TekExpress® USB3.2 Tx
USB 3.2 Automated Test Solution Software

Printable Application Help

077-0939-06

TekExpress® USB3.2 Tx
USB 3.2 Automated Test Solution Software

Printable Application Help

www.tek.com
077-0939-06

Contacting Tektronix

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

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

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

Table of Contents

Running tests

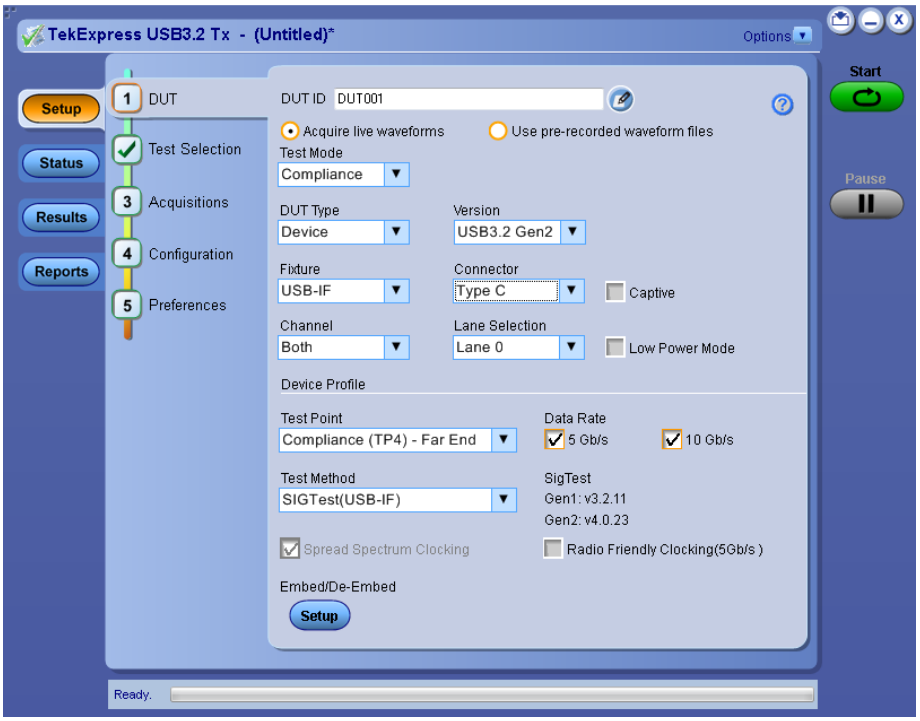
Saving and recalling test setup files

TekExpress programmatic interface

SCPI commands

Reference

Welcome



Welcome to the TekExpress® USB3.2 Tx Automated Test Solution Software application (referred to as USB3.2 Tx in the rest of the document). TekExpress USB3.2 Tx provides an automated, simple, and efficient way to test USB3.2 Tx transmitter interfaces and devices for USB-IF compliance and allows you to correlate Sig Test results with DPOJET for better margin, debugging, and analysis.

Key features and benefits

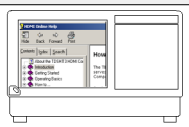
- Automated solution for USB Type C, Standard, and Micro connectors which support USB3.2 Tx specification and CTS (Gen1 & Gen 2).
- DPOJET plugin for USB Type C, Standard, and Micro connectors which support USB3.2 Tx specification and CTS (Gen1 & Gen 2) with setup files and MOI.
- Support embedding Channels (1 m, 2 m, and 3 m cable) and their respective filter files for Type C, Standard, and Micro connectors.
- Automated Lane Switching for reversible Type C connector.
- Command line support for SigTest (latest released version by USB-IF).
- Manual support for compliance to debug with DPOJET USB and USBSSP plug-ins.

Getting help and support

Related documentation

The following manuals are available as part of the TekExpress®
USB3.2 Tx Automated Test and Compliance Solution documentation set.

Table 1: Product documentation

Item	Purpose	Location
Help	Application operation and User Interface help	
PDF of the help	Printable version of the compiled help	 PDF file that ships with USB-TX and USBSSP-Tx software distribution (*USB-TX-Automated-Test-Solution-Software-Printable-Help-EN-US.pdf*).
DPOJET SuperSpeed (USB) and SuperSpeed Plus (USB SSP) Setup Library Methods of Implementation (MOI) for Verification, Debug and Characterization.	Detailed information on test setup and execution	PDF file that ships with USB-TX and USBSSP-Tx software distribution.

See also *Technical support*

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 a screen item (button, control, list item): 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.

Getting started

Minimum system requirements

The following table shows the minimum system requirements needed for an oscilloscope to run TekExpress USB3.2 Tx.

Table 2: USB3.2 system requirements

Component	Requirement
Oscilloscope	See *Supported instruments*
Processor	Same as the oscilloscope
Operating System	Same as the oscilloscope: ■ Windows 7/ Windows 10 (64-bit only) SP1 *Windows 7/ Windows 10 user account settings*
Memory	Same as the oscilloscope
Hard Disk	Same as the oscilloscope
Display	Super VGA resolution or higher video adapter (800 x 600 minimum video resolution for small fonts or 1024 x 768 minimum video resolution for large fonts). The application is best viewed at 96 dpi display settings [1]
Firmware	■ TekScope 10.8.1 and later (for Windows 7) ■ TekScope 10.10.0 and above (for Windows 10)
Software	■ Microsoft .NET 4.0 Framework ■ DPOJET Jitter and Eye Analysis Tool (version 10.0.10 or higher) with Advanced Jitter and Eye analysis (DJA option) installed. ■ Microsoft Internet Explorer 7.0 SP1 or later, or other Web browser for viewing reports. ■ Adobe Reader software 7.0 or later for viewing portable document format (PDF) files. ■ Serial Data Link Analysis (SDLA) software, version 3.0.4 or later, for Channel De-Embed, for custom filter development.

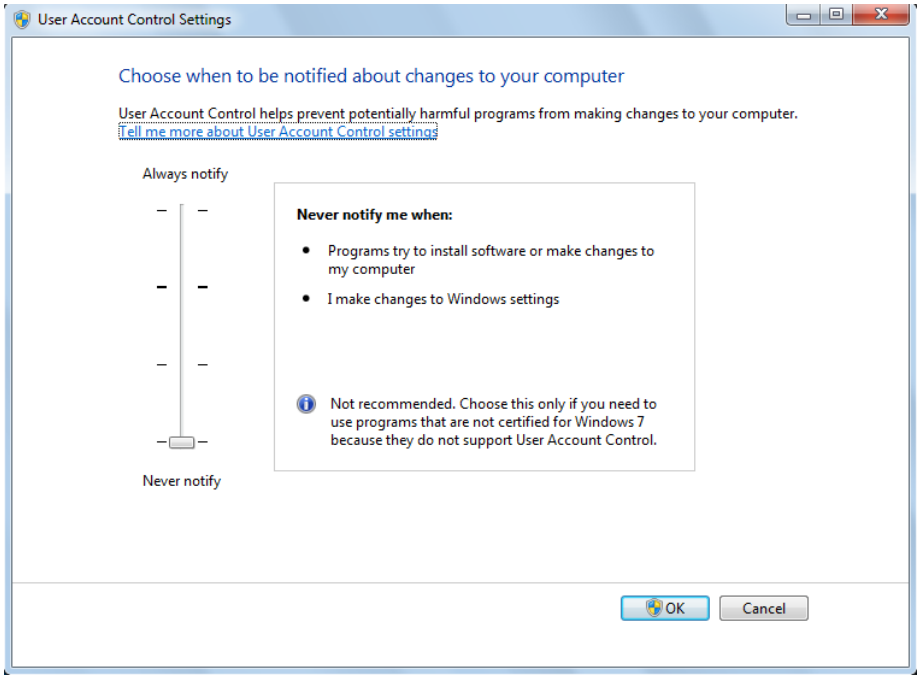
[1] If TekExpress is running on an instrument that has a video resolution less than 800x600, connect and configure a second monitor to the instrument.

Required windows 7/ windows 10 user account setting

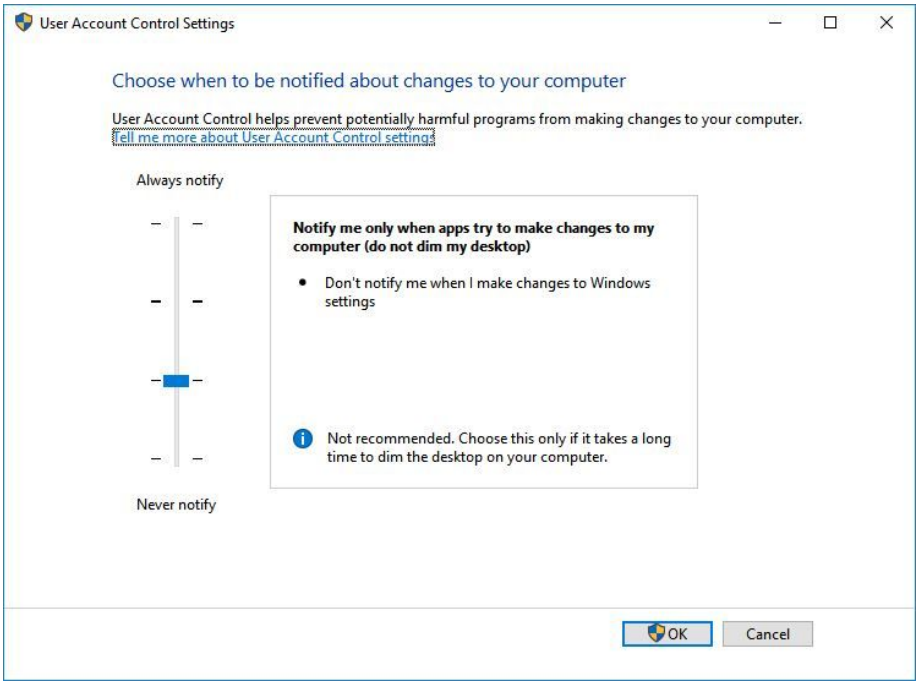
Windows 7/ Windows 10 instruments need to have the User Account Control Settings set to **Never Notify**. To set User Account Control Settings:

1. Go to **Control Panel > User Accounts > Change User Account Control settings**.
2. Set the sliding control to **Never Notify** as shown in the image, and click **OK**.

   Windows 7



Windows 10



See also *Supported oscilloscopes*

Supported instruments

Table 3: Required equipment

Resource	Model supported
Real-time oscilloscope	Tektronix DPO/DSA/MSO70000C, D, DX, and SX series oscilloscopes (Windows 7/ Windows 10 OS): ■ 16 GHz bandwidth and above required for both Gen1 (5 Gbps) and Gen2 (10 Gbps) Normative and Informative measurements. ■ 12.5 GHz bandwidth and above is suitable for Gen1 (5 Gbps) Normative and Informative measurements. ■ 8 GHz bandwidth model is suitable for Gen1 (5 Gbps) debug only.
Probes	Two TCA-SMA cables Two SMP cables P7313SMA differential probe P7500 TriMode probe
USB3.2 Standard and Micro-B fixtures	The USB31AET fixture includes all Standard and Micro-B test fixtures for USB3.2 Tx compliance testing. This fixture set includes Tx Host and device testing, captive device testing, and Rx cal testing. ■ For Gen1 (5 Gb/s), use the USB3.0 Electrical Test Fixture. ■ For Gen2 (10 Gb/s), use th USB31AET fixture set.
USB3.2 Type C fixtures	■ The USB31CET fixture includes Type C fixtures for USB3.2 Tx Type C compliance testing. ■ The fixture set includes Tx Host and Device testing, captive device testing, and Rx Cal testing.
Tektronix AWG/AFG instruments	AWG7102, AWG7122 series with options 6, 8 AWG70001A, AWG70002A AWG5202, AWG5204 and AWG5208, AWG5014B, AWG5014C, AWG5012C, AWG5002C AFG3051C, AFG3052C, AFG3101, AFG3101C, AFG3102, AFG3102C, AFG3251, AFG3251C, AFG3252, AFG3252C, AFG3151C, AFG3152C AFG31101 /AFG31102, AFG31151 /AFG31152 or AFG31251 / AFG31252
Tektronix Power Supply instruments	PWS4205, PWS4305, PWS4323, PWS4602, PWS4721 AWG7102, AWG7122B, AWG7122C

Resource	Model supported	
Connector Type	Standard	Standard A to B connector [2]
	Micro	Micro A and micro B connector
	Type C	Symmetrical connector on both side

See also *Minimum system requirements*

Install the software

Use the following steps to obtain the latest USB3.2 TX software from the Tektronix Web site and install on any compatible instrument running Microsoft Windows 7/ Windows 10 (64-bit). See *Minimum System Requirements* for details.

1. Close all applications (including the TekScope application).
2. Go to the www.tek.com Web site and locate the **Downloads** fields.
3. Enter **tekexpress USB3.2** in the *Model or Keyword* field, select **Software** from the *Select Download Type* list, and click **GO**.
4. Select the latest version of software, and then follow instructions to download the software file.
5. Copy or download the USB-TX installer executable file to the oscilloscope.
6. Double-click the installer .exe file to extract the installation files and launch the InstallShield Wizard.

   Follow the on-screen instructions. The software is installed at C:\Program Files (x86)\Tektronix\TekExpress\TekExpress USB3.2 Tx
7. *Verify application installation*

NOTE. *To test Gen2 capable hosts, use the HSET tool. The HSET tool configures the host into the compliance mode.*

[2] A to mini-B from USB-IF is not compliant any more.

See also *Minimum system requirements*

Supported instruments

Required My TekExpress folder settings

Verify application installation

To verify the installation was successful:

1. Open the TekScope application.
2. Click the **Analyze** menu.
3. Verify that TekExpress USB3.2 Tx is listed in the Analyze menu.
4. Click **TekExpress USB3.2 Tx** to open the application.

   Verify that the application opens successfully.

See also *Activate the license*

Required My TekExpress folder settings

Activate the license

Activate the license using the Option Installation wizard in the TekScope application:

1. In the TekScope application 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.
3. Follow the directions in the help topic to activate the license.

See also *View version and license information*

Required My TekExpress folder settings

View software version

To view version information for TekExpress USB3.2 Tx, click the **Options** button and select **About TekExpress**.

To view license and option key information in the TekScope applicaion:

1. In the TekScope application, select **Help > About TekScope**.
2. Scroll through the Options list to locate USB: TekExpress USB3.2 Tx and USBSSP-Tx: TekExpress USB3.2 Tx .
3. To view the Option installation key value, look below the Options list.

See also *Activate the license*

Options menu

Required my TekExpress folder settings

Before you run tests for the first time, you need to *Set the My TekExpress folder permissions*.

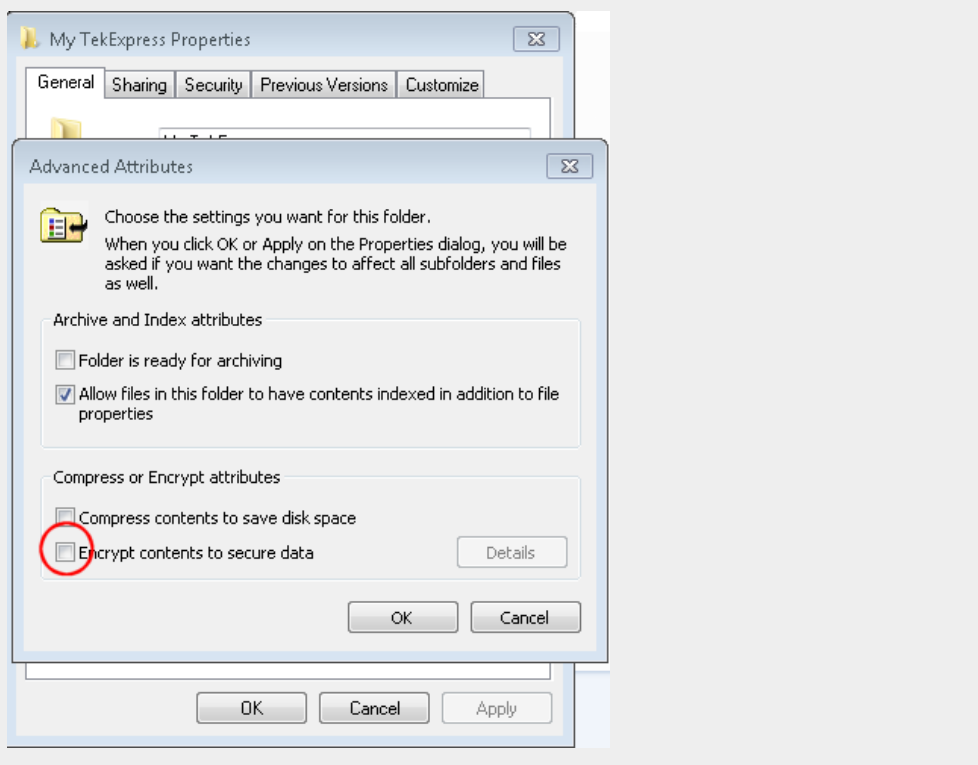
See also *Application directories and usage*

File name extensions

Set the my TekExpress folder permissions

Ensure that the My TekExpress folder has read and write access. Also verify that the folder is not set to be encrypted:

1. Right-click the folder and select **Properties**.
2. Select the **General** tab, and then click **Advanced**.
3. In the Advanced Attributes dialog box, ensure that the option Encrypt contents to secure data is NOT selected.



4. Click the **Security** tab and verify that the correct read and write permissions are set.

See also *Application directories and usage*

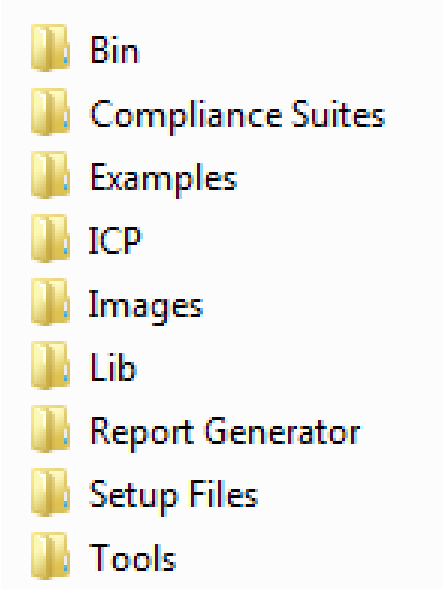
File name extensions

Application directories and their contents

TekExpress USB3.2 Tx application

The TekExpress USB3.2 Tx application files are installed at the following location:

C:\Program Files (x86)\Tektronix\TekExpress\TekExpress USB3.2 Tx



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

Table 4: Application directories and usage

Directory names	Usage
Bin	It contains USB3.2 TX application libraries.
Compliance Suites	It contains compliance-specific files.
Examples	It contains various support files.
ICP	It contains instrument and USB3.2 TX application-specific interface libraries.
Lib	It contains utility files specific to the USB3.2 TX application.
Report Generator	It contains style sheets for report generation.
Setup Files	It contains setup files.
Tools	It contains instrument and USB3.2 TX application-specific files.
.pdf, .chm	Help files

See also *View test-related files*

File name extensions

File name extensions

The TekExpress USB3.2 Tx application uses the following file name extensions:

File name extension	Description
.TekX	Application session files (the extensions may not be displayed)
.py	Python sequence file
.xml	Test-specific configuration information (encrypted) files Application log files
.wfm	Test waveform files
.mht	Test result reports (default) Test reports can also be *saved in HTML format*
.flt	Filter files
.xslt	Style sheet used to generate reports
.pdf, .chm	Help files

See also *View test-related files*

Application directories and their contents

Where test files are stored

When you launch TekExpress USB3.2 Tx for the first time, it creates the following folders on the oscilloscope:

- \My Documents\My TekExpress\USB3.2 Tx
- \My Documents\My TekExpress\USB3.2 Tx\Untitled Session

Every time you launch TekExpress USB3.2 Tx, an Untitled Session folder is created in the USB3.2 Tx folder. The Untitled Session folder is automatically deleted when you exit the USB3.2 Tx 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, the application creates a .TekX file, and a folder named for the session that contains associated files, on the oscilloscope X: drive.*

See also

Set the My TekExpress folder permissions

Application directories and usage

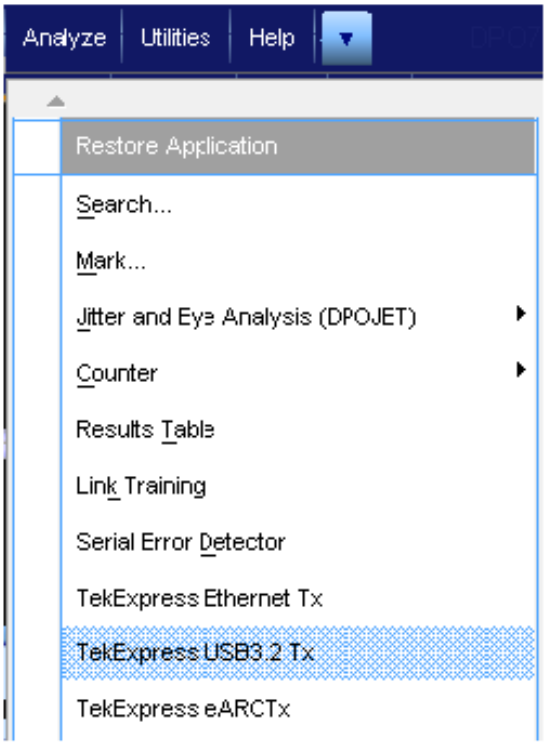
File name extensions

Operating basics

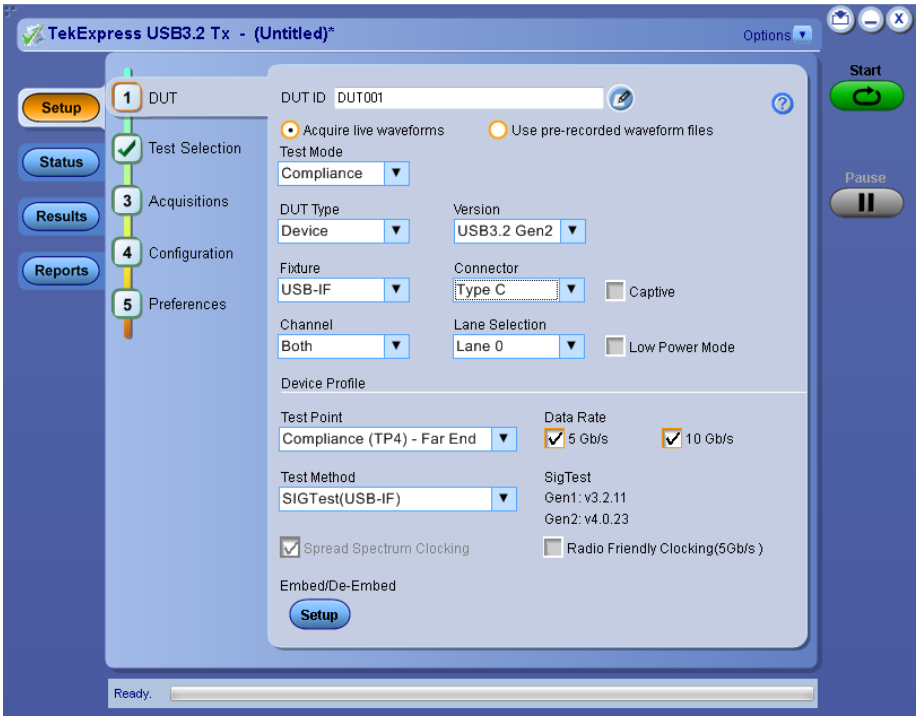
Run the application

To start the TekExpress USB3.2 Tx application, do either of the following:

- Select **Analyze > TekExpress USB3.2 Tx** from the TekScope menu.



- The oscilloscope opens the TekExpress USB3.2 Tx application.



When you first run the application after installation, the application checks for a file called Resources.xml located in the C:\Users\<username>\My TekExpress\USB3.2 Tx folder. The Resources.xml file gets mapped to the X: drive when the application launches. Session files are then stored inside the X:\USB3.2 Tx 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 to detect connected instruments before launching USB3.2 Tx.

NOTE. *Do the steps in the Required My TekExpress folder settings topic before running tests with the USB3.2 Tx application for the first time.*

To keep the USB3.2 Tx application window on top, select **Keep On Top** from the USB3.2 Tx *Options menu*. If the application goes behind the oscilloscope application, click **Analyze > TekExpress USB3.2 Tx** to move the application to be in front.

See also *Required My TekExpress folder settings*

Activate the license

Application controls

Application panel overview

Application panels overview

TekExpress USB3.2 Tx uses panels to group related configuration, test, 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.

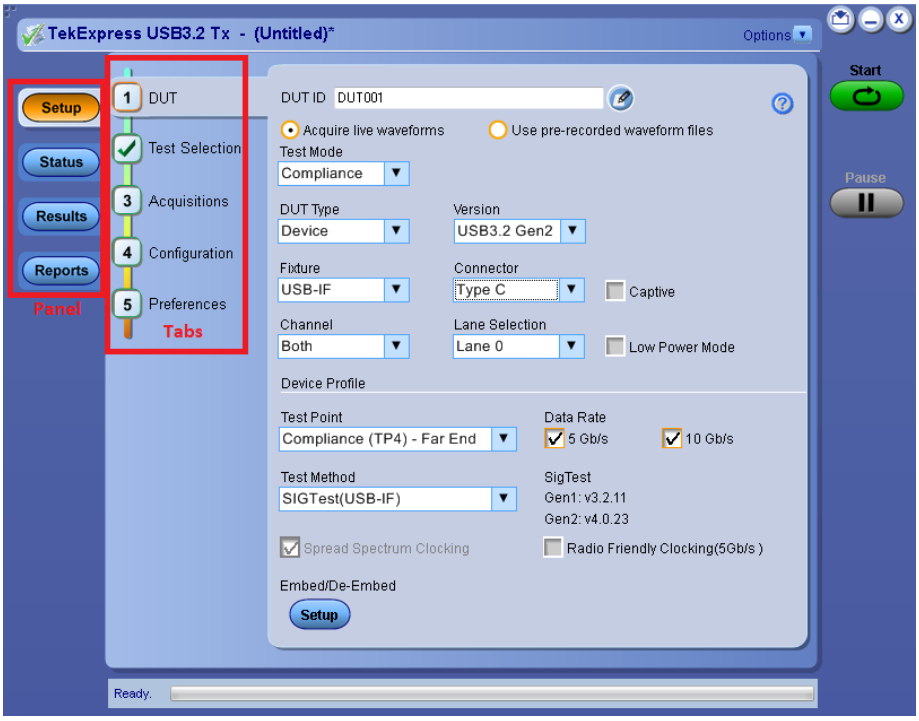


Table 5: Application panels overview

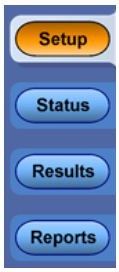
Panel Name	Purpose
Setup	The Setup panel shows the test setup controls. Click the **Setup** button to open this panel. Use this panel to: ■ *Select DUT parameters*. ■ *Select the test(s)*. ■ *Set acquisitions parameters* for selected tests. ■ *Select test notification preferences*.
Status	View the progress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

See also *Application controls*

Global application controls

Application controls

Table 6: Application controls descriptions

Item	Description
Options menu 	Menu to display global application controls.
Test Panel buttons 	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*.

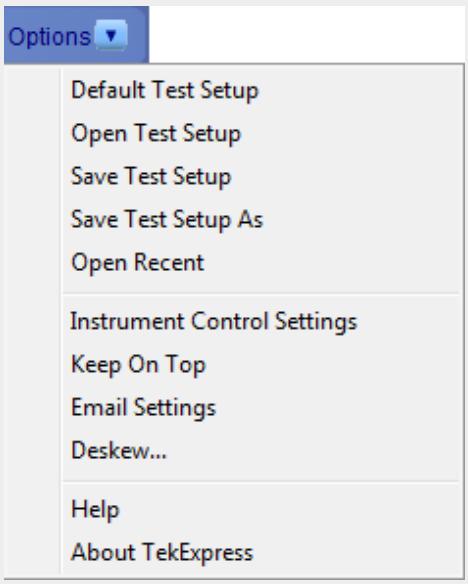
Item	Description
Minimize button 	Use the Minimize button to minimize the application.
Quit button 	Use the Quit button to exit the application.
Mini view/ Normal view 	Toggles the application between Mini view and Normal view. Mini view displays the run messages with the time stamp, progress bar, Start/Stop button, and Pause/Continue button. The application moves to Mini view when you click the Start button.

See also. *Application panel overview*

Options menu overview

The Options menu is located in the upper right corner of the application. The Options menu has the following selections:

Options menu



Menu	Function
Default Test Setup	Opens an untitled test setup with defaults selected
Open Test Setup	Opens a saved test setup

Menu	Function
Save Test Setup	Saves the current test setup selections
Save Test Setup As [1]	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	Detects, lists, and refreshes the connected instruments found on specified connections (LAN, GPIB, USB)
Keep On Top	Keeps the TekExpress USB3.2 Tx application on top of other open windows on the desktop
Email Settings	Use to configure email options for test run and results notifications
Deskew	Use to set deskew parameter and read instrument deskew/attenuation values
Help	Displays the TekExpress USB3.2 Tx help
About TekExpress	■ Displays application details such as software name, version number, and copyright ■ Provides access to *License information* for your USB3.2 Tx installation ■ Provides a link to the Tektronix Web site

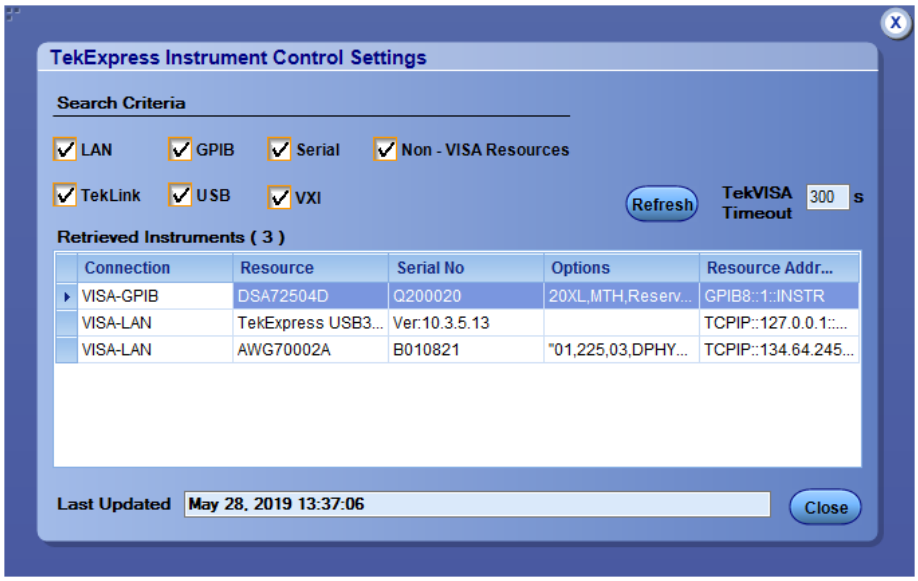
See also. *Application controls*

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

TekExpress instrument control settings

Instrument control settings. Use the TekExpress Instrument Control Settings dialog box to search for and list the connected resources (instruments) detected on selected connections (LAN, GPIB, USB), and each instruments connection information.

Access this dialog box from **Options > Instrument Control Settings**.



Use the Instrument Control Settings feature to *search for connected instruments* and view instrument connection details. You can select listed connected instruments for use in the Global Settings tab in the test configuration pane.

See also. *Options menu overview*

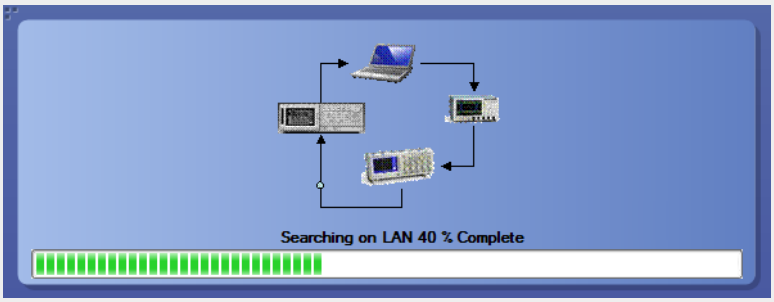
View connected instruments. Use the Instrument Control Settings dialog box to view or search for connected instruments required for the tests. The application uses TekVISA to discover the connected instruments on all selected connection types.

NOTE. *The correct instruments for the current test setup must be connected and recognized by USB-TX before running tests.*

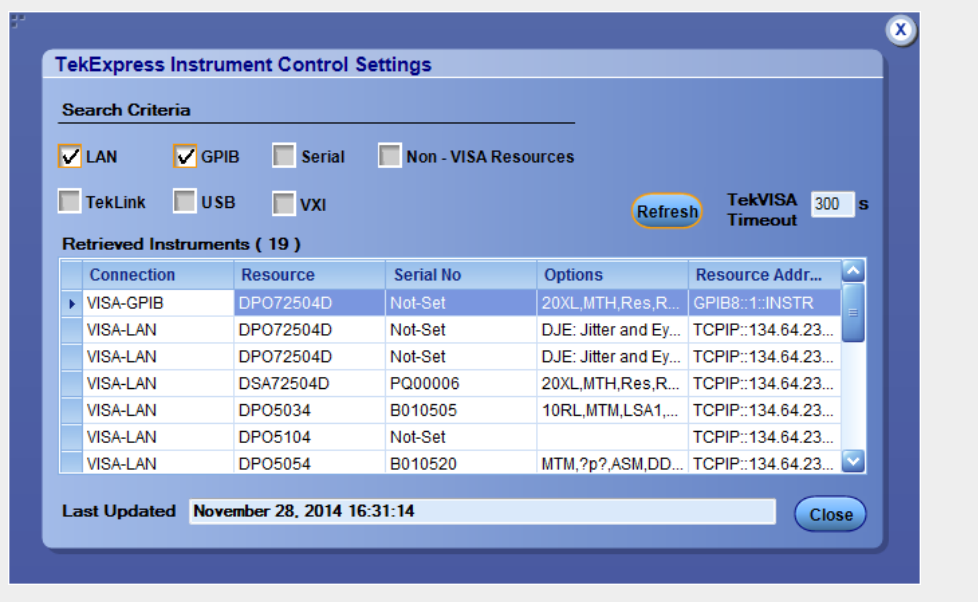
To refresh the list of connected instruments:

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.
3. Click **Refresh**. TekExpress searches for connected instruments.



4. After searching, 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, and then lists detected instruments on those connection types.



The Retrieved Instruments table lists instrument details. The time and date of the last time this table was updated is displayed in the Last Updated field.

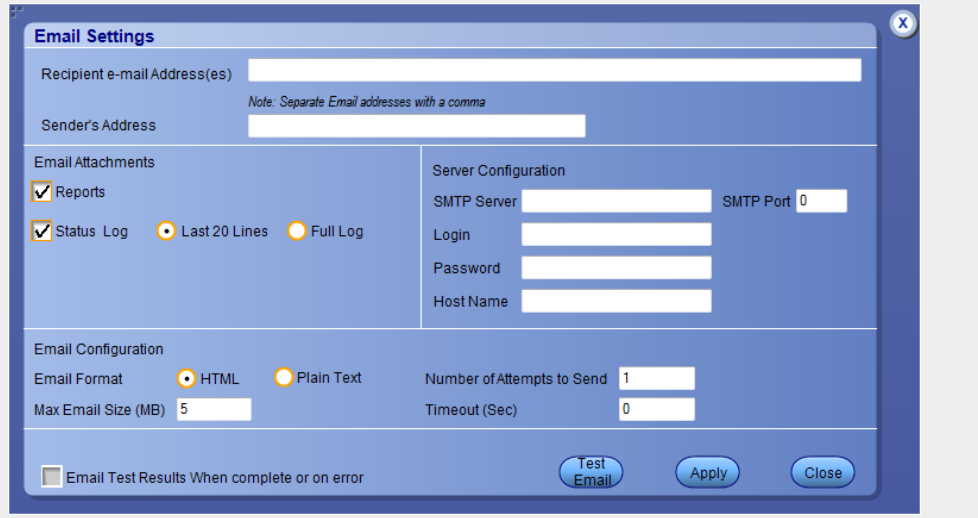
See also. *Equipment connection setup*

Configure email settings

Use the Email Settings dialog box to be notified by email when a test completes, fails, or produces an error:

1. Select **Options > Email Settings** to open the Email Settings dialog box.

Email settings



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.

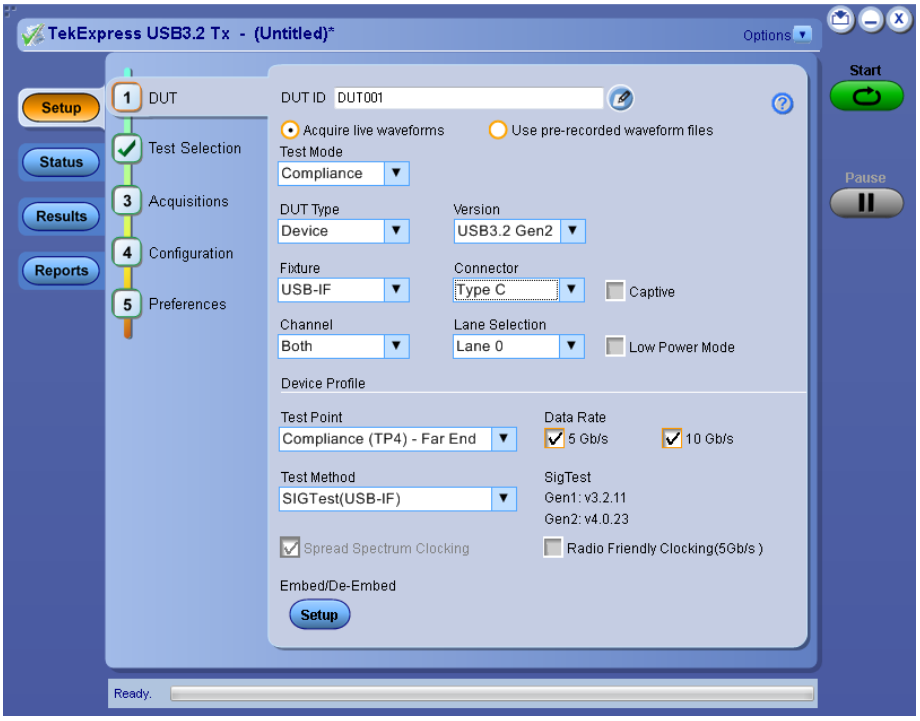
   NOTE. *If any of the above required fields are left blank, the settings will not be saved and email notifications will not be sent.*

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, select whether you want to receive the full log or just the last 20 lines.
6. In the Email Configuration section:
   - **Email Format**: Select the message file format to send: HTML (the default) or plain text.
   - **Max Email Size (MB)**: 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.
   - **Number of Attempts to Send**: Enter the number 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.

Setup panel

Setup panel overview

The Setup panel contains sequentially ordered tabs that help guide you through a typical test setup and execution process. Click a tab to open the associated panel and controls.



The tabs on this panel are:

DUT: *Set the DUT parameters*

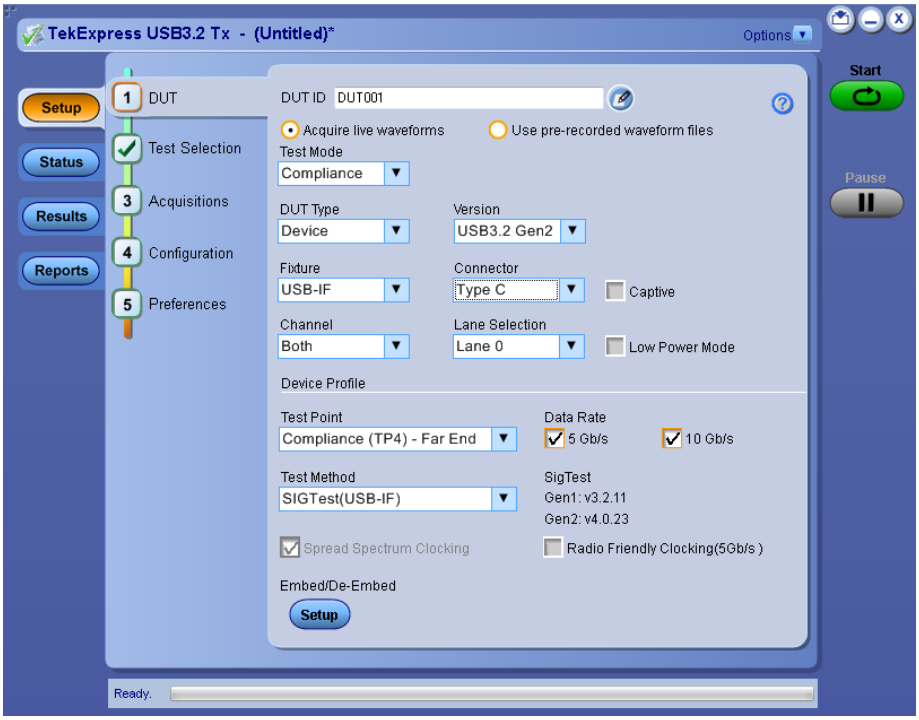
Test Selection: *Select test(s)*

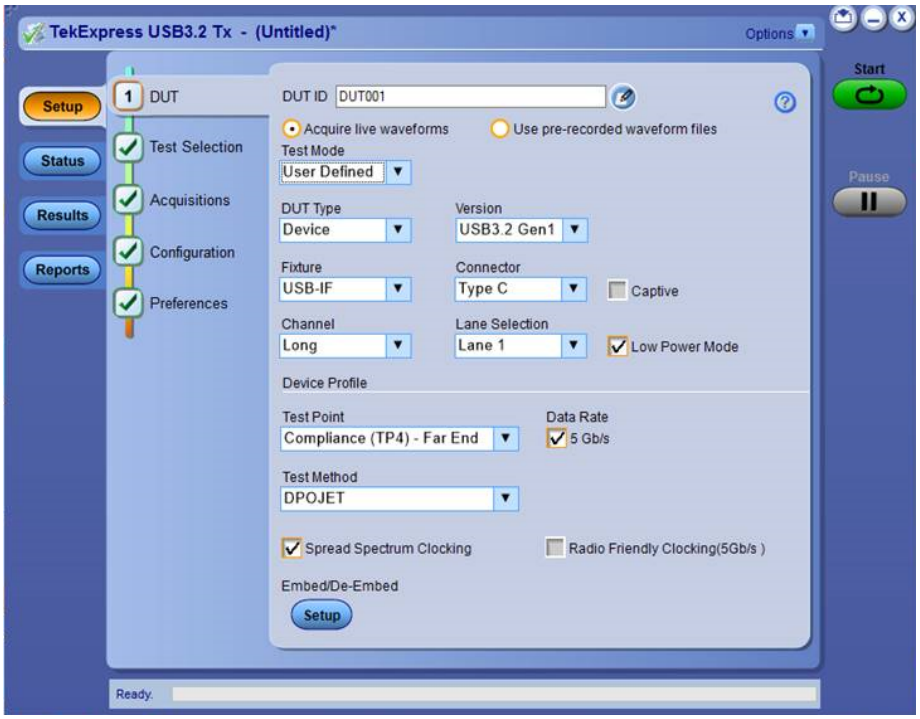
Acquisitions: *Select acquisition parameters*

Preferences: *Select test fail notification preferences*

Set DUT parameters

Use the DUT tab to select parameters for the device under test. The settings are global and apply to all tests for the current session. The DUT settings available and the options in the drop-down list depends on the selections made in the settings. DUT settings also affect the list of available tests in the Test Selection tab.





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

Table 7: DUT tab settings

Setting	Description
DUT ID	Adds an optional text label for the DUT to reports. The default value is DUT001. The maximum number of characters is 32. You cannot use the following characters in an ID name: (.,..,...,\,/:?"<>|*).
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. Maximum number of characters is 256. To enable or disable comments appearing on the test report, see *Select report options*).
Help	Opens the help document with DUT page selected.
Acquire live waveforms	Acquire active signals from the DUT for measurement and analysis.
Use prerecorded waveform files	Run tests on a saved waveform. *Open (load) a saved test setup*.
DUT Type	■ Device: Select the DUT type as Device. ■ Host: Select the DUT type as Host.

Setting	Description
Version	Lists the supported USB3.2 generations ■ USB3.2 Gen1 ■ USB3.2 Gen2
Fixture	■ USB-IF: For Standard and Type-C connector of USB3.2 Gen1 and USB3.2 Gen2
Connector	Select the appropriate connector form the options. ■ Standard: For Type A and Type B, select Standard. ■ Micro: For Micro B, select Micro. ■ Type C: Select for Type C
Captive	Select Captive check box if DUT type is Captive/ Tethered. Captive check box is applicable only if you select DUT Type as Device and Connector type as Type C or Standard.
Low power mode	Selected when USB 3.2 DUT is operating in low power mode.
Channel	It is a combination of different length of cables and PCBs. Depending on channel selection, different filters will be selected on the DUT tab. ■ Long: Consists of 3 m cable or 1 m cable with PCB assembly. ■ Short: Only PCBs, no cables attached. ■ Both: For Long and Short channel to be executed in single execution. During execution, user needs to change the channel from **Long** to **Short** as prompted by application.
Lane Selection	Select the Lane Selection as Lane 0, Lane 1 or Both. This configuration is enabled only when the Connector type is Type C.
Test Mode	■ Compliance: Preselects tests and parameters needed to meet compliance specifications for the selected device type. Disables the compliance filter controls. ■ User Defined: Enables the user to select specific tests and set custom parameters for tests (using the Configuration button).
Device Profile	

Setting	Description
Test Point	Select the appropriate test point location from those listed. Only **Compliance (TP4) - Far End** test point is available when Test Mode is set to **Compliance**. For Tx Pins - Near End and Custom test point, select Test Mode as User Defined.
Data Rates	Sets the test data rate (5 Gbps, 10 Gbps, or Both).
Test Method	Sets the algorithms used to measure and analyze the signal. SigTest (USB-IF) is default test method for Compliance. ■ DPOJET: Select to perform measurements implemented in DPOJET. ■ SIGTest(USB-IF): Select to perform measurements implemented in SIGTest (USB-IF). ■ Both: Select to perform measurement implemented in DPOJET and SIGTest (USB-IF) simultaneously.
SigTest	Displays the SigTest version for Gen1 and Gen2.
Spread Spectrum Clocking	Select this check box if your DUT supports Spread Spectrum Clocking (SSC). Selects SSC tests based on your DUT configuration.

Setting	Description
Radio Friendly Clocking(5Gb/s)	Select this check box to select the Radio Friendly Clocking. This option is available only when Datarate 5 Gb/s is selected.
Filter Selection	Click to view and select the filter files. Lists de-embed, embed, and CTLE filter files or settings used for the current DUT test points. Use filters to take cable and fixture signal path length and characteristics into account. This configuration is enabled only when the Test Mode is **User Defined** and the Channel type is **Long** or **Both**.  **Figure 1: Filter Selection** For USB3.2 Gen2, CTLE is performed using SDLA with default value as "Optimize". You can also select a specific CTLE index when in the User Defined test mode.

See also. *Select a test*

Select tests

Use the **Test Selection** tab to select **USB3.2** tests. Listed tests depend on settings in the DUT tab.

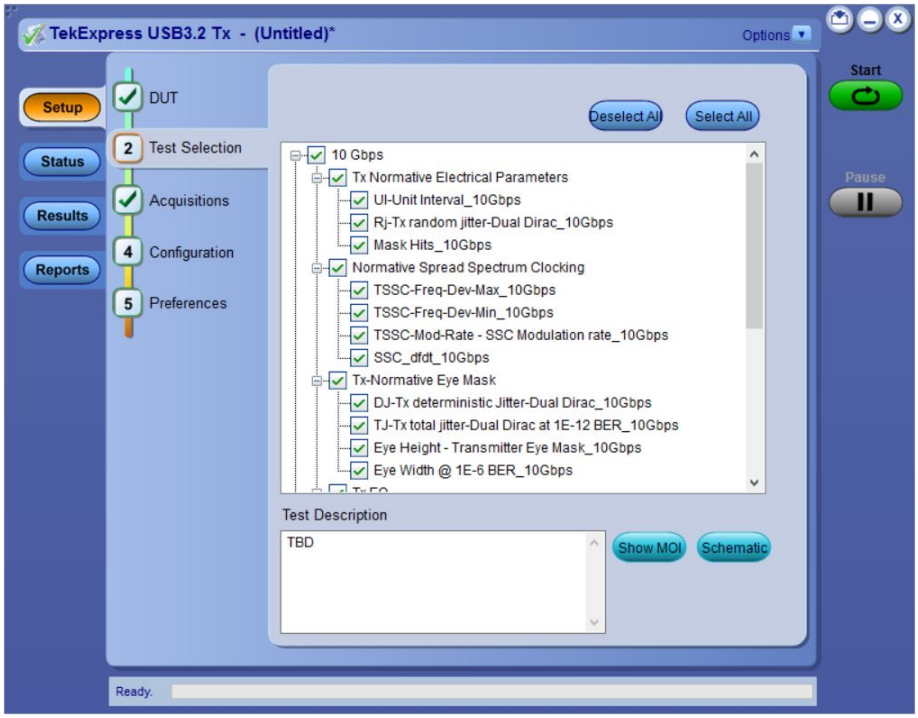


Table 8: Test Selection tab settings

Setting	Description
Deselect All, **Select All**	Deselect or select all tests in the list.
Tests	Click a test to select or deselect. Selecting a test also show details about that test in the Test Description pane. All required tests are selected when in Compliance test mode.
Show MOI	Opens the Method of Implementation (MOI) PDF file. You need to select a test before you can open the MOI.
Schematic	Displays equipment connection setup for the selected measurements. You need to select at least a measurement before you click the Schematic.

NOTE. *All tests are selected by default.*

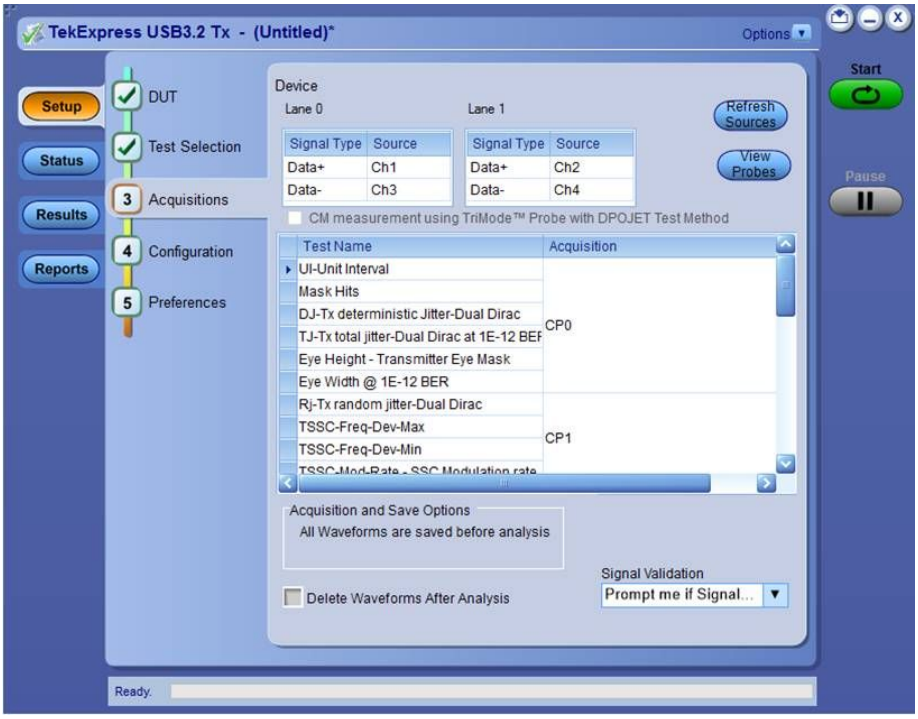
NOTE. *The application does not show the oscilloscope cursor1 and 2 for each burst. The application runs an analysis on the first five bursts of an acquisition and displays the result statistics.*

See also. *Set acquisition parameters*

Set acquisition parameters

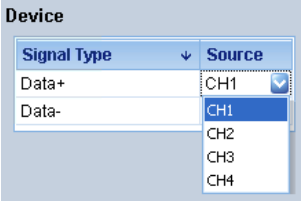
Use the **Acquisition** tab in the Setup panel to view test acquisition parameters. You also use this tab to load prerecorded (saved) test session waveform files on which to run tests.

Contents displayed on this tab depend on the DUT type and selected tests.



NOTE. *USB3.2 Tx acquires all waveforms required by each test group and generation being tested (Gen1, Gen2) before performing analysis.*

Table 9: Acquisitions tab settings

Setting	Description
Device	Lists the signal type and input channel assigned to that type. Click on Source fields to assign a channel source to a signal type.  The (Source) channels are auto selected, based on the probe type used and Lane selected on the DUT tab.
Refresh sources	Updates the list of available channel sources as used by the Source fields in the Device list. Click this button if you change channel connections in the test setup.
View probes	Use the View Probes dialog box to show the detected probe configurations, and to enable or disable probe signal source access in the application. Only available for the live waveforms.
CM measurement using TriMode™ Probe with DPOJET test method	Set this when using a supported Tektronix TriMode probe for signal acquisition of the LFPS Vcm-AC measurement. The TriMode Probe can switch between differential, single ended and common mode (CM) measurements without changing the probe. In CM mode, it generates CM signal by taking two Single Ended inputs (D + and D-). The application applies post-processing on the entire LFPS CM acquisition, after applying compliance filters. This control is only valid for the DPOJET test method.
Signal Validation	Sets the signal validation actions. Select from the available list items.
Delete waveform after analysis	Select to delete the waveforms after analyzing from the current acquisition. This is applicable only for live waveforms.

USB3.2 Tx saves all acquisition waveforms to files by default. 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:\TekExpress USB3.2 Tx\Untitled Session \<dutid>\<date>_<time>. Images created for each analysis, reports, and other information specific to that session are also saved in this folder.

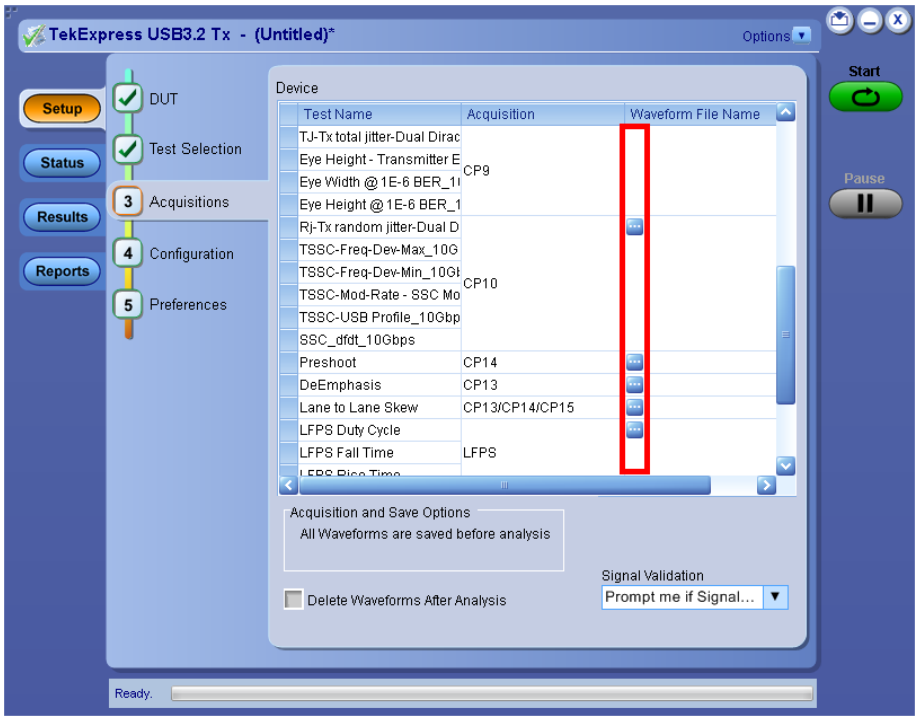
When the session is saved, content is moved to that session folder and the "Untitled Session" name is replaced by the session name.

See also. *Running tests on prerecorded saved waveforms*

Running tests on prerecorded (saved) waveforms

To load a saved waveform file:

1. Click **DUT**.
2. Click **Use pre-recorded waveform files**.
3. Click **Acquisitions**. The Waveform Filename column now shows browse buttons.



4. Click the browse button (…) for each test acquisition type (CP0, CP1, LFPS).
5. Navigate to and select the appropriate waveform file(s). You must select all waveforms required for the acquisition type.

6. To change, remove, or add a file to the list, click the browse button next to the file name to change, and use the menu items to replace, remove (delete) or add a file in the list.
7. Click **Start**.

Set configuration tab parameters

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

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

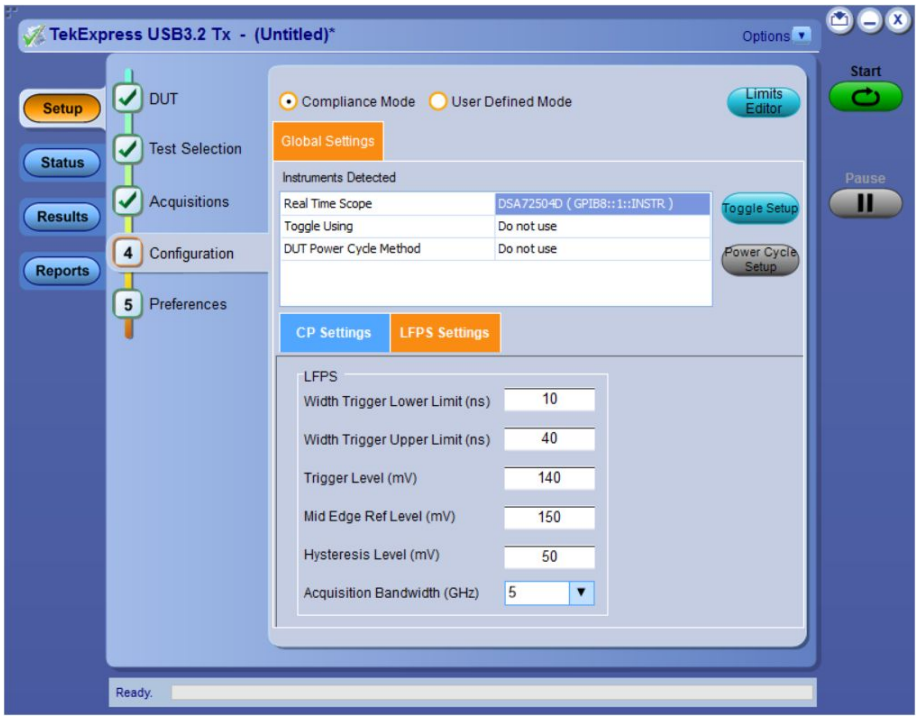
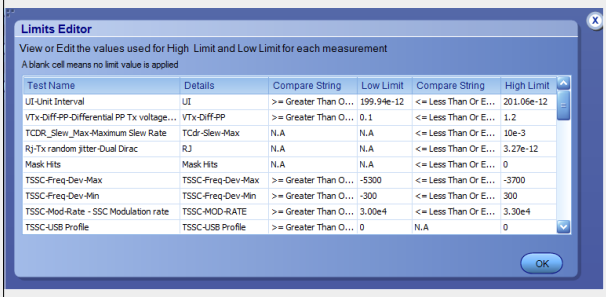
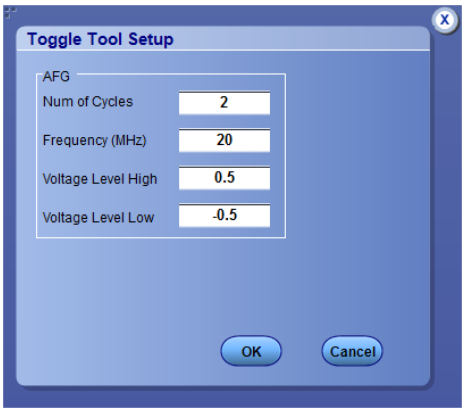
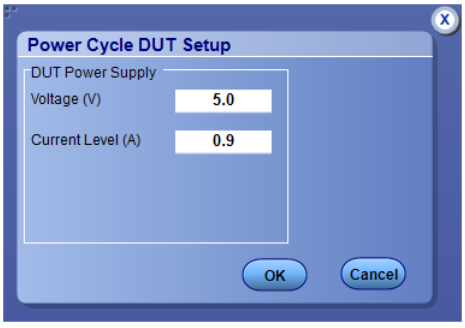


Table 10: Configuration tab settings

Setting	Description
Compliance Mode	Select Compliance Mode. By default, Compliance Mode is selected.
User Defined Mode	Select User Defined Mode.
Limits Editor	Opens the Limits Editor dialog box. In User Defined Mode, use the Limits Editor to edit individual test limit settings.  To edit a value, click that field and either select from the displayed list or enter a new value. Use scroll bars to view all available fields. In Compliance Mode, use the Limits Editor to view the measurement high and low limits used for selected tests. You cannot edit values while in Compliance mode.
Global Settings	

Setting	Description
Instruments Detected	Displays a list of the connected instruments found during the instrument discovery. Instrument types include equipment such as oscilloscopes and signal sources (AFG, AWG) and power supply. Select **Options > Instrument Control Settings** to *refresh the connected instrument list*.
Toggle tool setup	Click to configure the device selected to Toggle. The Toggle Tool Setup window displays the configurations for the selected **Toggle Using** in the Instruments Detected dialog box.  **Figure 2: Toggle Tool Setup for AFG**
Power Cycle DUT Setup	Click to configure the power cycle setup. The Power Cycle DUT Setup displays the configurations for the selected DUT Power Cycle Method in the Instruments Detected dialog box.  **Figure 3: Power Cycle DUT Setup for power supply**
CP Settings Select User Define Mode to enable the controls of CP Settings tab.	
Record Length (M)	Set the record length for 5 Gb/s and 10 Gb/s. The default values are as follows: ■ Gen1 CP0: 10 M ■ Gen1 CP1: 20 M ■ Gen2: 20 M Range: 5 M - 30 M

Setting	Description
Sample Rate (GS/s)	Set the sample rate for 5Gb/s and 10 Gb/s. The default values are as follows: ■ Gen1: 50 GS/s ■ Gen2: 100 GS/s The minimum and maximum values are as follows: ■ Gen1: Min: 25 GS/s, Max: 100 GS/s ■ Gen2: Min: 50 GS/s, Max: 100 GS/s
Bandwidth (GHz)	Set the bandwidth for 5 Gb/s and 10 Gb/s. The default values are as follows: ■ Gen1: 12.5 GHz ■ Gen2: 16 GHz The minimum and maximum bandwidth values depend on the oscilloscope bandwidth limits.
Do Not Autoset	Select not to autoset. Selecting this check box, enables the Vertical Scale (mV/div) control.
Vertical Scale (mV/dv)	Set the vertical scale for 5 Gb/s and 10 Gb/s. The default values are as follows: ■ Gen1: 60 mV/div ■ Gen2: 60 mV/div The minimum and maximum vertical scale values depend on the oscilloscope resolution.
Trigger Level (mV)	Set the trigger level (mV) for 5 Gb/s and 10 Gb/s. The default values are as follows: ■ Gen1: 10 mV ■ Gen2: 10 mV Range: -100 mV - 300 mV
LFPS Settings [1]	
Width Trigger Lower Limit (ns)	Set the width trigger lower limit in nanoseconds. The default value is 10 ns. Range: 1 ns - 15 ns
Width Trigger Upper Limit (ns)	Set the width trigger upper limit in nanoseconds. The default value is 40 ns. Range: 15 ns - 100 ns
Trigger Level (mV)	Set the trigger level in millivolt. The default value is 140 mV. Range: 20 mV - 250 mV

[1] If the oscilloscope does not properly trigger on the DUT LFPS signal, adjust these trigger settings to enable the oscilloscope to detect and trigger on the LFPS signal.

Setting	Description
Mid Edge Ref Level (mV)	Set the mid edge ref level in millivolt. The default value is 150 mV. Range: 100 mV - 1000 mV
Hysteresis Level (mV)	Set the hysteresis level in millivolt. The default value is 50 mV. Range: 10 mV - 300 mV
Acquisition Bandwidth (GHz)	Select the acquisition bandwidth from the drop-down list. The default value is 5 GHz.

NOTE. *If Toggle Using is AWG, select the check box in the Toggle Setup to verify the toggle status.*

NOTE. *If Toggle Using is Do not use, select the check box in the Toggle Setup to automatically recover the oscilloscope settings.*

Preferences tab

Use the Preferences tab to set the application action when a test measurement fails.

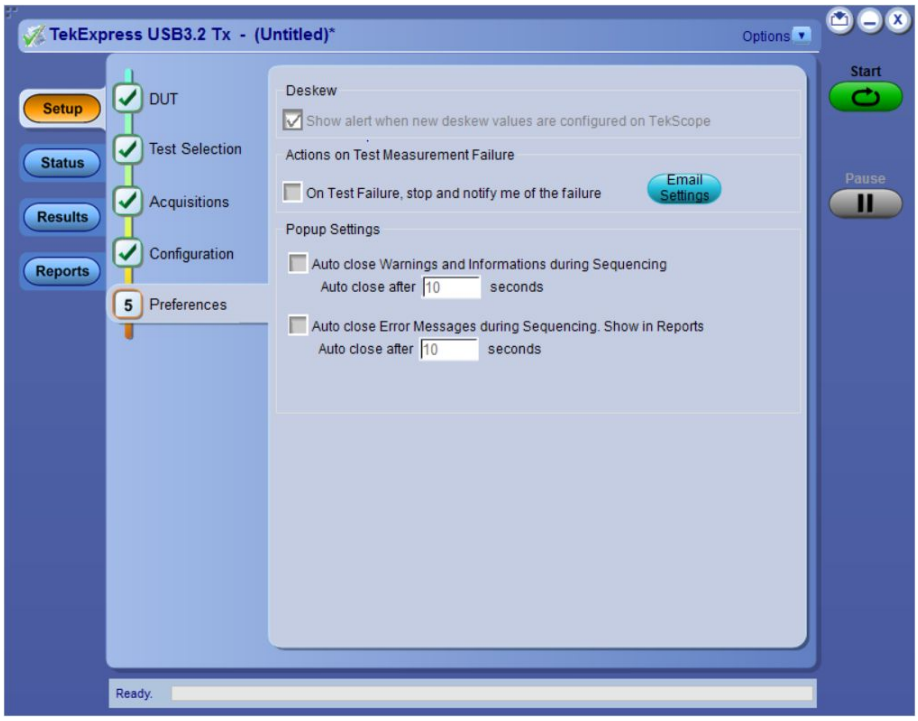


Table 11: Preferences tab settings

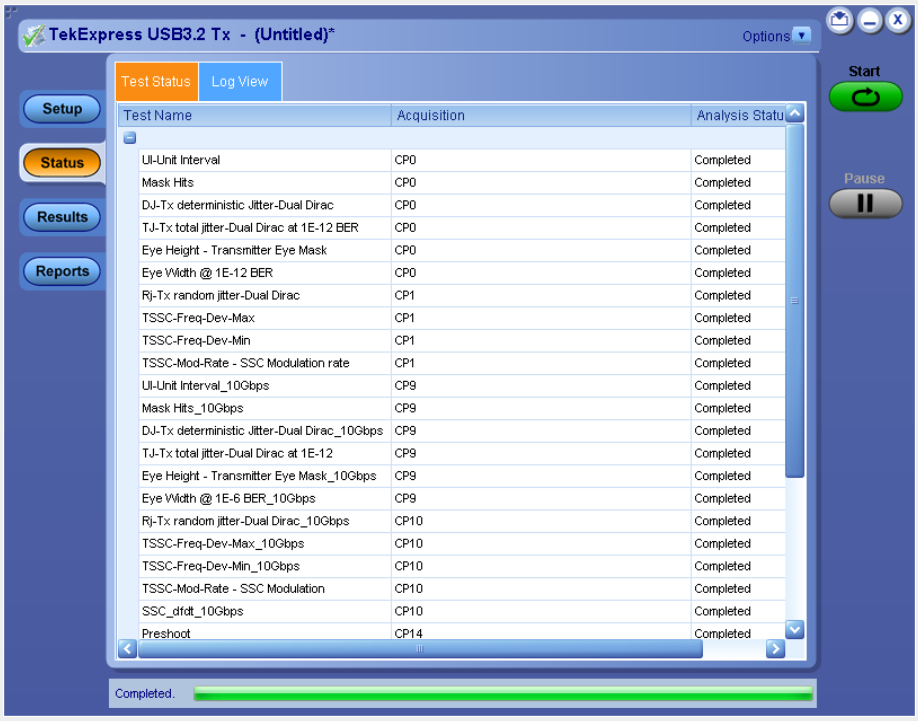
Setting	Description
Actions on Test Measurement Failure	
Show alert when new deskew values are configured on TekScope	Select to show alert when new deskew values are configured on TekScope. By default, it is selected.
On Test Failure, stop and notify me of the failure	Stops the test and sends an email when a test fails. Click **Email Settings** button and verify that "Email Test Results when complete or on error" is selected, and to verify the address to which the email is sent.
Popup Settings	
Auto close Warnings and Informations during Sequencing Auto close after <no> Seconds	Select to auto close warnings/informations during sequencing. Set the Auto close time. By default, it is unselected.
Auto close Error Messages during Sequencing. Show in Reports. Auto close after <no> Seconds	Select to auto close Error Messages during Sequencing. Set the Auto close time. By default, it is unselected.

Status panel

Status panel overview

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

Test status view



Log view

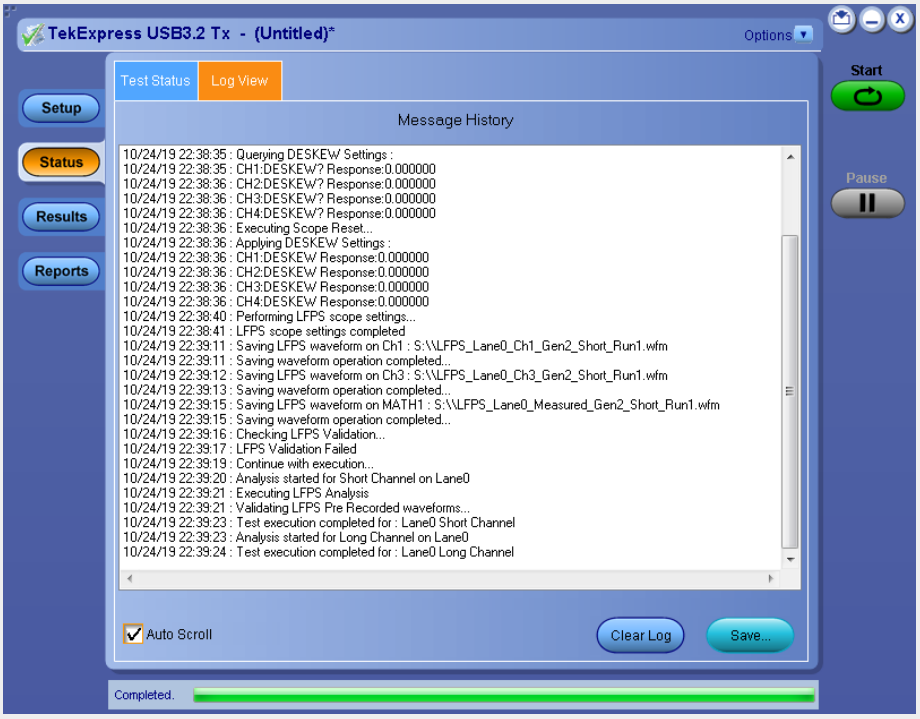


Table 12: Status panel settings

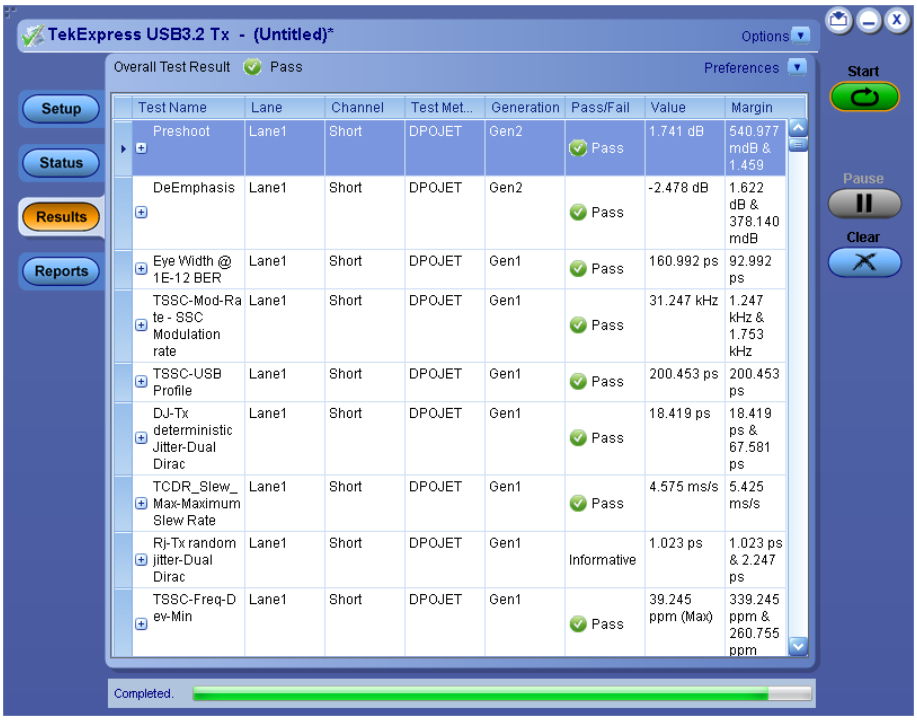
Control	Description
Message History	Window that lists all executed test operations and timestamp information.
Auto Scroll	Enables automatic scrolling of the log view as information is added to the log during the test.
Clear Log	Clears all messages from the log view.
Save	Saves the log file to a text file. Use the standard Save File window to navigate to and specify the folder and file name to which to save the log text.

See also. *Application panel overview*

Results panel

Results panel overview

When a test finishes, the application automatically opens the **Results** panel to display a summary of 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 or detailed results, and enabling wordwrap.

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 and collapse tests to show more or less detail, click the plus and minus buttons in the table.
- 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 clear all test results displayed, click **Clear**.
- Use the *Preferences menu* to change how some items display in the Results panel.

See also. *View a report*

Application panels overview

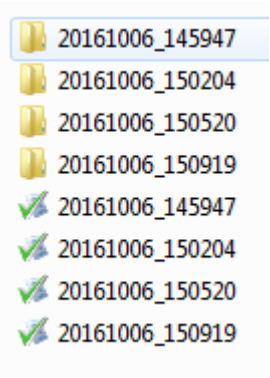
View test-related files

Files related to tests are stored in the My TekExpress\USB3.2 Tx folder. Each test setup in this folder has a test setup file and a test setup folder, both with the test setup name.

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

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

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



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

The first time you run a new, unsaved session, the session files are stored in the Untitled Session folder located at ..\My TekExpress\TekExpress USB3.2 Tx. 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 USB3.2 Tx application.

See also. *File name extensions*

Required My TekExpress folder settings

Preferences menu

The Preferences menu is part of the Results panel display. Use the Preferences menu to change how some items display in the Results panel.

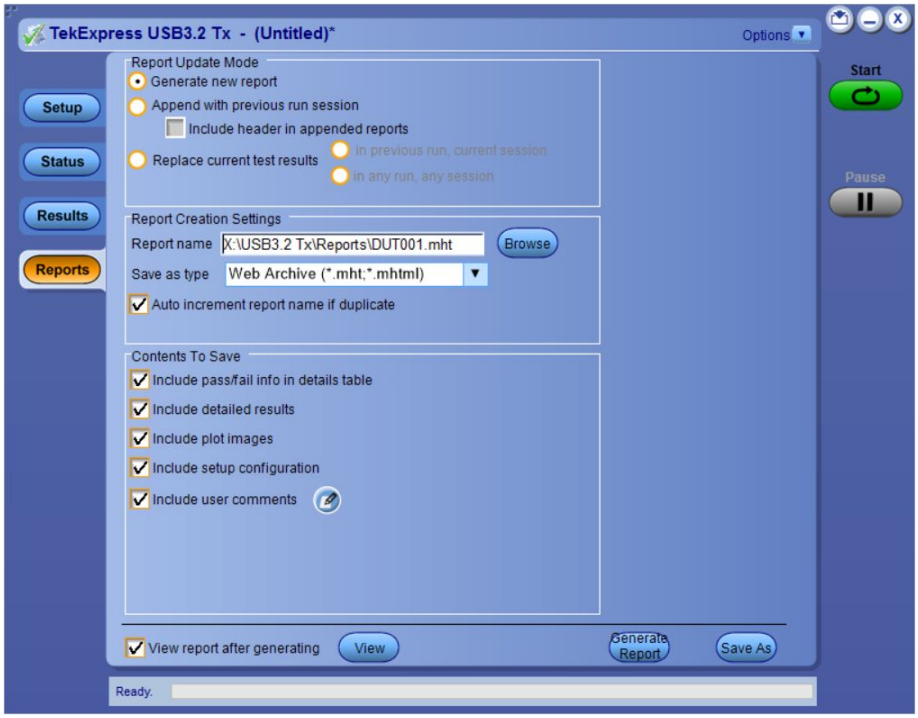
- To show or hide the Pass/Fail column, select **Preferences > Show Pass/Fail**.
- To collapse all expanded tests, select **Preferences > View Results Summary**.
- To expand all tests listed, select **Preferences > View Results Details**.
- To enable or disable the wordwrap feature, select **Preferences > Enable Wordwrap**.

See also. *Results panel overview*

Reports panel

Reports panel overview

Use the Reports panel to view saved reports, name and save reports from the current session, select test content to include in reports, and select report viewing options.



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

See also. *About panels*

Select report options

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

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

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

Table 13: Report options

Setting	Description
Report Update Mode	
Generate new report	Creates a new report.
Append with previous run session	Appends the latest test results to the end of the current session's test results report.
Include header in appended reports	Select to include header in the appended reports.
Replace current test in previous run session	Replaces the previous test results with the latest test results. Results from newly added tests are appended at the end of the report. ■ In previous run, current session: Select to replace the previous run of the current session. ■ In any run, any session: Select to replace the test of any previous run and session. Selecting this option enables a browse button which displays a list of previous sessions and runs to select.
Report Creation Settings	

Setting	Description
Report name	Displays the name and location from which to open a report. The default location is at *\My TekExpress\USB3.2 Tx\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\USB3.2 Tx\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 selected output format (Web archive, PDF or CSV).
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 info in details table	Select to include the column labeled Test Results (indicating whether the test passed or failed) in the report. For details, see Report Contents in *View a report*.
Include detailed results	Includes detailed results in the report
Include plot images	Sets the application to include plots such as Eye diagrams.

Setting	Description
Include setup configuration	Sets the application to include hardware and software information in the summary box at the top of the report. Information includes: the oscilloscope model and serial number, probe model and serial number, the oscilloscope firmware version, SPC and factory calibration status, and software versions for applications used in the measurements.
Include user comments	Select to include any comments about the test that you or another user added in the DUT tab of the Setup panel. Comments appear in the Comments section, under the summary box at the beginning of each report.
View Report After Generating	Automatically opens the report in a Web browser when the test completes. This option is selected by default.
View	Click to view the most current report.
Generate Report	Generates a new report based on the current (most-recent) analysis results.
Save As	Specify a name for the report.

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

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

Report contents

A report shows specified test details, such as detailed results and plots, as set 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.

Tektronix

TekExpress USB 3.2 Report

Test Report

Execution and Setup Information

DUT ID	DUT001	DUT Type	Device
Date/Time	2019-06-12 11:33:06	Scope Model	DPO73304DX
Test Point	Compliance (TP1) - Far End	Scope Serial Numer	B260572
Fixture Type	USB-IF	Scope F/W Version	10.8.3 Build 3
Connector Type	Type C	SPC Factory:S/W Calibration	PASS;PASS
Channel Type	Short	TekExpress Version	USB:10.3.5.20 Framework:4.9.0.5
Toggle Tool	Do not use	DPOJET Version	"10.0.8.179"
Acquisition Mode	PreRecorded	CTS Version	v1.0
SigTest Version	3.2.11.3(Gen1) & 4.0.23.2(Gen2)		
Total Acquisition Time	0 Seconds		
Total Analysis Time	3 Minutes 0 Seconds		
Over All Test Result	Pass		
DUT COMMENT:	General Comment - USB3.2 DUT		

UI-Unit Interval

Measurement Details	Lane	Channel	Generation	Method	Measured Value	Test Result	Margin	Low Limit	High Limit
UI-Unit Interval	Lane1	Short	Gen1	SigTest	200.446 ps	Pass	506.440 fs & 613.560 fs	199.94 ps	201.06 ps
COMMENTS	USB 3.2 Specification, Rev 1.0, Table 6-17								

Rj-Tx random jitter-Dual Dirac

Measurement Details	Lane	Channel	Generation	Method	Measured Value	Test Result	Margin	Low Limit	High Limit
Rj-Tx random jitter-Dual Dirac	Lane1	Short	Gen1	SigTest	1.015 ps	Informative	1.015 ps & 2.255 ps	0 s	3.27 ps
COMMENTS	USB 3.2 Specification, Rev 1.0, Table 6-19								

User comments

If you selected to include comments in the test report, any comments you added in the DUT tab are shown at the top of the report.

Test result summary

The Test Result column indicates whether a test passed or failed. If the test passed, the cell text is green. If the test failed, the text 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.

See also. *Results panel overview*

View test-related files

Running tests

Test process flow

Use the following list to set up and performing USB3.2 Tx tests.

1. Allow test instruments to warm up (~20 minutes).
2. *Deskew the real-time oscilloscope*.
3. *Set up test equipment*.
4. *Verify that required instruments are connected to USB3.2 Tx*.
5. *Set DUT parameters*.
6. *Select tests*.
7. *View acquisition settings*.
8. Set global signal-related parameters.
9. *Select test notification preferences*.
10. *Select report options*.
11. *Check the prerun checklist*
12. Click **Start** to *Run tests*.

See also *About test setups*

About running tests

Deskew real-time oscilloscopes

Use the following procedure to deskew direct input SMA channels on a real time oscilloscope.

NOTE. *DPOJET has an automatic deskew option under. Refer to your DPOJET online help for information on how to deskew the channels.*

1. Run Signal Path Compensation (SPC) on the oscilloscope.
2. Connect a SMA Power Splitter (preferred) or SMA 50 Ω coaxial "T" connector to the Fast Edge output of the oscilloscope.
3. Connect SMA cables from each of the two channels to be deskewed to the power splitter (or SMA coaxial "T" connector). It is best to use matched cables when making high speed serial measurements. **It is important to use the same cables during deskew that you will use for subsequent measurements**.
4. Select **Default Setup**, and then select **Autoset** on the oscilloscope front panel.
5. Set the oscilloscope for 70% to 90% full screen amplitude on both channels. Center both traces so that they overlap.
6. Make sure that volts/div, position, and offset are identical for the two channels being deskewed.
7. Set the time/div to approximately 100 ps/div or less, with sample rate at 1 ps/pt. These settings are not critical, but should be close.
8. Set the horizontal acquisition mode to average, which provides a more stable display.
9. Select **Deskew** from the **Vertical** menu.
10. Verify that the reference channel (typically CH1 or CH2) is set to 0 ps deskew.
11. In the deskew control window, select the channel to deskew (typically CH3 or CH4). Adjust the deskew to overlay the rising edge as best as possible.

NOTE. *Typical values are in the 10's of ps or less with cables connected directly from Fast Edge to SMA inputs. If you are using a switch box (for example, Keithley), deskew the complete path from where the test fixture connects, through the switch, and into the oscilloscope. Deskew values in these cases may be as much as 30 ps or more.*

NOTE. *There can be significant differences in the skew between two TCA-SMA adapters. If you find that a system requires a very large correction, obtain a pair of TCA-SMA adapters that closely match each other to reduce the amount of correction.*

NOTE. *TekExpress retains the user configured Deskew values, and does not override the values during test runs.*

Instrument and DUT connection setup

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

See also *Minimum system requirements*

View connected instruments

Running tests

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

The application displays a report when the tests are complete. While the tests are running, other applications may display windows in the background. The TekScope application takes precedence over other applications, but you can switch to other applications by using the **Alt + Tab** key combination. To keep the TekExpress USB3.2 Tx application on top, select **Keep On Top** from the TekExpress Options menu.

See also *Configuration tab parameters*

Prerun checklist

Do the following before you click Start to run a test:

NOTE. *If this is the first time you are running a test on the application, make sure that you have done the steps in Required My TekExpress folder settings before continuing.*

---

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**.
   c. Follow the on-screen instructions.
3. Verify that the correct instruments are connected (oscilloscope and signal sources):
   a. In USB3.2 Tx, click **Setup > Configuration**.
   b. Click **Global Settings**.
   c. In the **Instruments Detected** list, verify that the test setup instruments are shown. If they are not, click the arrow button to list and select from all detected instruments. If the required instrument is still not listed, use the TekExpress Instrument Control Settings dialog box to scan for and detect instruments (see *View connected instruments*).

See also *Instrument and DUT connection setup*

Saving and recalling test setup files

Test setup files overview

Saved test setup 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:\USB3.2 Tx**.

Use test setups to:

- Run a new session, acquiring live waveforms, using a saved test configuration.
- Create a new test setup based on an existing one.
- 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.
- Run a saved test using saved waveforms.

See also *Save a test setup*

Recall a saved test setup

Save a test setup file

Save a test setup before or after running a test to save the test settings. 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 immediately save the current setup session to the same setup name, select **Options > Save Test Setup**.

To immediately 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** to return the application to default test settings.
2. Click the application **Setup** button and use the setup tabs to set required options and parameters (DUT, Test Selection, and so on).
3. Click the application **Reports** button and set your *report options*.
4. Optional: Click **Start** to run the test and verify that it runs correctly and captures the specified test information and reports. 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 new setup file. The application saves the file to X:\USB3.2 Tx\<*session_name*>.

See also

Test process flow

View test-related files

Configuration tab parameters

Open (load) a saved test setup file

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:\USB3.2 Tx**.

See also

About test setups

Create a new test setup based on an existing one

Test setups overview

Run a saved test in prerecorded mode

Run a saved test in prerecorded mode

Use this option to rerun a complete test using just the oscilloscope and the saved test setup files, if you selected to save the captured waveforms when you originally ran the saved test.

NOTE. *When you run a saved test in prerecorded mode and then save it under the same name, the test results are saved in a new session folder named for the date and time of the session. Any test settings that you changed for the session are saved as a new test session file and be paired with a folder of the same name. Example. When you open a test setup that has multiple sessions and you select a session from the Run session list in the DUT tab, the settings associated with that test session are restored.*

Each test session folder has a matching test session file that stores the individual test settings for that session.

1. Use the Options menu to *Open a saved test setup file*
2. Select **Setup > DUT** and then select **Use pre-recorded waveform files**. A Run session drop-down list appears that displays the previous saved sessions for this test.
3. Select the session to run. NOTE. If you select a session for which no waveform files were saved, you will receive an error message. Either select another test session or select waveform files to use.
4. Click **Start**.
5. To save the test results, session settings, and related files, save the test setup before selecting another test setup or exiting the application.

See also

About test setups

Create a new test setup based on an existing one

Test setups overview

Create a new test setup file 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

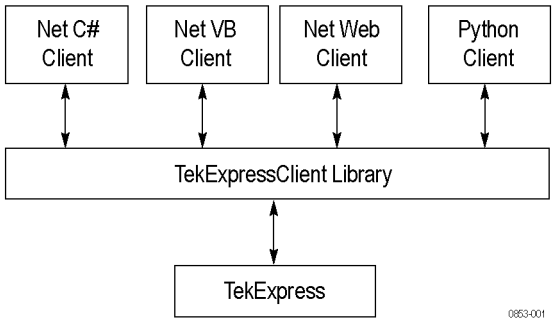
Set DUT parameters

Select acquisitions

TekExpress programmatic interface

About the programmatic interface

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



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

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

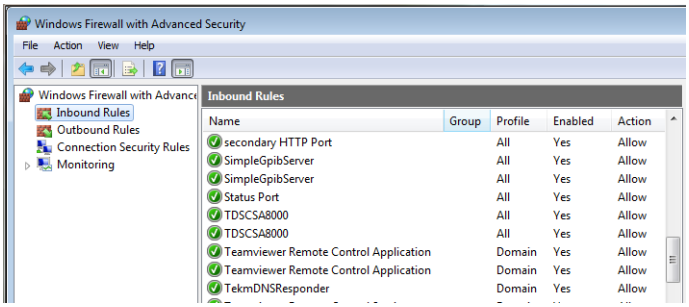
TekExpress leverages .Net Marshalling to enable the Programmatic Interface for TekExpress Client. TekExpress provides a client library for TekExpress clients to use the programmatic interface. The TekExpress client library is inherited from .Net MarshalByRef class to provide the proxy object for the clients. The TekExpress client library maintains a reference to the TekExpress Server and this reference allows the client to control the server state.

See also *Requirements for Developing TekExpress Client*

To enable remote access

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

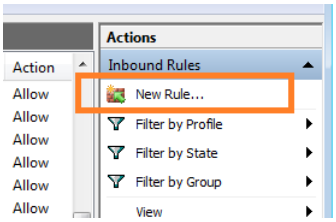
1. Access the Windows Control Panel and open the Windows Firewall tool (**Start > Control Panel > All Control Panel Items > Windows Firewall**).
2. Click **Advance Settings > Inbound Rules**.
3. Scroll through the **Inbound Rules** list to see if the following items (or with a similar name) are shown:
   - TekExpress USB3.2 Tx
   - TekExpress



4. If both items are shown, you do not need to set up any rules. Exit the Windows Firewall tool.
5. If one or both are missing, use the following procedure to run the **New Inbound Rule Wizard** and add these executables to the rules to enable remote access to the TekExpress application.
6. On the client side, include controller.exe through which TekExpress USB3.2 Tx application is remotely controlled. For example, if the application is controlled using python scripts the "ipy64.exe" should be included as part of Inbound rules.

Run the New Inbound Rule Wizard

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

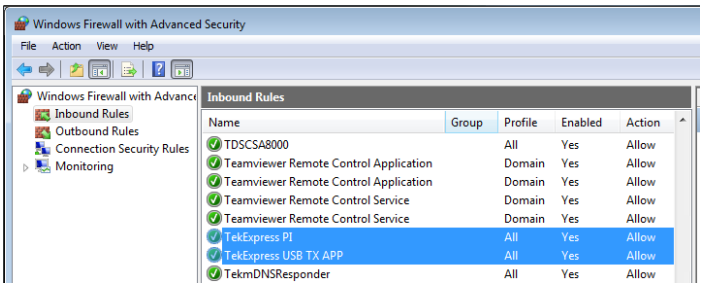


2. Verify that **Program** is selected in the Rule Type panel and click **Next**.
3. Click **Browse** in the Program panel and navigate to and select one of the following TekExpress applications (depending on the one for which you need to create a rule):
4. TekExpress USB3.2 Tx.exe
5. TekExpress.exe

NOTE. *See Application directories and contentfor 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, a name for the TekExpress USB3.2 Tx application could be **TekExpress USB3.2 Tx Application**. Add description text to further identify the rule.
10. Click **Finish** to return to the main Windows Firewall screen.
11. Scroll through the Inbound Rules list and verify that the list shows the rule that you just entered.



12. Repeat steps *1* through *11* to enter the other TekExpress executable if it is missing from the list. Enter **TekExpress PI** as the name.

13. Scroll through the Inbound Rules list and verify that the list shows the rule that you just entered.
14. Exit the Windows Firewall tool.

To use the remote access:

1. Obtain the IP address of the instrument on which you are running TekExpress USB. 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 USB3.2 Tx 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, Python, or Web application. The examples for interfaces in each of these applications are in the Samples folder.

References required

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

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

Required steps for a client

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

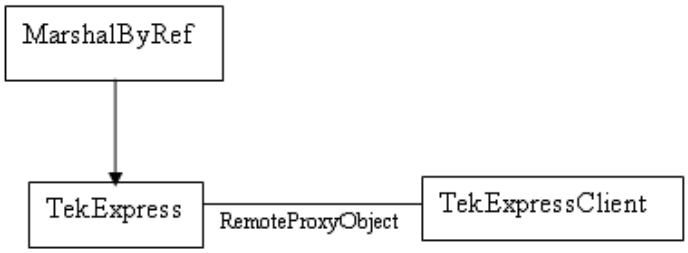
Develop a client UI to access the interfaces exposed through the server. This client loads TekExpressClient.dll to access the interfaces. After TekExpressClient.dll is loaded, the client UI can call the specific functions to run the operations requested by the client. When the client is up and running,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

1. To connect to the server, the client provides the IP address of the PC where the server is running.
2. The client locks the server application to avoid conflict with any other Client that may try to control the server simultaneously. “Lock” would also disable all user controls on the server so that server state cannot be changed by manual operation.

   If any other client tries to access a server that is locked, it will receive a notification that the server is locked by another client.
3. When the client has connected to and locked the server, the client can access any of the programmatic controls needed to run the remote automations.
4. After the client operations finish, the client unlocks the server.

Remote proxy object

The server exposes a remote object to let the remote client access and perform the server-side operations remotely. The proxy object is instantiated and exposed at the server-end through marshalling.



The following is an example:

RemotingConfiguration.RegisterWellKnownServiceType (typeof (TekExpressRemoteInterface), “TekExpress Remote interface”, WellKnownObjectMode.Singleton);

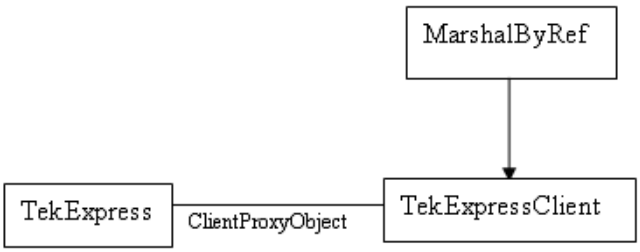
This object lets the remote client access the interfaces exposed at the server side. The client gets the reference to this object when the client gets connected to the server.

For example,

```
//Get a reference to the remote object
remoteObject =
(IRemoteInterface)Activator.GetObject(typeof(IRemoteInterface),
URL.ToString());
```

Client proxy object

Client exposes a proxy object to receive certain information.



For example,

```
//Register the client proxy object
WellKnownServiceTypeEntry[] e =
RemotingConfiguration.GetRegisteredWellKnownServiceTypes();
clientInterface = new ClientInterface();
RemotingConfiguration.RegisterWellKnownServiceType(typeof(ClientInterface)
, “Remote Client Interface”, WellKnownObjectMode.Singleton);
//Expose the client proxy object through marshalling
RemotingServices.Marshal(clientInterface, "Remote Client Inteface");
```

The client proxy object is used for the following:

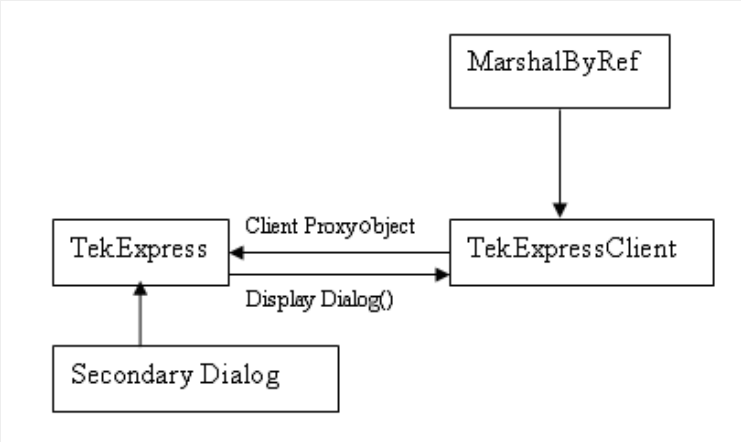
- To get the secondary dialog messages from the server.
- To get the file transfer commands from the server while transferring the report.

Examples

```
clientObject.clientIntf.DisplayDialog(caption, msg,iconType, btnType);
clientObject.clientIntf.TransferBytes(buffer, read, fileLength);
```

For more information, click the following links:

Secondary Dialog Message Handling



The secondary dialog messages from the Secondary Dialog library are redirected to the client-end when a client is performing the automations at the remote end.

In the secondary dialog library, the assembly that is calling for the dialog box to be displayed is checked and if a remote connection is detected, the messages are directed to the remote end.

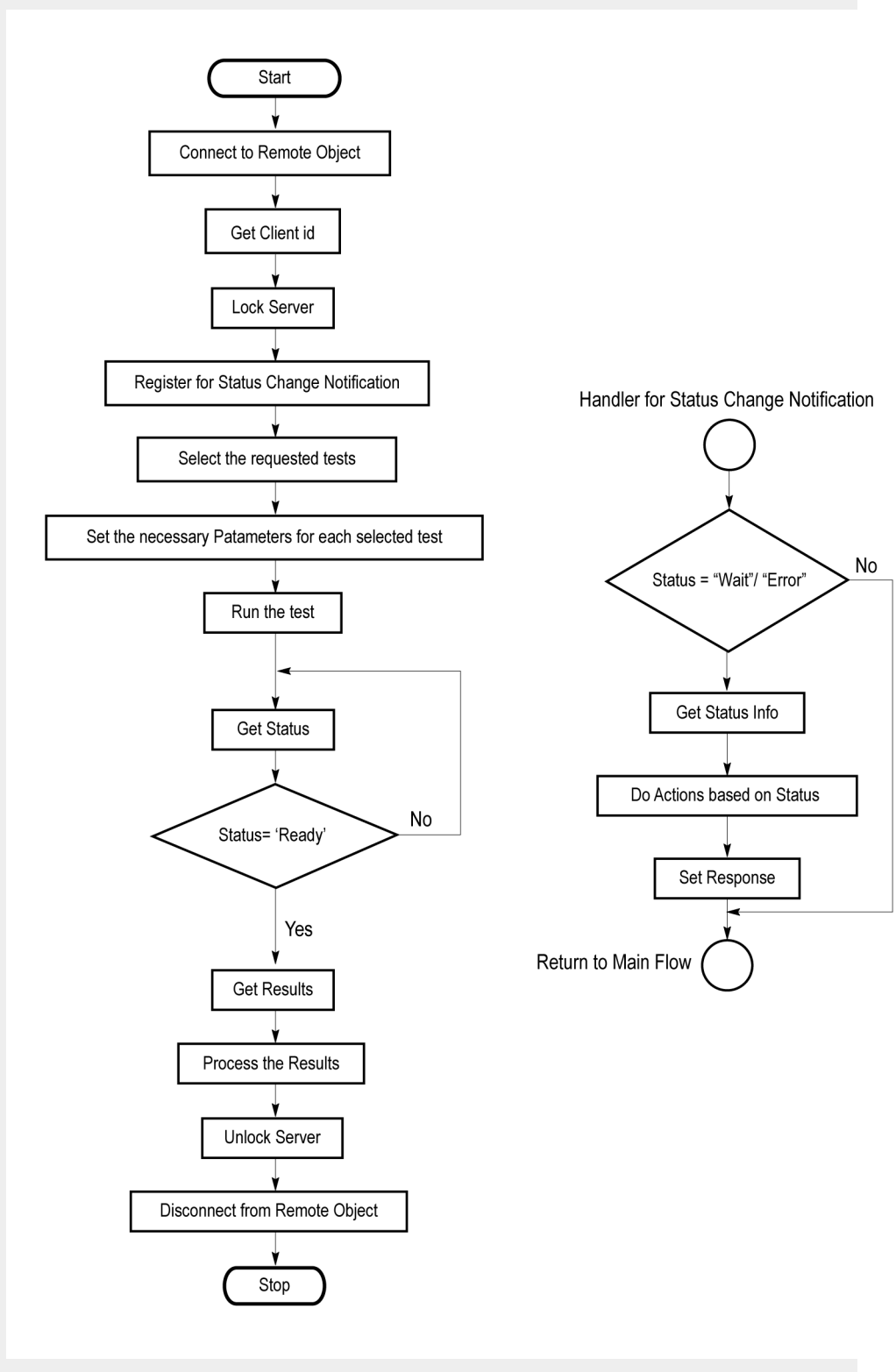
File Transfer Events

When the client requests the transfer of the report, the server reads the report and transfers the file by calling the file transfer methods at the client-end.

Client programmatic interface example

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

Process flowchart



1. Connect to a server or remote object using the programmatic interface provided.

2. Get the client ID that is created when connecting to the remote object. This client ID is one of the required parameters to communicate with the server.

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

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

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

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

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

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

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

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

Handler of status change notification

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

See also *Program remote access code example*

Program remote access code example

This code example shows how to communicate between a remote PC and TekExpress USB3.2 Tx.

Table 14: Remote access code example

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

USB-TX programmer interface commands

ApplicationStatus()

ApplicationStatus(clientId). This method gets the status (ready, running, paused) of the server application.

Parameters.

Name	Type	Direction	Description
clientId	string	IN	Identifier of the client that is performing the remote function. clientId example

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...".*

Return value. String value that gives the status of the server application.

Example. 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)

Comments. The application is in the Running, Paused, Wait, or Error state at any given time.

Related command(s). *GetCurrentStateInfo*

QueryStatus

SendResponse

Status

in string clientId example

clientId = <client_id_number>-<client_IP_address>.

For example, 1065–192.157.98.70

CheckSessionSaved()

CheckSessionSaved(clientID, savedStatus). This command checks whether the current session is saved.

Parameters.

Parameter	Type	Direction	Description
clientId	string	IN	Identifier of the client that is performing the remote function. clientId example
savedStatus	boolean	OUT	Boolean representing whether the current session is saved

Return value. Return value is either True or False.

Example. m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.

returnval as string

returnval=m_Client.CheckSessionSaved(m_clientID, out savedStatus)

Comments.

Related command(s). *RecallSession*

SaveSession

SaveSessionAs

in string clientId example

clientId = <client_id_number>-<client_IP_address>.

For example, 1065–192.157.98.70

Connect()

Connect(hostIPAddress, clientInterface, clientID). This command connects the client to the server. The client provides the IP address of the server to connect to the server. The server provides a unique clientId when the client is connected to it.

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

Parameters.

Parameter	Type	Direction	Description
HostIPAddress	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.
clientIntf	string	IN	The handle of the remote object interface
clientId	string	OUT	Identifier of the client that is performing the remote function. clientId example

Return value. Value that indicates the connection status (connection was established or an error occurred). The return value can be a boolean value (true), or a string (returning the error message).

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...".*

Example. try {

IPAddress[] hostIPAddr = Dns.GetHostAddresses(Dns.GetHostName());

// Connect to the remoter Server

remoteObject.Connect(hostIPAddress, clientInterface, out clientID);

return true;

}

catch (Exception error)

{

return error;

}

Comments. The server has to be active and running for the client to connect to the server. You can connect multiple clients to the server at a time. Each client is assigned a unique id.

Related command(s). *Disconnect*

in string clientId example

clientId = <client_id_number>-<client_IP_address>.

For example, 1065–192.157.98.70

Disconnect()

Disconnect(clientId). This command disconnects the client from the server it is connected to.

Parameters.

Parameter	Type	Direction	Description
clientId	string	IN	Identifier of the client that is performing the remote function. clientId example

Return value. Integer value that indicates the status of the operation upon completion.

1: Success

–1: Failure

Example. try

```
{
string returnVal = UnlockServer (clientId);
remoteObject.Disconnect (clientId);
return 1;
}
```

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

Related command(s). *Connect*

in string clientId example

clientId = <client_id_number>-<client_IP_address>.

For example, 1065–192.157.98.70

GetCurrentStateInfo()

GetCurrentStateInfo(clientID, WaitingMsbBxCaption, WaitingMsbBxMessage, WaitingMsbBxButtontexts). This command 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.

NOTE. *This command is used when the application is running and is in the wait or error state.*

Parameters.

Parameter	Type	Direction	Description
clientId	string	IN	Identifier of the client that is performing the remote function. clientId example
WaitingMsbBxCaption	string	OUT	The wait state or error state message sent to you
WaitingMsbBxMessage	string	OUT	The wait state/error state message sent to you
WaitingMsbBxButtontexts	string array	OUT	An array of strings containing the possible response types that you can send

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...".*

Return value. This command does not return any value.

This function populates the Out parameters that are passed when invoking this function.

Example. m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL

mClient.GetCurrentStateInfo(clientID, WaitingMsbBxCaption, WaitingMsbBxMessage, WaitingMsbBxButtontexts)

Comments.

Related command(s). *ApplicationStatus*

QueryStatus

SendResponse

in string clientId example

clientId = <client_id_number>-<client_IP_address>.

For example, 1065–192.157.98.70

GetDutId()

GetDutId(clientId, dutId). This command returns the DUT id of the current set-up.

Parameters.

Parameter	Type	Direction	Description
clientId	string	IN	Identifier of the client that is performing the remote function. clientId example
dutId	string	OUT	The DUT id of the set-up.

Return value. String that gives the timeout period (in seconds) of the client.

Example. returnVal = remoteObject.GetDutId(clientId, out dutId);

```
if ((OP_STATUS)returnVal == OP_STATUS.SUCCESS)
{
return id;
}
else
return CommandFailed(returnVal);
```

Comments. The dutId is an OUT parameter whose value is set after the server processes the request.

Related command(s). *SetDutId*

in string clientId example

clientId = <client_id_number>-<client_IP_address>.

For example, 1065–192.157.98.70

SetDutId()

SetDutId(clientID, newDutId). This command changes the DUT ID of the setup. The client must provide a valid DUT ID.

Parameters.

Parameter	Type	Direction	Description
clientId	string	IN	Identifier of the client that is connected to the server clientId example
newDutId	string	IN	The new DUT ID of the setup.

Return value. String that gives the status of the operation after it was performed.

Return value is “DUT Id Changed” on success.

Example. 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);

Comments.

Related command(s). *GetDutId*

in string clientId example

clientId = <client_id_number>-<client_IP_address>.

For example, 1065–192.157.98.70

GetGeneralParameter()

GetGeneralParameter(clientID, device, suite, test, paramString). This command gets the general parameter value based on the parameter name.

Parameters.

Parameter	Type	Direction	Description
clientId	string	IN	Identifier of the client that is performing the remote function. clientId example
device	string	IN	Device DUT type
suite	string	IN	Host type
test	string	IN	Specifies the name of the test for which to obtain Pass/Fail status or a test result value. Enter a null value for this field.
paramString	string	IN	Specifies the control to set.

Return value. String value that indicates the parameter value for a selected parameter name.

Example. m_Client.GetGeneralParameter(clientId, "Drive", "Transmitter", " ", "DUT control");

Where:

clientId = clientId

device ="Drive"

suite = "Transmitter"

paramString = "DUT control"

Related command(s). *SetGeneralParameter*

in string clientId example

clientId = <client_id_number>-<client_IP_address>.

For example, 1065–192.157.98.70

GetReportParameter()

GetReportParameter(clientId, device, suite, test, parameterString). This command gets the general report details such as oscilloscope model and TekExpress version.

Parameters.

Parameter	Type	Direction	Description
clientId	string	IN	Identifier of the client that is connected to the server clientId example
device	string	IN	Specifies the DUT type (**Host** or **Device**).
deviceConnector	string	IN	string with device connection type. Valid values are **Host Connector** and **Device Connector**
test	string	IN	Specifies the name of the test for which to obtain the pass or fail status or a test result value.
parameterString	string	IN	Specifies to return the measured value for the indicated test. Enter **"Scope Model"**, **"TekExpress Version"**, or **"Application Version"** for this argument

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...".*

Return value. The return value is the connected oscilloscope model, TekExpress base software version, or USB-TX application version.

Example. GetReportParameter(clientId, "Device", "Device Connector", test, "Application Version")

in string clientId example

clientId = <client_id_number>-<client_IP_address>.

For example, 1065–192.157.98.70

GetResultsValue()

GetResultsValue(clientId, device, deviceConnector, test, parameterString). This command gets the result values of the specified measurement after the run.

Parameters.

Parameter	Type	Direction	Description
clientId	string	IN	Identifier of the client that is connected to the server clientId example
device	string	IN	Specifies the DUT type (**Host** or **Device**).
deviceConnector	string	IN	string with device connection type. Valid values are **Host Connector** and **Device Connector**
test	string	IN	Specifies the name of the test for which to obtain the test result value.
parameterString	string	IN	Specifies to return the measured value for the indicated test. Enter **"Value"** for this argument

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...".*

Return value. String value that indicates the status of the operation upon completion. Returns the result value in the form of a string.

Example. GetResultsValue(clientId, "Device", "Device Connector", test, "Value");

in string clientId example

clientId = <client_id_number>-<client_IP_address>.

For example, 1065–192.157.98.70

GetSelectedVersions()

GetSelectedVersions(clientID, device, suite, versions). This command is used to select the particular version for a specific suite.

Parameters.

Parameter	Type	Direction	Description
clientId	string	IN	Identifier of the client that is performing the remote function clientId example
device	string	IN	Device DUT type
suite	string	IN	Host type
versions	string[ ]	IN	An array containing the versions of the specified site

Return value. Returns an empty string if the command is executed properly, otherwise returns a string as “Failed.”

in string clientId example

clientId = <client_id_number>-<client_IP_address>.

For example, 1065–192.157.98.70

Example. m_Client = new Client();

Note: m_Client is a reference to the Client class in the Client DLL

Versions as string= m_Client.GetSelectedVersions(clientId, Device, Suite, Version_Strings);

GetTimeOut()

GetTimeOut(clientId). Returns the current timeout period set by the client.

Parameters.

Parameter	Type	Direction	Description
clientId	string	IN	Identifier of the client that is connected to the server clientId example

Return value. String value that indicates the status of the operation upon completion. The default return value is 1800000. Returnval as string.

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...".*

Example. m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.

returnval as string

returnval=m_Client.GetTimeOut()

Comments.

Related command(s). *SetTimeOut*

in string clientId example

clientId = <client_id_number>-<client_IP_address>.

For example, 1065–192.157.98.70

LockServer()

LockServer(clientID). This command locks the server to which it is connected.

Parameters.

Parameter	Type	Direction	Description
clientId	string	IN	Identifier of the client that is performing the remote function clientId example

Return value. Integer value that indicates the status of the operation upon completion.

Example. try

{

string returnVal = remoteObject.lockServer(clientId);

remoteObject.connect(clientId);

return 1;

}

Related command(s). *UnlockServer*

in string clientId example

clientId = <client_id_number>-<client_IP_address>.

For example, 1065–192.157.98.70

LockSession()

LockSession(clientId). This command locks the server. The client has to call this command before running any of the remote automations. The server is locked by only one client.

Parameters.

Parameter	Type	Direction	Description
clientId	string	IN	Identifier of the client that is performing the remote function clientId example

Return value. Returns the status of the operation upon completion.

Example. if (locked)

return “Session has already been locked!”;

returnVal = remoteObject.LockSession(clientId);

if ((OP_STATUS)returnVal == OP_STATUS.SUCCESS)

{

locked = true;

return “Session Locked...”;

}

Comments. When the client tries to lock a server that is locked by another client, the client gets a message that the server is already locked and it has to 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 automatically unlocked from that client.

Related command(s). *UnlockSession*

in string clientId example

clientId = <client_id_number>-<client_IP_address>.

For example, 1065–192.157.98.70

QueryStatus()

QueryStatus(clientID, out status). This command transfers Analyze panel status messages from the server to the client.

Parameters.

Parameter	Type	Direction	Description
clientID	string	IN	Identifier of the client that is connected to the server clientId example
status	string array	OUT	The list of status messages generated during the run

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...".*

Return value. String value that indicates the status of the operation upon completion. On success the return value is “Transferred...”.

Example. returnVal=m_Client.QueryStatus(clientID, out statusMessages)

if ((OP_STATUS)returnVal == OP_STATUS.SUCCESS)

return “Status updated...”

else

return CommandFailed(returnVal)

Related command(s). *ApplicationStatus*

GetCurrentStateInfo

SendResponse

in string clientId example

clientId = <client_id_number>-<client_IP_address>.

For example, 1065–192.157.98.70

RecallSession()

RecallSession(clientId,sessionName). Recalls a saved session. The name of the session is provided by the client.

Parameters.

Parameter	Type	Direction	Description
clientId	string	IN	Identifier of the client that is performing the remote function. clientId example
sessionName	string	IN	The name of the session being recalled.

Return value. String that indicates the status of the operation upon completion.

Example. returnVal = remoteObject.RecallSession(clientId,sessionName);

if ((OP_STATUS)returnVal == OP_STATUS.SUCCESS)

return “Session Recalled...”;

else

return CommandFailed(returnVal);

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

Related command(s). *SaveSession*

SaveSessionAs

in string clientId example

clientId = <client_id_number>-<client_IP_address>.

For example, 1065–192.157.98.70

RegisterStatusChangeNotification()

RegisterStatusChangeNotification(clientID, statusChangeHandler). There are two ways to poll the application when it comes out of the Busy state. This command registers when there is an event, which indicates that activity is complete.

This command is used to select the particular version for a specific suite.

Parameters.

Parameter	Type	Direction	Description
clientId	string	IN	Identifier of the client that is performing the remote function clientId example
statusChangeHandler	Delegate of type TekExpressClient.StatusChangeHandler	IN	Handler

Return value. Returns an empty string when the operation is successful; otherwise it returns an error description.

Example. m_Client.RegisterStatusChangeNotification(clientId, new TekExpressClient.StatusChangeHandler (OnStatusChange));

```
public void OnStatusChange(string _status)
 {
 _status = m_Client.Application_Status(clientId);
 if (_status.CompareTo("Wait") == 0 || _status.CompareTo("Error") == 0)
   {
   string caption = "", message = "";
   string[] buttonTexts = null;
   m_Client.GetCurrentStateInfo(clientId, out caption, out message, out buttonTexts);
   Console.WriteLine("Caption:" + caption);
   Console.WriteLine("Message:" + message);
   Console.WriteLine("Message Type:" + FormatStringArray(buttonTexts));
   Console.WriteLine("Press Enter to send response . Waiting for Response...");
   string response = Console.ReadLine();
   m_Client.SendResponse(clientId, caption, message, response);
   Console.WriteLine("Message Response " + response + " Sent");
   }
 }
```

in string clientId example

clientId = <client_id_number>-<client_IP_address>.

For example, 1065–192.157.98.70

Run()

Run(clientId). Runs the setup. Once the server is set up and configured, it can be run remotely using this function.

Parameters.

Parameter	Type	Direction	Description
clientId	string	IN	Identifier of the client that is connected to the server clientId example

Return value. String that returns the status of the operation after completion.

Example. returnVal = remoteObject.Run(clientId);

if ((OP_STATUS)returnVal == OP_STATUS.SUCCESS)

return “Run started...”;

else

return CommandFailed(returnVal);

Comments. When the run is performed the status of the run is updated periodically using a timer.

Related command(s). *Stop*

in string clientId example

clientId = <client_id_number>-<client_IP_address>.

For example, 1065–192.157.98.70

SaveSession()

SaveSession(clientId,sessionName). Saves the current session. The name of the session is provided by the client.

Parameters.

Parameter	Type	Direction	Description
clientId	string	IN	Identifier of the client that is connected to the server clientId example
sessionName	string	IN	The name of the session being saved.

Return value. String that indicates the status of the operation upon completion.

Example. returnVal = remoteObject.SaveSession(clientId,sessionName);

if ((OP_STATUS)returnVal == OP_STATUS.SUCCESS)

return “Session Saved...”;

else

return CommandFailed(returnVal);

Comments. 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 command to save the session with a different name. Use SaveSessionAs to save the session to a new name.

Related command(s). *RecallSession*

SaveSessionAs

in string clientId example

clientId = <client_id_number>-<client_IP_address>.

For example, 1065–192.157.98.70

SaveSessionAs()

SaveSessionAs(clientId,sessionName). Saves the current session in a different name every time this command is called. The name of the session is provided by the client.

Parameters.

Parameter	Type	Direction	Description
clientId	string	IN	Identifier of the client that is connected to the server clientId example
sessionName	string	IN	The name of the session being saved.

Return value. String that indicates the status of the operation upon completion.

Example. returnVal = remoteObject.SaveSessionAs(clientId,sessionName);

if ((OP_STATUS)returnVal == OP_STATUS.SUCCESS)

return "Session Saved...";

else

return CommandFailed(returnVal);

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

in string clientId example

clientId = <client_id_number>-<client_IP_address>.

For example, 1065–192.157.98.70

Related command(s). *RecallSession*

SaveSession

SelectSingleTest()

SelectSingleTest(clientID, device, suite, version, test). This command is to select a single test from a group of tests.

Parameters.

Parameter	Type	Direction	Description
clientId	string	IN	Identifier of the client that is performing the remote function clientId example
device	string	IN	Device DUT type
suite	string	IN	Host type
version	string	IN	Enter a null value for this field
test	string	IN	Name of the test

Return value. Returns an empty string if the command is executed properly, otherwise returns the string “Failed.”

Example. m_Client = new Client()

Note: m_Client is a reference to the Client class in the Client DLL.

To return a string:

returnval=m_Client.SelectSingleTest(clientId, device, suite, Version, test)

Where:

clientId = clientId

device = "Device" or "Host"

suite = "Device Connector" or "Host Connector"

Version= "" (null)

test = "UI-Unit Interval"

in string clientId example

clientId = <client_id_number>-<client_IP_address>.

For example, 1065–192.157.98.70

SendResponse()

SendResponse(clientID, WaitingMsbBxCaption, WaitingMsbBxMessage, WaitingMsbBxButtontexts). After receiving the additional information using the command 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.

NOTE. *This command is used when the application is running and is in the wait or error state.*

Parameters.

Parameter	Type	Direction	Description
clientId	string	IN	Identifier of the client that is connected to the server clientId example
WaitingMsbBxCaption	string	OUT	The wait state or error state message sent to you
WaitingMsbBxMessage	string	OUT	The wait state/error state message sent to you
WaitingMsbBxButtontexts	string array	OUT	An array of strings containing the possible response types that you can send

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...".*

Return value. This command does not return any value.

Example. m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL

mClient.SendResponse(clientID, out WaitingMsbBxCaption, out WaitingMsbBxMessage, out WaitingMsbBxButtontexts)

Related command(s). *ApplicationStatus*

GetCurrentStateInfo

QueryStatus

in string clientId example

clientId = <client_id_number>-<client_IP_address>.

For example, 1065–192.157.98.70

SelectDevice()

SelectDevice(clientId, device, true). This command selects the DUT type (Host or Device).

Parameters.

Parameter	Type	Direction	Description
clientId	string	IN	Identifier of the client that is connected to the server clientId example
device	string	IN	String with the device DUT type. Valid values are **Host** and **Device**.

Return value. String value that indicates the status of the operation upon completion.

Example. SelectDevice(clientId, "Device", true);

SelectDevice(clientId, "Host", true);

in string clientId example

clientId = <client_id_number>-<client_IP_address>.

For example, 1065–192.157.98.70

SelectSuite()

SelectSuite(clientId, device, deviceConnector, true). This command selects one of the two suites: "Device Connector" or "Host Connector."

Parameters.

Parameter	Type	Direction	Description
clientId	string	IN	Identifier of the client that is connected to the server clientId example
device	string	IN	String with the device DUT type. Valid values are **Host** and **Device**.
deviceConnector	string	IN	string with device connection type. Valid values are **Host Connector** and **Device Connector**

Return value. String value that indicates the status of the operation upon completion.

Example. SelectSuite(clientId,"Device","Device Connector",true);

SelectSuite(clientId,"Device","Host Connector",true);

SelectSuite(clientId,"Host","Device Connector",true);

SelectSuite(clientId,"Host","Host Connector",true);

in string clientId example

clientId = <client_id_number>-<client_IP_address>.

For example, 1065–192.157.98.70

SelectTest()

SelectTest(clientId, device, deviceConnector, test, true). This command selects a test.

Parameters.

Parameter	Type	Direction	Description
clientId	string	IN	Identifier of the client that is connected to the server clientId example
device	string	IN	String with the device DUT type. Valid values are **Host** and **Device**.
deviceConnector	string	IN	string with device connection type. Valid values are **Host Connector** and **Device Connector**
test	string	IN	Name of the USB-TX/ USBSSP-Tx test as listed in the application UI for Gen1 measurements For USBSSP-Tx (Gen2) measurements, add the post-fix '_10Gbps' to the test name (without the quotes)

Return value. String value that indicates the status of the operation upon completion.

Example. SelectTest(clientId, device, deviceConnector, "UI-Unit Interval", true);

in string clientId example

clientId = <client_id_number>-<client_IP_address>.

For example, 1065–192.157.98.70

test values for SelectTest command

Table 15: Test Name for Test Point - Compliance (TP4) - Far End

test values for SelectTest command	Relevant test name in the UI
UI-Unit Interval	UI-Unit Interval
VTx-Diff-PP-Differential PP Tx voltage swing	VTx-Diff-PP-Differential PP Tx voltage swing
TCDR_Slew_Max-Maximum Slew Rate	TCDR_Slew_Max-Maximum Slew Rate
Rj-Tx random jitter-Dual Dirac	Rj-Tx random jitter-Dual Dirac
Mask Hits	Mask Hits
TSSC-Freq-Dev-Max	TSSC-Freq-Dev-Max
TSSC-Freq-Dev-Min	TSSC-Freq-Dev-Min
TSSC-Mod-Rate - SSC Modulation rate	TSSC-Mod-Rate - SSC Modulation rate
TSSC-USB Profile	TSSC-USB Profile
DJ-Tx deterministic Jitter-Dual Dirac	DJ-Tx deterministic Jitter-Dual Dirac
TJ-Tx total jitter-Dual Dirac at 10E-12 BER	TJ-Tx total jitter-Dual Dirac at 10E-12 BER
Eye Height - Transmitter Eye Mask	Eye Height - Transmitter Eye Mask
Width@BER	Eye Width @ 10E-12 BER
UI-Unit Interval_10Gbps	UI-Unit Interval_10Gbps
VTx-Diff-PP-Differential PP Tx voltage swing_10Gbps	VTx-Diff-PP-Differential PP Tx voltage swing_10Gbps
Rj-Tx random jitter-Dual Dirac_10Gbps	Rj-Tx random jitter-Dual Dirac_10Gbps
Mask Hits_10Gbps	Mask Hits_10Gbps
TSSC-Freq-Dev-Max_10Gbps	TSSC-Freq-Dev-Max_10Gbps
TSSC-Freq-Dev-Min_10Gbps	TSSC-Freq-Dev-Min_10Gbps
TSSC-Mod-Rate - SSC Modulation rate_10Gbps	TSSC-Mod-Rate - SSC Modulation rate_10Gbps
TSSC-USB Profile_10Gbps	TSSC-USB Profile_10Gbps
SSC_dfdt_10Gbps	SSC_dfdt_10Gbps
DJ-Tx deterministic Jitter-Dual Dirac_10Gbps	DJ-Tx deterministic Jitter-Dual Dirac_10Gbps
TJ-Tx total jitter-Dual Dirac at 10E-12 BER_10Gbps	TJ-Tx total jitter-Dual Dirac at 10E-12 BER_10Gbps
Eye Height - Transmitter Eye Mask_10Gbps	Eye Height - Transmitter Eye Mask_10Gbps
Width@BER_10Gbps	Eye Width @ 10E-6 BER_10Gbps
Height@BER_10Gbps	Eye Height @ 10E-6 BER_10Gbps
Preshoot_10Gbps	Preshoot
DeEmphasis_10Gbps	DeEmphasis
LFPS Duty Cycle	LFPS Duty Cycle
LFPS Fall Time	LFPS Fall Time
LFPS Rise Time	LFPS Rise Time
LFPS TPeriod	LFPS Tperiod
LFPS Vcm-AC	LFPS Vcm-AC
LFPS Vtx-DIFF-PP	LFPS Vtx-DIFF-PP

test values for SelectTest command	Relevant test name in the UI
LFPS TBurst	LFPS Tburst
LFPS TRepeat	LFPS Trepeat

Table 16: Test Name for Test Point - Tx Pins - Near End or Custom

test values for SelectTest command	Relevant test name in the UI
UI-Unit Interval	UI-Unit Interval
VTx-Diff-PP-Differential PP Tx voltage swing	VTx-Diff-PP-Differential PP Tx voltage swing
TCDR_Slew_Max-Maximum Slew Rate	TCDR_Slew_Max-Maximum Slew Rate
Rj-Tx random jitter-Dual Dirac	Rj-Tx random jitter-Dual Dirac
Mask Hits-NTBit	Mask Hits-NTBit
Mask Hits-Tbit	Mask Hits-Tbit
TSSC-Freq-Dev-Max	TSSC-Freq-Dev-Max
TSSC-Freq-Dev-Min	TSSC-Freq-Dev-Min
TSSC-Mod-Rate - SSC Modulation rate	TSSC-Mod-Rate - SSC Modulation rate
TSSC-USB Profile	TSSC-USB Profile
DJ-Tx deterministic Jitter-Dual Dirac	DJ-Tx deterministic Jitter-Dual Dirac
TJ-Tx total jitter-Dual Dirac at 10E-12 BER	TJ-Tx total jitter-Dual Dirac at 10E-12 BER
Eye Height-NTBit	Eye Height-NTBit
Eye Height-Tbit	Eye Height-Tbit
Width@BER	Eye Width @ 10E-12 BER
Tmin-Pulse-Tj - Tx min pulse	Tmin-Pulse-Tj - Tx min pulse
VTx-CM-AC-PP-Active-Tx AC common mode voltage active	VTx-CM-AC-PP-Active-Tx AC common mode voltage active
VTx-DC-CM-Tx DC common-mode voltage	VTx-DC-CM-Tx DC common-mode voltage
VTx-De-Ratio-Tx De-emphasis	VTx-De-Ratio-Tx De-emphasis
UI-Unit Interval_10Gbps	UI-Unit Interval_10Gbps
VTx-Diff-PP-Differential PP Tx voltage swing_10Gbps	VTx-Diff-PP-Differential PP Tx voltage swing_10Gbps
Rj-Tx random jitter-Dual Dirac_10Gbps	Rj-Tx random jitter-Dual Dirac_10Gbps
Mask Hits-NTBit_10Gbps	Mask Hits-NTBit_10Gbps
Mask Hits-Tbit_10Gbps	Mask Hits-Tbit_10Gbps
TSSC-Freq-Dev-Max_10Gbps	TSSC-Freq-Dev-Max_10Gbps
TSSC-Freq-Dev-Min_10Gbps	TSSC-Freq-Dev-Min_10Gbps
TSSC-Mod-Rate - SSC Modulation rate_10Gbps	TSSC-Mod-Rate - SSC Modulation rate_10Gbps
TSSC-USB Profile_10Gbps	TSSC-USB Profile_10Gbps
SSC_dfdt_10Gbps	SSC_dfdt_10Gbps
DJ-Tx deterministic Jitter-Dual Dirac_10Gbps	DJ-Tx deterministic Jitter-Dual Dirac_10Gbps
TJ-Tx total jitter-Dual Dirac at 10E-12 BER_10Gbps	TJ-Tx total jitter-Dual Dirac at 10E-12 BER_10Gbps

test values for SelectTest command	Relevant test name in the UI
Eye Height-NTBit_10Gbps	Eye Height-NTBit_10Gbps
Eye Height-Tbit_10Gbps	Eye Height-Tbit_10Gbps
Tmin-Pulse-Tj - Tx min pulse_10Gbps	Tmin-Pulse-Tj - Tx min pulse_10Gbps
Width@BER_10Gbps	Eye Width @ 10E-6 BER_10Gbps
Height@BER_10Gbps	Eye Height @ 10E-6 BER_10Gbps
VTx-CM-AC-PP-Active-Tx AC common mode voltage active_10Gbps	VTx-CM-AC-PP-Active-Tx AC common mode voltage active_10Gbps
VTx-DC-CM-Tx DC common-mode voltage_10Gbps	VTx-DC-CM-Tx DC common-mode voltage_10Gbps
Preshoot_10Gbps	Preshoot
DeEmphasis_10Gbps	DeEmphasis
LFPS Duty Cycle	LFPS Duty Cycle
LFPS Fall Time	LFPS Fall Time
LFPS Rise Time	LFPS Rise Time
LFPS Tperiod	LFPS Tperiod
LFPS Vcm-AC	LFPS Vcm-AC
LFPS Vtx-DIFF-PP	LFPS Vtx-DIFF-PP
LFPS Tburst	LFPS Tburst
LFPS Trepeat	LFPS Trepeat

NOTE. *Single-Space at the end of the test is mandatory for* ***Tx Pins - Near End*** *and* ***Custom*** *test point.*

SetInstrument()

SetInstrument(clientID, device, suite, test, paramString). Sets the specified instrument as a general configuration parameter to the selected test.

Parameters.

Parameter	Type	Direction	Description
clientId	string	IN	Identifier of the client that is performing the remote function clientId example
device	string	IN	Device DUT type
suite	string	IN	Host type
test	string	IN	Name of the test
paramString	string	IN	Specifies the control to set

Return value. Returns the string value of the instrument specified for setting in configuration parameter.

Example. mClient = new Client()

Dim clientId As String

Dim DUTType As String = “Device” or “Host”

Dim TekExpress_Suite As String = “Device Connector” or “Host Connector”

Dim str As String

Str= mClient.SetInstrument(clientId, DUTType, TekExpress_Suite, "UI-Unit Interval", " AnalyzeInstrument$Real Time Scope$ DPO71254B (GPIB0::1::INSTR)")

in string clientId example

clientId = <client_id_number>-<client_IP_address>.

For example, 1065–192.157.98.70

SetPreRecorded()

SetPreRecorded(clientID, bset, ERRORString). This command selects the “Use pre-recorded waveform files” control in the DUT tab of the application UI.

Parameters.

Parameter	Type	Direction	Description
clientId	string	IN	Identifier of the client that is performing the remote function clientId example
bset	string	IN	This should be “True” or “False” based on the condition
ERRORString	string	IN	Error message to print if the command did not execute

Return value. 1 if pass, -1 if fail.

Example. m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.

returnval as Integer = m_Client.SetPrerecorded(clientId, True, “ ”)

Where:

clientId = clientId

bset= True

Error= ""

Comments. Use *RecallSession()* before using this command.

in string clientId example

clientId = <client_id_number>-<client_IP_address>.

For example, 1065–192.157.98.70

SetTimeOut()

SetTimeOut(clientId, time). Sets a timeout period specified by the client. After this timeout period expires, the server is unlocked automatically.

Parameters.

Parameter	Type	Direction	Description
clientId	string	IN	Identifier of the client that is connected to the server clientId example
time	string	IN	The time in seconds that refers to the timeout period

Return value. String value that indicates the status of the operation upon completion. On success the return value is “TimeOut Period Changed”.

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...".*

Example. 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, time)

Comments.

Related command(s). *GetTimeOut*

in string clientId example

clientId = <client_id_number>-<client_IP_address>.

For example, 1065–192.157.98.70

SetVerboseMode()

SetVerboseMode(clientId, verboseMode). This command 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.

Parameters.

Parameter	Type	Direction	Description
clientId	string	IN	Identifier of the client that is connected to the server clientId example
verboseMode	boolean	IN	Sets the verbose mode to be turned ON (true) or OFF (false).

Return value. String that gives the status of the operation after it was performed. Returnval as string.

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”.

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...".*

Example. m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL.

Turn on verbose mode:

return=m_Client.SetVerboseMode(clientId, true);

Turn off verbose mode:

returnval=m_Client.SetVerboseMode(clientId, false);

in string clientId example

clientId = <client_id_number>-<client_IP_address>.

For example, 1065–192.157.98.70

Status()

Status(clientId, out statusMessages). This command gives the status of the run as messages. The status messages are generated once the run is started.

Parameters.

Parameter	Type	Direction	Description
clientId	string	IN	Identifier of the client that is connected to the server clientId example
statusMessage	string array	OUT	The list of status messages generated during run.

Return value. String that indicates the status of the operation upon completion.

Example. returnVal = remoteObject.QueryStatus(clientId, out statusMessages);

if ((OP_STATUS)returnVal == OP_STATUS.SUCCESS)

return “Status updated...”;

else

return CommandFailed(returnVal);

Comments. The status messages are updated periodically after the run begins. The status is an out parameter which is set when the server processes the request.

Related command(s). *ApplicationStatus*

in string clientId example

clientId = <client_id_number>-<client_IP_address>.

For example, 1065–192.157.98.70

Stop()

Stop(clientId). Stops the run operation.

Parameters.

Parameter	Type	Direction	Description
clientId	string	IN	Identifier of the client that is connected to the server clientId example

Return value. String that indicates the status of the operation upon completion.

Example. returnVal = remoteObject.Stop(clientId);

if ((OP_STATUS)returnVal == OP_STATUS.SUCCESS)

return “Stopped...”;

else

return CommandFailed(returnVal);

Comments. When the session is stopped the client is prompted to stop the session and is stopped at the consent.

Related command(s). *Run*

in string clientId example

clientId = <client_id_number>-<client_IP_address>.

For example, 1065–192.157.98.70

TransferImages()

TransferImages(clientId, filePath). This command transfers all the images (screen shots) to the specified client and folder (directory) from the current run.

NOTE. *Every time you click Start, a folder is created in the X: drive. Transfer the waveforms before clicking Start.*

Parameters.

Parameter	Type	Direction	Description
clientId	string	IN	Identifier of the client that is connected to the server clientId example
filePath	string	IN	The location where the screen shots 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. *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...".*

Return value. String value that indicates the status of the operation upon completion. Transfers all the images in the form of a string.

Example. TransferImages(clientId, "C:\Images”)

in string clientId example

clientId = <client_id_number>-<client_IP_address>.

For example, 1065–192.157.98.70

TransferResult()

TransferResult(clientID, Filepath). Transfers (saves) the result from the results panel information to the specified path.

Parameters.

Parameter	Type	Direction	Description
clientId	string	IN	Identifier of the client that is performing the remote function clientId example
Filepath	string	IN	Specifies the destination path of the file to be saved

Return value. Return a string as “Transferred…”

Example. m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL

TransferResult as string = m_Client.TransferResult(clientId, “C:\abc\Results\”);

in string clientId example

clientId = <client_id_number>-<client_IP_address>.

For example, 1065–192.157.98.70

TransferWaveforms()

TransferWaveforms(clientID, path). This command transfers all the acquired waveforms to the specified location.

Parameters.

Parameter	Type	Direction	Description
clientId	string	IN	Identifier of the client that is performing the remote function clientId example
path	string	IN	Path to location at which to store waveforms

Return value. Returns a string as “Transferred…”

Example. m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL

TransferWaveforms as string = m_Client.TransferWaveforms(clientId, “C:\abc\Waveforms\”);

in string clientId example

clientId = <client_id_number>-<client_IP_address>.

For example, 1065–192.157.98.70

UnlockServer()

UnlockServer(clientID). This command unlocks the server to which it is connected.

Parameters.

Parameter	Type	Direction	Description
clientId	string	IN	Identifier of the client that is performing the remote function clientId example

Return value. Returns and integer value that indicates the status of the operation upon completion. Session UnLocked…

Example. try

```
{
string returnVal = remoteObject.UnlockServer (clientId);
remoteObject.disconnect (clientId);
return 1;
}
```

Comments. When the client is disconnected, it is unlocked from the server and then disconnected. The ID is reused.

in string clientId example

clientId = <client_id_number>-<client_IP_address>.

For example, 1065–192.157.98.70

UnlockSession()

UnlockSession(clientId). This command unlocks the server from the client. The client id of the client to be unlocked has to be provided.

Parameters.

Parameter	Type	Direction	Description
clientId	string	IN	Identifier of the client that is connected to the server clientId example

Return value. String that indicates the status of the operation upon completion.

Example. returnVal = remoteObject.UnlockSession(clientId);

```
if ((OP_STATUS)returnVal == OP_STATUS.SUCCESS)
{
locked = false;
return “Session UnLocked...”;
}
```

Comments. When the client is disconnected, it is automatically unlocked.

Related commands. *LockSession*

in string clientId example

clientId = <client_id_number>-<client_IP_address>.

For example, 1065–192.157.98.70

GetPassFailStatus()

GetPassFailStatus(clientId, device, deviceConnector, test). This command gets the pass or fail status of the measurement after test completion.

NOTE. *Execute this command after completing the measurement.*

Parameters.

Parameter	Type	Direction	Description
clientId	string	IN	Identifier of the client that is connected to the server clientId example
device	string	IN	Specifies the DUT type (**Host** or **Device**).
deviceConnector	string	IN	string with device connection type. Valid values are **Host Connector** and **Device Connector**
test	string	IN	Specifies the name of the test for which to obtain the pass or fail status.

Return value. String value that indicates the status of the operation upon completion.

Example. GetPassFailStatus(clientId, “Device”, “Device Connector”, test);

GetPassFailStatus(clientId, “Host”, “Host Connector”, test);

in string clientId example

clientId = <client_id_number>-<client_IP_address>.

For example, 1065–192.157.98.70

SetGeneralParameter()

SetGeneralParameter(clientId, device, deviceConnection, "", paramString). This command sets the general parameter and its value based on the "paramString" argument values as listed.

Parameters.

Parameter	Type	Direction	Description
clientId	string	IN	Identifier of the client that is connected to the server clientId example
device	string	IN	Specifies the DUT type (**Host** or **Device**).
deviceConnector	string	IN	string with device connection type. Valid values are **Host Connector** and **Device Connector**
test	string	IN	Specifies the name of the test for which to obtain the pass or fail status or a test result value. Enter a null value for this field (""").
parameterString	string	IN	Specifies the control to set. See the following links for argument values and examples for this field.

Return value. String value that indicates the status of the operation upon completion.

in string clientId example

clientId = <client_id_number>-<client_IP_address>.

For example, 1065–192.157.98.70

See also. *paramString values for SetGeneralParameter command*

paramString values for SetGeneralParameter command

Select version. Use this paramString value to set the version. This is the same as using the **Version** controls on the **DUT** tab.

The value in bold font is the default value.

paramString value	<String>
Generation Version$<String>	**Generation Version$USB3.2 Gen1** or Generation Version$USB3.2 Gen2

Select data rates. Use this paramString value to set the data rates. This is the same as using the **Data Rates** controls on the **DUT** tab.

The value in bold font is the default value.

paramString value	<String>
DataRate 5Gbps Option Button$<String>	**Included** or Excluded
DataRate 10Gbps Option Button$<String>	**Included** or Excluded

Example. SetGeneralParameter(clientId, device, devicesuite, "", "DataRate 5Gbps Option Button$Included");

SetGeneralParameter(clientId, device, devicesuite, "", "DataRate 10Gbps Option Button$Included");

Select channel. Use this paramString value to set the channel type. This is the same as using the **Channel** controls on the **DUT** tab.

The value in bold font is the default value.

paramString value	<String>
Channel$<String>	Long, Short or **Both**

Example. SetGeneralParameter(clientId, device, devicesuite, "", "Channel $Long");

SetGeneralParameter(clientId, device, devicesuite, "", "Channel$Short");

SetGeneralParameter(clientId, device, devicesuite, "", "Channel$Both");

Select test mode. Use this paramString value to set the test mode. This is the same as using the **Test Mode** controls on the **DUT** tab.

The value in bold font is the default value.

paramString value	<String>
Test Mode$<String>	**Compliance** or User Defined

Example. SetGeneralParameter(clientId, device, devicesuite, "", "Test Mode $Compliance");

SetGeneralParameter(clientId, device, devicesuite, "", "Test Mode$User Defined");

Select fixture. Use this paramString value to set the fixture type. This is the same as using the **Fixture** controls on the **DUT** tab.

The value in bold font is the default value.

paramString value	<String>
Fixture$<String>	**USB-IF**

Example. SetGeneralParameter(clientId, device, devicesuite, "", "Fixture$USB-IF");

Select connector. Use this paramString value to set the connector type. This is the same as using the **Connector** controls on the **DUT** tab.

The value in bold font is the default value.

paramString value	<String>
Connector$<String>	■ **Type C** ■ Micro ■ Standard

Example. SetGeneralParameter(clientId, device, devicesuite, "", "Connector$Type C");

SetGeneralParameter(clientId, device, devicesuite, "", "Connector$Micro");

Select spread spectrum clocking. Use this paramString value to set the spread spectrum clocking. This is the same as using the **Spread Spectrum Clocking** control on the **DUT** tab.

The value in bold font is the default value.

paramString value	<String>
SSC On$<String>	**True** or False

Example. SetGeneralParameter(clientId, device, devicesuite, "", "SSC On$True");

Select test point. Use this paramString value to set the DUT test point. This is the same as using the **Test Point** controls on the DUT tab.

The value in bold font is the default value.

paramString value	<String>
Version$<String>	■ **Compliance (TP4) - Far End** ■ Tx Pins - Near End ■ Custom

Example. SetGeneralParameter(clientId, device, devicesuite, "", "Version $Compliance (TP4) - Far End");

SetGeneralParameter(clientId, device, devicesuite, "", "Version$Tx Pins - Near End");

SetGeneralParameter(clientId, device, devicesuite, "", "Version$Custom");

Select lane selection. Use this paramString value to set the lane selection type. This is the same as using the **Lane Selection** controls on the **DUT** tab.

The value in bold font is the default value.

paramString value	<String>
Flip Selection$<String>	■ **Lane 0** ■ Lane 1 ■ Both

Example. SetGeneralParameter(clientId, device, devicesuite, "", "Flip Selection $Lane 0");

SetGeneralParameter(clientId, device, devicesuite, "", "Flip Selection$Lane 1");

Select test method. Use this paramString value to set the test method. This is the same as using the **Test Method** controls on the **DUT** tab.

The value in bold font is the default value.

paramString value	<String>
Gen1 Test Method$<String>	■ DPOJET ■ **SIGTest(USB-IF)** ■ Both

Example. SetGeneralParameter(clientId, device, devicesuite, "", "Gen1 Test Method$DPOJET");

SetGeneralParameter(clientId, device, devicesuite, "", "Gen1 Test Method $SIGTest(USB-IF)");

SetGeneralParameter(clientId, device, devicesuite, "", "Gen1 Test Method $Both");

Select filter selection settings. Use this paramString value to set the filter setup parameters. This is the same as using the **Filter Setup** controls on the **DUT** tab.

The value in bold font is the default value.

paramString value	Description	<String>
Compliance (TP4) - Far End - Deembed Filter Option $<String>	Use to select Gen1 Deembed Filter option for test point Compliance (TP4) - Far End	**Included** or Excluded
Compliance (TP4) - Far End - Gen2 Deembed Filter Option $<String>	Use to select Gen2 Deembed Filter option for test point Compliance (TP4) - Far End	**Included** or Excluded
Compliance (TP4) - Far End - Embed Filter Option$<String>	Use to select Gen1 Embed Filter option for test point Compliance (TP4) - Far End	**Included** or Excluded
Compliance (TP4) - Far End - Gen2 Embed Filter Option $<String>	Use to select Gen2 Embed Filter option for test point Compliance (TP4) - Far End	**Included** or Excluded
Compliance (TP4) - Far End - CTLE Filter Option$<String>	Use to select Gen1 CTLE Filter option for test point Compliance (TP4) - Far End	**Included** or Excluded
Compliance (TP4) - Far End - Gen2 CTLE Filter Option $<String>	Use to select Gen2 CTLE Filter option for test point Compliance (TP4) - Far End	**Included** or Excluded
Gen2 Ctle Option$<String>	Use to select Gen2 CTLE option	Fixed or **Optimize**
Gen2 Ctle Index$<String>	Use to select Gen2 CTLE Index	0, 1, 2, 3, 4, 5, **6**
USB3.1 Gen1 - Long - Deembed Filter File Path $<String>	Use to select USB3.1 Gen1 - Long - Deembed Filter File Path	**Tx_Device_TF_8G.flt**
USB3.1 Gen2 - Long - Deembed Filter File Path $<String>	Use to select USB3.1 Gen2 - Long - Deembed Filter File Path	**SSP_De-embed_Tx_Device.flt**
USB3.1 Gen1 - Long - Embed Filter File Path$<String>	Use to select USB3.1 Gen1 - Long - Embed Filter File Path	**SSGen1_TxComp12p7dB_Embedding.flt**
USB3.1 Gen2 - Long - Embed Filter File Path$<String>	Use to select USB3.1 Gen2 - Long - Embed Filter File Path	**SSGen2_TxComp12p2dB_Embedding.flt**
USB3.1 Gen1 - Long - CTLE Filter File Path$<String>	Use to select USB3.1 Gen1 - Long - CTLE Filter File Path	**USB3CTLE.flt**

NOTE. *Before you select the filter file name, copy the filter files to the path C:\Program Files (x86)\Tektronix\TekExpress\TekExpress USB3.2 Tx\Setup Files\Filters.*

Example. SetGeneralParameter(clientId, device, devicesuite, "", "Compliance (TP4) - Far End - Deembed Filter Option$True");

SetGeneralParameter(clientId, device, devicesuite, "", "Compliance (TP4) - Far End - Gen2 Deembed Filter Option$True");

SetGeneralParameter(clientId, device, devicesuite, "", "Compliance (TP4) - Far End - Embed Filter Option$True");

Select auto recovery settings. Use this paramString value to set the auto recovery settings. This is the same as using the **Auto Recovery Settings** controls on the **DUT** tab.

The value in bold font is the default value.

paramString value	<String>
Auto Recovery Settings$<String>	Yes or **No**

Example. SetGeneralParameter(clientId, device, devicesuite, "", "Auto Recovery Settings$Yes");

Select radio friendly clocking. Use this paramString value to set the radio friendly clocking. This is the same as using the **Radio Friendly Clocking** control on the **DUT** tab.

The value in bold font is the default value.

paramString value	<String>
Radio Friendly Clocking$<String>	Included or **Excluded**

Example. SetGeneralParameter(clientId, device, devicesuite, "", "Radio Friendly Clocking$Yes");

Select low power mode. Use this paramString value to set the low power mode. This is the same as using the **Low Power Mode** control on the **DUT** tab.

The value in bold font is the default value.

paramString value	<String>
Low Power Mode$<String>	Included or **Excluded**

Example. SetGeneralParameter(clientId, device, devicesuite, "", "Low Power Mode$Yes");

Select signal validation. Use this paramString value to set the signal validation type. This is the same as using the **Signal Validation** controls on the **Acquisitions** tab.

The value in bold font is the default value.

paramString value	<String>
Pattern Validation$<String>	■ **Prompt me if Signal Check Fails** ■ Turn Off Signal Check

Example. SetGeneralParameter(clientId, device, devicesuite, "", "Pattern Validation$Turn Off Signal Check");

Select record length. Use this paramString value to set the record length. This is the same as using the **Record Length** controls on the **Configuration** tab.

The value in bold font is the default value.

Values	Description	<NR1> [1]
Record Length for CP0 CP1 CP7$<NR1>	Use to set the Record length for 5Gbps signal	5000000 to 30000000 **10M**
Record Length for CP9 CP10$<NR1>	Use to set the Record length for 10Gbps signal	5000000 to 30000000 **20M**

Example. SetGeneralParameter(clientId, device, devicesuite, "", "Record Length for CP0 CP1 CP7$15");

SetGeneralParameter(clientId, device, devicesuite, "", "Record Length for CP9 CP10$10000000");

[1] NR1 is unsigned integer value

Select LFPS settings. Use this paramString value to set the LFPS parameters. This is the same as using the **LFPS Settings** controls on the **Configuration** tab.

The value in bold font is the default value.

Values	Description	<NR1>[1]
LFPS Width Trigger Lower limit (ns)$<NR1>	Use to set the LFPS Width Trigger Lower limit	1 to 15 **10**
Gen2 LFPS Width Trigger Upper limit (ns)$<NR1>	Use to set the Gen2 LFPS Width Trigger Upper limit (ns)	15 to 100 **40**
LFPS Trigger level (mV) $<NR1>	Use to set the LFPS Trigger level (mV)	20 to 250 **140**
LFPS Mid Edge Ref Level (mV) $<NR1>	Use to set the LFPS Mid Edge Ref Level (mV)	100 to 1000 **150**
LFPS Hysteresis Level (mV) $<NR1>	Use to set the LFPS Hysteresis Level (mV)	10 to 300 **50**
Bandwidth for LFPS acquisition (GHz)$<NR1>	Use to set the bandwidth for LFPS acquisition (GHz)	5 to 12 **5**

Example. SetGeneralParameter(clientId, device, devicesuite, "", "LFPS Width Trigger Lower limit (ns)$15");

SetGeneralParameter(clientId, device, devicesuite, "", "Gen2 LFPS Width Trigger Upper limit (ns)$100");

Select report update mode settings. Use this paramString value to set the report update mode settings. This is the same as using the **Report Update Mode** controls on the **Reports** panel.

The value in bold font is the default value.

paramString value	<String>
Report Update Mode$<String>	■ **New** ■ Append ■ Replace

Example. SetGeneralParameter(clientId, device, devicesuite, "", "Report Update Mode$New");

SetGeneralParameter(clientId, device, devicesuite, "", "Report Update Mode $Append");

Select actions on test measurement failure. Use this paramString value to set the actions on test measurement failure. This is the same as using the **Actions on Test Measurement Failure** control on the **Preferences** tab.

The value in bold font is the default value.

paramString value	<String>
On Failure Stop and Notify$<String>	True or **False**

Example. SetGeneralParameter(clientId, device, devicesuite, "", “On Failure Stop and Notify$True");

Select reports contents to save. Use this paramString value to set the report contents to save. This is the same as using the **Contents To Save** controls on the **Reports** tab.

The value in bold font is the default value.

Value	Description	<String>
Include Pass/Fail Results Summary$<String>	Use to select / deselect Include Pass/Fail Results Summary	**True** or False
Include Detailed Results $<String>	Use to select / deselect Include detailed results	**True** or False
Include Plot Images$<String>	Use to select / deselect Include Plot Images	**True** or False
Include Setup Configuration $<String>	Use to select / deselect Include Setup Configuration	**True** or False
Include User Comment $<String>	Use to select / deselect Include User Comment	**True** or False

Example. SetGeneralParameter(clientId, device, devicesuite, "", "Include Pass/Fail Results Summary$False");

SetGeneralParameter(clientId, device, devicesuite, "", "Include Detailed Results $False");

Select report creation settings. Use this paramString value to set the report creation settings. This is the same as using the **Report Creation Settings** controls on the **Reports** panel.

The value in bold font is the default value.

paramString value	**<String>**
Report Path$<String>	■ Report path ■ **X:\USB3.2 Tx\Reports\USB.pdf**
Save As Type$<String>	■ **Web Archive (*.mht;*.mhtml)** ■ PDF(*.pdf;) ■ CSV(*.csv)
Auto increment report name if duplicate $<String>	**True** or False

Example. SetGeneralParameter(clientId, device, devicesuite, "", “Report Path$X:\USB3.2 Tx\Reports\USB.pdf");

SetGeneralParameter(clientId, device, devicesuite, "", “Save As Type$Web Archive (*.mht;*.mhtml)");

Select view report after generating. Use this paramString value to set to view report after generating. This is the same as using the **View report after generating** control on the **Reports** panel.

The value in bold font is the default value.

paramString value	**<String>**
View Report After Generating$<String>	**True** or False

Example. SetGeneralParameter(clientId, device, devicesuite, "", “View Report After Generating$False");

SCPI commands

About SCPI command

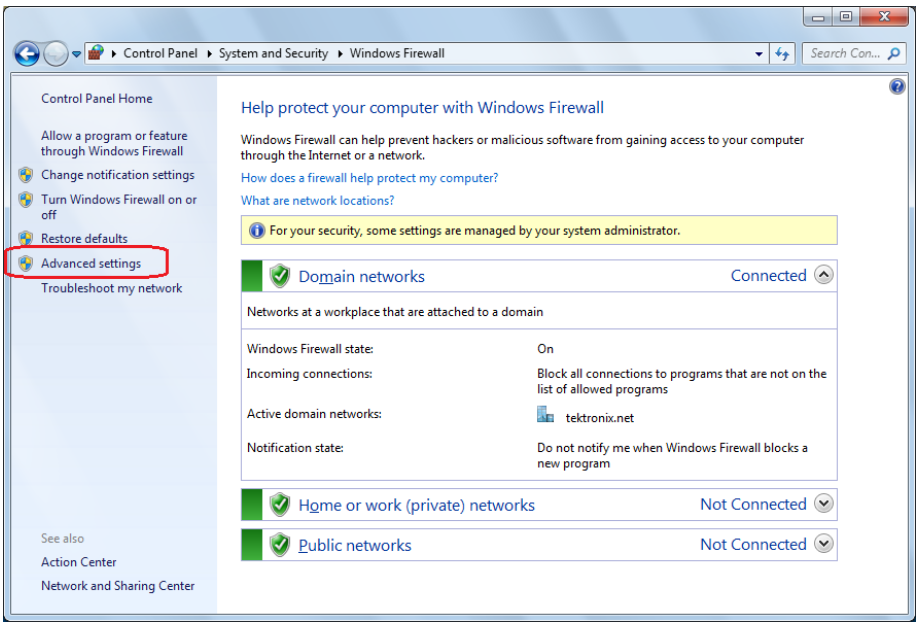
You can use Standard Commands for Programmable Instruments (SCPI) to communicate with the TekExpress application.

Socket configuration for SCPI commands

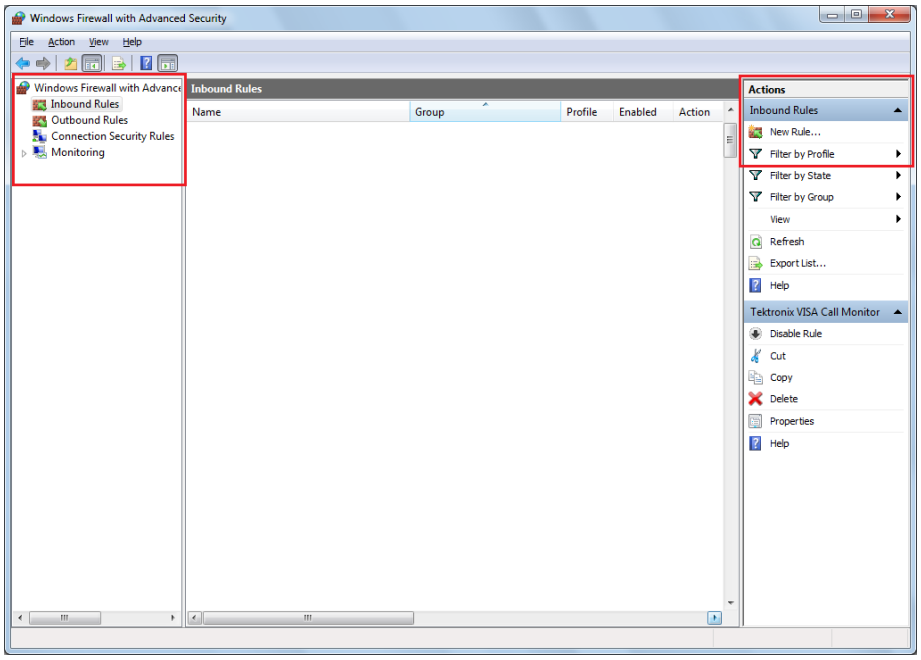
This section describes the steps for TCPIP socket configuration and TekVISA configuration to execute the SCPI commands.

TCPIP socket configuration

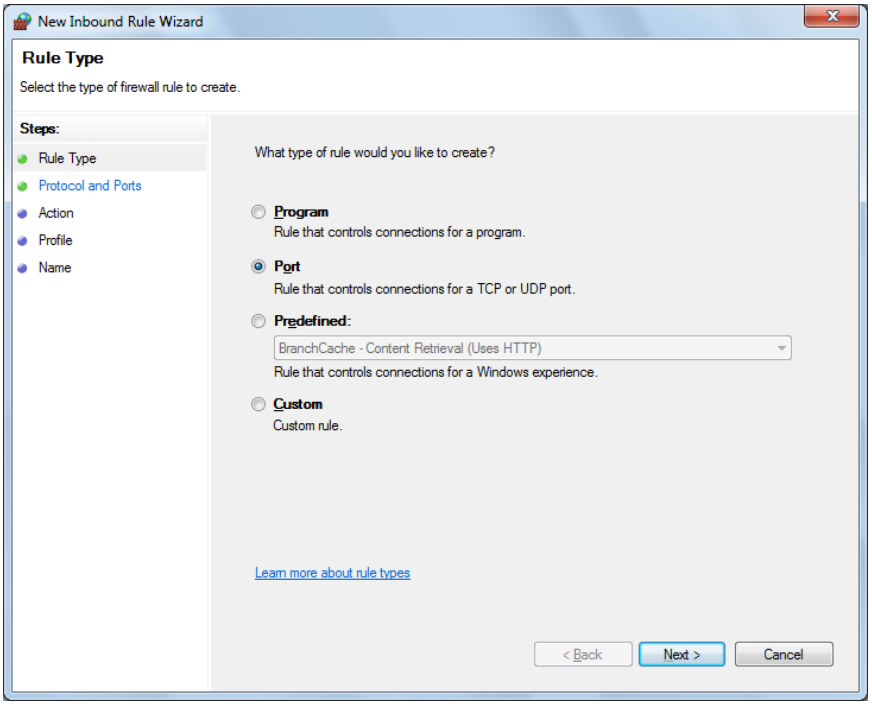
1. Click **Start** > **Control Panel** > **System and Security** > **Windows Firewall** > **Advanced settings**.



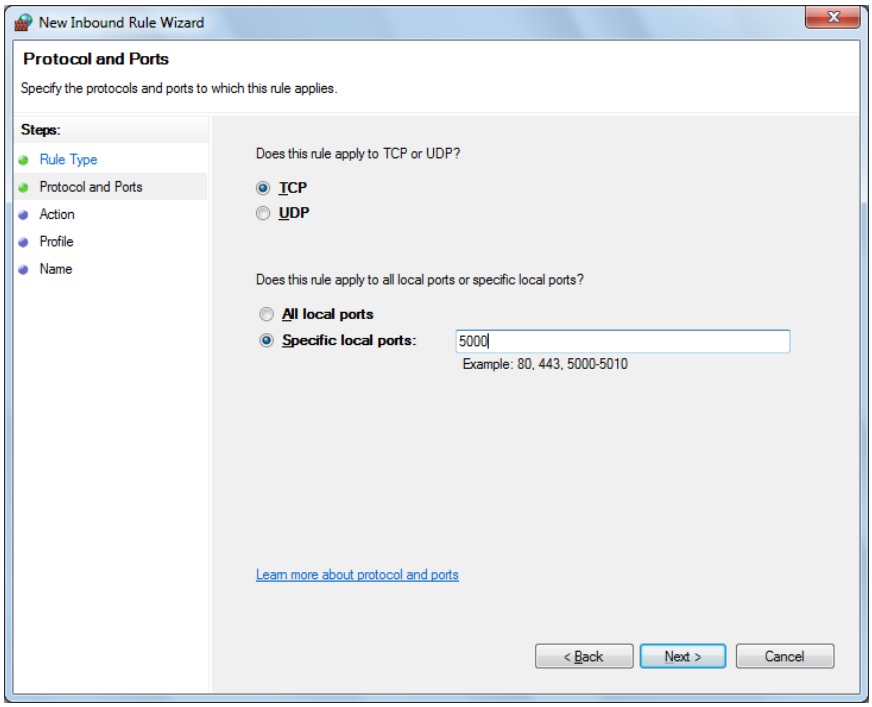
2. In Windows Firewall with Advanced Security menu, select **Windows Firewall with Advanced Security on Local Computer** > **Inbound Rules** and click New Rule…



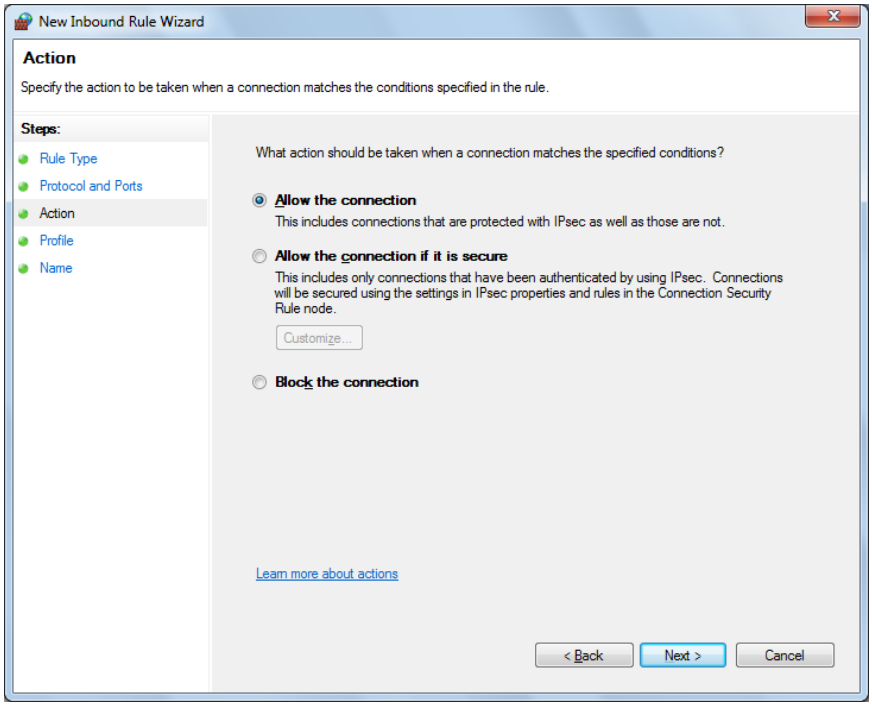
3. In New Inbound Rule Wizard menu
   a. Select **Port** and click **Next**.



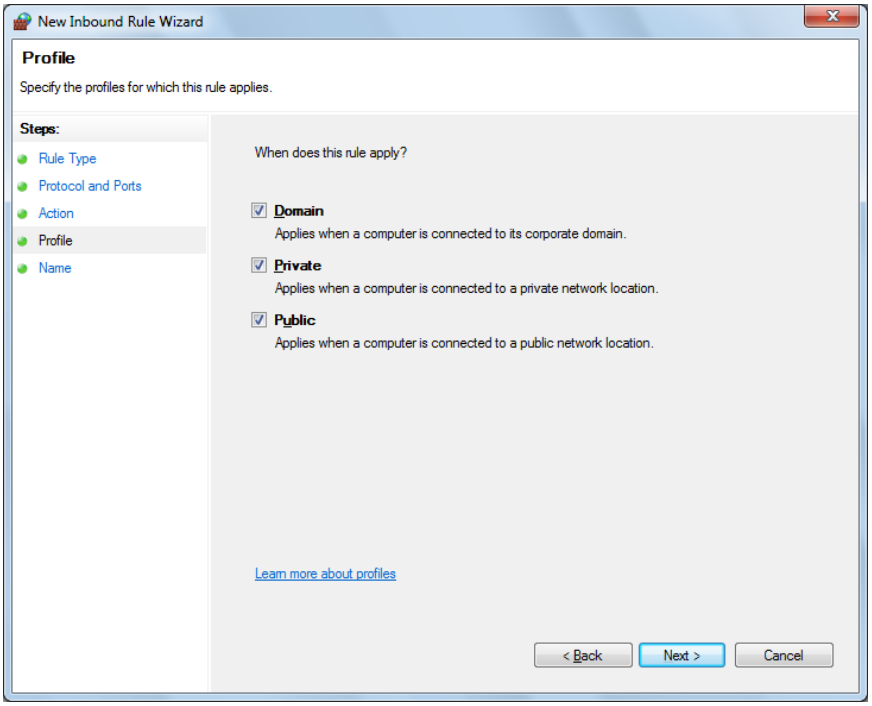
b. Select **TCP** as rule apply and enter 5000 for **Specific local ports** and click **Next**.



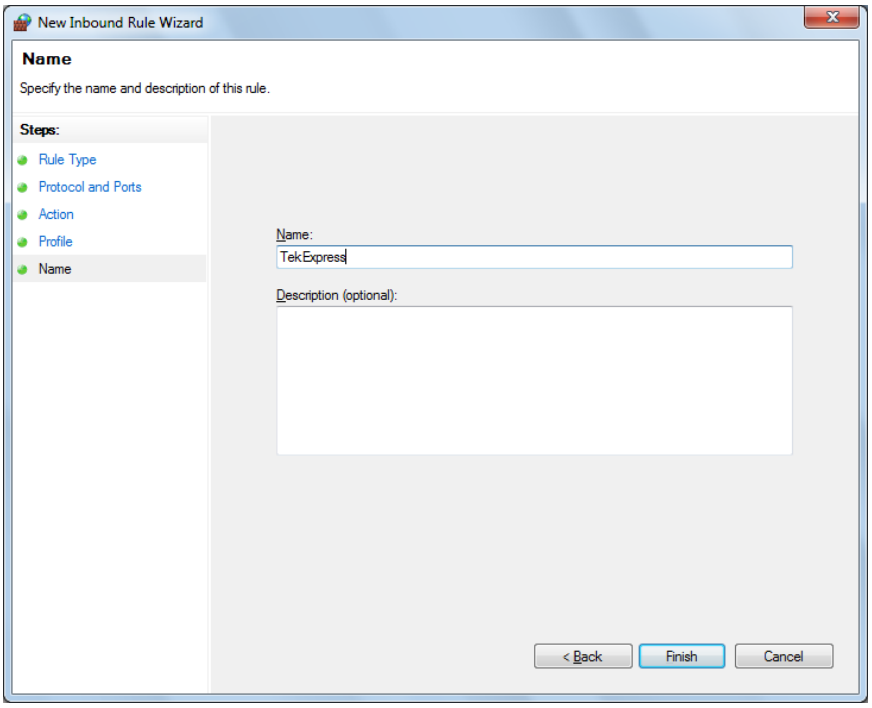
c. Select **Allow the connection** and click **Next**.



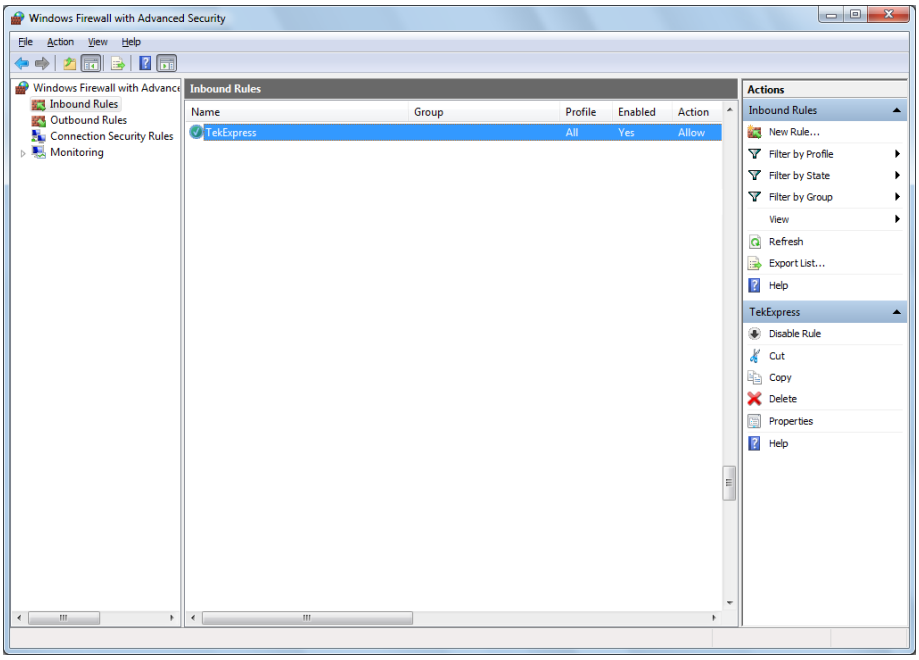
d. Select **Domain, Private, Public** and click **Next**.



e. Enter **Name**, Description (optional), and click **Finish**.

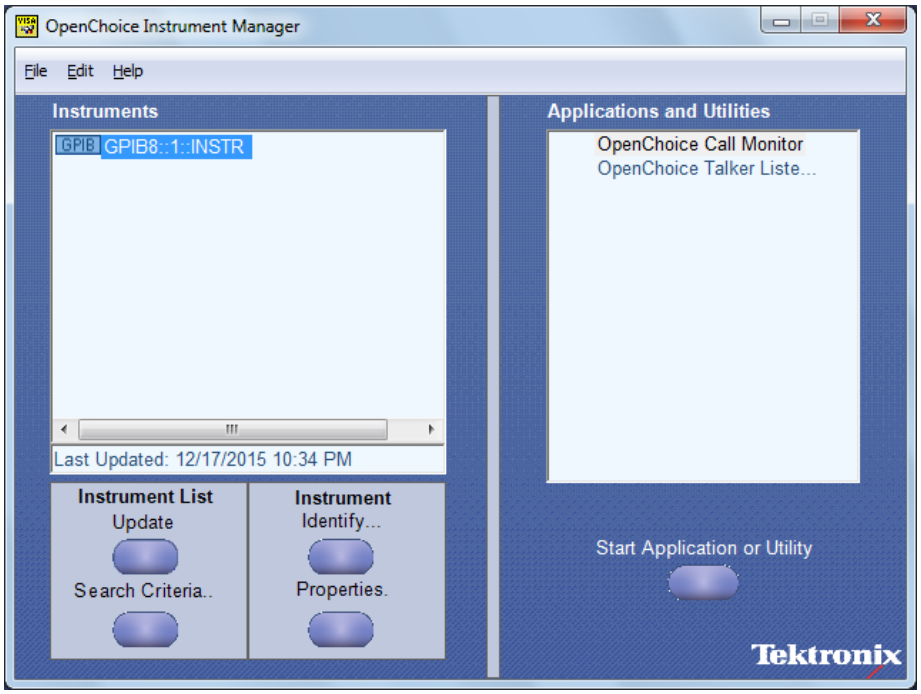


4. Check whether the Rule name is displayed in **Windows Firewall with Advanced Security menu** > **Inbound Rules**.



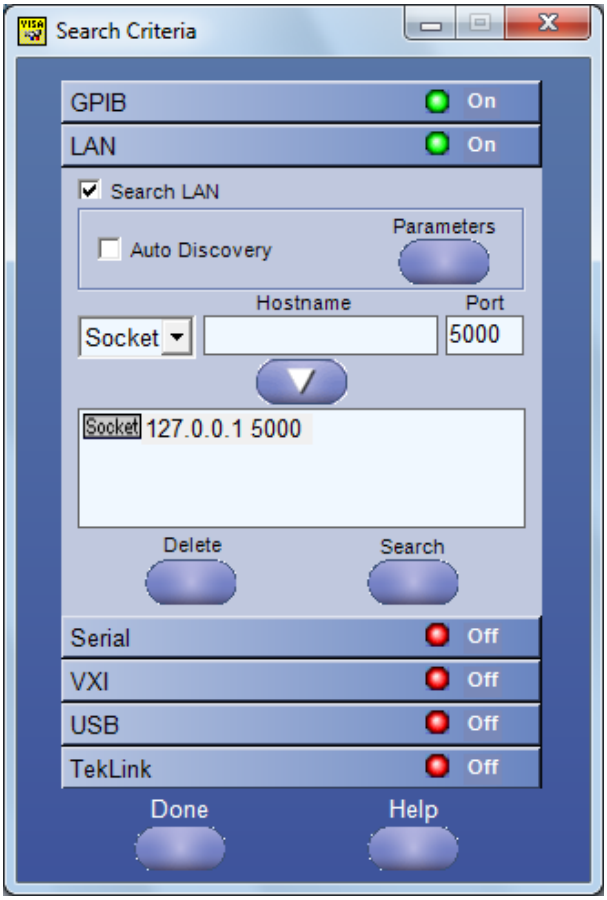
TekVISA configuration

1. Click **Start** > **All Programs** > **TekVISA** > **OpenChoice Instrument Manager**.

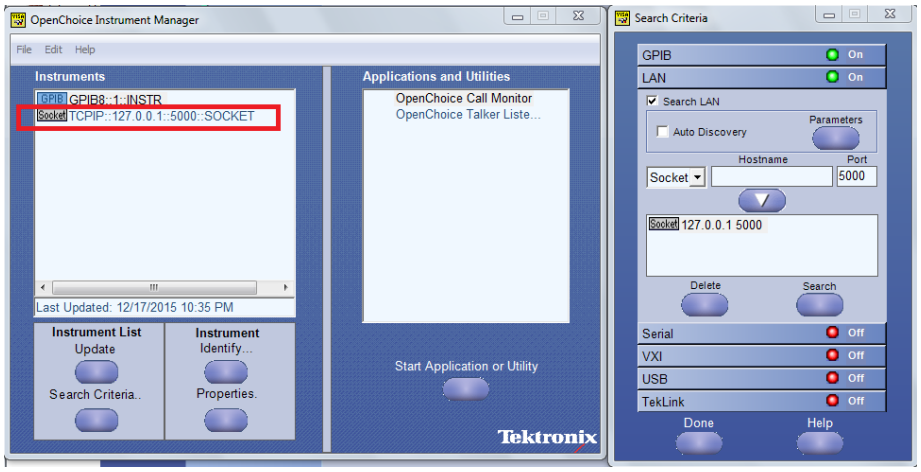


2. Click **Search Criteria**. In Search Criteria menu, click **LAN** to Turn-on. Select **Socket** from the drop-down list, enter the IP address of the TekExpress device in **Hostname** and type **Port** as 5000. Click  to configure the IP address with Port.

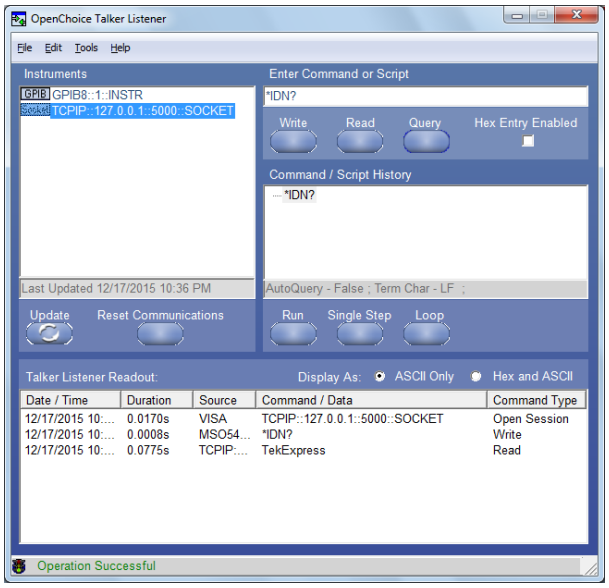
   Enter the Hostname as 127.0.0.1 if the TekVISA and TekExpress application are in the same system, else enter the IP address of the TekExpress application system.



3. Click **Search** to setup the TCPIP connection with the host. Check whether the TCPIP host name is displayed in **OpenChoice Instrument Manager** > **Instruments**.



4. Double-click **OpenChoice Talker Listener** and enter the Command *IDN? in command entry field and click **Query**. Check that the Operation is successful and Talker Listener Readout displays the Command / Data.



TEKEXP:*IDN?

This command queries the active TekExpress application name running on the oscilloscope.

Syntax TEKEXP:*IDN?\n

Inputs NA

Outputs Returns active TekExpress application name running on the oscilloscope.

TEKEXP:*OPC?

This command queries the execution status of the last executed command.

Syntax TEKEXP:*OPC?\n

Inputs NA

Outputs 0 - last command execution is not complete

1 - last command execution is complete

TEKEXP:ACQUIRE_MODE

This command sets the acquire mode as live or pre-recorded.

Syntax TEKEXP:ACQUIRE_MODE {LIVE | PRE-RECORDED}\n

Inputs {LIVE | PRE-RECORDED}

Outputs NA

TEKEXP:ACQUIRE_MODE?

This command queries the acquire mode type.

Syntax TEKEXP:ACQUIRE_MODE?\n

Inputs NA

Outputs {LIVE | PRE-RECORDED}

TEKEXP:EXPORT

This command returns all the bytes of data to the specified file.

Syntax	Outputs
TEKEXP:EXPORT REPORT\n	Returns the report file in bytes
TEKEXP:EXPORT WFM,"<FileName>"\n	Returns the specified waveform file in bytes
TEKEXP:EXPORT IMAGE,"<FileName>"\n	Returns the specified image file in bytes

Inputs FileName - Specifies the file name

TEKEXP:INFO?

This command queries the information about the file(s).

Syntax	Outputs
TEKEXP:INFO? REPORT\n	<ReportFileSize>,"<ReportFileName.mht>"
TEKEXP:INFO? WFM \n	<WfmFile1Size>,"<WfmFileName1.wfm>";<WfmFile2Size>,"<WfmFileName2.wfm>";...
TEKEXP:INFO? IMAGE\n	<Image1FileSize>,"<Image1FileName>";<Image2FileSize>,"<Image2FileName>" ;...

TEKEXP:INSTRUMENT

This command sets the value for the selected instrument type.

Syntax

TEKEXP:INSTRUMENT "<InstrumentType>",<Value>"\n

Inputs

InstrumentType

Value

TIP. *Check Command parameters list section for InstrumentType and Value parameters.*

Outputs

NA

TEKEXP:INSTRUMENT?

This command queries the instrument selected for the specified instrument type.

Syntax

TEKEXP:INSTRUMENT? "<InstrumentType>"\n

Inputs

InstrumentType

TIP. *Check Command parameters list section for InstrumentType parameters.*

Outputs

Returns the instrument selected for the specified instrument type

TEKEXP:LASTERROR?

This command queries the last error string occurred for the current TCP session. If there are no errors since startup, or since the last call to TEKEXP:LASTERROR?\n, this command returns an empty string.

Syntax TEKEXP:LASTERROR?\n

Inputs NA

Outputs <string>

TEKEXP:LIST?

This command queries the list of available device, suite, test, version or instrument.

Syntax	Outputs
TEKEXP:LIST? DEVICE\n	Returns the list of available device(s) as comma separated values.
TEKEXP:LIST? SUITE\n	Returns the list of available suite(s) as comma separated values.
TEKEXP:LIST? TEST\n	Returns the list of available test(s) as comma separated values.
TEKEXP:LIST? VERSION\n	Returns the list of available version(s) as comma separated values.
TEKEXP:LIST? INSTRUMENT,"<InstrumentType>"\n	Returns the list of available instruments' for the given Instrument type as comma separated values.

NOTE. *This command returns the list of items within double quotes ("). Iterate the receive procedure until the list ends with double quotes otherwise the next query commands won't work as expected.*

Inputs InstrumentType

TIP. *Check Command parameters list section for InstrumentType parameters.*

TEKEXP:POPUP

This command sets the response to the active popup shown in the application.

Syntax TEKEXP:POPUP “<PopupResponse>”\n

Inputs PopupResponse

Outputs NA

TEKEXP:POPUP?

This command queries the active popup information shown in the application.

Syntax TEKEXP:POPUP?\n

Inputs NA

Outputs Returns the active popup information in the application.

TEKEXP:REPORT

This command generates the report for the current session.

Syntax TEKEXP:REPORT GENERATE\n

Inputs GENERATE

Outputs NA

TEKEXP:REPORT?

This command queries the queried header field value in the report.

Syntax TEKEXP:REPORT? “<HeaderField>”\n

Inputs HeaderField - Specifies to return the measured value for the indicated test.

TIP. *Check **Report** for HeaderField parameters.*

Outputs Returns the queried header field value in the report

TEKEXP:RESULT?

This command queries the result available in report summary/details table.

Syntax	Outputs
TEKEXP:RESULT? "<TestName>"\n	Return Pass/Fail status of the test.
TEKEXP:RESULT? "<TestName>","<ColumnName>"\n	Returns all the row values of the specified column for the test.
TEKEXP:RESULT? "<TestName>","<ColumnName>",<RowNumber>\n	Returns the column value for the specified row number [1]

Inputs

TestName - Specifies the name of the test for which to obtain the test result value.

ColumnName - Specifies the column name for the measurement

RowNumber - Specifies the row number of the measurement

TIP. *Check **Results** panel for TestName, ColumnName, and RowNumber parameters.*

[1] Row number starts from zero.

TEKEXP:SELECT

This command selects the device, suite, version, or test.

Syntax

TEKEXP:SELECT <string1>,<string2>,<string4>\n

TEKEXP:SELECT TEST,<string3>,<string4>\n

Inputs

<string1> = {DEVICE | SUITE | VERSION}

<string2> = {DeviceName | SuiteName | VersionName}

<string3> = {“<TestName>"| ALL| REQUIRED }

<string4> = {TRUE | FALSE}

TIP. *Check Command parameters list section for DeviceName, SuiteName, VersionName, and TestName parameters.*

Outputs

NA

TEKEXP:SELECT?

This command queries the name of the selected device, suite, version, or test.

Syntax

TEKEXP:SELECT? {DEVICE | SUITE | TEST | VERSION}\n

Inputs

{DEVICE | SUITE | TEST | VERSION}

Outputs Returns the name of the selected device, suite, version, or test.

TEKEXP:SETUP

This command sets the value of the current setup.

Syntax	Outputs
TEKEXP:SETUP DEFAULT\n	Restore to default Setup
TEKEXP:SETUP OPEN,"<SessionName>"\n	Open the session
TEKEXP:SETUP SAVE\n	Saves the already existing modified session
TEKEXP:SETUP SAVE,"<SessionName>"\n	Save the session

Inputs SessionName - The name of the session

TEKEXP:STATE

This command sets the execution state of the application.

Syntax TEKEXP:STATE {RUN | STOP | PAUSE | RESUME}\n

Inputs {RUN | STOP | PAUSE | RESUME}

Outputs NA

TEKEXP:STATE?

This command queries the current setup state.

Syntax	Outputs
TEKEXP:STATE?	RUNNING | PAUSED | WAIT | ERROR | READY
TEKEXP:STATE? SETUP	SAVED | NOT_SAVED

TEKEXP:VALUE

This command sets the value of parameters of type General, Acquire, Analyze, or DUTID.

Syntax

TEKEXP:VALUE GENERAL,"<ParameterName>",“<Value>”\n

TEKEXP:VALUE ACQUIRE,“<TestName>","<AcquireType>", "<ParameterName>",“<Value>”\n

TEKEXP:VALUE ANALYZE,“<TestName>","<ParameterName>".“<Value>” \n

TEKEXP:VALUE DUTID,“<Value>”\n

TEKEXP:VALUE VERBOSE,{TRUE | FALSE}\n

TEKEXP:VALUE WFMFILE,<Test_Name>,<Aquire_Type>,<FilesName1$FileName2>\n

Inputs

ParameterName - Specifies the parameter name

TestName - Specifies the test name

AcquireType - Specifies the acquire type

Value - Specifes the value to set

FilesName1$FileName2 - Specifies the waveform file name

TRUE - Pop-ups are enabled

FALSE - Pop-ups are disabled

TIP. *Check Command parameters list section for ParameterName, AcquireType, and Value parameters.*

Outputs NA

TEKEXP:VALUE?

This command queries the value of the parameter for type General, Acquire, Analyze, or DUTID.

Syntax	Outputs
TEKEXP:VALUE? GENERAL,"<ParameterName>"\n	Returns the value of Parameter for type GENERAL
TEKEXP:VALUE? ACQUIRE,“<TestName>", "<AcquireType>","<ParameterName>"\n	Returns the value of Parameter for type ACQUIRE
TEKEXP:VALUE? ANALYZE, “<TestName>","<ParameterName>"\n	Returns the value of Parameter for type ANALYZE
TEKEXP:VALUE? DUTID\n	Returns the DUTID value
TEKEXP:VALUE? WFMFILE,<Test_Name>,<Aquire_Type>\n	Returns the waveform file name
TEKEXP:VALUE? VERBOSE	Returns the verbose mode type

Inputs ParameterName - Specifies the parameter name

TestName - Specifies the test name

AcquireType - Specifies the acquire type

TRUE - Pop-ups are enabled

FALSE - Pop-ups are disabled

TIP. *Check Command parameters list section for ParameterName and AcquireType parameters.*

Outputs Returns the value of Parameter for type GENERAL | ACQUIRE | ANALYZE | DUTID.

Command parameters

This section provides the parameters list for the SCPI commands.

Parameters	Description
InstrumentType	Specifies the instrument type. Valid value is Real Time Scope.

Value	Specifies the value parameters.
	■ For InstrumentType, valid values are:
	  ■ Do not use
	  ■ GPIB8::1::INSTR
	■ For DUTID, valid value is Comment.

DeviceName	Specifies the device name. Valid values are: ■ OIF-PAM4 CEI-VSR ■ OIF-PAM4 CEI-MR ■ OIF-PAM4 CEI-LR ■ IEEE-PAM4 AUI4 ■ IEEE-PAM4 CR4 ■ IEEE-PAM4 KR4
SuiteName	Specifies the suite name. Valid values are: ■ TP0a, TP1a, TP4 for OIF-PAM4 CEI-VSR ■ Testpoint-T for OIF-PAM4 CEI-MR and OIF-PAM4 CEI-LR ■ TP0a, TP1a, TP4 for IEEE-PAM4 AUI4 ■ Testpoint-TP2 for IEEE-PAM4 CR4 ■ Testpoint-TP0a for IEEE-PAM4 KR4

Parameters	Description
VersionName	Specifies the version name. Valid values are ■ 400G-TXE,Section-16.B.1.1,Table 16-10 ■ 400G-TXE,Section-16.3.2,Table 16-1 ■ 400G-TXE,Section-16.3.3,Table 16-4 ■ 400G-TXE,Section-17.3.1,Table 17-2 ■ 400G-TXE,Section-21.3,Table 21-2 ■ 400G-TXE,Annex 120D.3.1,Table 120D-1 ■ 400G-TXE,Annex 120E.3.1, Table 120E-1 ■ 400G-TXE,Annex 120E.3.2, Table 120E-3 ■ 400G-TXE,Section 136.9.3, Table 136-11 ■ 400G-TXE,Section 137.9.2
TestName for OIF-PAM4 CEI-VSR	■ DC Common Mode Output Voltage (TP0a, TP1a, TP4) ■ Diff Peak to Peak Output Voltage Tx Enabled (TP0a, TP1a, TP4) ■ AC Common Mode Output Voltage (TP0a, TP1a, TP4) ■ Transition Time (TP0a, TP1a, TP4) ■ Signal To Noise And Distortion Ratio (TP0a, TP1a, TP4) ■ Even Odd Jitter (TP0a) ■ Uncorrelated Bounded High Probability Jitter (TP0a) ■ Uncorrelated Unbounded Gaussian Jitter (TP0a) ■ Eye Width (TP1a) ■ Eye Height (TP1a) ■ Eye Linearity (TP1a) ■ Eye Symmetry Mask Width (TP4) ■ Near End Eye Width (TP4) ■ Near End Eye Height (TP4) ■ Near End Eye Linearity (TP4) ■ Near End Eye Symmetry Mask Width (TP4) ■ Far End Eye Width (TP4) ■ Far End Eye Height (TP4) ■ Far End Eye Symmetry Mask Width (TP4)

Parameters	Description
TestName for OIF-PAM4 CEI-MR	■ DC Common Mode Output Voltage ■ Diff Peak to Peak Output Voltage Tx Enabled ■ AC Common Mode Output Voltage ■ Single Ended Output Voltage ■ Signal To Noise And Distortion Ratio ■ Level Separation Mismatch Ratio ■ Linear Fit Pulse Peak ■ Steady State Voltage ■ Step size for coefficient C(-1) ■ Step size for coefficient C(0) ■ Step size for coefficient C(1) ■ Coefficient Range C(-1) ■ Coefficient Range C(0) ■ Coefficient Range C(1) ■ Even Odd Jitter ■ Jitter RMS ■ Uncorrelated J4 Jitter

Parameters	Description
TestName for OIF-PAM4 CEI-LR	■ DC Common Mode Output Voltage ■ Diff Peak to Peak Output Voltage Tx Enabled ■ AC Common Mode Output Voltage ■ Single Ended Output Voltage ■ Signal To Noise And Distortion Ratio ■ Level Separation Mismatch Ratio ■ Linear Fit Pulse Peak ■ Steady State Voltage ■ Step size for coefficient C(-2) ■ Step size for coefficient C(-1) ■ Step size for coefficient C(0) ■ Step size for coefficient C(1) ■ Coefficient Range C(-2) ■ Coefficient Range C(-1) ■ Coefficient Range C(0) ■ Coefficient Range C(1) ■ Even Odd Jitter ■ Jitter RMS ■ Uncorrelated J4 Jitter

Parameters	Description
TestName for IEEE-PAM4 AUI4	■ DC Common Mode Output Voltage (TP0a, TP1a, TP4) ■ Diff Peak to Peak Output Voltage Tx Disabled (TP0a, TP1a) ■ Diff Peak to Peak Output Voltage Tx Enabled (TP0a, TP1a, TP4) ■ AC Common Mode Output Voltage (TP0a, TP1a, TP4) ■ Signaling Rate (TP0a, TP1a, TP4) ■ Signal To Noise And Distortion Ratio (TP0a) ■ Level Separation Mismatch Ratio (TP0a) ■ Linear Fit Pulse Peak (TP0a) ■ Steady State Voltage (TP0a) ■ Pre Cursor Equalization (TP0a) ■ Post Cursor Equalization (TP0a) ■ Transmitter output residual ISI (TP0a) ■ Even Odd Jitter (TP0a) ■ Uncorrelated J4 Jitter (TP0a) ■ Jitter RMS (TP0a) ■ Single Ended Output Voltage (TP1a) ■ Transition Time (TP1a, TP4) ■ Eye Height (TP1a) ■ Eye Symmetry Mask Width (TP1a) ■ Near End Eye Height (TP4) ■ Near End Eye Symmetry Mask Width (TP4) ■ Far End Eye Height (TP4) ■ Far End Eye Symmetry Mask Width (TP4) ■ Far End Precursor ISI Ratio (TP4)

Parameters	Description
TestName for IEEE-PAM4 CR4 and KR4	■ DC Common Mode Output Voltage ■ Diff Peak to Peak Output Voltage Tx Disabled ■ Diff Peak to Peak Output Voltage Tx Enabled ■ AC Common Mode Output Voltage ■ Signaling Rate ■ Signal To Noise And Distortion Ratio ■ Level Separation Mismatch Ratio ■ Linear Fit Pulse Peak ■ Steady State Voltage ■ Coefficient Range ■ OUT_OF_SYNC ■ NEW_IC PRESET1 ■ NEW_IC PRESET2 ■ NEW_IC PRESET3 ■ Step size for coefficient C(-2) ■ Step size for coefficient C(-1) ■ Step size for coefficient C(0) ■ Step size for coefficient C(1) ■ Even Odd Jitter ■ Jitter RMS ■ Uncorrelated J3 Jitter

ParameterName and Value for General, Acquire and Analyze

Specifies the ParameterName and Value for General, Acquire and Analyze. The configuration parameters available are not same for measurements.

Table 17: ParameterName and Value for General

ParameterName	Value
DUTID Comment	User comment
MODE	■ COMPLIANCE ■ USER-DEFINED

ParameterName	Value
Report Update Mode	■ New ■ Append ■ Replace ■ ReplaceAny
Replace Runsession Path	Session file path. Example: X:\400G-TXE\Session1\DUT001\20170421_121534
Auto increment report name if duplicate	"True" or "False"
Include Pass/Fail Results Summary	"True" or "False"
Include Detailed Results	"True" or "False"
Include Plot Images	"True" or "False"
Include Setup Configuration	"True" or "False"
Include User Comments	"True" or "False"
Report Path	File path Example: TEKEXP:VALUE GENERAL,"Report Path", "X:\400G-TXE\Reports\"
Save As Type	■ Web Archive (*.mht;*.mhtml) ■ PDF (*.pdf;) ■ CSV (*.csv;)
View Report After Generating	"True" or "False"
Report Group Mode	■ Test Name ■ Test Result
Create report at the end	"True" or "False"
Run Test More than Once	"True" or "False"
Number of Runs	1 to 50
On Failure Stop and Notify	"True" or "False"
Timer Warning Info Message Popup	"True" or "False"
Timer Warning Info Message Popup Duration	1 to 300
Timer Error Message Popup	"True" or "False"
Timer Error Message Popup Duration	1 to 300

ParameterName	Value
Lane0 Connected to:Lane0+: Single Ended	Valid values are: ■ CH1 ■ CH2 ■ CH3 ■ CH4
DUT Type	Valid values are: ■ "56G" ■ "112G"
Data Rate (GBd) for OIF-PAM4	Valid values are: ■ For "56G", limit is 18 to 29 ■ For "112G", limit is 36 to 58
Samples per Symbol (M)	32 to 200
Linear pulse length (Np)	Valid values are: ■ For OIF-PAM4: (5 to 100) ■ For IEEE-PAM4: (5 to 200)
Linear pulse delay (Dp)	Valid values are: ■ For OIF-PAM4: (2 to Np-2) ■ For IEEE-PAM4: (2 to Np-2)
NearEnd Mask Width	0.1 to 0.5
FarEnd Mask Width	0.1 to 0.5
Bandwidth	■ "Full BW" ■ "50GHz"
Scope Noise	0 to 20
Target BER (1e-)	4 to 6
Mask Width	0.1 to 0.5

ParameterName	Value
CTLE FilterFile	■ ALL(1-9dB) ■ 0 dB ■ 1 dB ■ 1.5 dB ■ 2 dB ■ 2.5 dB ■ 3 dB ■ 3.5 dB ■ 4 dB ■ 4.5 dB ■ 5 dB ■ 5.5 dB ■ 6 dB ■ 6.5 dB ■ 7 dB ■ 7.5 dB ■ 8 dB ■ 9 dB ■ Custom ■ BestCTLE

ParameterName	Value
Near End CTLE FilterFile	For OIF, valid values are: ■ ALL(1-2dB) ■ 0 dB ■ 1 dB ■ 1.5 dB ■ 2 dB ■ Custom ■ BestCTLE For IEEE, valid values are: ■ ALL(1-3dB) ■ 0 dB ■ 1 dB ■ 1.5 dB ■ 2 dB ■ 2.5 dB ■ 3 dB ■ Custom ■ BestCTLE

ParameterName	Value
Far End CTLE FilterFile	■ ALL(1-9dB) ■ 0 dB ■ 1 dB ■ 1.5 dB ■ 2 dB ■ 2.5 dB ■ 3 dB ■ 3.5 dB ■ 4 dB ■ 4.5 dB ■ 5 dB ■ 5.5 dB ■ 6 dB ■ 6.5 dB ■ 7 dB ■ 7.5 dB ■ 8 dB ■ 9 dB ■ Custom ■ BestCTLE
Apply Filter	"True" or "False"
Data Positive De-Embedding filter	Filter file path Example: TEKEXP:VALUE GENERAL,"De-Embedding filter","C:\"
Data Negative De-Embedding filter	Filter file path Example: TEKEXP:VALUE GENERAL,"De-Embedding filter","C:\"
Crosstalk source	"True" or "False"
Phase Inverted Filter For Data-	"True" or "False"

Examples

This section provides the examples for the SCPI commands.

Example	Description
TEKEXP:*IDN?\n	It returns the active TekExpress application name running on the oscilloscope.
TEKEXP:*OPC?\n	It returns the last command execution status.
TEKEXP:ACQUIRE_MODE PRE-RECORDED\n	It sets the acquire mode as pre-recorded.
TEKEXP:ACQUIRE_MODE?\n	It returns LIVE when acquire mode is set to live.
TEKEXP:EXPORT REPORT\n	It returns the report file in bytes. This can be written into another file for further analysis.
TEKEXP:EXPORT IMAGE,"ImageA.png"\n	It returns the image file in bytes. This can be written into another file for further analysis.
TEKEXP:EXPORT WFM,"WaveformA.wfm"\n	It returns the waveform file in bytes. This can be written into another file for further analysis.
TEKEXP:INFO? REPORT\n	It returns “100,”ReportFileName.mht”", when 100 is the filesize in bytes for the filename ReportFileName.
TEKEXP:INFO? WFM\n	It returns “100,“WfmFileName1.wfm”";“200,“WfmFileName2.wfm”" when 100 is the filesize in bytes for the filename WfmFileName1.wfm and 200 is the filesize in bytes for the filename WfmFileName2.wfm.
TEKEXP:INFO? IMAGE	It returns the image file name.
TEKEXP:INSTRUMENT "Real Time Scope",DPO77002SX (GPIB8::1::INSTR)\n	It sets the instrument value as DPO77002SX (GPIB8::1::INSTR) for the selected instrument type Real Time Scope.
TEKEXP:INSTRUMENT? “Real Time Scope”\n	It returns "IDPO77002SX (GPIB8::1::INSTR), when DPO77002SX (GPIB8::1::INSTR)" is the selected instrument for the instrument type Real Time Scope.
TEKEXP:LASTERROR?\n	It returns ERROR: INSTRUMENT_NOT_FOUND, when no instrument is found.
TEKEXP:LIST? DEVICE\n	It returns "TX-Device,RX-Device" when TX-Device, RX-Device are the available device.
TEKEXP:LIST? INSTRUMENT,"Real Time Scope"\n	It returns "DPO77002SX (GPIB8::1::INSTR),MSO73304DX (TCPIP::134.64.248.91::INSTR)" when DPO72504D (GPIB8::1::INSTR), MSO73304DX (TCPIP:: 134.64.248.91::INSTR) are the list of available instruments.
TEKEXP:MODE COMPLIANCE\n	It sets the execution mode as compliance.
TEKEXP:MODE?\n	It returns COMPLIANCE when the execution mode is compliance.
TEKEXP:POPUP “OK”\n	It sets OK as the response to active popup in the application.
TEKEXP:POPUP?\n	It returns "OK", when OK is the active popup information shown in the application.
TEKEXP:REPORT GENERATE\n	It generates report for the current session.
TEKEXP:REPORT? “Scope Information”\n	It returns "DPO73304SX" when DPO73304SX is the scope model.
TEKEXP:REPORT? “DUT ID”\n	It returns "DUT001" when DNI_DUT001 is the DUT ID.

Example	Description
TEKEXP:RESULT? "Period using SCOPE (Acquire-Analyze Combined)"\n	It returns Pass when the test result is Pass.
TEKEXP:RESULT? "Period using SCOPE (Acquire-Analyze Combined)","Margin",1\n	It returns "L:-50.000ps H:2000.000ps" when L:-50.000ps H: 2000.000ps is the value.
TEKEXP:SELECT DEVICE, TX_Device, TRUE\n	It selects TX_Device
TEKEXP:SELECT? DEVICE\n	It returns "TX-Device" when TX-Device is the selected device type.
TEKEXP:SETUP DEFAULT\n	It restores the application to default setup.
TEKEXP:STATE STOP\n	It stops the test execution.
TEKEXP:STATE?\n	It returns as READY when the application is ready to run next measurement.
TEKEXP:STATE? SETUP\n	It returns as NOT_SAVED when the current setup is not saved.

Reference

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 is 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:*

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

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

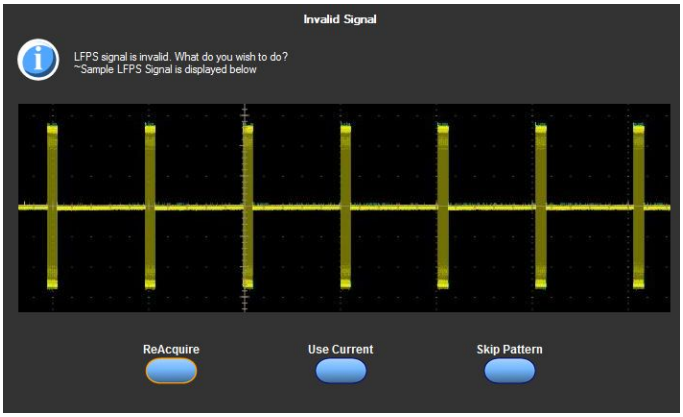
If the server is not found, the application displays " Server not found-Disconnect!".

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

Signal validation

LFPS pattern type validation

When the Pattern type validation is set to Yes, during the acquisition of LFPS pattern, a signal validation occurs. If the pattern is valid, the measurement continues normally. If the pattern is not valid, the application opens a message dialog box:



NOTE. *If Pattern type validation is selected as "No", then the measurement continues with the acquired waveform.*

NOTE. *Signal validation is not done for the SIGTest test method.*

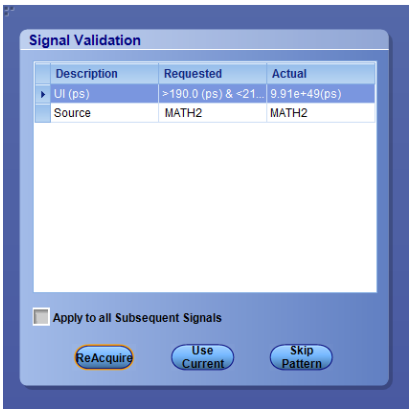
■ Click **Reacquire** to start the acquisition again.

■ Click **Use Current** to continue with the currently acquired waveform.

■ Click **Skip Pattern** to skip all LFPS tests. The rest of the selected measurements continue.

CPx pattern type validation

When the Pattern type validation is set to Yes, the application validates the CPx pattern (where x can be 0,1,7,9 or 10) during the acquisition. If the pattern is valid, the measurement continues normally. If the pattern is invalid, the following pop up displays.



NOTE. *If Pattern type validation is selected as "No", then the measurement continues with the acquired waveform.*

■ Click **Reacquire** to start the acquisition again.

■ Click **Use Current** to continue with the currently acquired waveform.

■ Click **Skip Pattern** to skip all CPx tests. The rest of the selected measurements continue.

Compliance pattern toggle mechanisms

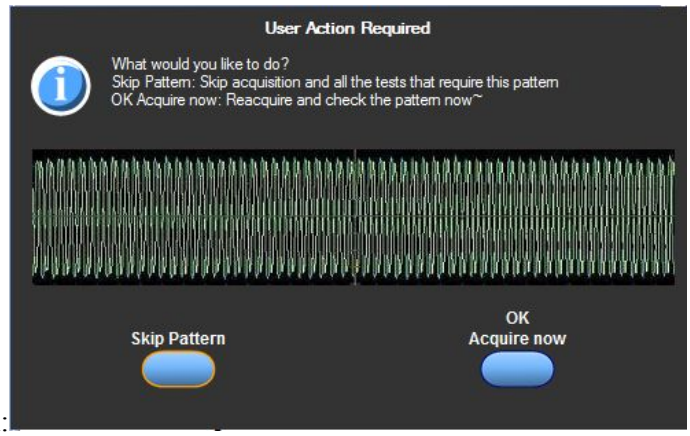
Oscilloscope-based toggle

To use the oscilloscope based toggle, do the following:

NOTE. *Oscilloscope based toggle is not guaranteed to work for all DUTs.*

1. In the Configuration panel, for the parameter **Toggle using**, select an oscilloscope (For example DPO72004 (TCPIP::192.158.96.152::INSTR)).
2. Connect the AUX OUT from the oscilloscope to the USB 3.0 Device Fixture 2 RX+ and connect a USB cable from USB 3.0 Device Fixture 2 to Device fixture 1.
3. Click the **Run** button. If the CP1 measurements are selected, then when the CP1 pattern is being acquired, a pop up displays to prompt you to make the necessary connections. Select to either skip the pattern or make a new acquisition after the DUT is transmitting CP1.

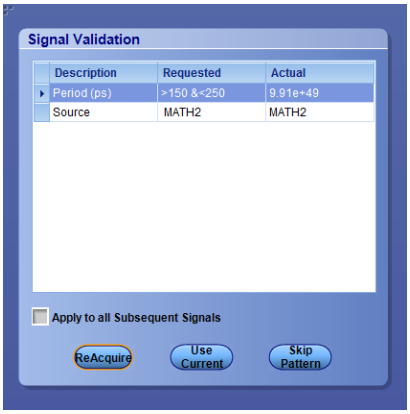
For CP1 signal:



For CP7 signal:

4. If you click **OK (Acquire Now)**, the application takes a new acquisition. If Pattern Type validation is set to Yes, a Pattern Type validation is run on the acquired signal to check if it is a CP1 signal. If it is a CP1 signal, the measurements continue normally. If not, the application shows the following dialog box.

NOTE. *If Pattern type validation is set to No, then the measurement continues with the acquired waveform.*



5. Choose how to continue:

   ■ Select **ReAcquire** to start the acquisition again.

   ■ Select **Use Current** to continue measurements using this acquired waveform.

   ■ Select **Skip Pattern** to skip all CP1 tests. The rest of the selected measurements are taken. If CP1 is skipped and CP0 is aquired, TJ and RJ are computed on CP0 for informational purposes.

See also. *AWG-based toggle*

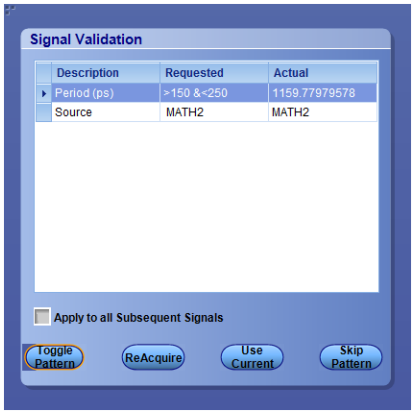
AFG-based toggle

Manual toggle

AWG-based toggle

To use the arbitrary waveform generator (AWG) based toggle method, do the following:

1. In the configuration panel, for the parameter **Toggle using**, select an AWG. For example: GPIB0::3::INSTR.
2. Connect the interleave (analog and analog) output of Ch1 of the AWG to the USB 3.0 Device Fixture 2 (RX+ and RX-) and connect a USB cable from the USB 3.0 Device Fixture 2 to USB 3.0 Device fixture 1.
3. Click the **Run** button. If the CP1 measurements are selected, then when the CP1 pattern is being acquired, a command is sent to the AWG to send a trigger to toggle the DUT from CP0 to CP1. Next, the waveform is acquired. If Pattern type validation is set to Yes, then the validation occurs. If the pattern is valid, the measurement continues normally. If the pattern is not valid, the application opens a dialog box similar to the following image:



NOTE. *If Pattern type validation is set to No, then the measurement continues with the acquired waveform.*

4. Choose how to continue:
   - Click **Toggle Pattern** to reinitiate the toggle sequence to toggle the DUT. (The pop up remains displayed during this toggle process.) You can visually verify whether the acquired pattern is correct. If not, keep clicking the Toggle Pattern button until the correct pattern is acquired.
   - Click **Reacquire** to start the acquisition again.
   - Click **Use Current** to continue with the currently acquired waveform.
   - Click **Skip Pattern** to skip the current CP tests. The rest of the selected measurements continue.

See also. *Oscilloscope-based toggle*

AFG-based toggle

Manual toggle

AFG-based toggle

To use the arbitrary function generator (AFG) based toggle, follow this procedure.

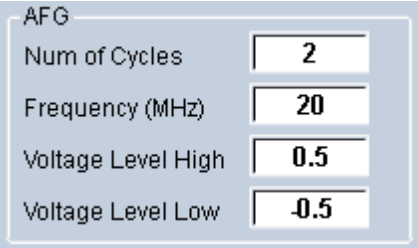
1. In the configuration panel, select an AFG instrument for the parameter **Toggle using**. For example: GPIB0::5::INSTR.
2. Connect Ch1 of the AFG to the Device fixture 2 (RX+).
3. Connect a 3 meter USB cable from Device fixture 2 to Device fixture 1.
4. Click the **Run** button. If the CP1 measurements are selected, a command is sent to AFG, when the CP1 pattern is being acquired, to toggle the DUT from CP0 to CP1. Next, the pattern is acquired. If Pattern type validation is set to Yes, then the validation occurs. If the pattern is valid, the measurement continues normally. If the pattern is not valid, the application opens a dialog box similar to the following image:

NOTE. *If Pattern type validation is set to No, then the measurement continues with the acquired waveform.*

---

5. Choose how to continue:
   - Click **Toggle Pattern** to reinitiate the toggle sequence to toggle the DUT. (The pop up remains displayed during this toggle process.) You can visually verify whether the acquired pattern is correct. If not, keep clicking the Toggle Pattern button until the correct pattern is acquired.
   - Click **Reacquire** to start the acquisition again.
   - Click **Use Current** to continue with the currently acquired waveform.
   - Click **Skip Pattern** to skip the current CP tests. The rest of the selected measurements continue.

User-Configurable AFG parameters.



You can configure the following parameters in the Configuration panel before the start of Test Execution when AFG is set as the toggle tool:

- **Num of Cycles**: Number of cycles per second. The range is from 1 to 5. The default value is 2.
- **Frequency (MHz)** : The range is from 10 MHz to 100 MHz. The default value is 20 MHz.
- **Voltage Level High**: The range is from –5 V to 5 V. The default value is 0.5 V.
- **Voltage Level Low**: The range is from –5 V to 5 V. The default value is –0.5 V.

See also. *Oscilloscope-based toggle*

AWG-based toggle

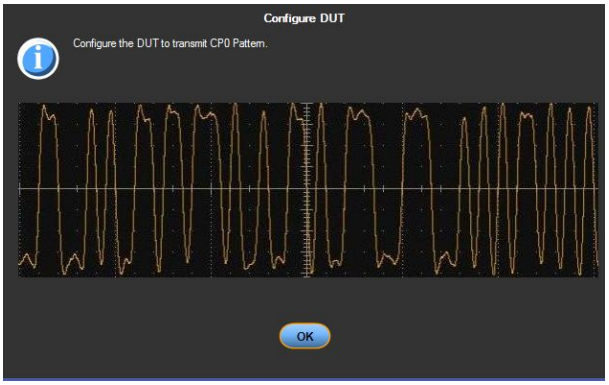
Manual toggle

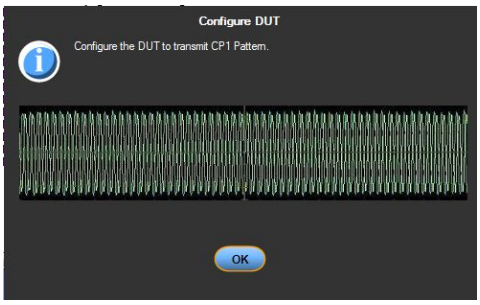
Manual toggle

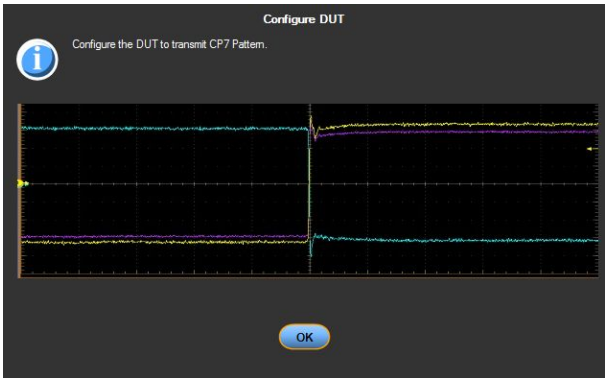
To not use the AWG-based toggle capability, do the following:

1. Click **Setup > Configuration > Global Settings**.
2. In the Instruments Detected field, set the **Toggle using** parameter to **Do not use**.

3. Run the test. When the application must acquire a CP0, CP1, CP7, CP9 or CP10 pattern, it opens windows similar to the following graphics, prompting you to manually transmit the pattern signal and acquire the waveform, and validate it against the displayed waveform.







4. Click **OK** to acquire the waveform. If Pattern type validation is set to **Prompt me if Signal Check Fails**, the application runs a pattern type validation on the acquired signal. If the acquired signal is a valid pattern, the measurement continues normally.

   If it is not a valid pattern, the application shows one of the following message dialog boxes:

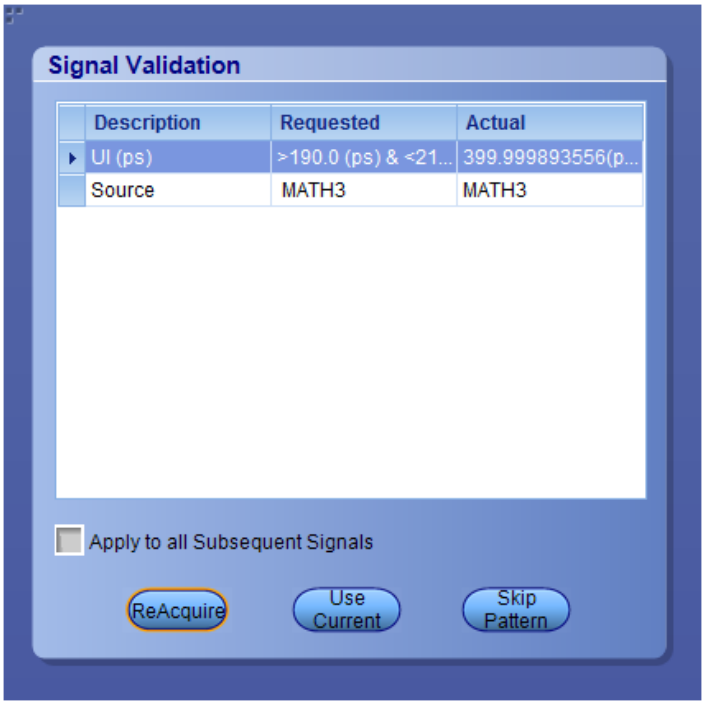


Figure 4: Compliance Pattern Signal Validation

NOTE. *For CP13, CP14 and CP15 - If CP13 Signal validation fails and if user selects Use Current or Skip, same will be applied for CP14 and CP15.*

---

5. Choose how to continue:

   ■ Click **Reacquire** to start the acquisition again.

   ■ Click **Use Current** to continue with the currently acquired waveform.

   ■ Click **Skip Pattern** to skip the current tests. The rest of the selected measurements continue.

See also. *Oscilloscope-based toggle*

AWG-based toggle

AFG-based toggle

Index

T

U

V

W