

**TekExpress® USB Power Adapter / EPS  
Compliance Automated Test Solution Software  
Printable Online Help**



077-0809-00



**TekExpress® USB Power Adapter / EPS  
Compliance Automated Test Solution Software  
Printable Online Help**

Copyright © Tektronix. All rights reserved. Licensed software products are owned by Tektronix or its subsidiaries or suppliers, and are protected by national copyright laws and international treaty provisions.

Tektronix products are covered by U.S. and foreign patents, issued and pending. Information in this publication supersedes that in all previously published material. Specifications and price change privileges reserved.

TEKTRONIX and TEK are registered trademarks of Tektronix, Inc.

TekExpress is a registered trademark of Tektronix, Inc.

### **Contacting Tektronix**

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

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

- In North America, call 1-800-833-9200.
- Worldwide, visit [www.tektronix.com](http://www.tektronix.com) to find contacts in your area.

# Table of Contents

## Getting help and support

|                                |   |
|--------------------------------|---|
| Related documentation .....    | 1 |
| Conventions used in help ..... | 1 |
| Technical support .....        | 2 |

## Getting started

|   |    |
|---|----|
| Installing the software                             |    |
| Minimum system requirements .....                   | 3  |
| Windows 7 user account settings .....               | 3  |
| Supported oscilloscopes .....                       | 4  |
| Install the software .....                          | 5  |
| Activate the license .....                          | 6  |
| View software version and license information ..... | 7  |
| Introduction to the application                     |    |
| USBPWR features and benefits .....                  | 8  |
| Application directories and their contents .....    | 8  |
| File name extensions .....                          | 10 |

## Operating basics

|  |    |
|--|----|
| Run the application .....              | 11 |
| Exit the application .....             | 11 |
| Application controls and menus         |    |
| Application controls .....             | 12 |
| Options menu                           |    |
| Options menu overview .....            | 13 |
| Instrument control settings .....      | 14 |
| Email settings                         |    |
| Email settings .....                   | 15 |
| Configure email settings .....         | 16 |
| Application panels                     |    |
| Application panel overview .....       | 18 |
| Setup panel                            |    |
| Setup panel overview .....             | 18 |
| Set DUT parameters .....               | 19 |
| Select tests .....                     | 20 |
| Acquisitions tab                       |    |
| Set acquisition parameters .....       | 21 |
| Set acquisition and save options ..... | 23 |

|  |    |
|--|----|
| Set acquisition waveform source for prerecorded waveform files ..... | 24 |
| Set test notification preferences .....                              | 25 |
| Configure test parameters  |    |
| About configuring test parameters .....                              | 25 |
| Configuration tab parameters .....                                   | 27 |
| Status panel overview .....  | 29 |
| Results panel  |    |
| Results panel overview .....   | 30 |
| View test-related files .....  | 32 |
| Reports panel  |    |
| Reports panel overview .....   | 33 |
| Select report options .....  | 33 |
| View a report .....  | 35 |
| Report contents .....  | 35 |

## Setting up and configuring tests

|                                  |    |
|----------------------------------|----|
| About setting up tests .....     | 39 |
| Equipment connection setup ..... | 40 |
| View connected instruments ..... | 41 |
| Test setup overview .....        | 43 |

## Running tests

|                                    |    |
|------------------------------------|----|
| About running tests .....          | 45 |
| Before you click start .....       | 45 |
| Map the My TekExpress folder ..... | 47 |
| Prerun checklist .....             | 48 |

## Saving and recalling test setups

|  |    |
|--|----|
| About test setups .....                                | 49 |
| Save a test setup .....                                | 49 |
| Open (load) a saved test setup .....                   | 50 |
| Create a new test setup based on an existing one ..... | 50 |

## TekExpress programmatic interface

|   |    |
|---|----|
| About the programmatic interface .....              | 51 |
| Requirements for developing TekExpress client ..... | 51 |
| Remote proxy object .....                           | 53 |
| Client proxy object .....                           | 53 |
| Client programmatic interface example .....         | 55 |
| Program remote access code example .....            | 59 |
| USBPWR application commands                         |    |
| USBPWR application commands flow .....              | 60 |

---

|   |    |
|---|----|
| Connect through an IP address .....                                 | 65 |
| Lock the server .....   | 66 |
| Disable the popups .....  | 67 |
| Set or get the DUT ID .....   | 68 |
| Set the configuration parameters for a suite or measurement .....   | 69 |
| Query the configuration parameters for a suite or measurement ..... | 70 |
| Select a test .....   | 71 |
| Select a suite .....  | 72 |
| Select a channel .....  | 73 |
| Configure the selected measurement .....                            | 74 |
| Run with set configurations or stop the run operation .....         | 75 |
| Handle error codes .....  | 76 |
| Get or set the timeout value .....                                  | 77 |
| Wait for the test to complete .....                                 | 78 |
| After the test is complete .....                                    | 80 |
| Save, recall, or query a saved session .....                        | 84 |
| Unlock the server .....   | 85 |
| Disconnect from the server .....                                    | 85 |

## Index

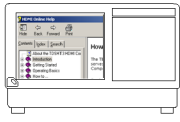






## Related documentation

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

**Table 1: Product documentation**

| Item                   | Purpose                                       | Location   |
|------------------------|---|--|
| Online Help            | In-depth operation and UI help                |   |
| PDF of the Online Help | Printable version of the compiled Online help |  + <br>www.Tektronix.com |

### See also

[Technical support \(see page 2\)](#)

## Conventions used in help

Online Help uses the following conventions:

- The term “DUT” is an abbreviation for Device Under Test.
- The term “select” is a generic term that applies to the two methods of choosing an option: using a mouse or using the touch screen.

## Technical support

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

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

### General information

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

### Application specific information

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

## Minimum system requirements

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

**Table 2: System requirements**

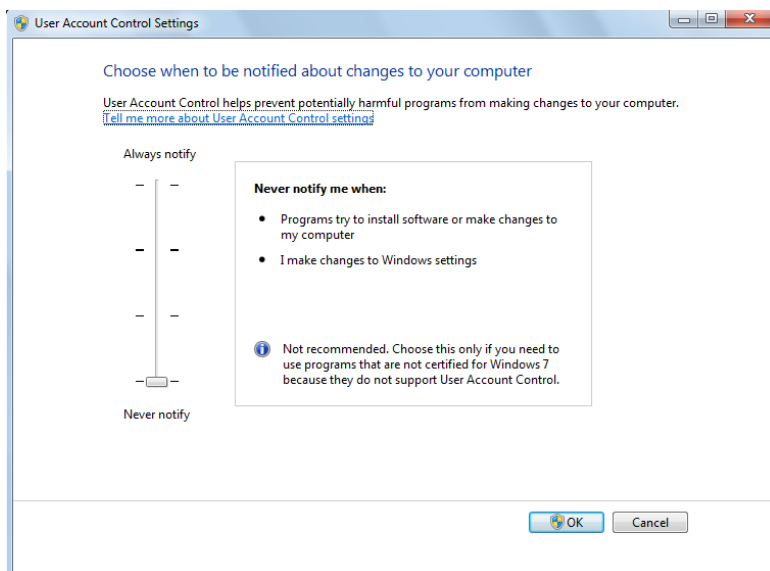
|                                |  |
|--------------------------------|--|
| Oscilloscope                   | See <a href="#">Supported oscilloscopes (see page 4)</a>   |
| Processor                      | Same as the oscilloscope   |
| Operating System               | Same as the oscilloscope: <ul style="list-style-type: none"> <li>■ Windows 7 (64-bit only) <a href="#">Windows 7 user account settings (see page 3)</a></li> </ul>   |
| Memory                         | Same as the oscilloscope   |
| Hard Disk                      | Same as the oscilloscope   |
| Display                        | Same as the oscilloscope <sup>1</sup>  |
| Firmware                       | <ul style="list-style-type: none"> <li>■ TekScope 6.4.5 and later (Windows 7)</li> </ul>   |
| Software                       | <ul style="list-style-type: none"> <li>■ Microsoft .NET 4.0 Framework</li> <li>■ Microsoft Internet Explorer 7.0 SP1 or later</li> <li>■ Microsoft Photo Editor 3.0 or equivalent software for viewing image files</li> <li>■ Adobe Reader 7.0 or equivalent software for viewing portable document format (PDF) files</li> </ul>  |
| <b>Recommended Accessories</b> | <ul style="list-style-type: none"> <li>■ One set of TPP0500, TPP1000, P6139B, or P5100A probe with standard accessories are recommended</li> <li>■ Additionally, one set of TCA to BNC converter and TCA-1MEG TekConnect 1 MΩ adapter is recommended when using a P6139B or P5100A probe with a DPO/DSA/MSO70000C/GSA Series Oscilloscope</li> </ul> <p>Please refer to <a href="http://www.tek.com/probes">www.tek.com/probes</a> for further information on recommended probes and probe adapters.</p> |

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

## 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.



### See also

[Supported oscilloscopes \(see page 4\)](#)

## Supported oscilloscopes

The TekExpress USBPWR application (Option USBPWR) is compatible with the following Tektronix oscilloscopes:

- MSO/DPO5000/GSA Series (350 MHz and above)
- DPO7000C/GSA Series
- DPO/DSA/MSO70000C/GSA Series

---

**NOTE.** *DPO/DSA/MSO70000D Series oscilloscopes are not supported for Option USBPWR.*

---

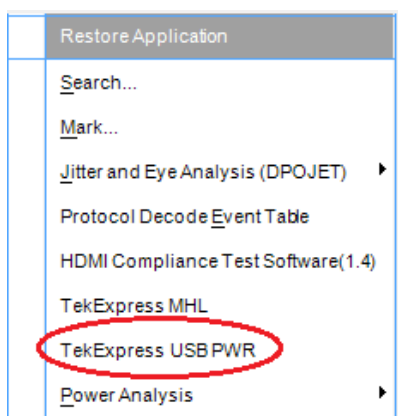
### See also

[Minimum system requirements \(see page 3\)](#)

## Install the software

Use the following steps to install the USBPWR application on any compatible instrument running Microsoft Windows 7 (64-bit). See [Minimum System Requirements \(see page 3\)](#) for details.

1. Close all applications (including the TekScope application).
2. Go to the [www.tek.com](http://www.tek.com) Web site and search for TekExpress USB PWR to locate the installation file. Download the file `TekExpress_USB_PWR_Deployment_Package.exe`.
3. Copy or download the USBPWR installer to the oscilloscope.
4. Double-click the installer .exe file to extract the installation files and launch the InstallShield Wizard. Follow the on-screen instructions.
5. The software installs in the following location:
  - `C:\Program Files\Tektronix\TekExpress\TekExpress USB-PWR`
6. The installer updates the TekScope Analyze menu to include the installed options.



### See also

[Minimum system requirements \(see page 3\)](#)

[Supported oscilloscopes \(see page 4\)](#)

## Activate the license

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

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

The TekScope Option Installation wizard opens.

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

### See also

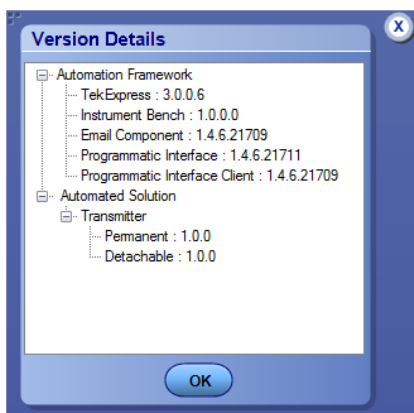
[View version and license information \(see page 7\)](#)

## View software version and license information

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

To view version information for USBPWR:

1. In the USBPWR application, click the **Options** button and select **About TekExpress**.
2. Click the View Version Details link to view the version numbers of the installed test suites.



To view license and option key information:

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

### See also

[Activate the license \(see page 6\)](#)

[Options menu \(see page 13\)](#)

## USBPWR features and benefits

Welcome to the TekExpress® USB Power Adapter / EPS Compliance Automated Test Solution Software (referred to as TekExpress USBPWR or USBPWR in the rest of the document). USBPWR provides an automated, simple, and efficient way to test USB power interfaces and devices consistent to the requirements of the USB2.0 Battery Charger Specification v1.1 and IEC 62684.

USBPWR is based on TekExpress version 2, the Tektronix Test Automation Framework, developed to support your current and future test automation needs. TekExpress uses a highly modular architecture that lets you deploy automated test solutions for various standards.

Key features and benefits of USBPWR include:

- Automates compliance measurements for USB2.0 Battery Charger Specification v1.1 and IEC 62684:
  - AC Voltage Frequency Component measurement
  - AC CommonMode Voltage measurement
  - Ripple Voltage measurement
  - Maximum Slew measurement
  - Reduces the time required to conduct testing
  - Minimizes user intervention when conducting time-consuming testing
- Provides individual or group test selection by using a tree-structure menu
- Built-in reporting features:
  - Provides a Pass/Fail summary table
  - Provides margin details on each test
  - Provides a consolidated report for all tests
- Complete programmatic interface enables automation scripts to call USBPWR functions

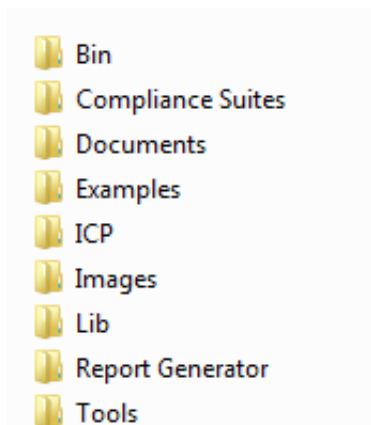
## Application directories and their contents

### TekExpress USBPWR application

The TekExpress USBPWR application files are installed at the following location:



- C:\Program Files\Tektronix\TekExpress\TekExpress USB-PWR



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

**Table 3: Application directories and usage**

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

### See also

[View test-related files \(see page 32\)](#)

[File name extensions \(see page 10\)](#)

## File name extensions

The USBPWR application uses the following file name extensions:

| File name extension | Description  |
|---------------------|--|
| .TekX               | The application session files (the extensions may not be displayed).   |
| .py                 | The test sequence file   |
| .xml                | The test-specific configuration information (encrypted) file<br>The application log file                         |
| .wfm                | The test waveform file   |
| .mht                | The test result reports (default). Test reports can also be <a href="#">saved in HTML format (see page 33)</a> . |

### See also

[Select report options \(see page 33\)](#)

[View test-related files \(see page 32\)](#)

[Application directories and their contents \(see page 8\)](#)

[Before you click start \(see page 45\)](#)

## Run the application

To run the USBPWR application, do either of the following:

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

When you first run the application after installation, the application checks for a file called `Resources.xml` located in the `C:\Users\\My Documents` folder. The `Resources.xml` file gets mapped to the X: drive when the application launches. Session files are then stored inside the `X:\USB PWR` folder. The `Resources.xml` file contains information about available network-connected instruments. If this file is not found, the application runs an instrument discovery program, before launching USBPWR, to locate available instruments.

To keep the application window on top, select **Keep On Top** from the USBPWR [Options menu \(see page 13\)](#). If the application goes behind the oscilloscope application, click **Analyze > TekExpress USB PWR** to move the application to be in front.

### See also

[Activate the license \(see page 6\)](#)


## Exit the application

Use the following method to exit the application:

---




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

---

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

## Application controls

Table 4: Application controls descriptions

| Item                                       | Description   |
|--|---|
| <a href="#">Options menu (see page 13)</a> | Menu to display global application controls   |
| <a href="#">Panels (see page 18)</a>       | Visual frames with sets of related options.   |
| Command buttons                            | Buttons that initiate an immediate action. Examples of command buttons are the Start, Stop, Pause, Continue, and Clear buttons.   |
| Start button                               | <p><b>Start</b></p>  <p>Use the Start button to start the execution of the measurements in the selected order. If prior acquired measurements have not been cleared, the new measurements are added to the existing set.</p> |
| Stop button                                | <p><b>Stop</b></p>  <p>Use the Stop button to abort the test.</p>  |
| Pause \ Continue button                    | <p><b>Pause</b>      <b>Continue</b></p>  <p>Use the Pause button to temporarily interrupt the current acquisition. When a test is paused, the button name changes to "Continue."</p>                                       |

**Table 4: Application controls descriptions (cont.)**

| Item                         | Description  |
|------------------------------|--|
| Clear button                 | <div data-bbox="878 296 971 359" data-label="Image"> </div> <p data-bbox="878 375 1453 588">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 <a href="#">Results panel (see page 30)</a>.</p> |
| Application window move icon | <div data-bbox="878 604 1060 758" data-label="Image"> </div> <p data-bbox="878 772 1453 865">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.</p>   |

## Options menu overview

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

The [Options menu \(see page 14\)](#) has the following selections:

| Menu  | Function  |
|---|---|
| Default Test Setup  | Opens an untitled test setup with defaults selected   |
| Open Test Setup   | Opens a saved test setup  |
| Save Test Setup   | Saves the current test setup selections   |
| Save Test Setup As  | Creates a new test setup based on an existing one   |
| Open Recent   | Displays a menu of recently opened test setups to select from   |
| <a href="#">Instrument Control Settings (see page 14)</a> | Shows the list of instruments connected to the test setup and allows you to locate and refresh connections to those instruments |
| Keep On Top   | Keeps the TekExpress USBPWR application on top of other open windows on the desktop   |
| <a href="#">Email Settings (see page 15)</a>              | Use to configure email options for test run and results notifications   |

| Menu             | Function   |
|------------------|--|
| Help             | Displays the TekExpress USBPWR Online help   |
| About TekExpress | <ul style="list-style-type: none"> <li>■ Displays application details such as software name, version number, and copyright</li> <li>■ Provides access to <a href="#">license information (see page 7)</a> for your USBPWR installation</li> <li>■ Provides a link to the Tektronix Web site</li> </ul> |



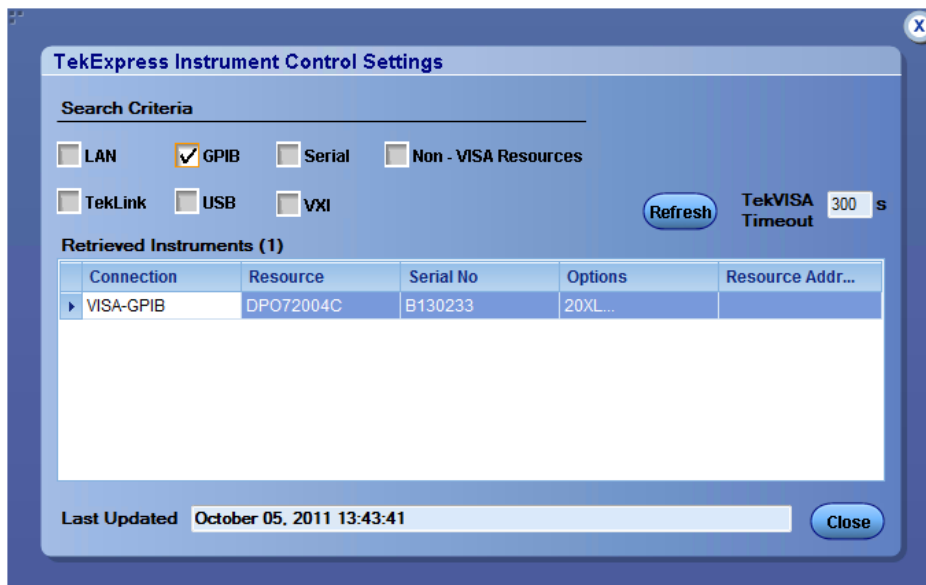
### See also

[Application controls \(see page 12\)](#)

## 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 each instruments connection information. You access this dialog box from the Options menu.

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



Use the Instrument Control Settings feature to [search for connected instruments \(see page 41\)](#) and view instrument connection details. Connected instruments displayed here can be selected for use under Global Settings in the test configuration section.

**See also**

[Options menu overview \(see page 13\)](#)

## Email settings

Use the Email Settings utility to [configure email notifications \(see page 16\)](#) to receive notifications when a test completes, produces an error, or fails. 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.

Access this dialog box from the Options menu.

---

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

---

### See also

[Configure email settings \(see page 16\)](#)

[Options menu \(see page 13\)](#)

[Select test notification preferences \(see page 25\)](#)

## Configure email settings

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

1. **Options > Email Settings** to open the [Email Settings \(see page 17\)](#) dialog box.
2. (Required) For Recipient email Address(es), enter one or more email addresses to which to send the test notification. To include multiple addresses, separate the addresses with commas.
3. (Required) For Sender's Address, enter the email address used by the instrument. This address consists of the instrument name followed by an underscore followed by the instrument serial number, then the @ symbol and the email server used. For example: DPO72016C\_B130099@yourcompany.com.
4. (Required) In the Server Configuration section, type the SMTP Server address of the Mail server configured at the client location, and the SMTP Port number, in the corresponding fields.

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

---

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

---



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

**Email settings**

**Email Settings**

Recipient e-mail Address(es)

*Note: Separate Email addresses with a comma*

Sender's Address

**Email Attachments**

Reports

ScoreCard

Analysis Screenshot

Status Log  Last 20 Lines  Full Log

**Server Configuration**

SMTP Server  SMTP Port

Login

Password

Host Name

**Email Configuration**

Email Format  HTML  Plain Text

Max Email Size (MB)

Number of Attempts to Send

Timeout

Email Test Results When complete or on error

Test Em... Apply Close

## Application panel overview

Panels group related configuration, test, and results settings.

The TekExpress USBPWR panels are:

**Table 5: Application panels**

| Panel Name                            | Purpose  |
|---------------------------------------|--|
| <a href="#">Setup (see page 18)</a>   | <p>The Setup panel shows the test setup controls. Click the <b>Setup</b> button to open this panel.</p> <p>Use this panel to:</p> <ul style="list-style-type: none"> <li>■ <a href="#">Select DUT parameters (see page 19)</a>.</li> <li>■ <a href="#">Select the test(s) (see page 20)</a>.</li> <li>■ <a href="#">Set acquisitions parameters (see page 21)</a> for selected tests.</li> <li>■ <a href="#">Configuration test parameters (see page 27)</a></li> <li>■ <a href="#">Select test notification preferences (see page 25)</a>.</li> </ul> |
| <a href="#">Status (see page 29)</a>  | View the progress and analysis status of the selected tests, and view test logs.   |
| <a href="#">Results (see page 30)</a> | View a summary of test results and select result viewing preferences.  |
| <a href="#">Reports (see page 33)</a> | Browse for reports, save reports as specific file types, specify report naming conventions, select report content to include (such as summary information, detailed information, user comments, setup configuration, application configuration), and select report viewing options.  |

### See also

[Application controls \(see page 12\)](#)  
[about setting up tests \(see page 39\)](#)

## Setup panel overview

The [Setup panel \(see page 19\)](#) contains sequentially ordered tabs that help guide you through a typical test setup process. Use the tabs on this panel to:

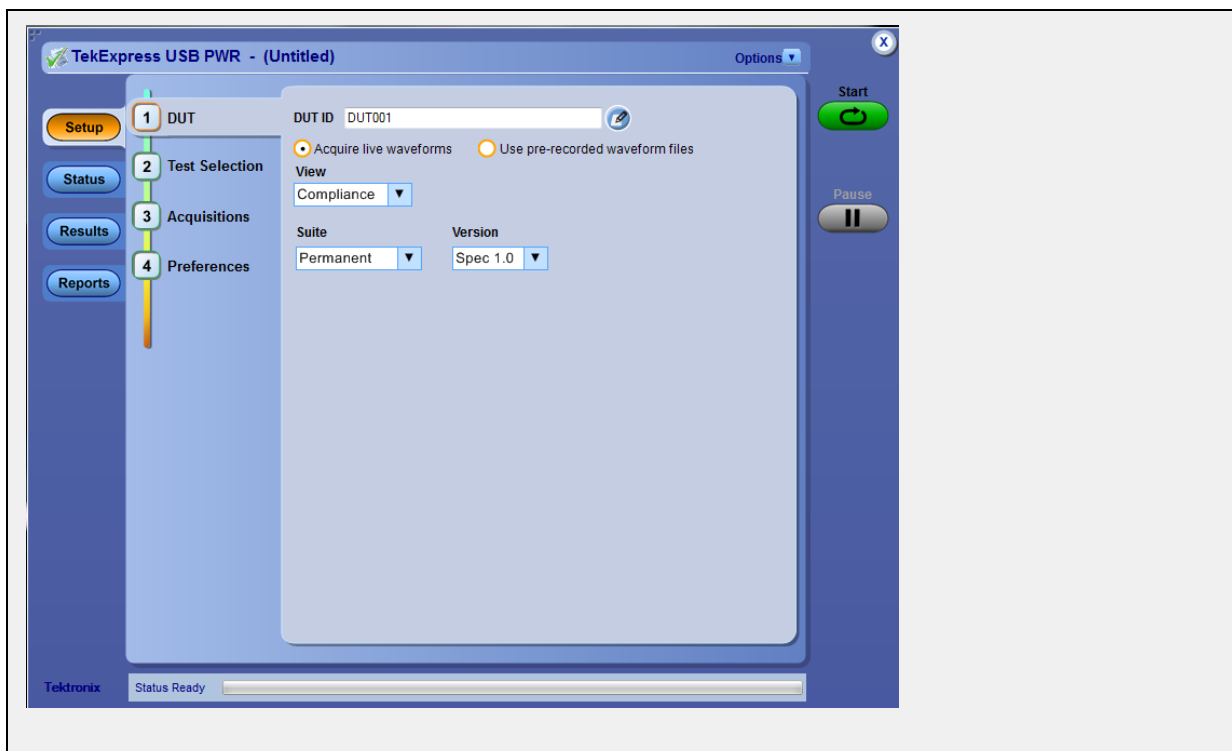
[Set the DUT parameters \(see page 19\)](#)

[Select test\(s\) \(see page 20\)](#)

[Select acquisition parameters \(see page 21\)](#)

[Set configuration parameters \(see page 27\)](#)

[Select test notification preferences \(see page 25\)](#)



## Set DUT parameters

Use the [Setup panel \(see page 19\)](#) 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 6: DUT tab settings**


| Setting  | Description  |
|--|--|
| DUT ID   | Adds an optional text label for the DUT to reports. The default value is DUT001.   |
|  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 <a href="#">Select report options (see page 33)</a> .) |
| Acquire live waveforms   | Acquire active signals from the DUT for testing.   |
| Use prerecorded waveform files   | Run tests on a saved waveform. <a href="#">Open (load) a saved test setup (see page 50)</a>  |
| Version  | Sets the DUT generation version.   |

Table 6: DUT tab settings (cont.)

| Setting       | Description  |
|---------------|--|
| Specification | USBPWR currently supports USB2.0 Battery Charger Specification v1.1 and IEC 62684.   |
| View          | <p>Sets the overall testing mode. Select Compliance or Advanced:</p> <ul style="list-style-type: none"> <li>■ Compliance: Preselects tests and parameters to meet compliance specifications for the selected version, specification, and device type.<br/>View configuration settings by clicking Setup &gt; Test Selection &gt; Configure.</li> <li>■ Advanced: Enables the user to select specific tests and set custom parameters for tests.<br/>This also enables the Configuration tab on the Setup panel to quickly access test parameters.</li> </ul> |
| Suite         | Sets the DUT device type (Permanent cable or Detachable type)  |

### See also

[About setting up tests \(see page 39\)](#)

[Select a test \(see page 20\)](#)

## Select tests

Use the **Test Selection** tab to select the tests to run on the connected DUT.

1. Click **Setup > Test Selection**.

2. Select the test(s) to run:

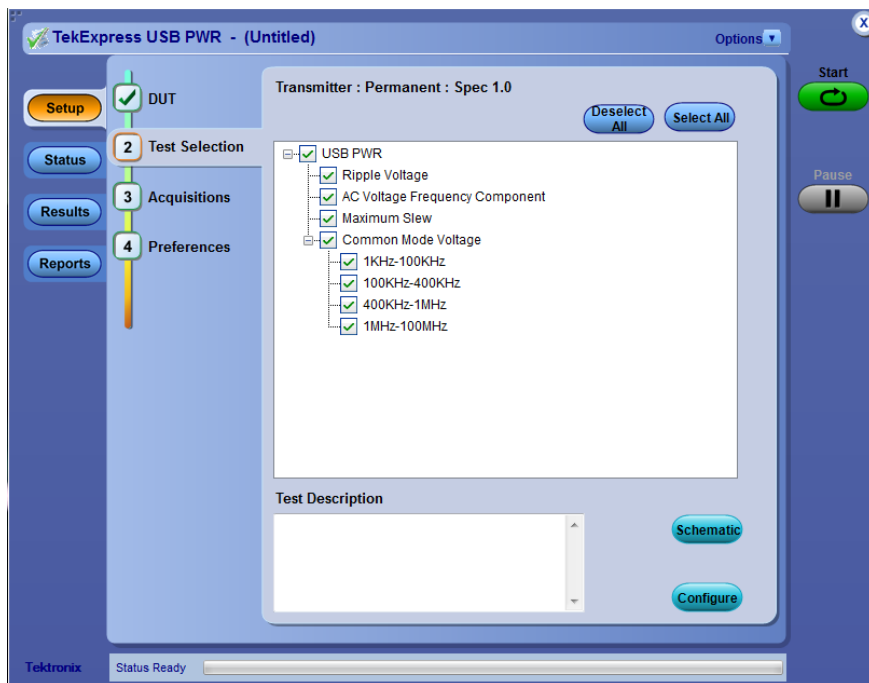
- Click + to expand a group of commands. Click the check box adjacent to a test group to select all tests in that group. Click check boxes adjacent to individual tests to select those tests.
- Click **Deselect All** to deselect all tests. All tests are selected by default.
- Click **Select All** to select all tests.
- Click **Schematic** to view a diagram that shows the correct DUT and equipment setup for the selected test. Use to verify your test equipment setup before running the test.
- Click **Configure** to open the configuration settings for a selected test.

---

**NOTE.** The **Configure** button is not displayed if the View (in the DUT tab) is set to Advanced.

---

- Click **Schematic** to display a schematic diagram that shows the DUT test setup. Use the diagram to verify the test setup before running the test.



### See also

[Set acquisition parameters \(see page 21\)](#)

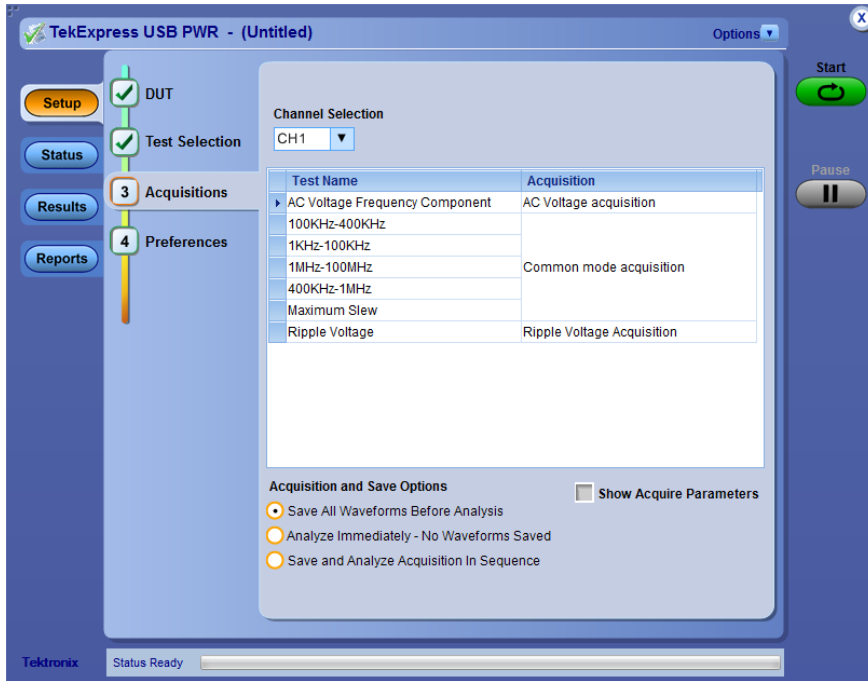
[About setting up tests \(see page 39\)](#)

## Set acquisition parameters

Use the **Acquisition** tab in the Setup panel to view and select test acquisition parameters, such as the signal source channel and waveform save options.

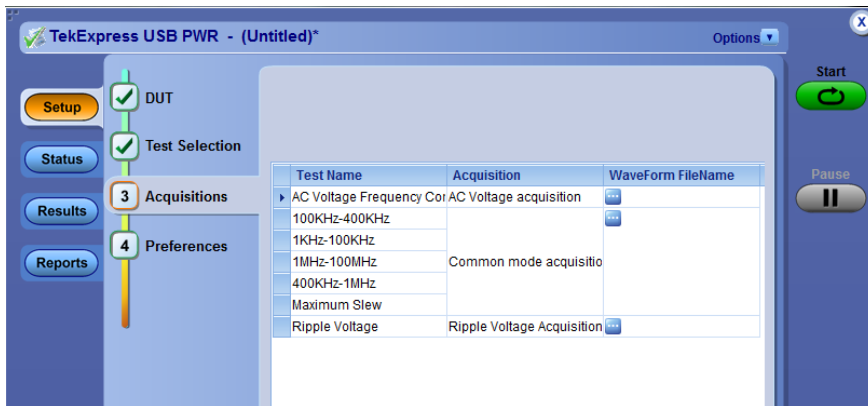
- Click on **Channel Selection** to select the channel to which the DUT has been connected. (Channel Selection is only available when acquiring live waveforms.)
- Click the **Acquisition and Save Options** (see page 23) controls to set how the application acquires and analyzes signals (Save all waveforms before analysis, Analyze immediately without saving waveforms, or Save and Analyze acquisitions in sequence).
- Click on **Show Acquire Parameters** to display the acquisition parameters for each acquisition.

### Active waveforms



Acquisitions tab: using active waveforms

### Prerecorded waveforms



Acquisitions tab: using prerecorded waveforms

When using prerecorded waveform files, this panel lists available prerecorded waveform files. You can only select the source of the prerecorded waveform file for each test. See [Set acquisition waveform source for prerecorded waveform files \(see page 24\)](#).

## Set acquisition and save options

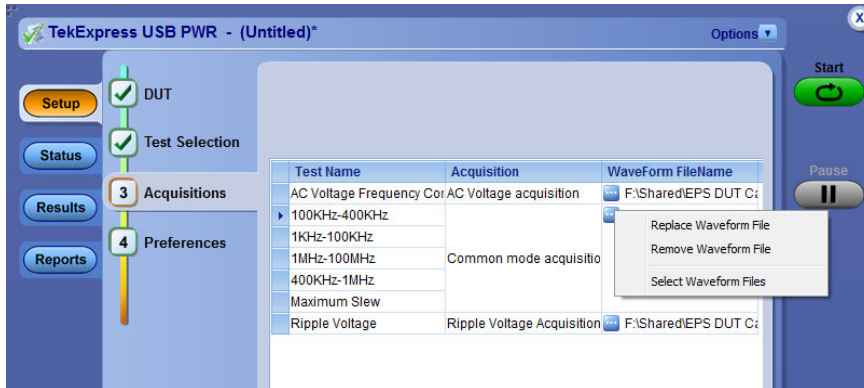
Select an **Acquisition and Save Options** to set the order in which waveforms are acquired and analyzed:

- **Save all waveforms before analysis:** Acquire & save all waveforms required by tests before performing analysis. All required user interventions (such as connecting the setup in different setups) are completed, and waveforms acquired & saved, before the analysis is run. You can turn off the DUT after the acquisitions are completed.
- **Analyze immediately - No waveforms saved:** Waveforms are acquired but not saved. For each type of acquisition, analysis is done just after acquisition. Waveforms are discarded when the analysis is complete.
- **Save and analyze acquisitions in sequence:** Acquire and save waveforms and analyze for each test before proceeding to the next test. Use this setting to stop the testing when an error occurs, investigate and correct DUT, instrument connections, or change application settings. Restart testing after any changes.

## Set acquisition waveform source for prerecorded waveform files

Select a **Save Option** to set how to save acquired test waveforms.

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



If you selected to use a prerecorded waveform file (in the DUT tab), the Acquisition tab shows a table of the waveforms used for the required test acquisitions.

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

1. Click the ellipsis button (⋮) of the waveform file to change.
2. Select the waveform task to perform (replace, remove, or select the waveform file).
3. Use the dialog box to navigate to and select the waveform file with which to replace the current file.

### See also

[Set acquisition options \(see page 23\)](#)



## Set test notification preferences

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

1. Click **Setup > Preferences**.
2. Select **On Test Failure, stop and notify me of the failure** to stop the test and send an email when a test fails. Click **Email Settings** to verify that **Email Test Results when complete or on error** is selected, and verify the email address.

### See also

[About setting up tests \(see page 39\)](#)

[Select report options \(see page 33\)](#)

## About configuring test parameters

Use the configuration settings to view the measurement parameters for selected tests. How the test configurations are accessed depend on the View selected in the DUT tab.

- If you set the view to **Compliance** in the DUT tab, then in the **Test Selection** tab, select the desired test in the list and then click the **Configure** button.

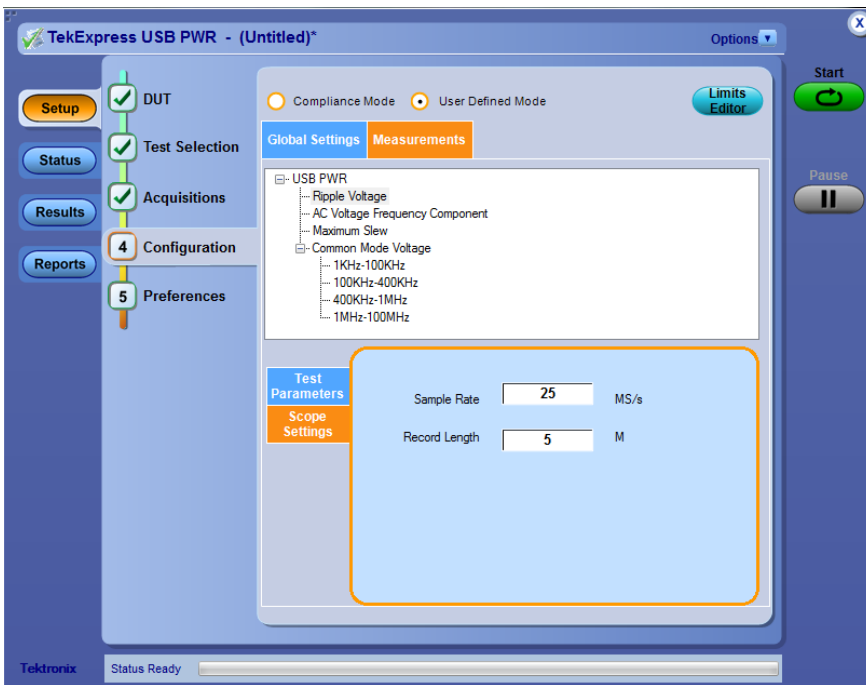
Use the **Test Selection** button to return to the Test selection display.

- If you set the view to **Advanced** in the DUT tab, click the **Configuration** tab in the Setup panel.

---

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

---



**See also**

[Configuration tab parameters \(see page 27\)](#)

[About setting up tests \(see page 39\)](#)

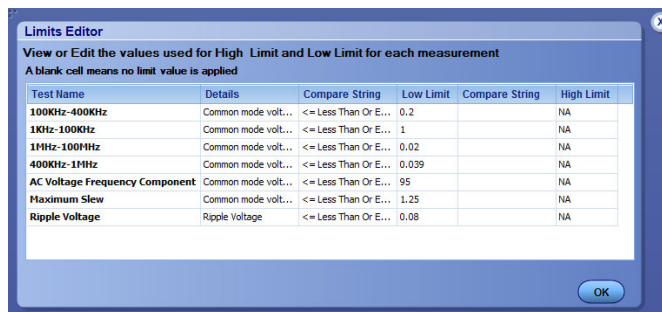
[About running tests \(see page 45\)](#)

## Configuration tab parameters

The following table lists the Configuration tab settings and parameters.

**Table 7: Common parameters and values**

| Parameter Type               | Parameter and Default Value   |
|------------------------------|---|
| Compliance/User Defined Mode | <p>Determines whether test parameters are in compliance or can be edited (User Defined Mode).</p> <ul style="list-style-type: none"> <li>■ Compliance: Most test parameter values cannot be edited.</li> <li>■ User Defined: Enables editing of most test parameters.</li> </ul>                                  |
| Limits Editor                | <p>Shows the upper and lower limits for the applicable measurement using different types of comparisons.</p> <p>In Compliance Mode, use the Limits Editor to view the measurement high and low limits used for selected tests.</p> <p>In User Defined Mode, use the Limits Editor to edit the limit settings.</p> |
| Instruments Detected         | <p>Displays a list of the connected instruments found during the instrument discovery. Instrument types include equipment such as oscilloscopes and signal generators. Select <b>Instrument Control Settings</b> to <a href="#">refresh the connected instrument list (see page 14)</a>.</p>                      |



To edit a value, click that field and either select from the displayed list or enter a new value. Use the bottom scroll bar to view all available fields.

**Table 7: Common parameters and values (cont.)**

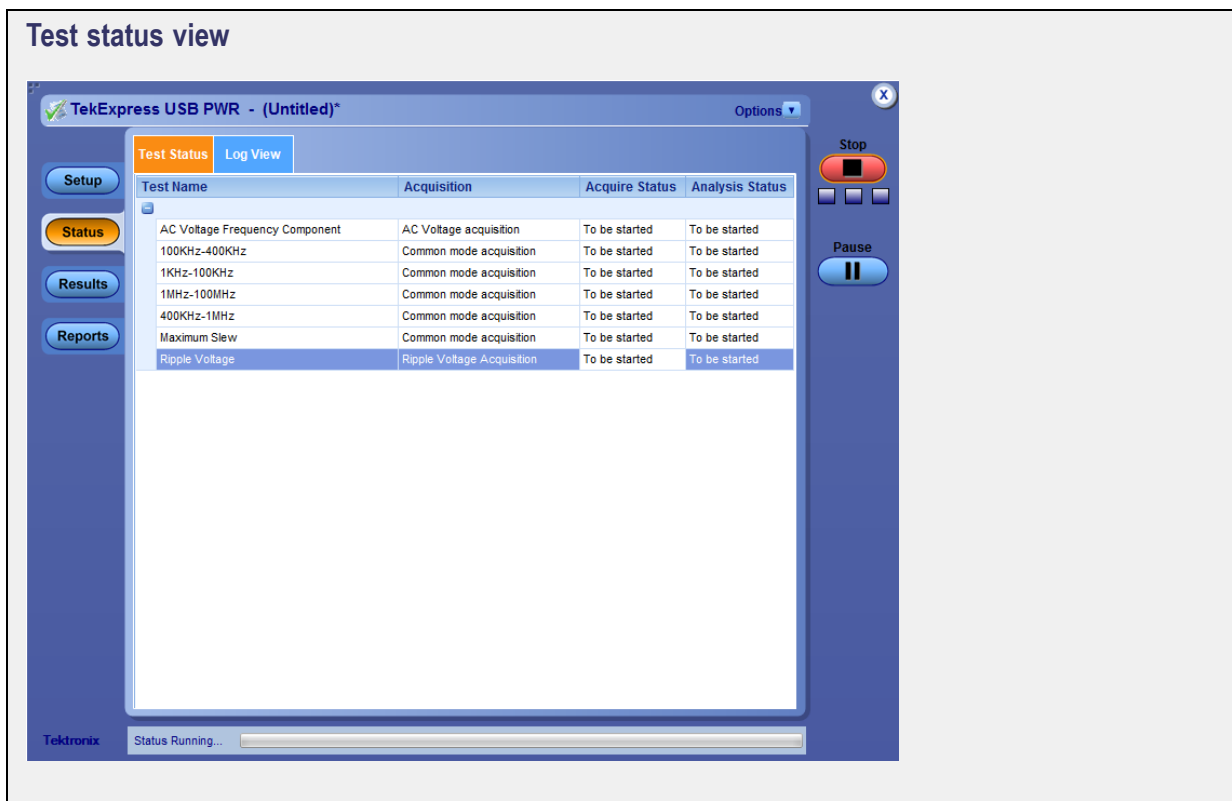
| <b>Parameter Type</b>                | <b>Parameter and Default Value</b>   |
|--------------------------------------|--|
| Test Parameters (for Ripple voltage) | To select the parameter values for AC Frequency, AC Voltage, Load & Temperature for Ripple test needs.   |
| Record Length, Sample Rate           | These settings apply to all tests selected for the indicated data rate. <ul style="list-style-type: none"><li>■ Record Length: Specifies the waveform record length.</li><li>■ Sample Rate: Specifies the oscilloscope sample rate to use for all tests.</li></ul> |

**See also**

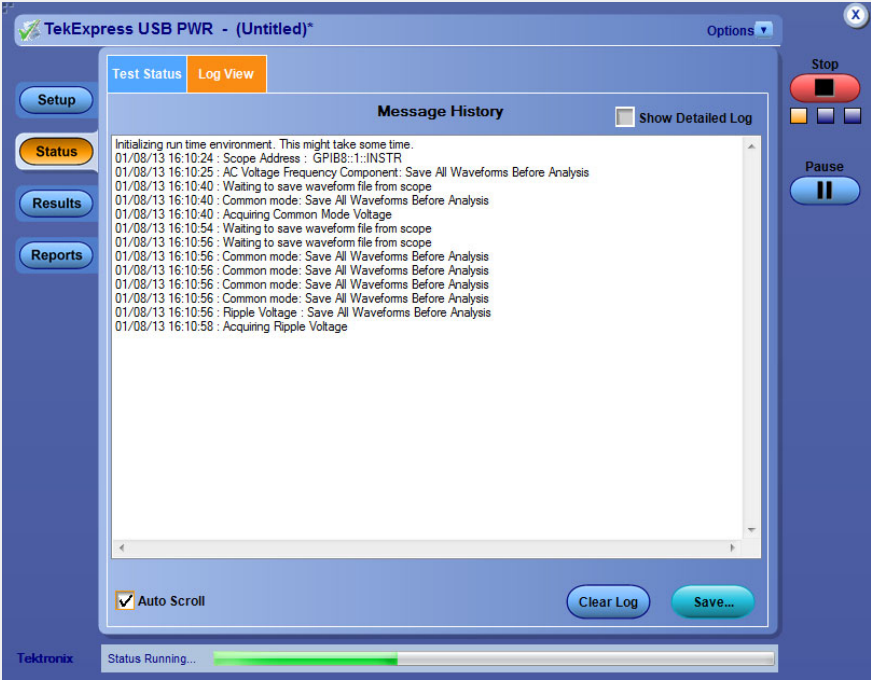
[About acquisitions \(see page 21\)](#)

## Status panel overview

The Status panel provides status on test acquisition and analysis ([Test Status \(see page 29\)](#) tab) and a listing of test tasks performed ([Log View \(see page 30\)](#) 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.



### Log view



The Log View display has several viewing options:

- Message History: This window timestamps and displays all run messages.
- Show Detailed Log: Select this check box to record a detailed history of test execution. This must be checked before starting a measurement.
- Auto Scroll: Select this check box to have the program automatically scroll down as information is added to the log during the test.
- Clear Log: Click this button to clear all messages from the display.
- Save: Click this button to save the log file as a text file. A standard Save File window is displayed to name and save the file.

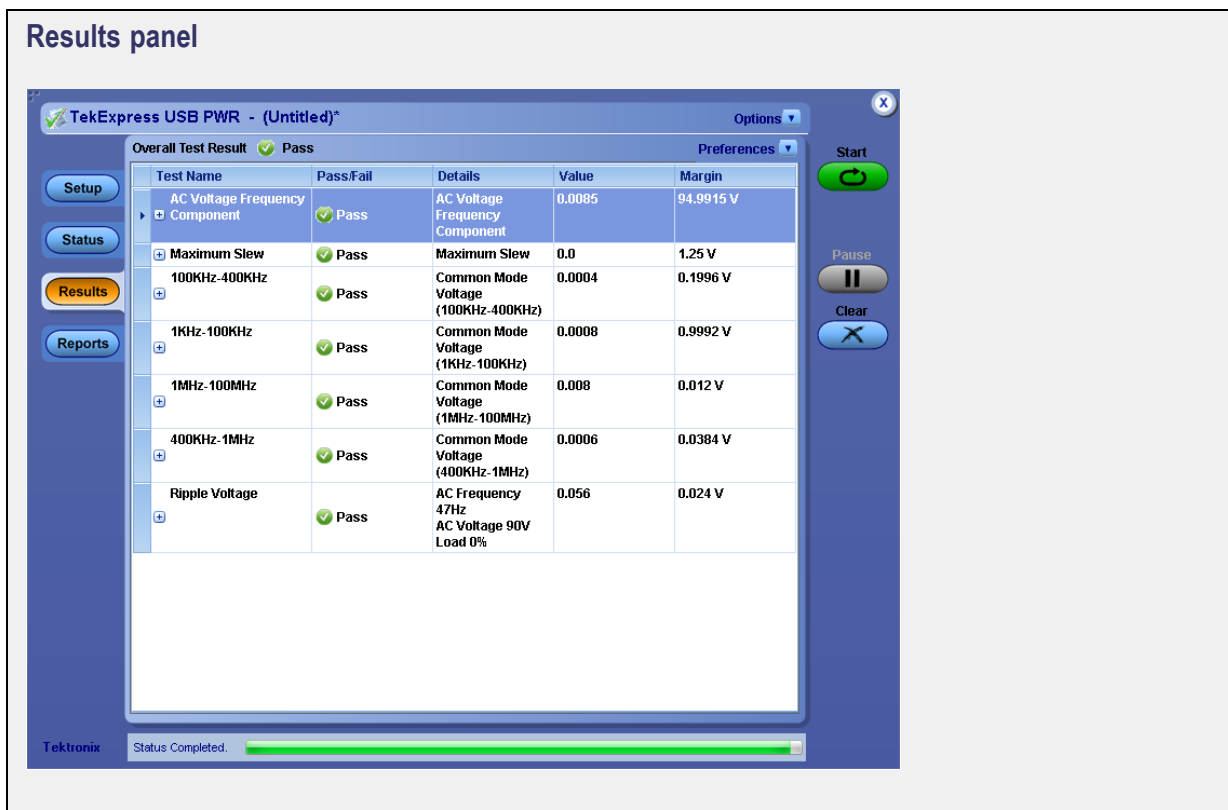
#### See also

[Application panel overview \(see page 18\)](#)

## Results panel overview

When a test finishes, the application switches to the [Results panel \(see page 31\)](#) to display a summary of signal and preset test results. The Overall Test Result is displayed at the top left of the Results table. If all of the tests for the session pass, the overall test result is **Pass**. If one or more tests fail, the overall test result is **Fail**.

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



When a test finishes, the application switches to the [Results panel \(see page 30\)](#), which displays a summary of test results.

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

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

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

- To expand the width of a column, place the cursor over the vertical line that separates the column from the column to the right. When the cursor changes to a double-ended arrow, hold down the mouse button and drag the column to the desired width.
- To clear all test results displayed, click **Clear**.

### See also

[View a report \(see page 35\)](#)

[About panels \(see page 18\)](#)

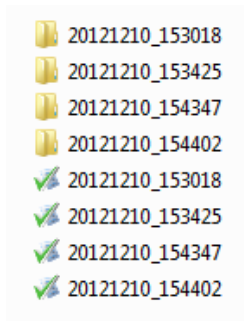
## View test-related files

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

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

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

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



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

The first time you run a new, unsaved session, the session files are stored in the `Untitled Session` folder located at `.\My TekExpress\USB PWR`. 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 USBPWR application.

### See also

[File name extensions \(see page 10\)](#)

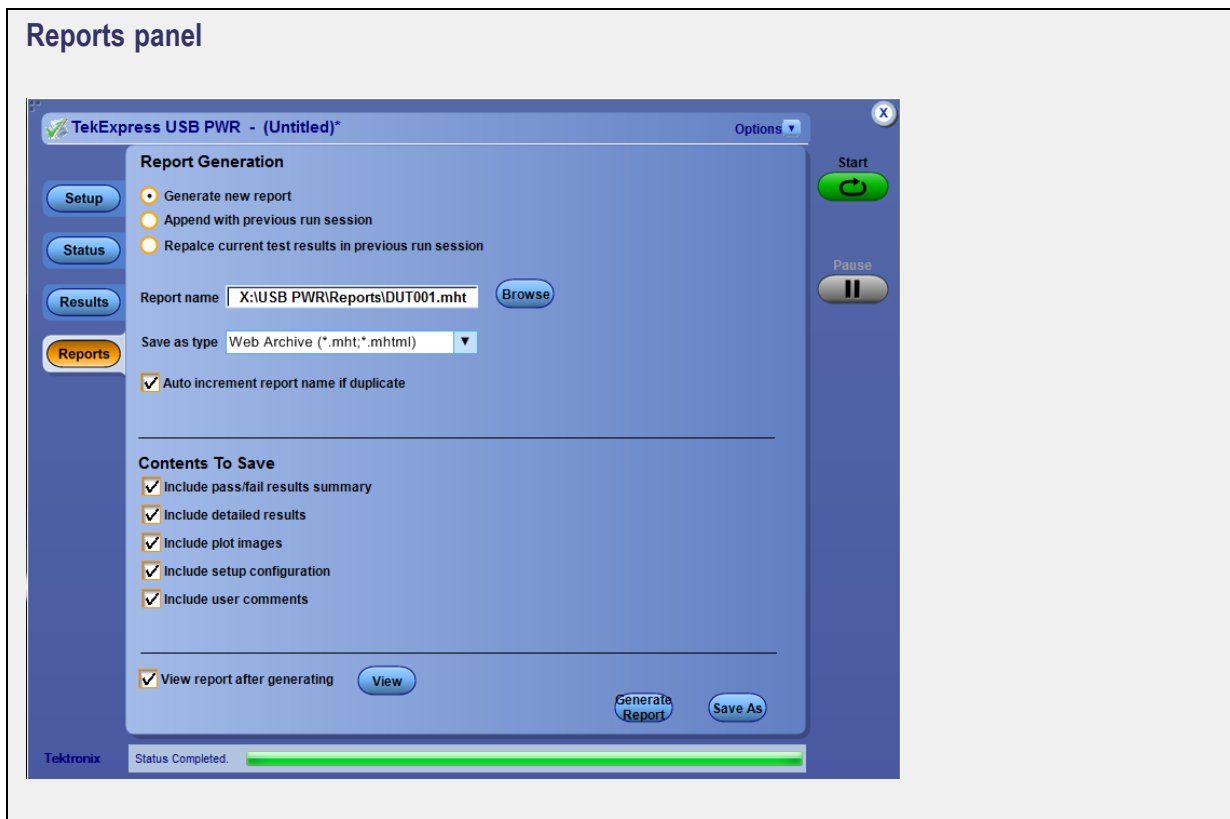
[Before you click start \(see page 45\)](#)



## Reports panel overview

Use the [Reports panel \(see page 33\)](#) to browse for reports, name and save reports, select report content to include, and select report viewing options.

For information on setting up reports, see [Select report options \(see page 33\)](#). For information on viewing reports, see [View a Report \(see page 35\)](#).



### See also

[About panels \(see page 18\)](#)

## Select report options

Use the [Reports panel \(see page 33\)](#) to select which test 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. Generally, you would 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 8: Report options**

| Setting  | Description   |
|--|---|
| Generate new report                              | Creates a new report.   |
| Replace current test values with the new results | Replaces the previous test results with the latest test results. Newly added tests results are appended to the end of the report.   |
| Append to previous report                        | Appends the latest test results to the end of the current test results report.  |
| Report Path                                      | <p>Displays the name and location from which to open a report. The default location is at <i>My TekExpress\USB PWR\Untitled Session</i>. 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.</p> <p>Do one of the following:</p> <ul style="list-style-type: none"> <li>■ In the Report Path field, type over the current folder path and name.</li> <li>■ Double-click in the Report Path field and then make selections from the popup keyboard and click the <b>Enter</b> button.</li> </ul> <p>Be sure to include the entire folder path, the file name, and the file extension. For example: C:\Documents and Settings\your user name\My Documents\My TekExpress\USB PWR\DUT001_Test_72.7.1.3.mht.</p> <p><b>NOTE.</b> You cannot set the file location using the Browse button.</p> <p>Open an existing report.</p> <p>Click <b>Browse</b>, locate and select the report file and then click <b>View</b> at the bottom of the panel.</p> |
| Save As Type                                     | <p>Saves a report in the specified file type. Lists supported file types to choose from.</p> <p><b>NOTE.</b> 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.</p>  |
| View Report After Generating                     | Automatically opens the report in a Web browser when the test completes. This option is selected by default.  |
| Include Pass/Fail Results Summary                | Sets the application to include the color block labeled Test Result (indicating whether the test passed or failed) in the report. For details, see Report Contents in <a href="#">View a report (see page 35)</a> .   |

**Table 8: Report options (cont.)**

| Setting                     | Description   |
|-----------------------------|---|
| Include Detailed Results    | Sets the application to include parameter limits and test-specific comments generated during the test.  |
| Include Plot Images         | Sets the application to include plotted diagrams.   |
| Include Setup Configuration | Sets the application to include hardware and software information in the summary box at the top of the report. Information includes: the oscilloscope model and serial number, probe model and serial number, the oscilloscope firmware version, SPC and factory calibration status, and software versions for applications used in the measurements. |
| Include User Comments       | Select to include any comments about the test that you or another user added in the DUT tab of the Setup panel.   |

**See also**

[View a report \(see page 35\)](#)

[About setting up tests \(see page 39\)](#)

## View a report

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

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

---

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

---

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

For information on changing the file type, file name, and other report options, see [Select report options \(see page 33\)](#).

## Report contents

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

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

Setup configuration information

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



**TekExpress USB PWR**

**Test Report**

| Setup Information                                |                                      |
|--|--------------------------------------|
| DUT ID : DUT001                                  | Scope Model : MSO72004C              |
| Suite Type : Detachable                          | Scope Serial No. : C230691           |
| MOI/CTS/Spec Version : Spec 1.0                  | Scope F/W Version : 6.4.0 Build 8    |
| Date/Time : 10-12-2012 20:57:52                  | Scope Calibration Status : PASS;PASS |
| Overall Execution Time : 1 min 6 sec             | TekExpress App Version : 1.0.0.49    |
| Overall Test Result : <b>Pass</b>                | TekExpress F/W Version : 3.0.0.6     |
| DUT Comment :General Comment - USB PWR Cable DUT |                                      |

| Test Name:Summary Table        |      |
|--------------------------------|------|
| 100KHz-400KHz                  | Pass |
| 1KHz-100KHz                    | Pass |
| 1MHz-100MHz                    | Pass |
| 400KHz-1MHz                    | Pass |
| AC Voltage Frequency Component | Pass |
| Maximum Slew                   | Pass |
| Ripple Voltage                 | Pass |

Test result summary

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

| 100KHz-400KHz                       |                |             |          |           |            |          |
|-------------------------------------|----------------|-------------|----------|-----------|------------|----------|
| Measurement Details                 | Measured value | Test Result | Margin   | Low Limit | High Limit | Comments |
| Common Mode Voltage (100KHz-400KHz) | 0.0004         | Pass        | 0.1996 V | NA        | 0.2        |          |

[Back To Summary Table](#)

| 1KHz-100KHz                       |                |             |          |           |            |          |
|-----------------------------------|----------------|-------------|----------|-----------|------------|----------|
| Measurement Details               | Measured value | Test Result | Margin   | Low Limit | High Limit | Comments |
| Common Mode Voltage (1KHz-100KHz) | 0.0008         | Pass        | 0.9992 V | NA        | 1.0        |          |

[Back To Summary Table](#)

| 1MHz-100MHz                       |                |             |          |           |            |          |
|-----------------------------------|----------------|-------------|----------|-----------|------------|----------|
| Measurement Details               | Measured value | Test Result | Margin   | Low Limit | High Limit | Comments |
| Common Mode Voltage (1MHz-100MHz) | 0.0078         | Pass        | 0.0122 V | NA        | 0.02       |          |

[Back To Summary Table](#)

| 400KHz-1MHz                       |                |             |          |           |            |          |
|-----------------------------------|----------------|-------------|----------|-----------|------------|----------|
| Measurement Details               | Measured value | Test Result | Margin   | Low Limit | High Limit | Comments |
| Common Mode Voltage (400KHz-1MHz) | 0.0006         | Pass        | 0.0384 V | NA        | 0.039      |          |

[Back To Summary Table](#)

| AC Voltage Frequency Component |                |             |           |           |            |          |
|--------------------------------|----------------|-------------|-----------|-----------|------------|----------|
| Measurement Details            | Measured value | Test Result | Margin    | Low Limit | High Limit | Comments |
| AC Voltage Frequency Component | 0.0086         | Pass        | 94.9914 V | NA        | 95.0       |          |

### See also

[Results panel overview \(see page 30\)](#)

[View test-related files \(see page 32\)](#)



## About setting up tests

Set up tests using the tabs in the [Setup panel \(see page 18\)](#).

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

### See also

[Test setup overview \(see page 43\)](#)

[Before you click start \(see page 45\)](#)

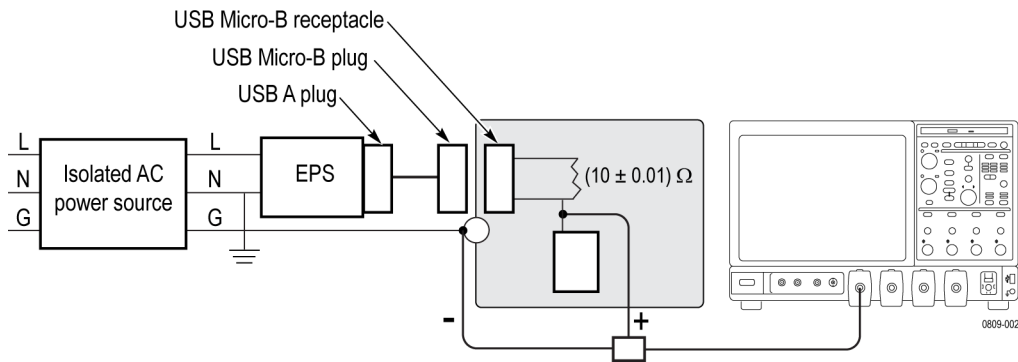
[About test setups \(see page 49\)](#)

[About running tests \(see page 45\)](#)

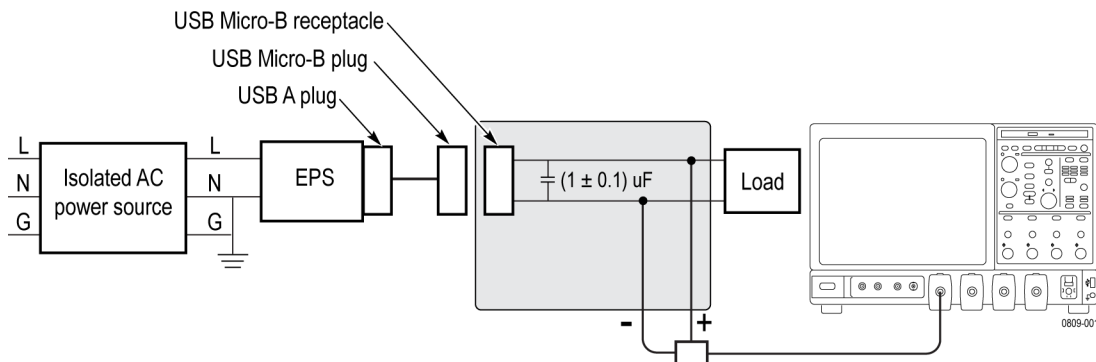
## Equipment connection setup

The following diagrams shows how to connect the DUT to the oscilloscope for the measurements.

### Common Mode connection



### Ripple Voltage connection



### See also

[Minimum system requirements \(see page 3\)](#)

[View connected instruments \(see page 41\)](#)

[About setting up tests \(see page 39\)](#)



## View connected instruments

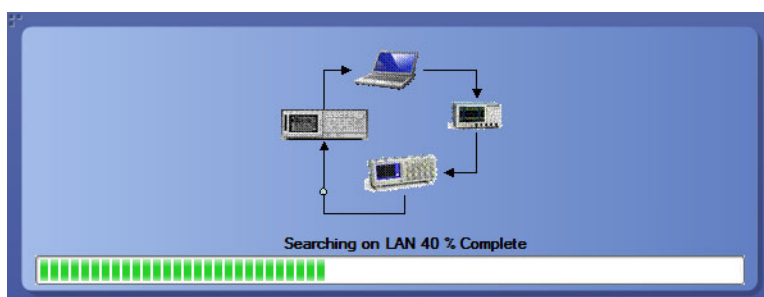
Use the Instrument Control Settings dialog box to view or search for connected instruments required for the tests. The application uses TekVISA to discover the connected instruments.

To refresh the list of connected instruments:

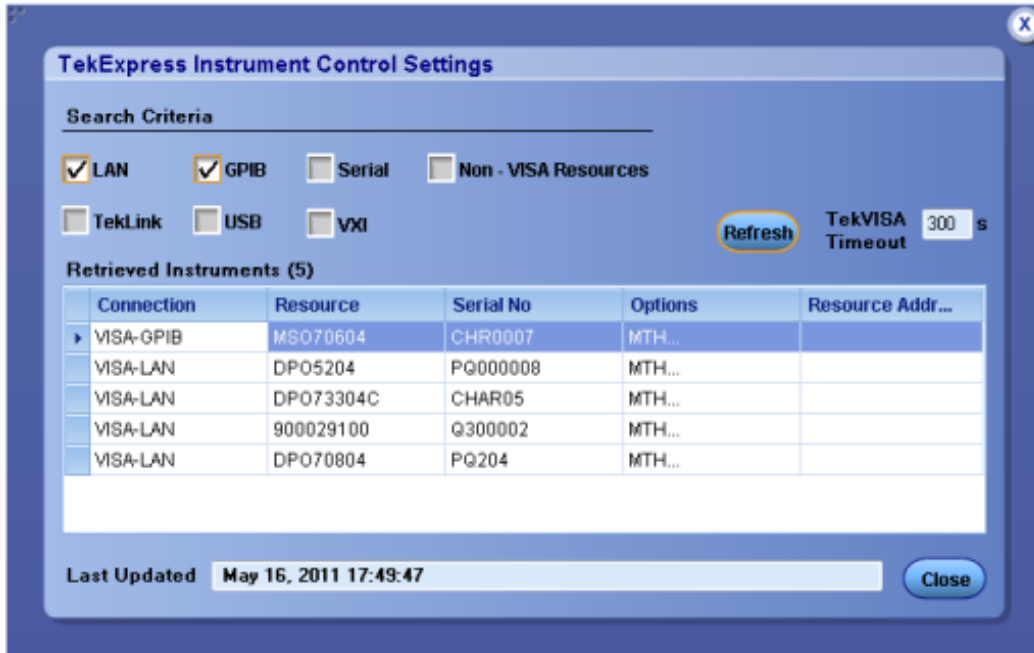
1. From the Options menu, select **Instrument Control Settings**.
2. In the Search Criteria section of the Instrument Control Settings dialog box, select the connection types of the instruments for which to search.

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

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



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



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

### See also

[Configuration test parameters \(see page 27\)](#)

[Equipment connection setup \(see page 40\)](#)

## Test setup overview

Test setup includes acquisition and configuration parameters. You can also select report options when setting up tests. Use the options in the [Setup panel \(see page 18\)](#) and [Reports panel \(see page 33\)](#) to select and configure tests.

1. [Set DUT parameters \(see page 19\)](#).
2. [Select one or more tests \(see page 20\)](#).
3. [Select acquisitions \(see page 21\)](#).
4. [Configuration test parameters \(see page 27\)](#).
5. [Set test measurement notification options \(see page 25\)](#).
6. [Select report options \(see page 33\)](#).

### See also

[About test setups \(see page 49\)](#)

[Prerun checklist \(see page 48\)](#)

[Before you click start \(see page 45\)](#)

[About running tests \(see page 45\)](#)



## About running tests

After selecting and configuring tests, review the [prerun checklist \(see page 48\)](#) and then click **Start** to run the tests. While tests are running, you cannot access the Setup or Reports panels. To monitor the test progress, switch back and forth between the Status panel and the Results panel.

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

### See also

[Before you click start \(see page 45\)](#)

[About configuring tests \(see page 25\)](#)

[About setting up tests \(see page 39\)](#)

## Before you click start

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

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

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

- \My Documents\My TekExpress\USB PWR
- \My Documents\My TekExpress\USB PWR\Untitled Session

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



---

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

---

2. [Map the shared My TekExpress folder \(see page 47\)](#) as **X:** (X drive) on the instruments used in test setups running Microsoft Windows Operating System.

The My TekExpress folder has the share name format <domain><user ID>My TekExpress. Or, if the instrument is not connected to a domain, the share name format is <instrument name><user

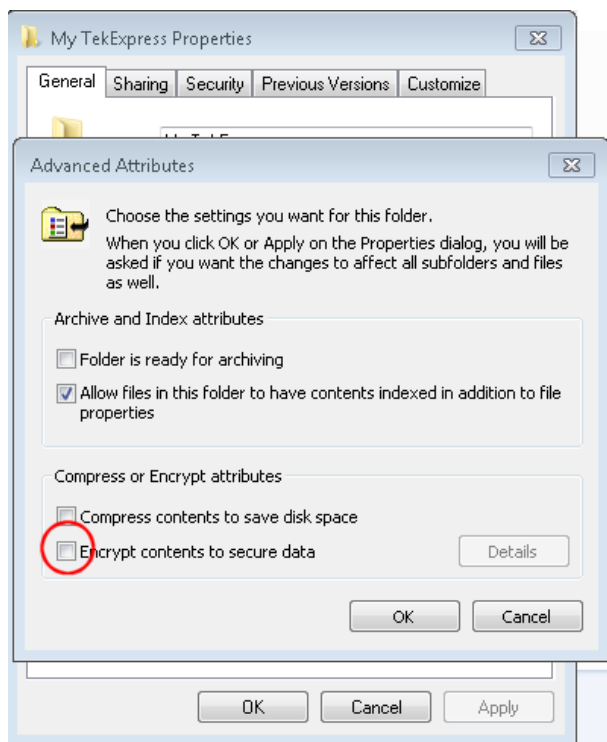
ID>My TekExpress. This shared folder is used to save the waveform files and is used during other file transfer operations.

---

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

---

3. Make sure that the My TekExpress folder has read and write access, and that the contents are not set to be encrypted:
  - a. Right-click the folder and select **Properties**.
  - b. Select the **General** tab and then click **Advanced**.
  - c. In the Advanced Attributes dialog box, make sure that the option **Encrypt contents to secure data** is NOT selected. Example.



4. See the [prerun checklist \(see page 48\)](#) before you run a test.

### See also

[Configuration test parameters \(see page 27\)](#)

[View test-related files \(see page 32\)](#)

[Application directories and usage \(see page 8\)](#)

[File name extensions \(see page 10\)](#)

## Map the My TekExpress folder

In the case where you operate the TekExpress application on one oscilloscope, but acquire data from another (remotely-accessed) oscilloscope, you need to share and map the My TekExpress folder on the remote instrument with the USBPWR application.

To map the My TekExpress folder on a remote instrument:

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

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

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

## Prerun checklist

Do the following before you click Start to run a test. If this is the first time you are running a test on a setup, refer to the information in [Before you click start \(see page 45\)](#).

1. Make sure that all the required instruments are properly warmed up (approximately 20 minutes).
2. Perform Signal Path Compensation (SPC).
  - a. On the oscilloscope main menu, select the **Utilities** menu.
  - b. Select **Instrument Calibration**.
3. Verify that the application is able to find the DUT. If it cannot, [perform a search for connected instruments \(see page 41\)](#).
  - a. In USBPWR, select the **Setup** panel and then click the **Test Selection** tab.
  - b. Select any test and then click **Configure**.
  - c. In the Configuration section, click **Global Settings**.
  - d. In the **Instruments Detected** section, click the drop-down arrow to the right of **Real Time Scope** and make sure that the oscilloscope with the (GPIB8::1::INSTR) designation is in the list.

### See also

[Equipment connection setup \(see page 40\)](#)



## About test setups

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

Use test setups to:

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

### See also

[About setting up tests \(see page 39\)](#)

[Save a test setup \(see page 49\)](#)

[Recall a saved test setup \(see page 50\)](#)

## Save a test setup

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

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

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

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

1. Select **Options > Default Test Setup**.
2. Select **Setup** and set required options and parameters in the tabs (DUT, Test Selection, and so on).
3. Select **Reports** and set your [report options \(see page 33\)](#).
4. Optional: Click **Start** to run the test and verify that it runs correctly and captures the information you want. If it does not, edit the parameters and repeat this step until the test runs to your satisfaction.

Running the test helps verify that all parameters are set correctly, but it is not a necessary step.

5. Select **Options > Save Test Setup**. Enter the file name for the setup file. The application saves the file to X:\USB PWR\*<session\_name>*.

**See also**

- [About setting up tests \(see page 39\)](#)
- [Test setup overview \(see page 43\)](#)
- [View test-related files \(see page 32\)](#)
- [About configuring tests \(see page 25\)](#)

## Open (load) a saved test setup

These instructions are for recalling saved test setups.

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

**See also**

- [About test setups \(see page 49\)](#)
- [Create a new test setup based on an existing one \(see page 50\)](#)
- [Test setups overview \(see page 43\)](#)

## Create a new test setup based on an existing one

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

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

**See also**

- [About test setups \(see page 49\)](#)
- [Set DUT parameters \(see page 19\)](#)
- [Configuration parameters \(see page 27\)](#)
- [Select acquisitions \(see page 21\)](#)

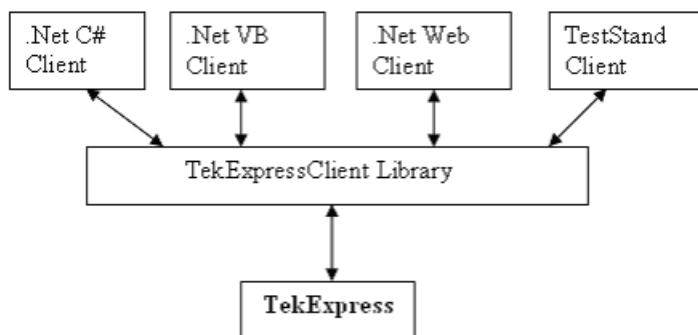
## About the programmatic interface

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 \(see page 51\)](#)

## Requirements for developing TekExpress client

While developing TekExpress Client, use the TekExpressClient.dll. The client can be a VB .Net, C# .Net, TestStand, 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.

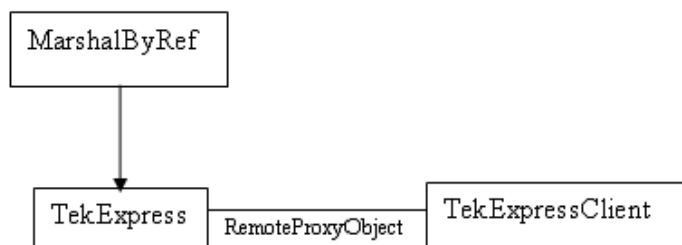
3. When the client has connected to and locked the server, the client can access any of the programmatic controls needed to run the remote automations.
4. After the client operations finish, the client unlocks the server.

## See also

[USB PWR application commands flow \(see page 60\)](#)

## Remote proxy object

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



The following is an example:

```
RemotingConfiguration.RegisterWellKnownServiceType (typeof (TekExpressRemoteInterface), "TekExpress Remote interface", WellKnownObjectMode.Singleton);
```

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

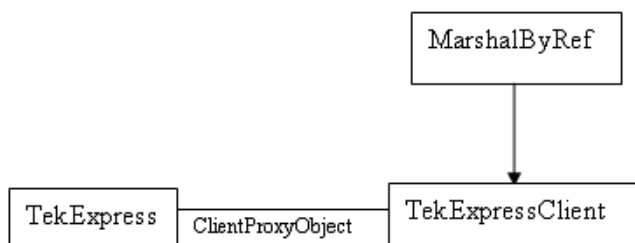
For example,

```
//Get a reference to the remote object
```

```
remoteObject = (IRemoteInterface)Activator.GetObject(typeof(IRemoteInterface), URL.ToString());
```

## Client proxy object

Client exposes a proxy object to receive certain information.



For example,

```
//Register the client proxy object
wellKnownServiceTypeEntry[] e = RemotingConfiguration.GetRegisteredWellKnownServiceTypes();

clientInterface = new ClientInterface();

RemotingConfiguration.RegisterWellKnownServiceType(typeof(ClientInterface),
"Remote Client Interface", wellKnownObjectMode.Singleton);

//Expose the client proxy object through marshalling
RemotingServices.Marshal(clientInterface, "Remote Client Inteface");
```

The client proxy object is used for the following:

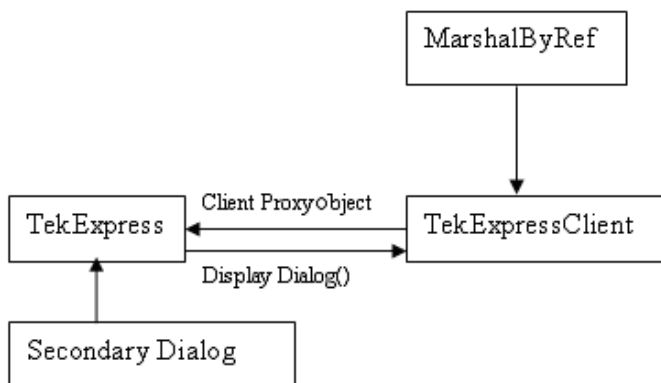
- To get the secondary dialog messages from the server.
- To get the file transfer commands from the server while transferring the report.

Examples

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

For more information, click the following links:

[Secondary Dialog Message Handling](#)



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

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

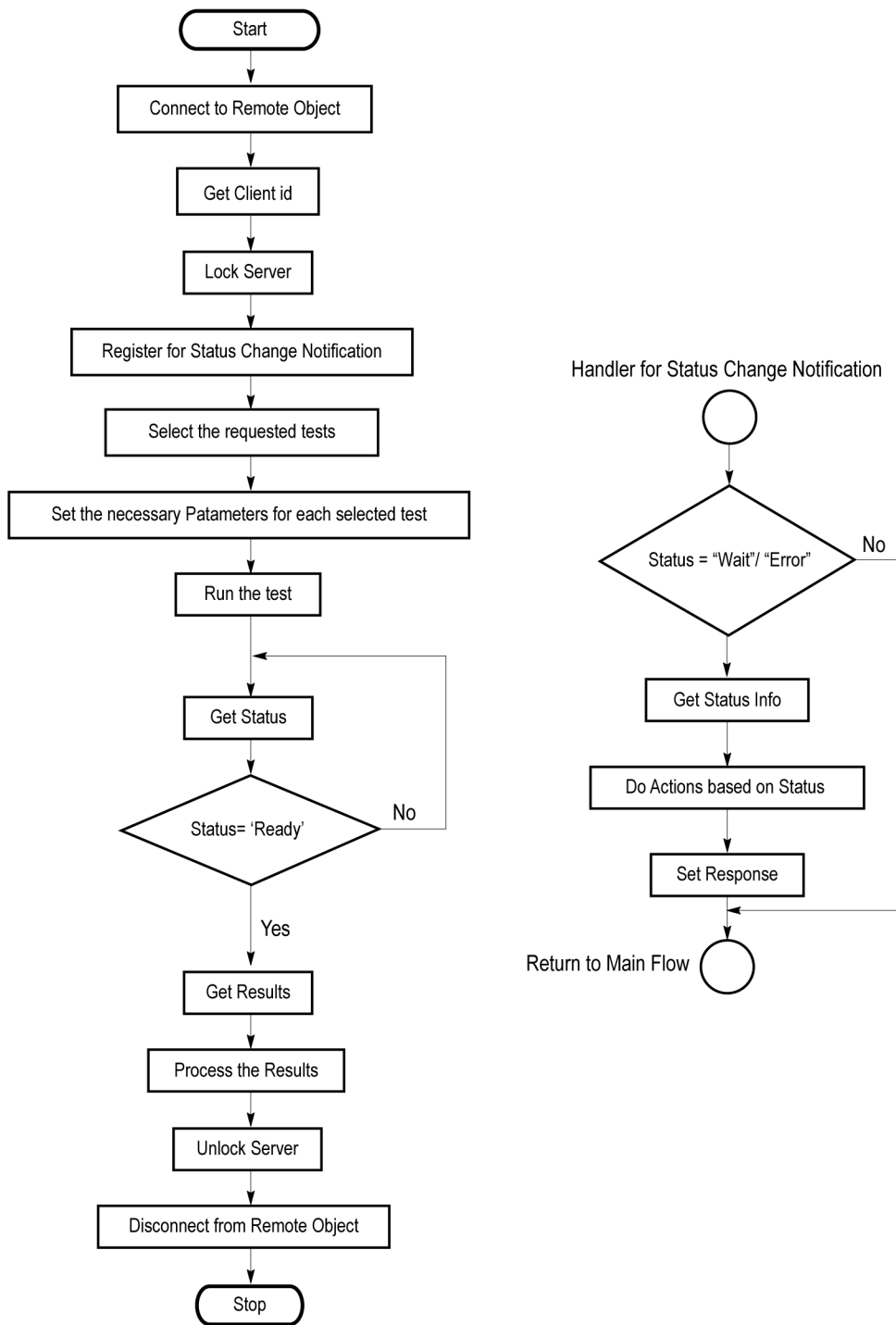
File Transfer Events

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

## Client programmatic interface example

The following is an overview of the client programmatic interface:

### **Process flowchart diagram**



**Process overview:**



1. Connect to a server or remote object using a programmatic interface.
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. For details, see [Handler of Status Change Notification \(see page 57\)](#).

---

**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**

[USB PWR application commands flow \(see page 60\)](#)

[Program remote access code example \(see page 59\)](#)

## Program remote access code example

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

**Table 9: Remote access code example**

| Task                              | Code   |
|-----------------------------------|--|
| Start the application             |  |
| Connect through an IP address.    | <code>m_Client.Connect("localhost")</code> True or False<br><code>clientID = m_Client.getClientID</code>   |
| Lock the server                   | <code>m_Client.LockServer(clientID)</code>   |
| Disable the Popups                | <code>m_Client.SetVerboseMode(clientID, false)</code>  |
| Set the DUT ID                    | <code>m_Client.SetDutId(clientID, "DUT_Name")</code>   |
| Select a test                     | <code>mClient.SelectsingtTest(clientID, "Transmitter", "Detachable", "Spec 1.0", "400KHz-1MHz", true)</code>   |
| Select a Sample Rate              | <code>mClient.SetAcquireParameter(clientID, "Transmitter", "Detachable", "400KHz-1MHz", "Sample Rate (Ms/sec)\$250")</code>                              |
| Select a Record Length            | <code>mClient.SetAcquireParameter(clientID, "Transmitter", "Detachable", "400KHz-1MHz", "Record Length\$10")</code>                                      |
| Run with set configurations       | <code>m_Client.Run(clientID)</code>  |
| Wait for the test to complete.    | Do<br><code>Thread.sleep(500)</code><br><code>m_Client.Application_Status(clientID)</code><br>Select case status<br>Case "wait"                          |
| Get the current state information | <code>mClient.GetCurrentStateInfo(clientID, waitingMsbBxCaption, waitingMsbBxMessage, waitingMsbBxButtonTexts)</code>                                    |
| Send the response                 | <code>mClient.SendResponse(clientID, waitingMsbBxCaption, waitingMsbBxMessage, waitingMsbBxResponse)</code><br>End Select<br>Loop Until status = "Ready" |
| Save results                      | Save all results values from folder for current run<br><code>m_Client.TransferResult(clientID, logDirname)</code>  |
| Unlock the server                 | <code>m_Client.UnlockServer(clientID)</code>   |
| Disconnect from server            | <code>m_Client.Disconnect()</code>   |
| Exit the application              |  |

## USBPWR application commands flow

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

[Connect through an IP address \(see page 65\)](#)

[Lock the server \(see page 66\)](#)

[Disable the popups \(see page 67\)](#)

[Set or get the DUT ID \(see page 68\)](#)

[Set the configuration parameters for a suite or measurement \(see page 69\)](#)

[Query the configuration parameters for a suite or measurement \(see page 70\)](#)

[Select a test \(see page 71\)](#)

[Select a suite \(see page 72\)](#)

[Select a channel \(see page 73\)](#)

[Configure the selected measurement \(see page 74\)](#)

[Run with set configurations or stop the run operation \(see page 75\)](#)

[Handle error codes \(see page 76\)](#)

[Get or set the timeout value \(see page 77\)](#)

[Wait for the test to complete \(see page 78\)](#)

[After the test is complete \(see page 80\)](#)

[Save, recall, or query a saved session \(see page 84\)](#)

[Unlock the server \(see page 85\)](#)

[Disconnect from the server \(see page 85\)](#)

| <b>string id</b>                          |             |                  |   |
|---|-------------|------------------|---|
| <b>Name</b>                               | <b>Type</b> | <b>Direction</b> | <b>Description</b>                                      |
| id  | string      | IN               | Identifier of the client performing the remote function |
| Ready: Test configured and ready to start |             |                  |   |
| Running: Test running                     |             |                  |   |
| Paused: Test paused                       |             |                  |   |
| Wait: A popup that needs your inputs      |             |                  |   |
| Error: An error is occurred               |             |                  |   |

| <b>string dutName</b> |             |                  |                             |
|-----------------------|-------------|------------------|-----------------------------|
| <b>Name</b>           | <b>Type</b> | <b>Direction</b> | <b>Description</b>          |
| dutName               | string      | IN               | The new DUT ID of the setup |

| <b>out bool saved</b> |             |                  |   |
|-----------------------|-------------|------------------|---|
| <b>Name</b>           | <b>Type</b> | <b>Direction</b> | <b>Description</b>  |
| saved                 | bool        | OUT              | Boolean representing whether the current session is saved |

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

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

| <b>out string clientID</b> |             |                  |   |
|----------------------------|-------------|------------------|---|
| <b>Name</b>                | <b>Type</b> | <b>Direction</b> | <b>Description</b>  |
| clientid                   | string      | OUT              | Identifier of the client that is connected to the server<br>clientid = unique number + ipaddress of the client. For example, 1065-192.157.98.70 |

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

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

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

**string dutId**

| Name  | Type   | Direction | Description              |
|-------|--------|-----------|--------------------------|
| dutId | string | OUT       | The DUT ID of the setup. |

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

**string device**

| Name   | Type   | Direction | Description                       |
|--------|--------|-----------|-----------------------------------|
| device | string | IN        | Specifies the name of the device. |

**string suite**

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

**string test**

| Name | Type   | Direction | Description  |
|------|--------|-----------|--|
| test | string | IN        | Specifies the name of the test to obtain the pass or fail status or a test result value. |

**string parameterString**

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

**int rowNr**

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

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

**out string[] status**

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

**string name**

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

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

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

**string name**

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

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

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

**string name**

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

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

**bool isSelected**

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

**string time**

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

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

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

**bool\_verbose**

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

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

**string filePath**

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

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

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

**out string WaitingMsbBxCaption**

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



| <b>out string WaitingMsbBxMessage</b> |             |                  |  |
|---------------------------------------|-------------|------------------|--|
| <b>Name</b>                           | <b>Type</b> | <b>Direction</b> | <b>Description</b>                             |
| message                               | string      | OUT              | The wait state/error state message sent to you |

| <b>out string[] WaitingMsbBxButtontexts</b> |              |                  |  |
|---|--------------|------------------|--|
| <b>Name</b>                                 | <b>Type</b>  | <b>Direction</b> | <b>Description</b>   |
| buttonTexts                                 | string array | OUT              | An array of strings containing the possible response types that you can send |

| <b>string WaitingMsbBxResponse</b> |             |                  |   |
|------------------------------------|-------------|------------------|---|
| <b>Name</b>                        | <b>Type</b> | <b>Direction</b> | <b>Description</b>  |
| response                           | string      | IN               | A string containing the response type that you can select (it must be one of the strings in the string array buttonTexts) |

| <b>out string clientID</b> |             |                  |  |
|----------------------------|-------------|------------------|--|
| <b>Name</b>                | <b>Type</b> | <b>Direction</b> | <b>Description</b>   |
| clientID                   | string      | OUT              | Identifier of the client that is connected to the server<br>clientID = unique number + IP address of the client. For example, 1065-192.157.98.70 |

## Connect through an IP address

| <b>Command name</b> | <b>Parameters</b>   | <b>Description</b>  | <b>Return value</b>                  | <b>Example</b>  |
|---------------------|---|---|--------------------------------------|---|
| Connect()           | <a href="#">string<br/>ipAddress (see<br/>page 61)</a><br><a href="#">out string<br/>clientID (see<br/>page 61)</a> | This method connects the client to the server.<br><a href="#">Note (see page 61)</a><br>The client provides the IP address to connect to the server.<br>The server provides a unique client identification number when connected to it. | Return value is either True or False | <pre>m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as boolean returnval = m_Client.Con- nect(ipaddress,m_clientID)</pre> |

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

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

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

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

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

## Lock the server

| Command name  | Parameters                                       | Description  | Return value  | Example   |
|---------------|--|--|---|---|
| LockSession() | <a href="#">string clientID</a><br>(see page 65) | This method locks the server.<br><a href="#">Note (see page 62)</a><br>The client must call this method before running any of the remote automations. The server can be locked by only one client. | String value that gives the status of the operation after it was performed<br>The return value is "Session Locked..." on success. | m_Client = new Client()<br>//m_Client is a reference to the Client class in the Client DLL.<br>returnval as string<br>returnval = m_Client.LockServer(clientID) |

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

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

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

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

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

## Disable the popups

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

| Command name     | Parameters   | Description  | Return value   | Example  |
|------------------|--|--|--|--|
| SetVerboseMode() | <a href="#">string clientID</a><br>(see page 65)<br><a href="#">bool _verbose</a><br>(see page 64) | This method sets the verbose mode to either true or false.<br><br>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.<br><br>When the value is set to false, all the message boxes are shown on the server machine. | String that gives the status of the operation after it was performed<br><br>When Verbose mode is set to true, the return value is "Verbose mode turned on. All dialog boxes will be shown to client".<br><br>When Verbose mode is set to false, the return value is "Verbose mode turned off. All dialog boxes will be shown to server". | <pre>m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string <b>Verbose mode is turned on</b> return=m_Client.SetVerbose- Mode(clientID, true) <b>Verbose mode is turned off</b> returnval=m_Client.SetVer- boseMode(clientID, false)</pre> |

---

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

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

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

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

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

---

## Set or get the DUT ID

| Command name | Parameters  | Description  | Return value  | Example   |
|--------------|---|--|---|---|
| SetDutId()   | <a href="#">string clientID (see page 65)</a><br><a href="#">string dutName (see page 61)</a> | This method changes the DUT ID of the setup. The client must provide a valid DUT ID. | String that gives the status of the operation after it was performed<br><br>Return value is "DUT Id Changed" on success | m_Client = new Client()<br>//m_Client is a reference to the Client class in the Client DLL.<br>returnval as string<br>return=m_Client.SetDutId(clientID,desiredDutId)<br><a href="#">Note (see page 61)</a> |
| GetDutId()   | <a href="#">string clientID (see page 65)</a><br><a href="#">string dutId (see page 62)</a>   | This method gets the DUT ID of the current setup.                                    | String that gives the status of the operation after it was performed  | m_Client = new Client()<br>//m_Client is a reference to the Client class in the Client DLL.<br>returnval as string<br>return=m_Client.GetDutid(clientID, out DutId)   |

---

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

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

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

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

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

---

## Set the configuration parameters for a suite or measurement

| Command name          | Parameters  | Description  | Return value  | Example   |
|-----------------------|---|--|---|---|
| SetGeneralParameter() | <a href="#">string clientID (see page 65)</a><br><a href="#">string device (see page 62)</a><br><a href="#">string suite (see page 62)</a><br><a href="#">string test (see page 62)</a><br><a href="#">string parameterString (see page 62)</a> | This method sets the general configuration parameters for a suite or measurement.  | String that gives the status of the operation after it has been performed<br><br>The return value is "" (an empty String) on success. | <pre>m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string <a href="#">Select email on failure preference example (see page 70)</a></pre>  |
| SetAnalyzeParameter() | <a href="#">string clientID (see page 65)</a><br><a href="#">string device (see page 62)</a><br><a href="#">string suite (see page 62)</a><br><a href="#">string test (see page 62)</a><br><a href="#">string parameterString (see page 62)</a> | This method sets the configuration parameters in the Analyze panel of the Configuration Panel dialog box for a suite or measurement. | String that gives the status of the operation after it has been performed The return value is "" (an empty String) on success.        | <pre>m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string</pre>   |
| SetAcquireParameter() | <a href="#">string clientID (see page 65)</a><br><a href="#">string device (see page 62)</a><br><a href="#">string suite (see page 62)</a><br><a href="#">string test (see page 62)</a><br><a href="#">string parameterString (see page 62)</a> | This method sets the configuration parameters in the Acquire panel of the Configuration Panel dialog box for a suite or measurement. | String that gives the status of the operation after it has been performed The return value is "" (an empty String) on success.        | <pre>//m_Client is a reference to the Client class in the Client DLL. returnval as string returnVal = remoteObject.SetAcquireParameter( id, device, suite, test, parameterString) if ((OP_STATUS) returnVal != OP_STATUS.SUCCESS) return CommandFailed(returnVal)</pre> |

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

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

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

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

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

**Select email on failure preference example**

```
returnval=mClient.SetGeneralParameter(clientID, "Transmitter", "Detachable", "400KHz-1MHz",
"On Failure Stop and Notify$False")
```

**Query the configuration parameters for a suite or measurement**

| Command name          | Parameters  | Description  | Return value   | Example   |
|-----------------------|---|--|--|---|
| GetGeneralParameter() | <a href="#">string clientID (see page 65)</a><br><a href="#">string device (see page 62)</a><br><a href="#">string suite (see page 62)</a><br><a href="#">string test (see page 62)</a><br><a href="#">string parameterString (see page 62)</a> | This method gets the general configuration parameters for a suite or measurement.  | The return value is the general configuration parameter for a specified suite or measurement that is set.  | m_Client = new Client()<br>//m_Client is a reference to the Client class in the Client DLL.<br>returnval as string<br><a href="#">Query email on failure preference example (see page 71)</a> |
| GetAnalyzeParameter() | <a href="#">string clientID (see page 65)</a><br><a href="#">string device (see page 62)</a><br><a href="#">string suite (see page 62)</a><br><a href="#">string test (see page 62)</a><br><a href="#">string parameterString (see page 62)</a> | This method gets the configuration parameters set in the Analyze panel of the Configuration Panel dialog box for a specified suite or measurement. | The return value is the configuration parameter set in the Analyze panel of the Configuration Panel dialog box for a specified suite or measurement. | m_Client = new Client()<br>//m_Client is a reference to the Client class in the Client DLL.<br>returnval as string  |
| GetAcquireParameter() | <a href="#">string clientID (see page 65)</a><br><a href="#">string device (see page 62)</a><br><a href="#">string suite (see page 62)</a><br><a href="#">string test (see page 62)</a><br><a href="#">string parameterString (see page 62)</a> | This method gets the configuration parameters set in the Acquire panel for a specified suite or measurement.                                       | The return value is the configuration parameter set in the Acquire panel for a specified suite or measurement.                                       | m_Client = new Client()<br>//m_Client is a reference to the Client class in the Client DLL.<br>returnval as string  |

---

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

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

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

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

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

---

#### Query email on failure preference example

```
returnval = mClient.GetGeneralParameter(clientID, "Transmitter", "Detachable", "400KHz-1MHz",
"On Failure Stop and Notify")
```

## Select a test

| Command name          | Parameters   | Description  | Return value   | Example  |
|-----------------------|--|--|--|--|
| GetGeneralParameter() | <a href="#">string clientID (see page 65)</a><br><a href="#">string device (see page 62)</a><br><a href="#">string suite (see page 62)</a><br><a href="#">string test (see page 62)</a><br><a href="#">bool isSelected (see page 63)</a> | <p>This method selects or deselects a specified test.</p> <p>If this Setting parameter is set to true, you can select a measurement.</p> <p>If this Setting parameter is set to false, you can deselect a measurement.</p> | <p>String that displays the status of the operation after it has been performed.</p> <p>The return value is "" (an empty String) on success.</p> | <p>//m_Client is a reference to the Client class in the Client DLL</p> <p>returnval as string</p> <p><a href="#">Select test example (see page 72)</a></p> |

---

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

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

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

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

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

---

**Select test example**

To select measurement test Common mode- 400KHz-1MHz:

```
returnval = mClient.SelectTest(clientID, "Transmitter", "Detachable", "400KHz-1MHz", true)
```

## Select a suite

Test suite must be set for Detachable or Permanent tests.

| Command name | Parameters   | Description   | Return value   | Example   |
|--------------|--|---|--|---|
| SelectSuite  | <a href="#">string clientID (see page 65)</a><br><a href="#">string device (see page 62)</a><br><a href="#">string suite (see page 62)</a><br><a href="#">string test (see page 62)</a><br><a href="#">bool isSelected (see page 63)</a> | This method selects or deselects a specified suite.<br><br>When this parameter is set to true, you can select a suite. When this parameter is set to false, you can deselect a suite. | String that gives the status of the operation after it has been performed.<br><br>The return value is "" (an empty String) on success. | <pre>m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string <b>Select Suite (Default):</b> returnval=mClient.SelectSuite(clientID, "Transmitter", "Detachable", true)</pre> |

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

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

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

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

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



## Select a channel

| Command name          | Parameters  | Description  | Return value   | Example   |
|-----------------------|---|--|--|---|
| SetGeneralParameter() | <a href="#">string clientID (see page 65)</a><br><a href="#">string device (see page 62)</a><br><a href="#">string suite (see page 62)</a><br><a href="#">string test (see page 62)</a><br><a href="#">string parameterString (see page 62)</a> | This method sets the parameters that are not specific to any one test.<br><br><b>NOTE.</b> Using this command we can select a lane, channel, or source type. | String that gives the status of the operation after it has been performed.<br><br>The return value is "" (an empty String) on success. | <pre>m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string</pre> <a href="#">Set channel source example (see page 74)</a>  |
| SetAnalyzeParameter() | <a href="#">string clientID (see page 65)</a><br><a href="#">string device (see page 62)</a><br><a href="#">string suite (see page 62)</a><br><a href="#">string test (see page 62)</a><br><a href="#">string parameterString (see page 62)</a> | This method sets the configuration parameters in the Analyze panel of the Configuration Panel dialog box for a specified suite or measurement.               | The return value is "" (an empty String) on success.   | <pre>m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string</pre>   |
| SetAcquireParameter() | <a href="#">string clientID (see page 65)</a><br><a href="#">string device (see page 62)</a><br><a href="#">string suite (see page 62)</a><br><a href="#">string test (see page 62)</a><br><a href="#">string parameterString (see page 62)</a> | This method sets the configuration parameters in the Acquire panel of the Configuration Panel dialog box for a specified suite or measurement.               | The return value is "" (an empty String) on success.   | <pre>m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string returnVal = remoteObject.SetAcquireParameter( id, device, suite, test, parameterString) if ((OP_STATUS) returnVal != OP_STATUS.SUCCESS) return CommandFailed(returnVal)</pre> |

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

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

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

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

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

**Set channel source example**

```
returnval = mClient.SetGeneralParameter(clientID, "Transmitter", "Detachable", "400KHz-1MHz", "Channel Selection$CH1")
```

## Configure the selected measurement

| Command name          | Parameters  | Description  | Return value   | Example   |
|-----------------------|---|--|--|---|
| SetAnalyzeParameter() | <a href="#">string clientID (see page 65)</a><br><a href="#">string device (see page 62)</a><br><a href="#">string suite (see page 62)</a><br><a href="#">string test (see page 62)</a><br><a href="#">string parameterString (see page 62)</a> | This method sets the Analyze parameters (Configuration parameters) for a specified test. | The return value is "" (an empty String) on success. | <pre>m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string</pre> |

---

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

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

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

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

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

---

## Run with set configurations or stop the run operation

| Command name | Parameters                                       | Description   | Return value  | Example   |
|--------------|--|---|---|---|
| Run()        | <a href="#">string clientID</a><br>(see page 65) | Runs the selected tests <a href="#">Note (see page 63)</a><br>After the server is set up and configured, run it remotely using this function. | String that gives the status of the operation after it was performed.<br>The return value is "Run started..." on success. | m_Client = new Client()<br>//m_Client is a reference to the Client class in the Client DLL.<br>returnval as string<br>returnval=m_Client.Run(clientID)  |
| Stop()       | <a href="#">string clientID</a><br>(see page 65) | Stops the running tests. <a href="#">Note (see page 64)</a>   | String that gives the status of the operation after it was performed<br>The return value is "Stopped..." on success.      | m_Client = new Client()<br>//m_Client is a reference to the Client class in the Client DLL.<br>returnval as string<br>returnval=m_Client.Stop(clientID) |

---

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

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

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

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

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

---

## Handle error codes

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

| Code | Value     | Description  |
|------|-----------|--|
| -1   | FAIL      | The operation failed   |
| 1    | SUCCESS   | The operation succeeded  |
| 2    | NOT FOUND | Server not found   |
| 3    | LOCKED    | The server is locked by another client, so the operation cannot be performed |
| 4    | UNLOCK    | The server is not locked; lock the server before performing the operation    |
| 0    | NULL      | Nothing  |

---

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

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

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

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

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

---

## Get or set the timeout value

| Command name | Parameters  | Description   | Return value   | Example   |
|--------------|---|---|--|---|
| GetTimeOut() | <a href="#">string clientID</a><br>(see page 65)  | Returns the current timeout period set by the client  | String that gives the status of the operation after it was performed<br><br>The default return value is 1800000.                     | m_Client = new Client()<br>//m_Client is a reference to the Client class in the Client DLL.<br>returnval as string<br>returnval=m_Client.GetTimeOut()                         |
| SetTimeOut() | <a href="#">string clientID</a><br>(see page 65)<br><a href="#">string time</a> (see page 64) | Sets a timeout period specified by the client. After this timeout period expires, the server is unlocked automatically. | String that gives the status of the operation after it was performed<br><br>On success the return value is "TimeOut Period Changed". | m_Client = new Client()<br>//m_Client is a reference to the Client class in the Client DLL.<br>returnval as string<br>returnval=m_Client.SetTimeOut(clientID, desiredTimeOut) |

---

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

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

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

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

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

---

## Wait for the test to complete

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

| Command name                     | Parameters  | Description   | Return value   | Example   |
|----------------------------------|---|---|--|---|
| <code>ApplicationStatus()</code> | <a href="#">string clientID</a><br>(see page 65)  | This method gets the status of the server application.<br>The states are <a href="#">Running</a> , <a href="#">Paused</a> , <a href="#">Wait</a> , or <a href="#">Error</a> (see page 60) | String value that gives the status of the server application   | <code>m_Client = new Client()</code><br><code>//m_Client is a reference to the Client class in the Client DLL.</code><br>returnval as string<br><code>returnval=m_Client.ApplicationStatus(clientID)</code> |
| <code>QueryStatus()</code>       | <a href="#">string clientID</a><br>(see page 65)<br><a href="#">out string[] status</a> (see page 63) | An interface for the user to transfer Analyze panel status messages from the server to the client   | String that gives the status of the operation after it was performed<br><br>On success the return value is "Transferred...". | <code>m_Client = new Client()</code><br><code>//m_Client is a reference to the Client class in the Client DLL.</code><br>returnval as string<br><a href="#">Query status example (see page 80)</a>          |

| Command name   | Parameters  | Description  | Return value   | Example  |
|--|---|--|--|--|
| GetCurrentState-Info()<br><b>NOTE.</b> This command is used when the application is running and is in the wait or error state. | <a href="#">string clientID (see page 65)</a><br><a href="#">out string WaitingMsbBxCaption (see page 64)</a><br><a href="#">out string WaitingMsbBxMessage (see page 65)</a><br><a href="#">out string[] WaitingMsbBxButtontexts (see page 65)</a> | This method gets the additional information of the states when the application is in Wait or Error state.<br><br>Except client ID, all the others are Out parameters.  | This command does not return any value.<br><br>This function populates the Out parameters that are passed when invoking this function. | <pre>m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL mClient.GetCurrentState-Info(clientID, WaitingMsbBxCaption, WaitingMsbBxMessage, WaitingMsbBxButtontexts)</pre> |
| SendResponse()<br><b>NOTE.</b> This command is used when the application is running and is in the wait or error state.         | <a href="#">string clientID (see page 65)</a><br><a href="#">out string WaitingMsbBxCaption (see page 64)</a><br><a href="#">out string WaitingMsbBxMessage (see page 65)</a><br><a href="#">string WaitingMsbBxResponse (see page 65)</a>          | After receiving the additional information using the method GetCurrentStateInfo(), the client can decide which response to send and then send the response to the application using this function. The response should be one of the strings that was received earlier as a string array in the GetCurrentStateInfo function. The _caption and _message should match the information received earlier in the GetCurrentStateInfo function. | This command does not return any value.  | <pre>m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL mClient.SendResponse(clientID, WaitingMsbBxCaption, WaitingMsbBxMessage, WaitingMsbBxResponse)</pre>            |

---

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

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

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

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

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

---

**Query status example**

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

**After the test is complete**

| Command name        | Parameters  | Description  | Return value  | Example  |
|---------------------|---|--|---|--|
| GetPassFailStatus() | <a href="#">string clientID (see page 65)</a><br><a href="#">string device (see page 62)</a><br><a href="#">string suite (see page 62)</a><br><a href="#">string test (see page 62)</a>   | This method gets the pass or fail status of the measurement after test completion.<br><br><b>NOTE.</b> <i>Execute this command after completing the measurement.</i> | String that gives the status of the operation after it was performed.<br><br>Returns the pass or fail status in the form of a string. | m_Client = new Client()<br>//m_Client is a reference to the Client class in the Client DLL.<br>returnval as string   |
| GetResultsValue()   | <a href="#">string clientID (see page 65)</a><br><a href="#">string device (see page 62)</a><br><a href="#">string suite (see page 62)</a><br><a href="#">string test (see page 62)</a><br><a href="#">string parameterString (see page 62)</a> | This method gets the result values of the measurement after the run.   | String that gives the status of the operation after it was performed.<br><br>Returns the result value in the form of a string.        | m_Client = new Client()<br>//m_Client is a reference to the Client class in the Client DLL.<br>returnval as string<br>returnval=m_Client.GetResultsValue(clientID, device, devicesuite, test, parameterString) |



| Command name                        | Parameters   | Description  | Return value  | Example  |
|-------------------------------------|--|--|---|--|
| GetReportParameter()                | <a href="#">string clientID (see page 65)</a><br><a href="#">string device (see page 62)</a><br><a href="#">string suite (see page 62)</a><br><a href="#">string test (see page 62)</a><br><a href="#">string parameterString (see page 82)</a>  | This method gets the general report details such as oscilloscope model and TekExpress version.                   | The return value is the oscilloscope model, TekExpress application version, or USBPWR application version.                          | <pre>m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string</pre> <p><b>Oscilloscope Model</b></p> <pre>returnval=m_Client.GetReportParameter(clientID,"Scope Model")</pre> <p><b>TekExpress Version</b></p> <pre>returnval=m_Client.GetReportParameter(clientID,"TekExpress Version")</pre> <p><b>USB PWR Version</b></p> <pre>returnval=m_Client.GetReportParameter(clientID,"Application Version")</pre>  |
| GetResultsValueForSubMeasurements() | <a href="#">string clientID (see page 65)</a><br><a href="#">string device (see page 62)</a><br><a href="#">string suite (see page 62)</a><br><a href="#">string test (see page 62)</a><br><a href="#">string parameterString (see page 82)</a><br><a href="#">int rowNr (see page 62)</a> | This method gets the result values for individual submeasurements after the run.                                 | String that gives the status of the operation after it has been performed.<br><br>Returns the result value in the form of a string. | <pre>m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string</pre> <p><a href="#">Get results for a submeasurement example (see page 83)</a></p>  |
| GetReportParameter()                | <a href="#">string clientID (see page 65)</a><br><a href="#">string device (see page 62)</a><br><a href="#">string suite (see page 62)</a><br><a href="#">string test (see page 62)</a><br><a href="#">string parameterString (see page 82)</a>  | This method gets the general report details such as oscilloscope model, TekExpress version, and USB-PWR version. | The return value is the oscilloscope model, TekExpress version, and USB-PWR version.  | <pre>m_Client = new Client() //m_Client is a reference to the Client class in the Client DLL. returnval as string</pre> <p><b>Oscilloscope Model</b></p> <pre>returnval=m_Client.GetReportParameter( clientID,"Scope Model")</pre> <p><b>TekExpress Version</b></p> <pre>returnval=m_Client.GetReportParameter(clientID,"TekExpress Version")</pre> <p><b>USB-PWR Version</b></p> <pre>returnval=m_Client.GetReportParameter(clientID,"Application Version")</pre> |

| Command name     | Parameters   | Description   | Return value   | Example   |
|------------------|--|---|--|---|
| TransferReport() | <a href="#">string clientID</a><br>(see page 65)<br><a href="#">string filePath</a><br>(see page 64) | This method transfers the report generated after the run.<br><br>The report contains the summary of the run.<br><br>The client must provide the location where the report is to be saved at the client-end.   | String that gives the status of the operation after it has been performed.<br><br>Transfers all the result values in the form of a string. | m_Client = new Client()<br>//m_Client is a reference to the Client class in the Client DLL.<br>returnval as string<br>returnval=m_Client.TransferReport(clientID,"C:\Report")     |
| TransferImages() | <a href="#">string clientID</a><br>(see page 65)<br><a href="#">string filePath</a><br>(see page 64) | This method transfers all the images (screen shots) from the specified client and folder for the current run (for a suite or measurement).<br><br><b>NOTE.</b> Every time you click Start, a folder is created in the X: drive. Transfer the waveforms before clicking Start. | String that gives the status of the operation after it was performed.<br><br>Transfers all the images in the form of a string.             | m_Client = new Client()<br>//m_Client is a reference to the Client class in the Client DLL.<br>returnval as string<br>returnval=m_Client.TransferImages(clientID, "C:\Waveforms") |

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

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

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

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

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

| string parameterString |        |           |   |
|------------------------|--------|-----------|---|
| Name                   | Type   | Direction | Description   |
| parameterString        | string | IN        | Specifies the oscilloscope model, TekExpress version, and USB-PWR version |

**Get results for a submeasurement example**

This example returns the specified submeasurement results for test Common mode test 400KHz-1MHz.

```
returnval=m_Client.GetResultsValue( clientID,"Transmitter", "Detachable", "400KHz-1MHz",  
"Measured Value",0)
```

```
returnval=m_Client.GetResultsValue( clientID,"Transmitter", "Detachable", "1MHz-100MHz",  
"Measured Value",1)
```

## Save, recall, or query a saved session

| Command name         | Parameters  | Description  | Return value   | Example   |
|----------------------|---|--|--|---|
| CheckSession-Saved() | <a href="#">string clientID (see page 65)</a><br><a href="#">out bool saved (see page 61)</a> | This method checks whether the current session is saved.   | Return value is either True or False.  | <code>m_Client = new Client()<br/>//m_Client is a reference to the Client class in the Client DLL.<br/>returnval as string<br/>returnval=m_Client.CheckSessionSaved(m_clientID, out savedStatus)</code> |
| RecallSession()      | <a href="#">string clientID (see page 65)</a><br><a href="#">string name (see page 63)</a>    | Recalls a saved session. The client provides the session name.   | String that gives the status of the operation after it has been performed.<br>The return value is "Session Recalled..."          | <code>m_Client = new Client()<br/>//m_Client is a reference to the Client class in the Client DLL.<br/>returnval as string<br/>returnval=m_Client.RecallSession(clientID, savedSessionName)</code>      |
| SaveSession()        | <a href="#">string clientID (see page 65)</a><br><a href="#">string name (see page 63)</a>    | Saves the current session. The client provides the session name.   | String that gives the status of the operation after it has been performed.<br>The return value is "Session Saved..."/"Failed..." | <code>m_Client = new Client()<br/>//m_Client is a reference to the Client class in the Client DLL.<br/>returnval as string<br/>returnval=m_Client.SaveSession(clientID, desiredSessionName)</code>      |
| SaveSessionAs()      | <a href="#">string clientID (see page 65)</a><br><a href="#">string name (see page 63)</a>    | Saves the current session under a different name every time this method is called. The client provides the session name. | String that gives the status of the operation after it has been performed.<br>The return value is "Session Saved..."             | <code>m_Client = new Client()<br/>//m_Client is a reference to the Client class in the Client DLL.<br/>returnval as string<br/>returnval=m_Client.SaveSessionAs(clientID, desiredSessionName)</code>    |

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

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

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

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

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

## Unlock the server

| Command name    | Parameters                                       | Description  | Return value   | Example   |
|-----------------|--|--|--|---|
| UnlockSession() | <a href="#">string clientID</a><br>(see page 65) | This method unlocks the server from the client. The ID of the client to be unlocked must be provided. <a href="#">Note (see page 64)</a> | String that gives the status of the operation after it has been performed. The return value is "Session UnLocked..." | m_Client = new Client()<br>//m_Client is a reference to the Client class in the Client DLL.<br>returnval as string<br>returnval=m_Client.UnlockServer(clientID) |

---

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

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

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

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

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

---

## Disconnect from the server

| Command name | Parameters                                       | Description  | Return value  | Example   |
|--------------|--|--|---|---|
| Disconnect() | <a href="#">string clientID</a><br>(see page 65) | This method disconnects the client from the server. <a href="#">Note (see page 61)</a> | Integer value that gives the status of the operation after it has been performed.<br>1 for Success<br>-1 for Failure. | m_Client = new Client()<br>//m_Client is a reference to the Client class in the Client DLL.<br>returnval as string<br>returnval=m_Client.Disconnect(m_clientID) |

---

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

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

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

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

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

---



# Index

## A

- About TekExpress, 8
- Acquire live waveforms, 19
- Acquire parameters
  - including in test reports, 33
  - viewing in reports, 35
- Acquisition and Save Options, 23
- Acquisition tab, 21
- Acquisitions tab
  - channel sources, 21
  - set channel sources, 21
  - set waveform source (prerecorded files), 24
- Activate the license, 6
- Advanced View, 39
- Application commands, 60
- Application controls, 12
- Application version, 7
- Application window
  - moving, 12
- ApplicationStatus(), 78

## B

- Bandwidth, 27

## C

- Channel selection, 21
- CheckSessionSaved(), 84
- Client proxy object, 53
- Code example, remote access, 59
- Command buttons, 12
- Commands
  - Connect(), 65
  - GetDutId(), 68
  - LockSession(), 66
  - SetDutId(), 68
- Commands list, 60
- Comments, 19
- Compliance Mode, 27
- Compliance View, 39
- Configuration parameters, 27
- Configuring email notifications, 16

- Connect(), 65
- Connected instruments
  - searching for, 41
- Connection requirements, 40

## D

- Default directory, 8
- Detailed log view, 29
- Device parameters, 19
- Device profile connections, 40
- Device profiles, 19
- Directories, 8
- Disable Popups command, 67
- Disconnect from Server
  - command, 85
- Disconnect(), 85
- DPOJET Plug-In, 5
- DUT ID, 19
- DUT ID commands, 68
- DUT parameters, 19
- DUT type
  - 10GBase-KR, 19
  - 40GBase-KR4, 19

## E

- Email notifications, 16
- Email settings, 15
- Equipment setup, 40
- Evaluation mode, 11
- Exiting the application, 11

## F

- Features, 8
- File name extensions, 10
- Flowchart for client programmatic interface, 55
- Free trials, 39

## G

- GetCurrentStateInfo(), 79
- GetDutId(), 68
- GetPassFailStatus(), 80

- GetReportParameter(), 81
- GetResultsValue(), 80
- GetTimeOut(), 77
- Global settings, 27

## H

- Help conventions, 1
- Host tests, 39

## I

- Installing the software
  - DPOJET Plug-In, 5
  - TekExpress application for USBPWR, 5
- Instruments
  - discovering connected, 14
  - viewing connected, 41
- Instruments detected, 27
- Interface, 51
- Interface error codes, 76
- IP address connection
  - commands, 65

## K

- Keep On Top, 11
- Key, 11

## L

- Lanes
  - number of, 19
- License, 11
- License activation, 6
- License agreement, 7
- Limits Editor, 27
- Loading a test setup, 50
- LockSession(), 66
- Log view
  - save file, 29
- Log view tab, 29

**M**

- Measurement limits, 27
- Menus, 12
- Mode
  - Compliance, 27
  - User Defined, 27
- Module tests, 39
- Moving the application window, 12
- My TekExpress folder
  - files stored in, 32
  - location of, 45

**N**

- Notifications, 25

**O**

- Opening a saved test setup, 50
- Options menu, 13
  - Instrument control settings, 14
  - Keep On Top, 11
- Oscilloscopes supported, 4
- Overall test result, 30
- Overview of USBPWR, 8

**P**

- Panels, 18
- Pass/Fail summary
  - viewing, 35
- Pass/Fail Summary
  - including in reports, 34
- Plot images
  - including in reports, 35
  - viewing, 35
- Preferences menu, 30
- Preferences tab, 18
- Prerecorded waveform files, 21
  - selecting run sessions for, 19
- Prerecorded waveform files (acquisitions), 24
- Prerun checklist, 48
- Program example, 59
- Programmatic interface, 51

**Q**

- QueryStatus(), 78

**R**

- Reactivate the license, 6
- Real time oscilloscope, 27
- Recalling a test setup, 50
- RecallSession(), 84
- Record length, 27
- Refresh sources (acquisitions), 21
- Related documentation, 1
- Remote proxy object, 53
- Report name, 34
- Report options, 33
- Report sections, 35
- Reports, 35
  - adding user comments to, 19
  - receiving in email notifications, 16
- Reports panel, 33
- Resource file, 11
- Results panel, 30
- Run commands, 75
- Run the application, 11
- Run(), 75

**S**

- Sample Rate, 27
- SaveSession(), 84
- SaveSessionAs(), 84
- Saving test setups, 49
- Saving tests, 32
- Selecting test report contents, 33
- Selecting tests, 20
- SendResponse(), 79
- Server, 51
- Server locking commands, 66
- Session files, 32
- Session folders, 32
- Session Status commands, 84
- Set Acquisition and Save Options, 23
- SetDutId(), 68
- SetGeneralParameter(), 69
- SetTimeout(), 77
- Setting up equipment, 40
- Setting up tests, 39

- Setup panel, 18
- Setup panel views, 19
- SetVerboseMode(), 67
- Software installation, 5
- Software version, 7
- Sources (acquisitions), 21
- Start the application, 11
- Status panel, 29
- Stop commands, 75
- Stop(), 75
- Support, 2
- System requirements, 3

**T**

- Technical support, 2
- TekExpress application for USBPWR, 5
- TekExpress client, 51
- TekExpress client requirements, 51
- TekExpress server, 51
- Test groups, 20
- Test notification preferences, 25
- Test parameters (Configuration tab), 27
- Test reports, 35
- Test results
  - emailing, 16
- Test Results commands, 80
- Test selection controls, 20
- Test setup files, 32
- Test setup steps, 43
- Test setups, 49
  - creating, 50
  - load, 50
  - open, 50
  - recalling, 50
  - saving, 49
- Test Status commands, 78
- Test status tab, 29
- Test-related files, 32
- Tests
  - running, 45
  - selecting, 20
  - setting up, 39
- Timeout Value commands, 77
- TransferImages(), 82
- TransferReport(), 82



**U**

Unlock Server command, 85  
UnlockSession(), 85  
Untitled session folder, 45  
USBPWR features, 8  
User account setting (Windows 7), 3

User comments  
    location in reports, 35  
User Comments  
    including in reports, 35  
User Defined Mode, 27

**W**

Waveform files  
    locating and storing, 32  
Windows 7 user account setting, 3