TekExpress® USB3 Tx
USB 3.1 Automated Test Solution Software

Printable Application Help

077-0939-01

Tektronix

TekExpress® USB3 Tx
USB 3.1 Automated Test Solution Software

Printable Application Help

www.tektronix.com
077-0939-01

Tektronix

Copyright © Tektronix. All rights reserved. Licensed software products are owned by Tektronix or its subsidiaries or suppliers, and are protected by national copyright laws and international treaty provisions. Tektronix products are covered by U.S. and foreign patents, issued and pending. Information in this publication supersedes that in all previously published material. Specifications and price change privileges reserved.

TEKTRONIX and TEK are registered trademarks of Tektronix, Inc.

Contacting Tektronix

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

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

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

Table of Contents

Welcome 5

Getting help and support

Related documentation 1
Conventions used in help 2
Technical support 2

Getting started

Installing the software 5
- Minimum system requirements 5
- Required windows 7 user account setting 6
- Supported instruments 7
- Install the software 8
- Verify application installation 8
- Activate the license 8
- View software version 9
- Required my TekExpress folder settings 9
- Set the my TekExpress folder permissions 9
- Application directories and their contents 11
- File name extensions 12
- Where test files are stored 12

Operating basics

Run the application 13
Exit the application 14
Application controls and menus 15
- Global application controls 15
- Application test panels 25

Running tests

Test process flow 49
Deskew real-time oscilloscopes 50
Instrument and DUT connection setup 51
Running tests 51
Prerun checklist 52

Saving and recalling test setup files

Test setup files overview 53
Save a test setup file 53
Open (load) a saved test setup file 54
Run a saved test in prerecorded mode 55
Create a new test setup file based on an existing one 56

TekExpress programmatic interface

About the programmatic interface 57
To enable remote access 58
Requirements for developing TekExpress client 60
Remote proxy object 61
Client proxy object 62
Client programmatic interface example 63
Program remote access code example 66
USB-TX programmer interface commands 67
- Command list 67
- ApplicationStatus() 67
- CheckSessionSaved() 68
- Connect() 68
- Disconnect() 70
- GetCurrentStateInfo() 71
- GetDutId() 72
- SetDutId() 73
- GetGeneralParameter() 73
- GetReportParameter() 74
- GetResultsValue() 76
- GetSelectedVersions() 77
- GetTimeOut() 77
- LockServer() 78
- LockSession() 79
- QueryStatus() 80
- RecallSession() 81
- RegisterStatusChangeNotification() 81
- Run() 83
- SaveSession() 83
- SaveSessionAs() 84

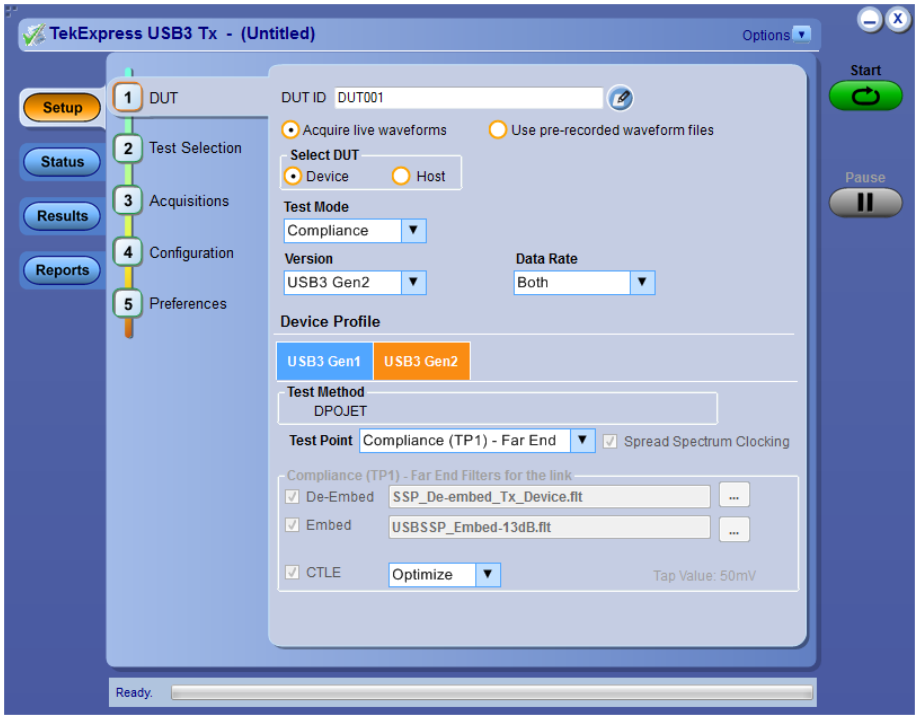
SelectSingleTest() 85
SendResponse() 86
SelectDevice() 87
SelectSuite() 88
SelectTest() 88
SelectVersions() 90
SetInstrument() 90
SetPreRecorded() 91
SetTimeOut() 92
SetVerboseMode() 93
Status() 94
Stop() 94
TransferImages() 95
TransferResult() 96
TransferWaveforms() 96
UnlockServer() 97
UnlockSession() 97
GetPassFailStatus() 98
The SetGeneralParameter command 99
SetGeneralParameter() 99
paramString values for SetGeneralParameter command 101
Select CTLE filter file 101
Select embed filter file 101
Select test method 102
Select test point 102
Set SDLA CTLE mode 103
Select de-embed filter file 103
Set SSC mode 104
Set embed filter mode 104
Set CTLE filter mode 105
Set probing configuration 105
Set bandwidth for LFPS acquisition 106
Set CM measurement TriMode probe mode 106
Set compliance test mode 106
Set auto recovery mode 107
Set verify toggle mode 107
Set LFPS trigger lower limit 107
Set LFPS trigger upper limit 107
Set LFPS trigger level 108

Set LFPS mid edge reference level 108
Set hysteresis level 108
Set AFG number of cycles 108
Set AFG frequency 109
Set AFG voltage level high 109
Set AFG voltage level low 109
Set signal pattern validation mode 109
Set de-embed filter mode 110
Set ADC for CTLE 110
Set DUT Power Cycle Method 110
Set Voltage 111
Set Current Level 111
Set AWG DC Output 112
Set Output ON or OFF Time 112
Set Radio Friendly Clocking mode 112
Set Toggle Using 113
Set record length 113

Reference

Handle error codes 115
Limits editor: compare string definitions 116
De-Embedding and channel embedding information 117
De-Embedding and channel embedding overview 117
Host filter information 118
Device filter information 120
USB3 gen2 filter information 122
DUT-Filter combinations 124
Signal validation 125
LFPS pattern type validation 125
CPx pattern type validation 126
CP0 CP1 CP7 toggle mechanisms 127
Oscilloscope-based toggle 127
AWG-based toggle 129
AFG-based toggle 130
Manual toggle 131

Welcome



Welcome to the TekExpress® USB3 Tx Automated Test Solution Software application (referred to as USB3 Tx in the rest of the document). TekExpress USB3 Tx provides an automated, simple, and efficient way to test USB3 Tx transmitter interfaces and devices consistent to the requirements of the USB 3.1 specifications (Gen1 and Gen2).

Key features and benefits

- ECN-18 Changes for USB3 Gen1
- Automatic DUT Power cycle
- Comprehensive test coverage; select or deselect individual tests
- Precise debugging and troubleshooting
- Intel SigTest is integrated into the TekExpress framework
- Minimizes user intervention when performing time-consuming testing
- SSC Measurements controlled through a single UI control
- Able to select individual filters for de-embedding, embedding, and equalization
- Automatically detect Tektronix TriMode™ P7500 series probes
- De-emphasis measurement uses CP7 pattern
- Automatic compliance pattern toggle

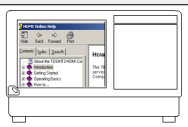
- Consolidated report with both SigTest and DPOJET measurement results
- LFPS Vcm-AC measurement support using a Tektronix TriMode Probe
- SDLA integration for Gen2 (10 Gbps) equalization
- Complete programmatic interface enables automation scripts to call USB3 Tx functions

Getting help and support

Related documentation

The following manuals are available as part of the TekExpress® USB3 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	www.Tektronix.com 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.

Table 2: Icon descriptions

Icon	Meaning
	This icon identifies important information.
	This icon identifies conditions or practices that could result in loss of data.
	This icon identifies additional information that will help you use the application more efficiently.

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

Installing the software

Minimum system requirements

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

Table 3: USB3 system requirements

Component	Requirement
Oscilloscope	See *Supported instruments*
Processor	Same as the oscilloscope
Operating System	Same as the oscilloscope: ■ Windows 7 (64-bit only) SP1 *Windows 7 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]

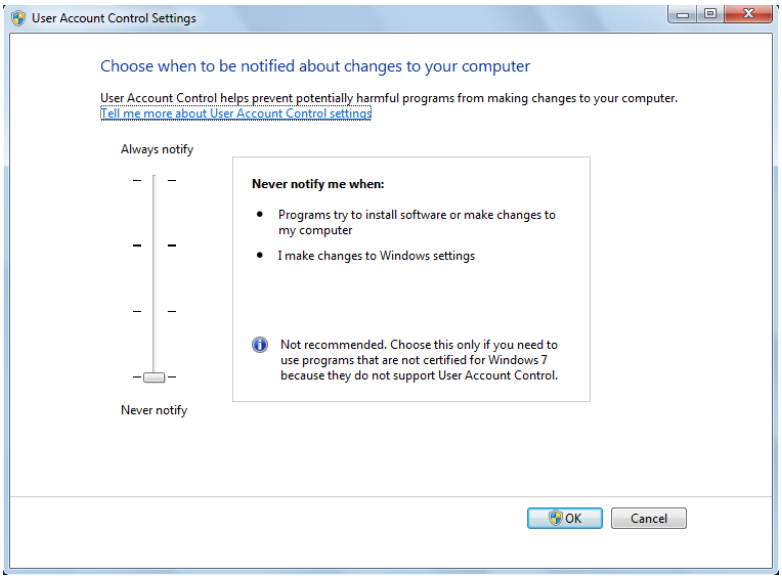
[1] If TekExpress is running on an instrument that has a video resolution less than 800x600, connect and configure a second monitor to the instrument.

Component	Requirement
Firmware	■ TekScope 7.2.0 and later (for Windows 7)
Software	■ TekExpress Framework (version 3.0.x or later) installed. ■ Microsoft .NET 4.0 Framework ■ DPOJET Jitter and Eye Analysis Tool (version 6.2.1 or later) 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 2.2.0 or later, for Channel De-Embed, for custom filter development.

Required windows 7 user account setting

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

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



See also. *Supported oscilloscopes*

Supported instruments

Table 4: Required equipment

Resource	Model supported
Real-time oscilloscope	Tektronix DPO/DSA/MSO70000C, D, and DX series oscilloscopes (Windows 7 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 P7313SMA differential probe P7500 differential probe
Host test fixtures	TF-USB3-A-P (for best signal quality) For more mechanical flexibility use TF-USB-B-R (with included 13 cm USB 3.0 Cable - Part number 174-5772-00). For precision De-embed of TF-USB3-A plug fixture, order TF-USB3-AB-KIT (includes Cal Kit). TF-USB3-KIT (includes short USB 3.0 cable) USB-IF fixtures [2]
Device test fixtures	TF-USB3-A-R (includes short USB 3.0 Cable) USB-IF fixtures [3]
Tektronix AWG/AFG instruments	AWG7102, AWG7122 Series with options 6,8 AWG70002A, AWG70001A AWG5014B, AWG5014C,AWG5012C, AWG5002C AFG3252, AFG3252C, AFG3251, AFG3251C, AFG3102, AFG3102C, AFG3101, AFG3101C
Tektronix Power Supply instruments	PWS4205,PWS4305, PWS4323, PWS4602,PWS4721. AWG7122B, AWG7122C, AWG7102

See also. *Minimum system requirements*

[2] Available through USB-IF.
[3] Available through USB-IF.

Install the software

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

1. Close all applications (including the TekScope application).
2. Go to the www.tek.com Web site and locate the **Downloads** fields.
3. Enter **tekexpress usb** in the *Model or Keyword* field, select **Software** from the *Select Download Type* list, and click **GO**.
4. Select the latest version of software. 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.

   Software is installed at C:\Program Files (x86)\Tektronix\TekExpress \TekExpress USB3 Tx
7. *Verify application installation*

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 Tx** is listed in the Analyze menu.
4. Click **TekExpress USB3 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 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 Tx** and **USBSSP-Tx: TekExpress USB3 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*.

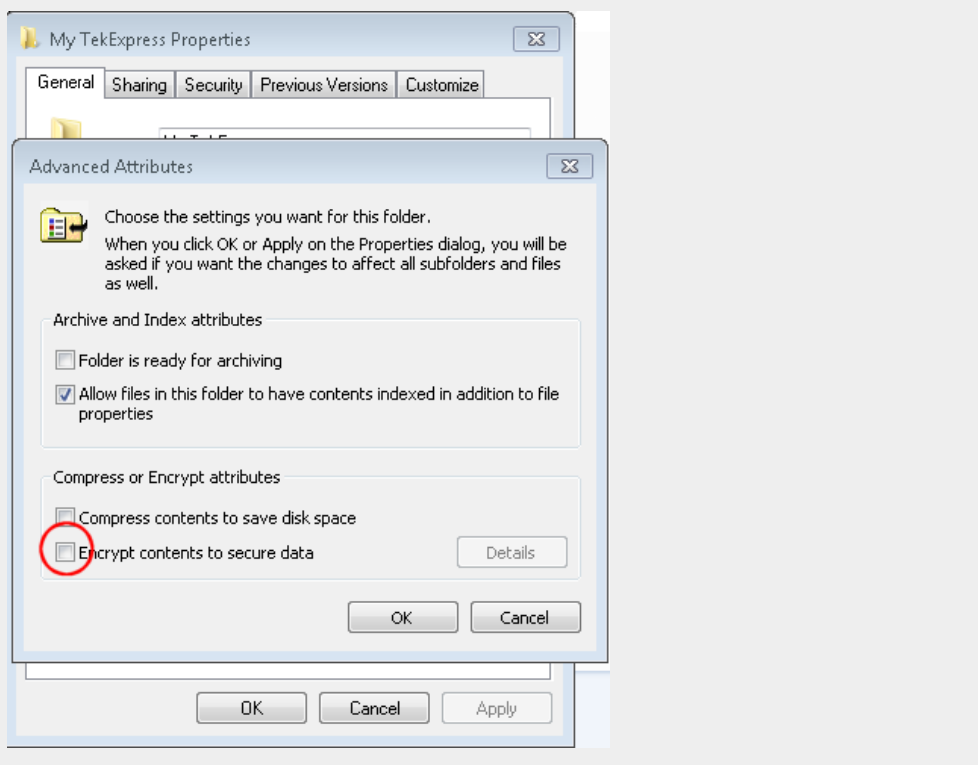
See also. *Application directories and usage*

File name extensions

Set the my TekExpress folder permissions

Make sure 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, make sure 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.

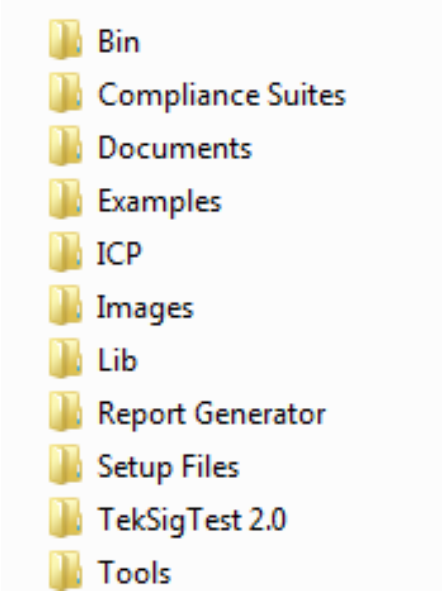
See also. *Application directories and usage*

File name extensions

Application directories and their contents

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

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



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

Table 5: Application directories and usage

Directory names	Usage
Bin	Contains USB3 TX application libraries
Compliance Suites	Contains compliance-specific files
Examples	Contains various support files
ICP	Contains instrument and USB3 TX application-specific interface libraries
Lib	Contains utility files specific to the USB3 TX application
Report Generator	Contains style sheets for report generation
Tools	Contains instrument and USB3 TX application-specific files
TekSigTest 2.0	Contains SigTest specific interface files
.pdf, .chm	Help files

See also. *View test-related files*

File name extensions

File name extensions

The TekExpress USB3 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

See also. *View test-related files*

Application directories and their contents

Where test files are stored

When you launch TekExpress USB3 Txfor the first time, it creates the following folders on the oscilloscope:

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

Every time you launch TekExpress USB3 Tx, an Untitled Session folder is created in the USB3 Tx folder. The Untitled Session folder is automatically deleted when you exit the USB3 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

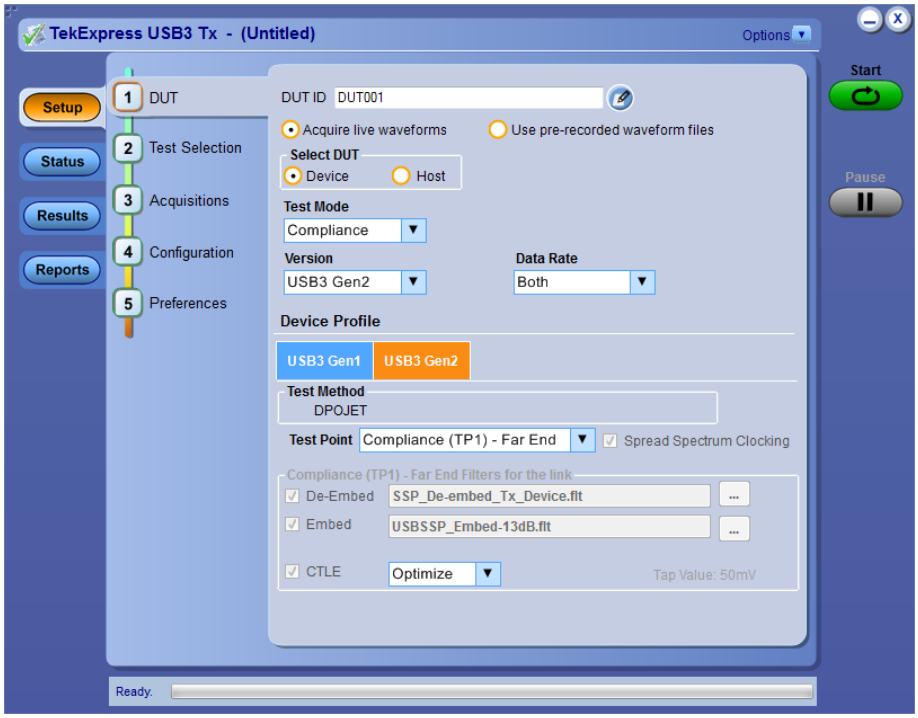
Operating basics

Run the application

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

- Select **Analyze > TekExpress USB3 Tx** from the TekScope menu.
- Double-click any saved TekExpress USB3 Tx session file (<file name>.TekX).

The oscilloscope opens the TekExpress USB3 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 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 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 Tx.

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

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

See also

Required My TekExpress folder settings

Activate the license

Exit the application

Application controls

Application panel overview

Exit the application

Use the following method to exit the application:

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

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

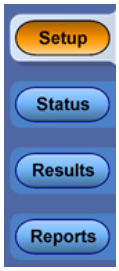
NOTE. *To minimize the application, click* *on the application title bar.*

Application controls and menus

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
Application window move icon	Place the cursor over the three-dot pattern in the upper left corner of the application window. When the cursor changes to a hand, drag the window to the desired location.

See also. *Application panel overview*

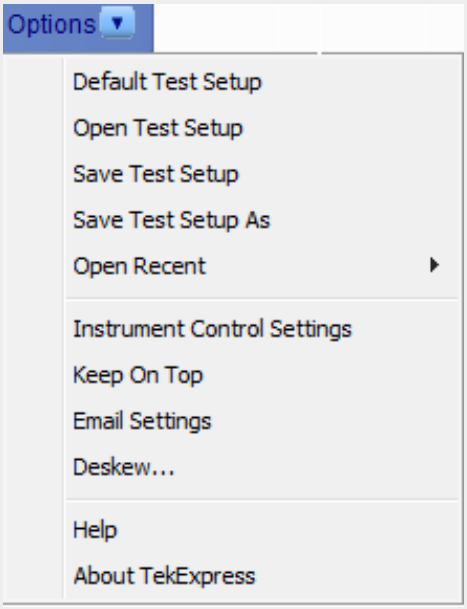
Options menu.

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

The Options menu has the following selections:

Menu	Function
Default Test Setup	Opens an untitled test setup with defaults selected
Open Test Setup	Opens a saved test setup
Save Test Setup	Saves the current test setup selections
Save Test Setup As	Creates a new test setup based on an existing one
Open Recent	Displays a menu of recently opened test setups to select from
Instrument Control Settings	Detects, lists, and refreshes the connected instruments found on specified connections (LAN, GPIB, USB, and so on)
Keep On Top	Keeps the TekExpress USB3 Tx application on top of other open windows on the desktop
Email Settings	Use to configure email options for test run and results notifications
Help	Displays the TekExpress USB3 Tx help
About TekExpress	■ Displays application details such as software name, version number, and copyright ■ Provides access to *License information* for your USB3 Tx installation ■ Provides a link to the Tektronix Web site

Options menu

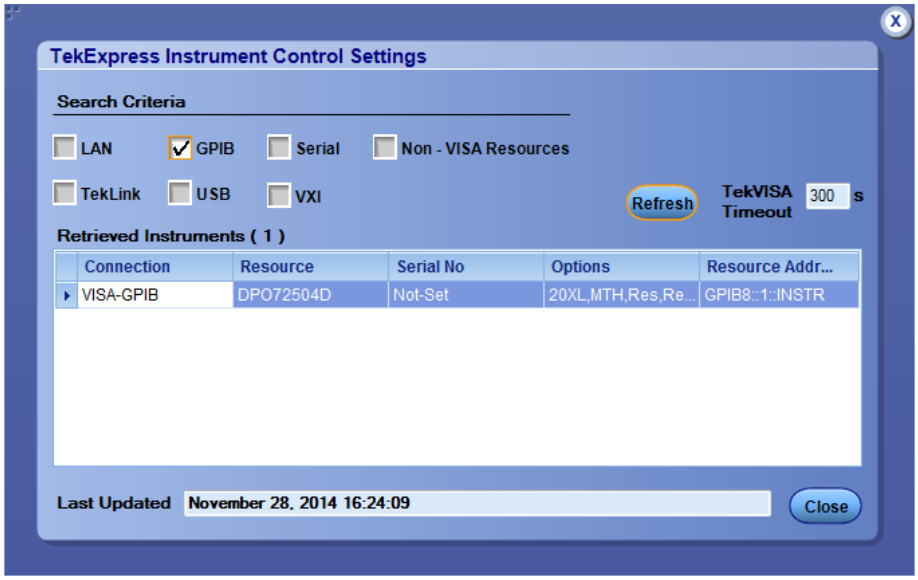


See also. *Application controls*

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*

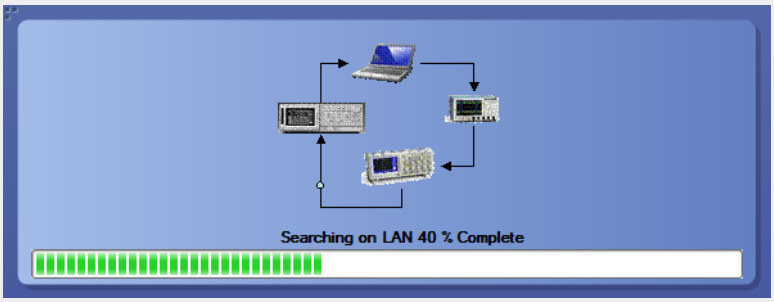
View connected instruments. Use the Instrument Control Settings dialog box to view or search for connected instruments required for the tests. The application uses TekVISA to discover the connected instruments on all selected connection types.

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

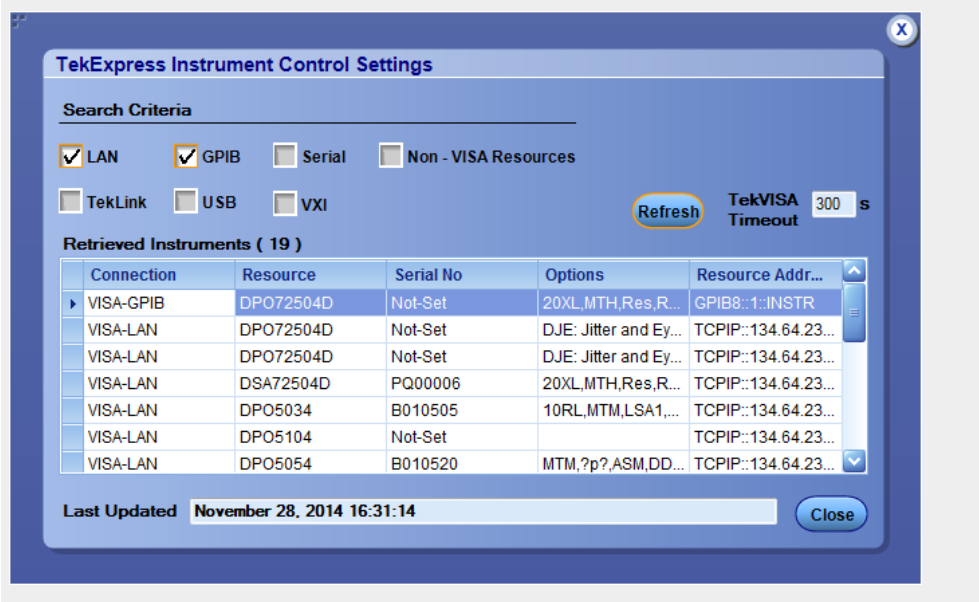
To refresh the list of connected instruments:

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.

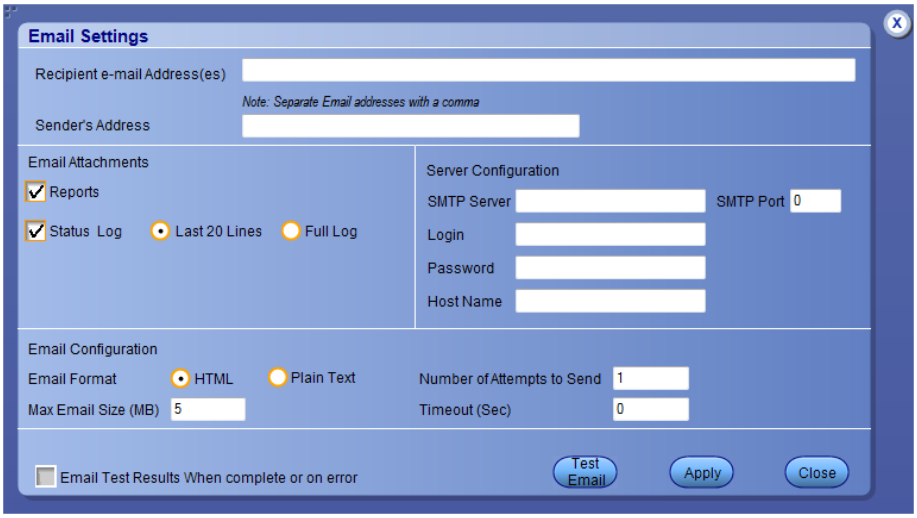
See also. *Equipment connection setup*

Email settings.

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

Select **Options > Email Settings** to open this dialog box.

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



See also. *Configure email settings*

Options menu

Select test notification preferences

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

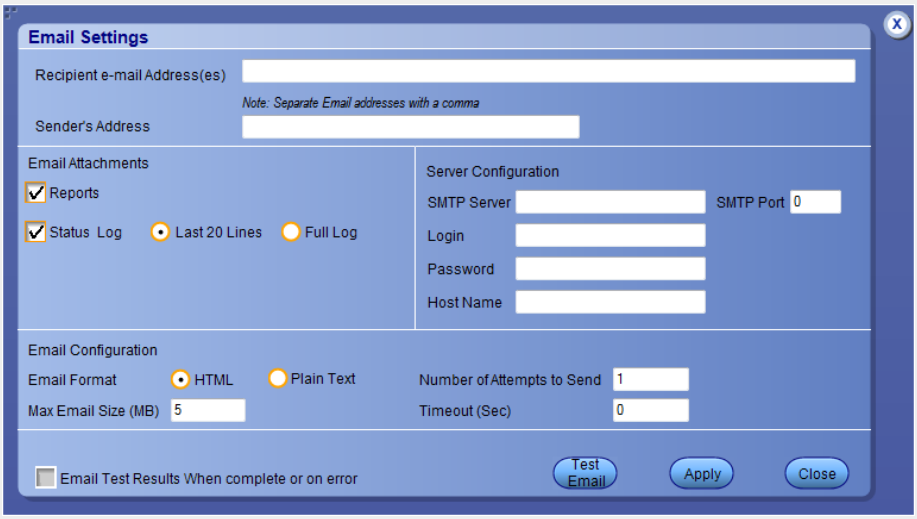
1. Select **Options > Email Settings** to open the Email Settings dialog box.
2. (Required) For Recipient email Address(es), enter one or more email addresses to which to send the test notification. To include multiple addresses, separate the addresses with commas.
3. (Required) For Sender's Address,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

Email settings



Application test panels

Application panels overview. TekExpress USB3 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.

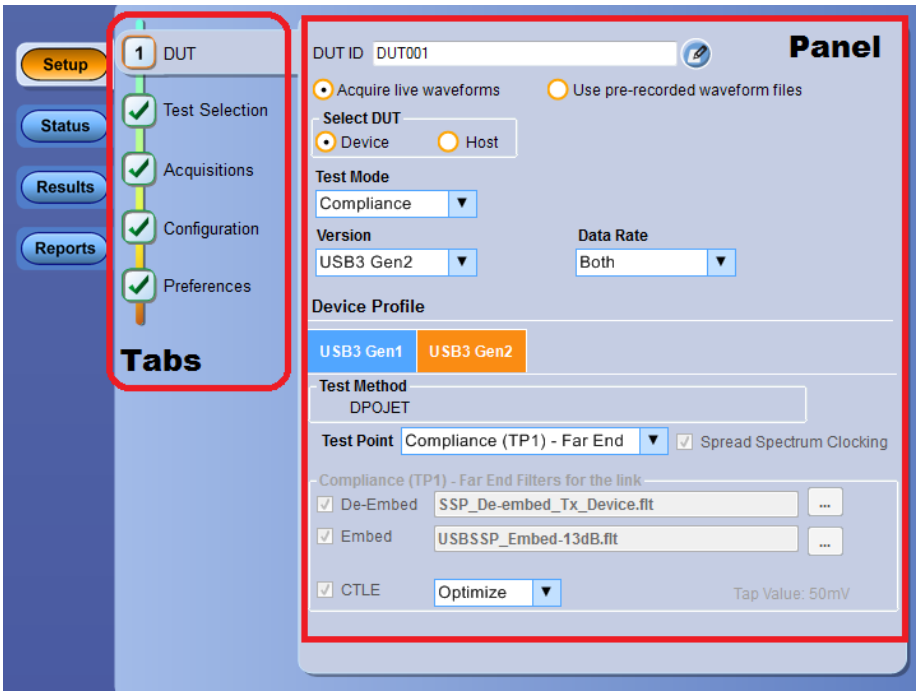


Table 7: Application panels overview

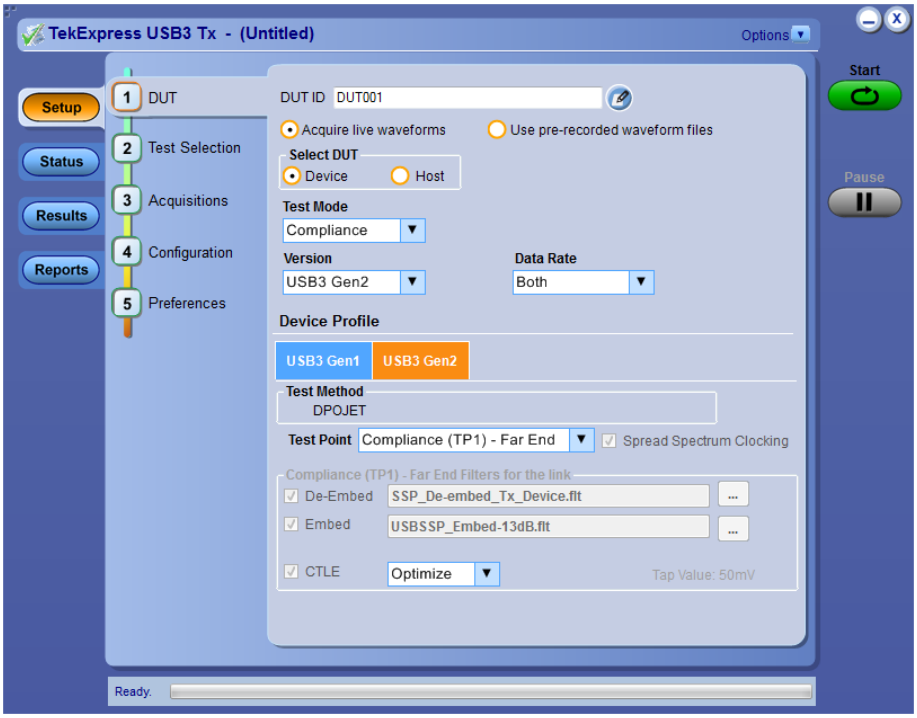
Panel Name	Purpose
Setup	The Setup panel shows the test setup controls. Click the **Setup** button to open this panel. Use this panel to: ■ *Select DUTparameters*. ■ *Select the test(s)*. ■ *Set acquisitions parameters* for selected tests. ■ *Select test notification preferences*.
Status	View the progress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

Panel Name	Purpose
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*

Setup tabs.

Setup controls 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*

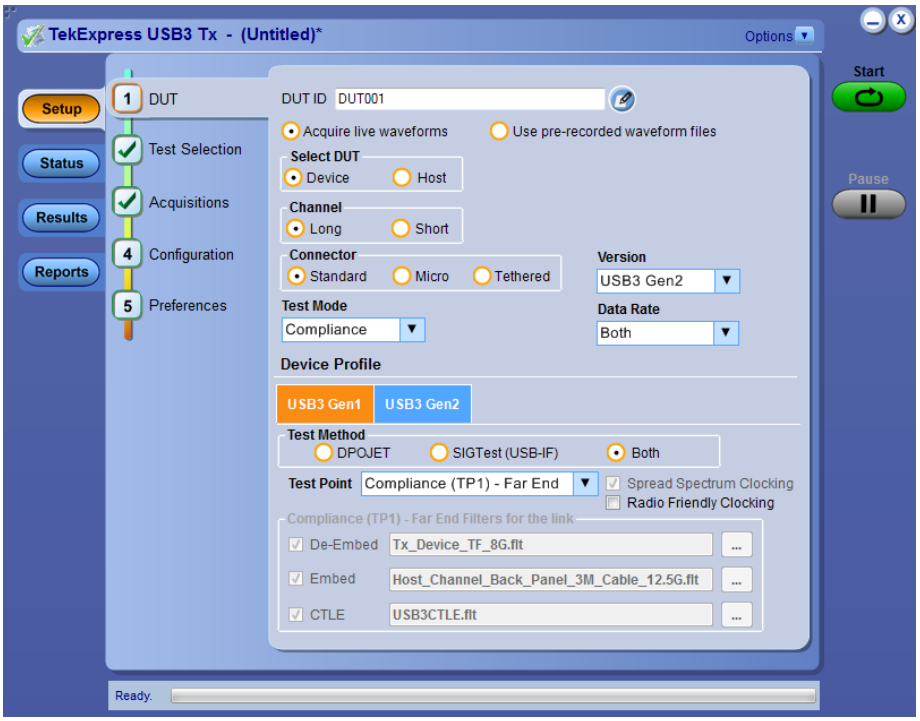
Test Selection: *Select test(s)*

Acquisitions: *Select acquisition parameters*

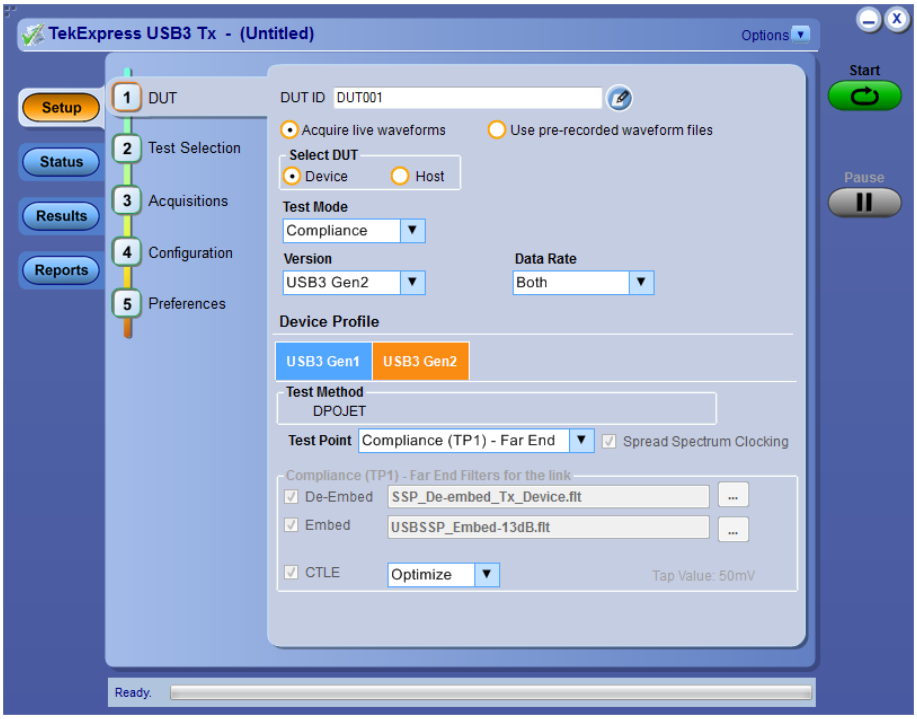
Preferences: *Select test fail notification preferences*

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. DUT settings also affect the list of available tests in the Test Selection tab.

DUT=USB3 Gen1



DUT=USB3 Gen2



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

Table 8: 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*.)
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*
Select DUT	Sets the DUT device type to test (Device or Host).

Setting	Description
Channel	It's a combination of different length of cables and PCBs. Dependening on channel selection, different filters will be selected on the DUT panel. ■ Long: Consists of 3m cable or 1 m cable with PCB assembly. ■ Short: Only PCBs, no cables attached.
Connector	Select the appropriate connector form the options. ■ Standard: For type A and type B, select Standard. ■ Micro: For Micro A and Micro B, select Micro. ■ Tethered: For Standard A plug, select Tethered. It is applicable only for Device DUT type.
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).
Version	Lists the supported USB3 generations.
Data Rate	Sets the test data rate (5 Gbps, 10 Gbps, or Both).
Device Profile	
USB3 Gen1, USB3 Gen2 tabs	Sets the available test parameters for the Device Profile area. Parameter availability is also dependent on the Test Mode setting.

Setting	Description
Test Method	Sets the algorithms used to measure and analyze the signal. DPOJET is the default test method. ■ 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. ***NOTE.*** *The options SIG Test(USB-IF) and Both are applicable only for USB3 Gen1 but not for USB3 Gen2.*
Test point	Select the appropriate test point location from those listed. Only Compliance (TP1) - Far End test point is available when Sigtest or Both are selected.
Spread Spectrum Clocking	Select this check box if your DUT supports Spread Spectrum Clocking (SSC). Selects SSC tests based on your DUT configuration.
Radio Friendly Clocking	This parameter allows you to change the SSC clocking to avoid direct interference with radios operating close to USB3.0 fundamental frequencies. ***NOTE.*** *In user defined mode, if SSC is turned off, then Radio Friendly Clocking also will be turned off.*
Filters for the link	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. ***NOTE.*** *These controls are not available when Test Mode is set to Compliance.* Click the ... button to view and select other filter files. For USB3 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. *DeEmbeding and Channel Embedding Overview*

See also. *Select a test*

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

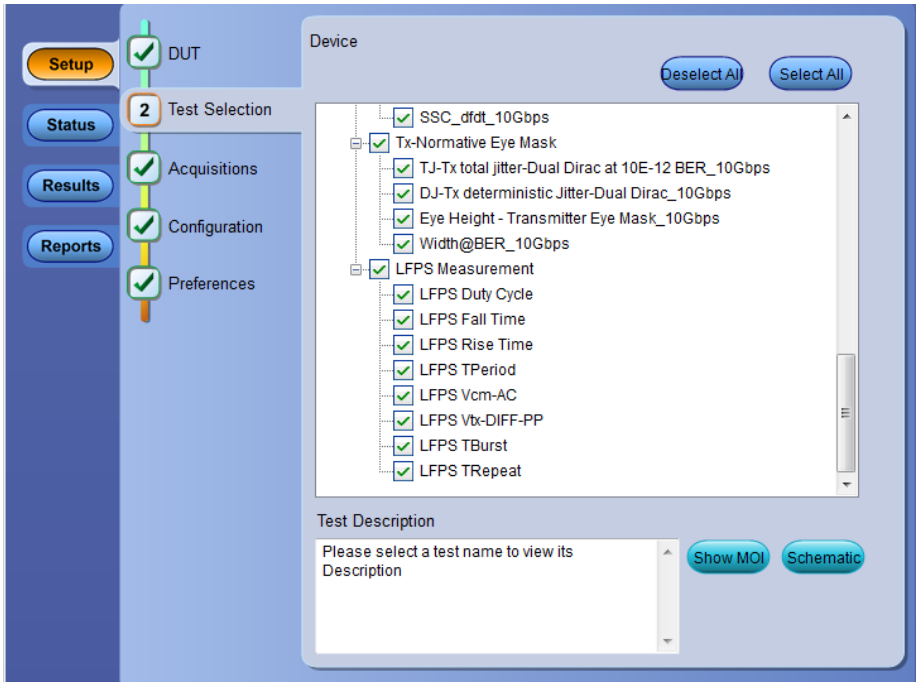


Table 9: Test Selection tab settings

Setting	Description
Deselect All, **Select All** buttons	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 button	Opens the Method of Implementation (MOI) PDF file. You must have selected a test before you can open the MOI.
Schematic button	Shows an equipment and test fixture setup schematic (connection diagram) for the selected test. Use to set up the equipment and fixtures or to verify the setup before running the test.

NOTE. *All tests are selected by default.*

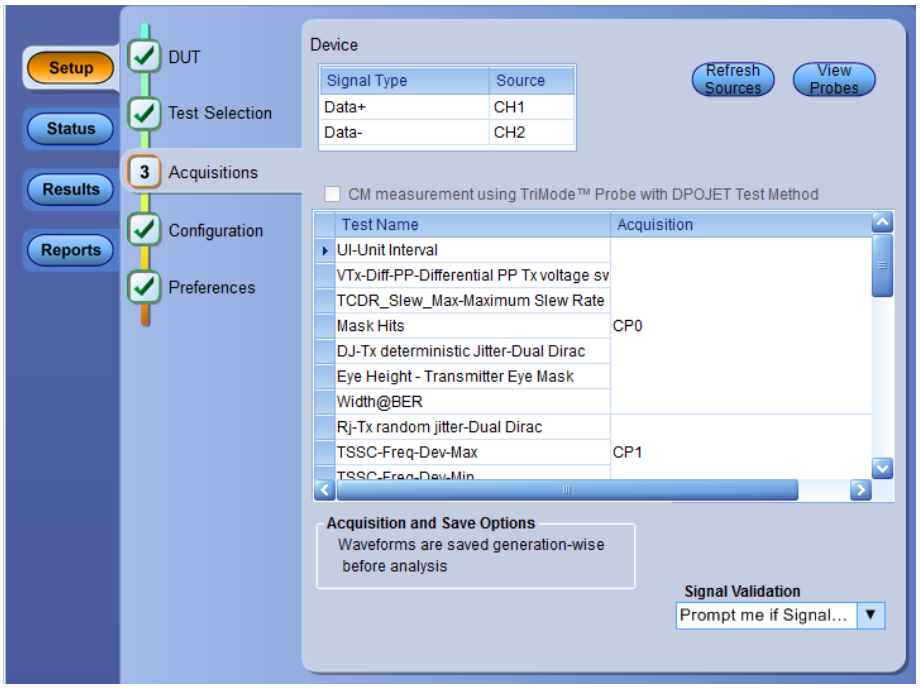
NOTE. *The application does not show the oscilloscope cursor1 and 2 for each burst. The application runs an analysis on the first five bursts of an acquisition and displays the result statistics.*

See also. *Set acquisition parameters*

Acquisitions tab.

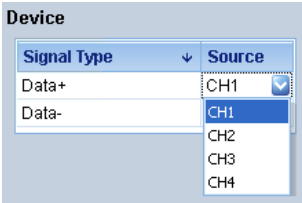
Set acquisition parameters. Use the **Acquisition** tab in the Setup panel to view test acquisition parameters. You also use this tab to load prerecorded (saved) test session waveform files on which to run tests.

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



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

Table 10: Acquisitions tab settings

Setting	Description
Device	Lists the signal type and input channel assigned to that type. Click in a Source fields to assign a channel source to a signal type. 

Setting	Description
Refresh Sources button	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 button	Use the View Probes dialog box to show the detected probe configurations, and enable or disable probe signal source access in the application. Only available for 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.

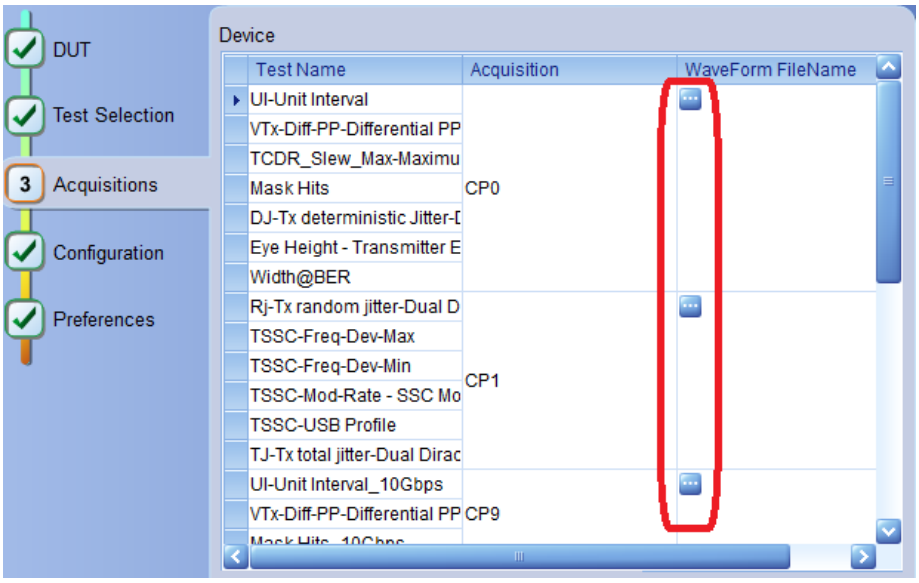
USB3 Tx saves all acquisition waveforms to files by default. Waveforms are saved to a folder that is unique to each session (a session starts when you click the Start button). The folder path is X:\TekExpress USB3 Tx\Untitled Session \<dutid>\<date>_<time>. Images created for each analysis, reports, 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.

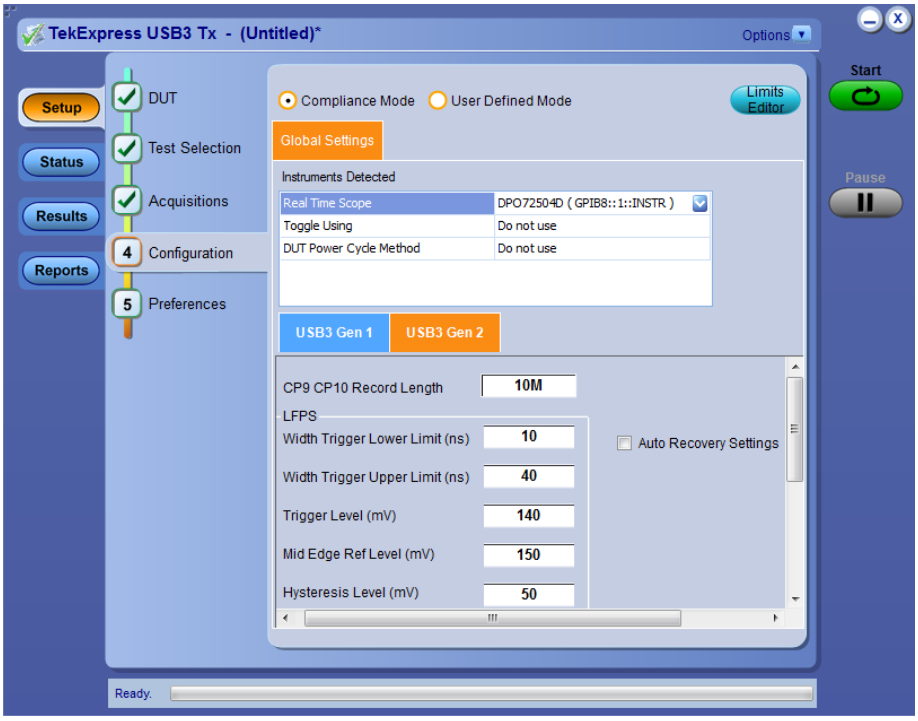


4. Click the browse button (...) for each test acquisition type (CP0, CP1, LFPS).
5. Navigate to and select the appropriate waveform file(s). You must select all waveforms required for the acquisition type.
6. To change, remove, or add a file to the list, click the browse button next to the file name to change, and use the menu items to replace, remove (delete) or add a file in the list.
7. Click **Start**.

Configure test parameters.

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.

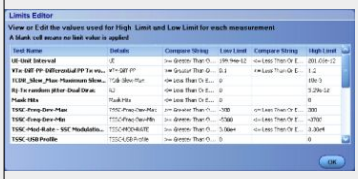
NOTE. *You cannot change test parameters that are grayed out.*



See also. *Configuration tab: Global settings parameters*

Configuration tab: global settings parameters. The following table lists the Configuration tab settings and parameters. Fields shown on this tab can change depending on selected items.

Table 11: Configuration tab Global Settings

Control	Description
Test Mode	Determines whether test parameters are in compliance or are able to be edited. ■ Compliance: Most test parameter values cannot be edited. Select this for compliance mode testing. ■ User Defined: Enables editing of filters for the link.
Limits Editor button	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. *Limits Editor: compare string definitions* 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.
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*.
CP0 CP1 CP7 Record Length (USB3 Gen1) CP9 CP10 Record Length (USB3 Gen2)	Sets the waveform record length.
LFPS fields	Sets LFPS waveform parameters. 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.

NOTE. *If AWG is selected as the Toggle tool, the application enables the check box. Use this function to identify if the DUT toggled twice instead of one time.*

NOTE. *If 'Do not use' is selected as the Toggle tool, the application enables the check box. Use this function to automatically recover the oscilloscope settings if modified by the user during test execution.*

See also. *About acquisitions*

Limits Editor: compare string definitions

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

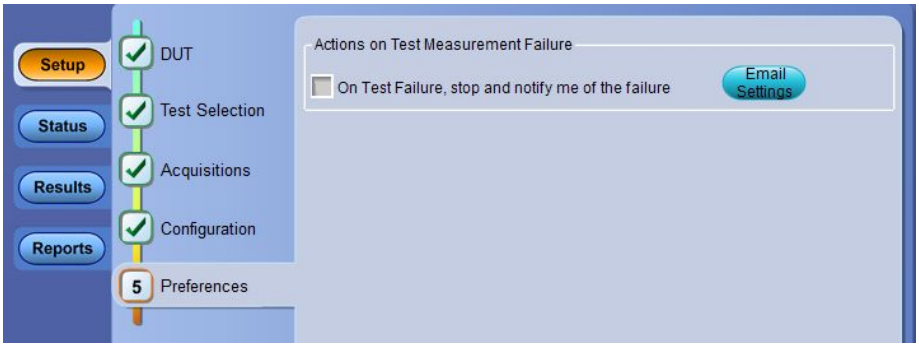
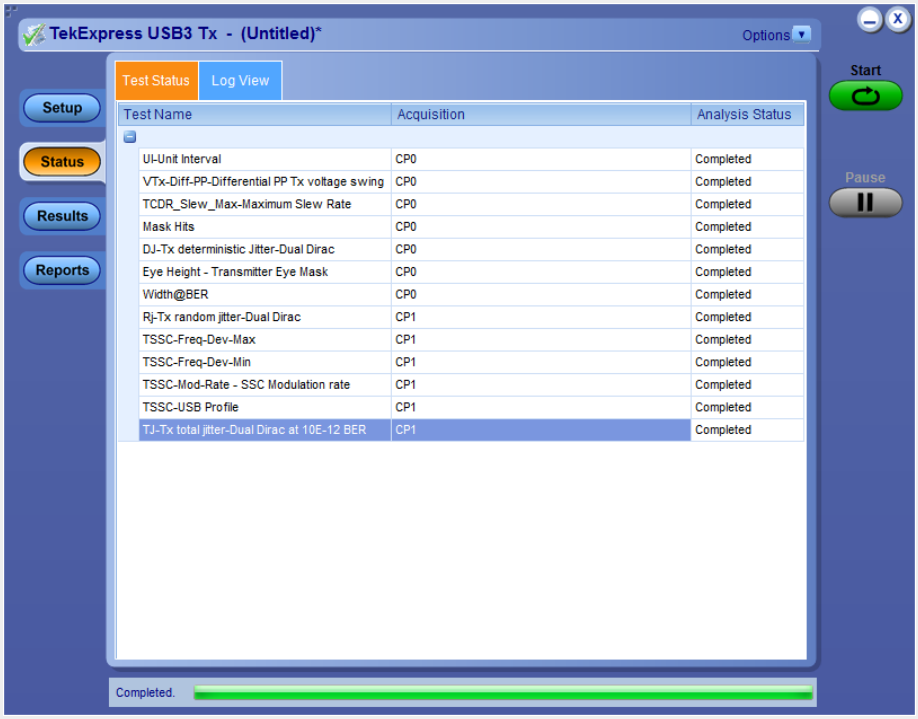


Table 12: Preferences tab settings

Setting	Description
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.

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

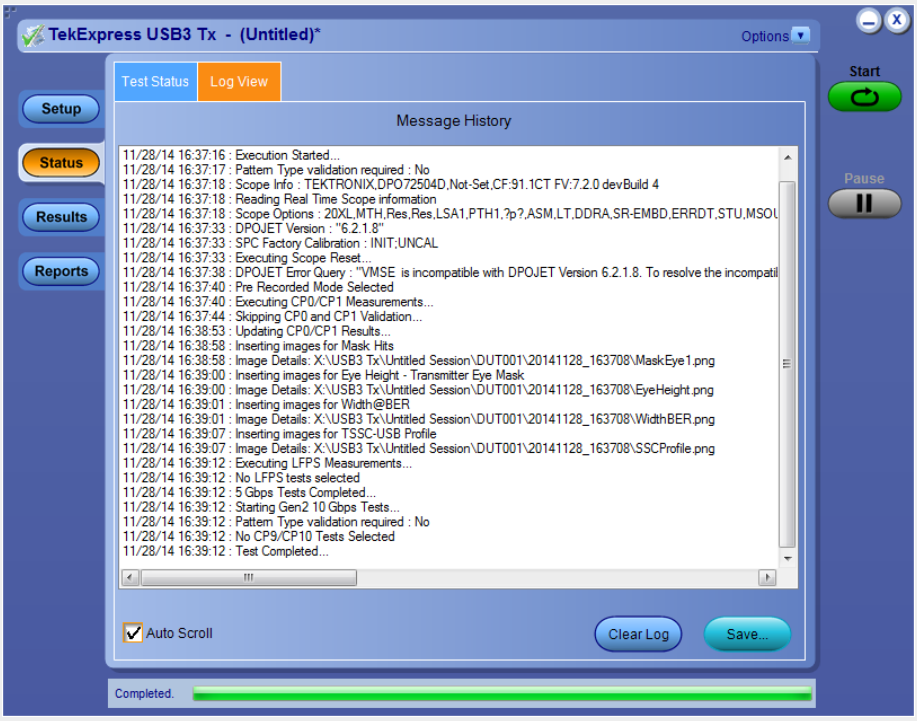


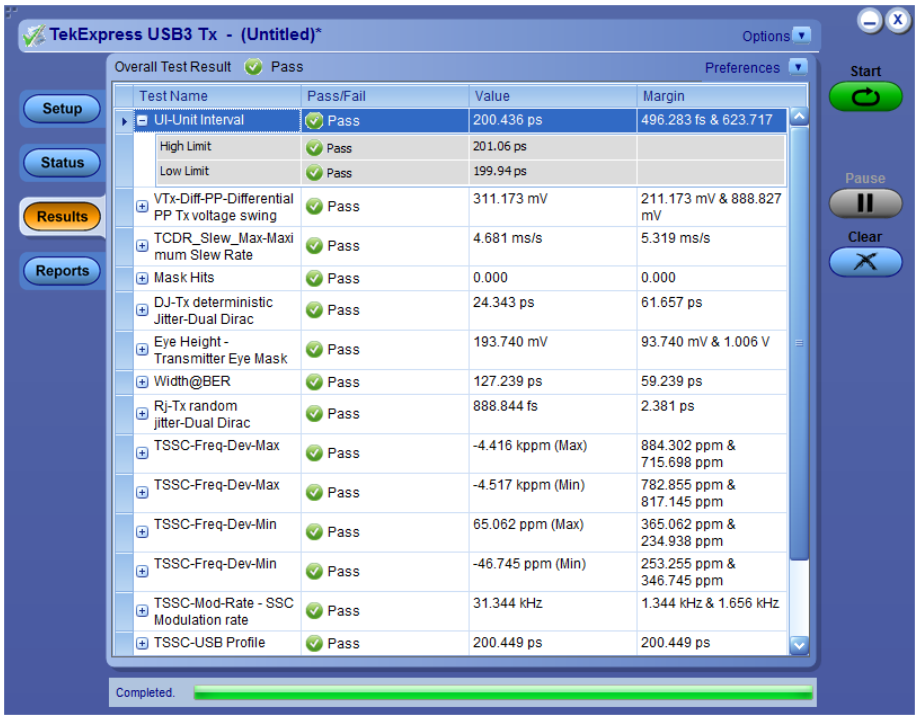
Table 13: Status panel settings

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

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**.

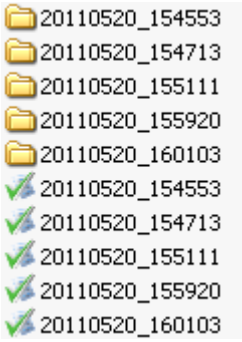
See also. *Results panel overview*

View test-related files. Files related to tests are stored in the My TekExpress \USB3 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.

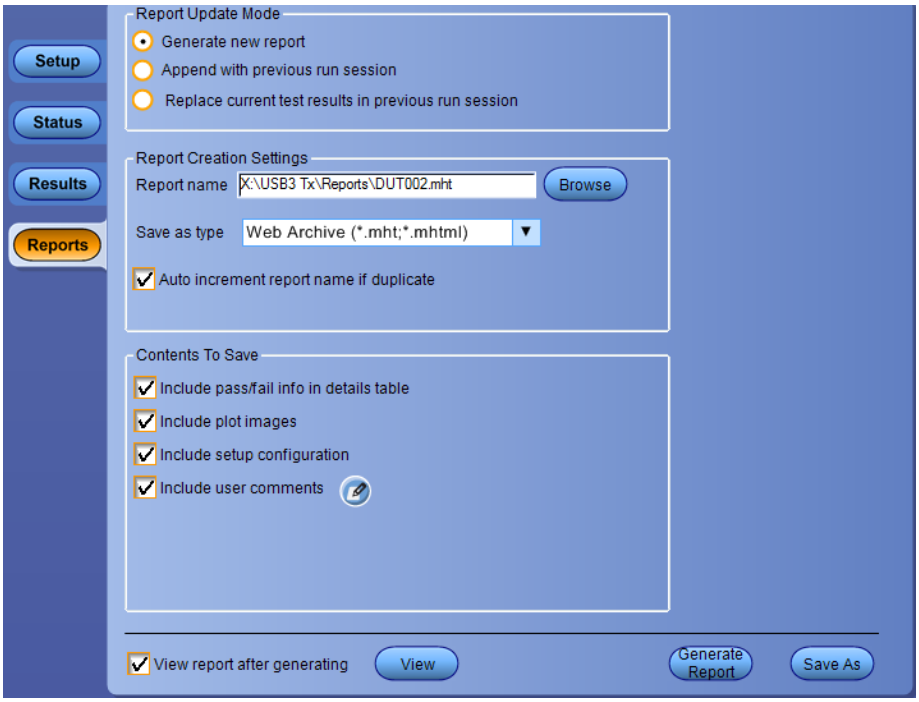
The first time you run a new, unsaved session, the session files are stored in the Untitled Session folder located at ..\My TekExpress\TekExpress USB3 Tx. When you name and save the session, the files are placed in a folder with the name that you specify. A copy of the test files stay in the Untitled Session folder until you run a new test or until you close the USB3 Tx application.

See also. *File name extensions*

Required \My TekExpress folder settings

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 14: Report options

Setting	Description
Report Generation	
Generate new report	Creates a new report.
Append to previous run session	Appends the latest test results to the end of the current session's test results report.
Replace current test in previous run session	Replaces the previous test results with the latest test results. Results from newly added tests are appended to the end of the report.
Report name	Displays the name and location from which to open a report. The default location is at *\My TekExpress\USB3 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\USB\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.

Setting	Description
Save as type	Saves a report in the selected output format (Web archive or PDF).
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 plot images	Sets the application to include plotted diagrams such as Eye diagrams.
Include setup configuration	Sets the application to include hardware and software information in the summary box at the top of the report. Information includes: the oscilloscope model and serial number, probe model and serial number, the oscilloscope firmware version, SPC and factory calibration status, and software versions for applications used in the measurements.
Include user comments	Select to include any comments about the test that you or another user added in the DUT tab of the Setup panel. Comments appear in the Comments section, under the summary box at the beginning of each report.
View Report After Generating	Automatically opens the report in a Web browser when the test completes. This option is selected by default.
View button	Click to view the most current report.
Generate Report	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 3.1 Report

Test Report

Execution and Setup Information

DUT ID	DUT001	Suite	Device
Date/Time	2014-11-28 16:37:17	Scope Model	DPO72504D
Connector Type	Standard	Scope Serial Numer	Not-Set
Test Point	Compliance (TP1) - Far End	Scope F/W Version	7.2.0 devBuild 4
DPOJET Version	"6.2.1.8"	SPC Factory:S/W Calibration	INIT;UNCAL
Total Execution Time	1 Minute 55 Seconds	TekExpress Version	USB:6.0.5.14 Framework:3.0.1.64
Toggle Tool	Do not use		
Acquisition Mode	PreRecorded		
Over All Test Result	Pass		
DUT COMMENT:	General Comment - USB3.1 DUT		

UI-Unit Interval

Measurement Details	Generation	Method	Measured Value	Test Result	Margin	Low Limit	High Limit
UI-Unit Interval	Gen 1	DPOJET	200.436 ps	Pass	496.283 fs & 623.717 fs	199.94 ps	201.06 ps
COMMENTS							

VTx-Diff-PP-Differential PP Tx voltage swing

Measurement Details	Generation	Method	Measured Value	Test Result	Margin	Low Limit	High Limit
VTx-Diff-PP-Differential PP Tx voltage swing	Gen 1	DPOJET	311.173 mV	Pass	211.173 mV & 888.827 mV	100.0 mV	1.2 V
COMMENTS							

TCDR_Slew_Max-Maximum Slew Rate

Measurement Details	Generation	Method	Measured Value	Test Result	Margin	Low Limit	High Limit
TCDR_Slew_Max-Maximum Slew Rate	Gen 1	DPOJET	4.681 ms/s	Pass	5.319 ms/s	N.A	10.0 ms/s
COMMENTS							

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 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 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 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 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 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 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 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

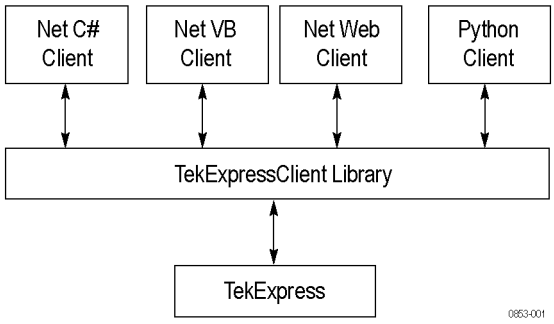
Set DUT parameters

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.

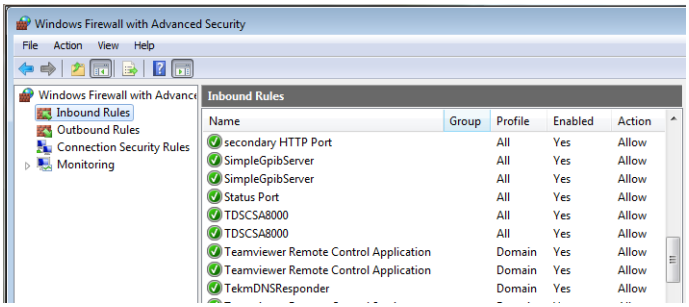
TekExpress leverages .Net Marshalling to enable the Programmatic Interface for TekExpress Client. TekExpress provides a client library for TekExpress clients to use the programmatic interface. The TekExpress client library is inherited from .Net MarshalByRef class to provide the proxy object for the 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:

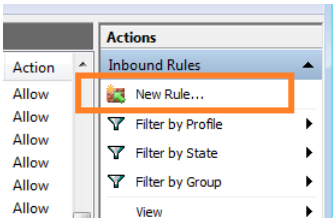
1. Access the Windows Control Panel and open the Windows Firewall tool (**Start > Control Panel > All Control Panel Items > Windows Firewall**).
2. Click **Advance Settings > Inbound Rules**.
3. Scroll through the **Inbound Rules** list to see if the following items (or with a similar name) are shown: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

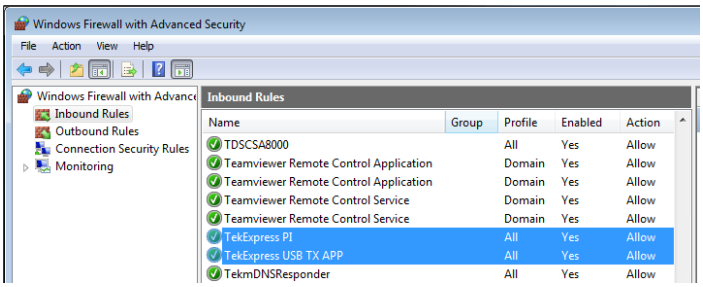


2. Verify that **Program** is selected in the Rule Type panel and click **Next**.
3. Click **Browse** in the Program panel and navigate to and select one of the following TekExpress applications (depending on the one for which you need to create a rule):
4. TekExpress USB3 Tx.exe
5. TekExpress.exe

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 Tx application could be **TekExpress USB3 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 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:

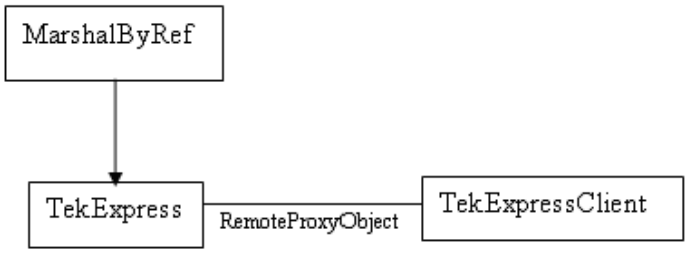
Develop a client UI to access the interfaces exposed through the server. This client loads TekExpressClient.dll to access the interfaces. After TekExpressClient.dll is loaded, the client UI can call the specific functions to run the operations requested by the client. When the client is up and running,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

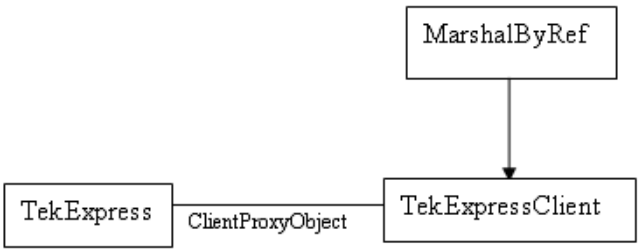
This object lets the remote client access the interfaces exposed at the server side. The client gets the reference to this object when the client gets connected to the server.

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:

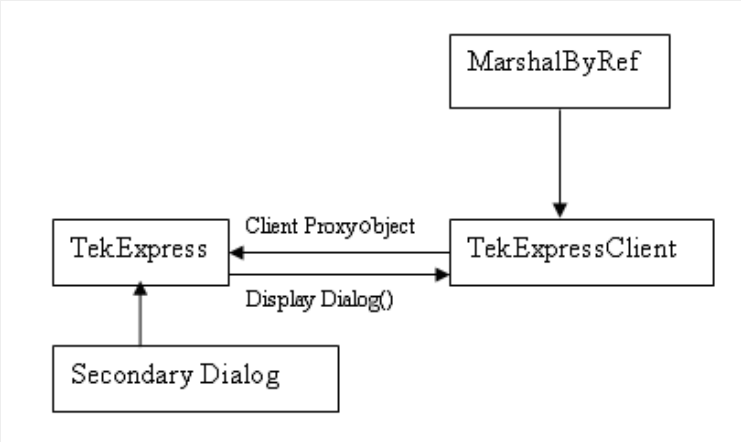
- To get the secondary dialog messages from the server.
- To get the file transfer commands from the server while transferring the report.

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.

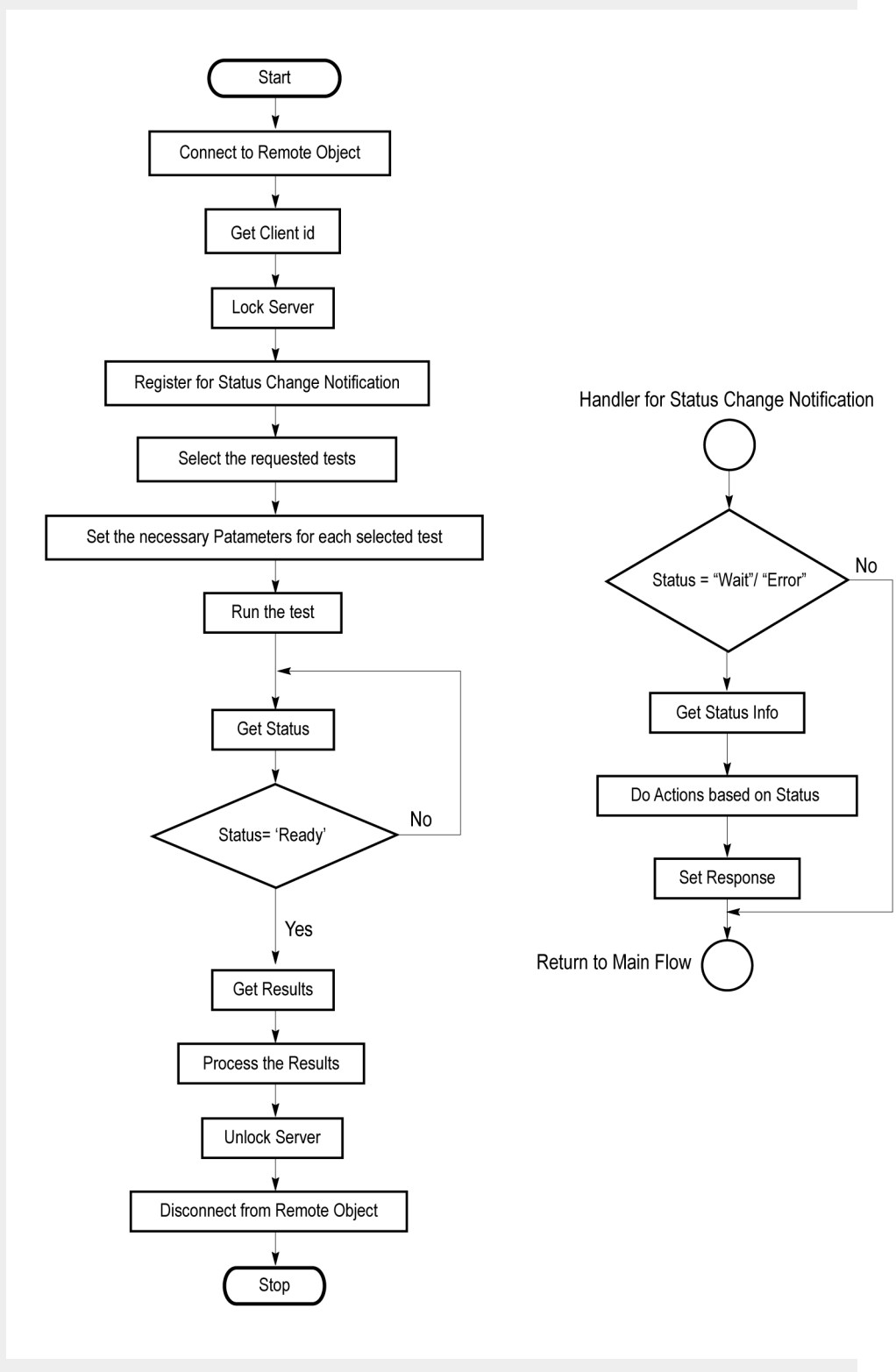
File Transfer Events

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 Tx.

Table 15: 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

Command list

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

SelectVersions()

SelectVersions(clientID, device, suite, version). Selects the version of the specified 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
version	string[ ]	IN	An array containing the versions of the specified suite

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.

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

in string clientId example

clientId = <client_id_number>-<client_IP_address>.

For example, 1065–192.157.98.70

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 panel 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, path). 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

The SetGeneralParameter command

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

paramString argument values. Use the following links to see the paramString values associated with specific application settings.

Select CTLE filter file

Select de-embed filter file

Select embed filter file

Select test method

Select test point

Set A_{DC} for CTLE

Set AFG frequency

Set AFG number of cycles

Set AFG voltage level high

Set AFG voltage level low

Set Auto Recovery mode

Set bandwidth for LFPS acquisition

Set CM Measurement TriMode Probe mode

Set compliance test mode

Set CTLE filter mode

Set de-embed filter mode

Set embed filter mode

Set LFPS hysteresis level

Set LFPS mid edge reference level

Set LFPS trigger level

Set LFPS trigger lower limit

Set LFPS trigger upper limit

Set probing configuration

Set record length

Set SDLA CTLE mode

Set signal pattern validation mode

Set SSC mode

Set verify toggle mode

paramString values for SetGeneralParameter command

Select CTLE filter file

Use this paramString value to select the CTLE filter file to embed for analysis. This is the same as selecting a file from the **CTLE** control on the **DUT** tab.

Custom filter files must be in the same directory as the application-provided filter files.

The value in bold font is the default value.

Values:.

- **Compliance (TP1) - Far End - CTLE Filter File Path$[USB3CTLE.flt.flt** | custom_file_name.flt]
- Tx Pins - Near End - CTLE Filter File Path$[USB3CTLE.flt | custom_file_name.flt]
- Custom - CTLE Filter File Path$[USB3CTLE.flt | custom_file_name.flt]

Example. SetGeneralParameter(clientId, device, deviceConnector, "", "Compliance (TP1) - Far End - CTLE Filter File Path$USB3CTLE.flt");

SetGeneralParameter(clientId, device, deviceConnector, "", “Compliance (TP1) - Far End - CTLE Filter File Path$SpecialTestCase.flt”);

Select embed filter file

Use this paramString value to select the filter file to embed for analysis. This is the same as selecting a file from the **Embed** control on the **DUT** tab.

Custom filter files must be in the same directory as the application-provided filter files.

The value in bold font is the default value.

Values (USB3 Gen1):.

- **Compliance (TP1) - Far End - Embed Filter File Path$ [Tx_Device_TF_8G.flt** | custom_file_name.flt]
- Tx Pins - Near End - Embed Filter File Path$[Tx_Device_TF_8G.flt | custom_file_name.flt]
- Custom - Embed Filter File Path$[Tx_Device_TF_8G.flt | custom_file_name.flt]

Values (USB3 Gen2):.

- **Compliance (TP1) - Far End - Gen2 Embed Filter File Path$ [USBSSP_Embed-13dB.flt** | custom_file_name.flt]
- Tx Pins - Near End - Gen2 Embed Filter File Path$ [USBSSP_Embed-13dB.flt | custom_file_name.flt]
- Custom - Gen2 Embed Filter File Path$[USBSSP_Embed-13dB.flt | custom_file_name.flt]

Example (USB3 Gen1). SetGeneralParameter(clientId, device, deviceConnector, "", “Compliance (TP1) - Far End - Embed Filter File Path $Tx_Device_TF_8G.flt”);

SetGeneralParameter(clientId, device, deviceConnector, "", “Compliance (TP1) - Far End - Embed Filter File Path$SpecialTestCase.flt”);

Example (USB3 Gen2). SetGeneralParameter(clientId, device, deviceConnector, "", “Compliance (TP1) - Far End - Gen2 Embed Filter File Path $USBSSP_Embed-13dB.flt”);

Select test method

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

The value in bold font is the default value.

Values:.

- Test Tool$USB-IF Software (SigTest)
- **Test Tool$DPOJET**
- Test Tool$Both

Example. SetGeneralParameter(clientId, Device, Device Connector, "", "Test Tool$USB-IF Software (SigTest)")

SetGeneralParameter(clientId, Device, Device Connector, "", "Test Tool$Both");

Select test point

Use this paramString value to set the DUT test point used by the application. This is the same as selecting the Test Point control on the DUT tab.

The value in bold font is the default value.

Values:.

- **Version$Compliance (TP1) - Far End**
- Version$Tx Pins - Near End
- Version$Custom

Example. SetGeneralParameter(clientId, Device, Device connector, "", "Version $Compliance (TP1) – Far End");

SetGeneralParameter(clientId, Host, Host Connector, "", "Version$Tx Pins – Near End");

Set SDLA CTLE mode

Use this paramString value to select fixed A_{DC} value for CTLE or to let the TekExpress USB3 Tx application choose the optimized A_{DC} value with maximum eye area.

The value in bold font is the default value.

Values:. Gen2 Ctle Option$[Optimize | Fixed]

Example. SetGeneralParameter(clientId, device, deviceConnector, "", “Gen2 Ctle Option$Optimize”);

Select de-embed filter file

Use this paramString value to select the de-embed file to use by the application. This is the same as selecting a file from the **De-Embed** control on the **DUT** tab.

Custom filter files must be in the same directory as the application-provided filter files.

The value in bold font is the default value.

Values (USB3 Gen1):.

- **Compliance (TP1) - Far End - Deembed Filter File Path$ [Tx_Device_TF_8G.flt** | custom_file_name.flt]
- Tx Pins - Near End - Deembed Filter File Path$[Tx_Device_TF_8G.flt | custom_file_name.flt]
- Custom - Deembed Filter File Path$[Tx_Device_TF_8G.flt | custom_file_name.flt]

Values (USB3 Gen2):.

- **Compliance (TP1) - Far End - Gen2 Deembed Filter File Path$[SSP_De-embed_Tx_Device.flt** | custom_file_name.flt]
- Tx Pins - Near End - Gen2 Deembed Filter File Path$[SSP_De-embed_Tx_Device.flt | custom_file_name.flt]
- Custom - Gen2 Deembed Filter File Path$[SSP_De-embed_Tx_Device.flt | custom_file_name.flt]

Example (USB3 Gen1). SetGeneralParameter(clientId, device, deviceConnector, "", “Compliance (TP1) - Far End - Deembed Filter File Path $Tx_Device_TF_8G.flt”);

SetGeneralParameter(clientId, device, deviceConnector, "", “Compliance (TP1) - Far End - Deembed Filter File Path$SpecialTestCase.flt”);

Example (USB3 Gen1). SetGeneralParameter(clientId, device, deviceConnector, "", “Compliance (TP1) - Far End - Gen2 Deembed Filter File Path$SSP_De-embed_Tx_Device.flt”);

Set SSC mode

Use this paramString value to set the enable or disable the Spread Spectrum Clocking (SSC) mode used by the application for supported DUTs. This is the same as selecting the **Spread Spectrum Clocking** control on the **DUT** tab.

The value in bold font is the default value.

Values:.

- **SSC On$true**
- SSC On$false

Example. SetGeneralParameter(clientId, Device, Device Connector, "", "SSC On $true");

SetGeneralParameter(clientId, Device, Device Connector, "", "SSC On$false");

Set embed filter mode

Use this paramString value to enable or disable filter embedding. This is the same as selecting the **Embed** check box on the DUT tab.

The value in bold font is the default value.

Values (USB3 Gen1):.

- **Compliance (TP1) - Far End - Embed Filter Option$[true** | false]
- Tx Pins - Near End - Embed Filter Option$[true | false]
- Custom - Embed Filter Option$[true | false]

Values (USB3 Gen2):.

- **Compliance (TP1) - Far End - Gen2 Embed Filter Option$[true** | false]
- Tx Pins - Near End - Gen2 Embed Filter Option$[true | false]
- Custom - Gen2 Embed Filter Option$[true | false]

Example (USB3 Gen1). SetGeneralParameter(clientId, device, deviceConnector, "", "Compliance (TP1) - Far End - Embed Filter Option$true”);

SetGeneralParameter(clientId, device, deviceConnector, "", "Custom - Embed Filter Option$false”);

Example (USB3 Gen2). SetGeneralParameter(clientId, device, deviceConnector, "", "Compliance (TP1) - Far End - Gen2 Embed Filter Option$true”);

SetGeneralParameter(clientId, device, deviceConnector, "", "Custom - Gen2 Embed Filter Option$false”);

Set CTLE filter mode

Use this paramString value to enable or disable CTLE filter embedding. This is the same as selecting the **CTLE** check box on the DUT tab.

The value in bold font is the default value.

Values (USB3 Gen1):.

- **Compliance (TP1) - Far End - CTLE Filter Option$[true** | false]
- Tx Pins - Near End - CTLE Filter Option$[true | false]
- Custom - CTLE Filter Option$[true | false]

Values (USB3 Gen2):.

- **Compliance (TP1) - Far End - Gen2 CTLE Filter Option$[true** | false]
- Tx Pins - Near End - Gen2 CTLE Filter Option$[true | false]
- Custom - Gen2 CTLE Filter Option$[true | false]

Example (USB3 Gen1). SetGeneralParameter(clientId, device, deviceConnector, "", "Compliance (TP1) - Far End - CTLE Filter Option$true”);

SetGeneralParameter(clientId, device, deviceConnector, "", "Custom - CTLE Filter Option$false”);

Example (USB3 Gen2). SetGeneralParameter(clientId, device, deviceConnector, "", "Compliance (TP1) - Far End - Gen2 CTLE Filter Option$true”);

SetGeneralParameter(clientId, device, deviceConnector, "", "Custom - Gen2 CTLE Filter Option$false”);

Set probing configuration

Use this paramString value to set the application probing options.

The value in bold font is the default value.

Values (probe type):. Probe Type: DUT Signal Probing$[Single Ended | Differential]

Values (differential):. Data$[CH1 | CH2 | CH3 | CH4]

Values (single ended):. Data+$[CH1 | CH2 | CH3 | CH4]

Data-$[CH3 | CH2 | CH1 | CH4]

Example (differential). SetGeneralParameter(clientId, device, deviceConnector, "", “Probe Type: DUT Signal Probing$Differential”);

SetGeneralParameter(clientId, device, deviceConnector, "", “Data$CH1”);

Example (single ended). SetGeneralParameter(clientId, device, deviceConnector, "", “Probe Type: DUT Signal Probing$Single Ended”);

SetGeneralParameter(clientId, device, deviceConnector, "", “Data+$CH1”);

SetGeneralParameter(clientId, device, deviceConnector, "", “Data-$CH3”);

Set bandwidth for LFPS acquisition

Use this paramString value to set the bandwidth (in GHz) used to acquire the LFPS signal. This is the same as selecting the **Acquisition Bandwidth (GHz)** control on the Configuration tab.

The value in bold font is the default value.

Values:.

- **Bandwidth for LFPS acquisition (GHz)$[5** | 6 | 7 | 8 | 9 | 10 | 11 | 12]

Example. SetGeneralParameter(clientId, device, deviceConnector, "", "Bandwidth for LFPS acquisition (GHz)$5");

Set CM measurement TriMode probe mode

Use this paramString value to enable CM measurement using a Tektronix TriMode™ probe. This is the same as selecting the **CM measurement using a TriMode™ Probe with DPOJET Test Method** control on the **Configuration** tab.

The value in bold font is the default value.

NOTE. *This command is for Gen1 only.*

Values:.

- **CM measurement using Trimode probe$[Yes** | No]

Example. SetGeneralParameter(clientId, device, deviceConnector, "", "CM measurement using Trimode probe$Yes");

SetGeneralParameter(clientId, device, deviceConnector, "", "CM measurement using Trimode probe$No");

Set compliance test mode

Use this paramString value to set the compliance test mode (Compliance or User Defined).

The value in bold font is the default value.

Values:.

- **Test Mode$[Compliance** | User Defined]

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

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

Set auto recovery mode

Use this paramString value to enable or disable auto recovery settings mode.

The value in bold font is the default value.

Values:.

- **Auto Recovery Settings$[Yes** | No]

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

Set verify toggle mode

Use this paramString value to set the Verify toggle status. This is the same as selecting the **Verify Toggle Status** control on the **Global Settings** tab of the **Configuration** panel. This is available only when AWG is the toggle tool selected.

The value in bold font is the default value.

Values:.

- **Verify Toggle Status$[Yes** | No]

Example. SetGeneralParameter(clientId, device, deviceConnector, "", "Verify Toggle Status$Yes");

Set LFPS trigger lower limit

Use this paramString value to set the LFPS signal width trigger lower limit (ns). This is the same as selecting the **Width Trigger Lower limit (ns)** control in the LFPS area of the **Configuration** tab.

The value in bold font is the default value.

Values:.

- **LFPS Width Trigger Lower limit (ns)$[10** | user_entered_value]

Example. SetGeneralParameter(clientId, device, deviceConnector, "", "LFPS Width Trigger Lower limit (ns)$12");

Set LFPS trigger upper limit

Use this paramString value to set the LFPS signal width trigger upper limit (ns). This is the same as selecting the **Width Trigger Upper limit (ns)** control in the LFPS area of the **Configuration** tab.

The value in bold font is the default value.

Values:.

- **LFPS Width Trigger Upper limit (ns)$[10** | user_entered_value]

Example. SetGeneralParameter(clientId, device, deviceConnector, "", "LFPS Width Trigger Upper limit (ns)$10");

Set LFPS trigger level

Use this paramString value to set the LFPS signal trigger level (mV). This is the same as selecting the **Trigger Level (mV)** control in the LFPS area of the **Configuration** tab.

The value in bold font is the default value.

Values:.

- **LFPS Trigger level (mV)$[140** | user_entered_value]

Example. SetGeneralParameter(clientId, device, deviceConnector, "", "LFPS Trigger level (mV)$140");

Set LFPS mid edge reference level

Use this paramString value to set the LFPS signal mid edge reference level (in mV). This is the same as selecting the **Mid Edge Ref Level (mV)** control in the LFPS area of the **Configuration** tab.

The value in bold font is the default value.

Values:.

- **LFPS Mid Edge Ref Level (mV)$[140** | user_entered_value]

Example. SetGeneralParameter(clientId, device, deviceConnector, "", "LFPS Mid Edge Ref Level (mV)$140");

Set hysteresis level

Use this paramString value to set the signal hysteresis level (in mV). This is the same as selecting the **Hysteresis Level (mV)** control in the LFPS area of the **Configuration** tab.

The value in bold font is the default value.

Values:.

- **Hysteresis Level (mV)$[50** | user_entered_value]

Example. SetGeneralParameter(clientId, device, deviceConnector, "", "Hysteresis Level (mV)$50");

Set AFG number of cycles

Use this paramString value to set the AFG number of cycles per second value. This is the same as selecting the **Num of Cycles** control in the AFG area of the **Configuration** tab.

The value in bold font is the default value.

Values:.

- **AFG Num of Cycles per Second$[2** | user_entered_value]

Example. SetGeneralParameter(clientId, device, deviceConnector, "", "AFG Num of Cycles per Second$2");

Set AFG frequency

Use this paramString value to set the AFG frequency in MHz. This is the same as selecting the **Frequency (MHz)** control in the AFG area of the **Configuration** tab.

The value in bold font is the default value.

Values:.

- **AFG Frequency (MHz)$[20** | user_entered_value]

Example. SetGeneralParameter(clientId, device, deviceConnector, "", "AFG Frequency (MHz)$20");

Set AFG voltage level high

Use this paramString value to set the AFG voltage level high (in volts). This is the same as selecting the **Voltage Level High** control in the AFG area of the **Configuration** tab.

The value in bold font is the default value.

Values:.

- **AFG Voltage Level High (V)$[0.5** | user_entered_value]

Example. SetGeneralParameter(clientId, device, deviceConnector, "", "AFG Voltage Level High (V)$0.5");

Set AFG voltage level low

Use this paramString value to set the AFG voltage level low (in volts). This is the same as selecting the **Voltage Level Low** control in the AFG area of the **Configuration** tab.

The value in bold font is the default value.

Values:.

- **AFG Voltage Level Low (V)$[-5** | user_entered_value]

Example. SetGeneralParameter(clientId, device, deviceConnector, "", "AFG Voltage Level Low (V)$-5");

Set signal pattern validation mode

Use this paramString value to set the signal patern validation mode. This is the same as using the **Signal Validation** control on the **Configuration** tab.

The value in bold font is the default value.

Values:.

- **Pattern Validation$[Turn Off Signal Check** | Prompt me if Signal Check Fails]

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

Set de-embed filter mode

Use this paramString value to enable or disable de-embedding filter files. This is the same as selecting the **De-Embed** check box on the DUT tab.

The value in bold font is the default value.

Values (USB3 Gen1):.

- **Compliance (TP1) - Far End - Deembed Filter Option$[true** | false]
- Tx Pins - Near End - Deembed Filter Option$[true | false]
- Custom - Deembed Filter Option$[true | false]

Values (USB3 Gen2):.

- **Compliance (TP1) - Far End - Gen2 Deembed Filter Option$[true** | false]
- Tx Pins - Near End - Gen2 Deembed Filter Option$[true | false]
- Custom - Gen2 Deembed Filter Option$[true | false]

Example (USB3 Gen1). SetGeneralParameter(clientId, device, deviceConnector, "", "Compliance (TP1) - Far End - Deembed Filter Option$true”);

SetGeneralParameter(clientId, device, deviceConnector, "", "Custom - Deembed Filter Option$false”);

Example (USB3 Gen2). SetGeneralParameter(clientId, device, deviceConnector, "", "Compliance (TP1) - Far End - Gen2 Deembed Filter Option$true”);

SetGeneralParameter(clientId, device, deviceConnector, "", "Custom - Gen2 Deembed Filter Option$false”);

Set A_{DC} for CTLE

Use this paramString value to select A_{DC} value for CTLE. This option is only enabled when “Gen2 Ctle Option” is set to “Fixed.”

The value in bold font is the default value.

Values:. Gen2 Ctle Index$[6 dB | 0dB | 1dB | 2dB | 3dB | 4dB | 5dB]

Example. SetGeneralParameter(clientId, device, deviceConnector, "", “Gen2 Ctle Index$6dB”);

Set DUT Power Cycle Method

To set 'DUT Power Cycle Method' option we have to use SetIntrument command that s ets the specified instrument as a general configuration parameter to the selected test.

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

Example. mClient.SetInstrument(clientId, "Device", "Device Connector", "UI-Unit Interval", "AnalyzeInstrument$DUT Power Cycle Method$PWS4721 (USB::0x0699::0x0394::006003206573001004::INSTR)")

Set Voltage

Use this paramString value to set the 'Voltage Level' to be used by the application for supported DUTs. This is the same as selecting the **Voltage** in the Configuration, when 'Tektronix Power Supply' is selected as 'DUT Power Cycle Method'

This command is applicable for Device/Host DUT Type when 'Tektronix Power Supply' is selected as 'DUT Power Cycle Method'

The value in bold font is the default value.

Return value. For Device DUT Type: Voltage Level (V)$[5.0 | user_entered_value]

For Host DUT Type: Voltage Level (V)$[0.010 | user_entered_value]

Example. SetGeneralParameter(clientId, "Device", "Device Connector", "", "Voltage Level (V)$4.0")

SetGeneralParameter(clientId, "Host", "Host Connector", "", "Voltage Level (V) $0.009")

Set Current Level

Use this paramString value to set the 'Current Level' to be used by the application for supported DUTs. This is the same as selecting the **Current Level** in the Configuration, when 'Tektronix Power Supply' is selected as 'DUT Power Cycle Method'

Applicable for Device/Host DUT Type when 'Tektronix Power Supply' is selected as 'DUT Power Cycle Method'

The value in bold font is the default value.

Return value. For Device DUT Type: Current Level (A)$[0.9 | user_entered_value]

For Host DUT Type: Current Level (A)$[0.010 | user_entered_value]

Example. SetGeneralParameter(clientId, "Device", "Device Connector", "", "Current Level (A)$0.7")

SetGeneralParameter(clientId, "Host", "Host Connector", "", "Current Level (A) $0.007")

Set AWG DC Output

Use this paramString value to set the 'AWG DC Output' to be used by the application for supported DUTs. This is the same as selecting the **AWG DC Output** in the Configuration, when 'Tektronix AWG' is selected as 'DUT Power Cycle Method'

Applicable for Device/Host DUT Type when 'AWG' is selected as 'DUT Power Cycle Method'.

The value in bold font is the default value.

Return value. AWG DC Output$[DC1 | DC2 | DC3 |DC4]

Example. SetGeneralParameter(clientId, "Device", "Device Connector", "", "AWG DC Output$DC2")

Set Output ON or OFF Time

Use this paramString value to set the 'Output ON or OFF Time' to be used by the application for supported HOST DUTs. This is the same as selecting the **Output ON or OFF Time** in the Configuration, when 'Tektronix AWG / Power Supply' is selected as 'DUT Power Cycle Method' and DUT Type is selected as HOST.

Return value. Output ON to OFF Time (s)$[6 | user_entered_value]

Example. SetGeneralParameter(clientId, "Host", "Host Connector", "", "Output ON to OFF Time (s)$2")

Set Radio Friendly Clocking mode

Use this paramString value to set the enable or disable the Radio Friendly Clocking mode used by the application for supported DUTs. This is the same as selecting the **Radio Friendly Clocking** control on the **DUT** tab, when SSC Mode is set to True.

The value in bold font is the default value.

Return value. Radio Friendly Clocking$true

Radio Friendly Clocking $false

Example. SetGeneralParameter(clientId, "Device", "Device Connector", "", "Radio Friendly Clocking$true")

SetGeneralParameter(clientId, "Device", "Device Connector", "", "Radio Friendly Clocking$false")

Set Toggle Using

To set 'Toggle Using' option, use SetInstrument command that sets the specified instrument as a general configuration parameter to the selected test.

The value in bold font is the default value.

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

Example. mClient.SetInstrument(clientId, "Device", "Device Connector", "UI-Unit Interval", "AnalyzeInstrument$Toggle Using$AWG7122C (GPIB0::3::INSTR)")

Set record length

Use this paramString value to set the acquisition record length for CP0 and CP1 signals. This is the same as selecting the **CP0 CP1 CP7 Record Length** or **CP9 CP10 Record Length** control on the **Configuration** tab.

The value in bold font is the default value.

Values:. Record Length for CP0 CP1 CP7$[10000000 | user_entered_value]

Example. SetGeneralParameter(clientId, device, deviceConnector, "", " Record Length for CP0 CP1 CP7$5000000");

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".

Limits editor: compare string definitions

The following table lists the definitions of the limit comparison strings:

Table 16: Limits Editor: comparison strings

Comparison string	Description
EQ(==)	Equal to
NE(!=)	Not equal to
GT(>)	Greater than
LT(<)	Less than
GE(>=)	Greater than or Equal to
LE(<=)	Less than or Equal to
GTLT(> <)	Greater than and Less than
GELE(>= <=)	Greater than or equal to and Less than or equal to
GELT(>= <)	Greater than or equal to and Less than
GTLE(> <=)	Greater than and Less or equal to
LTGT(< >)	Less than and Greater than
LEGE(<= >=)	Less than or equal to and Greater than or equal to
LEGT(<= >)	Less than or equal to and Greater than
LTGE(< >=)	Less than and Greater than or equal to

De-Embedding and channel embedding information

De-Embedding and channel embedding overview

Use the following filter files to meet the indicated conditions:

- Host + 3m cable (Host_Channel_Back_Panel_3M_Cable_12.5G.flt
- Device + 3m cable (Device_Channel_3M_cable_12.5G.flt)
- Device fixture (Tx_Device_TF_8G.flt)
- Host fixture (TX_Host_TF_8G.flt)
- USB3CTLE (same for Device and Host)
- USB3CTLE_short.flt (same for Device and Host)
- USBSSP_Embed-13dB.flt (for USB3 Gen2 Device and Host)

NOTE. *There is no filter for Host front + 3m cable.*

The first two S-parameter files in the following list are for test fixtures (Device and Host); the next two files are for Reference channels (Device and Host).

- USB-IF_ENA_DEVICE_CHANNEL_3MCABLE.s4p
- USB-IF_ENA_HOST_CHANNEL_3MCABLE.s4p
- INTEL DEVICE FIXTURE_PLUS SHORT CABLE.s4p
- INTEL HOST FIXTURE.s4p
- CABLE_MAX_STDA_STDB.s12p (for USB3 Gen2 Device and Host)

Front Panel and capacitive devices do not use a short cable. As the S-parameter files represent the combined reference channel and short cable parameters, the USB-IF does not provide S-parameter files for these devices. Front Panel and capacitive devices need a back channel with no cable

NOTE. *There is one set of filter files that support both 25 GS/s (8 GHz BW oscilloscopes) and 50 GS/s (12.5 GHz BW and above oscilloscopes) sampling rates. The difference between the 25 GS/s and 50 GS/s filters is that the 25 GS/s filters have a stop band setting of 10 GHz for embed filters.*

See also. *Host Filter Information*

Device Filter Information

DUT/Filter Combinations

Host filter information

Host embed filter.

- The Host embed filter name is Device_Channel_3M_cable_12.5G.flt.
- This filter is applied for HOST DUT and for the normative CP0 and CP1 measurements.
- The test point location is compliance TP1. The application uses the device and cable compliance channels to test the host designs.
- The filter response is generated using the SDLA when the input is set as SigTest (USB-IF) for the USB-IF_ENA_HOST_CHANNEL_3MCABLE.s4p file. The S-parameter file represents the combined response of the HOST reference channel and the 3 meter cable.
- This filter embeds the response of the Host 5 inch reference channel and a 3 meter cable.
- The filter response BW is 12.5 GHz and the stop band is 15.625 GHz at an -80 dB roll off.

Host de-embed filter. The Host de-embed filter file is Tx_Host_TF_8G.flt, which de-embeds the Host USB-IF test fixture. This filter is convolved with the Device_Channel_3M_cable_12.5G.flt filter to remove the effects of the test fixture.

The application uses the 'Intel Host fixture.s4p' S-parameter file to generate the filter response used by USB-IF.

The filter response BW is 8 GHz and the stop band is 10 GHz at an -80 db roll off.

The following is a representative plot, generated using SDLA. You can use this plot as a reference plot, or to view the response of the filter file.

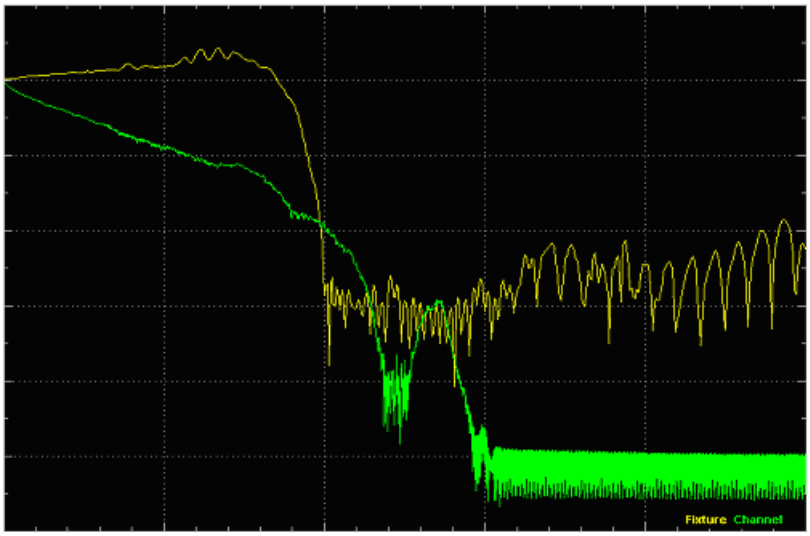


Figure 1: Host fixture and channel response

Continuous time linear equalizer (CTLE) filter. The USB3CTLE.flt filter is a Continuous Time Linear Equalizer (CTLE) filter. Due to the lossy nature of the channel (the combination of the reference channel, cable, and test fixture from TX pins(TP1)), the eye diagram at the receiver may be closed. This filter applies receiver equalization to meet the system timing and voltage margins.

The CTLE filter coefficients are generated by passing the following parameters to the SDLA:

- DC gain (A_{DC}) = 0.667
- Zero frequency (fz) = 650 MHz
- first pole frequency (fp1) = 1.95 Ghz
- Second pole frequency (fp2) = 5 GHz

The following is a representative plot, using SDLA, of the combined response of the convolution of Embed, De-embed and CTLE filters. You can use this plot as a reference plot, or to view the response of the filter file.

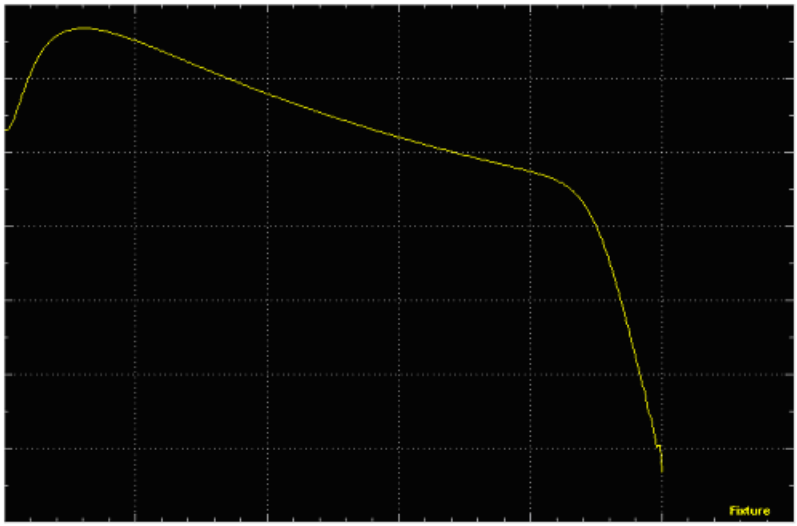
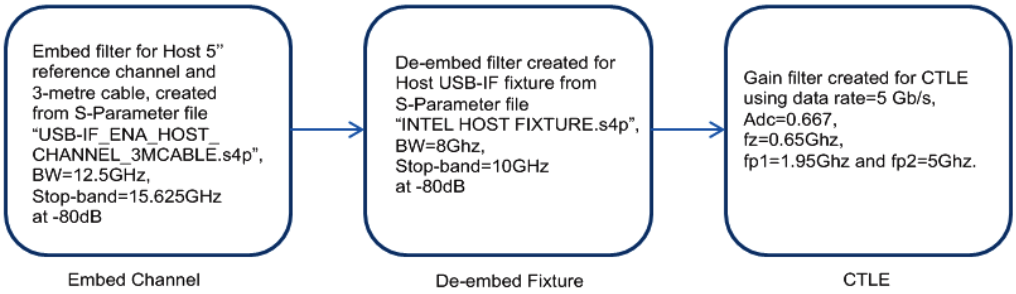


Figure 2: CTLE response

The USB3CTLE_short filter coefficients are generated by passing the following parameters to the SDLA:

- DC gain (A_{DC}) = 1 Zero frequency (fz) = 650 MHz
- First pole frequency (fp1) = 650 MHz
- Second pole frequency (fp2) = 10 GHz

The signal connection path for HOST is HOST <-> TF (B connector) <-> 3 meter reference cable <-> standard (1 meter) SMA cable <-> Oscilloscope channel.



NOTE. *These filters are applicable for 12.5 GHz and above oscilloscopes. For 8 GHZ oscilloscopes, only the embed filters BW (8 GHz) and stop band (10 GHz) are different; the de-embed parameters remain the same.*

See also. *De-Embedding and Channel Embedding Overview*

Device Filter Information

DUT/Filter Combinations

Device filter information

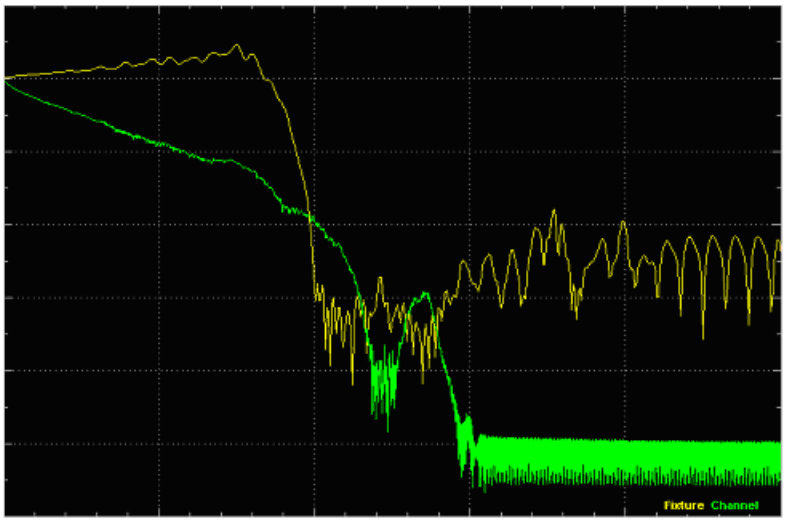
Device embed filter.

- The embed filter file is Host_Channel_Back_Panel_3M_Cable_12.5G.flt, and is applied for the Device DUT for the normative (CP0 and CP1) measurements.
- The test point location is compliance TP1.
- The application uses host and cable compliance channels for testing of device designs.
- This filter response is generated using the Tektronix SDLA filter generation application, with input from the USB-IF S-parameter file (USB-IF_ENA_DEVICE_CHANNEL_3MCABLE.s4p).
- This filter embeds the response of the Device 11 inch reference channel and a 3 meter cable.
- The filter response BW is set to 12.5 GHz and the stop band is set to 15.625 GHz at an -80 dB roll off.

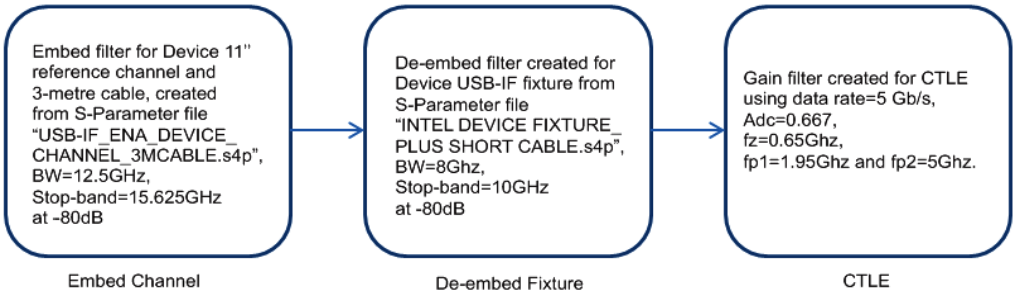
Device de-embed filter.

- The de-embed filter is Tx_Device_TF_8G.flt, and is applicable for Host.
- This is convolved with the Host_Channel_3M_cable_12.5G.flt filter to remove the test fixture effects.
- The filter response is generated using the 'Intel Device fixture.s4p' filter output provided by USB-IF.
- The filter de-embeds the Device USB-IF test fixture.
- The filter response BW is set to 8 GHz and the stop band is set to 10 GHz at an -80 db roll off.

The following is a representative plot, using SDLA, of the combined response of the convolution of Embed, De-embed and CTLE filters. You can use this plot as a reference plot, or to view the response of the filter file.



The connection path for DEVICE is DEVICE <-> Short USB cable (4 inch) <-> TF (A connector) <-> 3 meter reference cable <-> TF_convertor* <-> standard (1m) SMA cable <-> Oscilloscope channel.



NOTE. *The TF_convertor uses input from the 3 meter USB cable and provides output as SMA. This convertor is needed for testing the far end of the DUT.*

NOTE. *These filters are applicable for 12.5 GHz and above oscilloscopes. For 8 GHZ oscilloscopes, only the embed filters BW (8 GHz) and stop band (10 GHz) are different; the de-embed parameters remain the same.*

See also. *De-Embedding and Channel Embedding Overview*

Host Filter Information

DUT/Filter Combinations

USB3 gen2 filter information

The embed filter file for USB3 Gen2 is USBSSP_Embed-13dB.flt, and is applied for the Device/Host DUT for the normative (CP9 and CP10) measurements. The test point location is compliance TP1. The filter has an attenuation of -13 dB at 5 GHz.

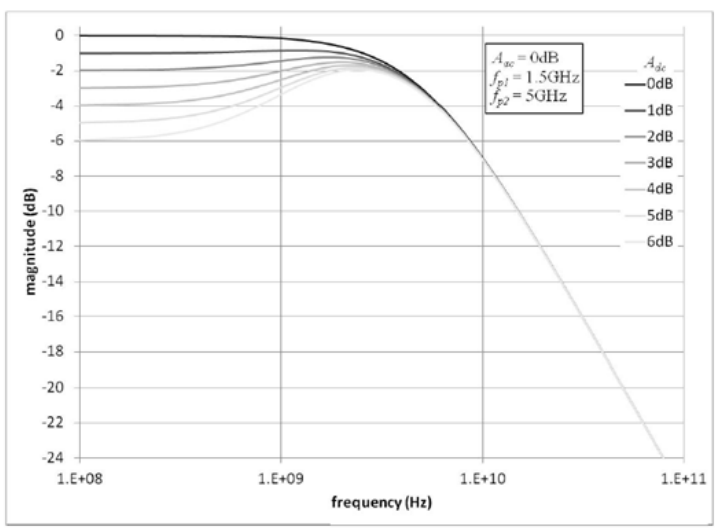
Continuous time linear equalizer (CTLE) and decision-feedback equalizer (DFE).
The frequency response for the Gen2 reference CTLE that is used for compliance testing is shown in the following equation:

$$H(s) = A_{ac}\omega_{p2}\frac{s + \frac{A_{dc}}{A_{ac}}\omega_{p1}}{(s+\omega_{p1})(s+\omega_{p2})}$$

Where:

- A*ac* is the high frequency peak gain
- A*dc* is the DC gain
- $\omega_{p1} = 2\pi fp1$ is the first pole frequency
- $\omega_{p2} = 2\pi fp2$ is the second pole frequency

The following plot shows the frequency response:



The DFE behavior for Gen2 is described by the equation:

$Yk = Xk - d1\mathrm{sgn}(Yk\text{-}1)$

Where:

y_k is the DFE differential output voltage

y^*_k is the decision function output voltage $|y^*k| = 1$

x_k is the DFE differential input voltage

d_1 is the DFE feedback coefficient

k is the sample index in UI

Several SDLA setup files are located “C:\Program Files (x86)\Tektronix \TekExpress\TekExpress USB3 Tx\Setup Files\SDLA Setups” for performing equalization for Gen2 signals.

Each setup file corresponds to a different A_{DC} gain.

The following table shows the inputs for Gen2 SDLA CTLE:

Table 17: SDLA CTLE input parameters (Gen2)

dB	Adc	Fz (GHz)	Fp1 (GHz)	Fp2 (GHz)
0	1	1.5	1.5	5
-1	0.891251	1.336876	1.5	5
-2	0.794328	1.191492	1.5	5
-3	0.707946	1.061919	1.5	5
-4	0.630957	0.946436	1.5	5
-5	0.562341	0.843512	1.5	5
-6	0.501187	0.751781	1.5	5

DUT-Filter combinations

Use the following table to help select the appropriate filter for listed DUT and embed/de-embed configurations.

Table 18: USB3 Gen1 Host/Device DUT embed, de-embed filter combinations for USB3 Gen1

DUT	Test point (TP) location	Embed filter (reference channel + cable effects)	De-embed filter (test fixture effects)	USB IF recommendation
Host	Compliance TP1	N/A	N/A	Informative
Host	Compliance TP1	Device_Channel_3M_Cable_12.5G.flt (default)	Tx_Host_TF_8G.flt	Normative
Host	Tx Pins (host connector)	N/A	N/A	Informative
Host	Custom	Custom_Device_Channel_3M_Cable.flt	Custom_Host_TF.flt	Informative
Device	Compliance TP1	N/A	N/A	Informative
Device	Compliance TP1	Host_Channel_Back_Panel_3M_Cable_12.5Gflt (default)	Tx_Device_TF_8G.flt	Normative
Device	Tx Pins (device connector)	N/A	N/A	Informative
Device	Custom	Custom_Host_Channel_3M_Cable.flt	Custom_Device_TF.flt	Informative
Device	Far end (TP1)	Tx_Device_Channel_3M_Cable.flt	Tx_Device_TF.flt	Normative

Table 19: USB3 Gen2 Host/Device DUT embed, de-embed filter combinations

DUT	Test point (TP) location	Embed filter (reference channel + cable effects)	De-embed filter (test fixture effects)	USB IF recommendation
Host	Compliance TP1 - Far End	USBSSP_Embed-13dB.flt	SSP_De-embed_Tx_Host.flt	Normative
Host	Tx Pins - Near End	N/A	N/A	Informative
Host	Custom	N/A	N/A	Informative
Device	Compliance TP1 - Far End	USBSSP_Embed-13dB.flt	SSP_De-embed_Tx_Device.flt	Normative
Device	Tx Pins - Near End	N/A	N/A	Informative
Device	Custom	N/A	N/A	Informative

See also. *De-Embedding and Channel Embedding Overview*

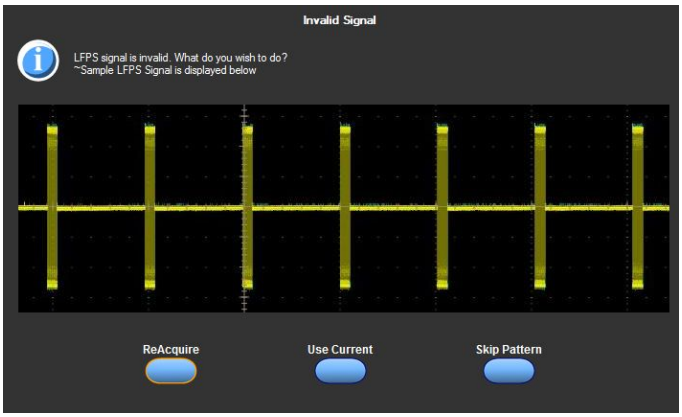
Host Filter Information

Device Filter Information

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.*

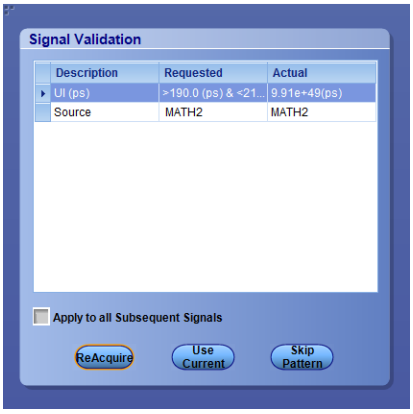
■ Click **Reacquire** to start the acquisition again.

■ 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.

CP0 CP1 CP7 toggle mechanisms

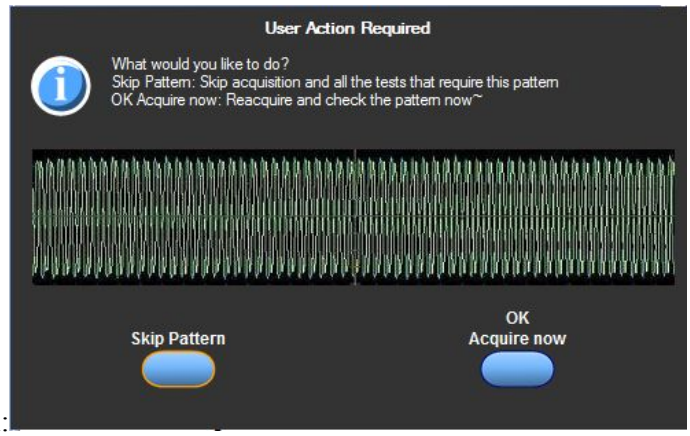
Oscilloscope-based toggle

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.

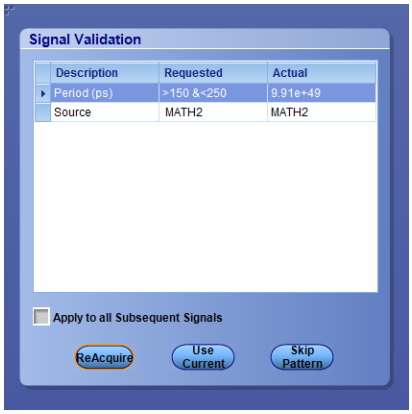
For CP1 signal:



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*

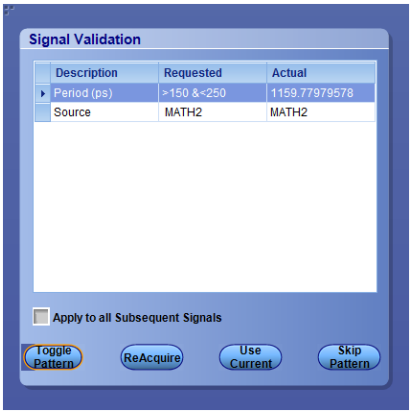
AFG-based toggle

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*

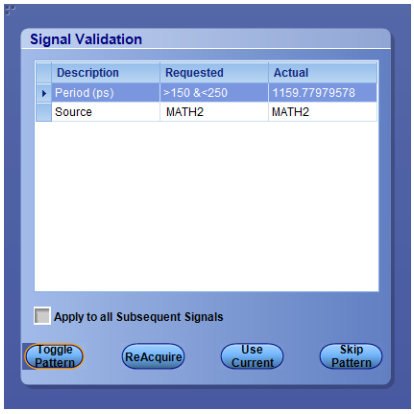
AFG-based toggle

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:

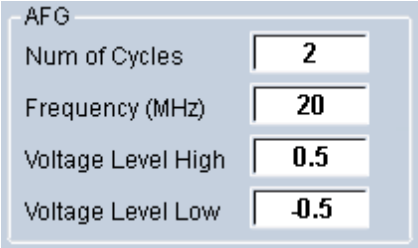


NOTE. *If Pattern type validation is set to No, then the measurement continues with the acquired waveform.*

---

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

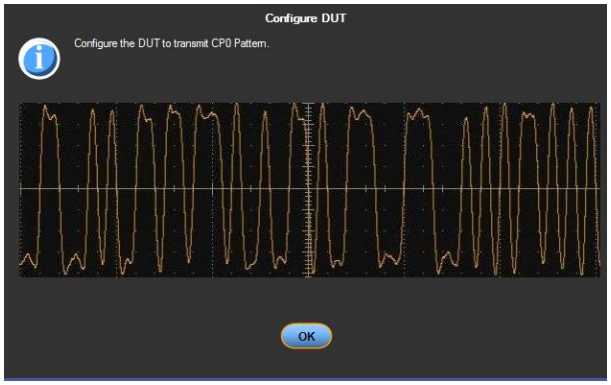
Manual toggle

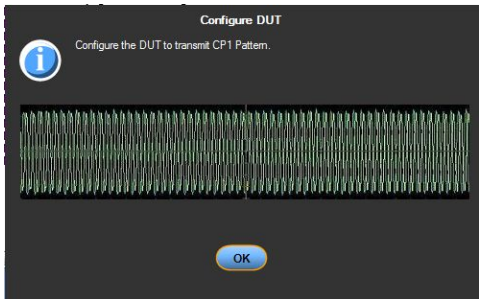
Manual toggle

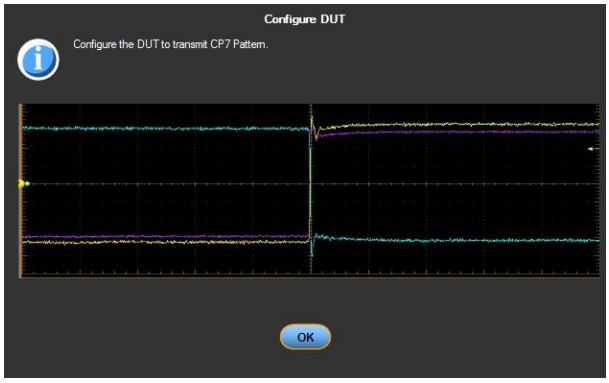
To not use the AWG-based toggle capability, do the following:

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.

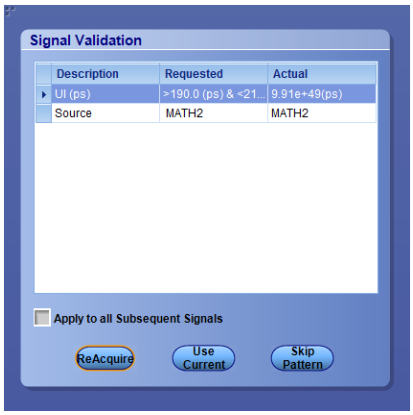


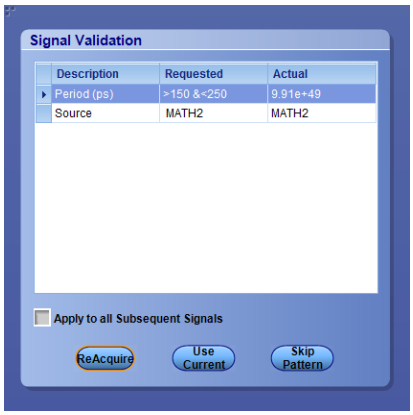


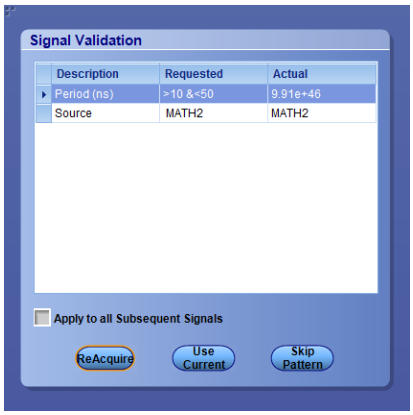


4. Click **OK** to acquire the waveform. If Pattern type validation is set to **Prompt me if Signal Check Fails**, the application runs a pattern type validation on the acquired signal. If the acquired signal is a valid pattern, the measurement continues normally.

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

   CP0: 

   CP1: 

   CP7: 

   NOTE. *If Pattern type validation is set to Turn Off Signal Check, then the measurement continues with the acquired waveform.*

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

A

Acquire live waveforms, 27
Acquire parameters
    including in test reports, 45
    viewing in reports, 46
Acquisition tab, 33
Activate the USB-TX license, 8
AFG parameters, 131
AFG-based toggle, 130
application directory setup, 9
Application version, 9
AWG-based toggle, 129

B

Bandwidth, 37
Before you click Start, 52

C

Client proxy object, 62
Code example, remote access, 66
Comments, 27
Compliance Mode, 37
Configuration parameters, 37
Configure AFG parameters, 131
Configuring email notifications, 23
Connected instruments
    searching for, 20
Connection requirements, 51
CPx pattern signal validation, 126
CTLE filter, 119

D

De-embed filters (Host, Device)
    Gen1, 124
    Gen2, 124
De-embedding, 117
Deselect All (tests), 31
Deskew
    real time oscilloscopes, 50
Detailed log view, 39
Device embed filter,
Device parameters, 27
Device profile connections, 51
Device profiles, 27
Do Not Use (manual toggle), 131
DUT ID, 27
DUT parameters, 27
DUT type
    device, 27
    host, 27
DUT/Filter combinations table
    Gen1, 124
    Gen2, 124

E

Email notifications, 22, 23
Embed filter information
    Gen2, 122
Embed filters (Host, Device)
    Gen1, 124
    Gen2, 124
Embedding, 117
Enable remote access, 58
Equipment setup, 51
Error notification (email), 22
Evaluation mode, 14
Exiting the application, 14

F

Fail notification (email), 22
File name extensions, 12
Filter/DUT combinations table
    Gen1, 124
    Gen2, 124
Firewall (remote access), 58
Free trials, 14

G

Gen1
    de-embed filters (Host, Device), 124

embed filters (Host, Device), 124
Filter/DUT combinations tables, 124
Gen2
de-embed filters (Host, Device), 124
embed filter information, 122
embed filters (Host, Device), 124
Filter/DUT combinations tables, 124
inputs for Gen2 SDLA CTLE, 124
reference CTLE frequency equation, 123
reference CTLE frequency plot, 123
Global settings, 37

H

Help conventions, 2
Host de-embed filter, 118
Host embed filter,

I

Inbound Rule Wizard (remote access), 58
Initial application directory setup, 9
Inputs for SDLA CTLE (Gen2), 124
Installing the software
TekExpress application for USB-TX, 8
Instruments
discovering connected, 19
viewing connected, 20
Instruments detected, 37
Interface, 57
Interface error codes, 115

K

Keep On Top, 14
Key, 14

L

LFPS pattern signal validation, 125
License, 14
License agreement, 9
Limits Editor, 37
Loading a test setup, 54
Loading saved waveform files, 33, 35
Log view
save file, 39
Log View tab, 39

M

Manual toggle, 131
Measurement limits, 37
Menus, 17
Minimum system requirements, 5
Mode
Compliance, 37
User Defined, 37
My TekExpress folder
files stored in, 43
My TekExpress folder permissions, 9

N

New Inbound Rule Wizard, 58
Notifications (email), 22

O

Opening a saved test setup, 54
Option Installation wizard, 8
Options menu
Instrument control settings, 19
Keep On Top, 14
Oscilloscope-based toggle, 127
Overall test result, 41

P

Panels, 25
Pass/Fail summary
viewing, 46
Pass/Fail Summary
including in reports, 46
Plot images
including in reports, 46
viewing, 46
Preferences menu, 41, 42
Preferences tab, 26, 38
Prerecorded waveform files
selecting run sessions for, 27
Prerun checklist, 52
Program example, 66
Programmatic interface, 57

R

Reactivate the USB-TX license, 8
Real time oscilloscope, 37
Recalling a test setup, 54
Record length, 37
Reference CTLE frequency equation (Gen2), 123
Reference CTLE frequency plot (Gen2), 123
Related documentation, 1
Remote access firewall settings, 58
Remote proxy object, 61
Report name, 45
Report options, 45
Report sections, 46
Reports
    receiving in email notifications, 23
Reports panel, 25, 44
Resource file, 14
Results panel, 41, 42
Run a saved test session, 55

S

Sample Rate, 37
Save log file, 39
Saving test setups, 53
Saving tests, 43
Schematic button, 31
Search for connected instruments, 20
Select All (tests), 31
Selecting test report contents, 45
Selecting tests, 31
Server, 60
Session files, 43
Session folders, 43
Set AFG parameters, 131
Set compliance test mode, 106
set My TekExpress folder permissions, 9
Set remote access, 58
Setting up equipment, 51
Setting up tests, 49
Setup files, 53
Setup panel, 25, 26
Setup panel views, 27
Show MOI button, 31
Signal Path Compensation (SPC), 50
Signal validation
    CPx pattern type, 126
    LFPS pattern type, 125
Software installation
    activate USB-TX license, 8
    TekExpress USB, 8
Software version, 9
Status panel, 39
Support, 2
System requirements, 5

T

Technical support, 2
TekExpress application install for USB-TX, 8
TekExpress client, 57
TekExpress client requirements, 60
TekExpress server, 57
Test completion notification (email), 22
Test groups, 31
Test limits, 116
Test parameters (Configuration tab), 37
Test reports, 46
Test results
    emailing, 23
Test selection controls, 31
Test setup files, 43, 53
Test setup steps, 49
Test setups
    creating, 56
    load, 54
    open, 54
    recalling, 54
    saving, 53
Test Status tab, 39
Test-related files, 43
Tests
    running, 51
    selecting, 31
Toggle
    AFG-based, 130
    AWG based, 129

Do not use selection, 131
manual toggle, 131
oscilloscope-based, 127
set AFG parameters, 131

U

USB-TX license activation, 8
Use saved waveforms to run a test, 55
User account setting (Windows 7), 6
User comments
location in reports, 46
User Comments
including in reports, 46
User Defined Mode, 37

V

Verify application installation, 8

W

Waveform files
locating and storing, 43
Windows 7 user account setting, 6