TekExpress® RIN
Measurement Solution for DSA8300 Digital Serial Analyzer Sampling Oscilloscope

Printable Application Help

077-1020-00

TekExpress® RIN
Measurement Solution for DSA8300 Digital Serial Analyzer Sampling Oscilloscope

Printable Application Help

www.tektronix.com
077-1020-00

Tektronix

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

Getting help and support

Getting started

Operating basics

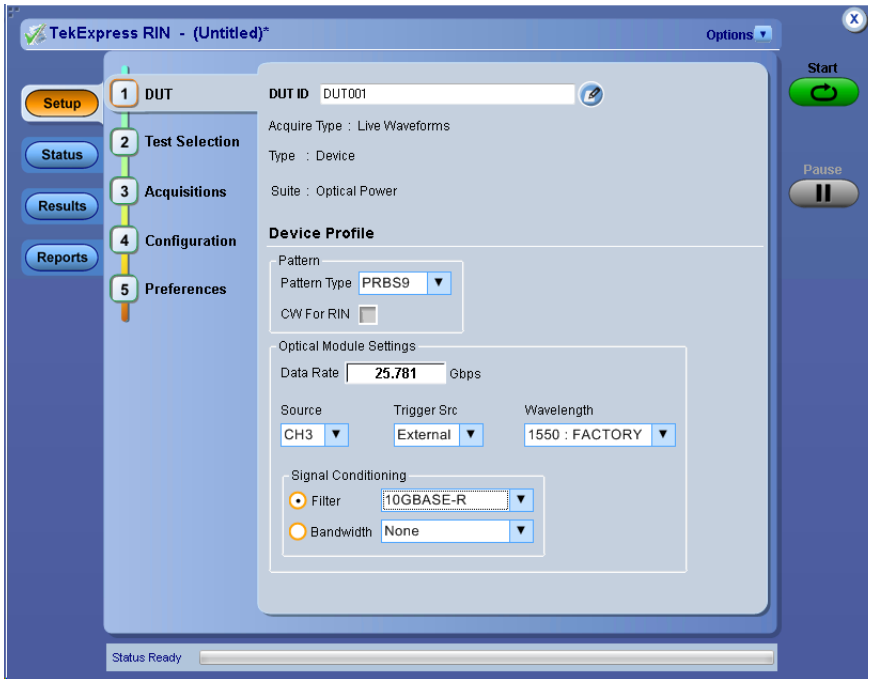
Pre-measurement calibration procedures

Saving and recalling test setup files

TekExpress programmatic interface

Algorithms

Welcome



DUT ID DUT001

Acquire Type : Live Waveforms

Type : Device

Suite : Optical Power

Device Profile

Pattern

Pattern Type PRBS9

CW For RIN □

Optical Module Settings

Data Rate 25.781 Gbps

Source	Trigger Src	Wavelength
CH3	External	1550 : FACTORY

Signal Conditioning

● Filter 10GBASE-R

○ Bandwidth None

Status Ready

Welcome to the TekExpress® RIN Automated Measurement Solution Software application (referred as TekExpress RIN in the rest of the document). TekExpress RIN provides an automated, simple, and efficient way to take RIN (Relative Intensity Noise) and RINxOMA (Optical Modulation Amplitude) measurements on optical signals.

Key features of TekExpress RIN include:

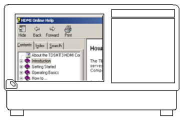
- Take accurate RIN and RINxOMA measurements on PRBS7, PRBS9, PRBS11, PRBS13, PRBS15, 8180, 4140, and 2120 patterns
- Measure RIN on CW signal
- Programmatic interface lets you run scripts for automated RIN testing

Getting help and support

Related documentation

The following documentation is available as part of the TekExpress® RIN Automated Measurement Solution application.

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 RIN software distribution (*TekExpress RIN-Automated-Test-Solution-Software-Printable-Help-EN-US.pdf*).

See also *Technical support*

Conventions used in help

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* at the front of this document for contact 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
- Modules 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

Getting started

Installing the software

Minimum system requirements

The following table shows the minimum system requirements to run TekExpress RIN.

Table 3: System requirements

Component	Description
Oscilloscope	Tektronix DSA8300 Digital Serial Analyzer FW Version: 6.3.1.3 or greater 80SJNB SW Version: 3.2.4.0 or greater Opt ADVTRIG Opt JNB01 80CXX series Optical Sampling module
Processor	Same as the oscilloscope
Operating System	Same as the oscilloscope: ■ Windows 7 ■ *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]
Firmware	■ TekScope 6.3.1.3 or greater (for Windows 7) ■ 80SJNB SW Version: 3.2.4.0 or greater

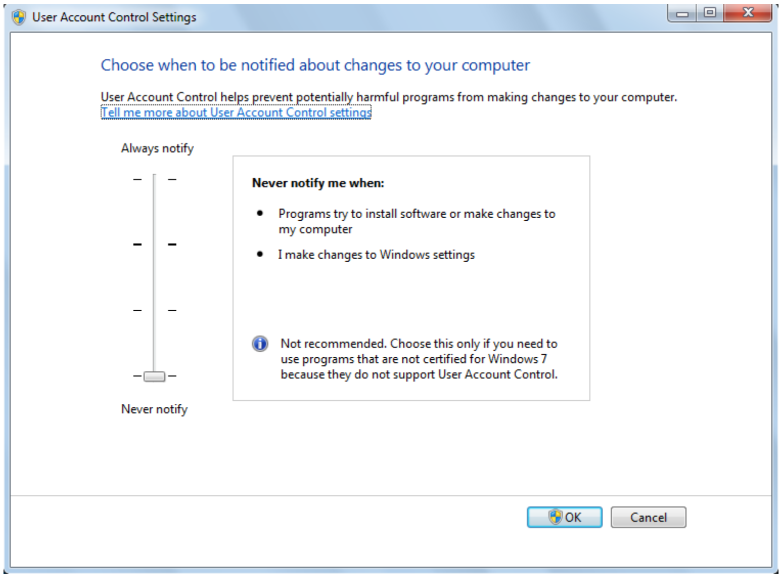
[1] If TekExpress is running on an instrument that has a video resolution less than 800x600, connect and configure a second monitor to the instrument.

Component	Description
Software	■ TekExpress Framework (version 3.0.x or greater) installed. ■ IronPython 2.7.3 installed ■ PyVisa 1.0.0.25 installed ■ Microsoft .NET 4.0 Framework ■ Opt ADVTRIG – Advanced triggers with pattern sync (required for RIN testing) ■ Microsoft Internet Explorer 7.0 SP1 or greater, or other Web browser for viewing reports ■ Adobe Reader software 7.0 or greater for viewing portable document format (PDF) files
Other Devices	■ Microsoft compatible mouse or compatible pointing device. ■ Two USB ports (four USB ports recommended).

Windows 7 user account settings

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

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



User Account Control Settings

Choose when to be notified about changes to your computer

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

Never notify me when:

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

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

Install the software

Use the following steps to obtain the latest TekExpress RIN software from the Tektronix Web site and install on any compatible instrument running Microsoft Windows 7 (32-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 RIN** 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 RIN 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\Tektronix\TekExpress \TekExpress RIN`
7. *Verify application installation*.

See also. *Minimum system requirements*

Required My TekExpress folder settings

Verify application installation

To verify the installation was successful:

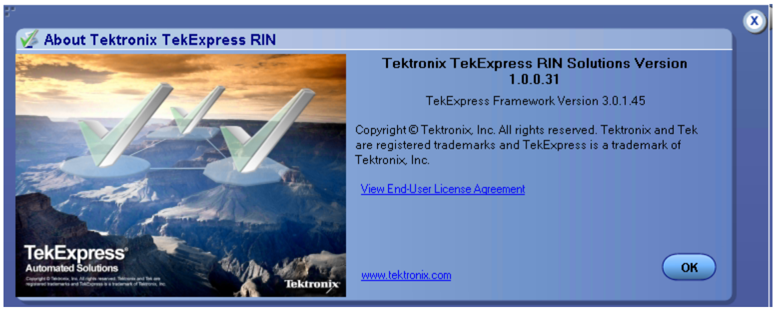
1. Open the TekScope application.
2. Click the **Application** menu.
3. Verify that **RIN** is listed in the Application menu.
4. Click **RIN** to open the TekExpress RIN application. Verify that the application opens successfully.

See also. *Required My TekExpress folder settings*

View software version

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

To view version information for RIN, click the **Options** button in TekExpress and select **About TekExpress**.



About Tektronix TekExpress RIN

Tektronix TekExpress RIN Solutions Version 1.0.0.31

TekExpress Framework Version 3.0.1.45

Copyright © Tektronix, Inc. All rights reserved. Tektronix and Tek are registered trademarks and TekExpress is a trademark of Tektronix, Inc.

View End-User License Agreement

www.tektronix.com

NOTE. *This example shows a typical Version Details dialog box, and may not reflect the actual values as shown when you open this item in the application.*

See also. *Options menu*

Required My TekExpress folder settings

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

1. *Map the My TekExpress folder to drive X*
2. *Set the My TekExpress folder permissions*

See also. *Application directories and usage*

File name extensions

Map the My TekExpress folder to drive X

The first time you run TekExpress RIN, it creates the following folders on the oscilloscope:

- `\My Documents\My TekExpress\RIN`
- `\My Documents\My TekExpress\RIN\Untitled Session`

Shared **My TekExpress** folder is mapped to drive **X:** on the instrument running the RIN application. RIN uses this shared folder to save session waveform files and for other application file transfer operations.

Follow the below procedure to map the My TekExpress folder on the instrument to be drive X:

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

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

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

NOTE. *The My TekExpress folder has the share name format* `<domain><user ID>My TekExpress`*.*

If the instrument is not connected to a domain, the share name format is `<instrument name><user ID>My TekExpress`*.*

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

See also. *Set the My TekExpress folder permissions*

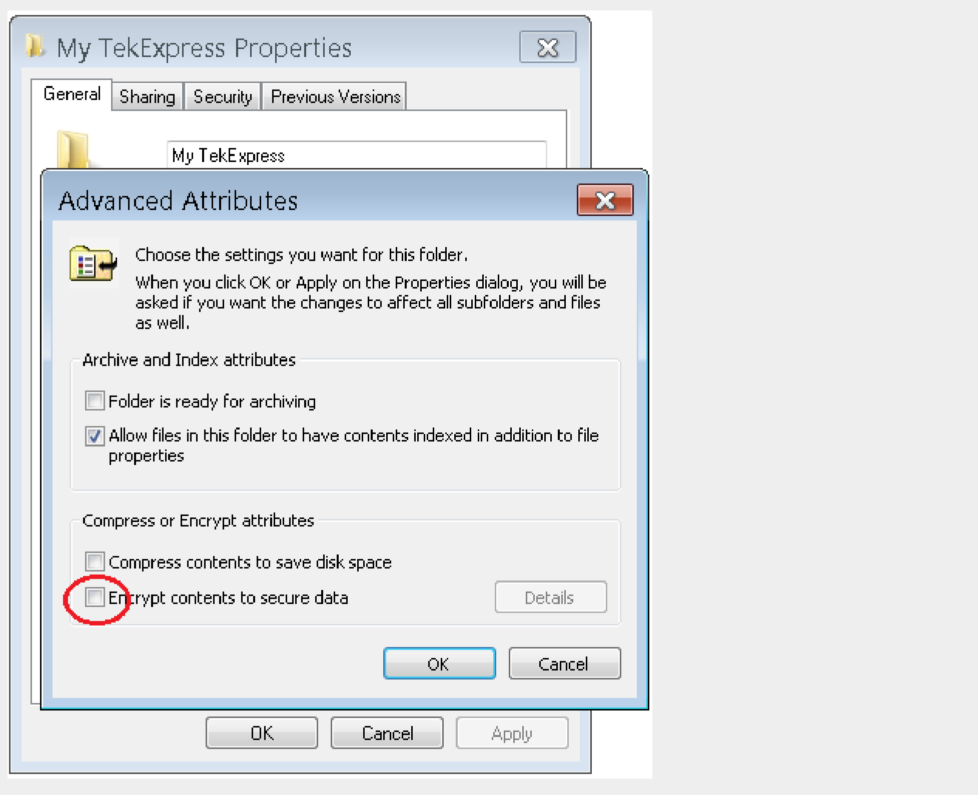
Application directories and usage

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.



My TekExpress

Choose the settings you want for this folder.

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

Archive and Index attributes

☐ Folder is ready for archiving

☑ Allow files in this folder to have contents indexed in addition to file properties

Compress or Encrypt attributes

☐ Compress contents to save disk space

☐ Encrypt contents to secure data

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

See also. *Map the My TekExpress folder to Drive X*

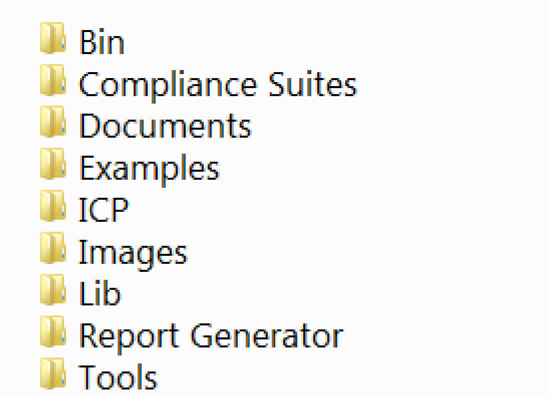
Application directories and usage

File name extensions

Application directories and their contents

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

```
C:\Program Files\Tektronix\TekExpress\TekExpress RIN
```



Bin
Compliance Suites
Documents
Examples
ICP
Images
Lib
Report Generator
Tools

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

Table 4: Application directories and usage

Directory names	Usage
Bin	Contains TekExpress RIN application libraries
Compliance Suites	Contains compliance-specific files
Documents	Contains the technical documentation for the TekExpress RIN application
Examples	Contains various support files
ICP	Contains instrument and TekExpress RIN application-specific interface libraries
Images	Contains images of the TekExpress RIN application
Lib	Contains utility files specific to the TekExpress RIN application
Report Generator	Contains style sheets for report generation
Tools	Contains instrument and TekExpress RIN application-specific files

See also. *View test-related files*

File name extensions

File name extensions

The TekExpress RIN 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*
.pdf	Test result reports Application help document
.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 RIN for the first time, it creates the following folders on the oscilloscope:

- `\My Documents\My TekExpress\RIN`
- `\My Documents\My TekExpress\RIN\Untitled Session`

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

⚠ ***CAUTION.***

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

See also. *Map the My TekExpress folder to drive X*

Set the My TekExpress folder permissions

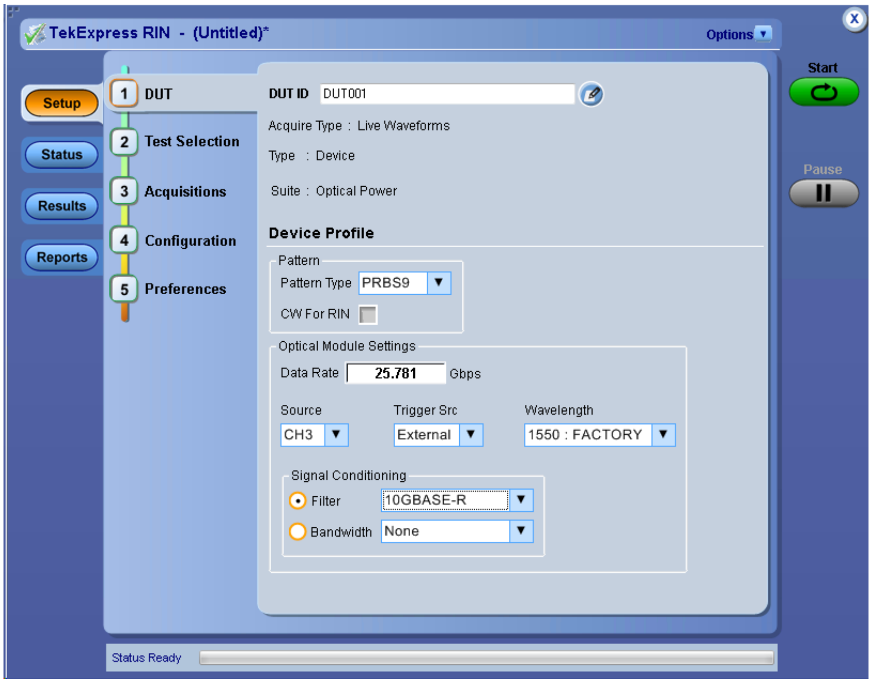
Application directories and usage

File name extensions

Operating basics

Run the application

To launch the TekExpress RIN application, select **Application** > **RIN** from the TekScope menu. The oscilloscope opens the TekExpress RIN application:



DUT ID DUT001

Acquire Type : Live Waveforms

Type : Device

Suite : Optical Power

Device Profile

Pattern

Pattern Type PRBS9

CW For RIN □

Optical Module Settings

Data Rate 25.781 Gbps

Source CH3

Trigger Src External

Wavelength 1550 : FACTORY

Signal Conditioning

● Filter 10GBASE-R

○ Bandwidth None

Status Ready

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

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

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

See also

Required My TekExpress folder settings

Exit the application

Application controls

Application panel overview

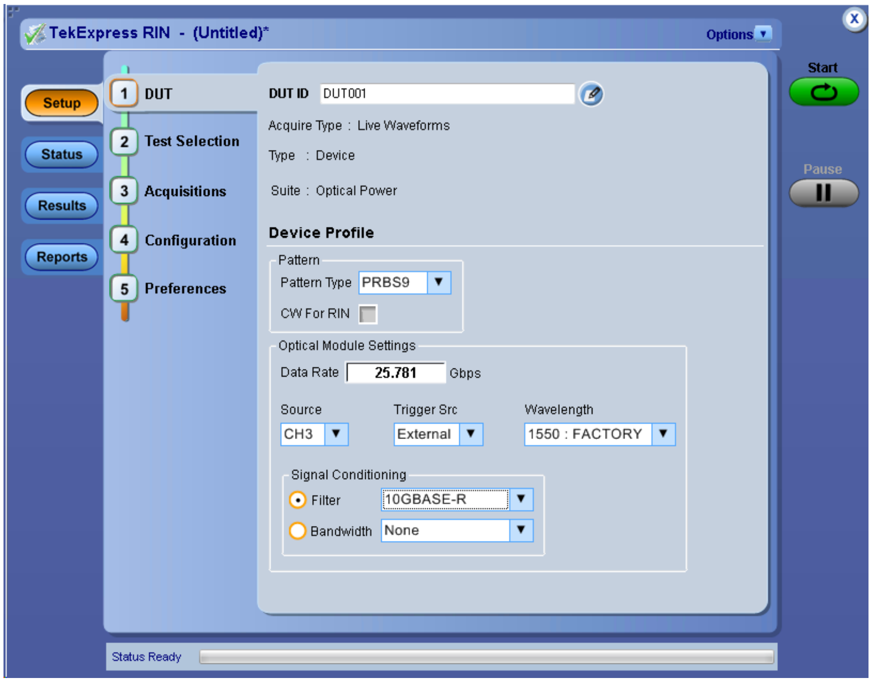
Exit the application

To exit the application, click X on the application title bar. Follow on-screen prompts to save any unsaved session, save test setup files, or exit the application.

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

Application panels overview

TekExpress RIN uses panels to group related configuration, test, and results settings. Click on 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.



DUT ID DUT001

Acquire Type : Live Waveforms

Type : Device

Suite : Optical Power

Device Profile

Pattern

Pattern Type PRBS9

CW For RIN □

Optical Module Settings

Data Rate 25.781 Gbps

Source CH3

Trigger Src External

Wavelength 1550 : FACTORY

Signal Conditioning

● Filter 10GBASE-R

○ Bandwidth None

Status Ready

Table 5: Application panels overview

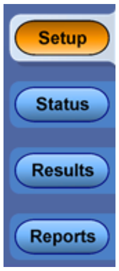
Panel Name	Purpose
Setup control overview	The Setup panel shows the test setup controls. Click the **Setup** button to open this panel. Use this panel to: ■ *Set DUT parameters* ■ *Select tests* ■ *Set acquisition tab parameters* ■ *Set the configuration tab parameters* ■ Set test notification parameters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

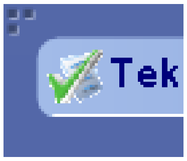
See also *Application controls*

Global application controls

Application controls

Table 6: Application controls descriptions

Item	Description
Options menu 	Menu to display global application controls.
Test Panel buttons  	Controls that open panels for configuring test settings and options.
Start / Stop button 	Use the Start button to start the test run of the measurements in the selected order. If prior acquired measurements have not been cleared, the new measurements are added to the existing set. The button toggles to the Stop mode while tests are running. Use the Stop button to abort the test.
Pause / Continue button 	Use the Pause button to temporarily interrupt the current acquisition. When a test is paused, the button name changes to “Continue.”

Item	Description
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*.
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 overview

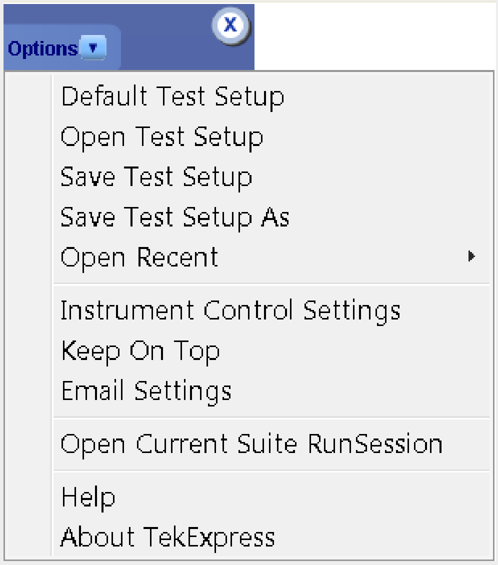
The Options menu is located near the upper-right corner of the application.

The Options menu has the following selections:

Menu	Function
Default Test Setup	Opens an untitled test setup with defaults selected **Pattern Type**: PRBS9 **CW for RIN**: Not selected **Data Rate**: 25.781 Gbps **Source**: CH1 (CH3, if installed module is only CH3). **Trigger Src**: External **Signal Conditioning**: Filter Values displayed in **Wavelength**, **Filter**, and **Bandwidth** fields depend on the installed optical module.
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 RIN application on top of other open windows on the desktop
Email Settings	Use to configure email options for test run and results notifications
Open Current Suite RunSession	Allows the user to select the specific run setup, from the saved session

Menu	Function
Help	Displays the TekExpress RIN help
About TekExpress	- Displays application details such as software name, version number, and copyright - Provides a link to the end-user license agreement - Provides a link to the Tektronix Web site

Options menu

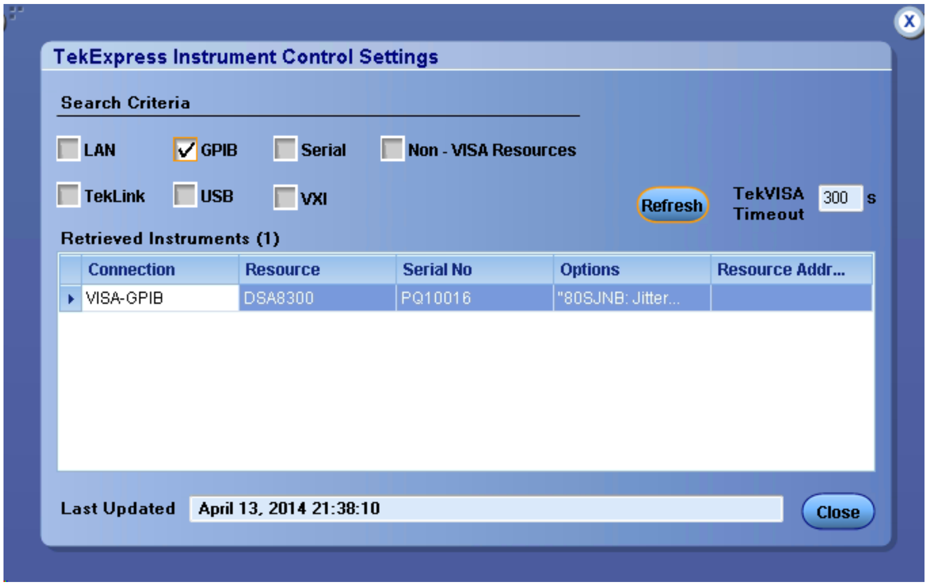


See also. *Application controls*

TekExpress instrument control settings

Use the TekExpress Instrument Control Settings dialog box to search for and list the connected resources (instruments) found on specified connections (LAN, GPIB, USB, and so on), and the connection information for each instrument.

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



TekExpress Instrument Control Settings

Search Criteria

□ LAN ☑ GPIB □ Serial □ Non - VISA Resources

□ TekLink □ USB □ VXI

Refresh

TekVISA Timeout 300 s

Retrieved Instruments (1)

Connection	Resource	Serial No	Options	Resource Addr...
VISA-GPIB	DSA8300	PQ10016	"80SJNB: Jitter...	

Last Updated April 13, 2014 21:38:10

Close

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

NOTE. *Under* ***Instrument Control Settings*** *select GPIB Option (Default setting) when using TekExpress RIN SW.*

See also. *Options menu overview*

View connected instruments

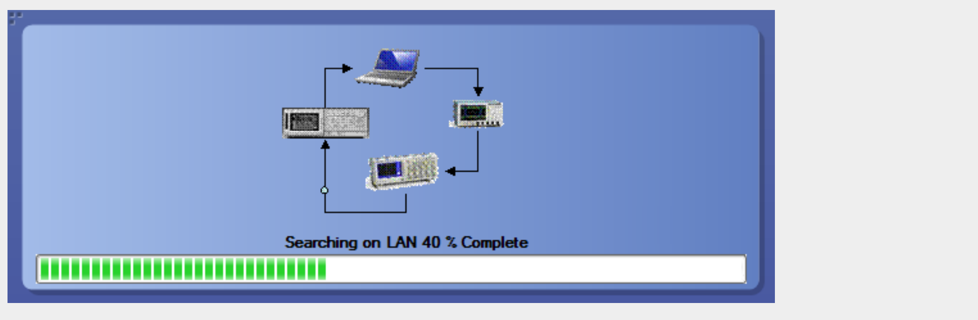
Use the Instrument Control Settings dialog box to view or search for connected instruments required for the tests. The application uses TekVISA to discover the connected instruments on all selected connection types.

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

To refresh the list of connected instruments:

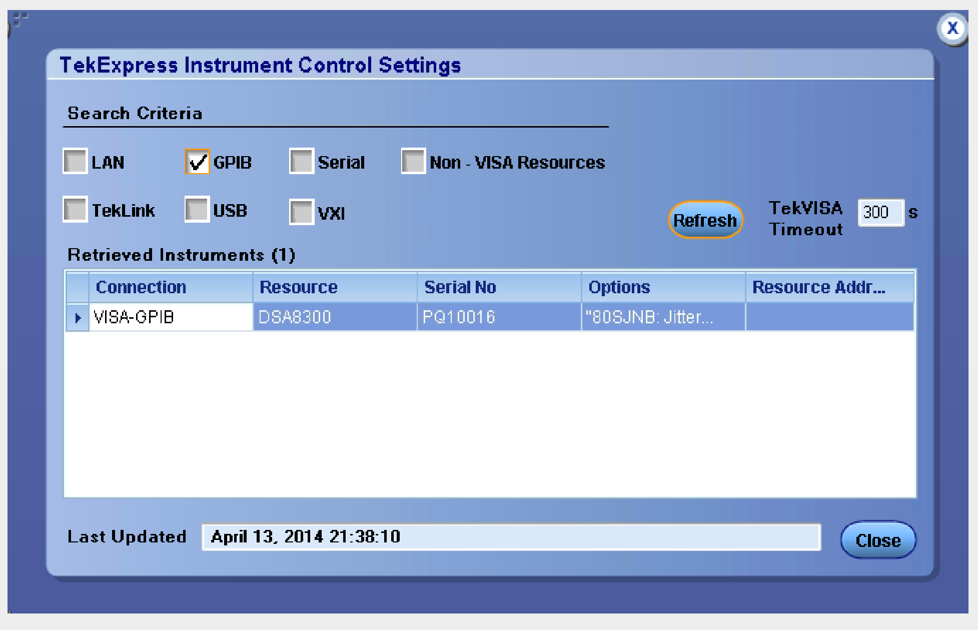
1. From the Options menu, select **Instrument Control Settings**.
2. In the **Search Criteria** section of the Instrument Control Settings dialog box, select the connection types of the instruments for which to search.

   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.



Searching on LAN 40 % Complete

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.



TekExpress Instrument Control Settings

Search Criteria

☐ LAN ☑ GPIB ☐ Serial ☐ Non - VISA Resources

☐ TekLink ☐ USB ☐ VXI

TekVISA Timeout 300 s

Retrieved Instruments (1)

Connection	Resource	Serial No	Options	Resource Addr...
VISA-GPIB	DSA8300	PQ10016	"80SJNB: Jitter...	

Last Updated April 13, 2014 21:38:10

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

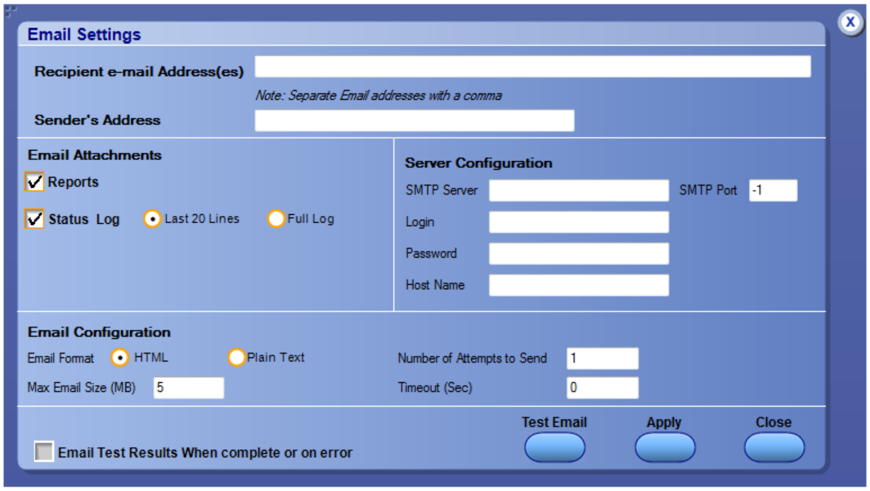
See also. *Equipment connection setup*

Email settings

Use the Email Settings utility to *configure email notifications* to receive notifications when a measurement completes or when there was an error in the measurement process. Select the type of test session information to include in the notification, 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.*



Email Settings

Recipient e-mail Address(es)

Note: Separate Email addresses with a comma

Sender's Address

Email Attachments

☑ Reports

☑ Status Log ● Last 20 Lines ○ Full Log

Server Configuration

SMTP Server | SMTP Port -1

Login

Password

Host Name

Email Configuration

Email Format ● HTML ○ Plain Text

Number of Attempts to Send 1

Max Email Size (MB) 5

Timeout (Sec) 0

☐ Email Test Results When complete or on error

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 measurement completes or produces an error condition:

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: DSA8300_B130099@yourcompany.com.

4. (Required) In the Server Configuration section, type the SMTP Server address of the Mail server configured at the client location, and the SMTP Port number, in the corresponding fields.

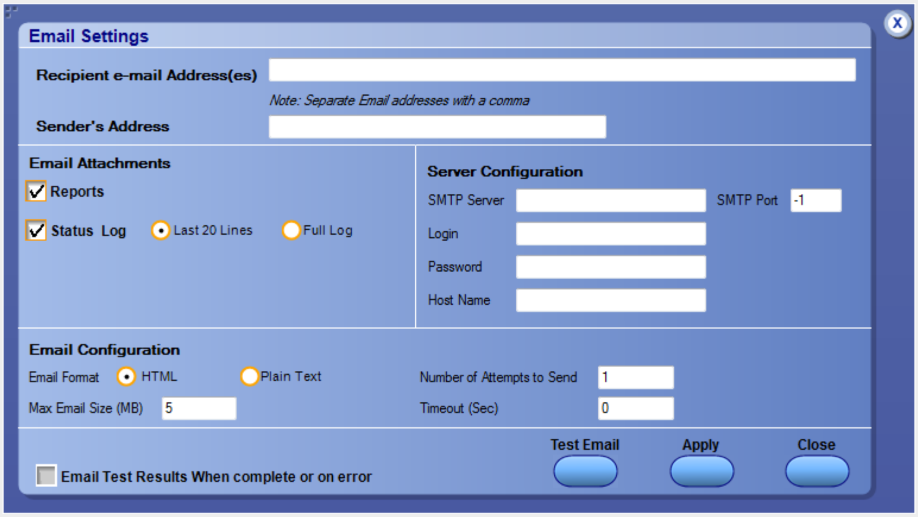
   If this server requires password authentication, enter a valid login name, password, and host name in the corresponding fields.

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

   ---

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

Email settings



Email Settings

Recipient e-mail Address(es)

Note: Separate Email addresses with a comma

Sender's Address

Email Attachments

☑ Reports

☑ Status Log ● Last 20 Lines ○ Full Log

Server Configuration

SMTP Server — SMTP Port: -1

Login

Password

Host Name

Email Configuration

Email Format: ● HTML ○ Plain Text

Max Email Size (MB): 5

Number of Attempts to Send: 1

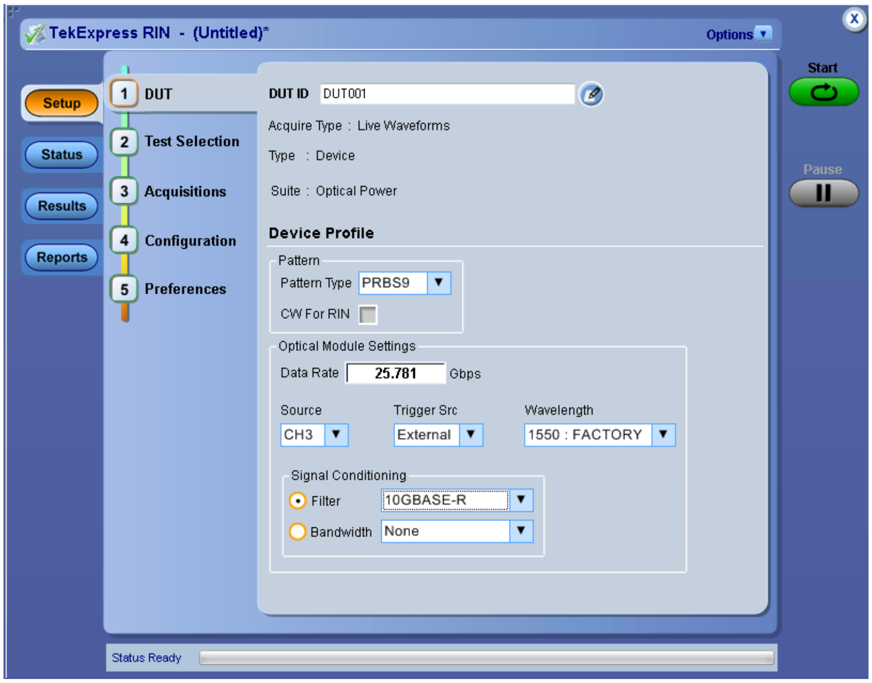
Timeout (Sec): 0

☐ Email Test Results When complete or on error

Application test panels

Setup tabs

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



DUT ID DUT001

Acquire Type : Live Waveforms

Type : Device

Suite : Optical Power

Device Profile

Pattern

Pattern Type PRBS9

CW For RIN □

Optical Module Settings

Data Rate 25.781 Gbps

Source	Trigger Src	Wavelength
CH3	External	1550 : FACTORY

Signal Conditioning

● Filter 10GBASE-R

○ Bandwidth None

Status Ready

The tabs on this panel are:

DUT: *Set the DUT parameters*

Test Selection: *Select test(s)*

Configuration: *Set the configuration tab parameters*

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.

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

Table 7: DUT tab settings

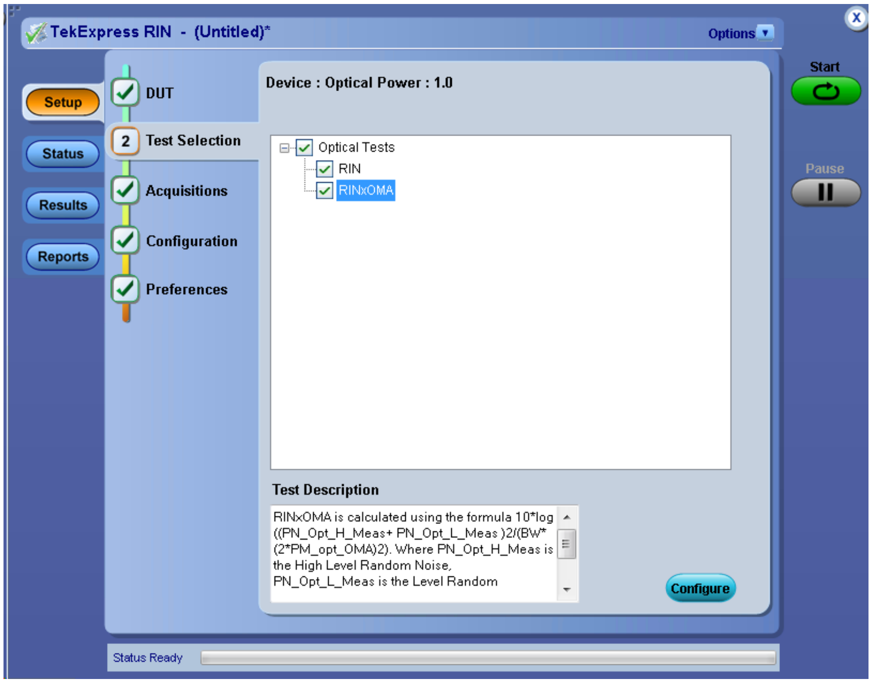
Setting	Description
DUT ID	Adds an optional text label for the DUT to reports. The default value is DUT001. The maximum number of characters is 32. You cannot use the following characters in an ID name: (.,..,...,\,/:?”<>|*)
✎ Comments icon (to the right of the DUT ID field)	Opens a Comments dialog box in which to enter optional text to add to a report. Maximum size is 256 characters. To enable or disable comments appearing on the test report, see (*Select report options*.)
Acquire Type	Live waveforms. The application performs analysis on live waveforms only. This field is not editable.
Type	Device. This field is not editable. RIN measurements are taken on Devices.
Suite	Optical Power. This field is not editable.
Device Profile	
Pattern Type	Select the testing pattern type from the drop-down list. Select **None** to take RIN-only measurements (requires selecting the CW for RIN check box).
CW for RIN	Select to use a continuous waveform for RIN measurement. To take RIN-only measurements, select **CW for RIN** and select **None** in the Pattern Type list.
Data Rate	Set the data rate to be tested.
Source	Select the optical module channel to use for testing (CH1 or CH3).
Trigger Src	Select the trigger source (Internal or External), from the drop-down list. Internal: The application only shows this selection if the selected optical module includes a clock recovery option. External: Select when using an external clock source. Connect the external clock signal to the CLOCK INPUT / PRESCALE TRIGGER connector on the front of the DSA8300.
Wavelength	Select the wavelength from the drop-down list.
Filter	Select the filter from the drop-down list.
Bandwidth	Select the bandwidth value from the drop-down list.

NOTE. *The installed optical module sets the available wavelength, filter, and bandwidth selections.*

You can set either the filter or bandwidth value, but not both.

See also. *Select a test*

Select tests. Use the **Test Selection** tab to select TekExpress RIN tests.



Device : Optical Power : 1.0

☑ Optical Tests
☑ RIN
☑ RINxOMA

Test Description

RINxOMA is calculated using the formula 10*log ((PN_Opt_H_Meas+ PN_Opt_L_Meas)2/(BW* (2*PM_opt_OMA)2). Where PN_Opt_H_Meas is the High Level Random Noise, PN_Opt_L_Meas is the Level Random

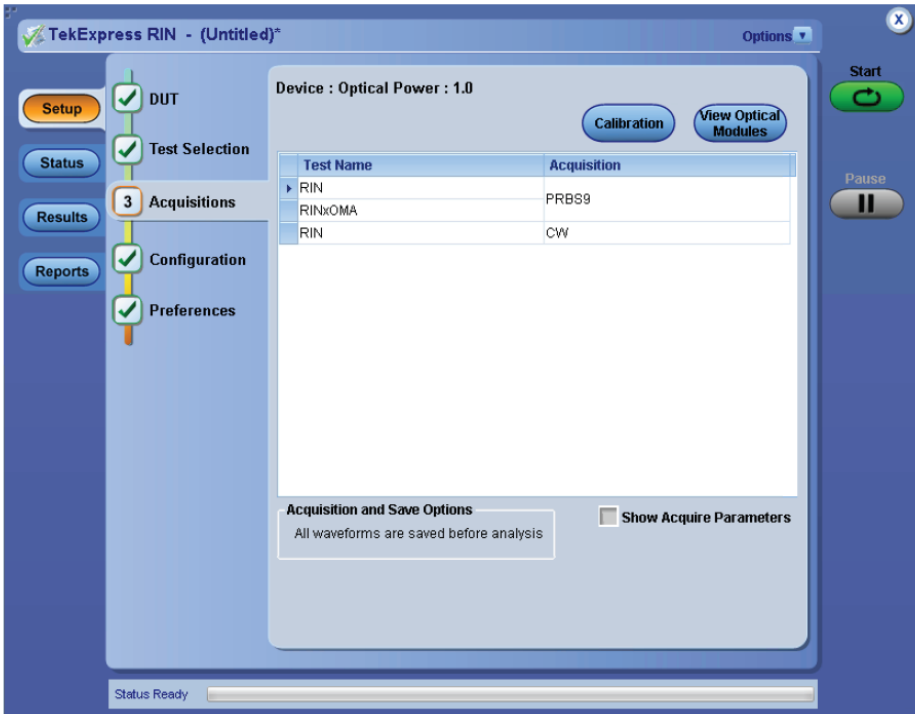
Table 8: Test Selection tab settings

Setting	Description
Tests	Click on a test to select or unselect. Highlight a test to show details in the Test Description pane.
Test Description	Shows a brief description of the highlighted test in the Test field.

See also. *Set acquisition tab parameters*

Set acquisition tab parameters. Use the **Acquisitions** tab in the Setup panel to view test acquisition parameters.

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



Device : Optical Power : 1.0

Test Name	Acquisition
RIN	PRBS9
RINxOMA	
RIN	CW

Acquisition and Save Options

All waveforms are saved before analysis

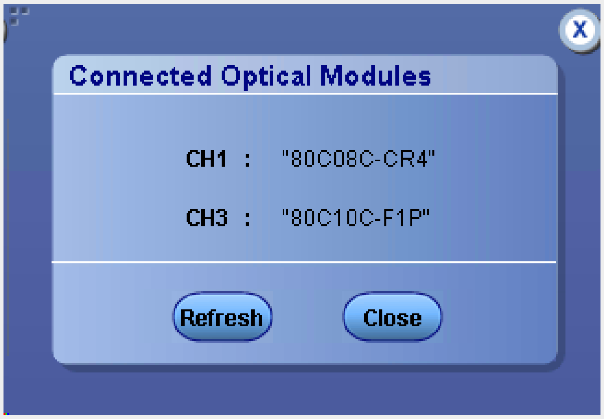
☐ Show Acquire Parameters

Status Ready

NOTE. *RIN acquires all waveforms needed by each test group before performing analysis.*

Table 9: Acquisitions tab settings

Setting	Description
Calibration button	Shows the results of the most recent instrument calibration. Use the Calibrations dialog box to view the status of oscilloscope calibration, external attenuation and instrumentation noise. Update these parameters by clicking the associated Refresh or Measure button. *Calibration guidelines*
View Optical Modules button	Shows the detected optical modules that are installed in the instrument.
Show Acquire Parameters	Shows the acquisition parameters.



Connected Optical Modules

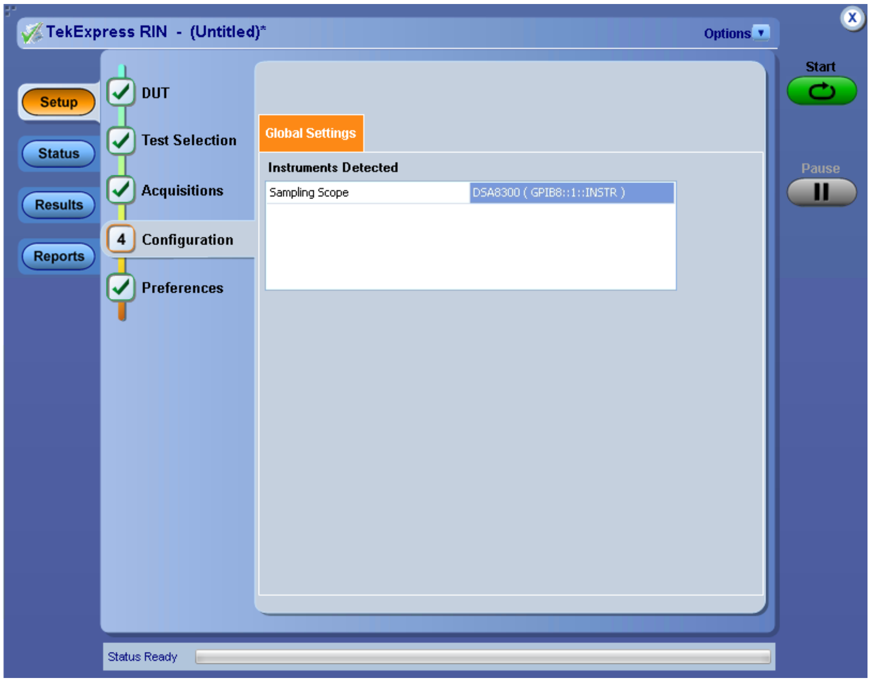
CH1 : "80C08C-CR4"

CH3 : "80C10C-F1P"

TekExpress RIN 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:\RIN\Untitled Session\<dutid>\<date>_<time>`. Images created for each analysis, CSV files with result values, reports, and other information specific to that particular execution are also saved in this folder.

Saving a session moves the session file contents from the Untitled Session folder to the specified folder name, and changes the session name to the specified name.

Set the configuration tab parameters. Use the **Configuration** tab to view the instruments detected (Global Settings).



Instruments Detected

Sampling Scope	DSA8300 (GPIB8::1::INSTR)

Figure 1: Configuration tab: Global Settings

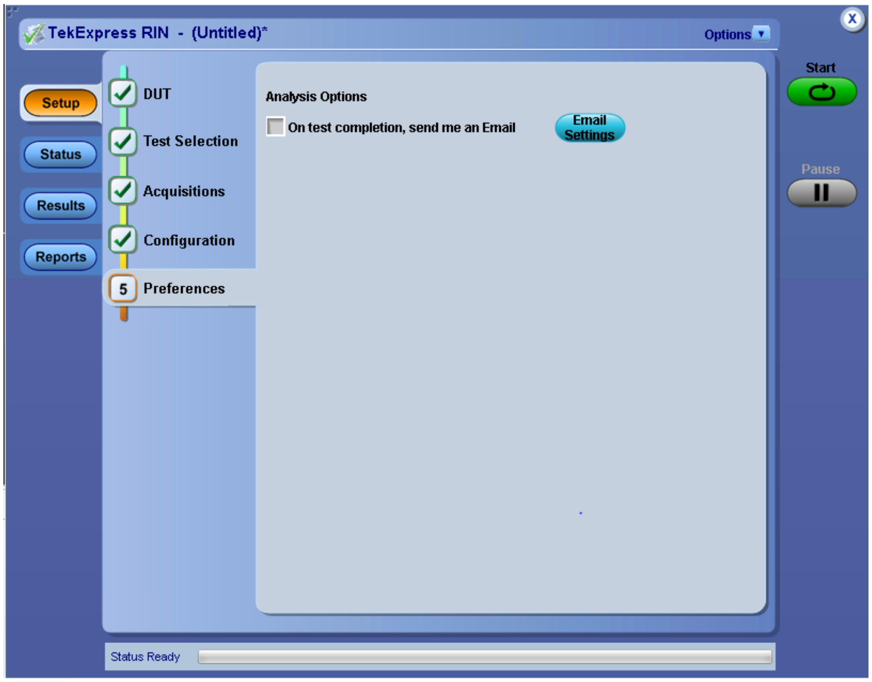
Global Settings tab. The Global Settings tab lists the connected instruments (oscilloscopes, signal sources, and so on) found during the instrument discovery operation. Use this list to verify that you are connected to the instrument with which you are acquiring waveforms. Click on the instrument name to open a list of available (detected) instruments.

Select **Options > Instrument Control Settings** and click Refresh to update the instrument list.

NOTE. *Verify that the* ***GPIB*** *search criteria (default setting) in the Instrument Control Settings is selected when using TekExpress RIN SW.*

Preferences tab

Use the Preferences tab to set the application action on completion of a measurement.



Analysis Options

☐ On test completion, send me an Email

Status Ready

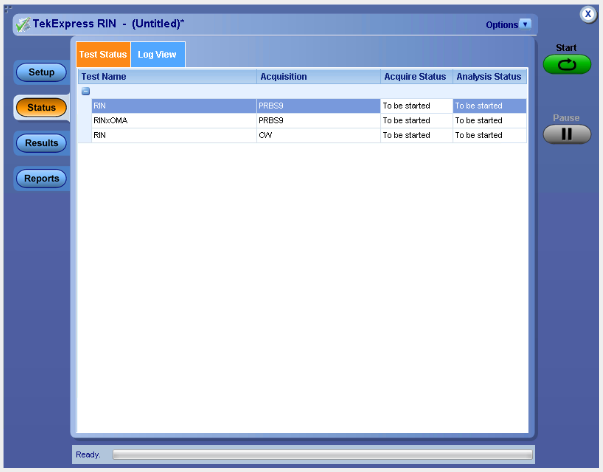
Table 10: Preferences tab settings

Setting	Description
On test completion, send me an E-mail	Sends an email when a test is completed. Click Email Settings to verify that Email test results when complete, and to verify the address to which the email is sent.

Status panel overview

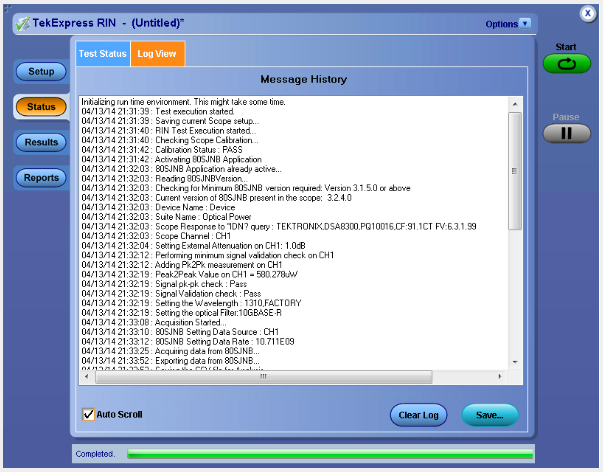
The Status button accesses the Test Status and Log View tabs, which provide status on test acquisition and analysis (Test Status tab) and a listing of test tasks performed (Log View tab). The application opens the Test Status tab when you start a test run. You can select the Test Status or the Log View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

Test status view



Test Name	Acquisition	Acquire Status	Analysis Status
RIN	PRBS9	To be started	To be started
RINxOMA	PRBS9	To be started	To be started
RIN	CW	To be started	To be started

Log view



```
Initializing run time environment. This might take some time.
04/13/14 21:31:39 : Test execution started.
04/13/14 21:31:39 : Saving current Scope setup...
04/13/14 21:31:40 : RIN Test Execution started...
04/13/14 21:31:40 : Checking Scope Calibration...
04/13/14 21:31:42 : Calibration Status : PASS
04/13/14 21:31:42 : Activating 80SJNB Application
04/13/14 21:32:03 : 80SJNB Application already active...
04/13/14 21:32:03 : Reading 80SJNBVersion...
04/13/14 21:32:03 : Checking for Minimum 80SJNB version required: Version 3.1.5.0 or above
04/13/14 21:32:03 : Current version of 80SJNB present in the scope:  3.2.4.0
04/13/14 21:32:03 : Device Name : Device
04/13/14 21:32:03 : Suite Name : Optical Power
04/13/14 21:32:03 : Scope Response to *IDN? query : TEKTRONIX,DSA8300,PQ10016,CF:91.1CT FV:6.3.1.99
04/13/14 21:32:03 : Scope Channel : CH1
04/13/14 21:32:04 : Setting External Attenuation on CH1: 1.0dB
04/13/14 21:32:12 : Performing minimum signal validation check on CH1
04/13/14 21:32:12 : Adding Pk2Pk measurement on CH1
04/13/14 21:32:19 : Peak2Peak Value on CH1 = 580.278uW
04/13/14 21:32:19 : Signal pk-pk check : Pass
04/13/14 21:32:19 : Signal Validation check : Pass
04/13/14 21:32:19 : Setting the Wavelength : 1310,FACTORY
04/13/14 21:32:19 : Setting the optical Filter:10GBASE-R
04/13/14 21:33:08 : Acquisition Started...
04/13/14 21:33:10 : 80SJNB Setting Data Source : CH1
04/13/14 21:33:12 : 80SJNB Setting Data Rate : 10.711E09
04/13/14 21:33:25 : Acquiring data from 80SJNB...
04/13/14 21:33:52 : Exporting data from 80SJNB...
```

Table 11: Status panel Log View controls

Control	Description
Message History	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.

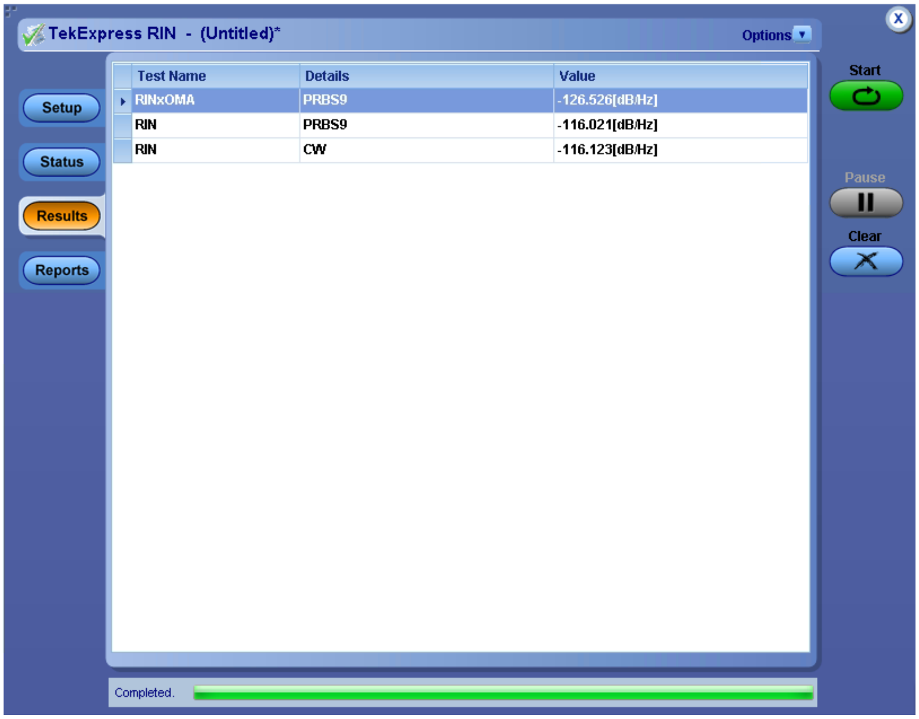
See also. *Application panel overview*

Results panel

Results panel overview. When a test finishes, the application automatically opens the **Results** panel to display a summary of test results.

See also. *View a report*

Application panels overview



Test Name	Details	Value
RINxOMA	PRBS9	-126.526[dB/Hz]
RIN	PRBS9	-116.021[dB/Hz]
RIN	CW	-116.123[dB/Hz]

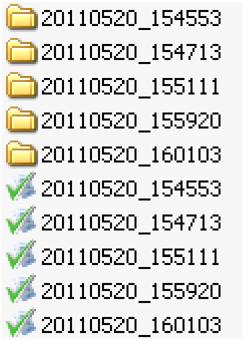
Completed.

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

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

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

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



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

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

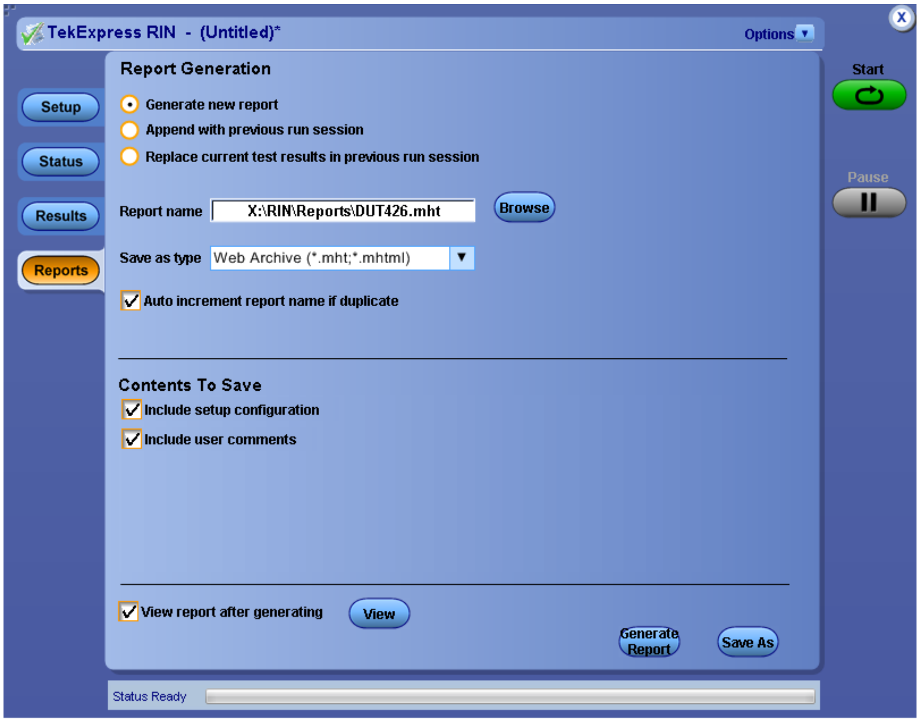
The first time you run a new, unsaved session, the session files are stored in the `Untitled Session` folder located at `..\My TekExpress\RIN`. When you name and save the session, the files are placed in a folder with the name that you specify. A copy of the test files stay in the Untitled Session folder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

See also. *File name extensions*

Required My TekExpress folder settings

Reports panel

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



TekExpress RIN - (Untitled)*

Report Generation

● Generate new report

○ Append with previous run session

○ Replace current test results in previous run session

Report name: X:\RIN\Reports\DUT426.mht

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

☑ Auto increment report name if duplicate

Contents To Save

☑ Include setup configuration

☑ Include user comments

☑ View report after generating

Status Ready

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

See also. *Applications panel overview*

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 12: Report options

Setting	Description
Report Generation	
Generate new report	Creates a new report. The report can be in either .mht or .pdf file formats.
Append with previous run session	Appends the latest test results to the end of the current test results report.
Replace current test in previous run session	Replaces the previous test results with the latest test results. Results from newly added tests are appended to the end of the report.

Setting	Description
Report name	Displays the name and location from which to open a RIN report. The default location is at *\My TekExpress\RIN\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\RIN\DUT001.mht. ***NOTE.*** *You cannot set the file location using the Browse button.* Open an existing report. Click **Browse**, locate and select the report file and then click **View** at the bottom of the panel.
Save as type	Saves a report in the specified file type. Lists supported file types to choose from. ***NOTE.*** *If you select a file type different from the default, be sure to change the report file name extension in the Report Name field to match.*
Auto increment report name if duplicate	Sets the application to automatically increment the name of the report file if the application finds a file with the same name as the one being generated. For example: DUT001, DUT002, DUT003. This option is enabled by default.
Contents To Save	
Include 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, the oscilloscope firmware version, and software versions for applications used in the measurements.

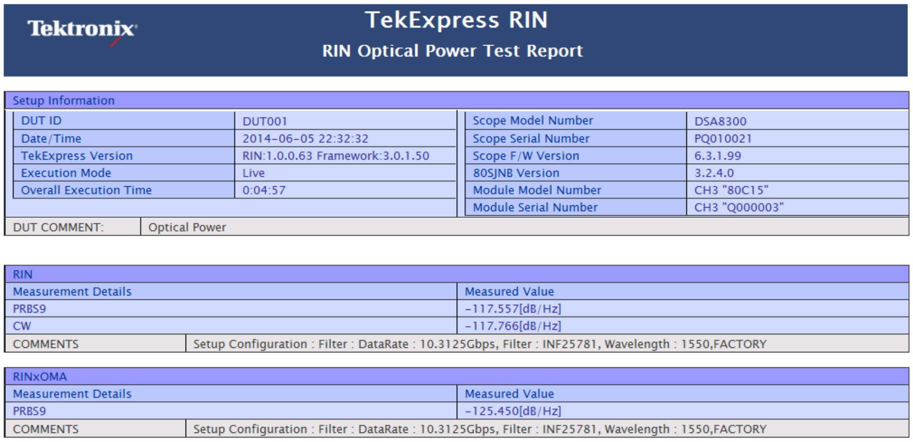
Setting	Description
Include user comments	Select to include any comments about the test that you or another user added in the DUT tab of the Setup panel. Comments appear in the Comments section, under the summary box at the beginning of each report.
View report after generating	Automatically opens the report in a Web browser when the test completes. This option is selected by default.
View	Click to view the most current report.
Generate Report	Generates a new report based on the current 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 detailed results and plots, as set in the Reports panel.



TekExpress RIN

RIN Optical Power Test Report

Setup Information

DUT ID	DUT001	Scope Model Number	DSA8300
Date/Time	2014-06-05 22:32:32	Scope Serial Number	PQ010021
TekExpress Version	RIN:1.0.0.63 Framework:3.0.1.50	Scope F/W Version	6.3.1.99
Execution Mode	Live	80SJNB Version	3.2.4.0
Overall Execution Time	0:04:57	Module Model Number	CH3 "80C15"
		Module Serial Number	CH3 "Q000003"

DUT COMMENT: Optical Power

RIN

Measurement Details	Measured Value
PRBS9	-117.557[dB/Hz]
CW	-117.766[dB/Hz]

COMMENTS Setup Configuration : Filter : DataRate : 10.3125Gbps, Filter : INF25781, Wavelength : 1550,FACTORY

RINxOMA

Measurement Details	Measured Value
PRBS9	-125.450[dB/Hz]

COMMENTS Setup Configuration : Filter : DataRate : 10.3125Gbps, Filter : INF25781, Wavelength : 1550,FACTORY

Setup configuration information

The summary box at the beginning of the report lists setup configuration information. This information includes the oscilloscope model and serial number, optical module model and serial number, and software version numbers of all associated applications.

To exclude this information from a report, clear the **Include Setup Configuration** check box in the Reports panel before running the test.

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.

See also. *Results panel overview*

View test-related files

Pre-measurement calibration procedures

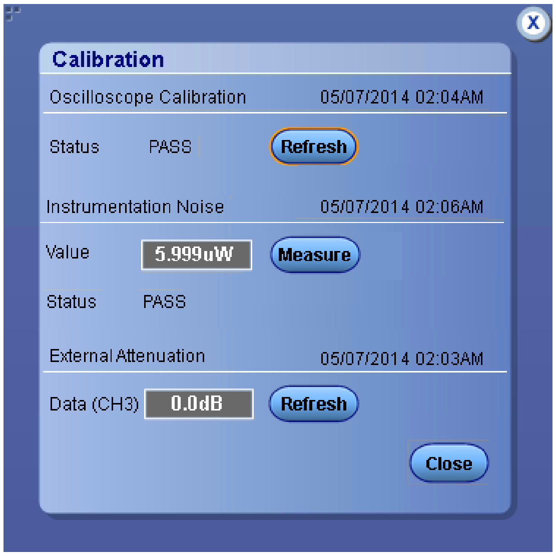
Pre-measurement calibration guidelines

- You need to perform the following procedures before starting a measurement session using the TekExpress RIN software, and any time after that you make changes to the setup configuration, such as after installing or moving any sampling modules, cables, or connectors.
- Perform the procedures in the following order:

  Oscilloscope calibration

  Instrument noise measurement

  External attenuation



Calibration

Oscilloscope Calibration 05/07/2014 02:04AM

Status PASS

Instrumentation Noise 05/07/2014 02:06AM

Value 5.999uW

Status PASS

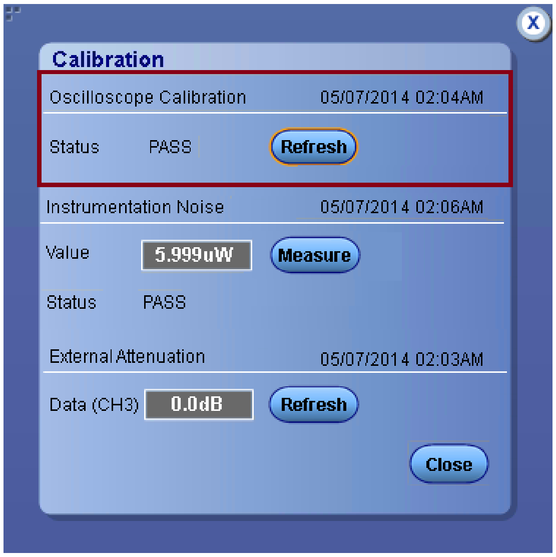
External Attenuation 05/07/2014 02:03AM

Data (CH3) 0.0dB

Oscilloscope calibration

Use the following procedure to check oscilloscope calibration status:

1. Select **TekExpress RIN** > **Setup** > **Acquisition** panel > **Calibration** button to open the Calibration dialog box.



Calibration

Oscilloscope Calibration 05/07/2014 02:04AM

Status PASS

Instrumentation Noise 05/07/2014 02:06AM

Value 5.999uW

Status PASS

External Attenuation 05/07/2014 02:03AM

Data (CH3) 0.0dB

Click **Refresh** (in the Oscilloscope Calibration area) to update the oscilloscope calibration status.

NOTE. *Its recommended to perform Scope Compensation in addition after 20 minutes of warm up. Scope compensation can be accessed from the Oscilloscope main menu,* ***Utilities*** *>* ***Instrument Compensation****. Click Help in the compensation window for further details.*

Instrument noise measurement

Instrumentation noise calibration

1. Disconnect all of the signals that are connected to the sampling oscilloscope.
2. Select **Setup** > **Vert** > **waveform C1** to **On**.
3. Set the minimum vertical scale per division to **1 mW/div** for Ch 1.
4. Set the Trigger Source to **Free Run**.
5. Select measurement **Setup** > **Meas** > **Meas 1** > **Pulse Amplitude: AC RMS**.
6. Set **Setup** > **Meas** > **Signal Type: Pulse**.
7. Set **Setups** > **Meas** > **Source: C1**.
8. Uncheck the **Use Wfm Database** control for the measurement.
9. Record the Ch 1 RMS value.

This measurement is performed automatically by the RIN SW.

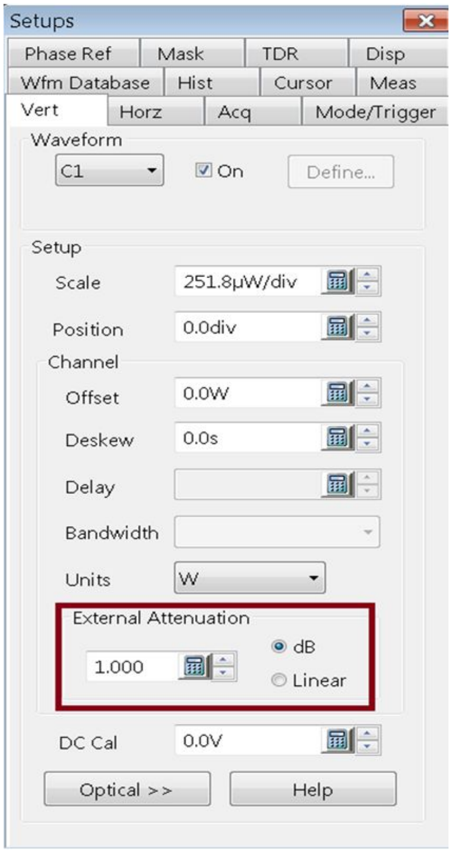
Noise level measurement should be in the range of **0.1 µW – 100 µW**.

If the noise level measurement is not within the limits, perform an oscilloscope compensation and then perform the instrument noise measurement again. If the measured noise level is still outside of the above limits, please *contact Tektronix* Customer Support.

External attenuation

Use the following procedure to set the external attenuation:

- On the DSA8300, set the optical source to **Ch1** or **Ch3**.
- Enter the Attenuation value for **Ch1** or **Ch3** in the External Attenuation field of the Vert tab, scope as shown in the following screen shot.



Waveform

C1 ☑ On

Setup

Scale 251.8μW/div

Position 0.0div

Channel

Offset 0.0W

Deskew 0.0s

Units W

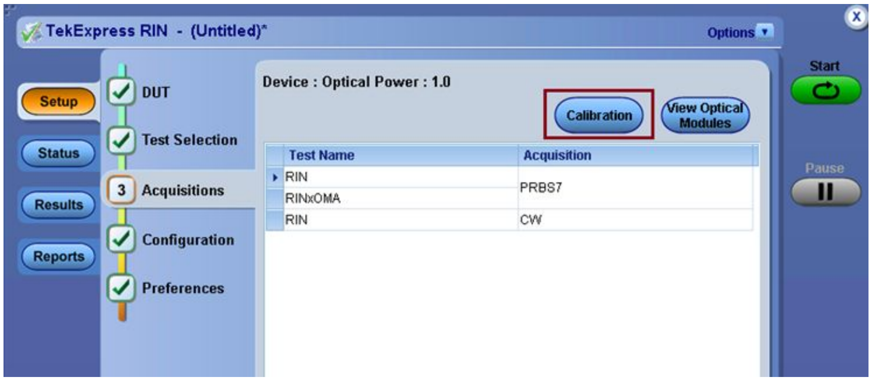
External Attenuation

1.000 ● dB ○ Linear

DC Cal 0.0V

- Select **Ch1** or **Ch3** from the **TekExpress RIN > Setup > DUT** panel > **Source**.

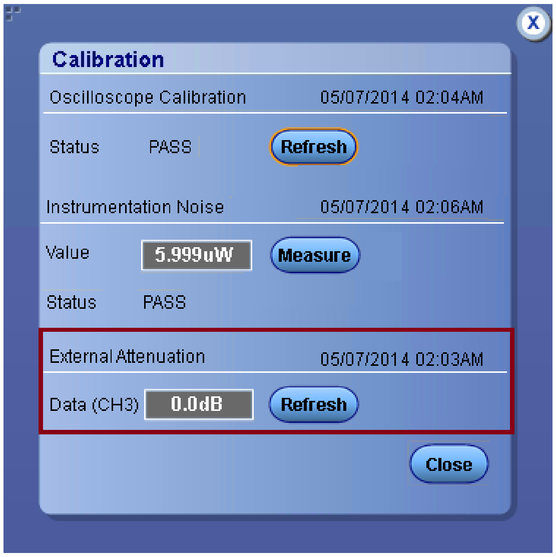
- Select **TekExpress RIN > Setup > Acquisition** panel and click the **Calibration** button.



Device : Optical Power : 1.0

Test Name	Acquisition
RIN	PRBS7
RINxOMA	
RIN	CW

- Click the **Refresh** button to update the external attenuation value in the **Calibration** dialog box.



Oscilloscope Calibration 05/07/2014 02:04AM

Status PASS

Instrumentation Noise 05/07/2014 02:06AM

Value 5.999uW

Status PASS

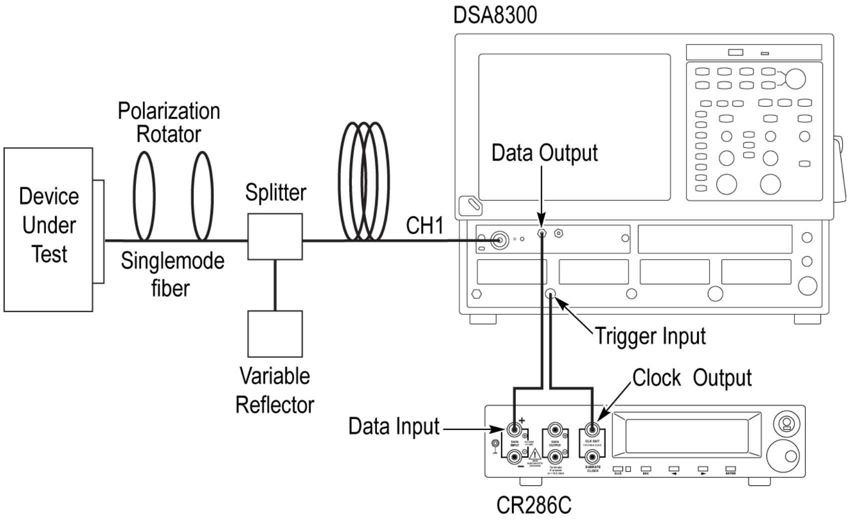
External Attenuation 05/07/2014 02:03AM

Data (CH3) 0.0dB

- Select the other channel from the **TekExpress RIN > Setup > DUT** panel > **Source** and refresh the **External Attenuation** button again.

Instrument and DUT connection setup

Click the **Setup > Test Selection > Schematic** button to open a PDF file that shows the test setup diagram(s) (instrument, DUT, connections, and cabling) for supported test configuration.



See also *Minimum system requirements*

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 between the Status panel and the Results panel.

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 RIN application on top, select **Keep On Top** from the TekExpress Options menu.

The application displays a report when the tests are complete.

NOTE. *The RIN software stores the DSA8300 oscilloscope setup at the beginning of the test execution and restores the oscilloscope to the same state at the end of the test execution.*

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 and the Calibration procedures, before continuing.*

1. Make sure that all the required instruments are properly warmed up (approximately 20 minutes).
2. Perform compensation:
   a. On the oscilloscope main menu, select **Utilities > Instrument Compensation**.
   b. Click the **Help** button in the Compensation window for information on how to perform instrument compensation.

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) is saved under the setup name at **X:\RIN**.

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:\RIN\<*session_name*>.

See also *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:\RIN**.

See also *About test setups*

Create a new test setup based on an existing one

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*

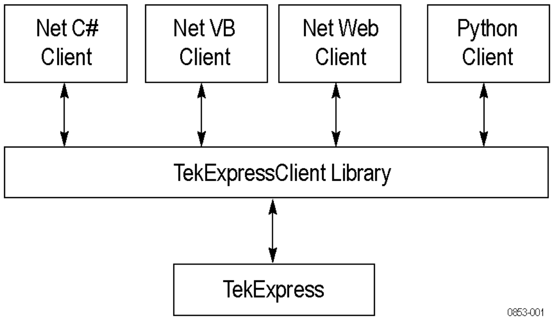
Set DUT parameters

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.

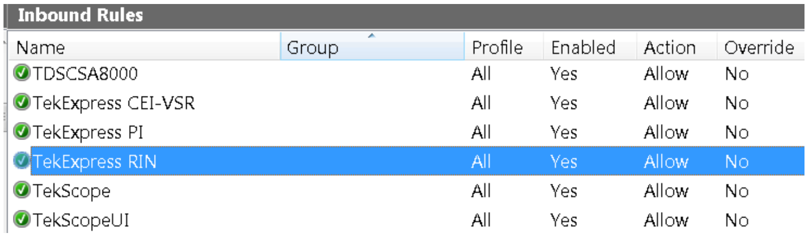
TekExpress leverages .Net Marshalling to enable the Programmatic Interface for TekExpress Client. TekExpress provides a client library for TekExpress clients to use the programmatic interface. The TekExpress client library is inherited from .Net MarshalByRef class to provide the proxy object for the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



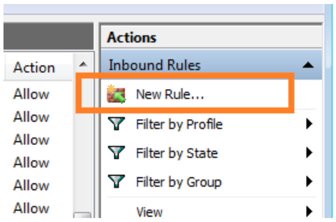
Inbound Rules

Name	Group	Profile	Enabled	Action	Override
TDSCSA8000		All	Yes	Allow	No
TekExpress CEI-VSR		All	Yes	Allow	No
TekExpress PI		All	Yes	Allow	No
TekExpress RIN		All	Yes	Allow	No
TekScope		All	Yes	Allow	No
TekScopeUI		All	Yes	Allow	No

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 Client application.exe through which TekExpress application is remotely controlled. For example if the application is controlled using python scripts then "ipy64.exe" should be included as part of Inbound rules.

Run the New Inbound Rule Wizard

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

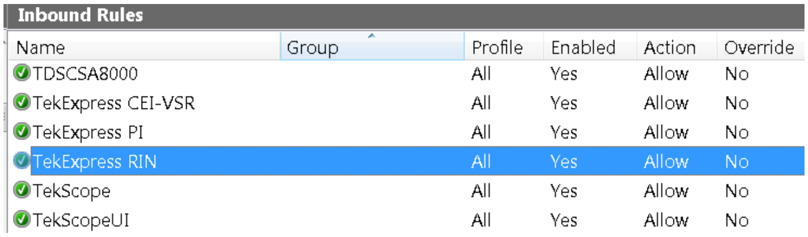


2. Verify that **Program** is selected in the Rule Type panel and click **Next**.
3. Click **Browse** in the Program panel and navigate to and select one of the following TekExpress applications (depending on the one for which you need to create a rule):

   - TekExpress RIN.exe
   - TekExpress.exe

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

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



Inbound Rules

Name	Group	Profile	Enabled	Action	Override
TDSCSA8000		All	Yes	Allow	No
TekExpress CEI-VSR		All	Yes	Allow	No
TekExpress PI		All	Yes	Allow	No
TekExpress RIN		All	Yes	Allow	No
TekScope		All	Yes	Allow	No
TekScopeUI		All	Yes	Allow	No

10. Repeat steps 1 through 9 to enter the other TekExpress executable if it is missing from the list. Enter **TekExpress PI** as the name.
11. Scroll through the Inbound Rules list and verify that the list shows the rule that you just entered.
12. Exit the Windows Firewall tool.

To use the remote access:

1. Obtain the IP address of the instrument on which you are running TekExpress RIN. 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 RIN PI code to access that instrument. For example:

   ```
   object obj = piClient.Connect(“134.64.235.198”,out
   clientid);
   ```

Requirements for developing TekExpress client

Use the TekExpressClient.dll to develop your client. 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:

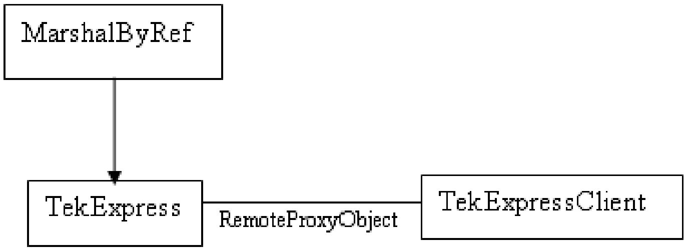
1. To connect to the server, the client provides the IP address of the PC where the server is running.
2. The client locks the server application to avoid conflict with any other Client that may try to control the server simultaneously. “Lock” would also disable all user controls on the server so that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

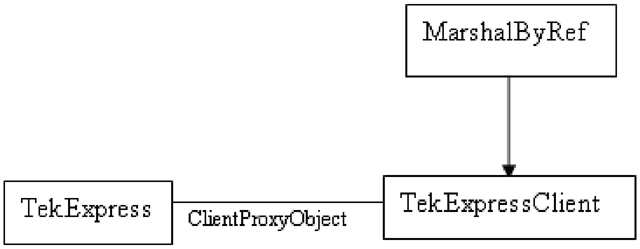
For example,

//Get a reference to the remote object

```
remoteObject =
(IRemoteInterface)Activator.GetObject(typeof(IRemoteInterfac
e), 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(Cl
ientInterface), “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.

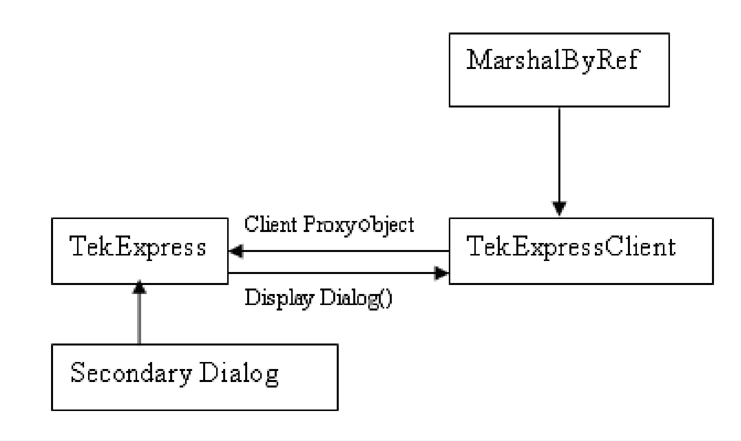
Examples

```
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.

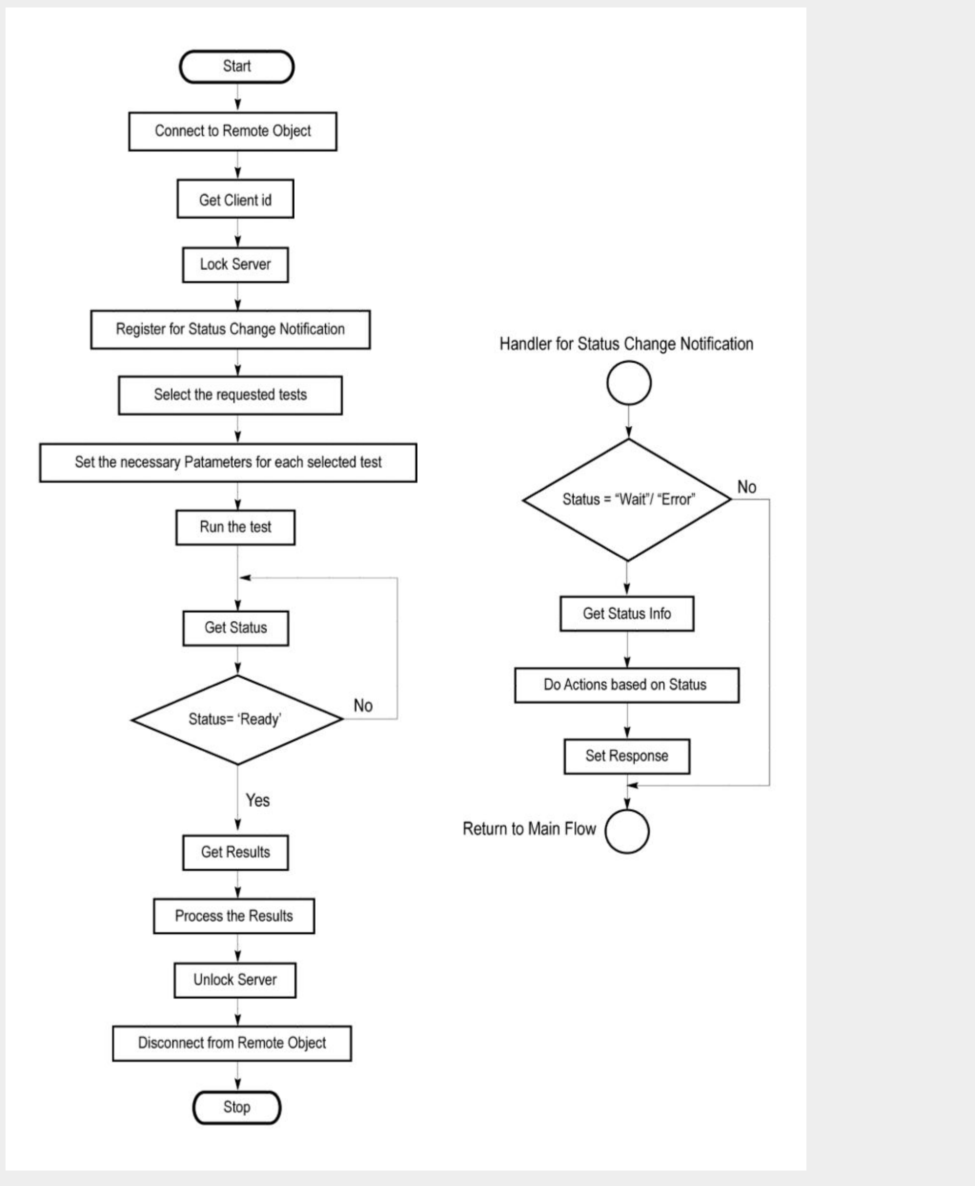
In the secondary dialog library, the assembly that is calling for the dialog box to be displayed is checked and if a remote connection is detected, the messages are directed to the remote end.

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:



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 RIN.

Table 13: Remote access code example

Task	Code
Start the application	

Connect through an IP address.	'assigns client IP address to variable clientID; address valid until connection or measurement session ends (Disconnect). See Connect()
	`clientID = “ ”`
	`m_Client.Connect("localhost",out clientID)'True or False`

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	

RIN programmer interface commands

ApplicationStatus()

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

Parameters.

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

clientId variable

clientId is a user-defined variable that stores the client Id address information. Use the Connect() command to fill this variable:

```
clientId = “ ”
```

```
m_Client.Connect("localhost",out clientId)'True or False
```

The clientId variable is stored until you call the Disconnect command.

NOTE.

The Fail condition for this command occurs in the following conditions:

If the server is LOCKED the command returns "Server is locked by another client".

If the session is UNLOCKED the command returns "Lock Session to execute the command".

If the server is NOTFOUND the command returns "Server not found...Disconnect!".

If none of these fail conditions occur the command returns "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

ChangeDutId()

ChangeDutId(clientId, dutName). This command changes the DUT id of the set-up. The client has to provide a valid DUT id.

Parameters.

Parameter	Type	Direction	Description
clientId	String	IN	Identifier of the client that is performing the remote function. clientId variable
dutName	String	IN	The new DUT id of the set-up.

clientId variable

clientId is a user-defined variable that stores the client Id address information. Use the Connect() command to fill this variable:

```
clientId = “ ”
```

```
m_Client.Connect("localhost",out clientId)'True or False
```

The clientId variable is stored until you call the Disconnect command.

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

NOTE. *The Fail condition for this command occurs in the following conditions:*

If the server is LOCKED the command returns "Server is locked by another client".

If the session is UNLOCKED the command returns "Lock Session to execute the command".

If the server is NOTFOUND the command returns "Server not found...Disconnect!".

If none of these fail conditions occur the command returns "Failed...".

Example. `If (dut Id.Length <=0 && locked == true)`

```
return “Enter a valid DUT-Id”;

returnVal = remoteObject.ChangeDutId(clientId, dutId);

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

return “DUT Id Changed...”;

else

return CommandFailed(returnVal);
```

Comments. If the dutName parameter is null, the client is prompted to provide a valid DUT id.

Related command(s). *GetDutId*

CheckSessionSaved()

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

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.
clientId	string	IN	Identifier of the client that is performing the remote function. clientId variable
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)

clientId variable

clientId is a user-defined variable that stores the client Id address information. Use the Connect() command to fill this variable:

```
clientID = “ ”
m_Client.Connect("localhost",out clientId)'True or False
```

The clientId variable is stored until you call the Disconnect command.

Comments.

Related command(s). *RecallSession*

SaveSession

SaveSessionAs

Connect()

Connect(string HostIPAddress, out string clientId). This command connects the client to the server; address is the IP address of the server to which the client is trying to connect. This is required to establish the connection between the client and the server.

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

Parameters.

Parameter	Type	Direction	Description
HostIPAddress	string	IN	Obtains 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.
clientId	string	OUT	Identifier of the client that is performing the remote function. clientId variable

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 this command occurs in the following conditions:*

If the server is LOCKED the command returns "Server is locked by another client".

If the session is UNLOCKED the command returns "Lock Session to execute the command".

If the server is NOTFOUND the command returns "Server not found...Disconnect!".

If none of these fail conditions occur the command returns "Failed...".

Example. `try {`

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

`clientId = “ ”`

`m_Client.Connect("localhost",out clientId)'True or False`

`}`

Comments. The server has to be active and running for the client to connect to the server. Any number of clients can be connected to the server at a time. Each client will get a unique id.

clientId variable

clientId is a user-defined variable that stores the client Id address information. Use the Connect() command to fill this variable:

```
clientId = “ ”
m_Client.Connect("localhost",out clientId)'True or False
```

The clientId variable is stored until you call the Disconnect command.

Related command(s). *Disconnect*

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 variable

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.

clientId variable

clientId is a user-defined variable that stores the client Id address information. Use the Connect() command to fill this variable:

clientId = “ ”

m_Client.Connect("localhost",out clientId)'True or False

The clientId variable is stored until you call the Disconnect command.

Related command(s). *Connect*

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 variable
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 this command occurs in the following conditions:*

If the server is LOCKED the command returns "Server is locked by another client".

If the session is UNLOCKED the command returns "Lock Session to execute the command".

If the server is NOTFOUND the command returns "Server not found...Disconnect!".

If none of these fail conditions occur the command returns "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)

clientId variable

clientId is a user-defined variable that stores the client Id address information. Use the Connect() command to fill this variable:

```
clientId = “ ”
m_Client.Connect("localhost",out clientId)'True or False
```

The clientId variable is stored until you call the Disconnect command.

Comments.

Related command(s). *ApplicationStatus*

QueryStatus

SendResponse

GetDutId()

GetDutId(clientId, out 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 variable
dutId	string	OUT	The DUT Id of the setup.

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

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

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

clientId variable

clientId is a user-defined variable that stores the client Id address information. Use the Connect() command to fill this variable:

```
clientId = “ ”
m_Client.Connect("localhost",out clientId)'True or False
```

The clientId variable is stored until you call the Disconnect command.

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

Related command(s). *ChangeDutId*

SetDutId

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 variable
device	string	IN	Specifies the DUT type (**Device**)
suite	string	IN	string with device connection type. Valid value is **Optical Power**.
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

clientId variable

clientId is a user-defined variable that stores the client Id address information. Use the Connect() command to fill this variable:

```
clientId = “ ”
```

```
m_Client.Connect("localhost",out clientId)'True or False
```

The clientId variable is stored until you call the Disconnect command.

NOTE. *The Fail condition for this command occurs in the following conditions:*

If the server is LOCKED the command returns "Server is locked by another client".

If the session is UNLOCKED the command returns "Lock Session to execute the command".

If the server is NOTFOUND the command returns "Server not found...Disconnect!".

If none of these fail conditions occur the command returns "Failed...".

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

Example. GetReportParameter(clientId, “Device”, “suite”, test, "Application Version")

GetResultsValue()

GetResultsValue(clientId, device, suite, 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 variable
device	string	IN	Specifies the DUT type (**Device**).
suite	string	IN	string with device connection type. Valid values is **Optical Power**.
test	string	IN	Specifies the name of the test for which to obtain the test result value. Measurement name (**RIN**).
parameterString	string	IN	Specifies to return the measured value for the indicated test. Enter **“Value”** for this argument

clientId variable

clientId is a user-defined variable that stores the client Id address information. Use the Connect() command to fill this variable:

```
clientId = " "
m_Client.Connect("localhost",out clientId)'True or False
```

The clientId variable is stored until you call the Disconnect command.

NOTE. *The Fail condition for this command occurs in the following conditions:*

If the server is LOCKED the command returns "Server is locked by another client".

If the session is UNLOCKED the command returns "Lock Session to execute the command".

If the server is NOTFOUND the command returns "Server not found...Disconnect!".

If none of these fail conditions occur the command returns "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", "suite", test, "Value");

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 variable

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 this command occurs in the following conditions:*

If the server is LOCKED the command returns "Server is locked by another client".

If the session is UNLOCKED the command returns "Lock Session to execute the command".

If the server is NOTFOUND the command returns "Server not found...Disconnect!".

If none of these fail conditions occur the command returns "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()

clientId variable

clientId is a user-defined variable that stores the client Id address information. Use the Connect() command to fill this variable:

```
clientId = “ ”
```

```
m_Client.Connect("localhost",out clientId)'True or False
```

The clientId variable is stored until you call the Disconnect command.

Comments.

Related command(s). *SetTimeOut*

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 variable

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.

clientId variable

clientId is a user-defined variable that stores the client Id address information. Use the Connect() command to fill this variable:

```
clientId = “ ”
```

```
m_Client.Connect("localhost",out clientId)'True or False
```

The clientId variable is stored until you call the Disconnect command.

Related command(s). *UnlockSession*

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 variable
status	string array	OUT	The list of status messages generated during the run

NOTE. *The Fail condition for this command occurs in the following conditions:*

If the server is LOCKED the command returns "Server is locked by another client".

If the session is UNLOCKED the command returns "Lock Session to execute the command".

If the server is NOTFOUND the command returns "Server not found...Disconnect!".

If none of these fail conditions occur the command returns "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)

clientId variable

clientId is a user-defined variable that stores the client Id address information. Use the Connect() command to fill this variable:

```
clientId = “ ”
```

```
m_Client.Connect("localhost",out clientId)'True or False
```

The clientId variable is stored until you call the Disconnect command.

Related command(s). *ApplicationStatus*

GetCurrentStateInfo

SendResponse

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 variable
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.

clientId variable

clientId is a user-defined variable that stores the client Id address information. Use the Connect() command to fill this variable:

```
clientId = “ ”

m_Client.Connect("localhost",out clientId)'True or False
```

The clientId variable is stored until you call the Disconnect command.

Related command(s). *SaveSession*

SaveSessionAs

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 variable

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);
```

clientId variable

clientId is a user-defined variable that stores the client Id address information. Use the Connect() command to fill this variable:

```
clientId = “ ”

m_Client.Connect("localhost",out clientId)'True or False
```

The clientId variable is stored until you call the Disconnect command.

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

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 variable
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.

clientId variable

clientId is a user-defined variable that stores the client Id address information. Use the Connect() command to fill this variable:

```
clientId = “ ”
m_Client.Connect("localhost",out clientId)'True or False
```

The clientId variable is stored until you call the Disconnect command.

Related command(s). *RecallSession*

SaveSessionAs

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 variable
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.

clientId variable

clientId is a user-defined variable that stores the client Id address information. Use the Connect() command to fill this variable:

```
clientId = “ ”
```

```
m_Client.Connect("localhost",out clientId)'True or False
```

The clientId variable is stored until you call the Disconnect command.

Related command(s). *RecallSession*

SaveSession

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 variable
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 this command occurs in the following conditions:*

If the server is LOCKED the command returns "Server is locked by another client".

If the session is UNLOCKED the command returns "Lock Session to execute the command".

If the server is NOTFOUND the command returns "Server not found...Disconnect!".

If none of these fail conditions occur the command returns "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)

clientId variable

clientId is a user-defined variable that stores the client Id address information. Use the Connect() command to fill this variable:

```
clientId = “ ”
```

```
m_Client.Connect("localhost",out clientId)'True or False
```

The clientId variable is stored until you call the Disconnect command.

Related command(s). *ApplicationStatus*

GetCurrentStateInfo

QueryStatus

SelectDevice()

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

Parameters.

Parameter	Type	Direction	Description
clientId	string	IN	Identifier of the client that is connected to the server clientId variable
device	string	IN	Specifies the DUT type (**Device**)

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

Example. SelectDevice(clientId,"Device")

SelectSuite()

SelectSuite(clientId, device, suite, true). This command selects the suite: "Optical Power."

Parameters.

Parameter	Type	Direction	Description
clientId	string	IN	Identifier of the client that is connected to the server clientId variable
device	string	IN	Specifies the DUT type (**Device**)
suite	string	IN	string with device connection type. Valid value is **Optical Power**.

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

clientId variable

clientId is a user-defined variable that stores the client Id address information. Use the Connect() command to fill this variable:

```
clientId = “ ”
m_Client.Connect("localhost",out clientId)'True or False
```

The clientId variable is stored until you call the Disconnect command.

Example. SelectSuite(clientId,"Device","Optical Power",true);

SelectTest()

SelectTest(clientId, device, suite, 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 variable
device	string	IN	Specifies the DUT type (**Device**).
suite	string	IN	string with device connection type. Valid value is **Optical Power**
test	string	IN	Name of the test (RIN or RINxOMA).

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

Example. SelectTest(clientId, device, suite, "RIN", true);

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 variable
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.
clientId variable

clientId is a user-defined variable that stores the client Id address information. Use the Connect() command to fill this variable:

```
clientId = " "
```

```
m_Client.Connect("localhost",out clientId)'True or False
```

The clientId variable is stored until you call the Disconnect command.

Related command(s). *GetDutId*

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 variable
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 this command occurs in the following conditions:*

If the server is LOCKED the command returns "Server is locked by another client".

If the session is UNLOCKED the command returns "Lock Session to execute the command".

If the server is NOTFOUND the command returns "Server not found...Disconnect!".

If none of these fail conditions occur the command returns "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)

clientId variable

clientId is a user-defined variable that stores the client Id address information. Use the Connect() command to fill this variable:

```
clientId = “ ”
```

```
m_Client.Connect("localhost",out clientId)'True or False
```

The clientId variable is stored until you call the Disconnect command.

Comments.

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 variable
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 this command occurs in the following conditions:*

If the server is LOCKED the command returns "Server is locked by another client".

If the session is UNLOCKED the command returns "Lock Session to execute the command".

If the server is NOTFOUND the command returns "Server not found...Disconnect!".

If none of these fail conditions occur the command returns "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)
```

clientId variable

clientId is a user-defined variable that stores the client Id address information. Use the Connect() command to fill this variable:

```
clientId = “ ”
```

```
m_Client.Connect("localhost",out clientId)'True or False
```

The clientId variable is stored until you call the Disconnect command.

TransferResult()

TransferResult(clientId, filePath). This command transfers the report generated after the run to the specified folder (directory). The report contains the summary of the run. The client has to provide the location where the report is to be saved at the client-end.

Parameters.

Parameter	Type	Direction	Description
clientId	string	IN	Identifier of the client that is connected to the server clientId variable
filePath	string	IN	Path to the target folder to which to transfer the report file. Enclose the path in quotes.

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

Example. TransferResult(clientId, "C:\\Report")

clientId variable

clientId is a user-defined variable that stores the client Id address information. Use the Connect() command to fill this variable:

```
clientId = “ ”
```

```
m_Client.Connect("localhost",out clientId)'True or False
```

The clientId variable is stored until you call the Disconnect command.

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

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 variable

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.

clientId variable

clientId is a user-defined variable that stores the client Id address information. Use the Connect() command to fill this variable:

```
clientId = “ ”
m_Client.Connect("localhost",out clientId)'True or False
```

The clientId variable is stored until you call the Disconnect command.

Related commands. *LockSession*

SetGeneralParameter command

SetGeneralParameter(). SetGeneralParameter(clientId, device, suite, "", paramString)

This command sets the general parameter and its value based on the "paramString" argument values as listed.

Table 14: Parameters

Parameter	Type	Direction	Description
clientId	string	IN	Identifier of the client that is connected to the server clientId variable
device	string	IN	Specifies the DUT type (**Device**).
suite	string	IN	Valid value is **Optical Power** .
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.

clientId variable

clientId is a user-defined variable that stores the client ID address information. Use the Connect() command to fill this variable:

```
clientID = “ ”
```

```
m_Client.Connect("localhost",out clientId)'True or False
```

The clientId variable is stored until you call the Disconnect command.

paramString argument values

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

Select source

Select trigger source

Select data rate

Select pattern type

Select CW for RIN

Select wavelength

Select filter

Select bandwidth

Select source. Use this paramString value to select the source which is used by the applicaiton. This is the same as selecting the Source in the DUT panel, Device profile, Optical module settings.

The value in bold font is the default value. If only one module is installed then the default value is the channel in which the module is installed.

Values:.

- **CH1**
- CH3

Example. `mClient.SetGeneralParameter(clientId, Device, Suite, "", "Optical signal Connected to the module:$ch1")`

Select trigger source. Use this paramString value to select the trigger source which is used by the application. This is the same as selecting the Trigger Src in the DUT panel, Device profile, Optical module settings.

The value in bold font is the default value. Internal is shown only the connected Optical module supports built in clock recovery.

Values:.

- **External**
- Internal

Example. mClient.SetGeneralParameter(clientId, Device, Suite, "", "Trigger Source$External")

Set data rate. Use this paramString value to set the data rate. This is the same as setting Data rate in the DUT panel, Device profile, Optical module settings.

The value in bold font is the default value.

Values:.

- **25.781**

Example. `mClient.SetGeneralParameter(clientId, Device, Suite, "", "DataRate (Gbps)$10.71")`

Select pattern type. Use this paramString value to select the pattern type. This is the same as selecting the Pattern type in the DUT panel, Device profile field.

The value in bold font is the default value.

Values:.

- PRBS7
- **PRBS9**
- PRBS11
- PRBS13
- PRBS15

- 8180
- 4140
- 2120
- None

Example. `mClient.SetGeneralParameter(clientId, Device, Suite, "", "PatternType$PRBS15")`

Select CW for RIN . Use this paramString value to select CW for RIN measurement. This is the same as selecting CW for RIN in the DUT panel, Device profile, Pattern.

The value in bold font is the default value.

Values:.

- **Excluded**
- Included

Example. `mClient.SetGeneralParameter(clientId, Device, Suite, "", "RinCWPattern$Excluded")`

Select wavelength. Use this paramString value to select wavelength. This is the same as selecting Wavelength in the DUT panel, Device profile, Optical module settings.

Values:.

- 1310 : FACTORY
- 1550 : FACTORY
- 850 : FACTORY
- 780 : FACTORY
- 1550 : USER

NOTE. *The above values are dynamically populated based on the options enabled on the optical module.*

Example. `mClient.SetGeneralParameter(clientId, Device, Suite, "", "Wavelength(nm)$1310 :FACTORY")`

Select filter. Use this paramString value to select the filter. This is the same as selecting Filter in the DUT panel, Device profile, Optical module settings.

Values:.

- FC11317
- 10GFC
- 10GBASE-R

- FEC10.66 Gb/s
- None

NOTE. *Values are dynamically populated based on the options enabled on the optical module.*

Example. `mClient.SetGeneralParameter(clientId, Device, Suite, "", "Filter$10GFC")`

Select bandwidth. Use this paramString value to select the oscilloscope bandwidth. This is the same as selecting Bandwidth in the DUT panel, Device profile, Optical module settings.

The value in bold font is the default value.

Values:.

NOTE. *Values are dynamically populated based on the options enabled on the optical module.*

Example. `mClient.SetGeneralParameter(clientId, Device, Suite, "", "BandWidth$32GHz")`

Algorithms

RIN and RIN CW

RIN

RIN (Relative Intensity Noise) is the ratio of the laser intensity's noise to the laser's power.

Use the following procedure to perform a RIN measurement:

1. Input the optical waveform exported from the 80SJNB application.
2. Find the position of the rising and falling edges in the input waveform, depending on the signal type. For example: If you are using a PRBS9 test pattern, search for a transition sequence four ones.
3. The vHigh is calculated by the mean logic one on the signal measured over the center 1 UI of the time intervals of the test pattern.
4. Use the following formula to calculate the RIN measurement:

$$RIN = 10 \cdot LOG \frac{P_N}{BW \cdot P_C}$$

Where:

- P_N is measured using 80 SJNB NoiseRMSHigh.
- P_C is measured OMA from high level.
- BW is the Optical bandwidth or bandwidth of Optical filter.

RIN CW

Use the following formula to calculate RIN CW measurement:

$$RIN = 10 \cdot LOG \frac{P_N}{BW \cdot P_C}$$

Where:

- P_N is measured using oscilloscope AC RMS measurement.
- P_C is measured from oscilloscope average Optical power measurement.
- BW is the Optical bandwidth or bandwidth of Optical filter.

For more details, refer to the*RIN and RIN OMA measurements on DSA8300* document on *www.tektronix.com*.

RINxOMA

RINxOMA is the Relative intensity noise on Optical modulation amplitude. Optical modulation amplitude is the difference between the high level and the low level, each level measured in a particular part of the waveform.

RINxOMA measurement verifies the relative intensity of noise optical modulation amplitude of the DUT signal.

Use the following procedure to perform RINxOMA measurement:

- Input the optical waveform exported from the 80SJNB application.
- Find the position of the rising and falling edges in the input waveform, depending on the signal type. For example: If you are using a PRBS9 test pattern, search for a transition sequence of five zeros and four ones.
- The vHigh and vLow is calculated by the mean logic one and mean logic zero on the signal measured over the center 1 UI of the two time intervals of the test pattern.
- OMA estimated value is the difference between the mean logic one and zero.
- The application calculates RINxOMA using the below formula:

$$\mathrm{RINxOMA}_2 = 10\log\left[\frac{\left(P_{N\_Opt\_H\_meas} + P_{N\_Opt\_L\_meas}\right)^2}{BW\cdot\left(P_{M\_Opt\_OMA}\right)^2}\right]$$

Where:

- $P_{N_Opt_H_meas}$is measured using 80 SJNB NoiseRMSHigh.
- $P_{N\_Opt\_L\_meas-}$ is measured using 80 SJNB NoiseRMSLow.
- BW is the Optical bandwidth or Bandwidth of Optical filter.
- $P_{M_opt_OMA-}$is measured OMA from high level.

NOTE. *RINxOMA measurement supports the following test patterns PRBS7, PRBS9, PRBS11, PRBS13, PRBS15, 8180, 4140, 2120.*

For more details, refer to the*RIN and RIN OMA measurements on DSA8300* document on *www.tektronix.com*.

Index