TekExpress® PCI Express Transmitter Compliance and Testing Solution Software

Printable Application Help

077-0774-01

Tektronix

TekExpress® PCI Express
Transmitter Compliance and Testing Solution Software

Printable Application Help

www.tektronix.com

077-0774-01

Tektronix

TekExpress PCIe Compliance and Debug Solution Online Help, 076-0294-01.

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:

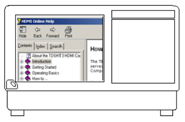
- In North America, call 1-800-833-9200.
- Worldwide, visit www.tektronix.com to find contacts in your area.

Table of Contents

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
Online Help	In-depth operation and UI help	
PDF of the Online Help	Printable version of the compiled Online help	

See also

Technical support (see page 2)

Conventions used in help

Online 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.

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

Minimum system requirements

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

Table 2: System requirements

Oscilloscope	See Supported oscilloscopes (see page 4)
Processor	Same as the oscilloscope
Operating system	Microsoft Windows 7 (64-bit only) Required Windows 7 user account settings (see page 5)
Memory	Same as the oscilloscope
Hard disk	Same as the oscilloscope
Display	Same as the oscilloscope [1]
Firmware	TekScope 6.8.1 and later (Windows 7)
Software	■ DPOJET, Jitter and Eye Diagram Analysis Tool (version 6.0.2 or greater for Windows 7 ■ Microsoft .NET 4.0 Framework ■ Microsoft Internet Explorer 6.0 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
Arbitrary Function Generator (AFG) [2] (for automatic test pattern toggling)	■ Tektronix AFG3252, AFG3252C

Table 2: System requirements (cont.)

Arbitrary Waveform Generator (AWG) [2] (for automatic test pattern toggling)	■ Tektronix AWG5002B/C, AWG5012B/C, AWG5014B/C ■ Tektronix AWG7082B/C, AWG7122B/C ■ Tektronix AWG70001A, AWG70002A
Other devices	■ SMP-SMA cables ■ TCA-SMA connectors ■ Matched pair cables ■ Tektronix P7313 SMA Differential Probe

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

Supported oscilloscopes

The TekExpress PCIe application runs on the following Tektronix oscilloscopes:

- MSO70604 [1]; DPO/MSO70604C (Gen1 testing only)
- MSO70804 [1]; DPO/MSO70804C (Gen1 and Gen2 testing only)
- MSO71254 [1]; DPO/MSO71254C (Gen1, Gen2, and Gen3 testing)
- MSO71604 [1]; DPO/MSO71604C (Gen1, Gen2, and Gen3 testing)
- MSO72004 [1]; DPO/MSO72004C (Gen1, Gen2, and Gen3 testing)
- DPO/MSO72304DX (Gen1, Gen2, and Gen3 testing)
- DPO/MSO72504DX (Gen1, Gen2 and Gen3 testing)
- DPO/MSO73304DX (Gen1, Gen2 and Gen3 testing)

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

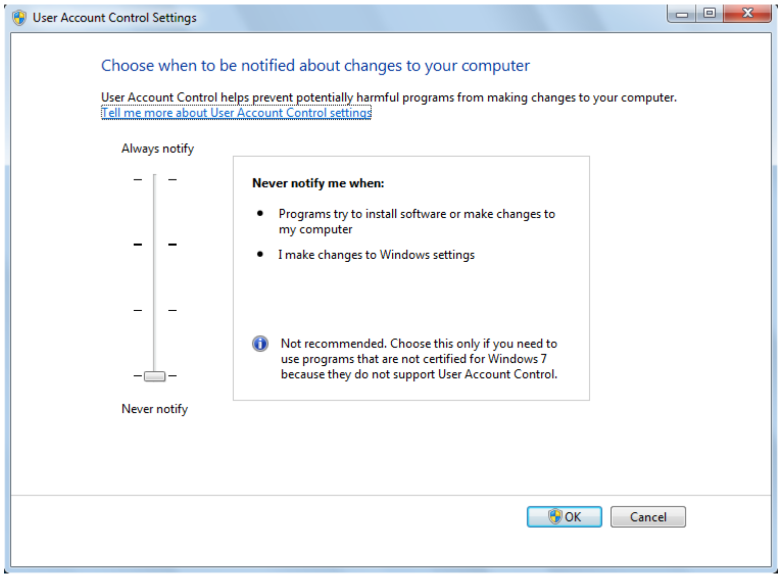
See also

Minimum system requirements (see page 3)

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.



User Account Control Settings

Choose when to be notified about changes to your computer

User Account Control helps prevent potentially harmful programs from making changes to your computer.
Tell me more about User Account Control settings

Always notify

Never notify me when:

- Programs try to install software or make changes to my computer
- I make changes to Windows settings

Not recommended. Choose this only if you need to use programs that are not certified for Windows 7 because they do not support User Account Control.

Never notify

See also

Supported oscilloscopes (see page 4)

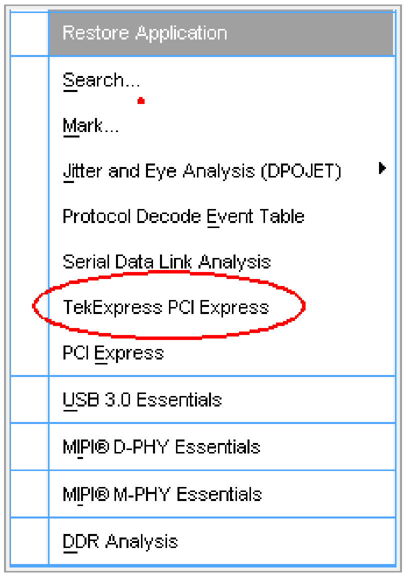
Install the software

Use the following steps to install PCI Express software on any compatible instrument running Microsoft Windows 7 (64-bit). See Minimum System Requirements (see page 3) 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 (see page 3)
Supported oscilloscopes (see page 4)

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 (see page 9) 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.

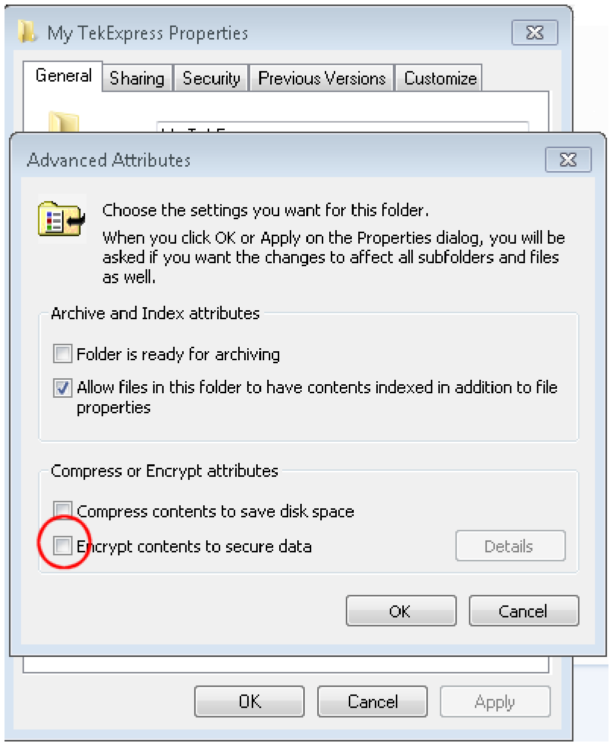
NOTE. *If the X: drive is mapped to any other shared folder, the application will display a warning message asking you to disconnect the X: drive manually.*

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.


My TekExpress Properties

General | Sharing | Security | Previous Versions | Customize

Advanced Attributes

Choose the settings you want for this folder.

When you click OK or Apply on the Properties dialog, you will be asked if you want the changes to affect all subfolders and files as well.

Archive and Index attributes

- [ ] Folder is ready for archiving
- [✓] Allow files in this folder to have contents indexed in addition to file properties

Compress or Encrypt attributes

- [ ] Compress contents to save disk space
- [ ] Encrypt contents to secure data

4. See the prerun checklist (see page 52) before you run a test.

See also

Configuration test parameters (see page 37)
View test-related files (see page 42)
Application directories and usage (see page 12)
File name extensions (see page 14)

Map the My TekExpress folder

Follow these steps to map the My TekExpress folder on the instrument:

1. Open Windows Explorer.
2. From the Windows Explorer menu, click **Computer**.
3. In the menu bar, select **Map network drive**.
4. Select the Drive letter as **X:** (if there is any previous connection on X:, disconnect it first through **Tools > Disconnect Network drive** menu of Windows Explorer. Windows 7 users: if you do not see the Tools menu, press the **Alt** key).
5. In the Folder field, enter the remote `My TekExpress` folder path (for example, `\\192.158.97.65\My TekExpress`).

To determine the IP address of the instrument where the My TekExpress folder exists, do the following:

1. On the instrument where the `My TekExpress` folder exists, click **Start** and select **Run**.
2. Type **cmd** and then press **Enter**.
3. At the command prompt, type **ipconfig** and then press **Enter**.

See also

Before you click start (see page 6)
Install the software (see page 6)

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

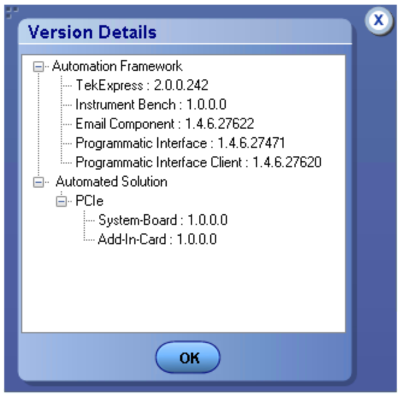
View version and license information (see page 10)

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**.
2. Click the View Version Details link to view the version numbers of the installed test suites.



```
Automation Framework
    TekExpress : 2.0.0.242
    Instrument Bench : 1.0.0.0
    Email Component : 1.4.6.27622
    Programmatic Interface : 1.4.6.27471
    Programmatic Interface Client : 1.4.6.27620
Automated Solution
    PCIe
        System-Board : 1.0.0.0
        Add-In-Card : 1.0.0.0
```

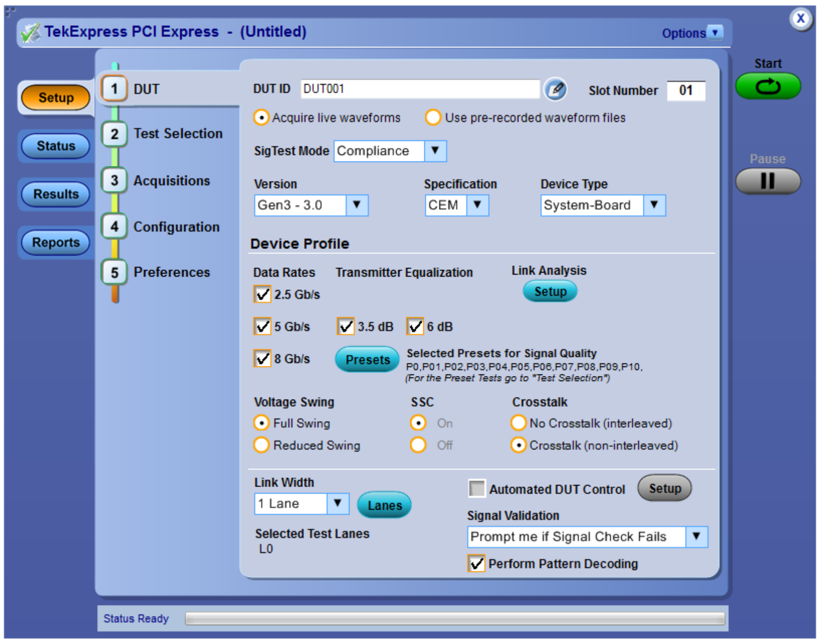
To view license and option key information:

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 (see page 9)
Options menu (see page 16)

TekExpress® PCI Express features and benefits



DUT ID DUT001 Slot Number 01

[✓ Acquire live waveforms] Use pre-recorded waveform files

SigTest Mode Compliance

Version Gen3 - 3.0 Specification CEM Device Type System-Board

Device Profile

Data Rates: [✓ 2.5 Gb/s] [✓ 5 Gb/s] [✓ 8 Gb/s]

Transmitter Equalization: [✓ 3.5 dB] [✓ 6 dB]

Link Analysis

Selected Presets for Signal Quality
P0,P01,P02,P03,P04,P05,P06,P07,P08,P09,P10,
(For the Preset Tests go to "Test Selection")

Voltage Swing: [✓ Full Swing] Reduced Swing

SSC: [✓ On] Off

Crosstalk: No Crosstalk (interleaved) [✓ Crosstalk (non-interleaved)]

Link Width 1 Lane

Selected Test Lanes
L0

Automated DUT Control

Signal Validation
Prompt me if Signal Check Fails

[✓ Perform Pattern Decoding]

Status Ready

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.

Key features and benefits of PCIe include:

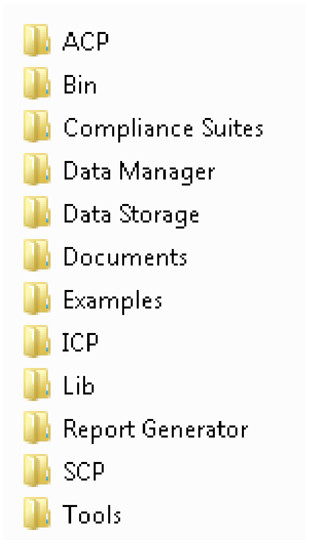
- Automates compliance measurements for PCI Express 3.0 CEM Specification Rev 0.9. for the following configurations:
  - PCIE_1_0a- PCIEX_TX_ADD_CON_250UI
  - PCIE_1_0a- PCIEX_TX_SYS_CON_250UI
  - PCIE_CEM_CARD_1_1
  - PCIE_CEM_SYS_1_1
  - PCIE_2_0_CARD
  - PCIE_2_0_SYS
  - PCIE_3_0_CARD
  - PCIE_3_0_SYS
- 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

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`.



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

Table 3: 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
Report Generator	Contains style sheets for report generation
SCP	Contains instrument and PCIe application-specific interface libraries
Tools	Contains instrument and PCIe application-specific files

TekExpress TekSig

The TekExpress TekSig application files are installed at `C:\Program Files (x86)\Tektronix\Tek-SigPackage`.

See also

View test-related files (see page 42)
File name extensions (see page 14)

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 (see page 43).
.flt	Filter files.

See also

Select report options (see page 43)
View test-related files (see page 42)
Application directories and their contents (see page 12)
Before you click start (see page 6)

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 (see page 16). 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 (see page 9)

Exit the application

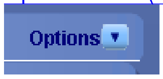
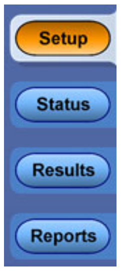
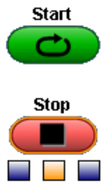
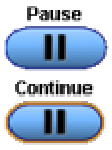
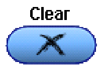
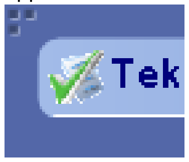
Use the following method to exit the application:

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**.

Application controls

Table 4: Application controls descriptions

Item	Description
Options menu (see page 16) 	Menu to display global application controls
Panel buttons (see page 23) 	Controls that open panels for configuring test settings and options.
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 panel (see page 40).
Application window move icon 	Place the cursor over the three-dot pattern in the upper left corner of the application window. When the cursor changes to a hand, drag the window to the desired location.

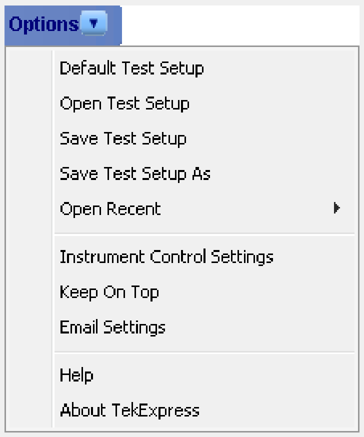
Options menu overview

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

The Options menu (see page 17) 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
Save Test Setup As	Creates a new test setup based on an existing one
Open Recent	Displays a menu of recently opened test setups to select from
Instrument Control Settings (see page 18)	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 (see page 20)	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 (see page 10) for your PCIe installation ■ Provides a link to the Tektronix Web site

Options menu



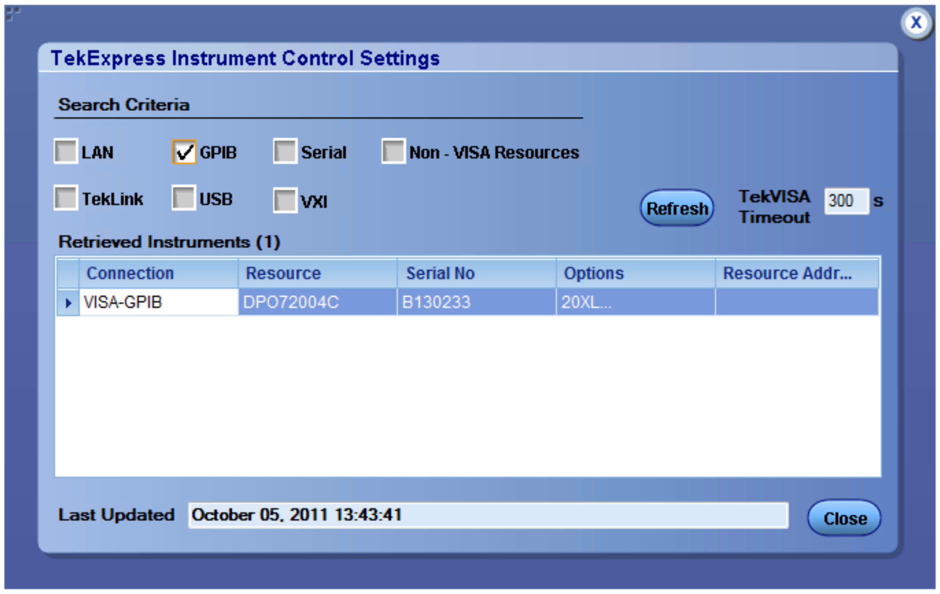
See also

Application controls (see page 16)

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.



TekVISA Timeout 300 s

Retrieved Instruments (1)

Connection	Resource	Serial No	Options	Resource Addr...
VISA-GPIB	DPO72004C	B130233	20XL...	

Last Updated October 05, 2011 13:43:41

Use the Instrument Control Settings feature to search for connected instruments (see page 18) 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 (see page 16)

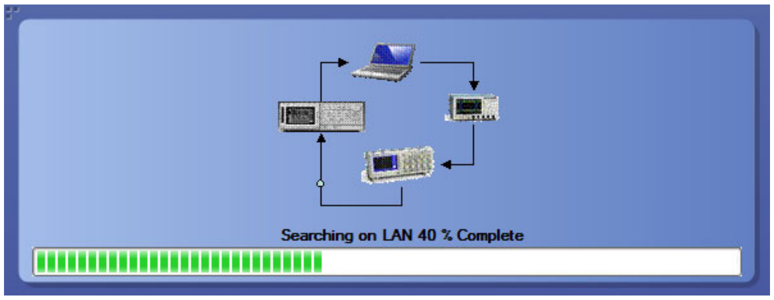
View connected instruments

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.

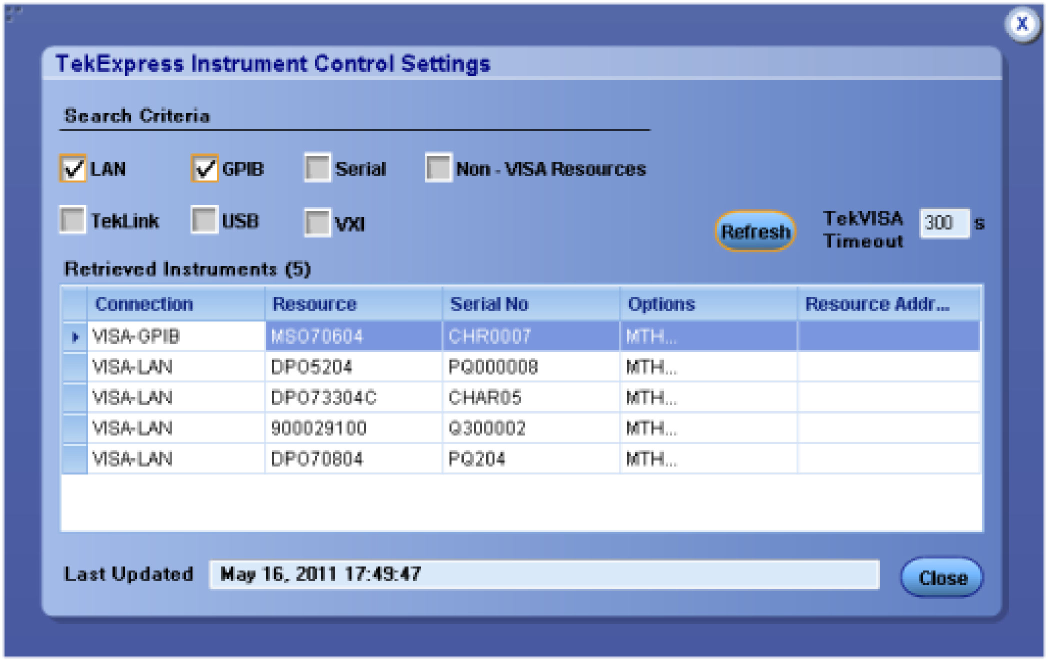
To refresh the list of connected instruments:

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.



TekExpress Instrument Control Settings

Search Criteria

[✓ LAN] [✓ GPIB] Serial Non - VISA Resources

TekLink USB VXI

TekVISA Timeout 300 s

Retrieved Instruments (5)

Connection	Resource	Serial No	Options	Resource Addr...
VISA-GPIB	MSO70604	CHR0007	MTH...	
VISA-LAN	DPO5204	PQ000008	MTH...	
VISA-LAN	DPO73304C	CHAR05	MTH...	
VISA-LAN	900029100	Q300002	MTH...	
VISA-LAN	DPO70804	PQ204	MTH...	

Last Updated May 16, 2011 17:49:47

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

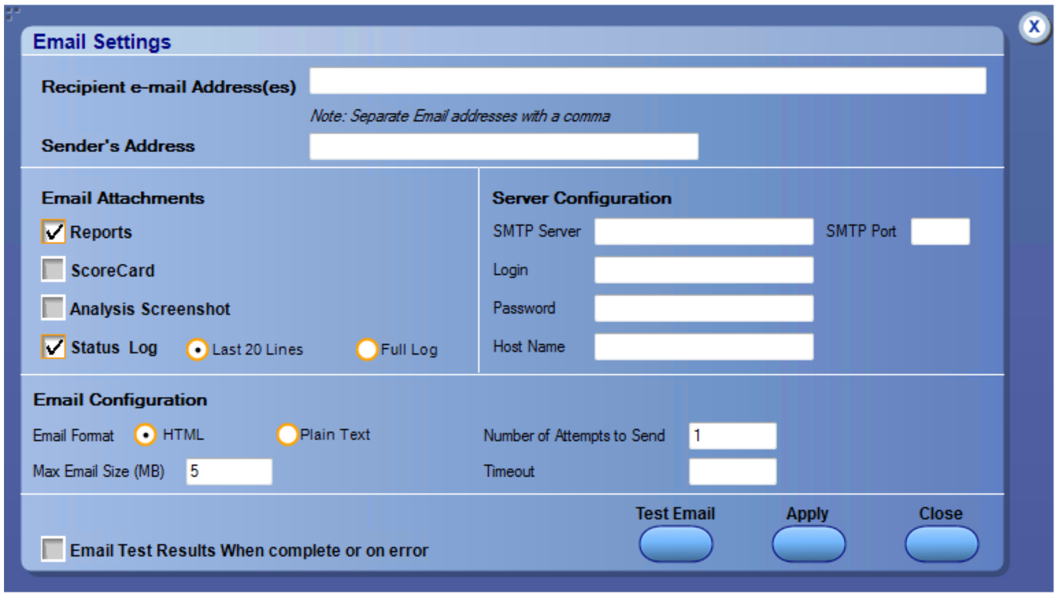
Configuration test parameters (see page 37)
Equipment connection setup (see page 50)

Email settings

Use the Email Settings utility to configure email notifications (see page 21) 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.*


Email Settings

Recipient e-mail Address(es)

Note: Separate Email addresses with a comma

Sender's Address

Email Attachments

[✓] Reports
[ ] ScoreCard
[ ] Analysis Screenshot
[✓] Status Log [✓ Last 20 Lines] Full Log

Server Configuration

SMTP Server SMTP Port
Login
Password
Host Name

Email Configuration

Email Format [✓ HTML] Plain Text
Max Email Size (MB) 5
Number of Attempts to Send 1
Timeout

[ ] Email Test Results When complete or on error

See also

Configure email settings (see page 21)
Options menu (see page 16)

Select test notification preferences (see page 37)

Configure email settings

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

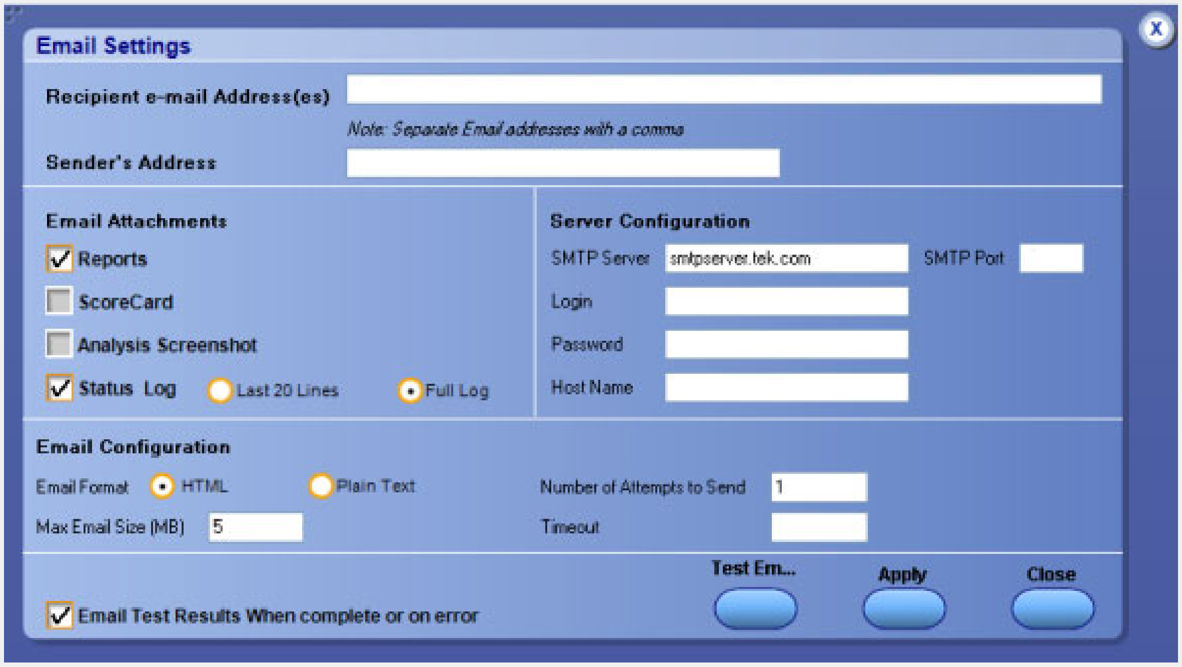
1. **Options** > **Email Settings** to open the Email Settings (see page 22) dialog box.
2. (Required) For Recipient email Address(es), enter one or more email addresses to which to send the test notification. To include multiple addresses, separate the addresses with commas.
3. (Required) For Sender's 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



Recipient e-mail Address(es)

Note: Separate Email addresses with a comma

Sender's Address

Email Attachments

[✓] Reports

[ ] ScoreCard

[ ] Analysis Screenshot

[✓] Status Log — Last 20 Lines [✓ Full Log]

Server Configuration

SMTP Server: smtpserver.tek.com — SMTP Port

Login

Password

Host Name

Email Configuration

Email Format: [✓ HTML] Plain Text

Number of Attempts to Send: 1

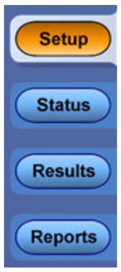
Max Email Size (MB): 5

Timeout

[✓] Email Test Results When complete or on error

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 5: Application panels

Panel Name	Purpose
Setup (see page 24)	The Setup panel shows the test setup controls. Click the **Setup** button to open this panel. Use this panel to: ■ Select DUTparameters (see page 24). ■ Select the test(s) (see page 30). ■ Set acquisitions parameters (see page 30) for selected tests. ■ Configuration test parameters (see page 37) ■ Select test notification preferences (see page 37).
Status (see page 39)	View the progress and analysis status of the selected tests, and view test logs.
Results (see page 40)	View a summary of test results and select result viewing preferences.
Reports (see page 43)	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.

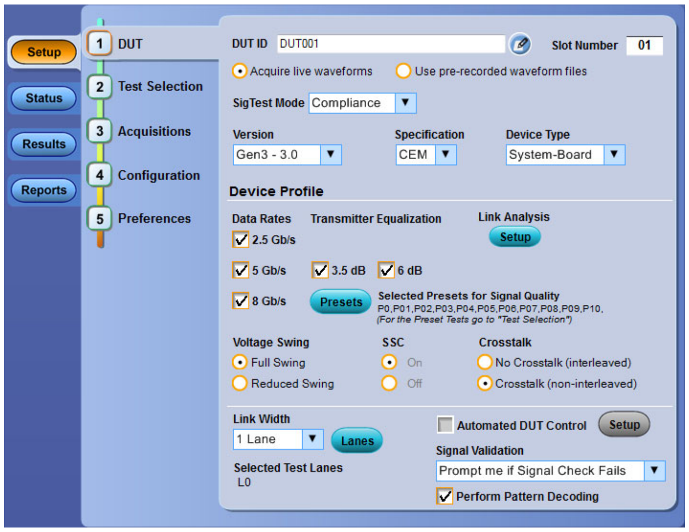
See also

Application controls (see page 16)

about setting up tests (see page 49)

Setup panel overview

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



DUT ID DUT001 Slot Number 01

[✓ Acquire live waveforms] Use pre-recorded waveform files

SigTest Mode Compliance

Version Gen3 - 3.0 Specification CEM Device Type System-Board

Device Profile

Data Rates: ✓ 2.5 Gb/s ✓ 5 Gb/s ✓ 8 Gb/s

Transmitter Equalization: ✓ 3.5 dB ✓ 6 dB

Link Analysis

Selected Presets for Signal Quality
P0,P01,P02,P03,P04,P05,P06,P07,P08,P09,P10,
(For the Preset Tests go to "Test Selection")

Voltage Swing: [✓ Full Swing] Reduced Swing

SSC: [✓ On] Off

Crosstalk: No Crosstalk (interleaved) [✓ Crosstalk (non-interleaved)]

Link Width 1 Lane

Selected Test Lanes
L0

Automated DUT Control

Signal Validation
Prompt me if Signal Check Fails

✓ Perform Pattern Decoding

Use the tabs on this panel to:

Set the DUT parameters (see page 24)

Select test(s) (see page 30)

Select acquisition parameters (see page 30)

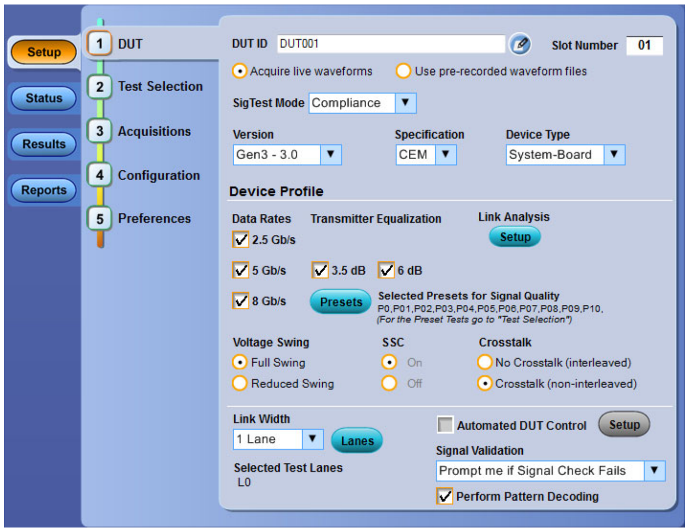
Set configuration parameters (see page 37)

Select test notification preferences (see page 37)

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.



DUT ID: DUT001 Slot Number: 01

[✓ Acquire live waveforms] Use pre-recorded waveform files

SigTest Mode: Compliance

Version: Gen3 - 3.0 Specification: CEM Device Type: System-Board

Device Profile

Data Rates: [✓ 2.5 Gb/s] [✓ 5 Gb/s] [✓ 8 Gb/s]

Transmitter Equalization: [✓ 3.5 dB] [✓ 6 dB]

Link Analysis

Selected Presets for Signal Quality
P0,P01,P02,P03,P04,P05,P06,P07,P08,P09,P10,
(For the Preset Tests go to "Test Selection")

Voltage Swing: [✓ Full Swing] Reduced Swing

SSC: [✓ On] Off

Crosstalk: No Crosstalk (interleaved) [✓ Crosstalk (non-interleaved)]

Link Width: 1 Lane

Selected Test Lanes
L0

Automated DUT Control

Signal Validation: Prompt me if Signal Check Fails

[✓ Perform Pattern Decoding]

Table 6: DUT tab settings

Setting	Description
DUT ID	Adds an optional text label for the DUT to reports. The default value is DUT001.
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 (see page 43).)
Acquire live waveforms	Acquire active signals from the DUT for testing.
Use prerecorded waveform files	Run tests on a saved waveform. Open (load) a saved test setup (see page 54)
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.
Version	Sets the DUT generation version. Available versions are Gen 1 (1.0a and 1.1), Gen2 (2.0), and Gen3 (3.0)
Specification	PCIe supports the CEM specification.
Device Type	Sets the DUT device type (System-Board or Add-in-Card).
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.

Table 6: DUT tab settings (cont.)

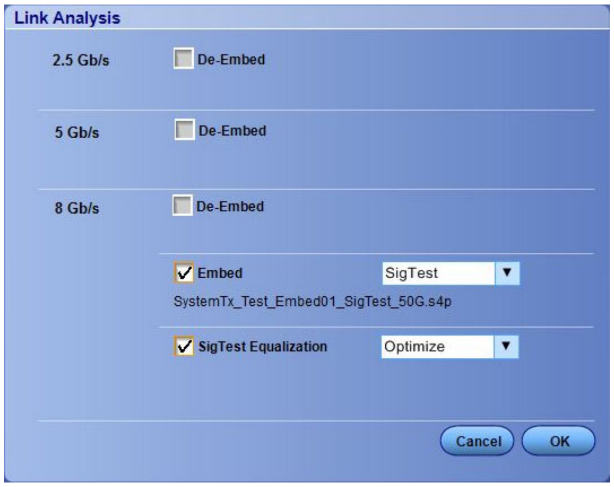
Setting	Description
Transmitter Equalization	Sets transmitter preemphasis levels. Available for Gen 2 and Gen 3 devices. The application selects both preemphasis levels by default when in the compliance mode for an Add-in-Card. At least one preemphasis level must be selected.
Link Analysis	Opens the Link Analysis dialog box to select custom filter files with which to perform link analysis on the source waveforms. Link Analysis dialog box (see page 27)
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.
Voltage Swing	Sets the lane/link transmitter p-p voltage swing.
SSC (spread spectrum clocking)	Enables or disables SSC clocking.
Crosstalk	Sets specific eye test limits depending on if the DUT design uses interleaved or noninterleaved routing. ■ When the DUT uses noninterleaved routing, select **Crosstalk (noninterleaved routing)**. ■ When the DUT uses interleaved routing, select **No Crosstalk (interleaved routing)**.
Link Width	Sets the link width in Lanes (1,2,4,8, or 16).
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 (see page 28)
Automated DUT Control	Enables automatic toggling of the DUT into different test modes (generation/equalization). Requires the use of an AFG or AWGinstrument. Click **Setup** to access the Automated DUT Control dialog box (see page 29)
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.

See also

About setting up tests (see page 49)
Select a test (see page 30)

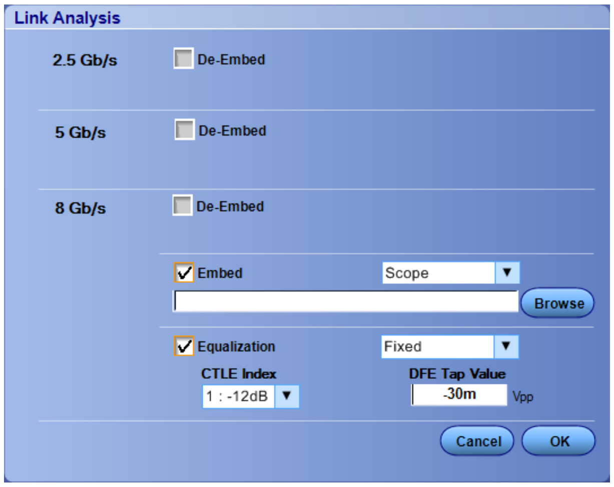
Link Analysis dialog box

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.



SystemTx_Test_Embed01_SigTest_50G.s4p

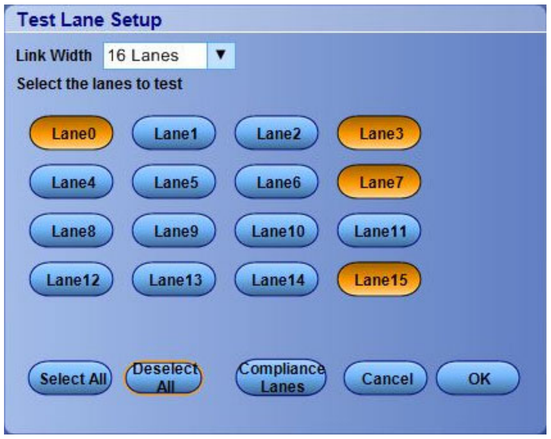
- For 2.5 Gb/s and 5 Gb/s data rates, only the De-embed option is available.
- There are two options for the 8 Gb/s data rate Embed mode: “Scope” and “SigTest.” This selection effects the fields shown in the Equalization selections.
- There are two options for the 8 Gb/s data rate equalization mode: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



1 : -12dB
-30m

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.



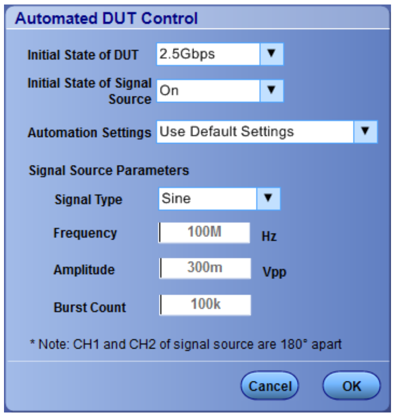
Link Width 16 Lanes

Select the lanes to test

- 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: 2.5Gbps
Initial State of Signal Source: On
Automation Settings: Use Default Settings
Signal Source Parameters
Signal Type: Sine
Frequency: 100M Hz
Amplitude: 300m Vpp
Burst Count: 100k
* Note: CH1 and CH2 of signal source are 180° apart

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.

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.*

Select tests

Use the **Test Selection** tab to select the Signal Quality 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** to deselect all tests. All tests are selected by default.
   - Click **Select Required** to select all tests that are required for compliance.
   - Click **Select All** to select all tests.
   - Click **Schematic** 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

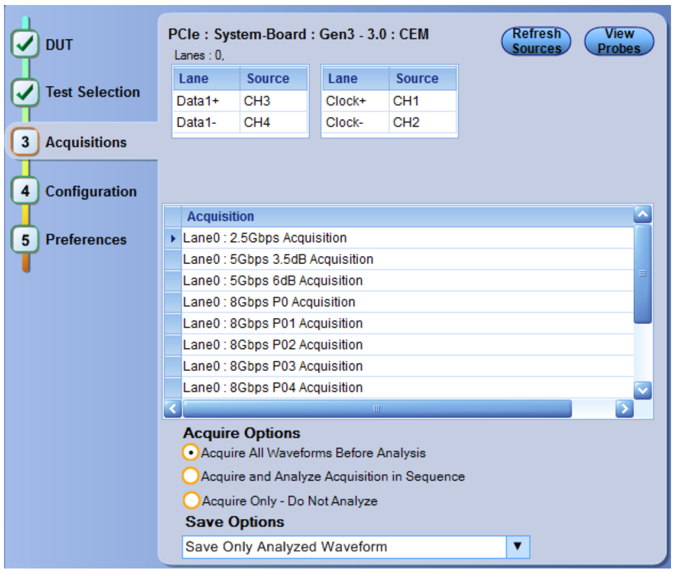
Set acquisition parameters (see page 30)
About setting up tests (see page 49)

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



PCIe : System-Board : Gen3 - 3.0 : CEM

Lanes : 0,

Lane	Source
Data1+	CH3
Data1-	CH4

Lane	Source
Clock+	CH1
Clock-	CH2

Acquisition
Lane0 : 2.5Gbps Acquisition
Lane0 : 5Gbps 3.5dB Acquisition
Lane0 : 5Gbps 6dB Acquisition
Lane0 : 8Gbps P0 Acquisition
Lane0 : 8Gbps P01 Acquisition
Lane0 : 8Gbps P02 Acquisition
Lane0 : 8Gbps P03 Acquisition
Lane0 : 8Gbps P04 Acquisition

Acquire Options

[✓ Acquire All Waveforms Before Analysis]

Acquire and Analyze Acquisition in Sequence

Acquire Only - Do Not Analyze

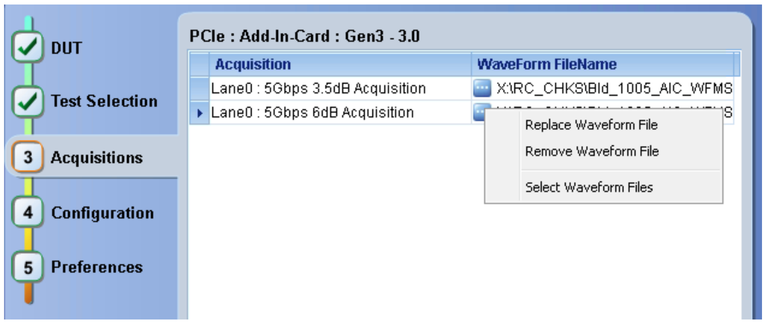
Save Options

Save Only Analyzed Waveform

Acquisitions tab: using active waveforms

- Click the Source (see page 32) 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.
- Click the Acquire Options (see page 33) controls to set how the application acquires and analyzes signals.
- Click the Save Options (see page 33) 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



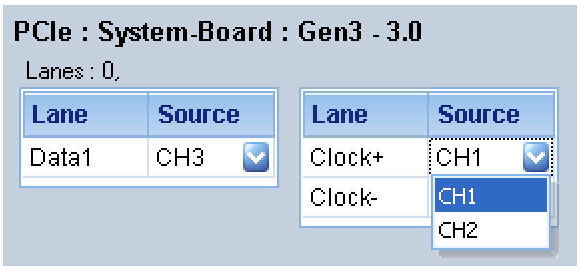
Acquisitions tab: using prerecorded waveforms

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 (see page 34).

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 (see page 33)
Set acquisition waveform save options (see page 33)
Set acquisition waveform source for prerecorded waveform files (see page 34)

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 (see page 34) mode.

See also

Set acquisitions signal source (see page 32)
Set acquisition waveform save options (see page 33)

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.

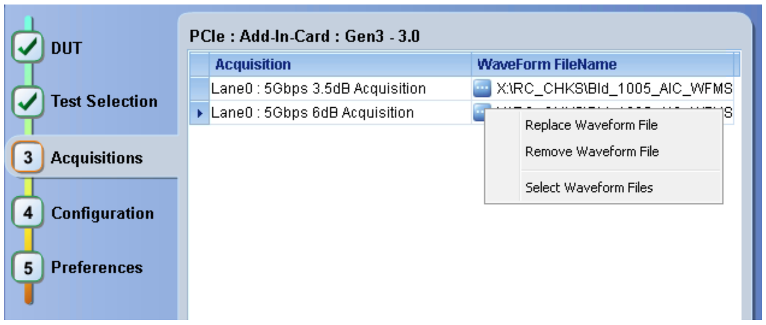
Waveforms are saved to a folder that is unique to each session (a session starts when you click the Start button). The folder path is `X:\PCI Express\Untitled Session\<dutid>\<date>_<time>`. Images created for each analysis, CSV files with result values, reports, and other information specific to that 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 (see page 32)
Set acquisition waveform source for prerecorded waveform files (see page 34)
Set acquisition Acquire options (see page 33)

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.



Acquisition	WaveForm FileName
Lane0 : 5Gbps 3.5dB Acquisition	X:\RC_CHKS\Bld_1005_AIC_WFMS
Lane0 : 5Gbps 6dB Acquisition	

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.*

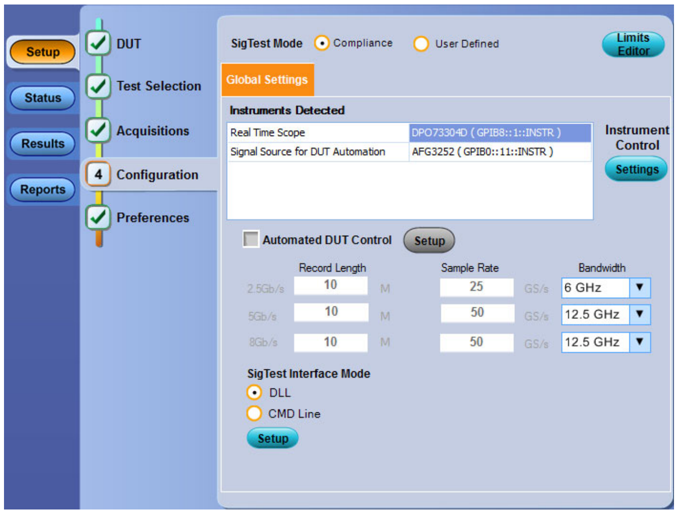
See also

Set acquisition signal source (see page 32)
Set acquisition options (see page 33)
Set acquisition waveform save options (see page 33)

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.*



SigTest Mode [✓ Compliance] User Defined

Instruments Detected

Real Time Scope	DPO73304D (GPIB8::1::INSTR)
Signal Source for DUT Automation	AFG3252 (GPIB0::11::INSTR)

Automated DUT Control

	Record Length		Sample Rate		Bandwidth
2.5Gb/s	10	M	25	GS/s	6 GHz
5Gb/s	10	M	50	GS/s	12.5 GHz
8Gb/s	10	M	50	GS/s	12.5 GHz

SigTest Interface Mode
[✓ DLL] CMD Line

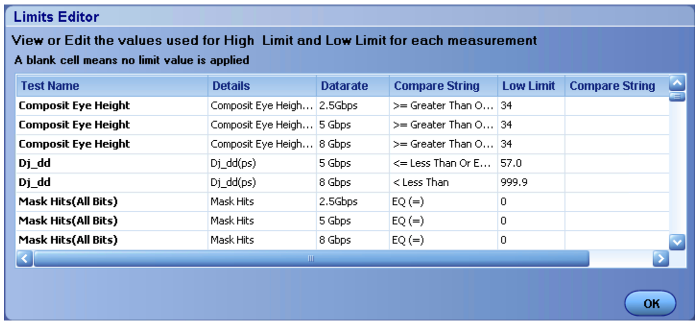
See also

Configuration tab parameters (see page 37)
About setting up tests (see page 49)
About running tests (see page 51)

Configuration tab parameters

The following table lists the Configuration tab settings and parameters.

Table 7: 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. Select **Instrument Control Settings** to refresh the connected instrument list (see page 18).
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.

Limits Editor

View or Edit the values used for High Limit and Low Limit for each measurement

A blank cell means no limit value is applied

Test Name	Details	Datarate	Compare String	Low Limit	Compare String
Composit Eye Height	Composit Eye Heigh…	2.5Gbps	>= Greater Than O…	34	
Composit Eye Height	Composit Eye Heigh…	5 Gbps	>= Greater Than O…	34	
Composit Eye Height	Composit Eye Heigh…	8 Gbps	>= Greater Than O…	34	
Dj_dd	Dj_dd(ps)	5 Gbps	<= Less Than Or E…	57.0	
Dj_dd	Dj_dd(ps)	8 Gbps	< Less Than	999.9	
Mask Hits(All Bits)	Mask Hits	2.5Gbps	EQ (=)	0	
Mask Hits(All Bits)	Mask Hits	5 Gbps	EQ (=)	0	
Mask Hits(All Bits)	Mask Hits	8 Gbps	EQ (=)	0	

Table 7: Configuration tab parameters (cont.)

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.
SigTest Interface Mode	Sets whether to use a Dynamic Link Library (DLL) or command line interface for running Sigtest. 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 acquisitions (see page 30)
De-embed using filter files (see page 119)

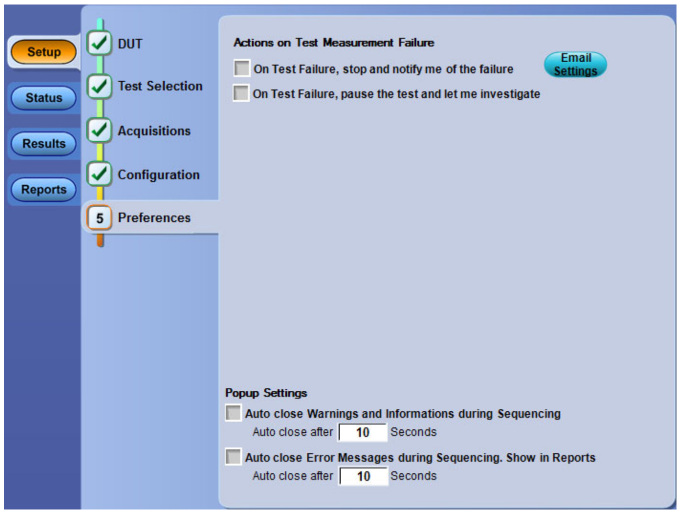
Set test notification preferences

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, 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.
   - 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**.

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.



Actions on Test Measurement Failure

- [ ] On Test Failure, stop and notify me of the failure
- [ ] On Test Failure, pause the test and let me investigate

Popup Settings

- [ ] Auto close Warnings and Informations during Sequencing
  Auto close after 10 Seconds
- [ ] Auto close Error Messages during Sequencing. Show in Reports
  Auto close after 10 Seconds

Table 8: Preferences tab parameters

Parameter	Description
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.
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.
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.

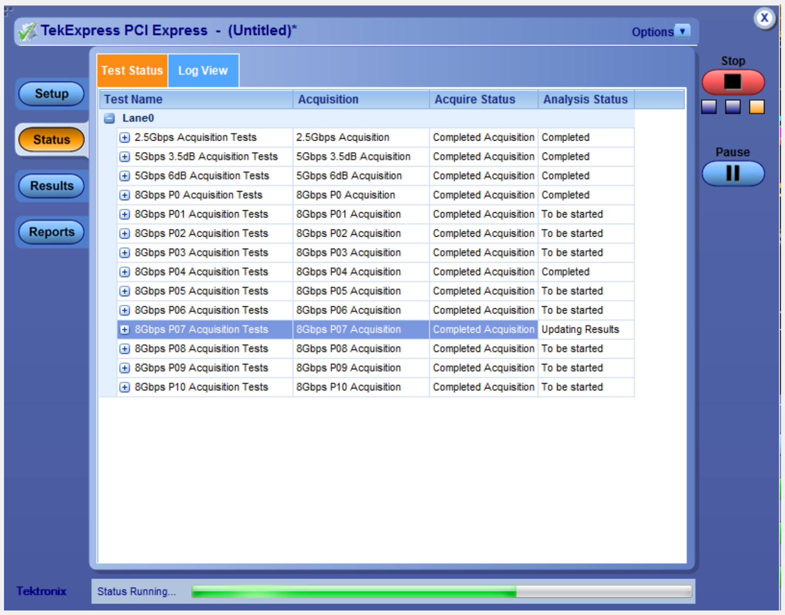
See also

About setting up tests (see page 49)
Select report options (see page 43)

Status panel overview

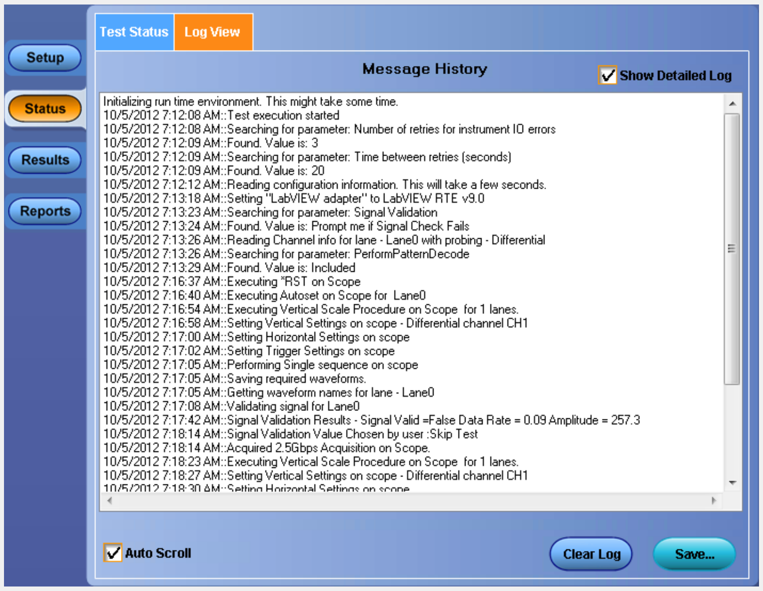
The Status panel provides status on test acquisition and analysis (Test Status (see page 39) tab) and a listing of test tasks performed (Log View (see page 40) tab). The application opens the Test Status tab when you start a test run. You can select the Test Status or the Log View tab to view these items while tests are running.

Test status view



Test Name	Acquisition	Acquire Status	Analysis Status
Lane0			
2.5Gbps Acquisition Tests	2.5Gbps Acquisition	Completed Acquisition	Completed
5Gbps 3.5dB Acquisition Tests	5Gbps 3.5dB Acquisition	Completed Acquisition	Completed
5Gbps 6dB Acquisition Tests	5Gbps 6dB Acquisition	Completed Acquisition	Completed
8Gbps P0 Acquisition Tests	8Gbps P0 Acquisition	Completed Acquisition	Completed
8Gbps P01 Acquisition Tests	8Gbps P01 Acquisition	Completed Acquisition	To be started
8Gbps P02 Acquisition Tests	8Gbps P02 Acquisition	Completed Acquisition	To be started
8Gbps P03 Acquisition Tests	8Gbps P03 Acquisition	Completed Acquisition	To be started
8Gbps P04 Acquisition Tests	8Gbps P04 Acquisition	Completed Acquisition	Completed
8Gbps P05 Acquisition Tests	8Gbps P05 Acquisition	Completed Acquisition	To be started
8Gbps P06 Acquisition Tests	8Gbps P06 Acquisition	Completed Acquisition	To be started
8Gbps P07 Acquisition Tests	8Gbps P07 Acquisition	Completed Acquisition	Updating Results
8Gbps P08 Acquisition Tests	8Gbps P08 Acquisition	Completed Acquisition	To be started
8Gbps P09 Acquisition Tests	8Gbps P09 Acquisition	Completed Acquisition	To be started
8Gbps P10 Acquisition Tests	8Gbps P10 Acquisition	Completed Acquisition	To be started

Log view



```
Initializing run time environment. This might take some time.
10/5/2012 7:12:08 AM::Test execution started
10/5/2012 7:12:08 AM::Searching for parameter: Number of retries for instrument IO errors
10/5/2012 7:12:09 AM::Found. Value is: 3
10/5/2012 7:12:09 AM::Searching for parameter: Time between retries (seconds)
10/5/2012 7:12:09 AM::Found. Value is: 20
10/5/2012 7:12:12 AM::Reading configuration information. This will take a few seconds.
10/5/2012 7:13:18 AM::Setting "LabVIEW adapter" to LabVIEW RTE v9.0
10/5/2012 7:13:23 AM::Searching for parameter: Signal Validation
10/5/2012 7:13:24 AM::Found. Value is: Prompt me if Signal Check Fails
10/5/2012 7:13:26 AM::Reading Channel info for lane - Lane0 with probing - Differential
10/5/2012 7:13:26 AM::Searching for parameter: PerformPatternDecode
10/5/2012 7:13:29 AM::Found. Value is: Included
10/5/2012 7:16:37 AM::Executing *RST on Scope
10/5/2012 7:16:40 AM::Executing Autoset on Scope for  Lane0
10/5/2012 7:16:54 AM::Executing Vertical Scale Procedure on Scope  for 1 lanes.
10/5/2012 7:16:58 AM::Setting Vertical Settings on scope - Differential channel CH1
10/5/2012 7:17:00 AM::Setting Horizontal Settings on scope
10/5/2012 7:17:02 AM::Setting Trigger Settings on scope
10/5/2012 7:17:05 AM::Performing Single sequence on scope
10/5/2012 7:17:05 AM::Saving required waveforms.
10/5/2012 7:17:05 AM::Getting waveform names for lane - Lane0
10/5/2012 7:17:08 AM::Validating signal for Lane0
10/5/2012 7:17:42 AM::Signal Validation Results - Signal Valid =False Data Rate = 0.09 Amplitude = 257.3
10/5/2012 7:18:14 AM::Signal Validation Value Chosen by user :Skip Test
10/5/2012 7:18:14 AM::Acquired 2.5Gbps Acquisition on Scope.
10/5/2012 7:18:23 AM::Executing Vertical Scale Procedure on Scope  for 1 lanes.
10/5/2012 7:18:27 AM::Setting Vertical Settings on scope - Differential channel CH1
10/5/2012 7:18:30 AM::Setting Horizontal Settings on scope
```

The Log View display has several viewing options:

- Message History: This window timestamps and displays all run messages.
- Show Detailed Log: Select this check box to record a detailed history of test execution.

  This must be checked before starting a measurement.
- Auto Scroll: Select this check box to have the program automatically scroll down as information is added to the log during the test.
- Clear Log: Click this button to clear all messages from the display.
- Save: 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

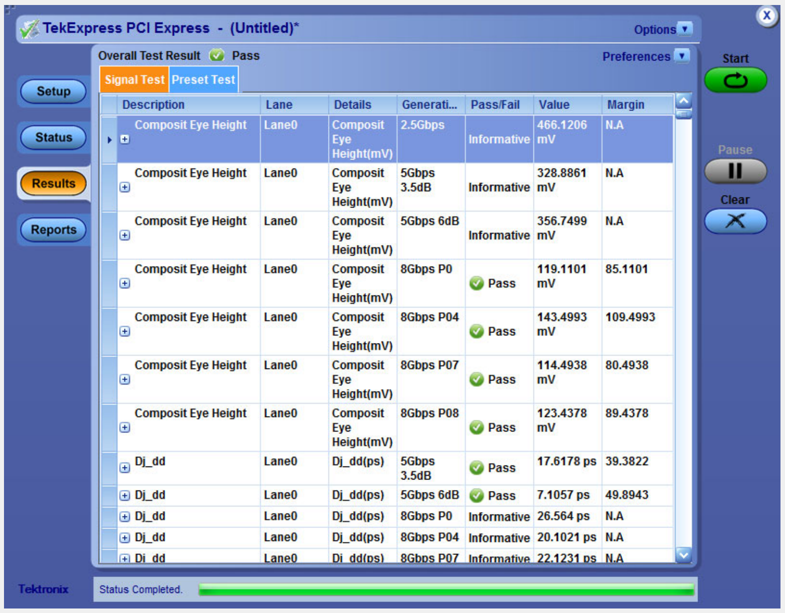
Application panel overview (see page 23)

Results panel overview

When a test finishes, the application switches to the Results panel (see page 41) 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



Overall Test Result ✓ Pass

Description	Lane	Details	Generati...	Pass/Fail	Value	Margin
Composit Eye Height	Lane0	Composit Eye Height(mV)	2.5Gbps	Informative	466.1206 mV	N.A
Composit Eye Height	Lane0	Composit Eye Height(mV)	5Gbps 3.5dB	Informative	328.8861 mV	N.A
Composit Eye Height	Lane0	Composit Eye Height(mV)	5Gbps 6dB	Informative	356.7499 mV	N.A
Composit Eye Height	Lane0	Composit Eye Height(mV)	8Gbps P0	✓ Pass	119.1101 mV	85.1101
Composit Eye Height	Lane0	Composit Eye Height(mV)	8Gbps P04	✓ Pass	143.4993 mV	109.4993
Composit Eye Height	Lane0	Composit Eye Height(mV)	8Gbps P07	✓ Pass	114.4938 mV	80.4938
Composit Eye Height	Lane0	Composit Eye Height(mV)	8Gbps P08	✓ Pass	123.4378 mV	89.4378
Dj_dd	Lane0	Dj_dd(ps)	5Gbps 3.5dB	✓ Pass	17.6178 ps	39.3822
Dj_dd	Lane0	Dj_dd(ps)	5Gbps 6dB	✓ Pass	7.1057 ps	49.8943
Dj_dd	Lane0	Dj_dd(ps)	8Gbps P0	Informative	26.564 ps	N.A
Dj_dd	Lane0	Dj_dd(ps)	8Gbps P04	Informative	20.1021 ps	N.A
Dj_dd	Lane0	Dj_dd(ps)	8Gbps P07	Informative	22.1231 ps	N.A

Status Completed.

When a test finishes, the application switches to the Results panel (see page 40), 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 (see page 45)
About panels (see page 23)

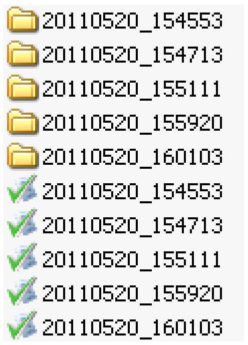
View test-related files

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:


20110520_154553
20110520_154713
20110520_155111
20110520_155920
20110520_160103
20110520_154553
20110520_154713
20110520_155111
20110520_155920
20110520_160103

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.

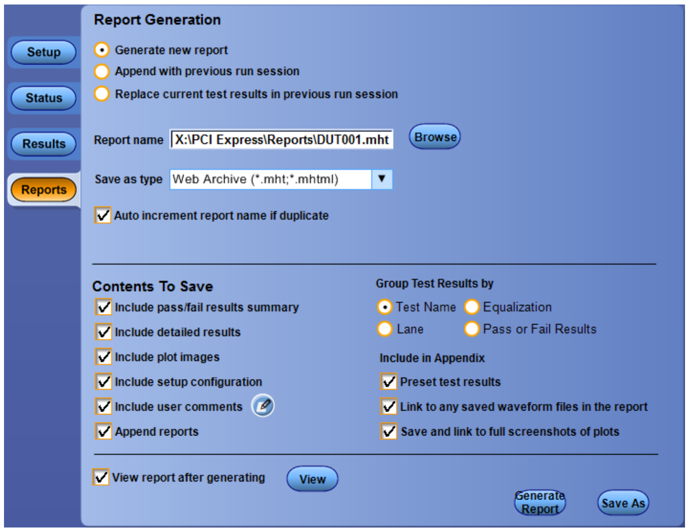
The first time you run a new, unsaved session, the session files are stored in the `Untitled Session` folder located at `..\My TekExpress\PCI Express`. When you name and save the session, the files are placed in a folder with the name that you specify. A copy of the test files stay in the Untitled Session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

See also

File name extensions (see page 14)
Before you click start (see page 6)

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.



Report Generation

[✓ Generate new report]
Append with previous run session
Replace current test results in previous run session

Report name X:\PCI Express\Reports\DUT001.mht

Save as type Web Archive (*.mht;*.mhtml)

✓ Auto increment report name if duplicate

Contents To Save

✓ Include pass/fail results summary
✓ Include detailed results
✓ Include plot images
✓ Include setup configuration
✓ Include user comments
✓ Append reports

Group Test Results by

[✓ Test Name] Equalization
Lane Pass or Fail Results

Include in Appendix

✓ Preset test results
✓ Link to any saved waveform files in the report
✓ Save and link to full screenshots of plots

✓ View report after generating

For information on setting up reports, see Select report options (see page 43). For information on viewing reports, see View a Report (see page 45).

See also

About panels (see page 23)

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 9: Report options

Setting	Description
Report Generation	
Generate new report	Creates a new report.

Table 9: Report options (cont.)

Setting	Description
Append with previous run session	Appends the latest test results to the end of the current test results report.
Replace current test in previous run session	Replaces the previous test results with the latest test results. Results from newly added tests are appended to the end of the report.
Report name	Displays the name and location from which to open a report. The default location is at *\My TekExpress\PCI Express\Untitled Session*. 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. 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. ***NOTE.*** *You cannot set the file location using the Browse button.* Open an existing report. Click **Browse**, locate and select the report file and then click **View** 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.*
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.
Contents To Save	
Include pass/fail results summary	Sets the application to include the color block labeled Test Result (indicating whether the test passed or failed) in the report. For details, see Report Contents in View a report (see page 45).
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.

Table 9: Report options (cont.)

Setting	Description
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.
Append reports	Append with previous run session
View Report After Generating	Automatically opens the report in a Web browser when the test completes. This option is selected by default.
Group Test Results by	
Group Test Results by	Sets how to group the test results in the results pane and report.
Include in Appendix	
Include in Appendix	Enables adding the selected preset test results, waveform files, and plots to the report appendix.
Other	
View	Click to view the most current report.
Generate Report	Generates a new report based on the current analysis results.
Save As	Specify a name for the report.

See also

View a report (see page 45)
About setting up tests (see page 49)

View a report

The application automatically generates a report when test analysis is completed and displays the report in your default Web browser (unless you cleared the **View Report After Generating** check box in the Reports panel before running the test). If you cleared this check box, or to view a different test report, do the following:

1. Click the **Browse** button and locate and select the report file to view.

NOTE. *If you did not save the test setup after running the report and you either closed the application or you ran another report, the report file was not saved.*

2. In the Reports panel, click **View**.

For information on changing the file type, file name, and other report options, see Select report options (see page 43).

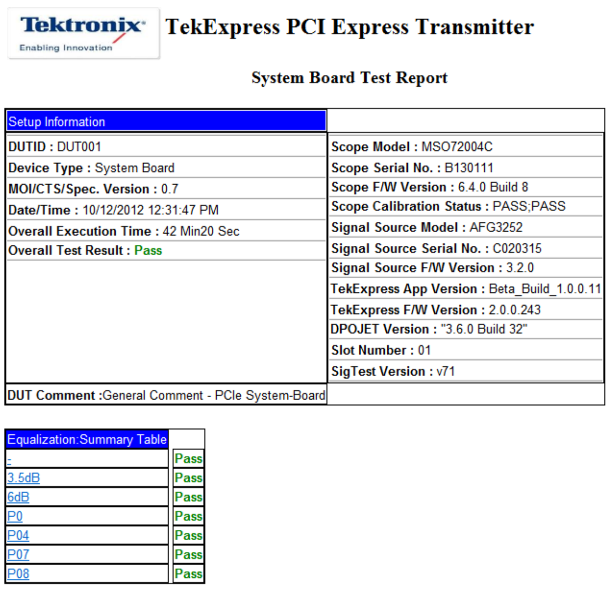
Report contents

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.*

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.



TekExpress PCI Express Transmitter

System Board Test Report

Setup Information	
DUTID : DUT001	**Scope Model :** MSO72004C
Device Type : System Board	**Scope Serial No. :** B130111
MOI/CTS/Spec. Version : 0.7	**Scope F/W Version :** 6.4.0 Build 8
Date/Time : 10/12/2012 12:31:47 PM	**Scope Calibration Status :** PASS;PASS
Overall Execution Time : 42 Min20 Sec	**Signal Source Model :** AFG3252
Overall Test Result : Pass	**Signal Source Serial No. :** C020315
	Signal Source F/W Version : 3.2.0
	TekExpress App Version : Beta_Build_1.0.0.11
	TekExpress F/W Version : 2.0.0.243
	DPOJET Version : "3.6.0 Build 32"
	Slot Number : 01
	SigTest Version : v71
DUT Comment : General Comment - PCIe System-Board	

Equalization:Summary Table	
-	Pass
3.5dB	Pass
6dB	Pass
P0	Pass
P04	Pass
P07	Pass
P08	Pass

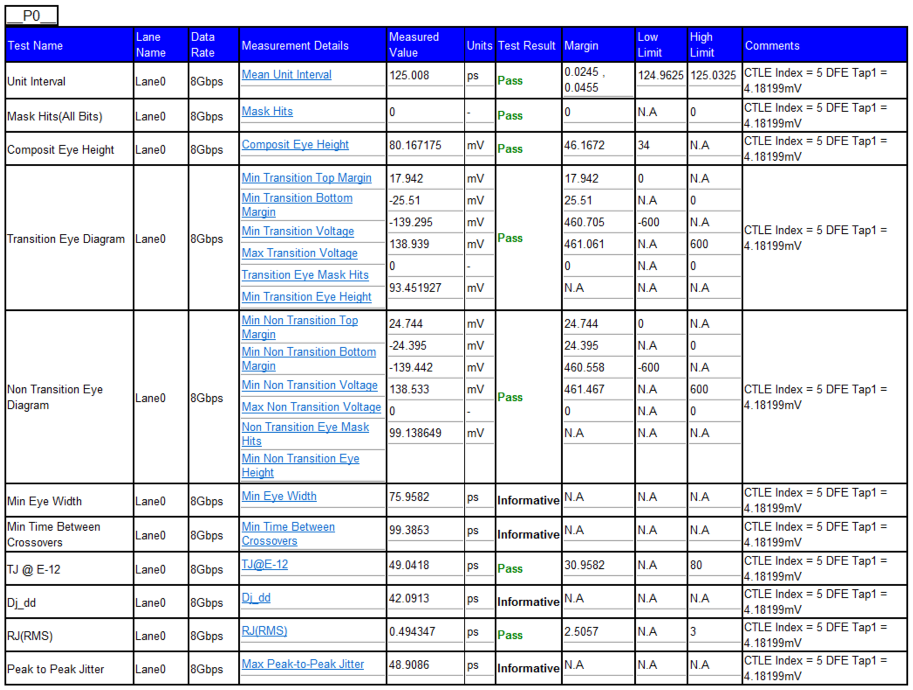
User comments

If you selected to include comments in the test report, any comments you added in the DUT tab of the Setup panel appear in the Comments section directly below the summary box.

Comments

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 Results Summary** check box in the Reports panel before running the test.



__P0__

Test Name	Lane Name	Data Rate	Measurement Details	Measured Value	Units	Test Result	Margin	Low Limit	High Limit	Comments
Unit Interval	Lane0	8Gbps	Mean Unit Interval	125.008	ps	Pass	0.0245 , 0.0455	124.9625	125.0325	CTLE Index = 5 DFE Tap1 = 4.18199mV
Mask Hits(All Bits)	Lane0	8Gbps	Mask Hits	0	-	Pass	0	N.A	0	CTLE Index = 5 DFE Tap1 = 4.18199mV
Composit Eye Height	Lane0	8Gbps	Composit Eye Height	80.167175	mV	Pass	46.1672	34	N.A	CTLE Index = 5 DFE Tap1 = 4.18199mV
Transition Eye Diagram	Lane0	8Gbps	Min Transition Top Margin	17.942	mV	Pass	17.942	0	N.A	CTLE Index = 5 DFE Tap1 = 4.18199mV
			Min Transition Bottom Margin	-25.51	mV		25.51	N.A	0	
			Min Transition Voltage	-139.295	mV		460.705	-600	N.A	
			Max Transition Voltage	138.939	mV		461.061	N.A	600	
			Transition Eye Mask Hits	0	-		0	N.A	0	
			Min Transition Eye Height	93.451927	mV		N.A	N.A	N.A	
Non Transition Eye Diagram	Lane0	8Gbps	Min Non Transition Top Margin	24.744	mV	Pass	24.744	0	N.A	CTLE Index = 5 DFE Tap1 = 4.18199mV
			Min Non Transition Bottom Margin	-24.395	mV		24.395	N.A	0	
			Min Non Transition Voltage	-139.442	mV		460.558	-600	N.A	
			Max Non Transition Voltage	138.533	mV		461.467	N.A	600	
			Non Transition Eye Mask Hits	0	-		0	N.A	0	
			Min Non Transition Eye Height	99.138649	mV		N.A	N.A	N.A	
Min Eye Width	Lane0	8Gbps	Min Eye Width	75.9582	ps	Informative	N.A	N.A	N.A	CTLE Index = 5 DFE Tap1 = 4.18199mV
Min Time Between Crossovers	Lane0	8Gbps	Min Time Between Crossovers	99.3853	ps	Informative	N.A	N.A	N.A	CTLE Index = 5 DFE Tap1 = 4.18199mV
TJ @ E-12	Lane0	8Gbps	TJ@E-12	49.0418	ps	Pass	30.9582	N.A	80	CTLE Index = 5 DFE Tap1 = 4.18199mV
Dj_dd	Lane0	8Gbps	Dj_dd	42.0913	ps	Informative	N.A	N.A	N.A	CTLE Index = 5 DFE Tap1 = 4.18199mV
RJ(RMS)	Lane0	8Gbps	RJ(RMS)	0.494347	ps	Pass	2.5057	N.A	3	CTLE Index = 5 DFE Tap1 = 4.18199mV
Peak to Peak Jitter	Lane0	8Gbps	Max Peak-to-Peak Jitter	48.9086	ps	Informative	N.A	N.A	N.A	CTLE Index = 5 DFE Tap1 = 4.18199mV

See also

Results panel overview (see page 40)
View test-related files (see page 42)

About setting up tests

Set up tests using the tabs in the Setup panel (see page 24). 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 (see page 51)
Before you click start (see page 6)
About test setups (see page 53)
About running tests (see page 51)

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 (see page 3)
View connected instruments (see page 18)
About setting up tests (see page 49)

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 (see page 24) and Reports panel (see page 43) to select and configure tests.

1. Set up equipment (see page 50).
2. Do the prerun checklist (see page 52).
3. Set DUT parameters (see page 24).
4. Select one or more tests (see page 30).
5. Select acquisitions (see page 30).
6. Configuration test parameters (see page 37).
7. Set test measurement notification options (see page 37).
8. Select report options (see page 43).

See also

About test setups (see page 53)
Before you click start (see page 6)
About running tests (see page 51)

About running tests

After selecting and configuring tests, review the prerun checklist (see page 52) 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 (see page 6)
About configuring tests (see page 34)
About setting up tests (see page 49)

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 (see page 6).

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 (see page 18).
    a. In PCIe, select the **Setup** panel and then click the **Test Selection** tab.
    b. Select any test and then click **Configure**.
    c. In 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 (see page 50)

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 (see page 49)
Save a test setup (see page 53)
Recall a saved test setup (see page 54)

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 (see page 43).
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 (see page 49)
Test setup overview (see page 51)
View test-related files (see page 42)
About configuring tests (see page 34)

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 (see page 53)
Create a new test setup based on an existing one (see page 54)
Test setups overview (see page 51)

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.

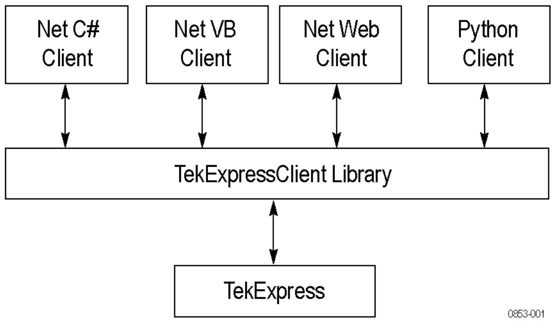
1. Select **Options** > **Open Test Setup**.
2. Select a setup from the list and then click **Open**.
3. Use the **Setup** and **Reports** panels to modify the parameters to meet your testing requirements.
4. Select **Options** > **Save Test Setup As**.
5. Enter a test setup name and click **Save**.

See also

About test setups (see page 53)
Set DUT parameters (see page 24)
Configuration parameters (see page 37)
Select acquisitions (see page 30)

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

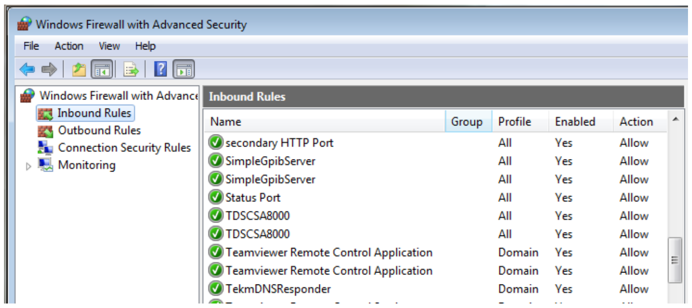
Requirements for developing TekExpress client (see page 57)

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

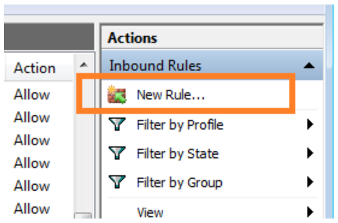

Windows Firewall with Advanced Security
File Action View Help
Windows Firewall with Advance
Inbound Rules
Outbound Rules
Connection Security Rules
Monitoring
Inbound Rules

Name	Group	Profile	Enabled	Action
secondary HTTP Port		All	Yes	Allow
SimpleGpibServer		All	Yes	Allow
SimpleGpibServer		All	Yes	Allow
Status Port		All	Yes	Allow
TDSCSA8000		All	Yes	Allow
TDSCSA8000		All	Yes	Allow
Teamviewer Remote Control Application		Domain	Yes	Allow
Teamviewer Remote Control Application		Domain	Yes	Allow
TekmDNSResponder		Domain	Yes	Allow

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.

Run the New Inbound Rule Wizard

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

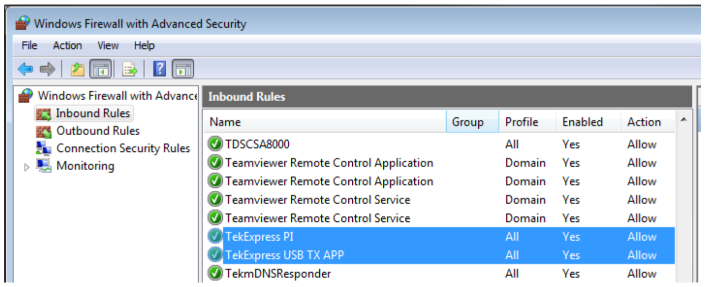


2. Verify that **Program** is selected in the Rule Type panel and click **Next**.
3. Click **Browse** in the Program panel and navigate to and select one of the following TekExpress applications (depending on the one for which you need to create a rule):
4. TekExpress PCI Express.exe
5. TekExpress.exe

NOTE. *See Application directories and content (see page 12) 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.


Windows Firewall with Advanced Security
File
Action
View
Help
Windows Firewall with Advance
Inbound Rules
Outbound Rules
Connection Security Rules
Monitoring
Inbound Rules

Name	Group	Profile	Enabled	Action
TDSCSA8000		All	Yes	Allow
Teamviewer Remote Control Application		Domain	Yes	Allow
Teamviewer Remote Control Application		Domain	Yes	Allow
Teamviewer Remote Control Service		Domain	Yes	Allow
Teamviewer Remote Control Service		Domain	Yes	Allow
TekExpress PI		All	Yes	Allow
TekExpress USB TX APP		All	Yes	Allow
TekmDNSResponder		All	Yes	Allow

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 `IRemoteInter-face.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.

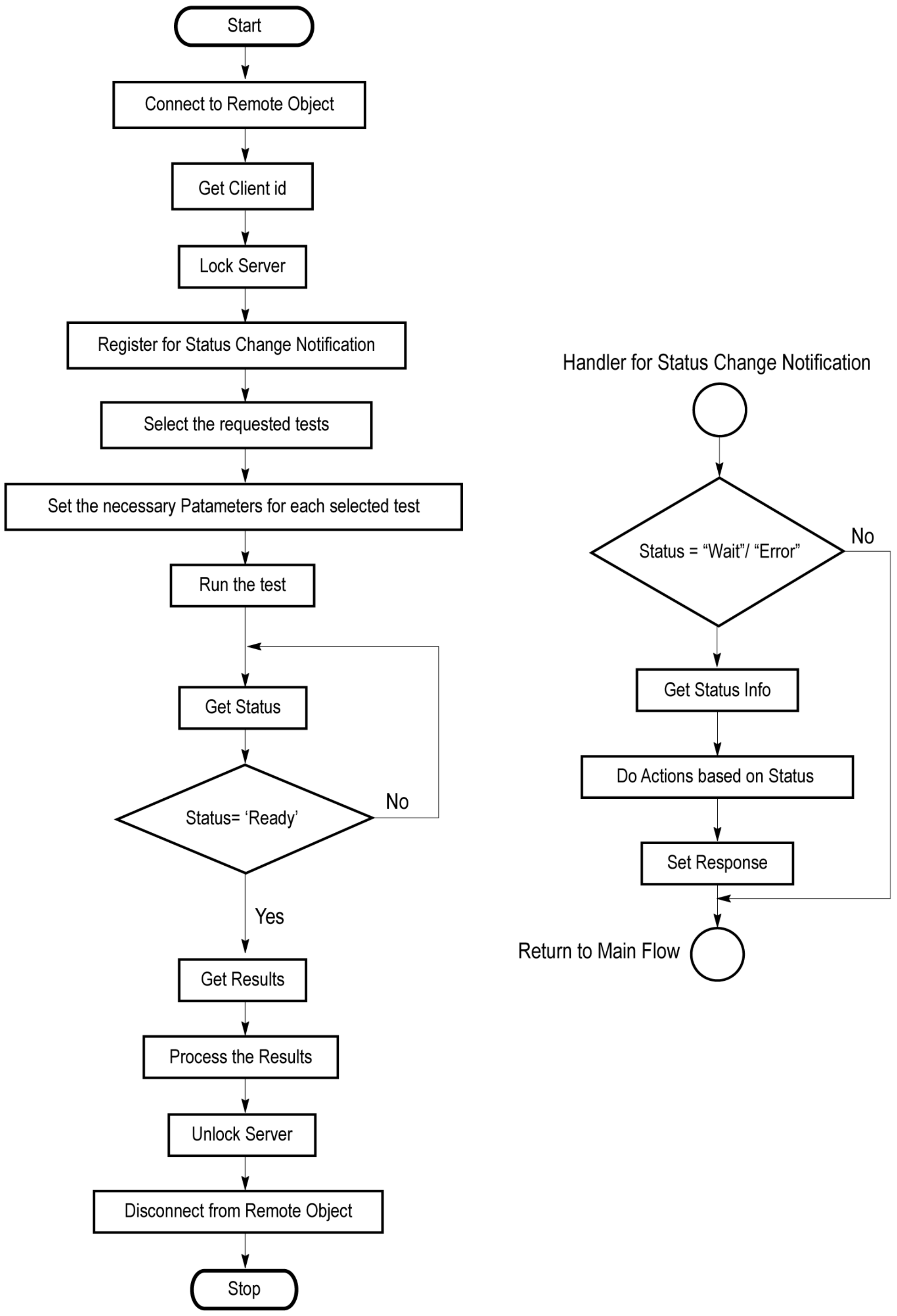
See also

PCIe application commands flow (see page 63)

Client programmatic interface example

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

Process flowchart


Start
Connect to Remote Object
Get Client id
Lock Server
Register for Status Change Notification
Select the requested tests
Set the necessary Patameters for each selected test
Run the test
Get Status
Status= 'Ready'
No
Yes
Get Results
Process the Results
Unlock Server
Disconnect from Remote Object
Stop
Handler for Status Change Notification
Status = "Wait"/ "Error"
No
Get Status Info
Do Actions based on Status
Set Response
Return to Main Flow

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 (see page 63)
Program remote access code example (see page 61)

Program remote access code example

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

Table 10: 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, WaitingMsb-BxMessage, 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 at
`TekExpress PCI Express\Examples\`.

PCIe application commands listing

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

Handle error codes (see page 106)
Get or set the timeout value (see page 107)
Wait for the test to complete (see page 108)
After the test is complete (see page 110)
Save, recall, or query a saved session (see page 116)
Unlock the server (see page 117)
Disconnect from the server (see page 117)

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 (see page 64)
	out string clientID (see page 65)
		This method connects the client to the server. Note (see page 65)
		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)

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

The server is LOCKED and the message displayed is "Server is locked by another client".

The session is UNLOCKED and the message displayed is "Lock Session to execute the command".

The server is NOTFOUND and the message displayed is "Server not found...Disconnect!".

When none of these fail conditions occur, then the message displayed is "Failed...".

Lock the server

Command name	Parameters	Description	Return value	Example

LockSession()	string clientID (see page 69)	This method locks the server.
		Note (see page 66)
		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)

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

The server is LOCKED and the message displayed is "Server is locked by another client".

The session is UNLOCKED and the message displayed is "Lock Session to execute the command".

The server is NOTFOUND and the message displayed is "Server not found...Disconnect!".

When none of these fail conditions occur, then the message displayed is "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 (see page 69)
	bool _verbose (see page 68)
		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.SetVerbose-Mode(clientID, true)
				Verbose mode is turned off
				returnval=m_Client.SetVer-boseMode(clientID, false)

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

The server is LOCKED and the message displayed is "Server is locked by another client".

The session is UNLOCKED and the message displayed is "Lock Session to execute the command".

The server is NOTFOUND and the message displayed is "Server not found...Disconnect!".

When none of these fail conditions occur, then the message displayed is "Failed...".

Set or get the DUT ID

Command name	Parameters	Description	Return value	Example

SetDutId()	string clientID (see page 69)
	string dutName (see page 64)
		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 (see page 65)

GetDutId()	string clientID (see page 69)
	string dutId (see page 65)
		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)

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

The server is LOCKED and the message displayed is "Server is locked by another client".

The session is UNLOCKED and the message displayed is "Lock Session to execute the command".

The server is NOTFOUND and the message displayed is "Server not found...Disconnect!".

When none of these fail conditions occur, then the message displayed is "Failed...".

Select the PCIe device

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

Command name	Parameters	Description	Return value	Example

SelectDevice	clientID (see page 69)
	device (see page 65)
		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 (see page 73)

Select PCIe device example

```
mClient.SelectDevice("1065-192.157.98.70", "PCIe", true);
```

Select the suite

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

Command name	Parameters	Description	Return value	Example

SelectSuite	clientID (see page 69)
	device (see page 65)
	devicesuite (see page 73)
		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 (see page 73)

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**.

Select suite example

```
mClient.SetGeneralParameter("1065-192.157.98.70", "PCIe", "Add-In-Card",
true);
```

Set the PCIe test version

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

Command name	Parameters	Description	Return value	Example

SelectVersion	clientID (see page 69)
	device (see page 65)
	testversion (see page 74)
		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 (see page 74)

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

Set test version example

```
mClient.SelectVersions("1065-192.157.98.70", "PCIe", "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

SetGeneralParame-ter	clientID (see page 69)
	device (see page 65)
	devicesuite (see page 73)
	parameter-String (see page 75)
		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 (see page 75)

string parameterString (for data 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", "PCIe", "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

SetGeneralParame-ter	clientID (see page 69)
	device (see page 65)
	devicesuite (see page 73)
	parameter-String (see page 75)
		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 (see page 75)

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"`.

Set test mode example

```
mClient.SetGeneralParameter("1065-192.157.98.70", "PCIe", "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 (see page 69)
	device ??? (see page 65)
	devicesuite ??? (see page 73)
		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 (see page 76)

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

SetGeneralParame-ter	clientID (see page 69)
	device (see page 65)
	devicesuite (see page 73)
	parameter-String (see page 77)
		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 (see page 77)

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", "PCIe", "System-Board",
"", "PreEmphasis3dB$Included");
```

Set the SSC parameter

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

Command name	Parameters	Description	Return value	Example

SetGeneralParame-ter	clientID (see page 69)
	device (see page 65)
	devicesuite (see page 73)
	parameter-String (see page 77)
		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 (see page 78)

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", "PCIe", "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 (see page 69)
	device (see page 65)
	devicesuite (see page 73)
	parameterString (see page 78)
		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 (see page 78)

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", "PCIe", "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 (see page 69)
	device (see page 65)
	devicesuite (see page 73)
	parameterString (see page 79)
		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 (see page 79)

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", "PCIe", "System-Board",
"", "SignalPreset$P0_P01_P04_");
```

To set all presets at one time:

```
mClient.SetGeneralParameter("1065-192.157.98.70", "PCIe", "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

SetGeneralParame- ter	clientID (see page 69)
	device (see page 65)
	devicesuite (see page 73)
	parameter-String (see page 80)
		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 (see page 80)

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", "PCIe", "System-Board",
"", "PresetLanes$Lane0_Lane1_Lane4");
```

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

```
mClient.SetGeneralParameter("1065-192.157.98.70", "PCIe", "System-Board",
"", "PresetLanes$Lane0_Lane1_Lane2_Lane3_Lane4_Lane5_Lane6_Lane7_Lane8_Lan-
e9_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

SetGeneralParame-ter	clientID (see page 69)
	device (see page 65)
	devicesuite (see page 73)
	parameter-String (see page 81)
		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 (see page 81)

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:Dif-ferential$CH1`".

Set lane source example

For differential probing:

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

For single ended probing:

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

```
mClient.SetGeneralParameter("1065-192.157.98.70", "PCIe", "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

SetGeneralParam-eter	clientID (see page 69)
	device (see page 65)
	devicesuite (see page 73)
	parameter-String (see page 83)
		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 (see page 83)

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 11:

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
P09	P04
P10	P04

Set preset test example

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

To set all presets:

```
mClient.SetGeneralParameter("1065-192.157.98.70", "PCIe", "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

SetGeneralParam-eter	clientID (see page 69)
	device (see page 65)
	devicesuite (see page 73)
	parameter-String (see page 84)
		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 (see page 84)

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", "PCIe", "System-Board",
"", "Acquisition$BeforeAnalysis");
```

Set the analysis mode parameter

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

Command name	Parameters	Description	Return value	Example

SetGeneralParam-eter	clientID (see page 69)
	device (see page 65)
	devicesuite (see page 73)
	parameter-String (see page 85)
		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 (see page 85)

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

SetGeneralParam-eter	clientID (see page 69)
	device (see page 65)
	devicesuite (see page 73)
	parameter-String (see page 86)
		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 (see page 86)

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", "PCIe", "System-Board",
"", "SigTestVersion$3_2_0");
```

For when AnalysisMode is set to CLI:

```
mClient.SetGeneralParameter("1065-192.157.98.70", "PCIe", "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

SetGeneralParam-eter	clientID (see page 69)
	device (see page 65)
	devicesuite (see page 73)
	parameter-String (see page 87)
		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 (see page 87)

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", "PCIe", "System-Board",
"", "On Failure Stop and Notify$False");
```

```
mClient.SetGeneralParameter("1065-192.157.98.70", "PCIe", "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

SetGeneralParam-eter	clientID (see page 69)
	device (see page 65)
	devicesuite (see page 73)
	parameter-String (see page 88)
		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 (see page 88)

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", "PCIe", "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

SetGeneralParam-eter	clientID (see page 69)
	device (see page 65)
	devicesuite (see page 73)
	parameter-String (see page 89)
		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 (see page 89)

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

SetGeneralParam-eter	clientID (see page 69)
	device (see page 65)
	devicesuite (see page 73)
	parameter-String (see page 90)
		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 (see page 90)

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", "PCIe", "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", "PCIe", "System-Board",
"", "CrossTalk$Off");
```

Set the waveform save parameter

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

Command name	Parameters	Description	Return value	Example

SetGeneralParam-eter	clientID (see page 69)
	device (see page 65)
	devicesuite (see page 73)
	parameter-String (see page 91)
		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 (see page 91)

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", "PCIe", "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

SetGeneralParam-eter	clientID (see page 69)
	device (see page 65)
	devicesuite (see page 73)
	parameter-String (see page 92)
		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 (see page 92)

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", "PCIe", "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

SetGeneralParam-eter	clientID (see page 69)
	device (see page 65)
	devicesuite (see page 73)
	parameter-String (see page 93)
		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 (see page 93)

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:`"Filter-FileEmbed8Gb$C:\PCI_Fil-ters\Gen1.flt"`. PCIe does not include any filter files.

Set embed/de-embed filter file example

```
mClient.SetGeneralParameter("1065-192.157.98.70", "PCIe", "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

SetGeneralParam-eter	clientID (see page 69)
	device (see page 65)
	devicesuite (see page 73)
	parameter-String (see page 94)
		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 (see page 94)

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", "PCIe", "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

SetGeneralParam-eter	clientID (see page 69)
	device (see page 65)
	devicesuite (see page 73)
	parameter-String (see page 94)
		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 (see page 94)

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$Opti-mize`".

Set equalization dropdown example

```
mClient.SetGeneralParameter("1065-192.157.98.70", "PCIe", "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

SetGeneralParam-eter	clientID (see page 69)
	device (see page 65)
	devicesuite (see page 73)
	parameter-String (see page 95)
		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 (see page 95)

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", "PCIe", "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 (see page 69)
	device (see page 65)
	devicesuite (see page 73)
	parameterString (see page 96)
		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 (see page 96)

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", "PCIe", "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 (see page 69)
	device (see page 65)
	devicesuite (see page 73)
	parameterString (see page 97)
		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 (see page 97)

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", "PCIe", "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

SetGeneralParam-eter	clientID (see page 69)
	device (see page 65)
	devicesuite (see page 73)
	parameter-String (see page 97)
		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 (see page 98)

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", "PCIe", "System-Board",
"", "Automation Settings$Use Custom Settings");
```

Set AFG/AWG signal type parameter

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

Command name	Parameters	Description	Return value	Example

SetGeneralParam-eter	clientID (see page 69)
	device (see page 65)
	devicesuite (see page 73)
	parameter-String (see page 98)
		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 (see page 98)

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", "PCIe", "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

SetGeneralParam-eter	clientID (see page 69)
	device (see page 65)
	devicesuite (see page 73)
	parameter-String (see page 99)
		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 (see page 99)

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

SetGeneralParam-eter	clientID (see page 69)
	device (see page 65)
	devicesuite (see page 73)
	parameter-String (see page 100)
		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 (see page 100)

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`".

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

SetGeneralParam-eter	clientID (see page 69)
	device (see page 65)
	devicesuite (see page 73)
	parameter-String (see page 100)
		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 (see page 100)

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`".

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

SetGeneralParam-eter	clientID (see page 69)
	device (see page 65)
	devicesuite (see page 73)
	parameter-String (see page 101)
		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 (see page 101)

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", "PCIe", "System-Board",
"", "RecordLength2Gb$10e6");
```

Set the sample rate parameter

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

Command name	Parameters	Description	Return value	Example

SetGeneralParam-eter	clientID (see page 69)
	device (see page 65)
	devicesuite (see page 73)
	parameter-String (see page 102)
		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 (see page 102)

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", "PCIe", "System-Board",
"", "SampleRate8Gb$25e9");
```

Set the bandwidth parameter

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

Command name	Parameters	Description	Return value	Example

SetGeneralParam-eter	clientID (see page 69)
	device (see page 65)
	devicesuite (see page 73)
	parameter-String (see page 103)
		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 (see page 103)

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", "PCIe", "System-Board",
"", "Bandwidth8Gb$6e9");
```

Set the signal validation parameter

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

Command name	Parameters	Description	Return value	Example

SetGeneralParam-eter	clientID (see page 69)
	device (see page 65)
	devicesuite (see page 73)
	parameter-String (see page 103)
		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 (see page 104)

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", "PCIe", "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

SetGeneralParam-eter	clientID (see page 69)
	device (see page 65)
	devicesuite (see page 73)
	parameter-String (see page 104)
		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 (see page 104)

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", "PCIe", "System-Board",
"", "SlotNumber$05");
```

Run with set configurations or stop the run operation

Command name	Parameters	Description	Return value	Example

Run()	string clientID (see page 69)	Runs the selected tests Note (see page 67)
		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 (see page 69)	Stops the running tests. Note (see page 68)	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)

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

The server is LOCKED and the message displayed is "Server is locked by another client".

The session is UNLOCKED and the message displayed is "Lock Session to execute the command".

The server is NOTFOUND and the message displayed is "Server not found...Disconnect!".

When none of these fail conditions occur, then the message displayed is "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
4	UNLOCK	The server is not locked; lock the server before performing the operation
0	NULL	Nothing

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

The server is LOCKED and the message displayed is "Server is locked by another client".

The session is UNLOCKED and the message displayed is "Lock Session to execute the command".

The server is NOTFOUND and the message displayed is "Server not found...Disconnect!".

When none of these fail conditions occur, then the message displayed is "Failed...".

Get or set the timeout value

Command name	Parameters	Description	Return value	Example

GetTimeOut()	string clientID (see page 69)	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 (see page 69)
	string time (see page 68)
		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)

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

The server is LOCKED and the message displayed is "Server is locked by another client".

The session is UNLOCKED and the message displayed is "Lock Session to execute the command".

The server is NOTFOUND and the message displayed is "Server not found...Disconnect!".

When none of these fail conditions occur, then the message displayed is "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 (see page 69)	This method gets the status of the server application.
		The states are Running, Paused, Wait, and Error (see page 64)
			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 (see page 69)
	out string[] status (see page 66)
		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 (see page 110)

Command name	Parameters	Description	Return value	Example

GetCurrentState-Info()
NOTE. *This command is used when the application is running and is in the wait or error state.*
	string clientID (see page 69)
	out string Wait-ingMsbBx-Caption (see page 68)
	out string Wait-ingMsbBxMes-sage (see page 69)
	out string[] WaitingMsb-BxButtontexts (see page 69)
		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.GetCurrentState-Info(clientID, WaitingMsbBx-Caption, WaitingMsbBxMes-sage, WaitingMsbBxButton-texts)

SendResponse()
NOTE. *This command is used when the application is running and is in the wait or error state.*
	string clientID (see page 69)
	out string Wait-ingMsbBx-Caption (see page 68)
	out string Wait-ingMsbBxMes-sage (see page 69)
	string Wait-ingMsbBxRe-sponse (see page 69)
		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(cli-entID, WaitingMsbBxCaption, WaitingMsbBxMessage, Wait-ingMsbBxResponse)

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

The server is LOCKED and the message displayed is "Server is locked by another client".

The session is UNLOCKED and the message displayed is "Lock Session to execute the command".

The server is NOTFOUND and the message displayed is "Server not found...Disconnect!".

When none of these fail conditions occur, then the message displayed is "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 (see page 69)
	string device (see page 65)
	string suite (see page 73)
	string test (see page 66)
		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 (see page 114)

GetResultsValue()	string clientID (see page 69)
	string device (see page 65)
	string suite (see page 73)
	string test (see page 66)
	string parameterString (see page 113)
		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 results value for a measurement example (see page 114)

Command name	Parameters	Description	Return value	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.			

GetReportParameter()	string clientID (see page 69)
	string device (see page 65)
	string suite (see page 73)
	string test (see page 66)
	string parameterString (see page 113)
		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 (see page 69)
	string filePath (see page 68)
		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 (see page 69)
	string filePath (see page 68)
		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:*

The server is LOCKED and the message displayed is "Server is locked by another client".

The session is UNLOCKED and the message displayed is "Lock Session to execute the command".

The server is NOTFOUND and the message displayed is "Server not found...Disconnect!".

When none of these fail conditions occur, then the message displayed is "Failed...".

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  `”

Get results value for a measurement example

This example returns the results value for the **Gen1** median peak jitter measurement:

returnval=m_Client.GetResultsValue(clientId, device, devicesuite, "Median Peak Jitter", "Value")

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

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

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  `”

NOTE. *GetResultsValue only returns one value of the respective group of tests.*

Save, recall, or query a saved session

Command name	Parameters	Description	Return value	Example

CheckSession-Saved()	string clientID (see page 69)
	out bool saved (see page 64)
		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.Check-SessionSaved(m_clientID, out savedStatus)

RecallSession()	string clientID (see page 69)
	string name (see page 67)
		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.RecallSes-sion(clientID, savedSession-Name)

SaveSession()	string clientID (see page 69)
	string name (see page 67)
		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 Sav-ed..."/"Failed..."
				m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.
				returnval as string
				returnval=m_Client.SaveSes-sion(clientID, desiredSession-Name)

SaveSessionAs()	string clientID (see page 69)
	string name (see page 67)
		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.SaveSes-sionAs(clientID, desiredSes-sionName)

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

The server is LOCKED and the message displayed is "Server is locked by another client".

The session is UNLOCKED and the message displayed is "Lock Session to execute the command".

The server is NOTFOUND and the message displayed is "Server not found...Disconnect!".

When none of these fail conditions occur, then the message displayed is "Failed...".

Unlock the server

Command name	Parameters	Description	Return value	Example

UnlockSession()	string clientID (see page 69)	This method unlocks the server from the client. The ID of the client to be unlocked must be provided. Note (see page 68)	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.Unlock-Server(clientID)

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

The server is LOCKED and the message displayed is "Server is locked by another client".

The session is UNLOCKED and the message displayed is "Lock Session to execute the command".

The server is NOTFOUND and the message displayed is "Server not found...Disconnect!".

When none of these fail conditions occur, then the message displayed is "Failed...".

Disconnect from the server

Command name	Parameters	Description	Return value	Example

Disconnect()	string clientID (see page 69)	This method disconnects the client from the server. Note (see page 65)	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.Discon-nect(m_clientID)

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

The server is LOCKED and the message displayed is "Server is locked by another client".

The session is UNLOCKED and the message displayed is "Lock Session to execute the command".

The server is NOTFOUND and the message displayed is "Server not found...Disconnect!".

When none of these fail conditions occur, then the message displayed is "Failed...".

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/A/B 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 series oscilloscopes is 100 GS/s, provided only 2 channels are on at a given time.

See also

Common test parameters and values (see page 37)
Configuration test parameters (see page 37)

Index