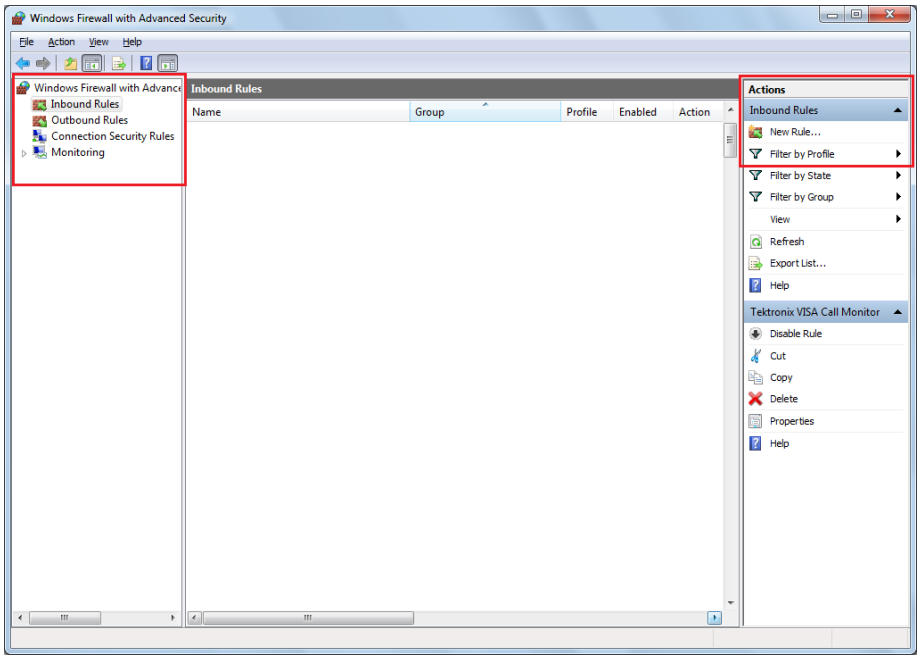
This section describes the steps for TCPIP socket configuration and TekVISA configuration to execute the SCPI commands.

TCPIP socket configuration

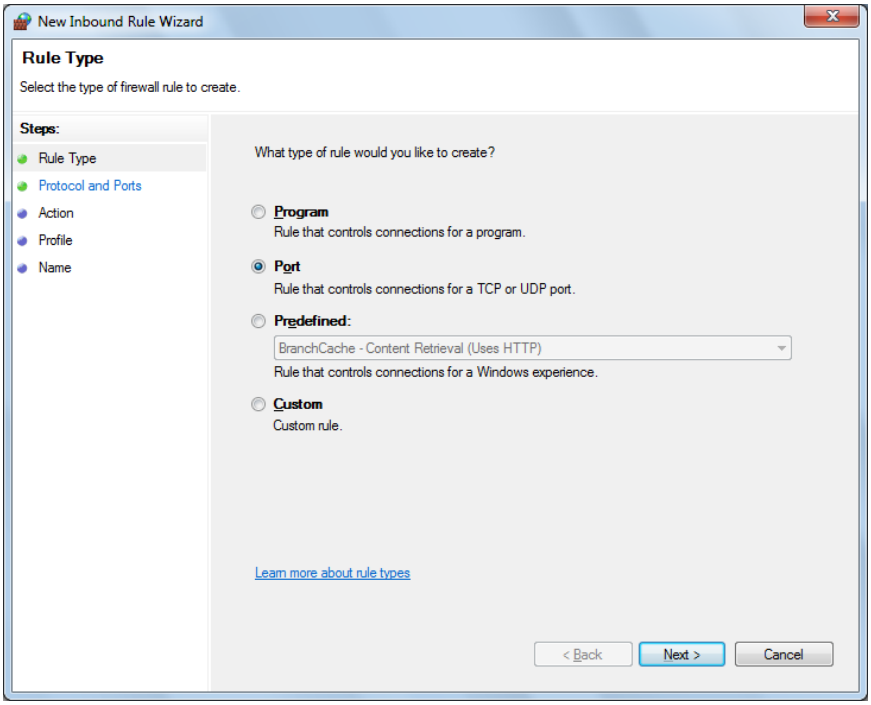
1. Click **Start** > **Control Panel** > **System and Security** > **Windows Firewall** > **Advanced settings**



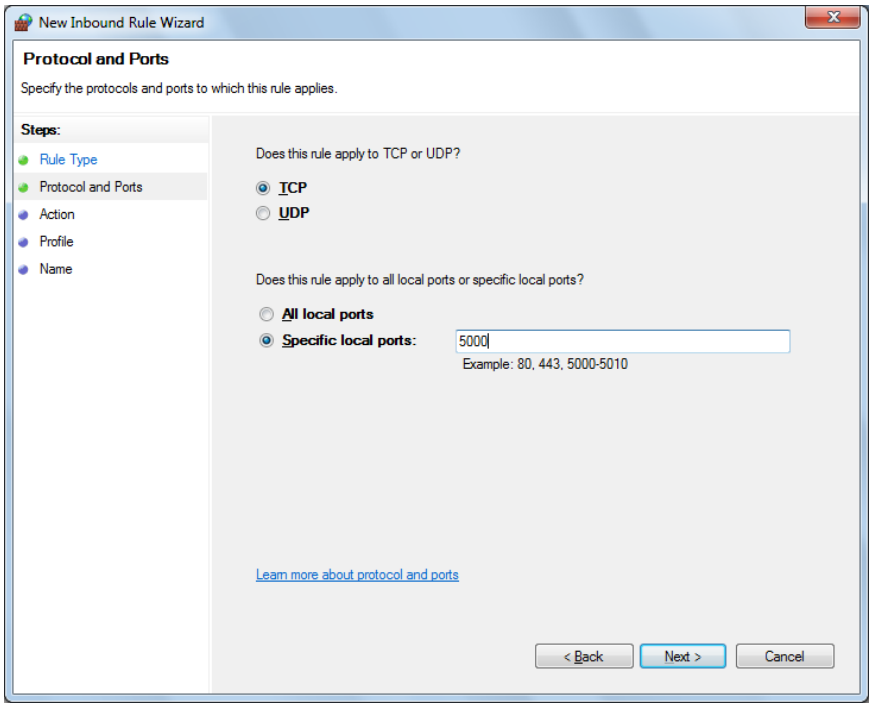
2. In Windows Firewall with Advanced Security menu, select **Windows Firewall with Advanced Security on Local Computer** > **Inbound Rules** and click New Rule…



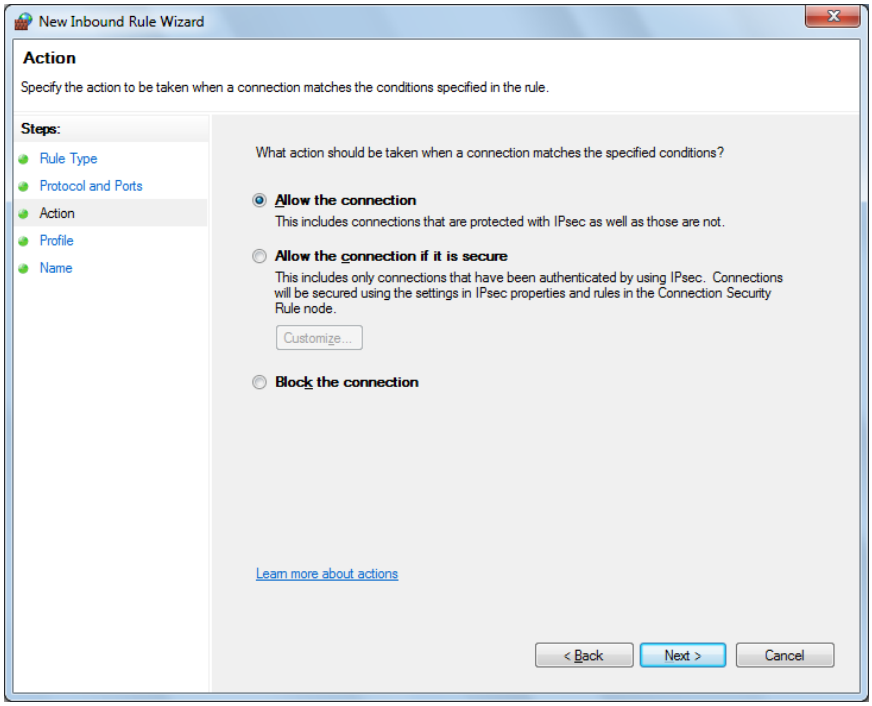
3. In New Inbound Rule Wizard menu
   a. Select **Port** and click **Next**



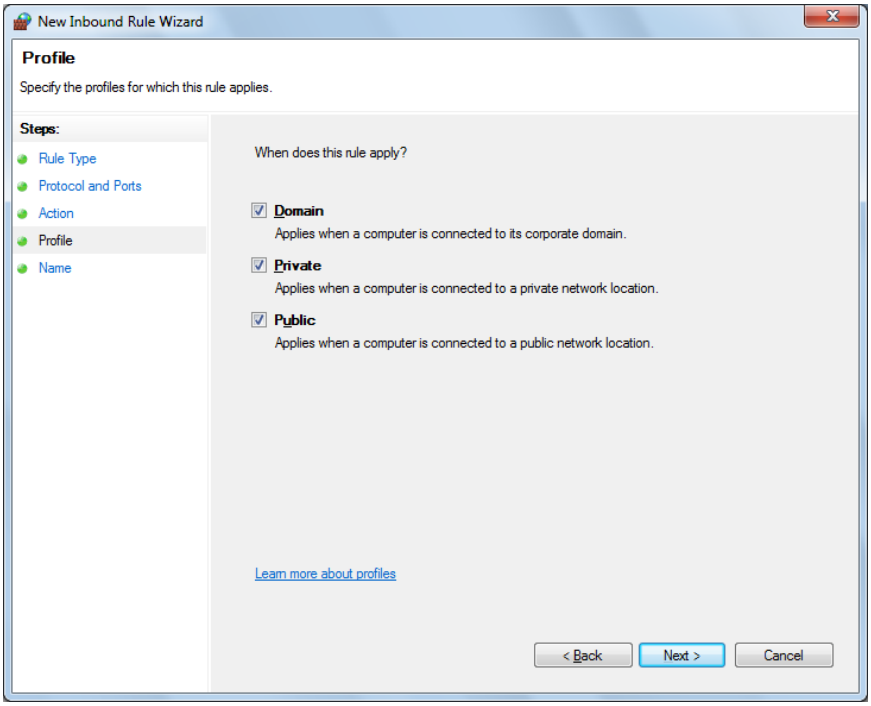
b. Select **TCP** as rule apply and enter 5000 for **Specific local ports** and click **Next**



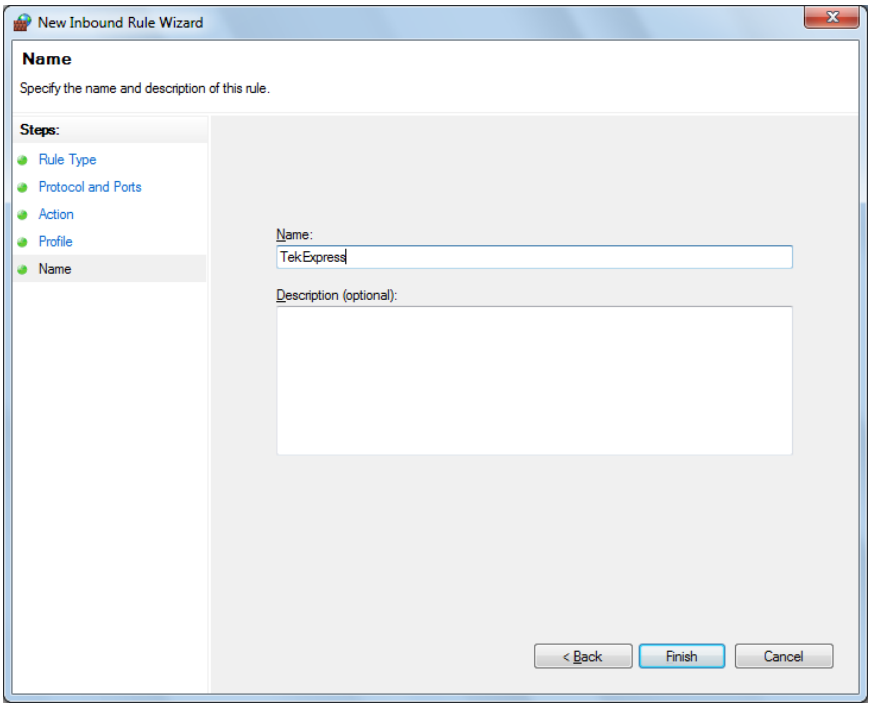
c. Select **Allow the connection** and click **Next**



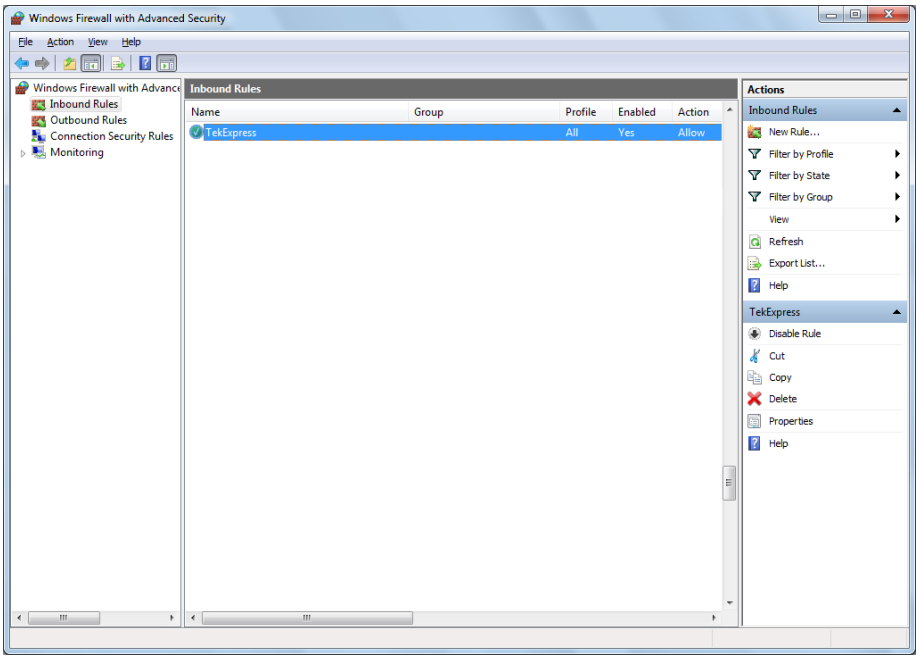
d. Select **Domain, Private, Public** and click **Next**



e. Enter **Name**, Description (optional), and click **Finish**

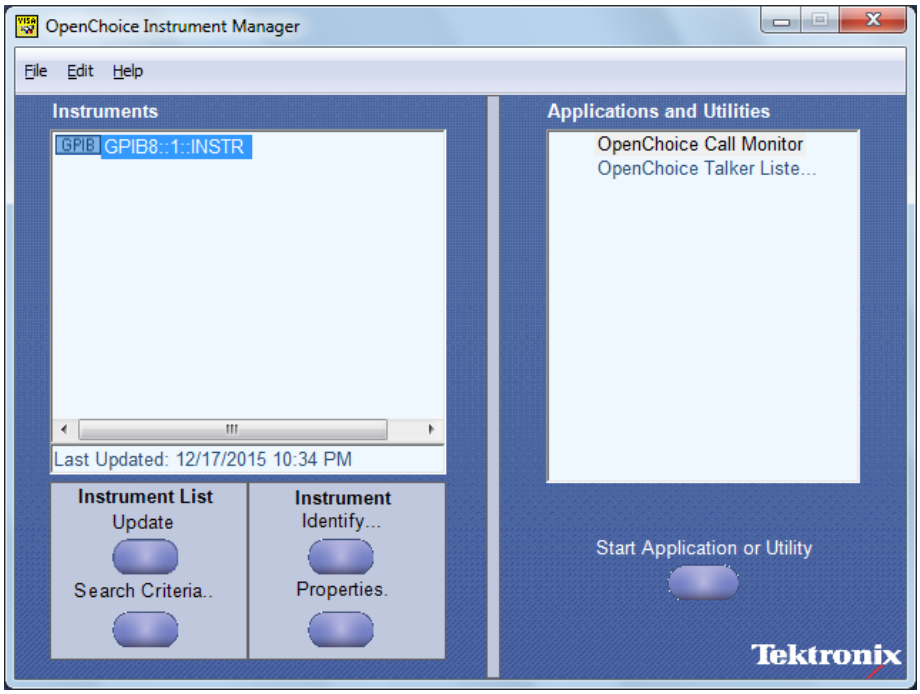


4. Check whether the Rule name is displayed in **Windows Firewall with Advanced Security menu** > **Inbound Rules**



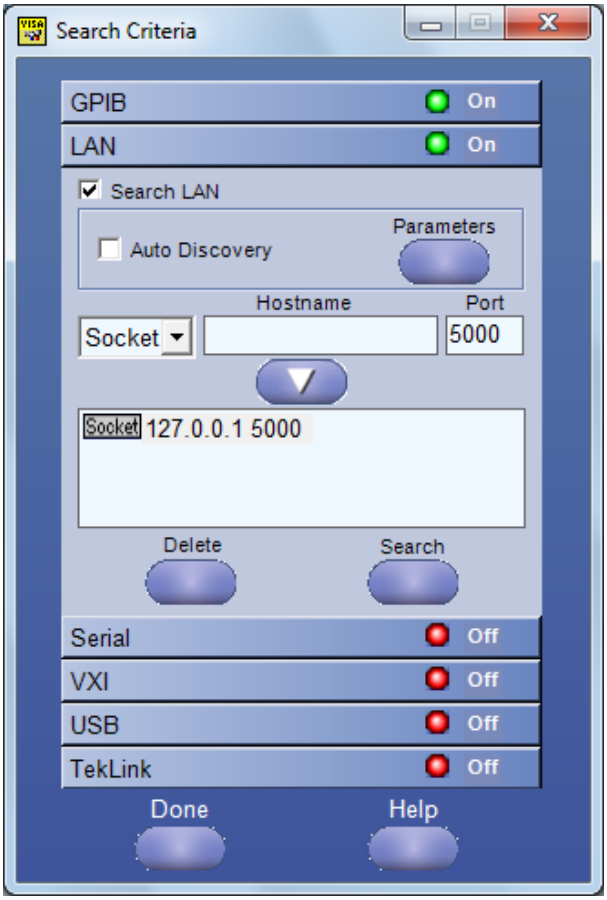
TekVISA configuration

1. Click **Start** > **All Programs** > **TekVISA** > **OpenChoice Instrument Manager**

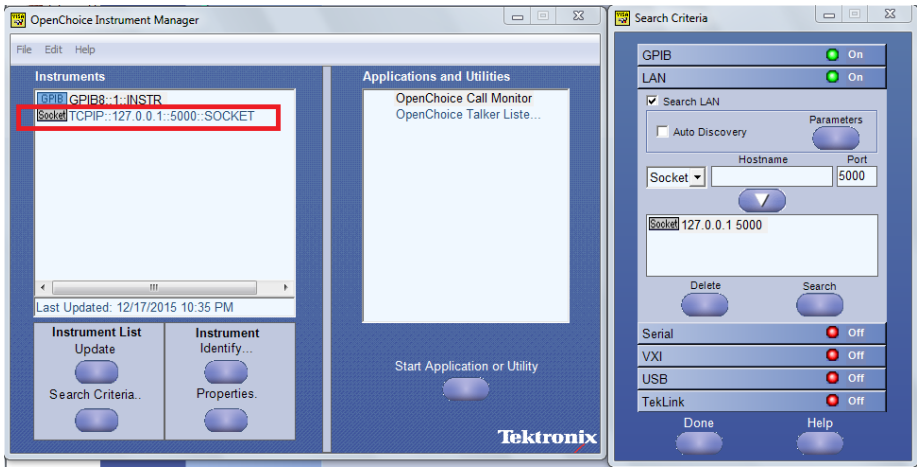


2. Click **Search Criteria**. In Search Criteria menu, click **LAN** to Turn-on. Select **Socket** from the drop-down list, enter the IP address of the TekExpress device in **Hostname** and type **Port** as 5000. Click  to configure the IP address with Port.

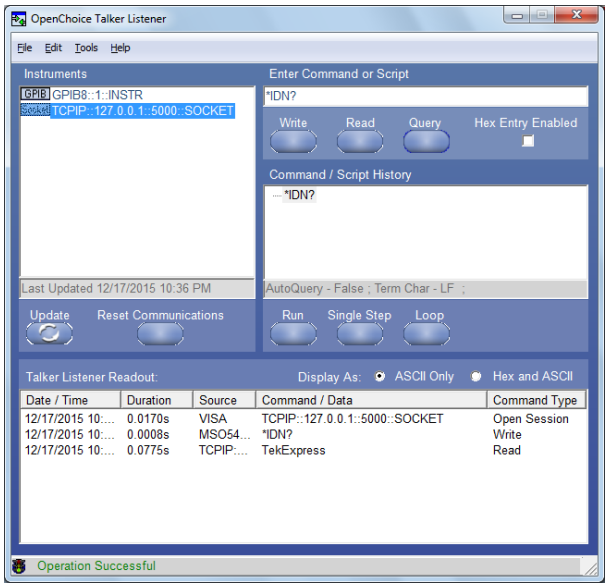
   Enter the Hostname as 127.0.0.1 if the TekVISA and TekExpress application are in the same system, else enter the IP address of the TekExpress application system.



3. Click **Search** to setup the TCPIP connection with the host. Check whether the TCPIP host name is displayed in **OpenChoice Instrument Manager** > **Instruments**



4. Double-click **OpenChoice Takler Listener** and enter the Command *IDN? in command entry field and click **Query**. Check that the Operation is successful and Talker Listener Readout displays the Command / Data.



TEKEXP:*IDN?

This command queries the active TekExpress application name running on the scope.

Syntax TEKEXP:*IDN?\n

Inputs NA

Outputs Returns active TekExpress application name running on the scope

TIP. *Click here for examples.*

TEKEXP:*OPC?

This command queries the execution status of the last executed command.

Syntax TEKEXP:*OPC?\n

Inputs NA

Outputs 0 - last command execution is not complete

1 - last command execution is complete

TIP. *Click here for examples.*

TEKEXP:ACQUIRE_MODE

This command sets the acquire mode as live or pre-recorded.

Syntax TEKEXP:ACQUIRE_MODE {LIVE | PRE-RECORDED}\n

Inputs {LIVE | PRE-RECORDED}

Outputs NA

TIP. *Click here for examples.*

TEKEXP:ACQUIRE_MODE?

This command queries the acquire mode type.

Syntax TEKEXP:ACQUIRE_MODE?\n

Inputs NA

Outputs {LIVE | PRE-RECORDED}

TIP. *Click here for examples.*

TEKEXP:EXPORT

This command returns all the bytes of data to the specified file.

Syntax	Outputs
TEKEXP:EXPORT REPORT\n	Returns the report file in bytes
TEKEXP:EXPORT WFM,"<FileName>"\n	Returns the specified waveform file in bytes
TEKEXP:EXPORT IMAGE,"<FileName>"\n	Returns the specified image file in bytes

Inputs FileName - Specifies the file name

TIP. *Click here for examples.*

TEKEXP:INFO?

This command queries the information about the file(s).

Syntax	Outputs
TEKEXP:INFO? REPORT\n	<ReportFileSize>,"<ReportFileName.mht>"
TEKEXP:INFO? WFM\n	<WfmFile1Size>,"<WfmFileName1.wfm>";<WfmFile2Size>,"<WfmFileName2.wfm>";...
TEKEXP:INFO? IMAGE\n	<Image1FileSize>,"<Image1FileName>";<Image2FileSize>,"<Image2FileName>" ;...

TIP. *Click here for examples.*

TEKEXP:INSTRUMENT

This command sets the value for the selected instrument type.

Syntax TEKEXP:INSTRUMENT "<InstrumentType>",<Value>"\n

Inputs InstrumentType

Value

TIP. *Check Command parameters list for InstrumentType and Value parameters.*

Outputs NA

TIP. *Click here for examples.*

TEKEXP:INSTRUMENT?

This command queries the instrument selected for the specified instrument type.

Syntax TEKEXP:INSTRUMENT? "<InstrumentType>"\n

Inputs InstrumentType

TIP. *Check Command parameters list for InstrumentType parameters.*

Outputs Returns the instrument selected for the specified instrument type

TIP. *Click here for examples.*

TEKEXP:LASTERROR?

This command queries the last error string occurred for the current TCP session. If there are no errors since startup, or since the last call to TEKEXP:LASTERROR?\n, this command returns an empty string.

Syntax TEKEXP:LASTERROR?\n

Inputs NA

Outputs <string>

TIP. *Click here for examples.*

TEKEXP:LIST?

This command queries the list of available device, suite, test, version or instrument.

Syntax	Outputs
TEKEXP:LIST? DEVICE\n	Returns the list of available device(s) as comma separated values.
TEKEXP:LIST? SUITE\n	Returns the list of available suite(s) as comma separated values.
TEKEXP:LIST? TEST\n	Returns the list of available test(s) as comma separated values.
TEKEXP:LIST? VERSION\n	Returns the list of available version(s) as comma separated values.
TEKEXP:LIST? INSTRUMENT,"<InstrumentType>"\n	Returns the list of available instruments' for the given Instrument type as comma separated values.

NOTE. *This command returns the list of items within double quotes (""). Iterate the receive procedure until the list ends with double quotes otherwise the next query commands won't work as expected.*

Inputs InstrumentType

TIP. *Check Command parameters list for InstrumentType parameters.*

TIP. *Click here for examples.*

TEKEXP:MODE

This command sets the execution mode as compliance or user defined.

Syntax TEKEXP:MODE {COMPLIANCE | USER-DEFINED}\n

Inputs {COMPLIANCE | USER-DEFINED}

Outputs NA

TIP. *Click here for examples.*

TEKEXP:MODE?

This command queries the execution mode type.

Syntax TEKEXP:MODE?\n

Inputs NA

Outputs {COMPLIANCE | USER-DEFINED}

TIP. *Click here for examples.*

TEKEXP:POPUP

This command sets the response to the active popup shown in the application.

Syntax TEKEXP:POPUP "<PopupResponse>"\n

Inputs PopupResponse

Outputs NA

TIP. *Click here for examples.*

TEKEXP:POPUP?

This command queries the active popup information shown in the application.

Syntax TEKEXP:POPUP?\n

Inputs NA

Outputs Returns the active popup information in the application.

TIP. *Click here for examples.*

TEKEXP:REPORT

This command generates the report for the current session.

Syntax TEKEXP:REPORT GENERATE\n

Inputs GENERATE

Outputs NA

TIP. *Click here for examples.*

TEKEXP:REPORT?

This command queries the queried header field value in the report.

Syntax TEKEXP:REPORT? “<HeaderField>”\n

Inputs HeaderField - Specifies to return the measured value for the indicated test.

TIP. *Check* ***Report*** *for HeaderField parameters.*

Outputs Returns the queried header field value in the report

TIP. *Click here for examples.*

TEKEXP:RESULT?

This command queries the result available in report summary/details table.

Syntax	Outputs
TEKEXP:RESULT? "<TestName>"\n	Return Pass/Fail status of the test.
TEKEXP:RESULT? "<TestName>","<ColumnName>"\n	Returns all the row values of the specified column for the test.
TEKEXP:RESULT? "<TestName>","<ColumnName>",<RowNumber>\n	Returns the column value for the specified row number [1]

[1] Row number starts from zero.

Inputs

TestName - Specifies the name of the test for which to obtain the test result value.

ColumnName - Specifies the column name for the measurement

RowNumber - Specifies the row number of the measurement

TIP. *Check* ***Results*** *panel for TestName, ColumnName, and RowNumber parameters.*

TIP. *Click here for examples.*

TEKEXP:SELECT

This command selects the device, suite, version, or test.

Syntax

TEKEXP:SELECT <string1>,<string2>,<string4>\n

TEKEXP:SELECT TEST,<string3>,<string4>\n

Inputs

<string1> = {DEVICE | SUITE | VERSION}

<string2> = {DeviceName | SuiteName | VersionName}

<string3> = {“<TestName>"| ALL| REQUIRED }

<string4> = {TRUE | FALSE}

TIP. *Check Command parameters list for DeviceName, SuiteName, VersionName, and TestName parameters.*

TIP. *Click here for examples.*

Outputs NA

TEKEXP:SELECT?

This command queries the name of the selected device, suite, version, or test.

Syntax TEKEXP:SELECT? {DEVICE | SUITE | TEST | VERSION}\n

Inputs {DEVICE | SUITE | TEST | VERSION}

Outputs Returns the name of the selected device, suite, version, or test.

TIP. *Click here for examples.*

TEKEXP:SETUP

This command sets the value of the current setup.

Syntax	Outputs
TEKEXP:SETUP DEFAULT\n	Restore to default Setup
TEKEXP:SETUP OPEN,"<SessionName>"\n	Open the session
TEKEXP:SETUP SAVE\n	Save the session
TEKEXP:SETUP SAVE,"<SessionName>"\n	Save the session

Inputs SessionName - The name of the session

TIP. *Click here for examples.*

TEKEXP:STATE

This command sets the execution state of the application.

Syntax TEKEXP:STATE {RUN | STOP | PAUSE | RESUME}\n

Inputs {RUN | STOP | PAUSE | RESUME}

Outputs NA

TIP. *Click here for examples.*

TEKEXP:STATE?

This command queries the current setup state.

Syntax	Outputs
TEKEXP:STATE?	RUNNING | PAUSED | WAIT | ERROR | READY | STOPPED
TEKEXP:STATE? SETUP	SAVED | NOT_SAVED

TIP. *Click here for examples.*

TEKEXP:VALUE

This command sets the value of parameters of type General, Acquire, Analyze, or DUTID.

Syntax

TEKEXP:VALUE GENERAL,"<ParameterName>",“<Value>”\n

TEKEXP:VALUE ACQUIRE,“<TestName>","<AcquireType>", "<ParameterName>",“<Value>”\n

TEKEXP:VALUE ANALYZE,“<TestName>","<ParameterName>".“<Value>” \n

TEKEXP:VALUE DUTID,“<Value>”\n

Inputs

ParameterName - Specifies the parameter name

TestName - Specifies the test name

AcquireType - Specifies the acquire type

Value - Specifes the value to set

TIP. *Check Command parameters list for ParameterName, AcquireType, and Value parameters.*

Outputs

NA

TIP. *Click here for examples.*

TEKEXP:VALUE?

This command queries the value of the parameter for type General, Acquire, Analyze, or DUTID.

Syntax	Outputs
TEKEXP:VALUE? GENERAL,"<ParameterName>"\n	Returns the value of Parameter for type GENERAL
TEKEXP:VALUE? ACQUIRE,“<TestName>", "<AcquireType>","<ParameterName>"\n	Returns the value of Parameter for type ACQUIRE
TEKEXP:VALUE? ANALYZE, “<TestName>","<ParameterName>"\n	Returns the value of Parameter for type ANALYZE
TEKEXP:VALUE? DUTID\n	Returns the DUTID value

Inputs

ParameterName - Specifies the parameter name

TestName - Specifies the test name

AcquireType - Specifies the acquire type

TIP. *Check Command parameters list for ParameterName and AcquireType parameters.*

Outputs

Returns the value of Parameter for type GENERAL | ACQUIRE | ANALYZE | DUTID.

TIP. *Click here for examples.*

Command parameters list

This section provides the parameters list for the SCPI commands.

Parameters	Description
InstrumentType	Specifies the instrument type. Valid values are: ■ Alternate Real Time Scope ■ Real Time Scope
Value	Specifies the value parameters. ■ For InstrumentType, valid values are: ■ Comment ■ For DUTID, valid values are: ■ Comment
DeviceName	Specifies the device name. Valid values are **CEM**, **U.2(SFF-8639)**, and **BaseSpec**
SuiteName	Specifies the suite name. For CEM, valid values are System-Board, Add-In-Card For U.2(SFF-8639), valid values are Host, Module For BaseSpec, valid value is TX Test Board

Parameters	Description
VersionName	Specifies the version name. Valid values are: ■ Gen1-1.0a (Applicable only for DeviceName = CEM) ■ Gen1-1.1 (Applicable only for DeviceName = CEM) ■ Gen2-2.0 (Applicable only for DeviceName = CEM) ■ Gen3-3.0

Parameters	Description
TestName	Specifies the test measurement name. Valid values for CEM and U.2(SFF-8639) are: ■ Unit Interval ■ Mask Hits(All Bits) ■ Composit Eye Height ■ Number Passing Eyes ■ Number Failing Eyes ■ Transition Eye Diagram ■ Non Transition Eye Diagram ■ Min Eye Width ■ Min Time Between Crossovers ■ RMS Jitter (Per Edge) ■ Median Peak Jitter ■ TJ @ E-12 ■ Dj_dd ■ RJ(RMS) ■ Peak to Peak Jitter ***NOTE.*** *Add one space at the end of the test name to select the 5Gbps tests and two spaces to select the 8Gbps tests.* Valid values for BaseSpec are: ■ AC CM 4GHz ■ AC CM 30KHz-500MHz ■ EIEOS Min Voltage ■ V Tx_no_eq ■ DDj ■ F/2 Jitter ■ PS21 Ratio ■ PWJ RJ (RMS) ■ TIE RJ (RMS) ■ Uncorrelated PWJ DJ dd@E-12 ■ Uncorrelated PWJ TJ@E-12 ■ Uncorrelated TIE DJ dd@E-12 ■ Uncorrelated TIE TJ @ E-12

ParameterName and Value for General

Specifies the ParameterName and Value for General. The configuration parameters available are not same for measurements.

Table 15: ParameterName and Value for General

ParameterName	Value
DataRate2Gb	■ Included ■ Excluded
DataRate5Gb	■ Included ■ Excluded
DataRate8Gb	■ Included ■ Excluded
PreEmphasis3dB	■ Included ■ Excluded
PreEmphasis6dB	■ Included ■ Excluded
SSC	■ On ■ Off
VoltageSwing	■ Full ■ Reduced
Link Widths	1 Lane
SignalPreset	P0 For multiple signal preset, specify as P0_P1_P2
Preset	P0 For multiple preset, specify as P0_P1_P2
Acquisition	BeforeAnalysis
SaveOptions	Save All the Waveforms
DeEmbed2Gb	■ Included ■ Excluded
DeEmbed5Gb	■ Included ■ Excluded
DeEmbed8Gb	■ Included ■ Excluded

ParameterName	Value
Embed8Gb	■ Included ■ Excluded
Equalization8Gb	■ Included ■ Excluded
Filterfile2Gb	Filterfile2Gb.flt
Filterfile5Gb	Filterfile5Gb.flt
FilterfileDeEmbed8Gb	FilterfileDeEmbed8Gb.flt
FilterfileEmbed8Gb	FilterfileEmbed8Gb.flt
EmbedDropdown	■ SigTest ■ Scope
EqualizationDropdown	■ Optimize ■ Fixed
CTLE Index	1 : -12dB
DFE	-30e-3
EnableDUTAutomation	■ Included ■ Excluded
Automation Settings	Use Default Settings
Signal Type	■ Square ■ Sine
RecordLength2Gb	2.5e6
RecordLength5Gb	10e6
RecordLength8Gb	10e6
SampleRate2Gb	50e9
SampleRate5Gb	50e9
SampleRate8Gb	50e9
Bandwidth2Gb	7e9
Bandwidth5Gb	12.5e9
Bandwidth8Gb	12.5e9
Signal Validation	Turn Off Signal Check
SlotNumber	05
Toggle from Gen3P10 to Gen1	■ On ■ Off
Signal Validation Threshold(mV)	200

ParameterName	Value
AnalysisMode	■ DLL ■ CLI
Report Update Mode	■ New ■ Append ■ Replace
Auto increment report name if duplicate	TRUE or FALSE
Include Pass/Fail Results Summary	TRUE or FALSE
Include Detailed Results	TRUE or FALSE
Include Plot Images	TRUE or FALSE
Include Setup Configuration	TRUE or FALSE
Include Complete Application Configuration	TRUE or FALSE
Include User Comments	TRUE or FALSE
Save As Type	■ Web Archive (*.mht;*.mhtml) ■ PDF (*.pdf;)
View Report After Generating	TRUE or FALSE
Report Group Mode	■ Test Name ■ Lane Name ■ Test Result ■ Equalization
Create report at the end	■ Included ■ Excluded
DUTID Comment	User comment
Number of retries for instrument IO errors	0 to 5
Run Test More than Once	TRUE or FALSE
Number of Runs	1 to 100
On Failure Rerun	TRUE or FALSE
Number of Reruns On Failure	1 to 100
Time between retries (seconds)	5 to 60
Timer Warning Info Message Popup	■ "True" ■ "FALSE"

ParameterName	Value
Timer Warning Info Message Popup Duration	0 to 20
Timer Error Message Popup	■ "True" ■ "False"
Timer Error Message Popup Duration	0 to 20
On Failure Stop and Notify	TRUE or FALSE

Examples

This section provides the examples for the SCPI commands.

Example	Description
TEKEXP:*IDN?\n	It returns the active TekExpress application name running on the scope.
TEKEXP:*OPC?\n	It returns the last command execution status.
TEKEXP:ACQUIRE_MODE PRE-RECORDED\n	It sets the acquire mode as pre-recorded.
TEKEXP:ACQUIRE_MODE?\n	It returns LIVE when acquire mode is set to live.
TEKEXP:EXPORT REPORT\n	It returns the report file in bytes. This can be written into another file for further analysis.
TEKEXP:INFO? REPORT\n	It returns “100,”ReportFileName.mht””, when 100 is the filesize in bytes for the filename ReportFileName.
TEKEXP:INFO? WFM\n	It returns “100,“WfmFileName1.wfm””;“200,“WfmFileName2.wfm”" when 100 is the filesize in bytes for the filename WfmFileName1.wfm and 200 is the filesize in bytes for the filename WfmFileName2.wfm.
TEKEXP:INSTRUMENT "Real Time Scope",DPO72504D (GPIB8::1::INSTR)\n	It sets the instrument value as DPO72504D (GPIB8::1::INSTR) for the selected instrument type Real Time Scope.
TEKEXP:INSTRUMENT? “Real Time Scope” \n	It returns "IDPO72504D (GPIB8::1::INSTR), when DPO72504D (GPIB8::1::INSTR)" is the selected instrument for the instrument type Real Time Scope.
TEKEXP:LASTERROR?\n	It returns ERROR: INSTRUMENT_NOT_FOUND, when no instrument is found.
TEKEXP:LIST? DEVICE\n	It returns "TX-Device,RX-Device" when TX-Device, RX-Device are the available device.
TEKEXP:LIST? INSTRUMENT,"Real Time Scope"\n	It returns "DPO72504D (GPIB8::1::INSTR),MSO72504 (TCPIP::134.64.248.91::INSTR)" when DPO72504D (GPIB8::1::INSTR), MSO72504 (TCPIP::134.64.248.91::INSTR) are the list of available instruments.
TEKEXP:MODE COMPLIANCE\n	It sets the execution mode as compliance.
TEKEXP:MODE?\n	It returns COMPLIANCE when the execution mode is compliance.
TEKEXP:POPUP “OK”\n	It sets OK as the response to active popup in the application.
TEKEXP:POPUP?\n	It returns "OK", when OK is the active popup information shown in the application.
TEKEXP:REPORT GENERATE\n	It generates report for the current session.
TEKEXP:REPORT? “Scope Model”\n	It returns "DPO73304SX" when DPO73304SX is the scope model.
TEKEXP:REPORT? “DUT ID”\n	It returns "DUT001" when DNI_DUT001 is the DUT ID.

Example	Description
TEKEXP:RESULT? "Period using SCOPE (Acquire-Analyze Combined)"\n	It returns Pass when the test result is Pass.
TEKEXP:RESULT? "Period using SCOPE (Acquire-Analyze Combined)","Margin",1\n	It returns "L:-50.000ps H:2000.000ps" when L:-50.000ps H:2000.000ps is the value.
TEKEXP:SELECT DEVICE, TX_Device, TRUE\n	It selects TX_Device
TEKEXP:SELECT? DEVICE\n	It returns "TX-Device" when TX-Device is the selected device type.
TEKEXP:SETUP DEFAULT\n	It restores the application to default setup.
TEKEXP:STATE STOP\n	It stops the test execution.
TEKEXP:STATE?\n	It returns as READY when the application is ready to run next measurement.
TEKEXP:STATE? SETUP\n	It returns as NOT_SAVED when the current setup is not saved.
TEKEXP:VALUE GENERAL,"Signal Type", "N1N0"\n	It sets the signal type parameter value to N1N0.
TEKEXP:VALUE? GENERAL,"Signal Type"\n	It returns "N1N0" when N1N0 is the Signal Type value.

Switch Matrix commands

This section describes the switch matrix commands. It is recommended to execute the switch matrix commands through GPIB interface.

SWITCH:*IDN

This command queries the switch matrix version information.

Syntax. SWITCH:*IDN?\n

Inputs. NA

Outputs. Returns the switch martrix version information.

Example. SWITCH:*IDN?\n returns "Tektronix,Switch Matrix,v1.0.0.0", where v1.0.0.0 is the Switch Matrix version.

SWITCH:*OPC

This command queries the previously executed commands execution status.

Syntax. SWITCH:*OPC?\n

Inputs. NA

Outputs. Returns 0 if the previously executed command execution is in progress.

Returns 1 if the previously executed command execution is done.

Example. SWITCH:*OPC?\n returns 1, when the previously executed command execution is done.

SWITCH:CONFIG

This command sets or queries the config file.

Syntax. SWITCH:CONFIG {"<ConfigName>" | "<UserConfigFilePath>"}\n

SWITCH:CONFIG?\n

Inputs. <ConfigName> specifies the config file.

<UserConfigFilePath> specifies the config file from the given path.

Outputs. Returns the loaded config file name with path.

Examples. SWITCH:CONFIG "Keithley S46T"\n sets the config file of Keithley.

SWITCH:CONFIG "E:\myconfig.xml"\n sets the config file from the given path.

SWITCH:CONFIG?\n returns "E:\myconfig.xml".

SWITCH:DE-EMBED:ALL:FILTER_FILE

This command sets the filter file for all connections.

Syntax. SWITCH:DE-EMBED:ALL:FILTER_FILE "<SwitchFilterFilePath>","<FixtureFilterFilePath>"\n

Inputs. <SwitchFilterFilePath> specifies the switch filter file path.

<FixtureFilterFilePath> specifies the fixture filter file path.

Outputs. NA

Example. SWITCH:DE-EMBED:ALL:FILTER_FILE "C:\FilterFiles \SWTCH1.flt","C:\FilterFiles\CABLE_1.flt"\n sets the switch filter file for all connections and the fixture filter file for all cables connected.

SWITCH:DE-EMBED:CONN:FILTER_FILE

This command sets the filter file for the switch, fixture, and cable for the specified connection.

Syntax. SWITCH:DE-EMBED:CONN:FILTER_FILE "<RelayName>","<InputName>","<SwitchFilterFilePath>","<FixtureFilterFilePath>","<CableFilterFilePath>"\n

Inputs. <RelayName> specifies the relay name.

<InputName> specifies the input name.

<SwitchFilterFilePath> specifies the switch filter file path.

<FixtureFilterFilePath> specifies the fixture filter file path.

<CableFilterFilePath> specifies the cable filter file path.

Example. SWITCH:DE-EMBED:CONN:FILTER_FILE "Relay A","1","C:\FilterFiles\RA_1.flt","C:\FilterFiles\Fxtre_1.flt","C:\FilterFiles\cbl_1.flt"\n sets filter files to Relay A's input 1 and to the connected cable.

SWITCH:DE-EMBED:FILTER_FILE

This command queries the filter file based on the mode selected in the application.

Syntax. SWITCH:DE-EMBED:FILTER_FILE?\n

Inputs. NA

Outputs. Returns the filter file in any of the the below specified format, based on the mode selected.

None
ALL;"<SwitchFilterFilePath>","<FixtureFilterFilePath>";
RELAY_TYPE;"<RelayType1>","<SwitchFilterFilePath1>","<FixtureFilterFilePath1>";"<RelayType2>","<SwitchFilterFilePath2>","<FixtureFilterFilePath2>";
RELAY;"<RelayName1>","<SwitchFilterFilePath1>","<FixtureFilterFilePath1>";"<RelayName2>","<SwitchFilterFilePath2>","<FixtureFilterFilePath2>";
CONN;"<RelayName1>","<InputName1>,"<SwitchFilterFilePath1>","<FixtureFilterFilePath1>","<CableFilterFIlePath1>;"<RelayName1>","<InputName2>,"<SwitchFilterFilePath2>","<FixtureFilterFilePath2>","<CableFilterFIlePath2>;

Example. SWITCH:DE-EMBED:FILTER_FILE?\n returns ALL;"C:\FilterFiles\SWTCH1.flt" ,"C:\FilterFiles\Fxtre_1.flt".

SWITCH:DE-EMBED:MODE

This command sets or queries the De-Embed mode.

Syntax. SWITCH:DE-EMBED:MODE {NONE | ALL | RELAY_TYPE | RELAY | CONN}\n

SWITCH:DE-EMBED:MODE?\n

Inputs. {NONE | ALL | RELAY_TYPE | RELAY | CONN}

Outputs. Returns the De-Embed mode.

Examples. SWITCH:DE-EMBED:MODE ALL\n sets the De-Embed mode as ALL.

SWITCH:DE-EMBED:MODE?\n returns ALL.

SWITCH:DE-EMBED:RELAY:FILTER_FILE

This command sets the filter file and fixture file for the specified relay.

Syntax. SWITCH:DE-EMBED:RELAY:FILTER_FILE "<RelaName>","<SwitchFilterFilePath>","<FixtureFilterFilePath>"\n

Inputs. <RelaName> specifies the relay name.

<SwitchFilterFilePath> specifies the switch filter file path.

<FixtureFilterFilePath> specifies the fixture filter file path.

Outputs. SWITCH:DE-EMBED:RELAY:FILTER_FILE "Relay A","C:\FilterFiles\RA.flt","C:\FilterFiles\Fxtre_1.flt"\n sets filter files for all connections in Relay A.

SWITCH:DE-EMBED:RELAY_TYPE:FILTER_FILE

This command sets the filter file for specified relay type.

Syntax. SWITCH:DE-EMBED:RELAY_TYPE:FILTER_FILE "<RelayType>","<SwitchFilterFilePath>","<FixtureFilterFilePath>"\n

Inputs. <RelayType> specifies the relay type.

<SwitchFilterFilePath> specifies the switch filter file path.

<FixtureFilterFilePath> specifies the fixture filter file path.

Example. SWITCH:DE-EMBED:RELAY_TYPE:FILTER_FILE "SP2T","C:\FilterFiles\SP2T.flt","C:\FilterFiles\Fxtre_1.flt"\n sets filter file for SP2T relay type.

SWITCH:LASTERROR

This command queries the error occurred while executing last command.

Syntax. SWITCH:LASTERROR?\n

Inputs. NA

Outputs. Returns the error occurred while executing last command.

Error message	Description
NO_ERROR	No error occurred executing last command.
INVALID_COMMAND	The last command sent is either invalid or syntax is not correct.
"No config file loaded."	The last command to load the config file is failed.
"Invalid configuration."	The last command has invalid configuration.
"Invalid linkwidth."	The last command has invalid linkwidth.
"Invalid de-embed mode."	The last command has invalid de-embed mode.
"Invalid signal polarity."	The last command has invalid signal polarity.
"Invalid relay name."	The last command has invalid relay name.
"Invalid relay type."	The last command has invalid relay type.
"Invalid signal name."	The last command has invalid signal name.
"Invalid common value."	The last command has invalid value for common.
"Invalid input ID."	The last command has invalid input ID.

Example. SWITCH:LASTERROR?\n returns the last error occurred.

SWITCH:LINKWIDTH

This command sets or queries the link width.

Syntax. SWITCH:LINKWIDTH {X2 | X4 | X8 | X16}\n

SWITCH:LINKWIDTH?\n

Inputs. {X2 | X4 | X8 | X16}

Outputs. Returns the link width value.

Examples. SWITCH:LINKWIDTH X8\n sets the linkWidth to 8.

SWITCH:LINKWIDTH?\n returns 8.

SWITCH:RELAY:CASCADE

This command sets or queries the cascade of the relay.

Syntax. SWITCH:RELAY:CASCADE "<FromRelayName>","<ToRelayName>","<InputName>"\n

SWITCH:RELAY:CASCADE? "<RelayName>"\n

Inputs. <FromRelayName> specifies the relay name from which to cascade.

<ToRelayName> specifies the relay name to cascade.

<InputName> specifies the input name of the relay to cascade.

<RelayName> specifies the relay name.

Outputs. Returns the cascading info of the relay if cascaded else returns NOT_CASCADED.

Example. SWITCH:RELAY:CASCADE "Relay A","Relay B","2"\n sets Relay A's output cascade to Relay B's input 2.

SWITCH:RELAY:CASCADE? "Relay A"\n returns "Relay B","2".

SWITCH:RELAY:CASCADE? "Relay B"\n returns NOT_CASCADED.

SWITCH:RELAY:CASCADED

This command sets or queries cascaded state of the relay.

Syntax. SWITCH:RELAY:CASCADED "<RelayName>",{TRUE | FALSE}\n

SWITCH:RELAY:CASCADED? "<RelayName>"\n

Inputs. <RelayName> specifies the relay name.

TRUE or FALSE to cascade the relay or not.

Outputs. Returns whether the relay is cascaded or not.

Examples. SWITCH:RELAY:CASCADED "Relay A",TRUE\n sets the cascaded state of Relay A to TRUE.

SWITCH:RELAY:CASCADED? "Relay A"\n returns TRUE.

SWITCH:RELAY:COMMON

This command sets or queries the relays common connection connected to scope channels.

Syntax. SWITCH:RELAY:COMMON "<RelayName>",{CH1 | CH2 | CH3 | CH4}\n

SWITCH:RELAY:COMMON? "<RelayName>"\n

Inputs. <RelayName> specifies the relay name.

{CH1 | CH2 | CH3 | CH4} specifies the channel number.

Outputs. Returns the channel connected to the common connection of relay.

Example. SWITCH:RELAY:COMMON "Relay A",CH1\n sets the common connection of Relay A to CH1.

SWITCH:RELAY:COMMON? "Relay A"\n returns CH1.

SWITCH:RELAY:POLARITY

This command sets or queries the signal polarity of the specified relay.

Syntax. SWITCH:RELAY:POLARITY "<RelayName>",{POS | NEG | DIFF}\n

SWITCH:RELAY:POLARITY? "<RelayName>"\n

Inputs. <RelayName> specifies the relay name.

{POS | NEG | DIFF} specifies the signal polarity as Positive, Negative or Differential respectively.

Outputs. Returns the signal polarity of the relay.

Examples. SWITCH:RELAY:POLARITY "Relay A",POS\n sets the Relay A's signal polarity to POS.

SWITCH:RELAY:POLARITY? "Relay A"\n returns POS.

SWITCH:RELAY:SIGNAL

This command sets or queries the signal name connected to input port.

Syntax. SWITCH:RELAY:SIGNAL "<RelayName>","<InputName>","<SignalName>"\n

SWITCH:RELAY:SIGNAL? "<RelayName>","<InputName>"\n

Inputs. <RelayName> specifies the relay name.

<InputName> specifies the input name of the relay.

<SignalName> specifies the signal name to connect to the relays input.

Outputs. Returns the signal name connected to the input port.

Example. SWITCH:RELAY:SIGNAL "Relay A","1","Lane0+"\n sets the signal name Lane0+ to Relay A's input 1.

SWITCH:RELAY:SIGNAL? "Relay A","1"\n returns "Lane0+".

Reference

De-embed using filter files

TekExpress PCIe provides an option to de-embed the signal path using filter files. You create the filter files. The filter files are .flt files composed of de-embed filter coefficients for a particular sampling rate. A filter file created for one sampling rate might not work for other sampling rates, so it is important to understand at what sampling rate the measurements are being performed.

Also, the de-embedding filters might differ based on the type of input. For example, if a single ended input is made using a matched SMA cable pair, a filter file for de-embedding a single SMA cable must be provided, since matched SMA cables mostly have similar s-parameters. So in this case, the same filter file is used to de-embed the SMA cable pair.

The maximum sampling rate provided on any channel combination on MSO/DPO/DSA70000/C/D/DX series oscilloscopes is 50 GS/s in realtime mode. The maximum sampling rate provided on Ch1-Ch3 and Ch2-Ch4 channel combinations on MSO/DPO/DSA70000C/D/DX series oscilloscopes is 100 GS/s, provided only 2 channels are on at a given time.

See also

Common test parameters and values

Configuration test parameters

Setup files

TekExpress PCI Express package contains setup files (*.TekX) which can be used at PCIE Gen3 workshop for compliance tests.

Table 16: Setup files configuration details

Setup files (*.TekX)	Configuration details (exclusively used in Gen1/2/3 Gold Suite of PCI-SIG Work Shop (WS))
WorkShop_CEM_AIC _x1	■ Specification - CEM ■ Device Type - Add-In-Card ■ Version - Gen3 - 3.0 ■ Data Rates - 2.5 Gbps, 5 Gbps (Tx equalization 3.5dB, 6 dB) and 8 Gbps ■ Signal Quality Preset Selection - P0, P7 and P8 for L0 and All Presets from P0 to P10 for Lane0 ■ Link Width - 1 Lane (Selected test lane: L0) ■ Automated DUT Control - unchecked ■ Signal Validation - Pattern Decoding ■ SigTest Interface Mode: CMD line - SigTest displayed and its HTML report saved with TE Report
WorkShop_CEM_AIC_x2	■ Specification - CEM ■ Device Type - Add-In-Card ■ Version - Gen3 - 3.0 ■ Data Rates - 2.5 Gbps, 5 Gbps (Tx equalization 3.5dB, 6 dB) and 8 Gbps ■ Signal Quality Preset Selection - P0, P7 and P8 for L0 and All Presets from P0 to P10 for Lane0 ■ Link Width - 2 Lanes (Selected test lane: L0) ■ Automated DUT Control - unchecked ■ Signal Validation - Pattern Decoding ■ SigTest Interface Mode: CMD line - SigTest displayed and its HTML report saved with TE Report
WorkShop_CEM_AIC _x4	■ Specification - CEM ■ Device Type - Add-In-Card ■ Version - Gen3 - 3.0 ■ Data Rates - 2.5 Gbps, 5 Gbps (Tx equalization 3.5dB, 6 dB) and 8 Gbps ■ Signal Quality Preset Selection - P0, P7 and P8 for L0 and All Presets from P0 to P10 for Lane0 ■ Link Width - 4 Lanes (Selected test lane: L0, L03) ■ Automated DUT Control - unchecked ■ Signal Validation - Pattern Decoding ■ SigTest Interface Mode: CMD line - SigTest displayed and its HTML report saved with TE Report

Setup files (*.TekX)	Configuration details (exclusively used in Gen1/2/3 Gold Suite of PCI-SIG Work Shop (WS))
WorkShop_CEM_AIC_x8	■ Specification - CEM ■ Device Type - Add-In-Card ■ Version - Gen3 - 3.0 ■ Data Rates - 2.5 Gbps, 5 Gbps (Tx equalization 3.5dB, 6 dB) and 8 Gbps ■ Signal Quality Preset Selection - P0, P7 and P8 for L0 and All Presets from P0 to P10 for Lane0 ■ Link Width - 8 Lanes (Selected test lane: L0, L03, L07) ■ Automated DUT Control - unchecked ■ Signal Validation - Pattern Decoding ■ SigTest Interface Mode: CMD line - SigTest displayed and its HTML report saved with TE Report
WorkShop_CEM_AIC _x16	■ Specification - CEM ■ Device Type - Add-In-Card ■ Version - Gen3 - 3.0 ■ Data Rates - 2.5 Gbps, 5 Gbps (Tx equalization 3.5dB, 6 dB) and 8 Gbps ■ Signal Quality Preset Selection - P0, P7 and P8 for L0 and All Presets from P0 to P10 for Lane0 ■ Link Width - 16 Lanes (Selected test lane: L0, L07, L15) ■ Automated DUT Control - unchecked ■ Signal Validation - Pattern Decoding ■ SigTest Interface Mode: CMD line - SigTest displayed and its HTML report saved with TE Report
WorkShop_CEM_SYB_x1	■ Specification - CEM ■ Device Type - Add-In-Card ■ Version - Gen3 - 3.0 ■ Data Rates - 2.5 Gbps, 5 Gbps (Tx equalization 3.5dB, 6 dB) and 8 Gbps ■ Signal Quality Preset Selection - P0, P7 and P8 for L0 and All Presets from P0 to P10 for Lane0 ■ Link Width - 1 Lane (Selected test lane: L0) ■ Automated DUT Control - unchecked ■ Signal Validation - Pattern Decoding ■ SigTest Interface Mode: CMD line - SigTest displayed and its HTML report saved with TE Report

Setup files (*.TekX)	Configuration details (exclusively used in Gen1/2/3 Gold Suite of PCI-SIG Work Shop (WS))
WorkShop_CEM_SYB_x2	■ Specification - CEM ■ Device Type - Add-In-Card ■ Version - Gen3 - 3.0 ■ Data Rates - 2.5 Gbps, 5 Gbps (Tx equalization 3.5dB, 6 dB) and 8 Gbps ■ Signal Quality Preset Selection - P0, P7 and P8 for L0 and All Presets from P0 to P10 for Lane0 ■ Link Width - 2 Lanes (Selected test lane: L0) ■ Automated DUT Control - unchecked ■ Signal Validation - Pattern Decoding ■ SigTest Interface Mode: CMD line - SigTest displayed and its HTML report saved with TE Report
WorkShop_CEM_SYB_x4	■ Specification - CEM ■ Device Type - Add-In-Card ■ Version - Gen3 - 3.0 ■ Data Rates - 2.5 Gbps, 5 Gbps (Tx equalization 3.5dB, 6 dB) and 8 Gbps ■ Signal Quality Preset Selection - P0, P7 and P8 for L0 and All Presets from P0 to P10 for Lane0 ■ Link Width - 4 Lanes (Selected test lane: L0, L03) ■ Automated DUT Control - unchecked ■ Signal Validation - Pattern Decoding ■ SigTest Interface Mode: CMD line - SigTest displayed and its HTML report saved with TE Report
WorkShop_CEM_SYB_x8	■ Specification - CEM ■ Device Type - Add-In-Card ■ Version - Gen3 - 3.0 ■ Data Rates - 2.5 Gbps, 5 Gbps (Tx equalization 3.5dB, 6 dB) and 8 Gbps ■ Signal Quality Preset Selection - P0, P7 and P8 for L0 and All Presets from P0 to P10 for Lane0 ■ Link Width - 8 Lanes (Selected test lane: L0,L03,L07) ■ Automated DUT Control - unchecked ■ Signal Validation - Pattern Decoding ■ SigTest Interface Mode: CMD line - SigTest displayed and its HTML report saved with TE Report

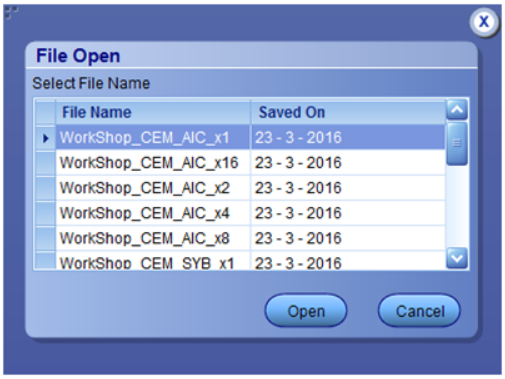
Setup files (*.TekX)	Configuration details (exclusively used in Gen1/2/3 Gold Suite of PCI-SIG Work Shop (WS))
WorkShop_CEM_SYB_x16	■ Specification - CEM ■ Device Type - Add-In-Card ■ Version - Gen3 - 3.0 ■ Data Rates - 2.5 Gbps, 5 Gbps (Tx equalization 3.5dB, 6 dB) and 8 Gbps ■ Signal Quality Preset Selection - P0, P7 and P8 for L0 and All Presets from P0 to P10 for Lane0 ■ Link Width - 16 Lanes (Selected test lane: L0,L07,L15) ■ Automated DUT Control - unchecked ■ Signal Validation - Pattern Decoding ■ SigTest Interface Mode: CMD line - SigTest displayed and its HTML report saved with TE Report
WorkShop_U2_Module_x1	■ Specification - U.2 (SFF8639) ■ Device Type - Module ■ Version - Gen3 - 3.0 ■ Data Rates - 8 Gbps ■ Signal Quality Preset Selection - P0, P7 and P8 for L0 and All Presets from P0 to P10 for Lane0 ■ Link Width - 1 Lane (Selected test lane: L0) ■ Automated DUT Control - unchecked ■ Signal Validation - Pattern Decoding ■ SigTest Interface Mode: CMD line - SigTest displayed and its HTML report saved with TE Report
WorkShop_U2_Module_x2	■ Specification - U.2 (SFF8639) ■ Device Type - Module ■ Version - Gen3 - 3.0 ■ Data Rates - 8 Gbps ■ Signal Quality Preset Selection - P0, P7 and P8 for L0 and All Presets from P0 to P10 for Lane0 ■ Link Width - 2 Lanes (Selected test lane: L0) ■ Automated DUT Control - unchecked ■ Signal Validation - Pattern Decoding ■ SigTest Interface Mode: CMD line - SigTest displayed and its HTML report saved with TE Report

Setup files (*.TekX)	Configuration details (exclusively used in Gen1/2/3 Gold Suite of PCI-SIG Work Shop (WS))
WorkShop_U2_Module_x4	■ Specification - U.2 (SFF8639) ■ Device Type - Module ■ Version - Gen3 - 3.0 ■ Data Rates - 8 Gbps ■ Signal Quality Preset Selection - P0, P7 and P8 for L0 and All Presets from P0 to P10 for Lane0 ■ Link Width - 4 Lanes (Selected test lane: L0,L03) ■ Automated DUT Control - unchecked ■ Signal Validation - Pattern Decoding ■ SigTest Interface Mode: CMD line - SigTest displayed and its HTML report saved with TE Report
WorkShop_U2_Host_x1	■ Specification - U.2 (SFF8639) ■ Device Type - Host ■ Version - Gen3 - 3.0 ■ Data Rates - 8 Gbps ■ Signal Quality Preset Selection - P0, P7 and P8 for L0 and All Presets from P0 to P10 for Lane0 ■ Link Width - 1 Lane (Selected test lane: L0) ■ Automated DUT Control - unchecked ■ Signal Validation - Pattern Decoding ■ SigTest Interface Mode: CMD line - SigTest displayed and its HTML report saved with TE Report

Setup files (*.TekX)	Configuration details (exclusively used in Gen1/2/3 Gold Suite of PCI-SIG Work Shop (WS))
WorkShop_U2_ Host _x2	■ Specification - U.2 (SFF8639) ■ Device Type - Host ■ Version - Gen3 - 3.0 ■ Data Rates - 8 Gbps ■ Signal Quality Preset Selection - P0, P7 and P8 for L0 and All Presets from P0 to P10 for Lane0 ■ Link Width - 2 Lanes (Selected test lane: L0) ■ Automated DUT Control - unchecked ■ Signal Validation - Pattern Decoding ■ SigTest Interface Mode: CMD line - SigTest displayed and its HTML report saved with TE Report
WorkShop_U2_ Host _x4	■ Specification - U.2 (SFF8639) ■ Device Type - Host ■ Version - Gen3 - 3.0 ■ Data Rates - 8 Gbps ■ Signal Quality Preset Selection - P0, P7 and P8 for L0 and All Presets from P0 to P10 for Lane0 ■ Link Width - 4 Lanes (Selected test lane: L0, L03) ■ Automated DUT Control - unchecked ■ Signal Validation - Pattern Decoding ■ SigTest Interface Mode: CMD line - SigTest displayed and its HTML report saved with TE Report

How to open a setup file

1. Click **Options** > **Open Test Setup**
2. Select the TekExpress Setup File as per your required configuration. Check Setup files configuration details table for configuration details.



3. Make the configuration details and start the test execution.

4. Click **Save Test Setup As** and save the setup.

NOTE. *You cannot edit the TekExpress Test Setup files as they are in* ***Read Only*** *mode.*

NOTE. *The setup files path is C:\Program Files (x86)\Tektronix\TekExpress \TekExpress PCI Express\Setup Files*

RF Switch configuration files

TekExpress PCI Express package contains RF Switch configuration files.

File name	Configuration details
PCE_Keithley_SYB_x6	Recall this file for Keithley S46T RF Switch and System-Board device type. The below are the configuration details: ■ Lane0 Positive to Lane05 Positive connected as Signal Inputs to Relay S1. ■ Lane0 Negative to Lane05 Negative connected as Signal Inputs to Relay S3. ■ Common output of relay A, B is connected to CH1, CH3 of the oscilloscope respectively. For 2-Unit, SX box >= 59 GHz, set the common outputs to CH1, CH3 respectively. ■ Connect the Ref Clock Positive and Negative from CLB to CH3, CH4 of oscilloscope. For 2-Unit, SX box >=59GHz, set the Ref Clock Positive and Negative to CH2 and CH4 respectively.
PCE_KthCCD_SYB_x12	Recall this file for Keithley S46T RF Switch and System-Board device type. The below are the configuration details: ■ Lane0 Positive to Lane05 Positive connected as Signal Inputs to Relay A which is Cascade to Relay 4 and Input to NO. ■ Lane0 Negative to Lane05 Negative connected as Signal Inputs to Relay B which is Cascade to Relay 5 and Input to NO. ■ Lane06 Positive to Lane11 Positive connected as Signal Inputs to Relay C which is Cascade to Relay 4and Input to NC. ■ Lane06 Negative to Lane11 Negative connected as Signal Inputs to Relay D which is Cascade to Relay 5 and Input to NC. ■ Relay 4 is selected as Positive and Relay 5 as Negative; common output of 4 and 5 is connected to CH1, CH2 of the oscilloscope respectively. For 2-Unit, SX box >= 59 GHz, set the common outputs to CH1, CH3 respectively. ■ Connect the Ref Clock Positive and Negative from CLB to CH3, CH4 of oscilloscope. For 2-Unit, SX box >=59GHz, set the Ref Clock Positive and Negative to CH2 and CH4 respectively.

File name	Configuration details
PCE_Giga_SYB_x8	Recall this file for Gigatronics ASCOR 8000 Series RF Switch and System-Board device type. The below are the configuration details: ■ Lane0 Positive to Lane07 Positive connected as Signal Inputs to Relay S1. ■ Lane0 Negative to Lane07 Negative connected as Signal Inputs to Relay S3. ■ Common output of relay S1, S3 is connected to CH1, CH3 of the oscilloscope respectively. For 2-Unit, SX box >= 59 GHz, set the common outputs to CH1, CH3 respectively. ■ Connect the Ref Clock Positive and Negative from CLB to CH3, CH4 of oscilloscope. For 2-Unit, SX box >=59GHz, set the Ref Clock Positive and Negative to CH2 and CH4 respectively.
PCE_GigCCD_SYB_x16	Recall this file for Gigatronics ASCOR 8000 Series RF Switch and System-Board device type. The below are the configuration details: ■ Lane0 Positive to Lane07 Positive connected as Signal Inputs to Relay S1 which is Cascade to Relay S2 and Input to NO. ■ Lane0 Negative to Lane07 Negative connected as Signal Inputs to Relay S3 which is Cascade to Relay S5 and Input to NO. ■ Lane08 Positive to Lane15 Positive connected as Signal Inputs to Relay S4 which is Cascade to Relay S2 and Input to NC. ■ Lane08 Negative to Lane15 Negative connected as Signal Inputs to Relay S6 which is Cascade to Relay s5 and Input to NC. ■ Relay S2 is selected as Positive and Relay S5 as Negative; common output of S2 and S5 is connected to CH1, CH2 of the oscilloscope respectively. For 2-Unit, SX box >= 59 GHz, set the common outputs to CH1, CH3 respectively. ■ Connect the Ref Clock Positive and Negative from CLB to CH3, CH4 of oscilloscope. For 2-Unit, SX box >=59GHz, set the Ref Clock Positive and Negative to CH2 and CH4 respectively.

File name	Configuration details
PCE_Keithley_AIC_x12	Recall this file for Keithley S46T RF Switch and Add-In-Card device type. The below are the configuration details: ■ Lane0 Positive to Lane05 Positive connected as Signal Inputs to Relay A. ■ Lane0 Negative to Lane05 Negative connected as Signal Inputs to Relay B. ■ Lane08 Positive to Lane11 Positive connected as Signal Inputs to Relay C. ■ Lane08 Negative to Lane11 Negative connected as Signal Inputs to Relay D. ■ Common output of relay A, B, C ,D is connected to CH1, CH2, CH3, CH4 of the oscilloscope respectively. For 2-Unit, SX box >= 59 GHz, set the common outputs to CH1, CH3, CH2 and CH4 respectively.
PCE_Giga_AIC_x16	Recall this file for Gigatronics ASCOR 8000 Series RF Switch and Add-In-Card device type. The below are the configuration details: ■ Lane0 Positive to Lane07 Positive connected as Signal Inputs to Relay S1. ■ Lane0 Negative to Lane07 Negative connected as Signal Inputs to Relay S3. ■ Lane08 Positive to Lane15 Positive connected as Signal Inputs to Relay S4. ■ Lane08 Negative to Lane15 Negative connected as Signal Inputs to Relay S6. ■ Common output of relay S1, S2, S3 , S4 is connected to CH1, CH2 , CH3, CH4 of the oscilloscope respectively. For 2-Unit, SX box >= 59 GHz, set the common outputs to CH1, CH3, CH2 and CH4 respectively.

Index

G

H

I

K

L

M

N

O

P

Q

R

S

T

U

V

W