
Keysight Infiniium Oscilloscopes User Defined Application Generator

Notices

© Keysight Technologies, Inc. 2009-2023

No part of this manual may be reproduced in any form or by any means (including electronic storage and retrieval or translation into a foreign language) without prior agreement and written consent from Keysight Technologies, Inc. as governed by United States and international copyright laws.

Revision

Version 04.20.0000

Edition

May 1, 2023

Available in electronic format only

Published by:

Keysight Technologies, Inc.
1900 Garden of the Gods Road
Colorado Springs, CO 80907 USA

Warranty

The material contained in this document is provided "as is," and is subject to being changed, without notice, in future editions. Further, to the maximum extent permitted by applicable law, Keysight disclaims all warranties, either express or implied, with regard to this manual and any information contained herein, including but not limited to the implied warranties of merchantability and fitness for a particular purpose. Keysight shall not be liable for errors or for incidental or consequential damages in connection with the furnishing, use, or performance of this document or of any information contained herein. Should Keysight and the user have a separate written agreement with warranty terms covering the material in this document that conflict with these terms, the warranty terms in the separate agreement shall control.

Technology License

The hardware and/or software described in this document are furnished under a license and may be used or copied only in accordance with the terms of such license.

U.S. Government Rights

The Software is "commercial computer software," as defined by Federal Acquisition Regulation ("FAR") 2.101. Pursuant to FAR 12.212 and 27.405-3 and Department of Defense FAR Supplement ("DFARS") 227.7202, the U.S. government acquires commercial computer software under the same terms by which the software is customarily provided to the public. Accordingly, Keysight provides the Software to U.S. government customers under its standard commercial license, which is embodied in its End User License Agreement (EULA), a copy of which can be found at <http://www.keysight.com/find/sweula>. The license set forth in the EULA represents the exclusive authority by which the U.S. government may use, modify, distribute, or disclose the Software. The EULA and the license set forth therein, does not require or permit, among other things, that Keysight: (1) Furnish technical information related to commercial computer software or commercial computer software documentation that is not customarily provided to the public; or (2) Relinquish to, or otherwise provide, the government rights in excess of these rights customarily provided to the public to use, modify, reproduce, release, perform, display, or disclose commercial computer software or commercial computer software documentation. No additional government requirements beyond those set forth in the EULA shall apply, except to the extent that those terms, rights, or licenses are explicitly required from all providers of commercial computer software pursuant to the FAR and the DFARS and are set forth specifically in writing elsewhere in the EULA. Keysight shall be under no obligation to update, revise or otherwise modify the Software. With respect to any technical data as defined by FAR 2.101, pursuant to FAR 12.211 and 27.404.2 and DFARS 227.7102, the U.S. government acquires no greater than Limited Rights as defined in FAR 27.401 or DFAR 227.7103-5 (c), as applicable in any technical data.

Safety Notices

CAUTION

A **CAUTION** notice denotes a hazard. It calls attention to an operating procedure, practice, or the like that, if not correctly performed or adhered to, could result in damage to the product or loss of important data. Do not proceed beyond a **CAUTION** notice until the indicated conditions are fully understood and met.

WARNING

A **WARNING** notice denotes a hazard. It calls attention to an operating procedure, practice, or the like that, if not correctly performed or adhered to, could result in personal injury or death. Do not proceed beyond a **WARNING** notice until the indicated conditions are fully understood and met.

User Defined Application Generator—At a Glance

The User Defined Application Generator is an easy-to-use tool that:

- Lets you rapidly generate custom, automated test GUIs (graphical user interfaces) with minimal programming, or "Add-ins" that can be installed into existing compliance applications.
- Lets you develop applications that use the proprietary automated test application framework. This gives your applications features similar to compliance applications developed by Keysight.
- Lets you generate applications that use any Infiniium or InfiniiVision oscilloscope or FlexDCA feature that can be accessed via SCPI commands, including:
 - Serial data equalization.
 - Serial data analysis.
 - Noise reduction.
 - InfiniiScan.

You can combine features to generate powerful test applications.

- Is fully compatible with MATLAB and Infiniium's user-defined function. Your generated applications can import MATLAB graphics into test results and the HTML test report.
- Lets you generate applications that control external instruments (like pulse pattern generators) as well the Infiniium or InfiniiVision oscilloscope or FlexDCA.
- Lets you control a switch matrix, enabling you to use RF switches to automatically route your test points to the oscilloscope and avoid having to change physical connections during a run.
- Lets you generate test applications that interact with console applications (like DUT controller programs) or external applications (like MATLAB or other data post-processing tools).
- Lets you test your applications as they are being developed via:
 - A "Debug Run" feature that lets you step through test commands one at a time.
 - A "Run" button that lets you run your application before installing it on an Infiniium oscilloscope.
- Generates an install package that is easily distributed to your test application users and installed on their Infiniium real-time or DCA-X oscilloscopes, or PCs.
- Comes with a Keysight web site where you can download example applications or share your own: www.keysight.com/find/share_uda

Automated test applications created by the application generator:

- Guide users through the process of selecting and configuring tests, making oscilloscope connections, running tests, and evaluating the test results.

- Let users select individual or multiple tests to run.
- Show users how to make oscilloscope connections to the device under test.
- Automatically check for proper oscilloscope configuration.
- Automatically set up the oscilloscope for each test.
- Provide detailed information for each test that has been run.
- Let users specify the thresholds at which marginal or critical warnings appear.
- Create a printable HTML report of the tests that have been run.
- Can be controlled from a remote PC using the N5452A Remote Interface for Automated DigitalTest Applications.

See

- **"In This Guide"** on page 5

In This Guide

For an overview and list of features, see: "[User Defined Application Generator—At a Glance](#)" on page 3

This guide shows you how to use the User Defined Application generator. It contains these chapters:

- **Chapter 1**, "Installation and Set Up," starting on page 15
- **Chapter 2**, "Getting Started Tutorial," starting on page 21
- **Chapter 3**, "How To ...," starting on page 69
- **Chapter 4**, "Determining Add-In Compatibility," starting on page 81
- **Chapter 5**, "Setting Up the Application," starting on page 85
- **Chapter 6**, "Adding/Exporting/Importing Tests," starting on page 95
- **Chapter 7**, "Adding Config Variables," starting on page 147
- **Chapter 8**, "Adding Connection Instructions," starting on page 151
- **Chapter 9**, "Adding Switch Matrix Control," starting on page 153
- **Chapter 10**, "Adding Run Actions on Events," starting on page 171
- **Chapter 11**, "Setting Miscellaneous Options," starting on page 173
- **Chapter 12**, "Setting Debug Run Options," starting on page 177
- **Chapter 13**, "Generating the Application Software Installer," starting on page 183
- **Chapter 14**, "Automating the Generated Application," starting on page 185
- **Chapter 15**, "Working with Variables and Values," starting on page 189
- **Chapter 16**, "Managing Other Resources," starting on page 211
- **Chapter 17**, "Saving / Converting / Opening Projects," starting on page 239
- **Chapter 18**, "Using Generated Applications," starting on page 249
- **Chapter 19**, "Solving Problems," starting on page 399
- **Chapter 20**, "Menu/Toolbar/Options Reference," starting on page 417

For a printable version of this guide, see: [🔗 User Defined Application Generator User's Guide](#).

See Also

- *Oscilloscope Online Help* – for information on using your Infiniium oscilloscope.
- *Oscilloscope Programmer's Reference* – describes the SCPI commands for programming the oscilloscope. You can find a link to the *Programmer's Reference* in the oscilloscope online help "Manuals" topic.

Contents

User Defined Application Generator—At a Glance / 3

In This Guide / 5

1 Installation and Set Up

Requirements / 16

Installing the User Defined Application Generator / 18

Licensing / 19

2 Getting Started Tutorial

Running Tests Manually and Preparing Setup Files / 22

Organizing Your Working Files / 23

Creating an Application or Add-In / 24

Step 1: Give your application a name and version number / 27

Step 2: Add/Export/Import tests / 28

Step 3: Test the application / 35

Step 4: Add Config variables (optional) / 38

Step 5: Add connection instructions (optional) / 43

Step 6: Add/Export/Import subroutines (optional) / 49

Step 7: Add run actions (optional) / 51

Step 8: Add logos, images, help files (optional) / 53

Step 9: Specify Sample Projects (optional) / 56

Step 10: Specify Default User Preferences (optional) / 57

Step 11: Configure result detail limits (optional) / 57

Step 12: (FlexDCA and Infiniium Only) Enable InfiniiSim / 58

Step 13: Generate the installer / 59

Installing and Running the Application / 61

Step 14a: Installing an application / 61

Step 14b: Installing/Uninstalling and Running an Add-In / 61

Step 15: Restart the Infiniium Oscilloscope / 65

Step 16: Launch and use your application / 65

For More Information / 68

3 How To ...

- Find/Replace Text Strings in a Project / 70
- Change Oscilloscope Settings Based on Waveform Data / 72
- Make an External Measurement and Report the Result / 73
- Run Tests Until a Certain Result Occurs / 74
- Ensure Test 1 Executes Before Test 2 / 75
- Create Custom-Formatted Test Results / 76
- Execute MATLAB Scripts / 78

4 Determining Add-In Compatibility

5 Setting Up the Application

- Enter Application Name and Version / 86
- Specify Required Oscilloscope Software / 87
- (Application-Mode Only) Enable/Disable Test Group Filters / 88
- Add External Instruments / 91

6 Adding/Exporting/Importing Tests

- Enter Test Name, Description, References / 96
- Remote Interface ID / 97

Adding Steps	/ 98
Defaulting the Oscilloscope	/ 100
Loading an Oscilloscope Setup File	/ 100
Executing a Single SCPI Command	/ 100
Executing a SCPI Command File	/ 102
Loading a Scope Mask File	/ 103
(Infiniium Only) Applying a Transfer Function	/ 104
Using a Console Application	/ 105
Using an External Application	/ 108
Setting a Variable	/ 111
Calling a Subroutine	/ 112
Aborting	/ 112
Writing to a File	/ 113
Transferring Files To/From Instruments	/ 115
Displaying Messages	/ 116
Displaying List Selection	/ 118
Pausing	/ 119
Sleeping	/ 121
Verifying a Condition	/ 121
Manually Managing SCPI Errors	/ 123
Reporting an Intermediate Value	/ 124
Reporting Tabular Data	/ 125
Adding a Comment	/ 126
Return Steps	/ 126
If, Else, and EndIf Steps	/ 127
While, Break, Continue, and EndWhile Steps	/ 129
Refactoring: Extracting Steps to a Subroutine	/ 133
Defining Final Result Display Behavior	/ 135
Exporting/Importing Tests or Subroutines	/ 137
Grouping Tests	/ 140
Debug Run Button	/ 142

7 Adding Config Variables

Using Variable Substitutions	/ 148
Grouping Config Variables	/ 149

8 Adding Connection Instructions

9 Adding Switch Matrix Control

- Step 1: Identify Test Points / 154
- Step 2: Identify Compatible Switch Instruments / 156
- Step 3: Configure Options / 157
- Step 4: Assign Test Points to Connections / 159
- Step 5: (Optional) Include Sample Switch Settings / 160
- Assigning Test Points to Connections / 161
 - Defining a Signal Path / 161
- Simulators / 168
 - Creating a Simulator / 169

10 Adding Run Actions on Events

11 Setting Miscellaneous Options

- Add Online Help / 174
- Add Logos/Images / 175
- Specify Default User Preferences / 176

12 Setting Debug Run Options

- Oscilloscope SICL Address and Setup/Mask/Transfer Function File Paths / 178
- External Instruments SICL Address Setting/Testing / 181

13 Generating the Application Software Installer

14 Automating the Generated Application

15 Working with Variables and Values

- Managing Internal Variables / 190
- Managing Math Libraries / 192
- Managing Text Libraries / 194
- Managing Reserved Variables / 196

Getting Values /	198
To get a value from a Scope source /	198
To get a value from a Config Variable source /	200
To get a value from a Math Expression source /	201
To get a value from a Text source /	203
To get a value from a File source /	206
To get a value from an External Instrument source /	207
Variable Substitution /	209

16 Managing Other Resources

Managing Command Files /	212
Managing Connection Diagrams (HTML) /	214
Managing Connection Diagrams (Images) /	216
Managing Console Applications /	218
Managing External Application Scripts /	221
Managing Other Files /	223
Managing Repository /	226
Managing Scope Setup Files /	228
Managing Scope Mask Files /	230
Managing Transfer Function Files /	231
Managing Generated App Compatibility /	232
Scope Option Requirements /	234
IP Protection /	237

17 Saving / Converting / Opening Projects

Saving Application Generator Projects /	240
Opening Previously Saved Application Generator Projects /	242
Converting an Add-in Project to an Application Project /	244
Converting an Application Project to an Add-In Project /	245
Application Generator Project Sharing/Collaboration /	247

18 Using Generated Applications

Starting the Automated Test Application /	250
Creating or Opening a Test Project /	252
To set load preferences /	252

Setting Up the Test Environment /	253
Selecting Tests /	254
Configuring Tests /	256
To activate/refresh limit sets /	257
To create/edit limit sets /	258
To access InfiniiSim and PrecisionProbe /	263
(Infiniium Only) To access Acquisition Setup in Debug Mode /	265
Connecting the Oscilloscope to the DUT /	267
Running Tests /	268
To select the "store mode" /	270
To run multiple times /	271
To send email on pauses or stops /	273
To configure result tags /	273
To pause or stop on events /	288
To specify the event /	289
To set the display preferences /	290
Config Variable Auto-Iteration /	292
Automating the Application /	295
To enter commands in Script mode /	295
To enter commands in Files mode /	298
To begin Script or Files execution /	298
To display automation settings status /	299
To try a command line /	300
Executing Miscellaneous Scripts /	301
Viewing Results /	303
To change margin thresholds /	304
To set test display preferences /	305
To set trial display preferences /	305
To select jump actions /	306
Viewing/Exporting/Printing the Report /	308
To export the report /	309
To print the report /	312

Exporting Measurement Results to Web Repository /	314
Launching the Upload Results to Repository window /	314
Connecting to Dataset Web Server /	316
Accessing Service Account Credentials for Test Application /	327
Accessing the Web Repository for Datasets /	331
Selecting a Dataset /	333
Selecting and Uploading Measurement Results /	336
Defining and Modifying Dataset Properties /	352
Saving Test Projects /	380
To set AutoRecovery preferences /	380
Executing Scripts /	382
Controlling the Application via a Remote PC /	383
To identify the remote interface version /	383
To enable the remote interface /	384
To enable remote interface hints /	385
Switch Matrix Automation /	386
Controller Tab /	386
Signal Paths Tab /	389
Response Correction Tab /	396
Buttons /	397

19 Solving Problems

If there are problems connecting to instruments /	400
If there are problems with SCPI commands /	401
Adding Instruments Using Keysight Connection Expert /	401
Interactively Sending SCPI Commands to Instruments /	404
To see variable values as tests run /	407
If there are problems uninstalling a generated app /	410

Build Errors /	411
Error UDA100 /	411
Error UDA101 /	411
Error UDA200 /	412
Error UDA201 /	412
Error UDA300 /	412
Error UDA301 /	412
Error UDA400 /	412
Error UDA401 /	412
Error UDA500 /	412
Error UDA501 /	412
Error UDA502 /	413
Error UDA503 /	413
Error UDA504 /	413
Error UDA505 /	413
Error UDA506 /	413
Error UDA508 /	414
Error UDA600 /	414
Error UDA700 /	414
Error UDA701 /	414
Error UDA800 /	414
Restoring the Project /	415

20 Menu/Toolbar/Options Reference

Main Menu /	418
Toolbar Reference /	422
Options Reference /	423

21 Glossary

Index

1 Installation and Set Up

Requirements / 16

Installing the User Defined Application Generator / 18

Licensing / 19

Requirements

The User Defined Application Generator can be installed and used on:

- An Infiniium real-time or DCA-X oscilloscope – this may be easiest if the oscilloscope already has the necessary testing and post-processing software installed, because it is the same environment used to manually step through your tests, save setup files, etc.
- A development PC – for InfiniiVision apps, or for the other platforms if this is more convenient. You may have to spend some extra time installing the software that will be present in your oscilloscope test environment.

Requirements of a Development PC

When the User Defined Application Generator is used on a development PC (not the oscilloscope), your development PC requires:

- Keysight IO Libraries Suite, version 18.0 or greater.

You can download the Keysight IO Libraries Suite from the Keysight web site at: www.keysight.com/find/iosuite

Once the IO libraries are installed on your development PC, you can use its tools to add or troubleshoot connections to the oscilloscope and external instruments. See "[If there are problems with SCPI commands](#)" on page 401.

- Any additional console applications (like DUT controller programs) or external applications (like MATLAB or other data post-processing tools) that will be used with your generated application.

Requirements of Generated Applications

Applications generated by the User Defined Application Generator work on:

- (FlexDCA apps):
 - Keysight DCA-X Series oscilloscopes or PC, with FlexDCA software version 1.80 or greater.
- (Infiniium apps):
 - Keysight Infiniium MXR/EXR-Series oscilloscopes, with software version 11.05 or greater.
 - Keysight Infiniium UXR-Series oscilloscopes, with software version 10.0 or greater.
 - Keysight Infiniium 90000 Series, 90000 X-Series, 90000 Q-Series, and Z-Series oscilloscopes, with software version 1.41 or greater.
 - Keysight Infiniium 9000 Series oscilloscopes, with software version 2.00 or greater.
 - Keysight Infiniium 80000B Series oscilloscopes, with software version 5.70 or greater.
 - Keysight Infiniium 8000 Series oscilloscopes, with software version 5.70 or greater.
- (InfiniiVision/All apps):

- a PC.

The oscilloscope or PC must have Microsoft .NET version 4.5.2 or newer.

Installing the User Defined Application Generator

You can download the D9010UDAA User Defined Application Generator from the Keysight web site at: www.keysight.com/find/uda

You can install and run the User Defined Application Generator on the oscilloscope or on a development PC.

Your generated applications run on oscilloscopes and PCs. They can also be controlled from a remote PC using the N5452A Remote Interface for Automated Digital Test Applications (free download: www.keysight.com/find/rpi).

Licensing

The User Defined Application Generator is available at no cost.

In order to run generated applications or add-ins, however, a D9010UDAA license is required. Install the license on the same machine where the app/add-in will be running; it will be able to connect to and use any compatible oscilloscope.

NOTE

Legacy instrument-based N1019A and N5467B/C licenses are supported, but they continue to enable use of the instrument they are installed on only.

See the Keysight web site for information about ordering the D9010UDAA license.

1 Installation and Set Up

2 Getting Started Tutorial

Running Tests Manually and Preparing Setup Files / 22

Organizing Your Working Files / 23

Creating an Application or Add-In / 24

Installing and Running the Application / 61

For More Information / 68

Running Tests Manually and Preparing Setup Files

It is a good idea to perform your tests manually using the oscilloscope's front panel before trying to automate your test procedure. You can save setup files as you go.

Your generated applications and add-ins do not need to use setup files. They can just send "default setup" or "autoscale" commands instead of loading setup files. This works fine for simple applications.

However, for more complex applications, you may want to load setup files instead of trying to perform all the oscilloscope setup using SCPI commands. Performing as much of the oscilloscope set up as you can with setup files will simplify your test procedures.

Next • ["Organizing Your Working Files"](#) on page 23

Organizing Your Working Files

Many kinds of files can be used in an automated test application. We recommend you organize your working files and saved projects in the following directory structure:

```
C:\My UDA Files\
  Command Files\
  Connection HTML Files\
    images\
  Connection Image Files\
  Console Applications\
    cmd\
  External App Files\
  Help Files\
  Logo Files\
  Mask Files\
  Math Libraries\
  Other Files\
  Setup Files
  UDF Files\
C:\My UDA Projects\
```

The application generator copies your working files into similar directory structures when you:

- Test your applications and add-ins (it copies files to an internal cache before running your application).
- Create install packages.
- Save your projects into a project directory.

Even though the application generator copies your files to several places, it always needs access to the ones from your working directory.

This directory structure is helpful for preventing duplicate file names, when collaborating on projects, or when opening projects that others have shared. When you open a project (with full control instead of read-only), the application generator makes sure you have a working directory for the project's files and that all of the project's files are in your working directory.

CAUTION

Do not overlap the directories containing your working files and saved projects (otherwise, you may lose working files).

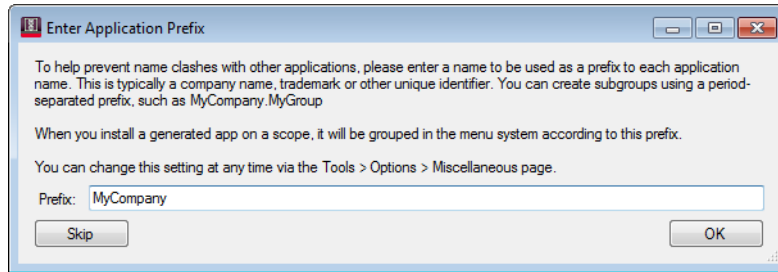
Next • ["Creating an Application or Add-In"](#) on page 24

Creating an Application or Add-In

- 1 From the Windows 10 Start menu, choose **All Apps > Keysight Scope Applications > User Defined Application**.

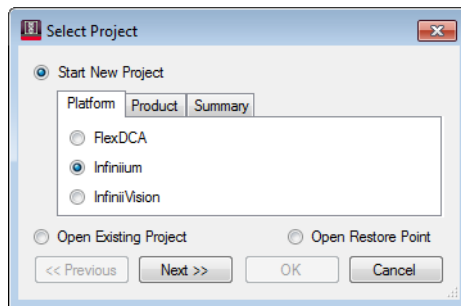
Or, from the Windows 7 Start menu, choose **All Programs > Keysight Scope Applications > User Defined Application > User Defined Application**.

The first time you start the application generator, you get this dialog box:



Enter your application prefix; then, click **OK**.

When you launch the application generator or start a new project, the following New Project dialog box appears.



When you use the application generator to create a "Complete Application", it will create an automated test application that can be installed on an oscilloscope or PC as a standalone product.

When you use the application generator to create an "Application Add-in", it will create one or more tests that can be installed into an existing "host" compliance application much like you can enhance modern web browsers with add-on additions. Each host application can contain one add-in at a time, although a single add-in may contain an unlimited number of tests. You may install the same add-in into multiple host apps.

NOTE

You cannot install an Add-in into a UDA application.

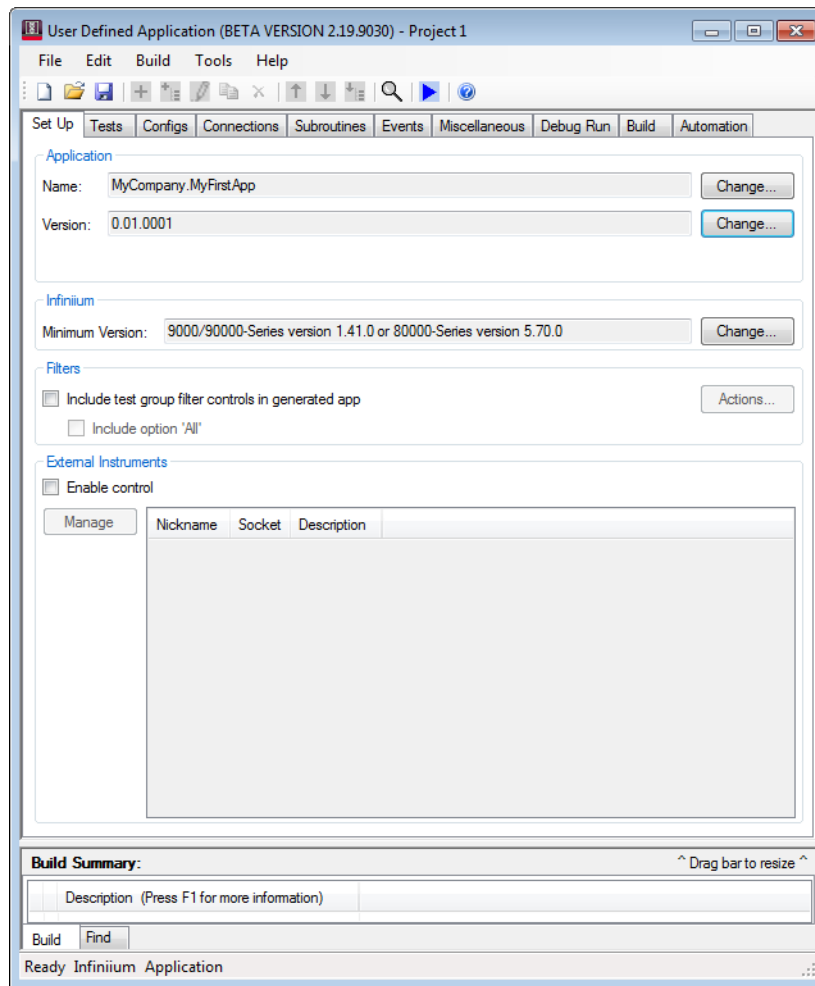
Differences between a complete application and an add-in include:

- You can define many categories of Run Events for an application while for an add-in you may only modify two events. Because your add-in's tests are subject to the other existing run event actions (if any) of its host, you may want to send the SCPI command `":system:preset"` in the Group User Defined Ends event to give your add-in a clean slate to start with. This will not impact the other tests in the host because the add-in's test group will always execute last.
- An application can have a help file associated with it and automatically generates a remote-interface help file. Neither of these apply to add-ins.
- An application's tests have positive test IDs (starting at 1) while an add-in has negative IDs (starting at -1001).
- When building an application, name collisions are detected automatically between:
 - External Instrument Nicknames
 - Test Group Names
 - Config Labels and Variable Names

When building an add-in, name collisions with the host are unknown until you install the add-in. Thus, to minimize collisions which require changes to your add-in whenever compliance application maintenance releases come out, it may be helpful to name your add-in's conflictible elements with a prefix that is unlikely to be used by the compliance application.

- When running the application generator on an oscilloscope, you can bypass the installer generation step and use "Install Now". For add-ins, you must generate an installer because it is executed by the host application.
- Users of an application can create user defined pass limits while users of an add-in cannot modify the add-in tests' limits.
- Compliance applications that enable the use of external instruments provide controls on the 'Set Up' tab to enable specifying the device's address while add-ins provide config variables for doing this (unless you reuse the host's device connection - see below).

Once you select to generate either a complete application or an application add-in, the User Defined Application generator's main window is displayed.



To create a user-defined application, follow these steps:

- **"Step 1: Give your application a name and version number"** on page 27
- **"Step 2: Add/Export/Import tests"** on page 28
- **"Step 3: Test the application"** on page 35
- **"Step 4: Add Config variables (optional)"** on page 38
- **"Step 5: Add connection instructions (optional)"** on page 43
- **"Step 6: Add/Export/Import subroutines (optional)"** on page 49
- **"Step 7: Add run actions (optional)"** on page 51
- **"Step 8: Add logos, images, help files (optional)"** on page 53
- **"Step 9: Specify Sample Projects (optional)"** on page 56
- **"Step 10: Specify Default User Preferences (optional)"** on page 57
- **"Step 11: Configure result detail limits (optional)"** on page 57
- **"Step 12: (FlexDCA and Infinium Only) Enable InfiniiSim"** on page 58

- **"Step 13: Generate the installer"** on page 59

Reusing Host Elements

In order to more tightly integrate your add-in with the host application, you may choose to have the add-in use certain elements already present in the host, including Configs and External Instruments.

To reuse a host application's config, define one in your add-in with the same variable name as the host's config. You do not have to enter valid description or choices since neither of these affect the build. Then, check the "Exclude this definition from the Add-in's installer" box in the Config Definition dialog box.

To reuse a host application's external instrument, define one in your add-in with the same nickname as the host's instrument. You don't have to enter valid description, error query, or no error string since none of these affects the build. Check the "Exclude this definition from the Add-In's installer" box in the External Instruments Definition dialog box.

Although when developing your add-in, you must still define these elements in order to enable successful building, you can exclude them from the add-in's installer so they will not conflict with the host. Then, at run-time, your add-in will automatically find the required elements in the host.

To find out the names of the elements used by your target host application, look in the application's (AppName)_Remote_Prog_Ref.chm file which can be viewed from the host's **Help > Remote Interface** menu. The file may also be found in its installation help directory on the oscilloscope, typically:

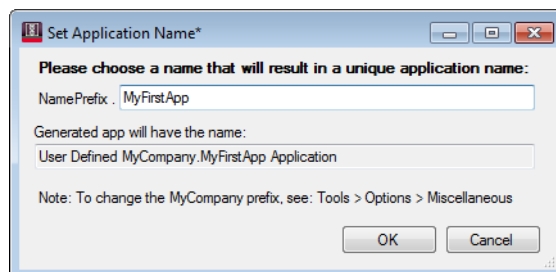
```
c:\program files\keysight\infiniium\apps\\help
```

-or on older oscilloscopes-

```
c:\scope\apps\\help
```

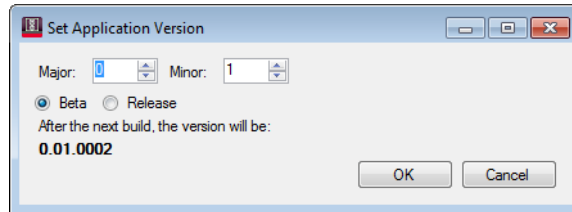
Step 1: Give your application a name and version number

- 1 In the User Defined Application generator's Set Up tab:
 - a To enter the application name, click **Set...**; then, in the Set Application Name dialog box, type in the name of your application and click **OK**.



- b To enter your application's version number, click **Change...**

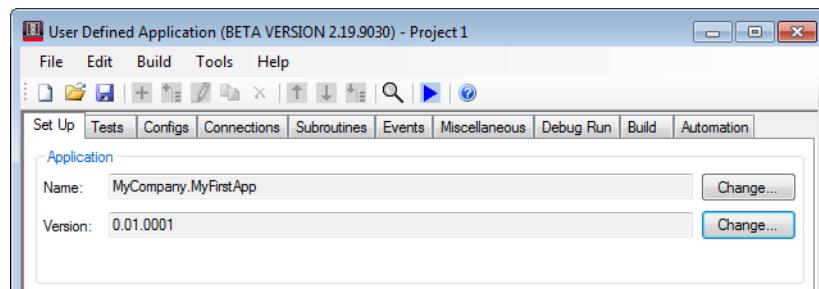
In the Set Application Version dialog box, select the major and minor version numbers. The last four digits of the version show the build number. The build number is incremented each time you build your application.



Then, select whether it is a beta or release version:

- Beta version applications show "Beta" and the version information in the main window's frame title.
- Release versions do not show the build number information.


When you are done, click **OK**.



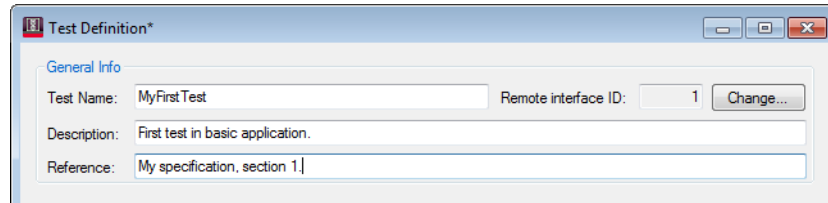
Next • **"Step 2: Add/Export/Import tests"** on page 28

Step 2: Add/Export/Import tests

Adding a test is all you need to do before generating and running your application.

- 1 In the User Defined Application generator's **Tests** tab, add a test by:
 - Clicking the **Add...** button.
 - Choosing the **Edit > Add Test...** command from the main menu.
 - Using the keyboard shortcut Ctrl+a.
 - Clicking the  Add Test toolbar icon.

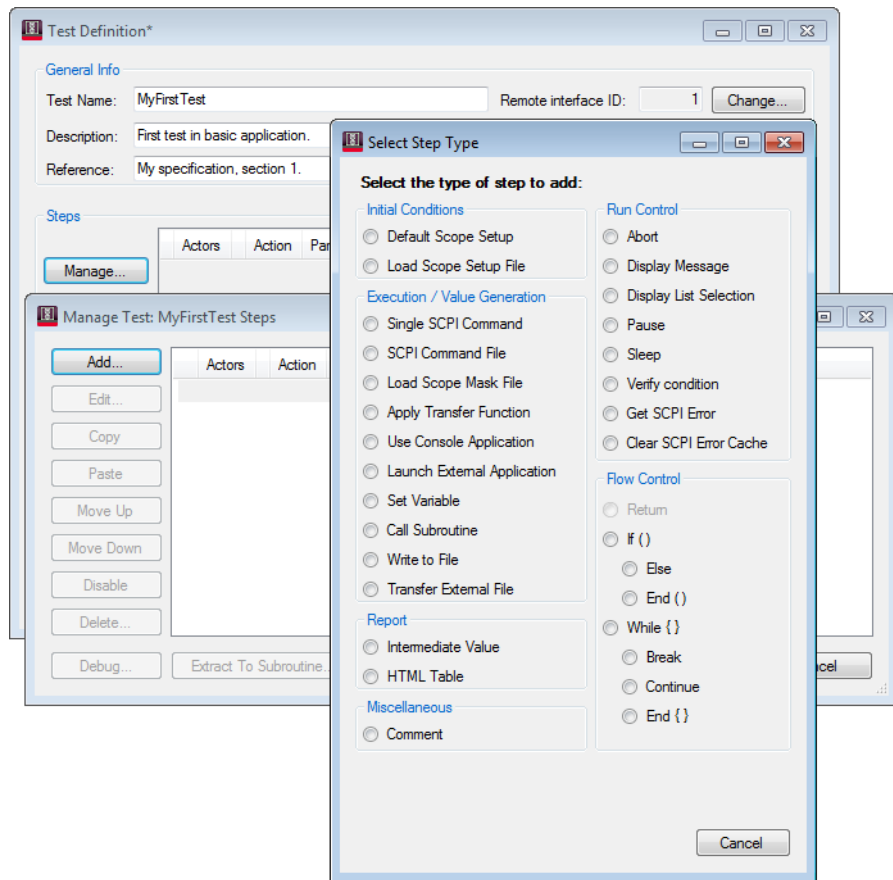
- 2 In the Test Definition dialog box:
 - a Enter the **Test Name**.
 - b Enter the **Description**.
 - c Enter any **Reference** information, for example, the section of a specification document that describes the rationale for the test.



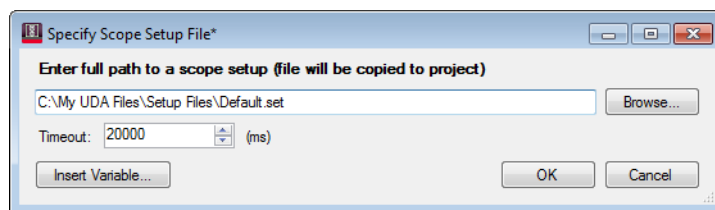
The test name, description, and reference appear in generated application's Select Tests tab.

(Ignore the remote interface ID number and **Change...** button for now. They are used with the application remote control feature.)

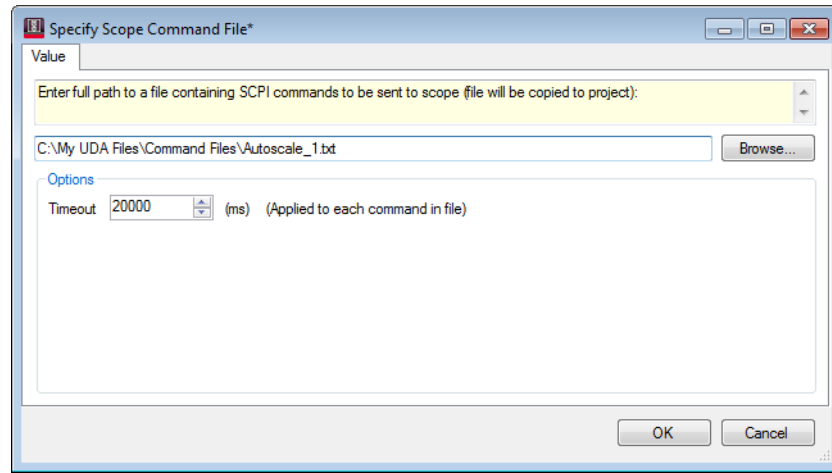
- 3 Specify one or more test steps.
 - a Click **Manage...**
 - b In the Manage Test dialog box, click **Add...**



- c In the Select Step Type dialog box, select the type of step you want to add. For example, if you select **Load Scope Setup File**, browse to select the setup file name.



- Then, click **OK**.
- d Click **Add...** again to add additional steps. For example, if you select **SCPI Command File**, browse to select the command file name.



Then, click **OK**.

SCPI command files are ASCII text files that contain SCPI commands. For example:

```
:AUToscale
# Comment: installed measurement appears in screen capture.
:MEASure:FREQuency
```

Command files only contain SCPI commands and comments. Comment lines begin with the "#" character.

Command files should not contain SCPI queries because query results are ignored.

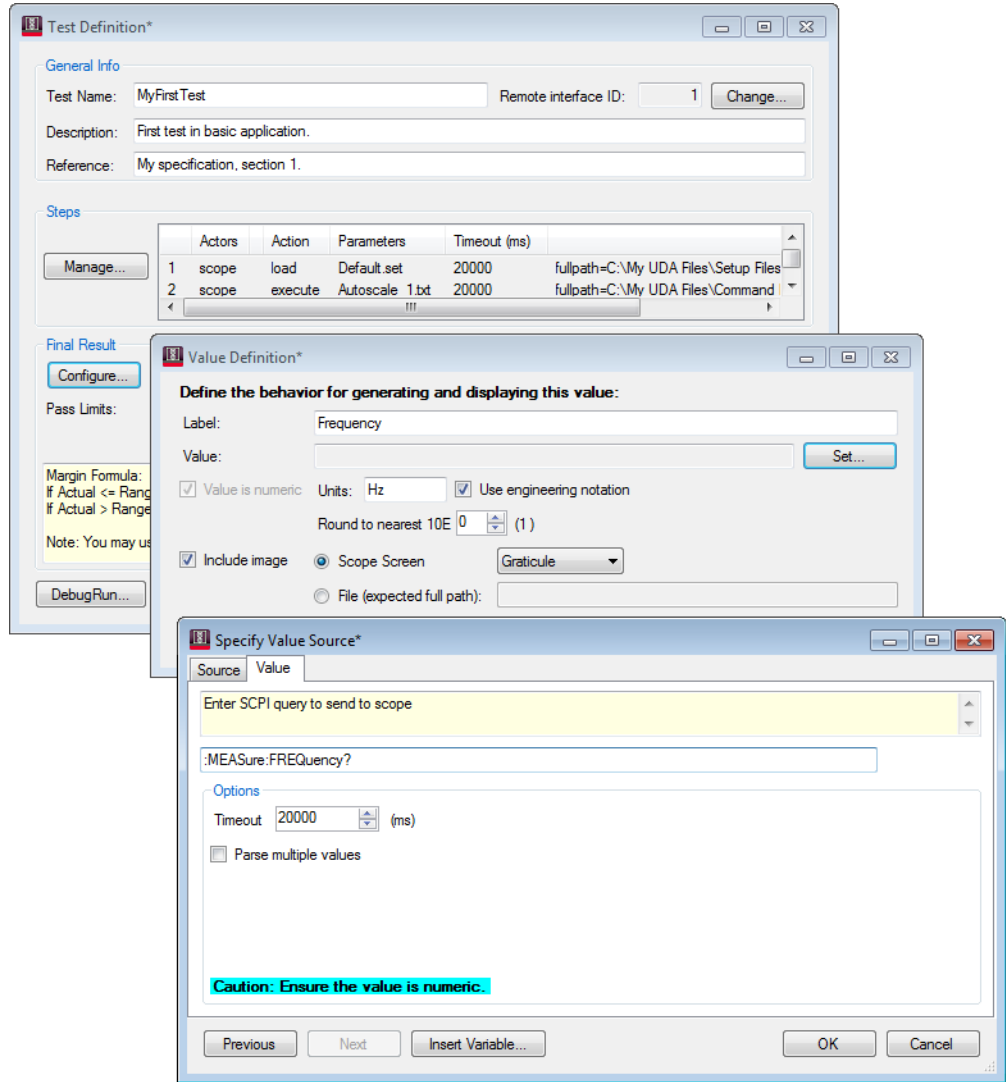
Command files typically contain oscilloscope setup commands.

- e When you are done adding steps, click **OK** to close the Manage Test dialog box.
- 4 In the Final Result area of the Test Definition dialog box, click **Configure...**
- 5 In the Value Definition dialog box:
 - a Enter a **Label** for the test result.
 - b Specify the **Value** to be used. (Click **Set...** and use the Specify Value Source dialog box.)
 - c Optionally, enter the **Units** of the result.
 - d Specify the **Round to nearest 10E** precision of the result.

CAUTION

Precision settings from 10E+18 through 10E-15 will round the final result and pass limits before they are compared for pass/fail determination, as well as round their displayed values. Precisions from 10E-16 through 10E-18 will only round the displayed value; pass/fail determination will be done at maximum precision.

- e Specify whether to **Include image** which is an oscilloscope **Graticule** or **FullScreen** capture.



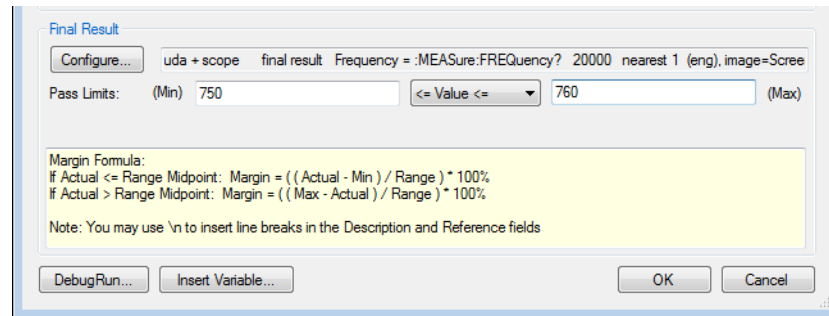
- f Click **OK**.
- 6 Enter the result **Pass Limits**:
 - a Select the limit type.

If you select "Information Only", the test will not pass or fail; it will simply report the final result.
 - b Enter the minimum and maximum values.

NOTE

You may enter either a number or an internal variable. When using a variable, do not enter expressions (such as "x+1").

- c If you selected a single-ended range type (for example, **Value <**, **<= Value**, etc.) and a value of 0 or internal variable, you may enter a "typical" value at which you want the margin to be 100%. The info box at the bottom of the dialog box describes how margin is calculated for the currently-selected limit type.



- 7 Click **OK**.

The test is listed in the **Tests** tab.

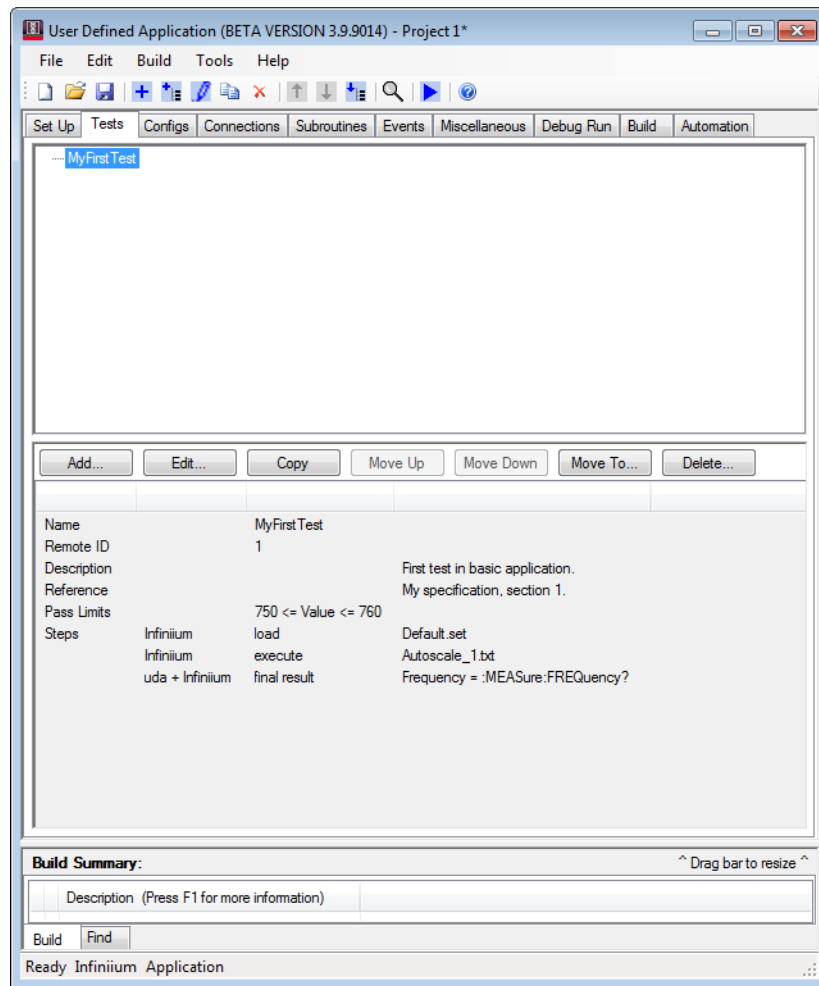
NOTE

When entering engineering-formatted numbers, use the following format:

1.2E+03

4.56E-07

Never start a number entry with the decimal point as in .89E+10



Config variables, connection instructions, run actions, and other capabilities are optional. If you like, you can go straight to **"Step 13: Generate the installer"** on page 59. However, you will generally want to test your application before generating an installer, and before you can test your application, you must identify the oscilloscope and its setup files location.

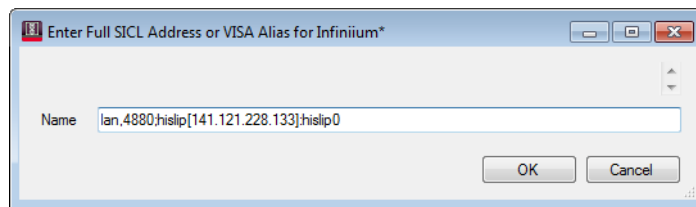
Once tests are added, they may be exported or imported in order to build up a library of UDA "fragments" that can be reused on other projects and shared with other users. For more information, see **"Exporting/Importing Tests or Subroutines"** on page 137.

- Next
- **"Step 3: Test the application"** on page 35
 - **"Step 4: Add Config variables (optional)"** on page 38
 - **"Step 13: Generate the installer"** on page 59

Step 3: Test the application

You can test your application while it is being developed, before generating an installer.

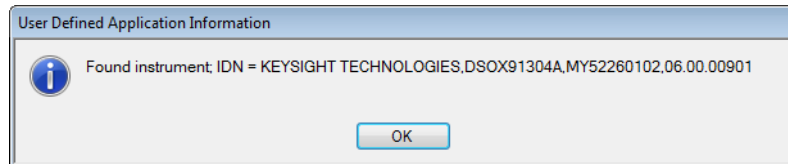
- 1 When working on a development PC (instead of an oscilloscope), you must identify the oscilloscope you want to test with. You might also need to enter additional information related to oscilloscope setup and mask files.
 - a In the User Defined Application generator's **Debug Run** tab, click **Set**.
 - b In the Enter Full Address or VISA Alias dialog box, enter the SICL address or VISA alias of the oscilloscope.



You can find an instrument's SICL address using the IO libraries' Keysight Connection Expert.

- c Click **OK**.

The SICL address is tested, and an information dialog box appears.



This same test can be performed back in the **Debug Run** tab by clicking **Test**.

- d Click **OK**.
 - e (Infiniium real-time only, up to version 4.10) Back in the **Debug Run** tab, if a Load Scope Setup File step or a Load Scope Mask File step will be executed, enter the **Setup File Path** or **Mask File Path** where the oscilloscope should look to find the file.

NOTE

For FlexDCA, InfiniiVision, and Infiniium real-time version 4.20 and newer, oscilloscope setup and mask files are automatically pushed to the oscilloscope even when the generated application is running on a PC.

There are three options for this path (depending on where you choose to put the file):

- **Option 1: Put the file on the oscilloscope (for example, C:\Uda Inputs\Setups):**

First, share the oscilloscope's directory where you put the file (for example, as "Uda Setups"). This allows the application generator running on the PC to locate the setup file when you create the test and build the application. Then, in the Debug Run tab, enter the local path on the oscilloscope to the directory where you put the file (for example, C:\Uda Inputs\Setups). This is the path that will be used by the oscilloscope to locate the file during the generated app's test execution.


- **Option 2: Put the file on the PC (for example, C:\MyUdaUserFiles\Setups):**

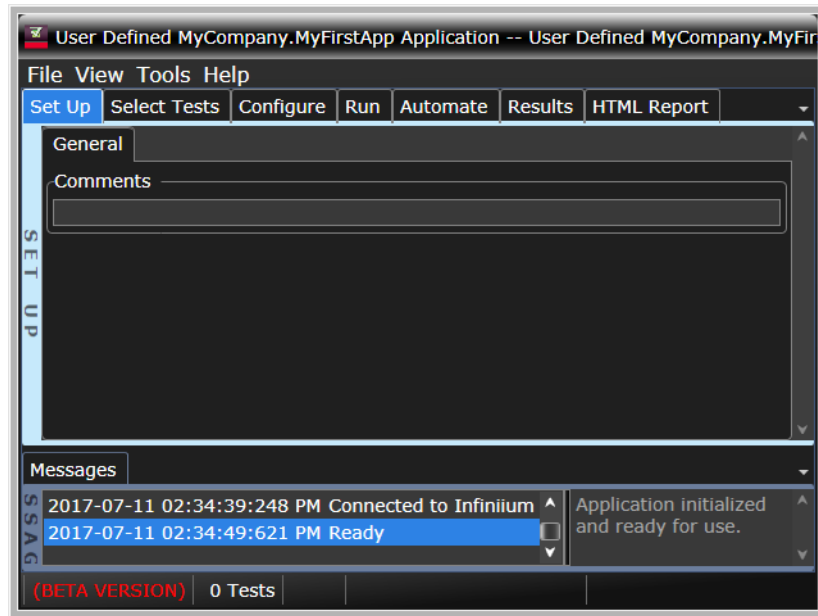
First, share the PC's directory where you put the file (for example, as "Uda Setups"). Then, in the Debug Run tab, enter the network path to the share on the PC (for example, \\mypc\Uda Setups).

- **Option 3: Put the file on a shared network drive visible to both the scope and the PC:**

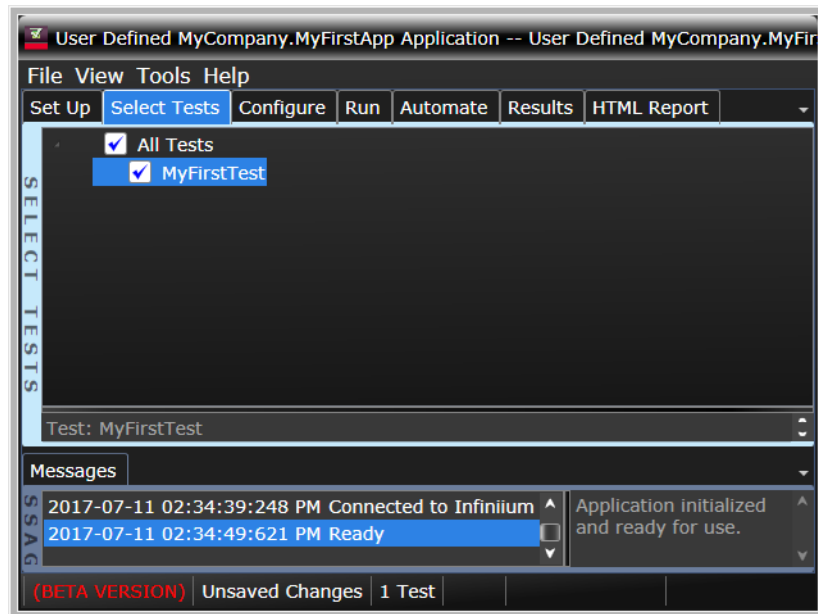
In the Debug Run tab, enter the network path to the share (for example, \\myserver\Uda Setups).

For all options, copy any setup files used by your application to the specified location.

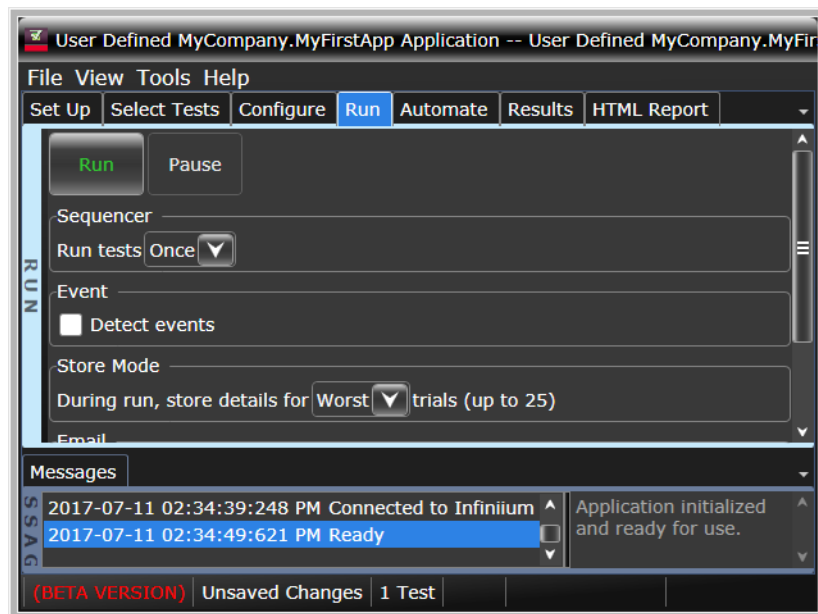
- 2 Click the  Build and Launch Application toolbar icon.
- 3 When the generated application window opens, in the Set Up tab, enter any desired setup information.



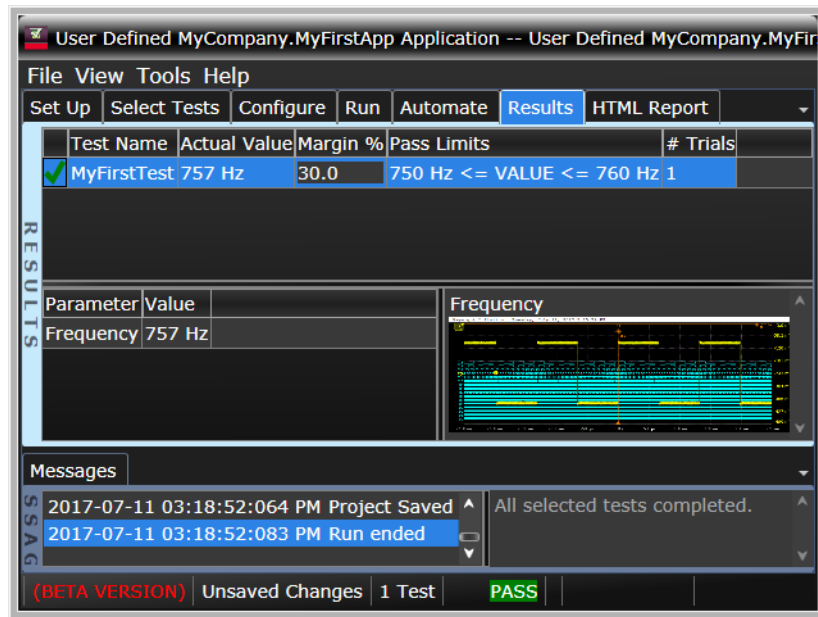
- 4 In the Select Tests tab, select the test.



- 5 In the Run tab, click **Run**.



- 6 And, after tests are complete, in the Results tab, view the results.




Config variables, connection instructions, run actions, and other capabilities are optional. If you like, you can go straight to **"Step 13: Generate the installer"** on page 59.

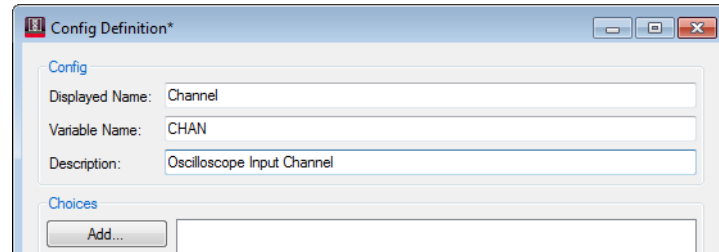
- Next
- **"Step 4: Add Config variables (optional)"** on page 38
 - **"Step 5: Add connection instructions (optional)"** on page 43
 - **"Step 6: Add/Export/Import subroutines (optional)"** on page 49
 - **"Step 7: Add run actions (optional)"** on page 51
 - **"Step 8: Add logos, images, help files (optional)"** on page 53
 - **"Step 9: Specify Sample Projects (optional)"** on page 56
 - **"Step 10: Specify Default User Preferences (optional)"** on page 57
 - **"Step 11: Configure result detail limits (optional)"** on page 57
 - **"Step 13: Generate the installer"** on page 59

Step 4: Add Config variables (optional)

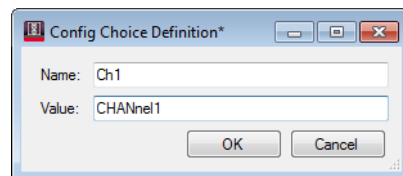
Config variables are optional. They allow the user of the generated application to make choices that impact test execution. If you like, you can skip the Config variable steps and go straight to **"Step 13: Generate the installer"** on page 59.

- 1 In the User Defined Application generator's **Configs** tab, add a Config variable by:
 - Clicking the **Add...** button.
 - Choosing the **Edit > Add Config...** command from the main menu.
 - Using the keyboard shortcut **Ctrl + a**.

- Clicking the  Add Config toolbar icon.
- 2 In the Config Definition dialog box:
 - a Enter the **Displayed Name**.
 - b Enter the **Variable** name.
 - c Enter the **Description**.



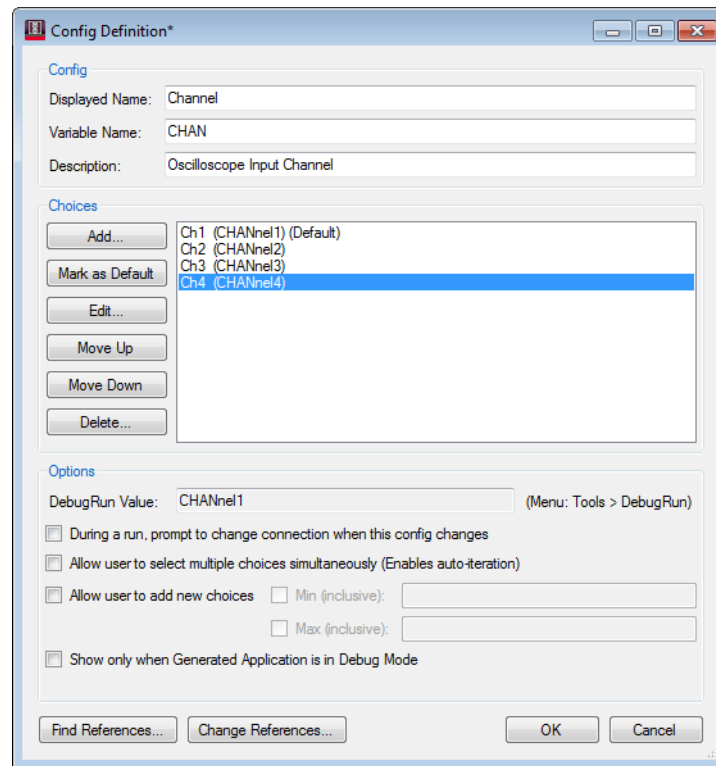
- 3 Now, add selection choices by clicking **Add**.
In the Config Choice Definition dialog box:
 - a Enter the choice name. This is displayed in the generated application's combo box.
 - b Enter the choice value. This is used as SCPI command parameters.



- c Click **OK**.

Repeat these substeps for each selection choice.

- 4 If necessary, select the choice you want to be the default; then, click **Mark as Default**.



- 5 If the Config variable has already been defined, you can click **Set** to set the config variable to one of its values for use in a Debug Run.
- 6 Check **During a run, prompt to change connection when this config changes** if you are using connection instructions (see [Chapter 8](#), "Adding Connection Instructions," starting on page 151) and this config variable controls a setting that impacts the current connection.

In the generated app, when the user selects a different choice for this config, the app's "current connection" will be invalidated thus causing a connection prompt when the next test is run.

- 7 (Apps Only) Check **Allow user to select multiple choices simultaneously** if you want the user to be able to simultaneously select more than one choice for this config.

This option automatically enables "auto-iteration", described in ["Config Variable Auto-Iteration"](#) on page 292.

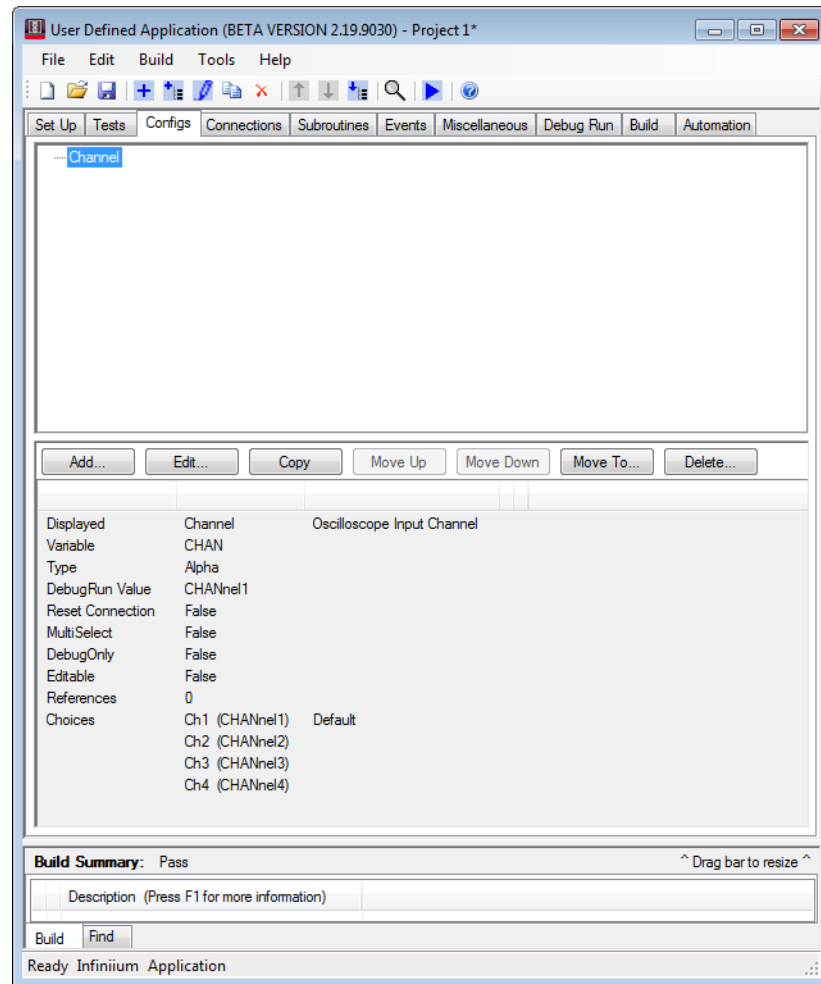
- 8 Check **Allow user to add new choices** if you want to give users of the generated application that ability.

If you check this option, you may optionally specify a valid range of acceptable values.

- 9 (App mode only) Check **Show only when Generated Application is in Debug Mode** if you want to hide this config from users until they put the generated app into Debug Mode (via its Configure tab controls).

- 10 *(Add-in mode only) Check **Exclude...** to enable the add-in to re-use the host application's config (same variable name).
- 11 Click **OK** to close the Config Definition dialog box.

The Configs tab shows the added Config variable definition.

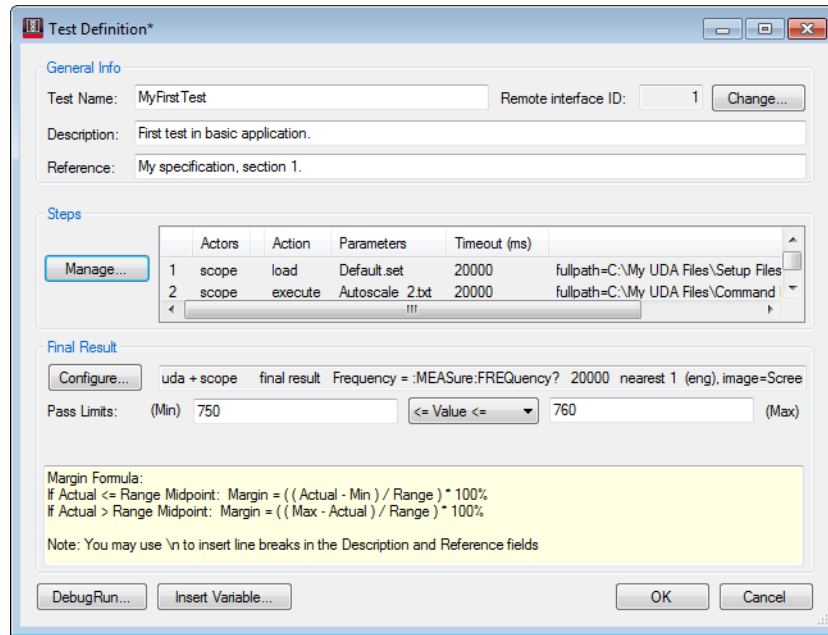


- 12 Now, you can include Config variable definitions in SCPI commands, command files, connection descriptions, etc., by using this format:

```
%VAR:MyVariableName%
```

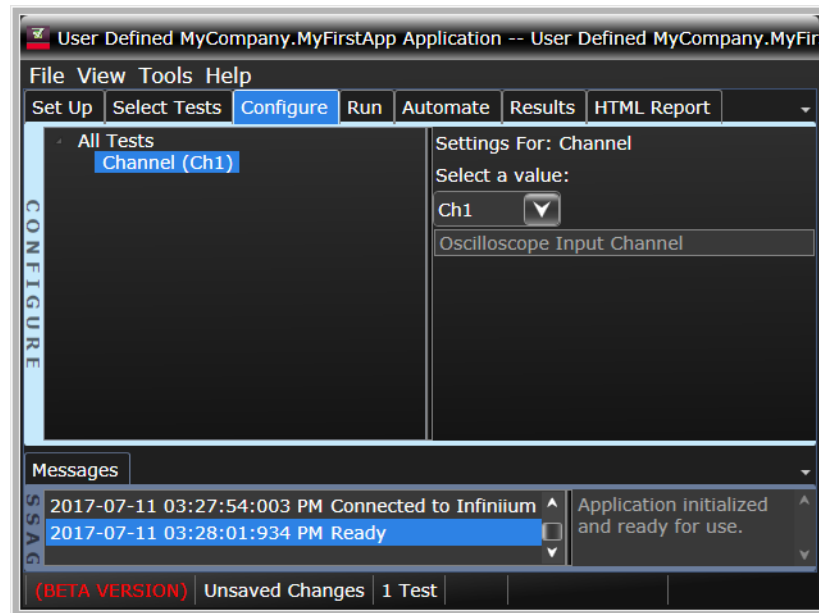
For example, you can add the Config variable to a test's command file and result SCPI query:

```
:AUToscale
# Comment: installed measurement appears in screen capture.
:MEASure:FREQuency %VAR:CHAN%
```



13 Click the  Build and Launch Application toolbar icon.

In the application's Configure tab, you will see the configuration variable selection:




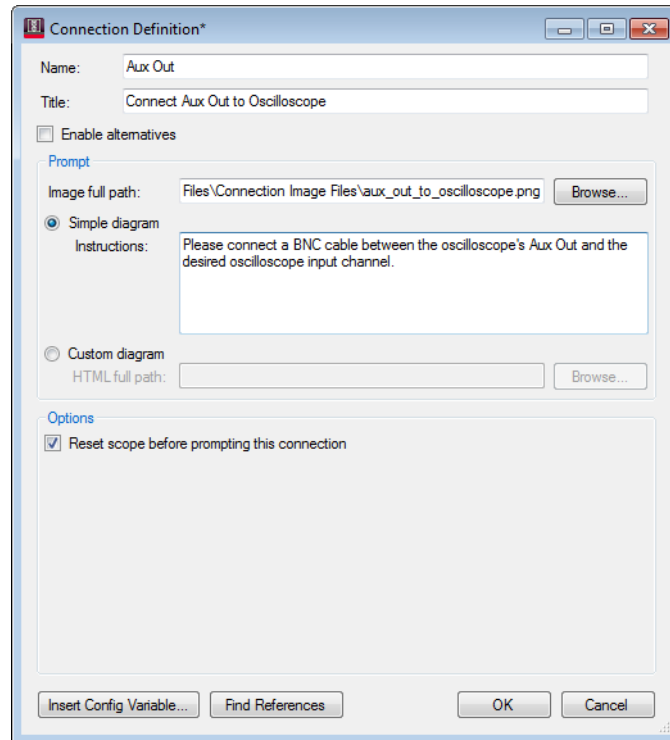
Connection instructions, run actions, and other capabilities are optional. If you like, you can go straight to **"Step 13: Generate the installer"** on page 59.

Next • **"Step 5: Add connection instructions (optional)"** on page 43

- "Step 6: Add/Export/Import subroutines (optional)" on page 49
- "Step 7: Add run actions (optional)" on page 51
- "Step 8: Add logos, images, help files (optional)" on page 53
- "Step 9: Specify Sample Projects (optional)" on page 56
- "Step 10: Specify Default User Preferences (optional)" on page 57
- "Step 11: Configure result detail limits (optional)" on page 57
- "Step 13: Generate the installer" on page 59

Step 5: Add connection instructions (optional)

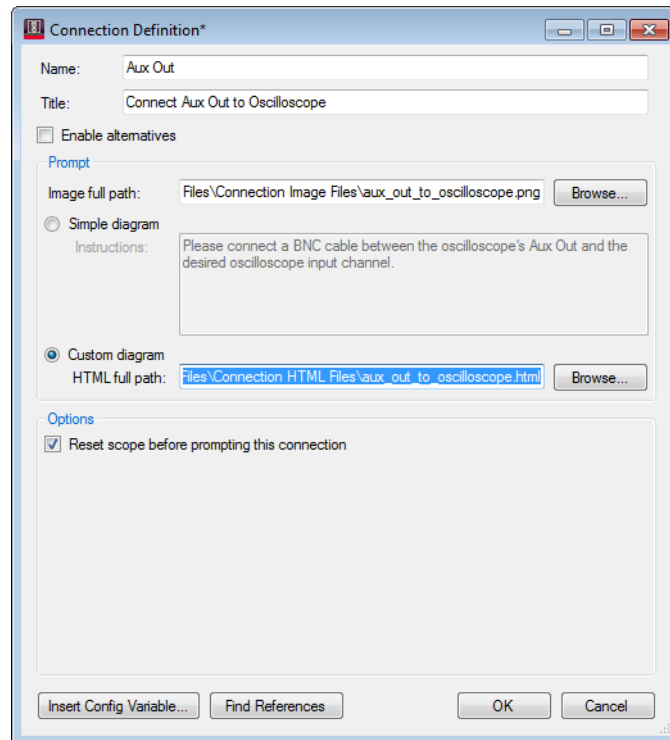
- 1 In the User Defined Application generator's **Connections** tab, add connection instructions by:
 - Clicking the **Add...** button.
 - Choosing the **Edit > Add Connection...** command from the main menu.
 - Using the keyboard shortcut Ctrl + a.
 - Clicking the  Add Connection toolbar icon.
- 2 In the Connection Definition dialog box:
 - a Enter the **Title**.
 - b (Optional) Check **Enable Alternatives** to use a text variable to control which connection prompt gets displayed.
 - c (Optional) Select an **Image file** that illustrates the connection.
 - d Select either **Simple diagram** or **Custom diagram**.
 - e If you selected **Simple diagram**, enter connection instructions. Type ALT+I to insert a config reference, specifying whether you want the instructions to show the choice's name or value when connection prompt is displayed.



- f If you selected **Custom diagram**, select an HTML file that contains the image reference and connection instructions.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
  <meta content="text/html; charset=ISO-8859-1" http-equiv="Content
-Type">
  <title>Connect Aux Out to Oscilloscope</title>
</head>
<body>
<h4>Connect Aux Out to Oscilloscope</h4>
<ol>
  <li>
    <p>Please connect a BNC cable between the oscilloscope's Aux
      Out and the desired oscilloscope input channel.</p>
    <p></p>
  </li>
  <li>
    <p></p>
  </li>
  <li>
    <p></p>
  </li>
</ol>
</body>
</html>
```

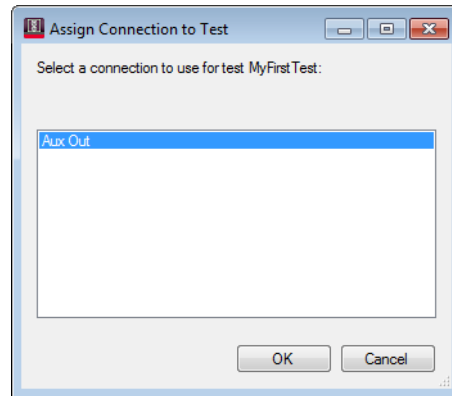
Note that when testing your application or generating an installer, and your working files are copied to an internal cache or install directory, the image file is copied to an "images/" subdirectory relative to the HTML file. Therefore, make sure the tag in your HTML source file includes "images/" in the src attribute (as shown above).



- g Check **Reset scope** to cause the generated app to reset the oscilloscope before displaying the connection prompt.
- h Click **OK**.

The connection definition now appears in Connections tab.

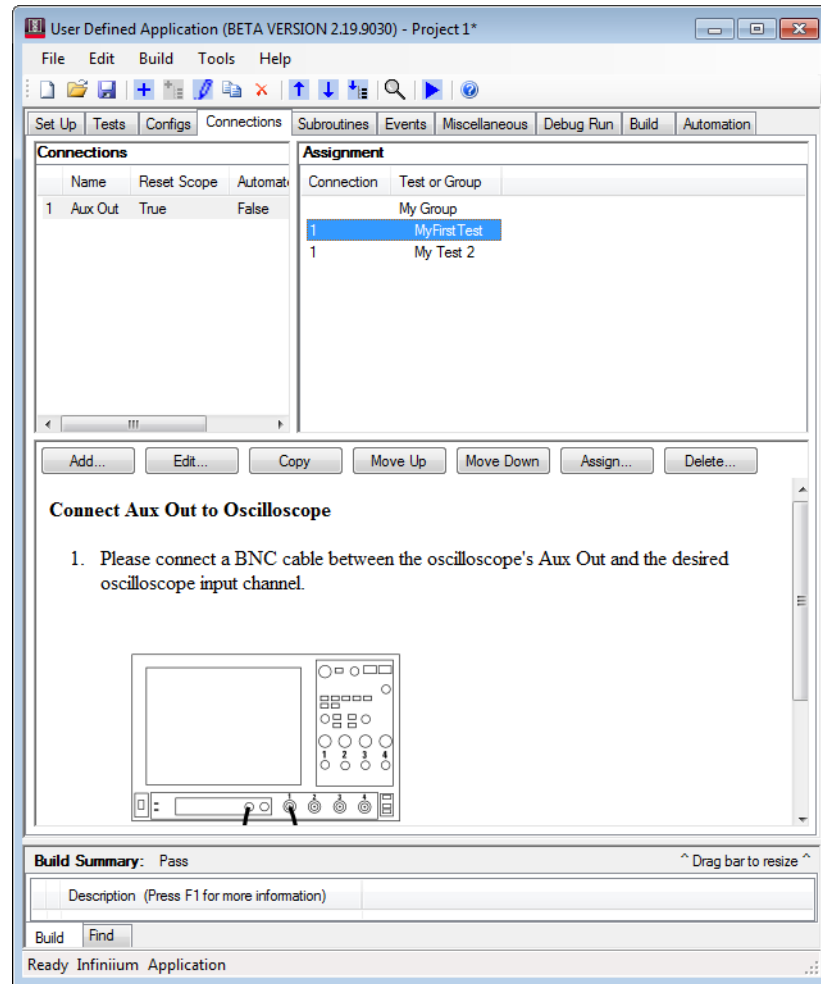
- 3 To assign the connection to tests:
 - a Select the test and click **Assign...**
 - b In the Assign Connection to Test dialog box, select the connection to use for the test.



c Click **OK**.

Repeat these substeps for each test you want to assign a connection to.

After you are done, the Connections tab shows which connection instructions are assigned to which tests.



The left pane is a simple list of connection instructions and the right pane is a tree of test groups and tests. You can also select a group and click **Assign...** to associate it with a connection. Associating a connection to an entire group enables the user to take advantage of this sequence of events:

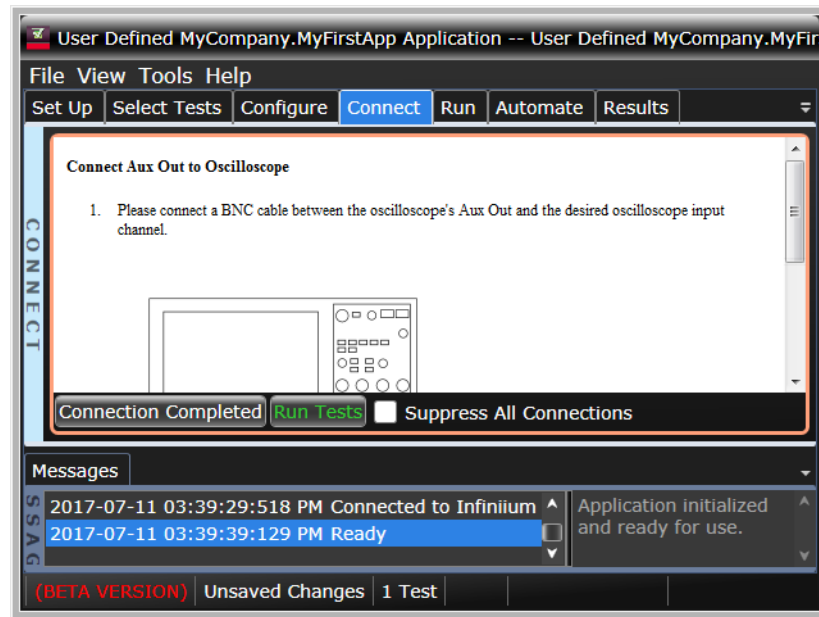
At runtime, the following actions occur in this order:

- a Connection Instructions associated the current test group are displayed.
- b Event steps associated with the current test group are executed.
- c Connection instructions associated with the first test to be executed are displayed.

So, if you want connection instructions to display before a group's event executes, you need to associate the connection with the group.

- 4 Click the  Build and Launch Application toolbar icon.

In the application's Connect tab, you will see the connection instructions:



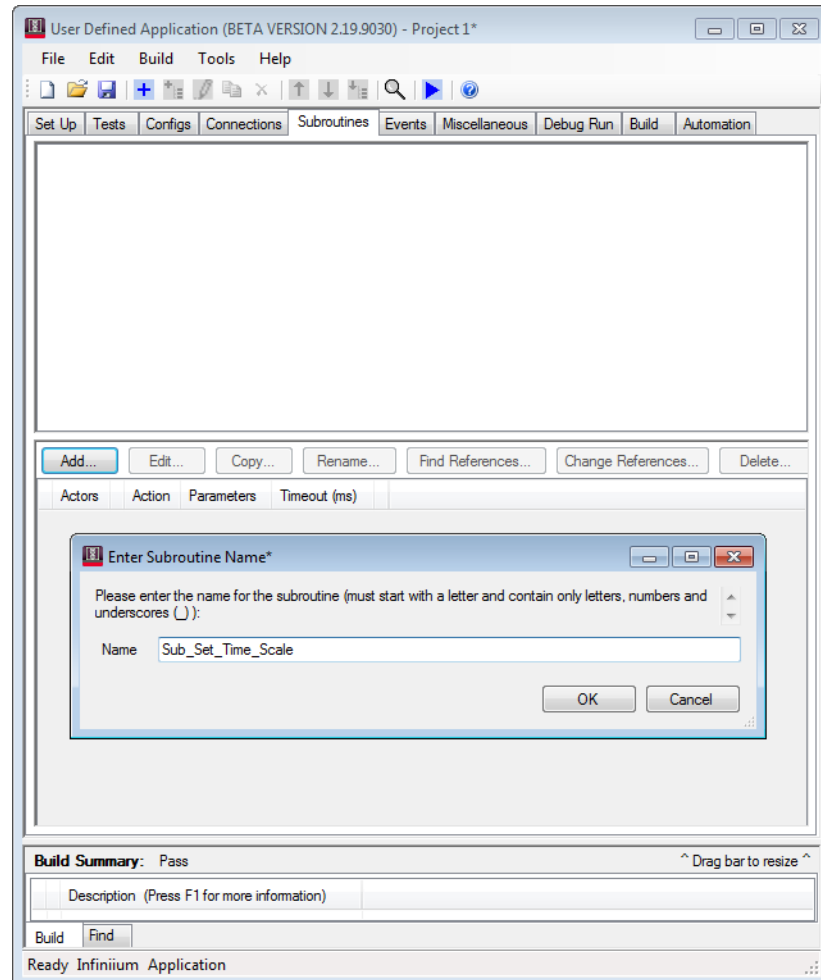
Run actions and other capabilities are optional. If you like, you can go straight to **"Step 13: Generate the installer"** on page 59.

- Next
- **"Step 6: Add/Export/Import subroutines (optional)"** on page 49
 - **"Step 7: Add run actions (optional)"** on page 51
 - **"Step 8: Add logos, images, help files (optional)"** on page 53
 - **"Step 9: Specify Sample Projects (optional)"** on page 56
 - **"Step 10: Specify Default User Preferences (optional)"** on page 57
 - **"Step 11: Configure result detail limits (optional)"** on page 57
 - **"Step 13: Generate the installer"** on page 59

Step 6: Add/Export/Import subroutines (optional)

You can use the Subroutines tab to define a set of steps that can be called from tests, events, or other subroutines.

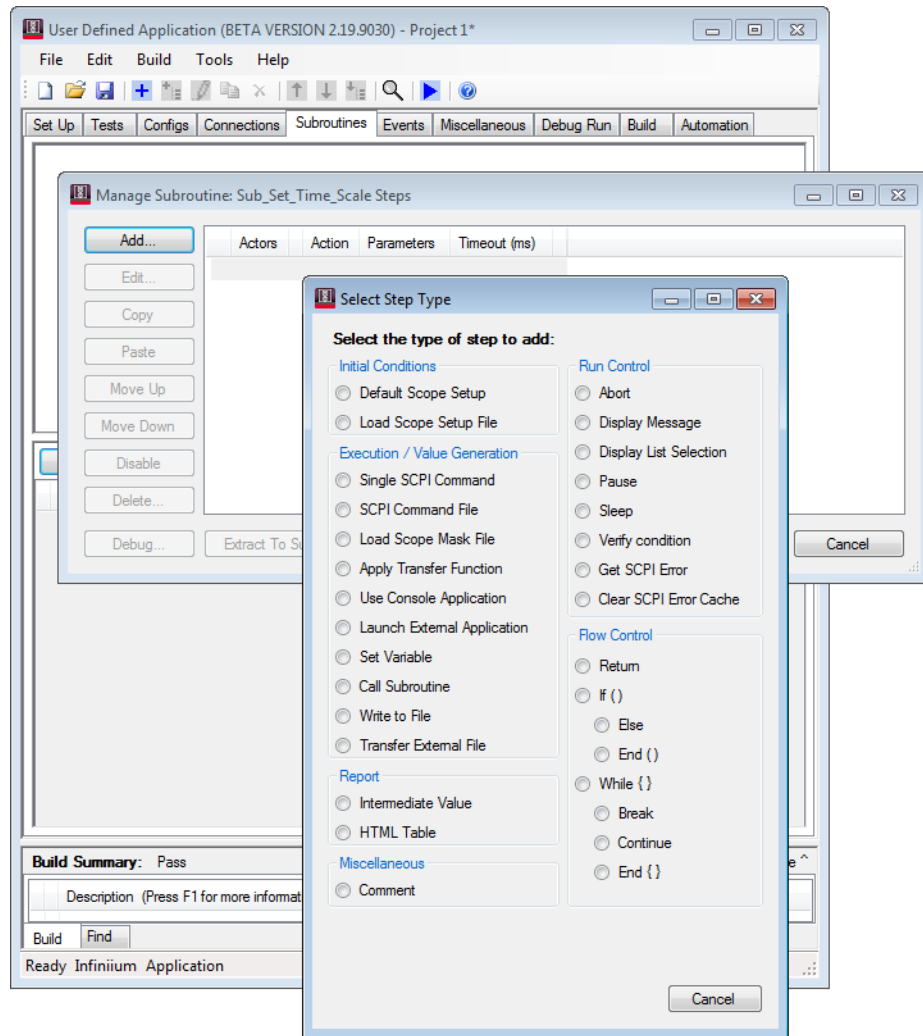
- 1 In the User Defined Application generator's Subroutines tab, first click **Add...**; then, enter the name of the subroutine into the Enter Subroutine Name dialog box. Then, click **OK**.



- 2 The Manage Subroutine dialog box then appears where you can:
 - Define steps (**Add...**).
 - Edit steps (highlight the step and press **Edit...**).
 - Copy steps (**Copy**).
 - Paste steps (**Paste**).
 - Move steps up or down the list (**Move Up** and **Move Down**).
 - Disable a step (**Disable**).

- Delete a step (**Delete...**).

The following screen shot shows the various step types you can define after clicking **Add...**.



You add steps to a subroutine in the same way as you them to a test.

Once you have defined a subroutine, you can add it to a test's steps or another subroutine's steps in the same way as you add steps – just select the **Call Subroutine** step type.

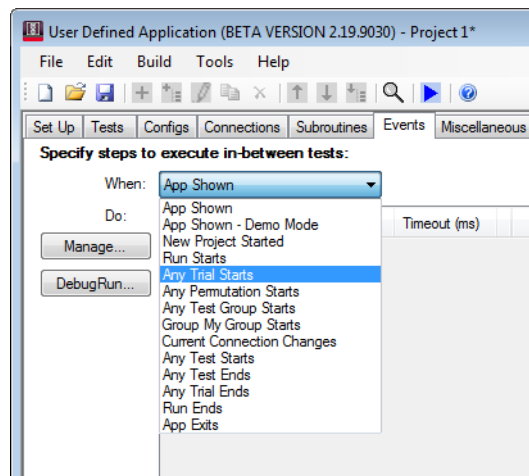
Once subroutines are added, they may be exported in order to build up a library of UDA "fragments" that can be reused (imported) on other projects and shared with other users. For more information, see **"Exporting/Importing Tests or Subroutines"** on page 137.

- Next • **"Step 7: Add run actions (optional)"** on page 51

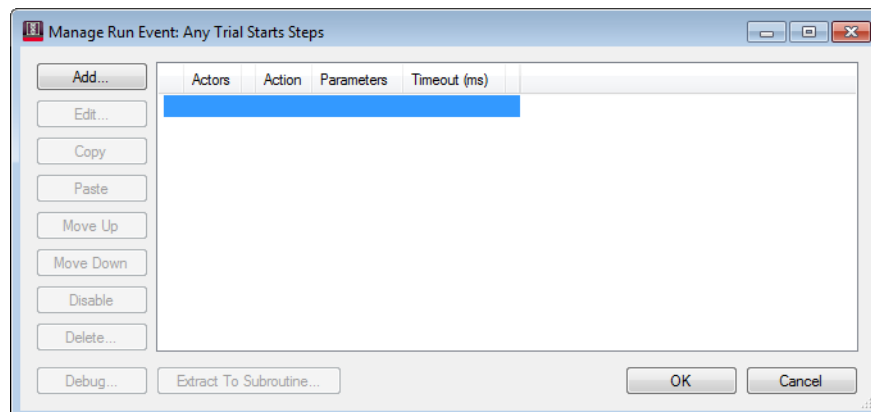
- "Step 8: Add logos, images, help files (optional)" on page 53
- "Step 9: Specify Sample Projects (optional)" on page 56
- "Step 10: Specify Default User Preferences (optional)" on page 57
- "Step 11: Configure result detail limits (optional)" on page 57
- "Step 13: Generate the installer" on page 59

Step 7: Add run actions (optional)

- 1 In the User Defined Application generator's **Events** tab, first select when the action should take place.

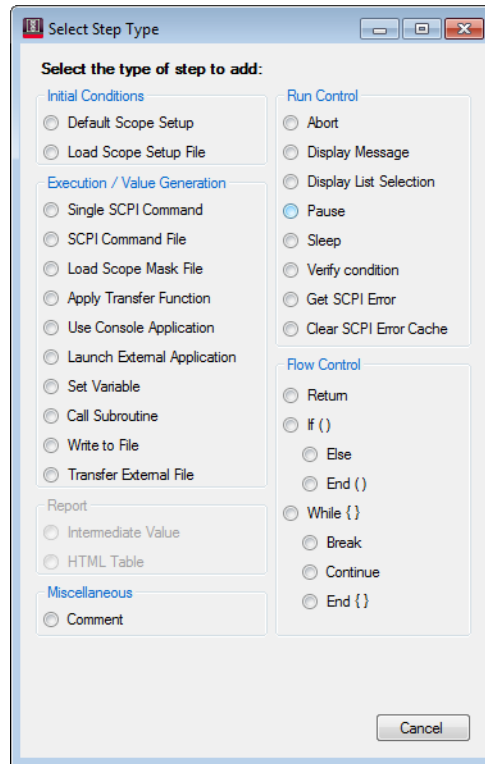


- 2 Then click **Manage...** to open the Manage Run Starts Steps dialog box.



- 3 Then, click **Add...**
- 4 In the Select Step Type dialog box, select the type of step you want to add, enter any additional information, and click **OK**.

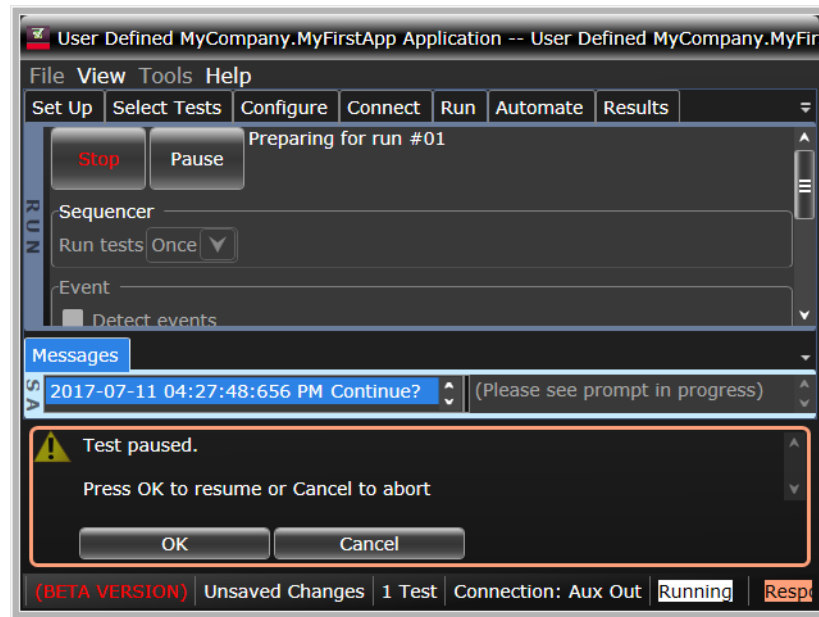
For example to add a display message step, select **Pause** and click **OK**.



The added step is then displayed in the Events tab.

- 5 Click the  Build and Launch Application toolbar icon.

When you run the tests, you will see the results of the added run actions:



- 6 The **Default Scope before app** option is shown. Uncheck this option to prevent the generated app from loading the default oscilloscope setup when it launches.

The **Default Scope at start of run** option is shown. Uncheck this option to prevent the generated app from loading the default oscilloscope setup at the start of each run.

NOTE



(Infiniium real-time only) the generated application's **Tools > Infiniium > InfiniiSim** menu will only be shown when this option is checked.

Other capabilities are optional. If you like, you can go straight to "[Step 13: Generate the installer](#)" on page 59.

- Next
- "[Step 8: Add logos, images, help files \(optional\)](#)" on page 53
 - "[Step 9: Specify Sample Projects \(optional\)](#)" on page 56
 - "[Step 10: Specify Default User Preferences \(optional\)](#)" on page 57
 - "[Step 11: Configure result detail limits \(optional\)](#)" on page 57
 - "[Step 13: Generate the installer](#)" on page 59

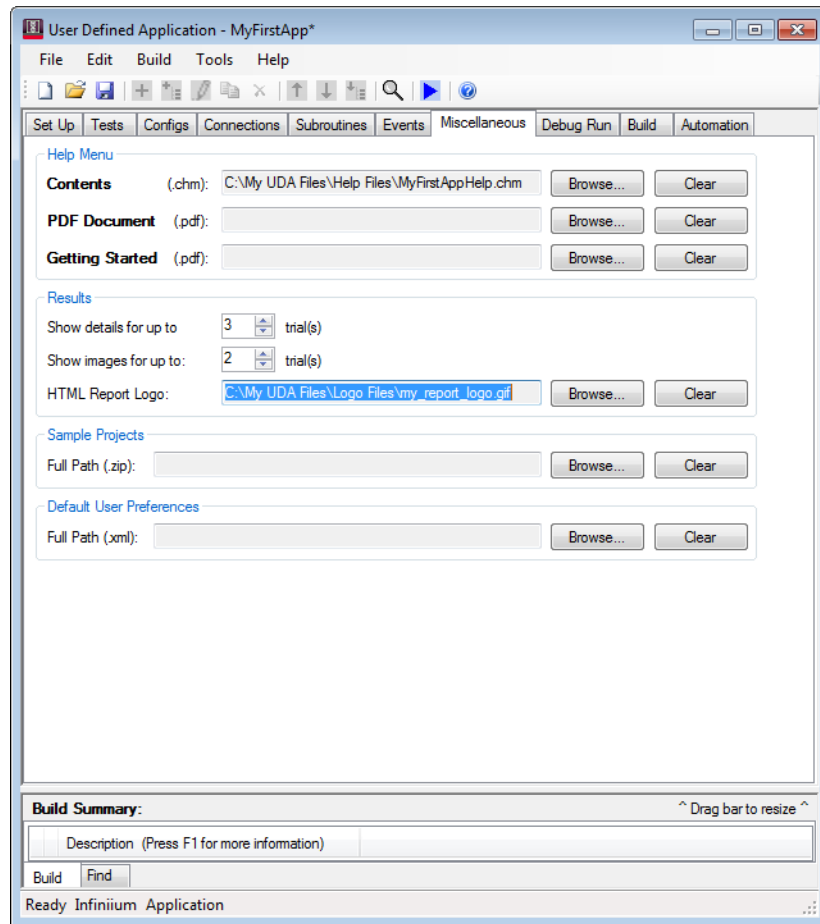
Step 8: Add logos, images, help files (optional)

Suppose you want to add the following files to your application:

 <p>MyFirstAppHelp.chm Compiled HTML Help File 1,020 KB</p>	<p>A Microsoft HTML Help format online help file that describes how to use your application.</p>
	<p>A GIF logo image for the generated HTML reports. This image can be any size. It appears at the top left of the HTML report..</p>

To do this, use options in the application generator's Miscellaneous tab:


- 1 In the User Defined Application generator's **Miscellaneous** tab, you may optionally select the online help .chm file, the PDF file, and a Getting Started PDF file.



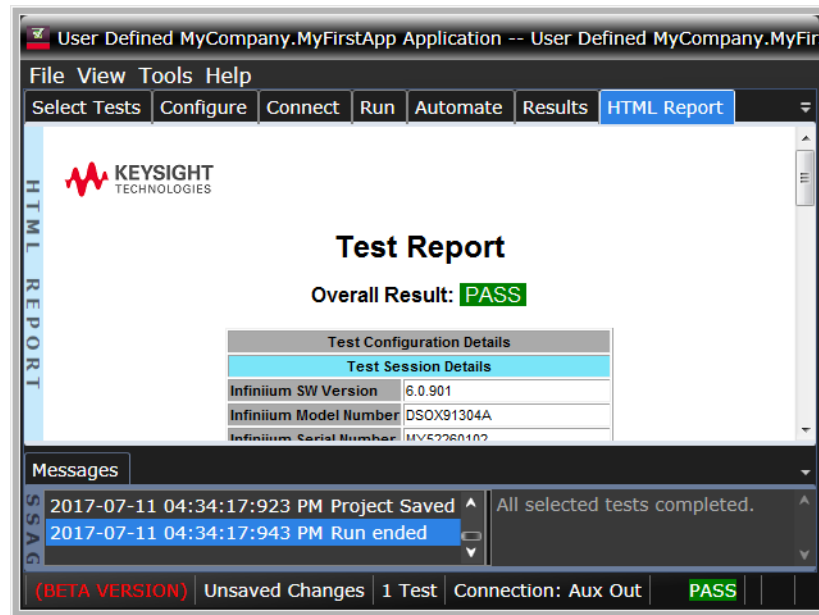
These will be installed to the generated app's help directory "C:\Program Files (x86)\Keysight\

- **Help > Contents**
- **Help > View Test Procedure**

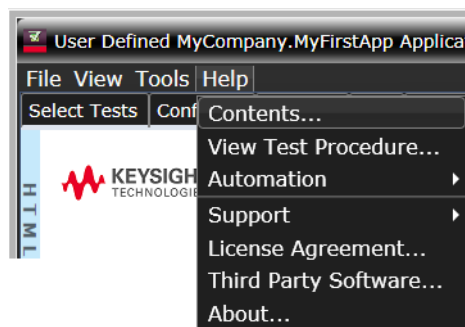
- **Help > Getting Started**

- 2 In the Results box, browse to select the GIF format image file to be displayed in the HTML report.
- 3 Click the  Build and Launch Application toolbar icon.

After you have run the tests, you will see the image in the splash screen:

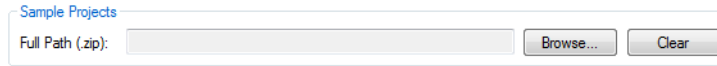


Finally, if you choose **Help > Contents...** from your application's main menu, you will see the online help:



- Next
- **"Step 9: Specify Sample Projects (optional)"** on page 56
 - **"Step 10: Specify Default User Preferences (optional)"** on page 57
 - **"Step 11: Configure result detail limits (optional)"** on page 57
 - **"Step 13: Generate the installer"** on page 59

Step 9: Specify Sample Projects (optional)



Sometimes it is helpful for a generated app's developer to provide sample test results to users. To do this:

- 1 Run the Generated App and execute one or more tests.
- 2 Save the test results. The generated app will create a directory structure such as:

```
... \Project Name \ (multiple files, including the .proj file)
... \Project Name \app \ (multiple files)
... \Project Name \etc.
```

- 3 Compress the directory tree (in the example above, ... \Project Name) to a .zip file.

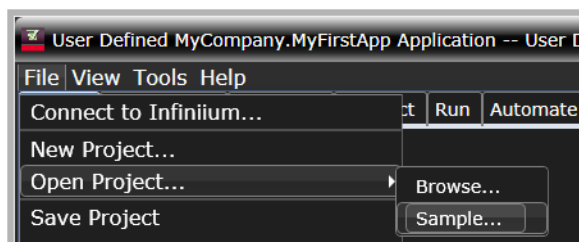
NOTE

You can include multiple project directories in your zip file to install multiple sample projects.

- 4 Browse to, or enter, the full path to the .zip file.

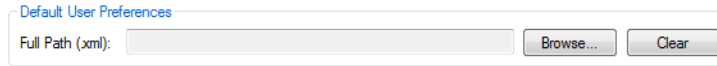
When you install the generated app, the .zip file will be inflated and installed to a "Sample Projects" subdirectory in the generated app's default project location.

In the generated app, you will be able to access the sample projects through:



- Next
- **"Step 10: Specify Default User Preferences (optional)"** on page 57
 - **"Step 11: Configure result detail limits (optional)"** on page 57
 - **"Step 13: Generate the installer"** on page 59

Step 10: Specify Default User Preferences (optional)



To make and provide a user preferences XML file:

- 1 Install the generated app and launch it.
- 2 Modify settings as desired in the **View > Preferences** window.
- 3 Exit the app and copy this file:

```
C:\ProgramData\Keysight\Infiniium\Apps\User Defined\<app name>\
userprefs.xml
```

to where you store your user files.

- 4 Browse to this new file to include it in the UDA project.

Now, your generated app installer will deliver this file to your users to be used as the *initial* user preferences file to affect the *first launch* of the app only. After that, users are free to make changes using the generated app's **View > Preferences** window and those settings will then become the active user preferences.

- Next
- **"Step 11: Configure result detail limits (optional)"** on page 57
 - **"Step 13: Generate the installer"** on page 59

Step 11: Configure result detail limits (optional)

The tests in a generated application may be run continuously, waiting for the user to press "stop". For this reason, limits must be placed on:

- 1 The number of trials per test that will have viewable details (to prevent the "Results" tab from having too many trial result tabs). However, all trials executed will still be included in the overall run statistics, and details for all trials are available in the generated app's CSV results (see **File > Export Results**).
- 2 The number of trials per test that may have scope images (to protect the hard drive from filling up, or to improve runtime performance when you only want images for the worst few trials).

NOTE

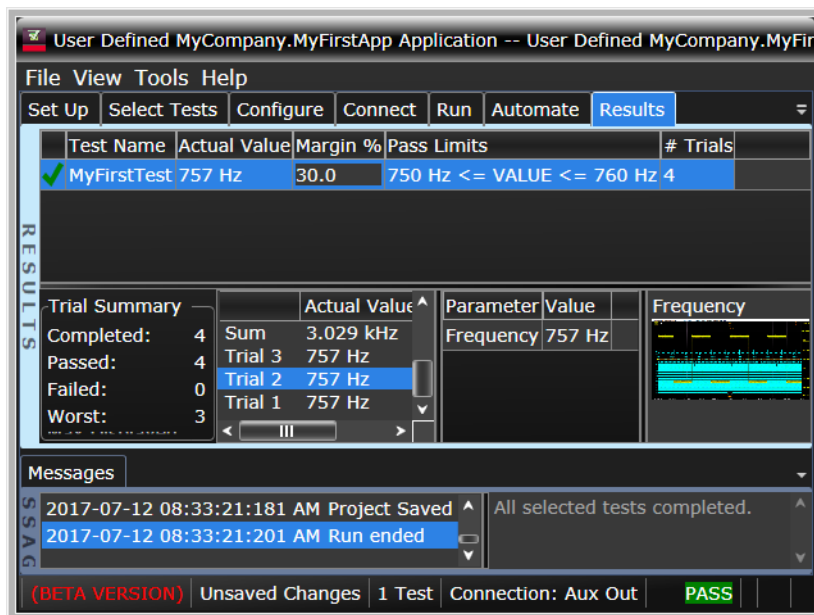
These two limits correspond to the generated app's **View > Preferences > Report > Trial Display** settings.

The default for both of these settings is 25. To change these settings, use options in the application generator's Miscellaneous tab:

- 1 In the User Defined Application generator's **Miscellaneous** tab, first select the maximum number of trial details.
- 2 Next, select the maximum number of those trials that may contain images.



If you generate an application using the example settings shown above, and then run a test four times, you would get results such as this:



Notice that only the worst 3 trials may be viewed in the detail tabs, and only the worst 2 saved their images.

- Next
- **"Step 12: (FlexDCA and Infiniium Only) Enable InfiniiSim"** on page 58
 - **"Step 13: Generate the installer"** on page 59

Step 12: (FlexDCA and Infiniium Only) Enable InfiniiSim

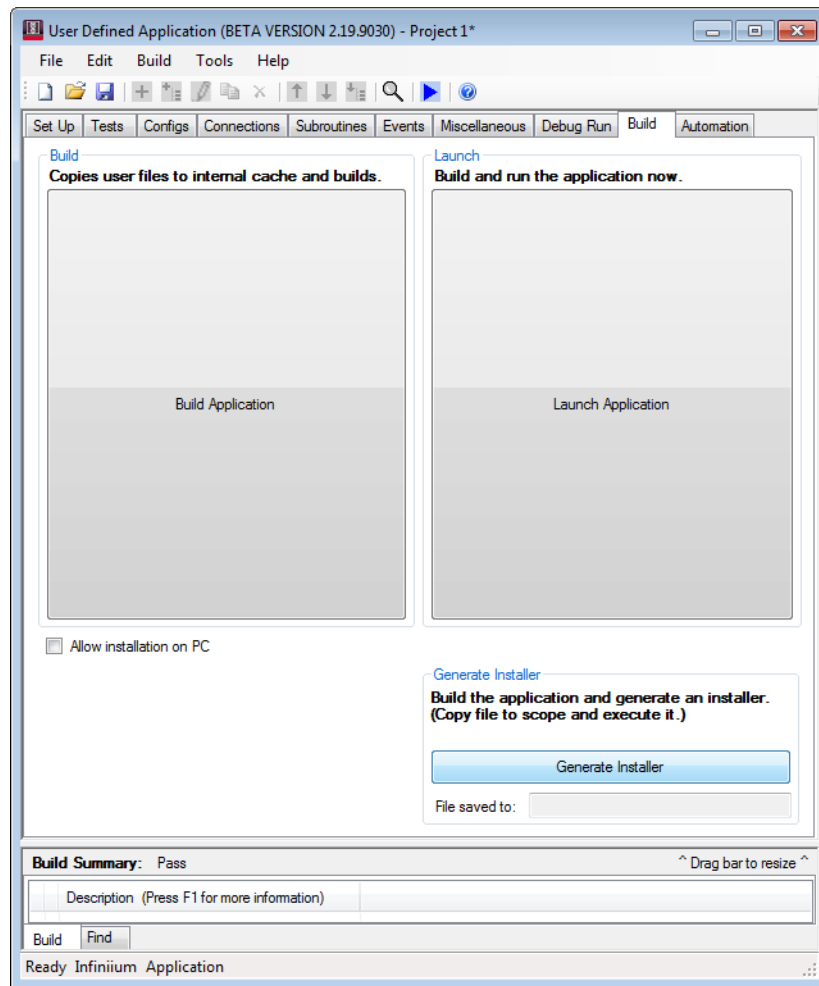
Check the box to enable the Generated App to control the oscilloscope's InfiniiSim features.



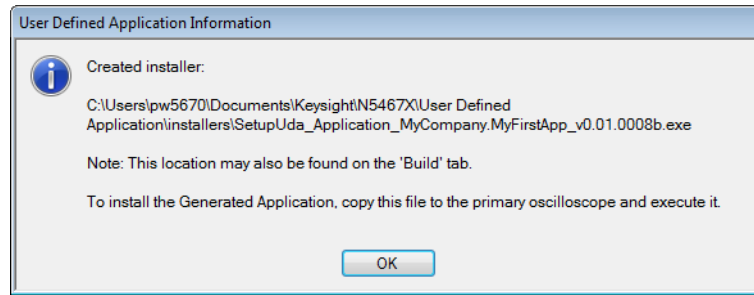
Next · **"Step 13: Generate the installer"** on page 59

Step 13: Generate the installer

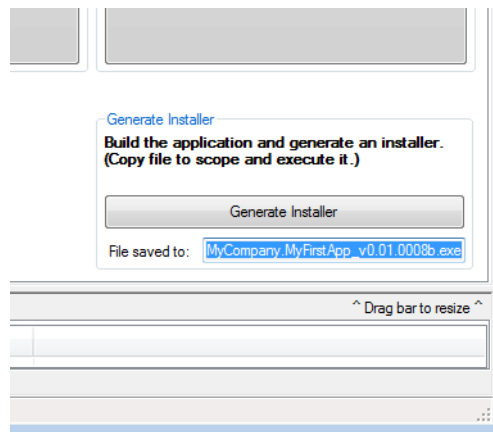
- 1 In the User Defined Application generator's **Build** tab, click **Generate Installer**.



When the install package is generated, an information dialog box appears to show you the location (this location can be specified in the Build tab of the Options dialog box):



The installer location also appears back in the Build tab where you can copy the file name.



Next · **"Step 14a: Installing an application"** on page 61

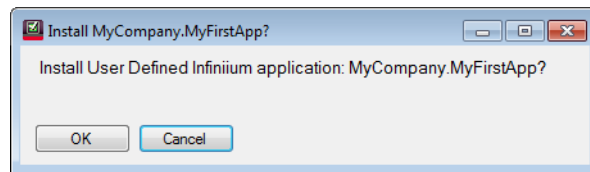
Installing and Running the Application

To run user-defined application, follow these steps:

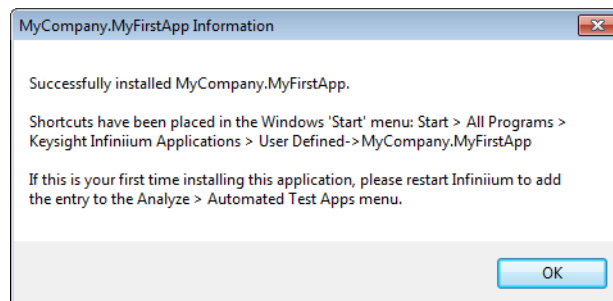
- **"Step 14a: Installing an application"** on page 61
- **"Step 14b: Installing/Uninstalling and Running an Add-In"** on page 61
- **"Step 15: Restart the Infiniium Oscilloscope"** on page 65
- **"Step 16: Launch and use your application"** on page 65

Step 14a: Installing an application

- 1 Copy the generated application installer file to your oscilloscope.
- 2 Run the installer.
- 3 Click **OK** in the confirmation dialog box.



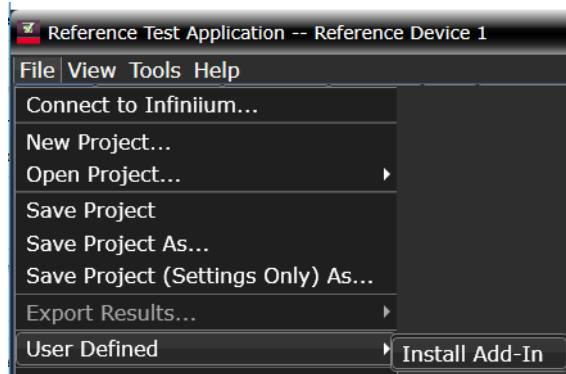
You are notified when the installation completes.



- Next
- **"Step 14b: Installing/Uninstalling and Running an Add-In"** on page 61
 - **"Step 15: Restart the Infiniium Oscilloscope"** on page 65

Step 14b: Installing/Uninstalling and Running an Add-In

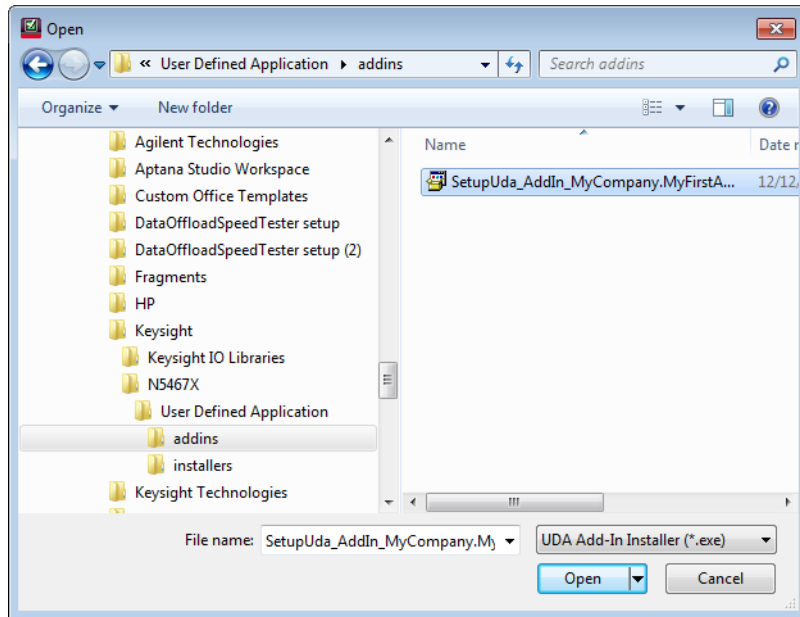
- 1 Copy the add-in installer to your oscilloscope.
- 2 Launch the compliance application you wish to install the add-in into.
- 3 In the compliance application, choose **File > User Defined > Install Add-In**.



NOTE

If this menu option is not present, you will need to update your compliance application to the latest version to use Add-Ins.

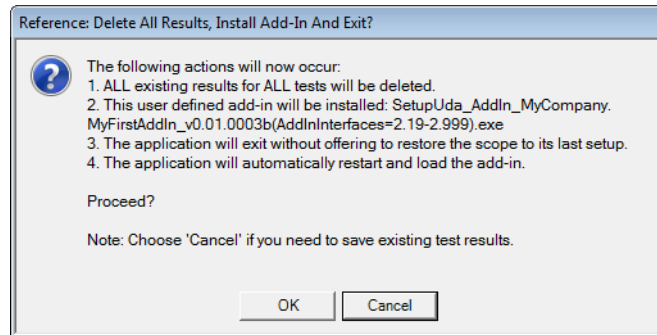
- 4 A file selection dialog box will display. Select the add-in installer and click Open.



- 5 A warning dialog box will display.

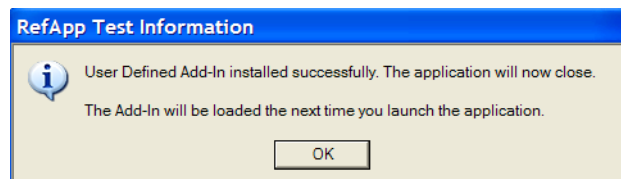
CAUTION

If this application project has unsaved test results, choose **Cancel** and save the project before retrying.



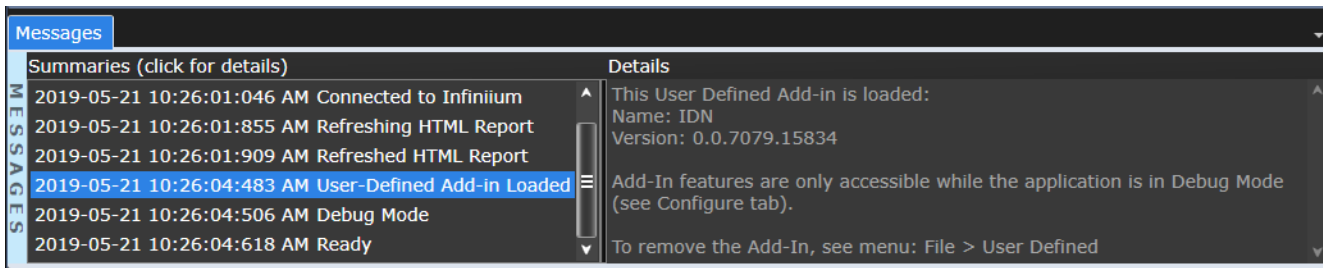
Read the contents of the dialog box carefully and either click **OK** to proceed or **Cancel** to abort the installation.

- 6 On older host applications, you will need to manually restart the application. A reminder dialog box will display.



Click **OK**.

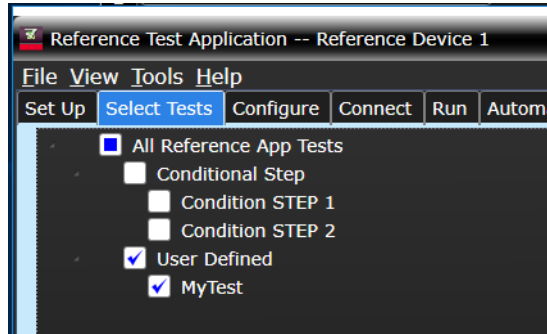
- 7 When the application restarts, reference information appears.



Add-in tests are only visible in Debug Mode so when there is an active add-in installed, the compliance application will start up in this mode.

Click **OK**.

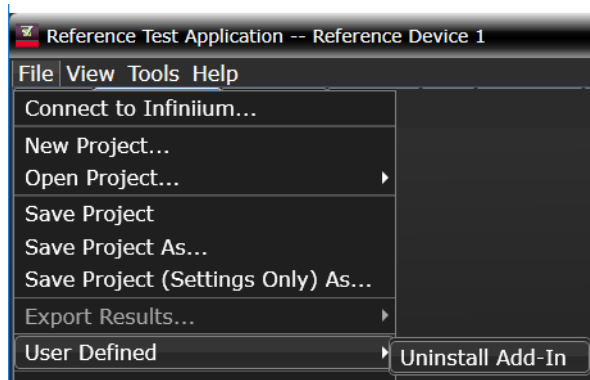
- 8 Add-in tests are displayed in their own test group on the 'Select Tests' tab.



To uninstall a user-defined add-in

Follow these steps:

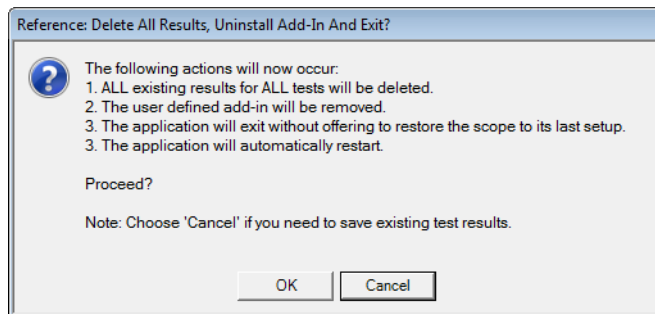
- 1 Launch the compliance application you wish to uninstall the add-in from.
- 2 In the compliance application, click **File > User Defined > Uninstall Add-In...**:



- 3 A warning dialog box will display.

CAUTION

If this application project has unsaved test results, choose Cancel and save the project before retrying.



Read the contents of the dialog box carefully and either click **OK** to proceed or **Cancel** to abort the uninstallation.

Next · **"Step 15: Restart the Infiniium Oscilloscope"** on page 65

Step 15: Restart the Infiniium Oscilloscope

(Infiniium real-time only) After the application is installed on the oscilloscope for the first time, restart the Infiniium oscilloscope software. This adds the application to the Infiniium **Analyze** menu.

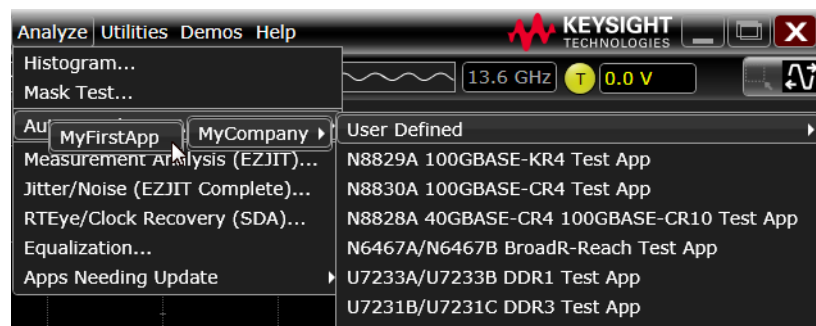
Next · **"Step 16: Launch and use your application"** on page 65

Step 16: Launch and use your application

1 Launch your application:

- Infiniium Real-Time:

After the Infiniium oscilloscope software is restarted, from the Infiniium oscilloscope software main menu, choose **Analyze > Automated Test Apps > User Defined > YourAppName**.

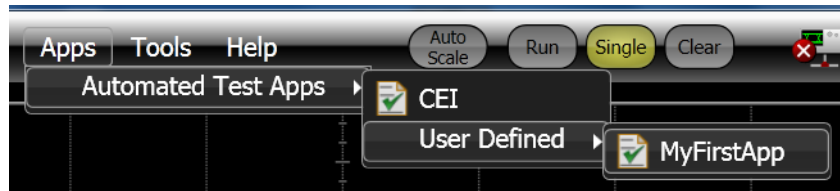


Or, from the Windows Start menu, choose **Start > All Programs > Keysight Infiniium Applications > User Defined > ...**

- FlexDCA:

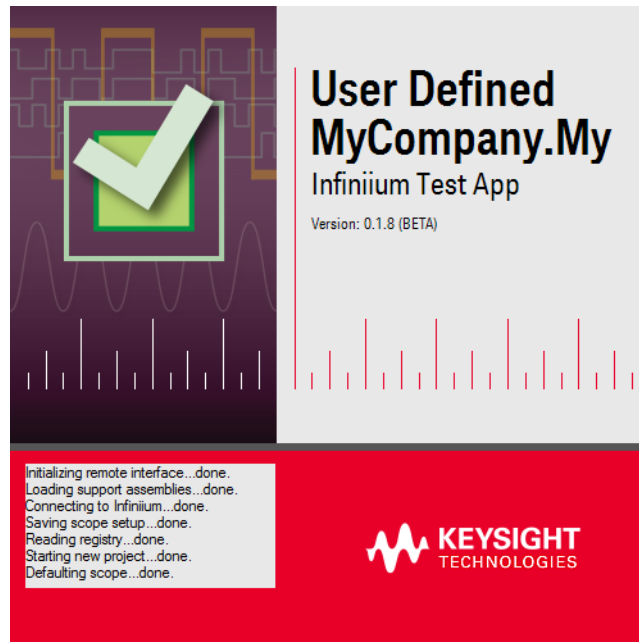
No restart is required for FlexDCA because it dynamically updates its Apps menu every time you click on it.

From the FlexDCA software main menu, choose **Apps > Automated Test Apps > User Defined > YourAppName**.

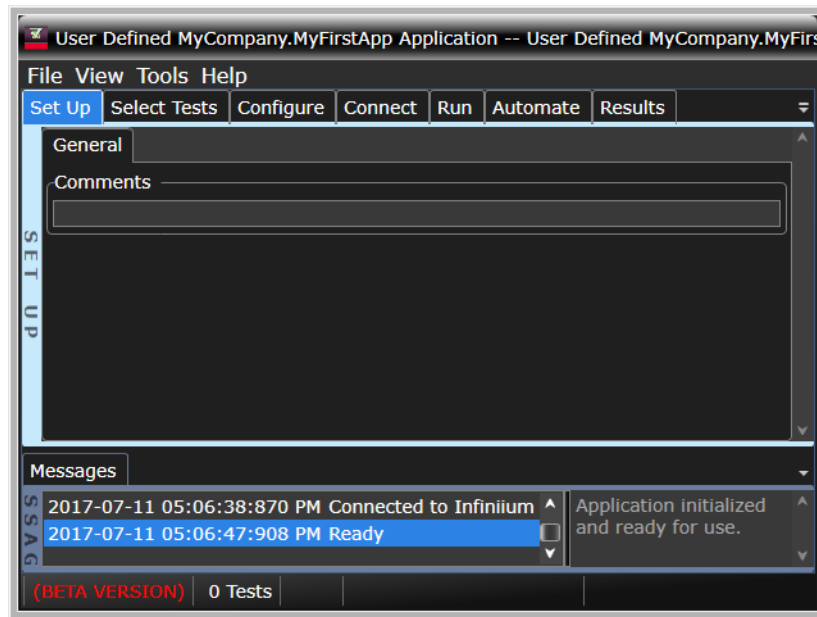


Or, from the Windows Start menu, choose **Start > All Programs > Keysight FlexDCA Applications > User Defined > ...**

- 2 Your application's splash screen appears:



Then its main window appears:



- 3 Generally, you use applications by following the Task Flow pane and the tabs in the application's user interface, that is:
 - a Perform any test set up.
 - b Select the tests you want to run.
 - c Make configuration settings.
 - d Make connections to the device under test (DUT).
 - e Run the selected tests.
 - f View the test results.
 - g View/print the HTML test report.

Remember, the generated application requires the D9010UDAA license to be installed locally to enable running its tests, although you can install and launch it without a license.

For more information on running generated applications, see [Chapter 18](#), “Using Generated Applications,” starting on page 249.

Next · ["For More Information"](#) on page 68

For More Information

The previous getting started steps show you the basics of generating and running applications. For more detailed information, see:


- **Chapter 3**, “How To ...,” starting on page 69
- **Chapter 4**, “Determining Add-In Compatibility,” starting on page 81
- **Chapter 5**, “Setting Up the Application,” starting on page 85
- **Chapter 6**, “Adding/Exporting/Importing Tests,” starting on page 95
- **Chapter 7**, “Adding Config Variables,” starting on page 147
- **Chapter 8**, “Adding Connection Instructions,” starting on page 151
- **Chapter 9**, “Adding Switch Matrix Control,” starting on page 153
- **Chapter 10**, “Adding Run Actions on Events,” starting on page 171
- **Chapter 11**, “Setting Miscellaneous Options,” starting on page 173
- **Chapter 12**, “Setting Debug Run Options,” starting on page 177
- **Chapter 13**, “Generating the Application Software Installer,” starting on page 183
- **Chapter 14**, “Automating the Generated Application,” starting on page 185
- **Chapter 15**, “Working with Variables and Values,” starting on page 189
- **Chapter 16**, “Managing Other Resources,” starting on page 211
- **Chapter 17**, “Saving / Converting / Opening Projects,” starting on page 239
- **Chapter 18**, “Using Generated Applications,” starting on page 249
- **Chapter 19**, “Solving Problems,” starting on page 399
- **Chapter 20**, “Menu/Toolbar/Options Reference,” starting on page 417

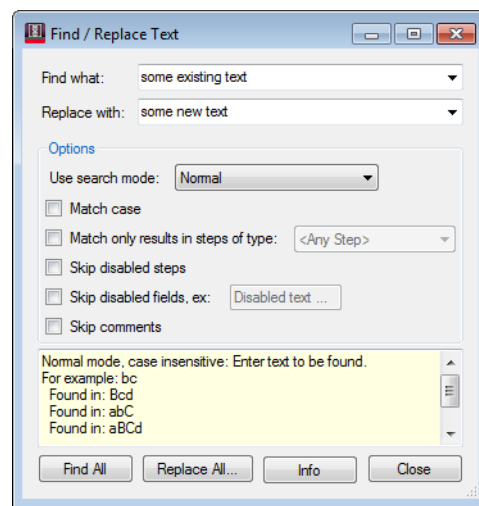
3 How To ...

- Find/Replace Text Strings in a Project / 70
- Change Oscilloscope Settings Based on Waveform Data / 72
- Make an External Measurement and Report the Result / 73
- Run Tests Until a Certain Result Occurs / 74
- Ensure Test 1 Executes Before Test 2 / 75
- Create Custom-Formatted Test Results / 76

Find/Replace Text Strings in a Project

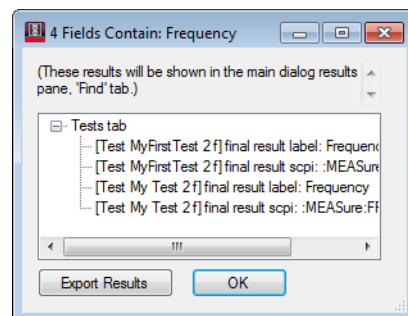
The basic steps are:

- 1 Choose **Edit > Find...** or click the Find  toolbar button to open the Find / Replace Text dialog box.
- 2 Enter the text string to find.
- 3 (Optional) Enter the text string to replace with.
- 4 Select the search mode and options.



There are several search modes to choose from (**Normal**, **RegularExpressions**, and **Wildcards**) and several options you can select. Examples for the selected search mode and options are shown. To see more information about what is searched, click **Info**.

- 5 Click **Find All** or **Replace All...**
- 6 The search results are displayed in a dialog box.



You can export the find results to a text file by clicking on **Export Results**.

To confirm a replace all action, click **Replace All**; otherwise, click **OK** or **Cancel**.

Change Oscilloscope Settings Based on Waveform Data

The basic steps are:

- 1** Save waveform data to a file.
- 2** Use external application to analyze the data, determine the proper setting, and save it to a file.
- 3** Read the file into an internal variable.
- 4** Execute an oscilloscope command, using the variable, to change the setting.

Make an External Measurement and Report the Result

The basic steps are:

- 1 Save waveform data to a file.
- 2 Use external application to make measurement on data and save result to a file.
- 3 Report the result as an intermediate value or final result of a test.

For example, see: "[Using an External Application](#)" on page 108

Run Tests Until a Certain Result Occurs

The basic steps are:

- 1** In the application generator's Events tab:
 - a** On Run Starts: set a variable = some initial condition (query, variable, or constant).
 - b** On Any Trial Starts: set $\text{variable} = \text{variable} + 1$.
- 2** In the application generator's Tests tab:
 - a** Report the intermediate value of variable.
 - b** Final Result: Make some measurement that is dependent upon variable and set the limits so the test fails once you get to the condition you are interested in.
- 3** When running the generated application:
 - a** Select run until Forever.
 - b** Select stop on event Fail.
- 4** When the application stops, the last trial contains the value of variable you are interested in.

Ensure Test 1 Executes Before Test 2

For performance reasons, you may want to design one test to use results already calculated by another test. Therefore, you need to handle the case in which a user runs only the latter test. Here is how you can handle this situation:

- 1** Define an internal variable, for example "CalculationCompleted".
- 2** Put the required calculation steps in a subroutine, for example "DoCalc". At the end of the subroutine, set CalculationCompleted = 1.
- 3** In the "Any Trial Starts" event, set CalculationCompleted = 0.
- 4** Inside Test #1: Call DoCalc.
- 5** Inside Test #2: if CalculationCompleted = 0, then call DoCalc.

Create Custom-Formatted Test Results

The generated app produces an HTML report and also enables you to export results in a CSV format. However, you can also create custom-formatted results using the application's remote interface. Here is an example using the Python scripting language.

Once-Per-Scope Tasks

- 1 Install Python 2.7.x:
 - a Download from <http://www.python.org/getit/>.
 - b Install on the oscilloscope.
- 2 Install Python for .Net:
 - a Download pythonnet-2.0-alpha2-clr2.0_py27 from <http://sourceforge.net/projects/pythonnet/files/>.
 - b Copy files clr.pyd and Python.Runtime.dll found in the zip file to the Python27 DLL folder on the oscilloscope (typically c:\python27\dlls\).
- 3 Enable Python to communicate with Keysight automated test apps:
 - a Copy file Keysight.DigitalTestApps.Framework.Remote.dll to the Python27 DLL folder on the oscilloscope (typically c:\python27\dlls\).

Every app has a version of this file in its installation directory on the oscilloscope. You can find it at:

- Windows XP – C:\Program Files\Keysight\[\Infiniium|FlexDCA]\Apps\User Defined\\
- Windows 7 – C:\Program Files (x86)\Keysight\[\Infiniium|FlexDCA]\Apps\User Defined\\
- Best – The N5452A Remote Toolkit contains a copy of the latest version of the file (works with all apps) in its Tools folder.

App-Specific Tasks

- 1 Create the script (sample provided below).

```
# Import the compiled Python for .Net module
import clr

# Import the Keysight automated test app remote library and utilities
clr.AddReference("Keysight.Infiniium.AppFW.Remote")
from Keysight.Infiniium.AppFW.Remote import *
from xml.dom.minidom import parseString

# Connect to the generated app running on the scope
scopeIpAddress = "localhost"
remoteObj = RemoteAteUtilities.GetRemoteAte(scopeIpAddress)
remoteApp = IRemoteAte(remoteObj)

# Get interesting data
description = remoteApp.GetConfig('UserComment')
scopeInfo = remoteApp.SendScpiQuery('*IDN?')
name = remoteApp.ApplicationName
version = remoteApp.ApplicationVersion
```

```

resultsContainer = remoteApp.GetResults()

# Create the custom results
lines = []
lines.append("""
<CustomOutput>
  <DeviceDescription>""" + str(description) + """</DeviceDescription>
  <ScopeInfo>""" + scopeInfo + """</ScopeInfo>
  <ApplicationName>""" + name + """</ApplicationName>
  <ApplicationVersion>""" + str(version) + """</ApplicationVersion>
  <Results>""")

results = resultsContainer.ExtremeResults
for result in results:
  lines.append("""
    <Result>""")
  lines.append("""
    <TestName>""" + result.TestName + """</TestName>
    <ParameterName>""" + str(result.ParameterName) + """</ParameterName>
    <ActualValue>""" + str(result.ActualValue) + """</ActualValue>
    <TestLimits>""" + str(result.TestLimits) + """</TestLimits>""")
  for i, imageFile in enumerate(result.ImageFiles):
    lines.append("""
      <ImageFile>""" + str(result.ImageFiles[i]) + """</ImageFile>""")
  lines.append("""
    <Passed>""" + str(result.Passed) + """</Passed>
    <TrialNum>""" + str(result.TrialNum) + """</TrialNum>""")
  for extra in result.ExtraData:
    lines.append("""
      <Extra name=" """ + extra.Name + """ " value=" """ + extra.Value + """ " />""")
  lines.append("""
    </Result>""")

lines.append("""
  </Results>
</CustomOutput>""")

fileContents = ''.join(lines)
doc = parseString(fileContents)

# Save results to file
f = open('c:\\temp\\CustomResults.xml', 'w')
f.write(doc.toxml())
f.close()

```

- 2 Integrate the script into the app's script access system (see topic: **Executing Scripts** on page 382).

After executing the sample script above, your custom results will be located in c:\temp\CustomResults.xml.

Execute MATLAB Scripts

In **"Using an External Application"** on page 108, there is a simple example of how to execute a MATLAB script in your generated application. In this topic, you will see how you can make two improvements to that technique.

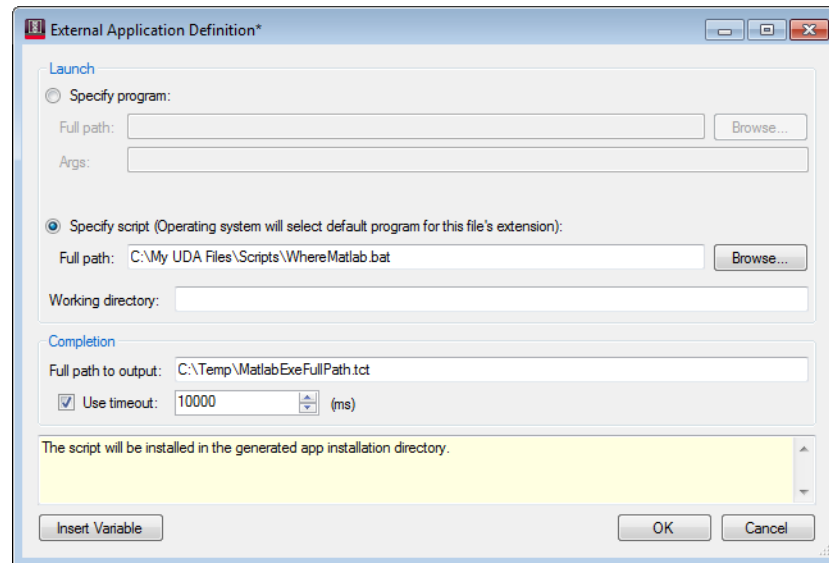
Improvement: Handle different versions of installed MATLAB

If the machines running the generated app may have different versions of MATLAB installed, you should not hardcode the path to matlab.exe. A better method is to dynamically discover the path:

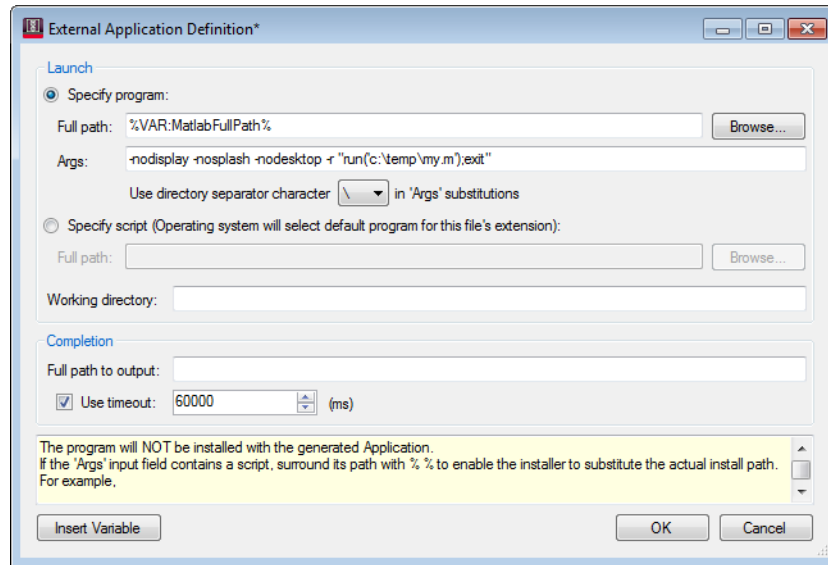
- 1 Create a batch file (for example, named "WhereMatlab.bat") containing one line:

```
where matlab.exe > C:\Temp\MatlabFullPath.txt
```

- 2 Execute this batch file in a Launch External App step using Script mode.



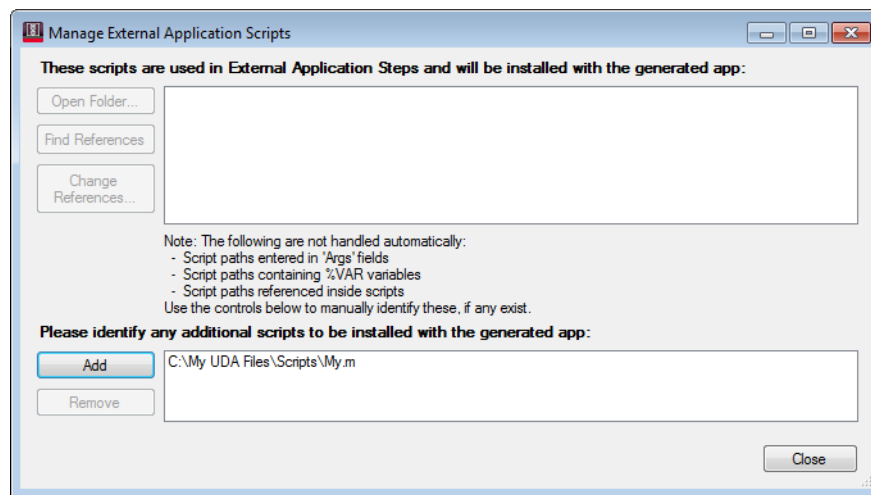
- 3 Use a Set Variable step to copy the contents of the output file (step 2) into a text variable.
- 4 Use this text variable in the Launch External App step that executes your MATLAB script (Program mode, "Full path").



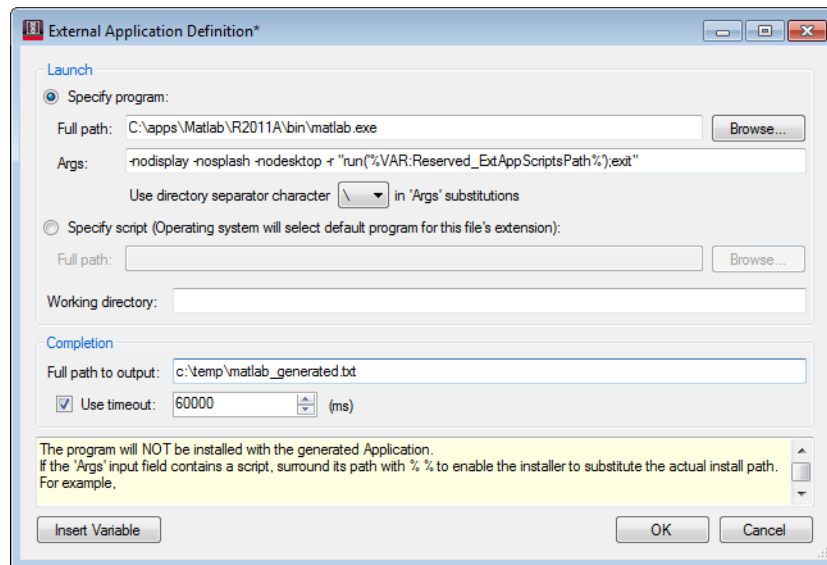
Improvement:
Install your .m file
with the generated
app

The above example requires you to manually place your .m file in the c:\temp\ directory of the machine where the generated app is running. A better method is to have your generated app include the .m file in its own installation:

- 1 Use the **Tools > Manage > External Application Scripts...** menu item to add your MATLAB script:



- 2 In the Launch External App step in which you execute this script, replace the hardcoded path with reserved variable "ExtAppScriptsPath" (Program mode, "Args"):



Use the **Tools > Manage > Reserved Variables** menu item to get a description of the ExtAppScriptsPath reserved variable.

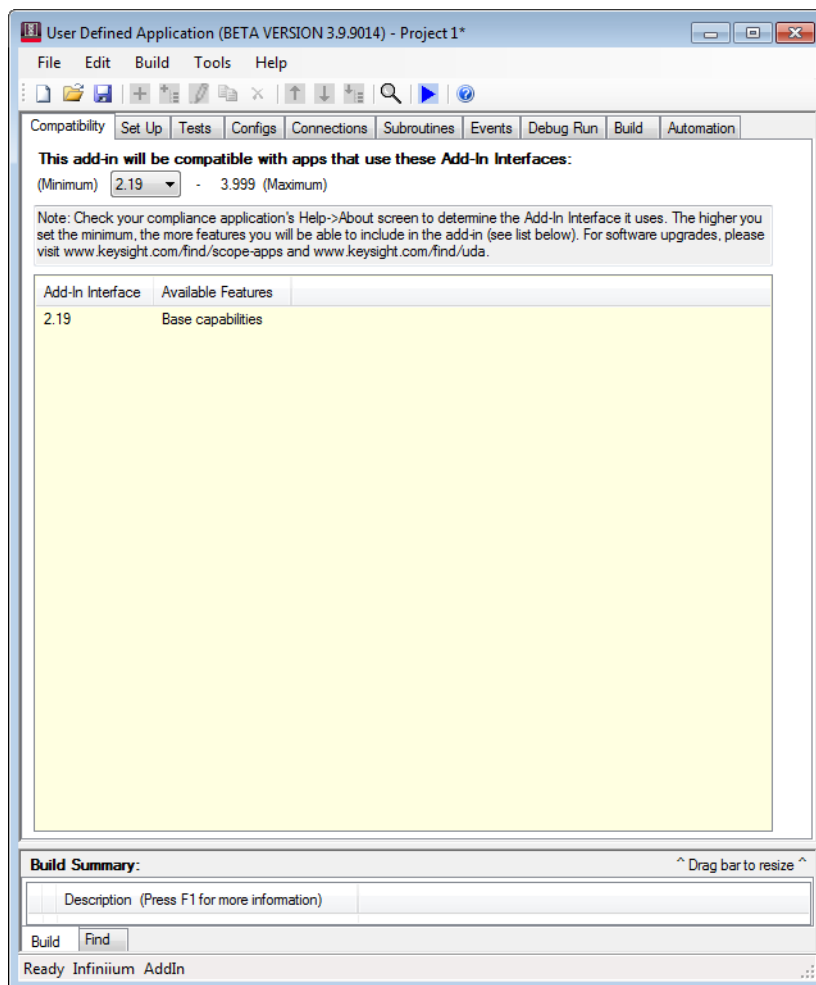
Use MATLAB compiled DLLs

UDA cannot directly access functions defined in arbitrary 3rd party DLLs. However, you can still make use of DLLs with a little extra work:

- 1 Compile a small .exe using the language of your choice. Design this .exe to link to the your DLL and to automatically call the desired function in the DLL. If the DLL's function requires arguments, design the .exe to accept these arguments when it is launched and to pass them on to the DLL function.
- 2 Use a Launch External App step, Program mode, to execute your .exe, passing it any arguments that it needs to forward on to the DLL. Remember that every Launch External App step needs to detect file creation to determine when the step is completed. Therefore, if the DLL's function does not create any files, then your .exe needs to do this.

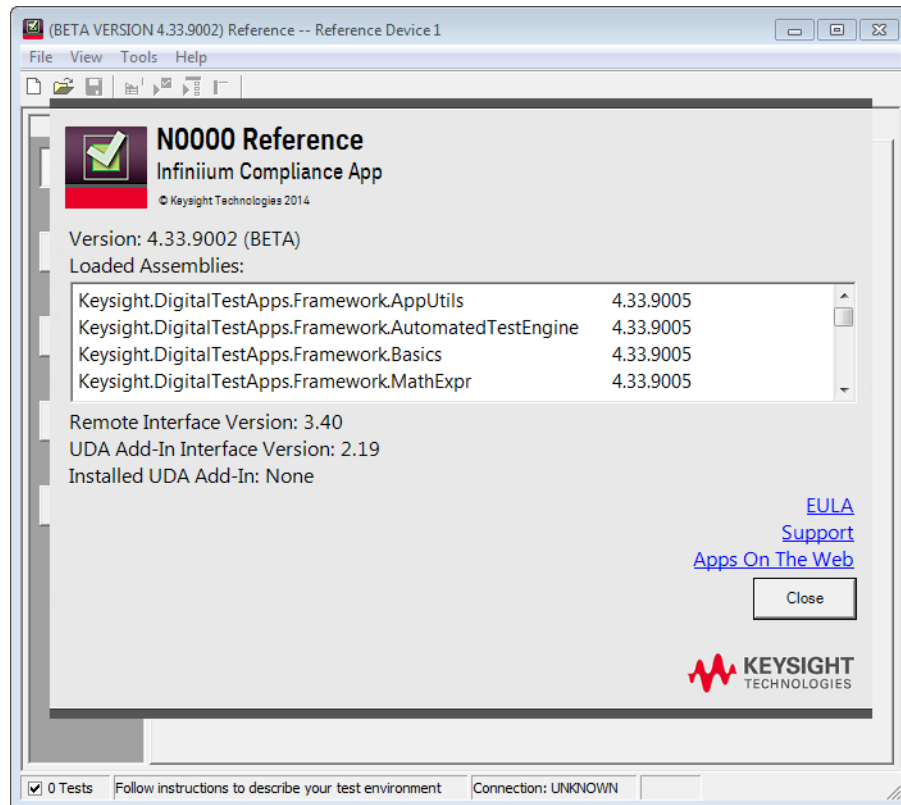
4 Determining Add-In Compatibility

There is a "Compatibility" tab that is only displayed for Add-In projects:



On this tab, you make a tradeoff between maximizing compatibility with compliance applications vs. including more features in your add-in. The top of the tab shows you the range of Add-In versions the current project will be compatible with. To choose a minimum:

- 1 On the oscilloscope or PC, launch the compliance application you are developing the add-in for. Click **Help > About** to see what Add-In interface it currently requires. For example:



- 2 In the Application Generator "Compatibility" tab, select minimum Add-In interface to be less than or equal to the number found in step 1.

NOTE

When you start a new project, the Minimum value is defaulted. You can choose the default value using settings on the **Tools > Options Miscellaneous** tab. However, when you convert an application project to an add-in project, the minimum value is set to the highest level to ensure the project will build.

The table at the bottom of the tab shows you which additional features become available as you increase the minimum Add-In Version number.

CAUTION

If you will be installing the add-in on multiple oscilloscopes, keep in mind that some oscilloscopes may have older versions of the compliance application installed; these versions might require older (lower) Add-In Interfaces.

NOTE

If the compliance app requires an add-in interface that is higher than the maximum shown on the Compatibility tab, you will need to upgrade your D9010UDAA Application Generator to the latest version.

4 Determining Add-In Compatibility

5 Setting Up the Application

Enter Application Name and Version / 86

Specify Required Oscilloscope Software / 87

(Application-Mode Only) Enable/Disable Test Group Filters / 88

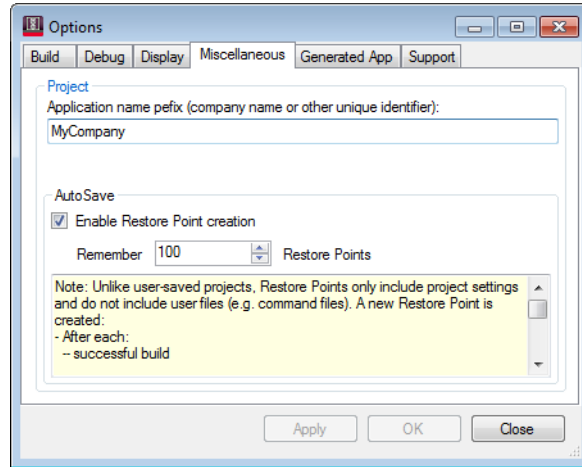
Add External Instruments / 91

Enter Application Name and Version

See "**Step 1: Give your application a name and version number**" on page 27.

To set or change the application name prefix

- 1 From the application generator's main menu, choose **Tools > Options**.
- 2 In the Options dialog box's Miscellaneous tab, enter the **Application name prefix**.



Once the application name prefix is entered, it is saved and applied to all new projects, even after restarting the application generator.

The prefix you enter becomes a subdirectory in the install folder and a submenu in the Infiniium oscilloscope's "Automated Test Apps" menu.

Application name prefixes are optional. If you leave this field empty, the install directory and the launch menu will have just the application name.

You can also use multi-segment prefixes, separated by periods (.), like "MyCompany.MyDivision". In this case, each segment will be a separate folder in the install directory and submenu in the "Automated Test Apps" menu.

- 3 Click **OK**.

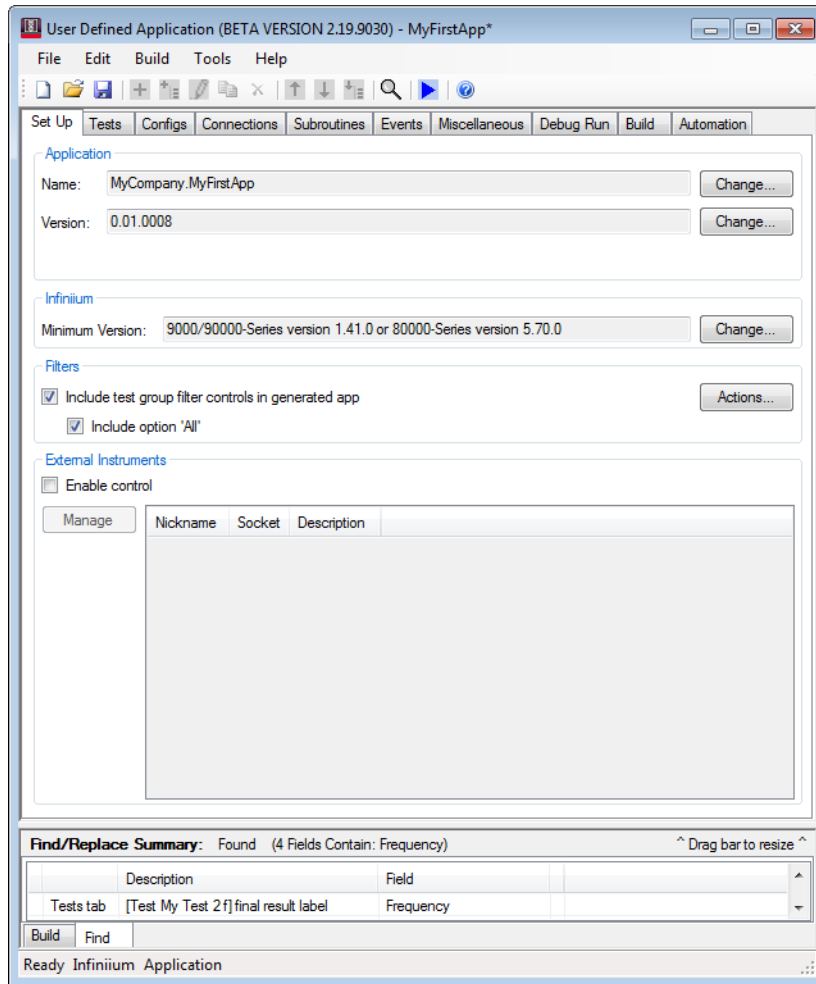
The application name prefix you enter is saved and is used for other new applications you create.

Specify Required Oscilloscope Software

If your application or add-in makes use of an Infiniium or FlexDCA feature that was introduced recently, you can ensure the app checks the version for compatibility. For Infiniium, you can also specify whether the app or add-in should be allowed to run on different generations of oscilloscopes. Click **Change** to specify the minimum required software version.

(Application-Mode Only) Enable/Disable Test Group Filters

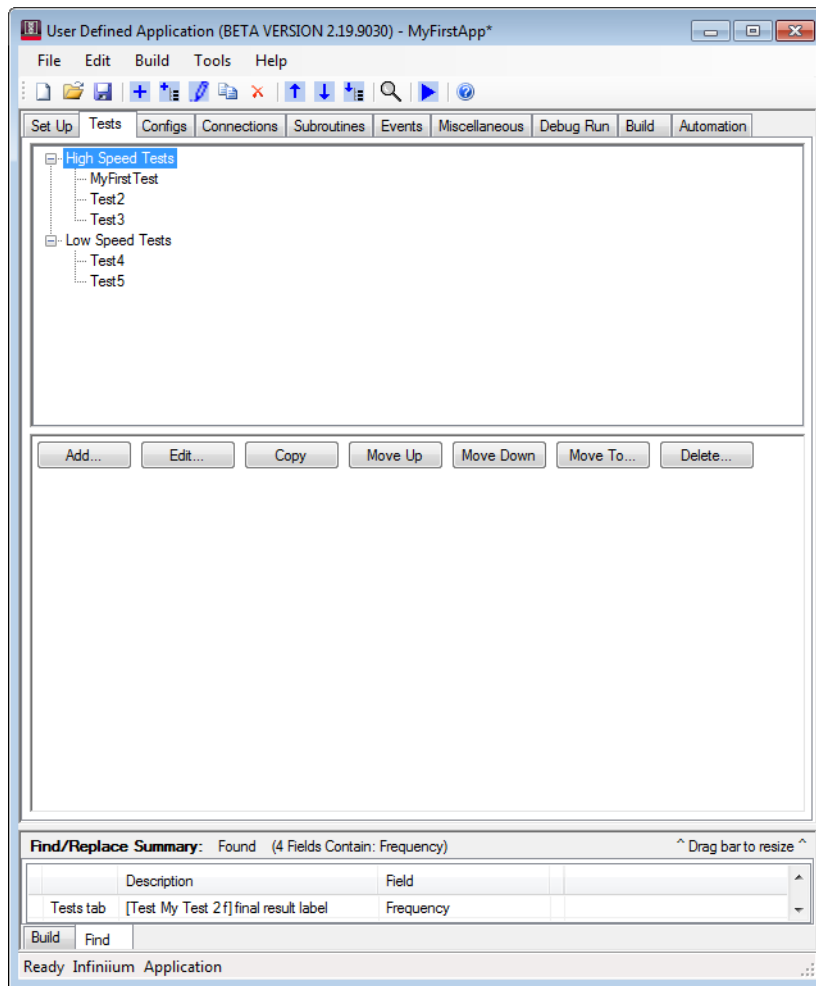
In the application generator's Set Up tab, you can check **Include test group filter controls** to give your application's users the ability to limit the test groups that tests can be selected from.



Check **Include option 'All'** to create a filter that will enable all tests to be selected at the same time.

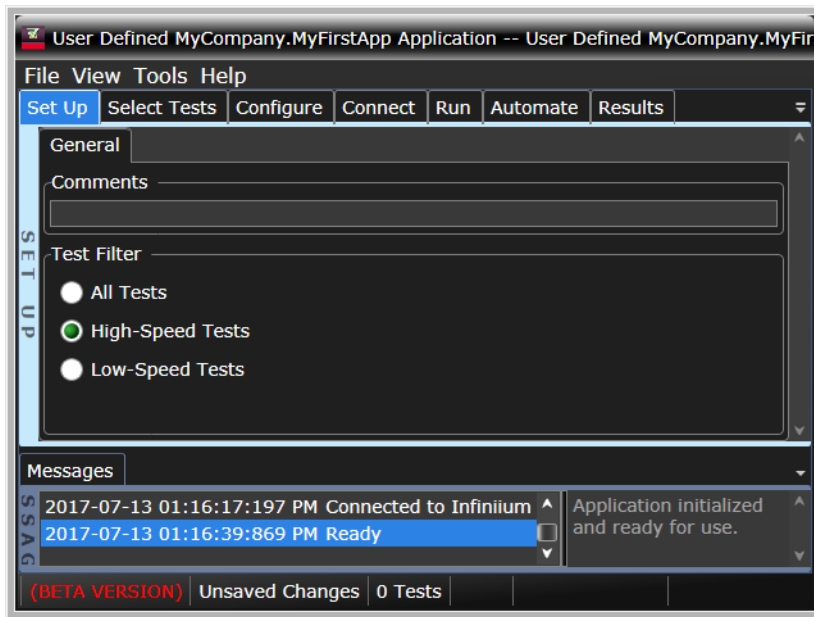
Click **Actions...** to assign Set Variable steps to a filter if you want to set variables when the user selects the filter in the generated app.

For example, if you have test grouped like this:

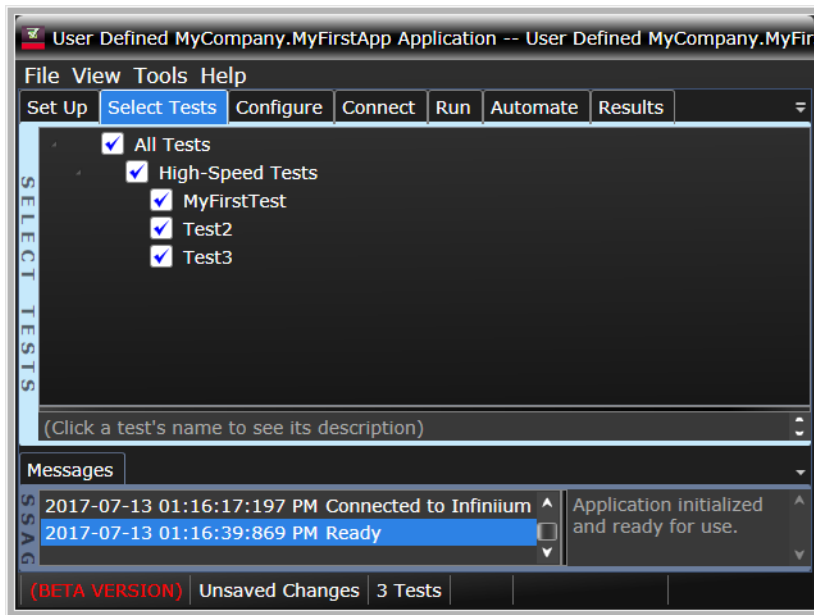


When user's run your application, they will see the **Test filter** options in the Set Up tab.

5 Setting Up the Application



Selecting one of the test group options limits the tests displayed in the application's Select Tests tab.



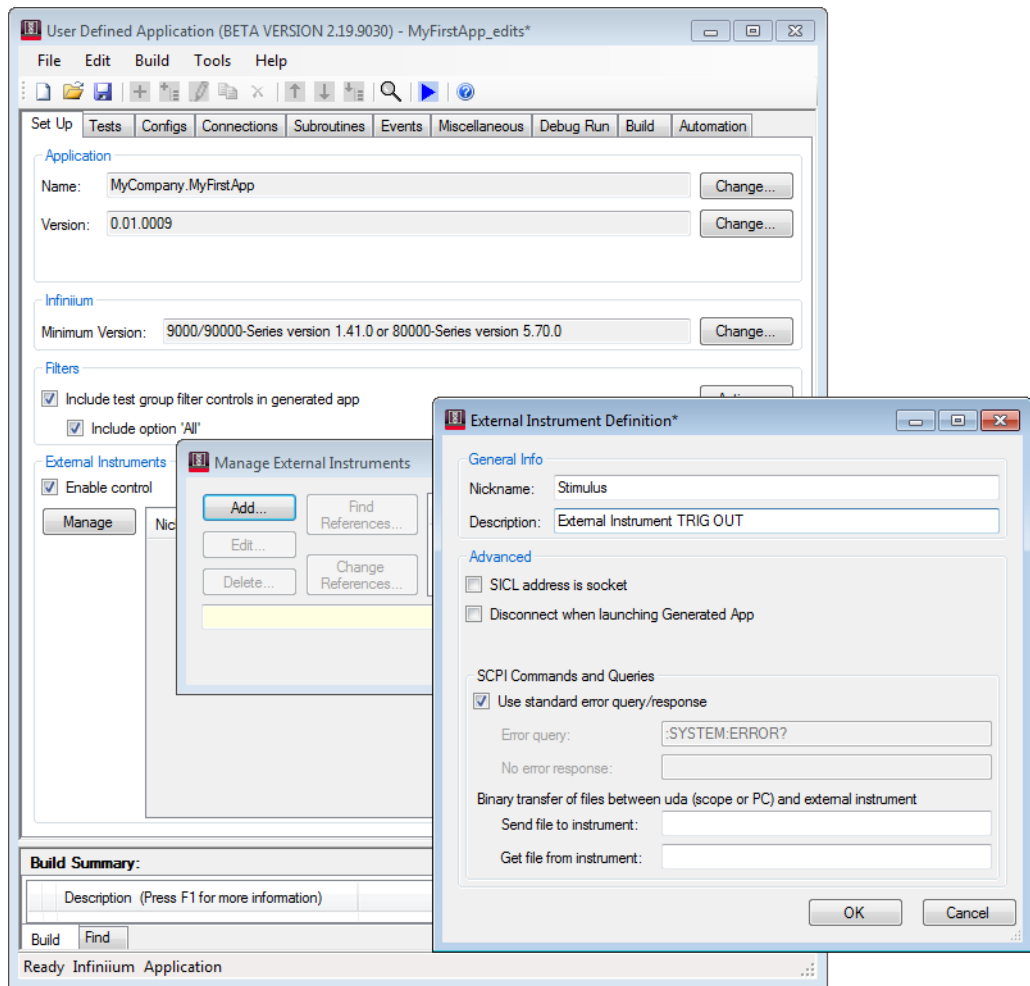
Add External Instruments

In the application generator's Set Up tab, you can add, edit, check usage of, or delete external instruments.

To add external instruments:

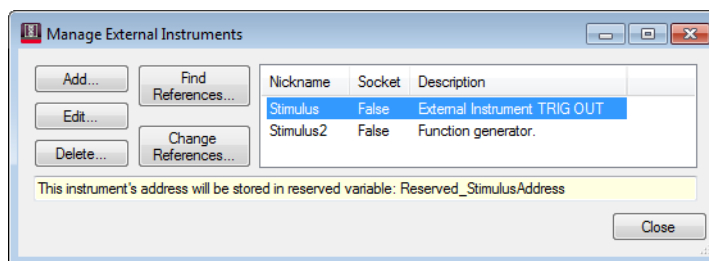
- 1 Check **Enable Control**.
- 2 Click **Manage...**
- 3 Click **Add...** in the Manage External Instruments dialog box.
- 4 In the External Instrument Definition dialog box, enter:
 - Nickname – a short name for the external instrument. (This name may also be used in remote interface programs.)
 - Description – a longer description of the external instrument.
 - SICL - Check this if the instrument's SICL address specifies a socket such as the N4903A J-Bert. You can determine this by looking at the full SICL address for the instrument which you can get using Keysight IO Libraries. If the address is of the form: lan,#[ipaddress], then it is a socket address.
 - (Add-in Mode Only) Exclude – Check this to reuse the host application's external instrument connection (only if both have the same nickname).
 - Use standard – Uncheck this to specify a nonstandard external instrument (not common).
 - Error query – the SCPI query used to check the device for errors, typically ":SYST:ERR?".
 - No error string – the first significant few characters of the error query reply that indicate "no error". For example, if the error query reply is '+0,"No error"', you can enter "+0".
 - If the instrument supports binary file transfers (not common), you may enter the associated SCPI commands. This will enable you to use a "Transfer External File" step in your tests.

5 Setting Up the Application



Then, click **OK**.

After external instruments are added, you can edit them, check their usage (in test subroutines or event steps), or delete them in the application generator's Manage External Instruments dialog box. If more than one instrument has been defined, you may change existing references from one to another using the 'Change References' button. This may be useful when importing fragments into your project.

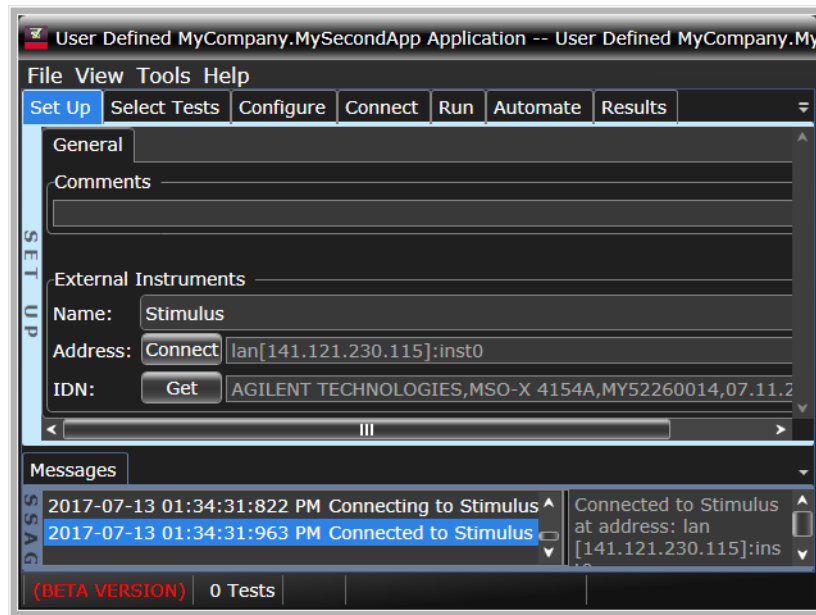


Once you have added external instruments, you can then add test steps to control them. See:

- ["Executing a Single SCPI Command"](#) on page 100
- ["Executing a SCPI Command File"](#) on page 102

For debug runs, you can enter the external instruments' SICL addresses in the Debug Run tab. See ["External Instruments SICL Address Setting/Testing"](#) on page 181.

When running generated applications that use external instruments, the external instruments appear in your application's Set Up tab, where you need to identify and test their SICL addresses.



When running add-ins that use external instruments, the instruments appear as Configs which enable you to enter the SICL address.

5 Setting Up the Application

6 Adding/Exporting/Importing Tests

- Enter Test Name, Description, References / 96
- Remote Interface ID / 97
- Adding Steps / 98
- Refactoring: Extracting Steps to a Subroutine / 133
- Defining Final Result Display Behavior / 135
- Exporting/Importing Tests or Subroutines / 137
- Grouping Tests / 140
- Debug Run Button / 142

Enter Test Name, Description, References

When adding tests, you get a Test Definition dialog box that lets you enter the name, description, and references associated with a test.

When running your application, this information appears in the Select Tests tab and in the HTML Report.

NOTE

You can enter line breaks in the description or references by typing the characters "\n" in those fields.

Remote Interface ID

Every test is given a remote interface ID number. As you add and delete tests, and as you move tests from one test group to other test groups, the remote interface IDs numbers can become varied.

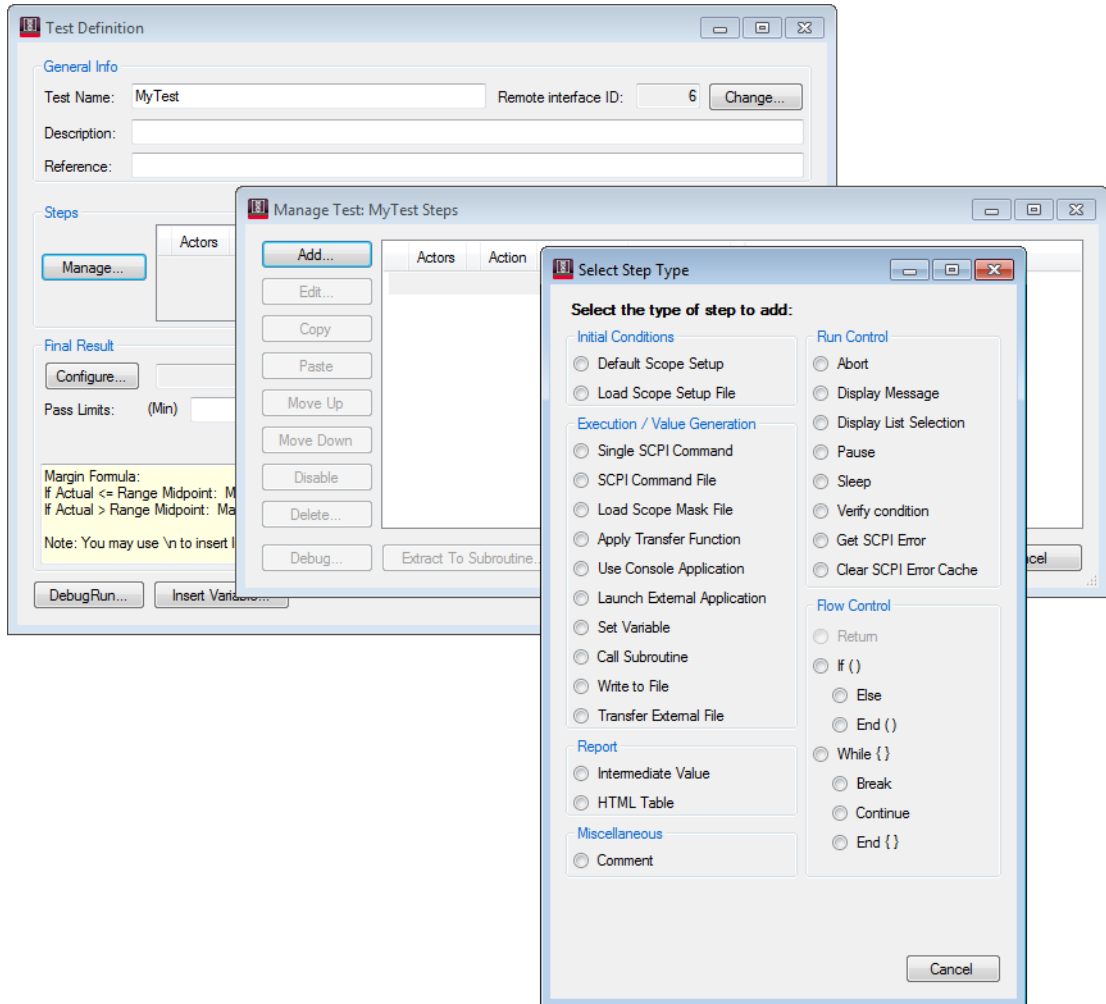
If you would like to change the remote interface ID numbers of your tests before releasing your generated application (perhaps to provide some order), click the **Change...** in the Test Definition dialog box.

However, we recommend that you never change test ID numbers of released apps or add-ins for remote client backward compatibility reasons.

For applications, this ID is a positive number starting at 1. For add-ins, it is a negative number starting at -1001.

Adding Steps

When adding or editing a test in the application generator's Tests tab, you can add multiple steps as part of a test. Once steps are added to the Steps section of the Test Definition dialog box, you can double-click on a step to edit it in a separate dialog box.



In the Manage Test dialog box (accessed by clicking the **Manage** button in the Test Definition dialog box), you can use the buttons listed below. You can also select all of the steps in this dialog box by pressing Ctrl+A on your keyboard.

- **Add...** – Opens the Select Step Type dialog box that lets you add a step to the end of the current list. For more information on the types of steps you can add, see:
 - **"Defaulting the Oscilloscope"** on page 100
 - **"Loading an Oscilloscope Setup File"** on page 100

- "Executing a Single SCPI Command" on page 100
- "Executing a SCPI Command File" on page 102
- "Loading a Scope Mask File" on page 103
- "(Infiniium Only) Applying a Transfer Function" on page 104
- "Using a Console Application" on page 105
- "Using an External Application" on page 108
- "Setting a Variable" on page 111
- "Calling a Subroutine" on page 112
- "Writing to a File" on page 113
- "Transferring Files To/From Instruments" on page 115
- "Aborting" on page 112
- "Displaying Messages" on page 116
- "Displaying List Selection" on page 118
- "Pausing" on page 119
- "Sleeping" on page 121
- "Verifying a Condition" on page 121
- "Manually Managing SCPI Errors" on page 123
- "Reporting an Intermediate Value" on page 124
- "Reporting Tabular Data" on page 125
- "Adding a Comment" on page 126
- "Return Steps" on page 126
- "If, Else, and EndIf Steps" on page 127
- "While, Break, Continue, and EndWhile Steps" on page 129
- **Edit...** – Edits the selected step.
- **Copy** (or Ctrl+c)– Copies the currently-selected steps. (You can Ctrl+click to select multiple steps.)
- **Paste** (or Ctrl+v)– Pastes the copied steps in between test, subroutine, or event test lists.
- **Delete** – Deletes the currently-selected steps.
- **Move Up** – Moves the currently-selected steps up one place in the list.
- **Move Down** – Moves the currently-selected steps down one place in the list.
- **Disable** – Disables the currently selected steps.
- **Debug ...** – Enables you to debug the steps you see in this dialog box.
- **Extract To Subroutine ...** – Enables you to create a new subroutine consisting of the steps currently selected in the list above, and then replaces them with a Call Subroutine step.

Defaulting the Oscilloscope

- 1 In the application generator's Tests tab, either add or edit a test.
- 2 In the Test Definition dialog box's Steps area, click **Manage**
- 3 In the Manage Test dialog box, click **Add**
- 4 In the Select Step Type dialog box, select **Default Scope Setup.**

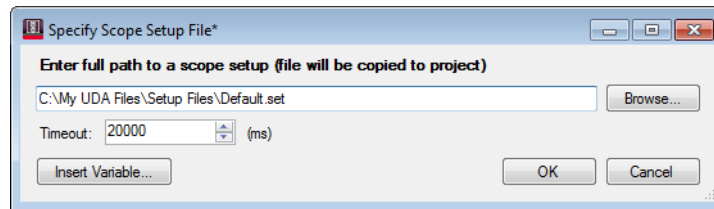
CAUTION

(Infiniium-only) To preserve InfiniiSim settings, be sure to use this step instead of sending a *RST.

Loading an Oscilloscope Setup File

- 1 In the application generator's Tests tab, either add or edit a test.
- 2 In the Test Definition dialog box's Steps area, click **Manage**
- 3 In the Manage Test dialog box, click **Add**
- 4 In the Select Step Type dialog box, select **Load Scope Setup File.**
- 5 In the Specify Scope Setup File dialog box, enter the full path of the Infiniium .set or FlexDCA .setx setup file to be loaded. Type ALT+I to insert a variable reference.

If the file is accessible from your development computer, it will be copied into the installer. Otherwise, you should ensure the file is available when the application is run.

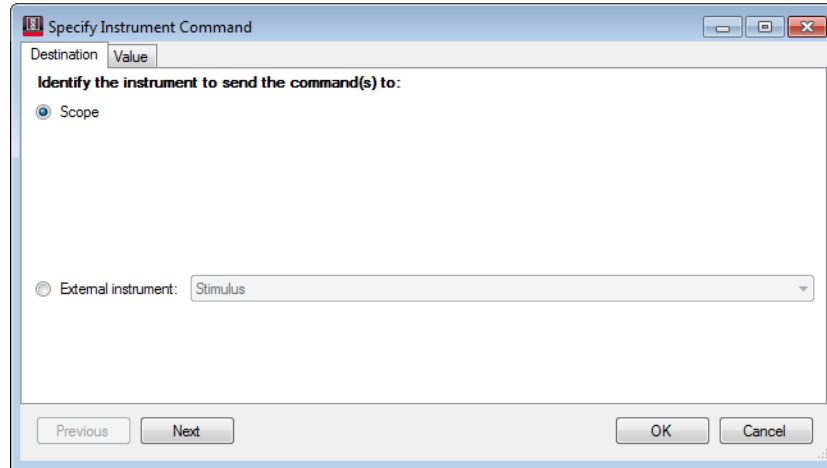


- 6 In the **Timeout** field, specify the maximum allowed time to execute the step.
- 7 Click **OK.**

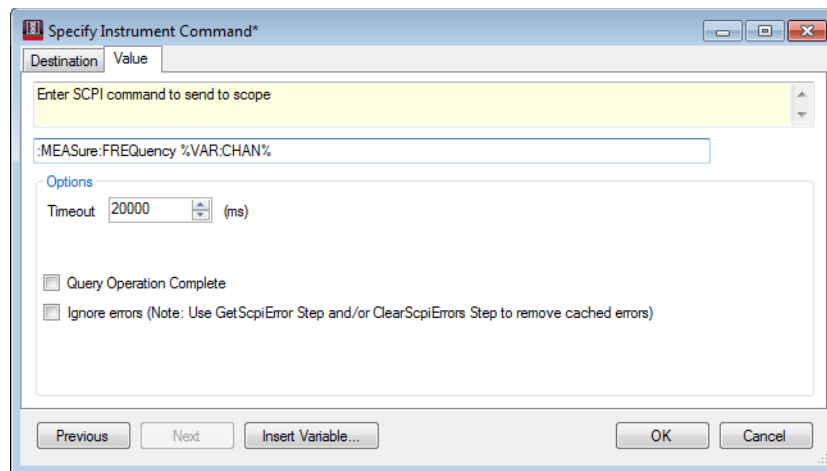
Executing a Single SCPI Command

- 1 In the application generator's Tests tab, either add or edit a test.
- 2 In the Test Definition dialog box's Steps area, click **Manage....**
- 3 In the Manage Test dialog box, click **Add...**
- 4 In the Select Step Type dialog box, select **Single SCPI Command.**

- 5 In the Specify Command dialog box, if external instruments are defined, select either the oscilloscope or the instrument as the destination of the command.



- 6 In the Specify Command dialog box, select the Value tab and enter a single SCPI command. Type ALT+I to insert a variable reference.



NOTE

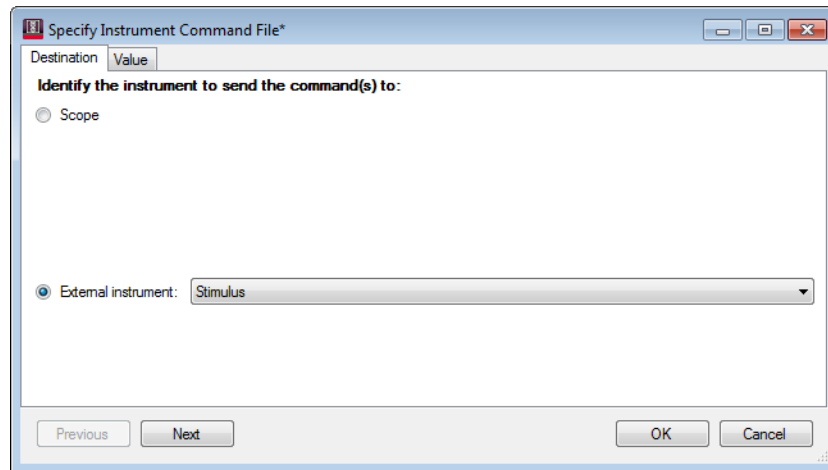
Enter only SCPI commands, not SCPI queries, because SCPI query results are discarded by this step type.

- 7 Change the **Timeout** value if desired.
- 8 If the command you enter requires checking the "operation complete" flag on the oscilloscope (not common), you can check **Query Operation Complete** to ensure the next step does not begin until the oscilloscope has completed executing this step.

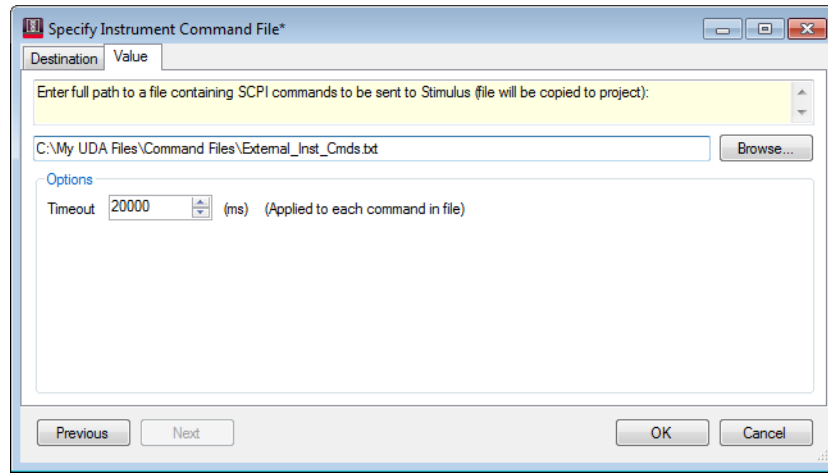
- 9 If the command you enter executes successfully but still generates a SCPI error, you can check **Ignore** and use Get SCPI Error steps to manually manage the error queue. See "**Manually Managing SCPI Errors**" on page 123.
- 10 Click **OK**.

Executing a SCPI Command File

- 1 In the application generator's Tests tab, either add or edit a test.
- 2 In the Test Definition dialog box's Steps area, click **Manage...**
- 3 In the Manage Test dialog box, click **Add...**
- 4 In the Select Step Type dialog box, select **SCPI Command File**.
- 5 In the Specify Command File dialog box, if external instruments are defined, select either the oscilloscope or the instrument as the destination of the commands.



- 6 In the Specify Command File dialog box, select the Value tab and browse to select the path of a text file containing one or more SCPI commands.
The file must be accessible on your development computer because it will be copied into the installer.

**NOTE**

'If' and 'While' statements are not supported in Command Files (these files may only contain SCPI commands and comments).

7 Click **OK**.

Loading a Scope Mask File

This step causes the oscilloscope to load the specified mask file and enables the mask test settings on the oscilloscope.

If you are using a 90000A or 9000A Series oscilloscope with Infiniium software versions 2.10 through 4.10, you can use the same two options available for setup files - file path located on the oscilloscope or a network path (See: "[Oscilloscope SICL Address and Setup/Mask/Transfer Function File Paths](#)" on page 178).

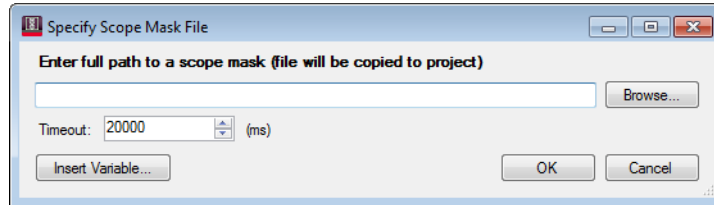
Ensure that the oscilloscope's directory where you place the mask file is shared (for example, as UDA Mask Test, for example). Then, the scope mask file path would be the full path to the share on the oscilloscope. In the Debug Run tab, enter the local path on the oscilloscope to the directory where you put the mask file. This step will cause the mask file to be saved with the UDA project and installed by the generated app's installer to the generated app's install location (not Infiniium's mask directory).

To load a scope mask file:

- 1 In the application generator's Tests tab, either add or edit a test.
- 2 In the Test Definition dialog box's Steps area, click **Manage...**
- 3 In the Manage Test dialog box, click **Add...**
- 4 In the Select Step Type dialog box, select **Load Scope Mask File**.

- 5 In the Specify Scope Mask File dialog box:
 - a Enter the full path to the Infiniium.msk or FlexDCA .mskx mask file to be loaded. Type ALT+I to insert a variable reference.

If the file is accessible from your development computer, it will be copied into the installer. Otherwise, you should ensure the file is available when the application is run.



- b In the **Timeout** field, specify the maximum allowed time to execute the step.
- 6 Click **OK**.

(Infiniium Only) Applying a Transfer Function

Use this step type to apply an InfiniiSim transfer function to an oscilloscope channel.

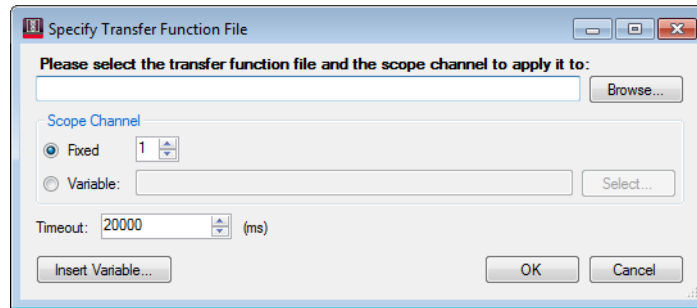
NOTE

This step will fail unless InfiniiSim is turned on for that channel.

To execute a Transfer Function step in a debug run

- 1 In the application generator's Tests tab, either add or edit a test.
- 2 In the Test Definition dialog box's Steps area, click **Manage...**
- 3 In the Manage Test dialog box, click **Add...**
- 4 In the Select Step Type dialog box, select **Apply Transfer Function**.
- 5 In the Specify Transfer Function File dialog box:
 - a Enter the full path to the desired transfer function file. Type ALT+I to insert a variable reference.

If the file is accessible from your development computer, it will be copied into the installer. Otherwise, you should ensure the file is available when the application is run.



To apply a transfer function:

- 1 Identify the location of the file containing the transfer function.
- 2 Identify the oscilloscope channel to apply it to:
 - Fixed: Select the number of the channel to apply the transfer function to.
 - Variable: Select an internal variable or config that will contain the number of the channel.
- 3 In the **Timeout** field, specify the maximum allowed time to execute the step.

Using a Console Application

A console application is typically software that controls the device under test (DUT) somehow and is interactive; for example, you launch the console application at the beginning of a test, then at different times during the test you send the console application commands that change DUT parameters. There are different step actions for launching a console application and for sending it a command.

TIP

For console applications developed using C++, if you use `printf` to write to the console then use `fflush(stdout)` to enable UDA to intercept the message. If you use `std::cout`, the `std::endl` will automatically accomplish the required flush.

Console applications must be available to the development PC (when testing and building your application) and will be copied to the oscilloscope by your application's installer.

Before you can launch or send a command to a console application, you must add the console application to the project (see **"Managing Console Applications"** on page 218).

To launch a console application:

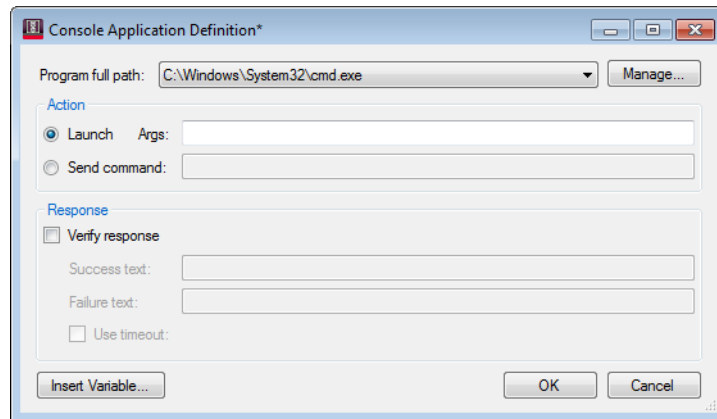
- 1 In the application generator's Tests tab, either add or edit a test.
- 2 In the Test Definition dialog box's Steps area, click **Manage...**
- 3 In the Manage Test dialog box, click **Add...**

- 4 In the Select Step Type dialog box, select **Use Console Application**.
- 5 In the Console Application Definition dialog box:
 - a Select the console application.
 - b In the Action box, select **Launch** and enter any arguments for the launch command.
 - c If you would like to verify the response of the program launch, check **Verify response**; then, enter the **Success text** and/or the **Failure text**.

Verifying the response is useful for making sure the console application is ready before continuing on with the next step.

The generated application will check console application responses until either the failure text or the success text is found.

- d If you would like to use a timeout when verifying the response, check **Use timeout**; then, enter the timeout value.



- 6 Click **OK**.

To send a command to a console application:

- 1 In the application generator's Tests tab, either add or edit a test.
- 2 In the Test Definition dialog box's Steps area, click **Manage...**
- 3 In the Manage Test dialog box, click **Add...**
- 4 In the Select Step Type dialog box, select **Use Console Application**.
- 5 In the Console Application Definition dialog box:
 - a Select the console application.
 - b In the Action box, select **Send command** and enter the command to send to the console application.
 - c If you would like to verify the command response, check **Verify response**; then, enter the **Success text** and/or the **Failure text**.

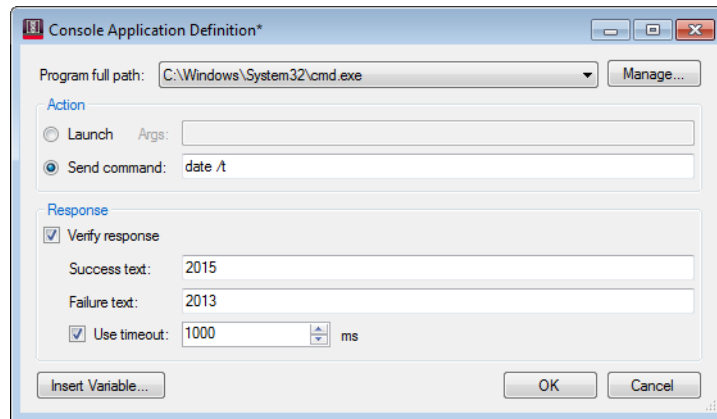
Verifying the response is useful for making sure the console application is ready before continuing on with the next step.

The generated application will check console application responses until either the failure text or the success text is found.

- d If you would like to use a timeout when verifying the response, check **Use timeout**; then, enter the timeout value.

NOTE

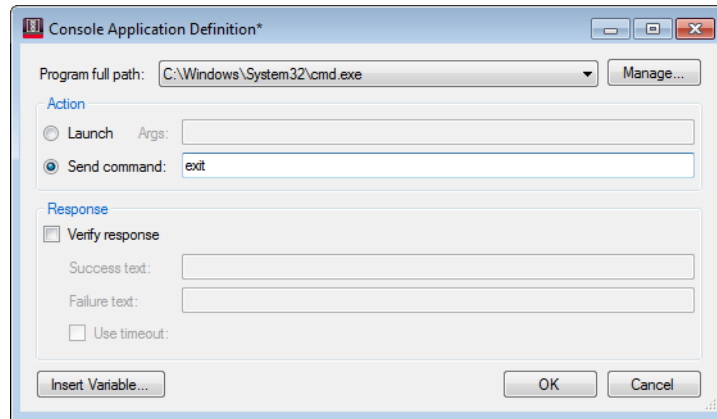
While your cursor is in a text field, you can type ALT+I to insert a variable reference.



- 6 Click **OK**.

To exit a console application:

- 1 In the application generator's Tests tab, either add or edit a test.
- 2 In the Test Definition dialog box's Steps area, click **Manage...**
- 3 In the Manage Test dialog box, click **Add...**
- 4 In the Select Step Type dialog box, select **Use Console Application**.
- 5 In the Console Application Definition dialog box:
 - a Select the console application.
 - b In the Action box, select **Send command** and enter the exit command.
 - c If you would like to verify the command response, check **Verify response**; then, enter the **Success text** and/or the **Failure text**.
 - d If you would like to use a timeout when verifying the response, check **Use timeout**; then, enter the timeout value.



- 6 Click **OK**.

Using an External Application

External applications are typically used to provide some feature or data post-processing capability that is not built-in to the oscilloscope. External applications typically output results (or intermediate results) for a test. In this use model, external application steps are typically used in conjunction with other types of steps. For example, you might:

- 1 Use **Single SCPI Command** or **SCPI Command File** steps to save acquired oscilloscope data, as with this SCPI command:

```
:DISK:SAVE:WAVEform %VAR:CHAN%, "C:\temp\waveform.csv", CSV, OFF
```

For more information on these types of steps, see ["Executing a Single SCPI Command"](#) on page 100 or ["Executing a SCPI Command File"](#) on page 102.

- 2 Use a **Launch External Application** step to process the saved data and output a result. (For more information on defining an external application, see the description that follows.)
- 3 Use an **Intermediate Value** step or a **Result** configuration to report the output of the external application, getting the result from a file source and perhaps including an image file. For more information, see ["Reporting an Intermediate Value"](#) on page 124 or ["Defining Final Result Display Behavior"](#) on page 135 and ["To get a value from a File source"](#) on page 206.

To use an external application:

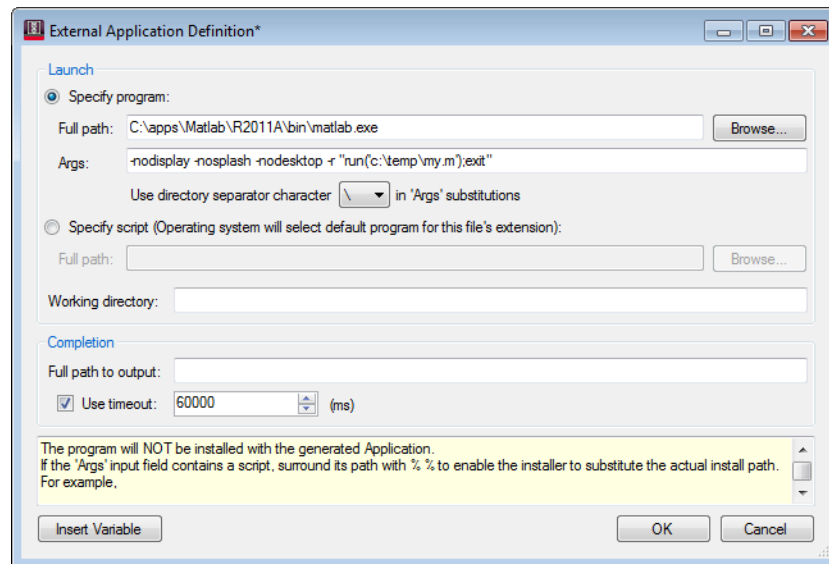
- 1 In the application generator's Tests tab, either add or edit a test.
- 2 In the Test Definition dialog box's Steps area, click **Manage...**
- 3 In the Manage Test dialog box, click **Add...**
- 4 In the Select Step Type dialog box, select **Launch External Application**.

5 In the External Application Definition dialog box:

a Choose the **Launch** method:

There are two ways to launch the external application:

- **Specify program** – In this case, you enter the name of an executable program file on the oscilloscope, and you use command arguments to pass information to the executable program. This method is used when running MATLAB scripts as illustrated in the following picture.



External Application program files are not copied and installed with the generated application. They must already exist on the oscilloscope where the generated application will be run.

Enter the **Full path** to the executable command on the oscilloscope.

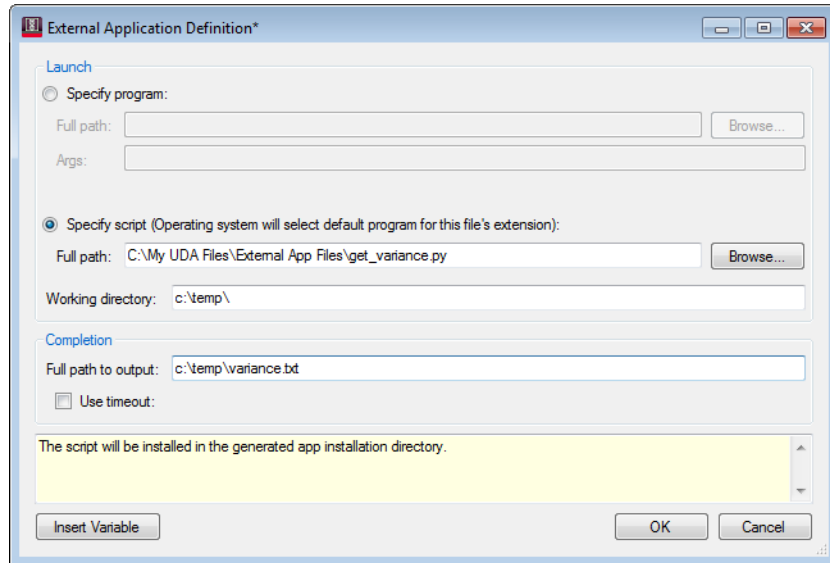
Enter command arguments in the **Args** field. Arguments can contain the names of input files or paths (as is the case with MATLAB scripts); in this case, surround paths with percent (%) characters so that the generated application installer can substitute the path with the actual install path.

CAUTION

Files included in External Application program arguments are not automatically copied to the application installer. To include these files with the application installer, you must add them to the list of additional scripts to be installed with the generated application. See "**Managing External Application Scripts**" on page 221.

The **Use directory separator character** selection lets you specify the type of directory separator character used in path substitutions (for paths enclosed by % %). For example, MATLAB path arguments use forward slashes (/). Other executable programs may use back slashes (\).

- **Specify script** – In this case, you enter the name of a file whose type's "open" association specifies the application to run. For example, a file with the ".py" extension can specify that the Python interpreter be run.



Script files are always copied and installed with the user defined application.

Note that you can use %VAR variables in the script file's **Full path** filename as a way of choosing from a number of scripts depending on a variable's setting.

- b** In the **Working directory** field, enter the working directory to be passed to the external application.
- c** Whether external applications are launched by specifying programs or scripts, their **Completion** is determined by the presence of an output file.

In the **Full path to output** field, enter the full path to the output file on the oscilloscope.

The User Defined Application expects external applications to produce text or image files containing the results. Enter the path on the oscilloscope where the generated app can expect to find this file.

If the external application generates multiple files (for example, an image file and a text file), be sure to enter the name of the last file created; otherwise, test steps may continue before the external application is completed, possibly looking for an output file that has not yet been created.

- d If you would like to use a timeout when verifying the output, check **Use timeout**; then, enter the timeout value.

NOTE

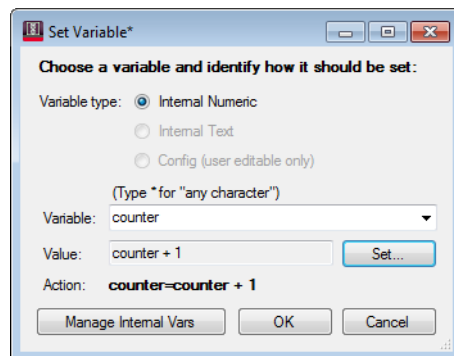
While your cursor is in a text field, you can type ALT+I to insert a variable reference.

- 6 Click **OK**.

Setting a Variable

You may set internal variables and user-editable configs in a test step. Internal variables can be used like Config variables; however, they are not visible to users of the generated application. Internal variables can only hold numeric values. User-editable configs are those configured to allow users to add choices.

- 1 In the application generator's Tests tab, either add or edit a test.
- 2 In the Test Definition dialog box's Steps area, click **Manage...**
- 3 In the Manage Test dialog box, click **Add...**
- 4 In the Select Step Type dialog box, select **Set Variable**.
- 5 In the Set Variables dialog box, select the type of variable to edit. You can set internal variables by clicking the **Internal** radio button and you can set configs by clicking the **Config (user editable only)** radio button.
- 6 If you need to add or remove an internal variable, click **Manage Internal Vars ...**. See "[Managing Internal Variables](#)" on page 190.
- 7 Click **Set** to set the variable value. See "[Getting Values](#)" on page 198.



The **Variable:** selection box provides filtering so you can just start typing characters (including * asterisks for wildcards) to quickly find existing variables. The text you type does not have to match the case of the variable you are looking for, and if you end your text with a space then only variables ending with your last letter will be matched. For example, "myvar " matches "MyVar" but not "MyVar2". This works in any combo box that lists variable names.

- 8 Click **OK**.

TIP

You may use a "Write to File" step to store text that you need to use later. For example, in a test or subroutine:

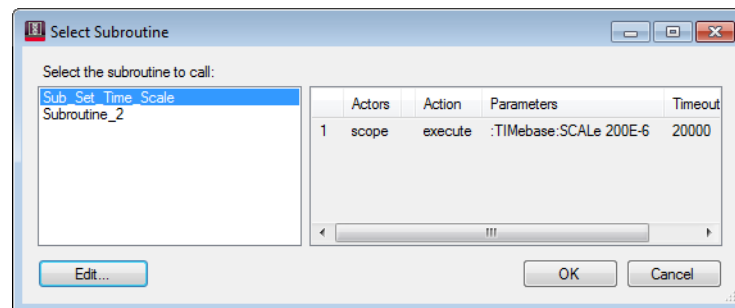
Step n: (Write to File step) Write "enabled" to a file.

Step n+5: (Intermediate Value step) report contents of the file you wrote to.

Calling a Subroutine

Use this step to call a subroutine from within a test, event, or other subroutine. In order to use this step, you must have previously defined (or imported) subroutines using the Subroutine tab located in the main dialog box.

- 1 In the application generator's Tests tab, either add or edit a test.
- 2 In the Test Definition dialog box's Steps area, click **Manage...**
- 3 In the Manage Test dialog box, click **Add...**
- 4 In the Select Step Type dialog box, select **Call Subroutine**.
- 5 In the Select Subroutine dialog box, select the subroutine to call.



- 6 (Optional) Click **Edit** to modify the subroutine.
- 7 Click **OK**.

Aborting

Use this step to stop the current test from completing and, optionally, the current run from continuing.

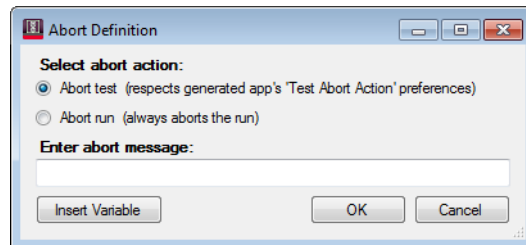
If you select **Abort Test**, then when the generated app executes the step, the test-in-progress will abort. If the generated app's "Test Abort Action" (see **View > Preferences > Run**) is set to **Continue**, the next selected test will execute. Otherwise, the entire run will be aborted.

If you select **Abort Run**, then when the generated app executes the step, the entire run will be aborted regardless of the "Test Abort Action" setting.

NOTE

If you include an **Abort Step** in a **Run Action (Event)** step list, then regardless of your selection, the step will always execute as an "Abort run" since Run Actions (Events) execute in between tests.

- 1 In the application generator's Tests tab, either add or edit a test.
- 2 In the Test Definition dialog box's Steps area, click **Manage...**
- 3 In the Manage Test dialog box, click **Add...**
- 4 In the Select Step Type dialog box, select **Abort**.
- 5 In the Abort Definition dialog box, either select **Abort Test** or **Abort Run** and specify the message to be displayed. At any time, type ALT+I to insert a variable reference.



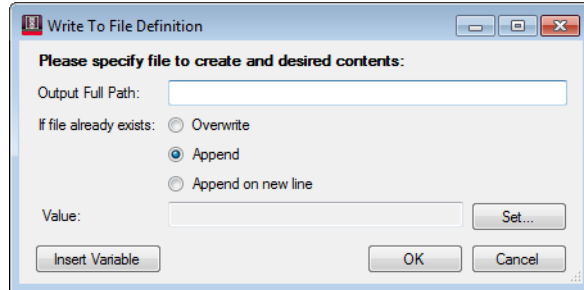
- 6 Click **OK**.

Writing to a File

Use this step to write to a file.

- 1 In the application generator Tests tab, either add or edit a test.
- 2 In the Test Definition dialog box's Steps area, click **Manage...**
- 3 In the Manage Test dialog box, click **Add...**
- 4 In the Select Step Type dialog box, select **Write to File**.

- 5 In the Write to File Definition dialog box:
 - a Set the parameters:



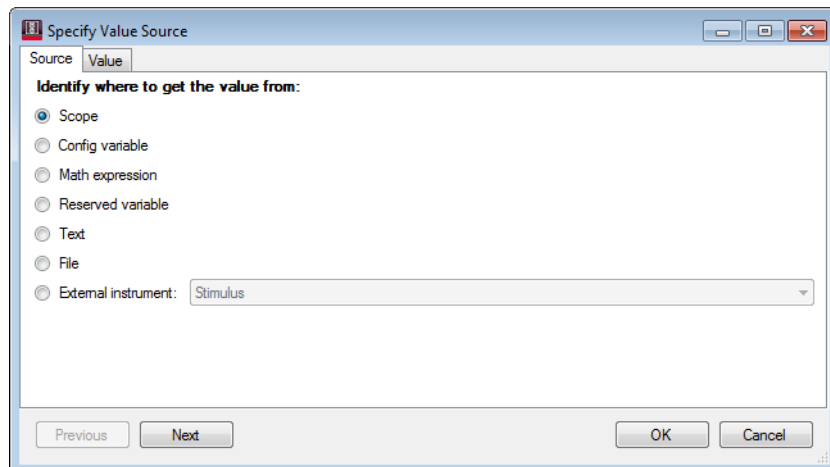
Output Full Path Specifies the file to be created or modified when the step executes. At any time, type ALT+I to insert a variable reference.

Overwrite If the file specified in the Output Full Path field already exists, it will be deleted and replaced.

Append If the file specified in the Output Full Path field already exists, it will be appended to. Else a new file will be created.

Append On New Line Same as Append except the new information will be added starting on a new line in the file. Also, if the text being written contains "\n" characters, then new lines will be started.

Value Defines what will be written to the file. Selectable from the following dialog box (see "[Getting Values](#)" on page 198):



- 6 Click **OK**.

Transferring Files To/From Instruments

Using a Transfer step, you can copy files to or from the scope application (FlexDCA or Infiniium) and to or from external instruments defined on the Set Up tab. For example, after sending a command to a VNA to take a screen shot, you can use a Transfer step to copy the image to the scope.

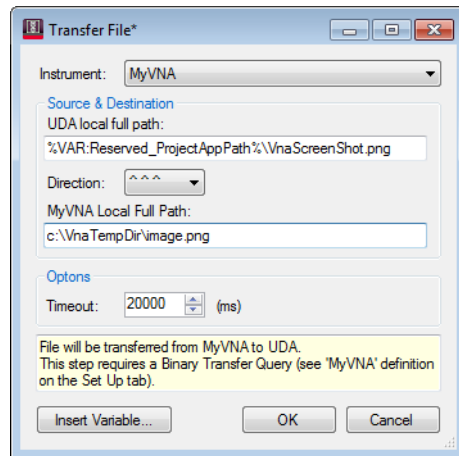
NOTE

For transfers involving external instruments, you must first enter the Binary Transfer command and query for the external instrument (see its programming guide) on the Set Up tab.

Select the **Instrument** from the list.

To send a file to the instrument:

- 1 Enter the **UDA Local Full Path** on the UDA machine (scope or PC) where the file will be found. Type ALT+I to insert a variable.
- 2 Select **Direction** ^ ^ ^.
- 3 Enter the **Instrument Local Full Path** on the instrument where the file should be created. Type ALT+I to insert a variable.



- 4 Click **OK**.

To get a file from the instrument:

- 1 Enter the **UDA Local Full Path** on the UDA machine (scope or PC) where the file should be created. Type ALT+I to insert a variable.
- 2 Select **Direction** v v v.
- 3 Enter the **Instrument Local Full Path** on the instrument where the file will be found. Type ALT+I to insert a variable.
- 4 Click **OK**.

Displaying Messages

Using the Display Message step, you can display messages with response choices such as "Ok/Cancel", "Yes/No", etc. and access the user's response in a variable.

If you select the **Tell** option, a message box is displayed with only an **OK** button.

If you select the **Ask** option, a message box is displayed with specific alternatives.

The **Buttons** drop-down menu lets you choose which type of buttons will be shown with the message.

The **Default** drop-down menu lets you choose which button will execute if the user just presses Enter on the keyboard when the message is first displayed.

The **Suppression Response** drop-down menu lets you choose which button will execute if a remote programming client executes this step while the remote interface property "SuppressMessages" is set to "True".

If you check **Remote Suppression Autoclick**, the drop-down menu lets you choose which button will execute if a remote programming client executes this step while the remote interface property "SuppressMessages" is set to "True". If this box is unchecked, the test will abort when a remote user executes it while suppressing messages.

You can use the internal variable "Reserved_AskResponse" in the predicate term of an If or While step to access the user's response to the most recent "Ask" message.

If you select the **Ask For Data** option, a message box is displayed prompting the user to enter a value.

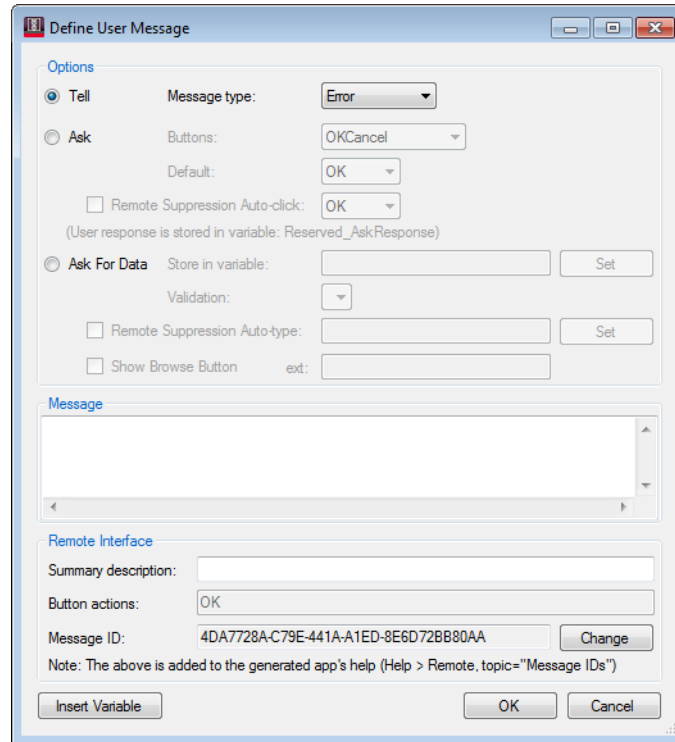
The **Store in variable Set** button lets you choose a variable to store what the user types.

The **Validation** drop-down menu, enabled if you chose an internal store variable, lets you choose between requiring the user to enter a whole number or allowing the user to enter any number.

If you check **Remote Suppression Autotype**, the **Set** button lets you choose a default value to use if a remote programming client executes this step while the remote interface property "SuppressMessages" is set to "True". If this box is unchecked, the store variable will be left unmodified when a remote user executes the test while suppressing messages.

- 1 In the application generator Tests tab, either add or edit a test.
- 2 In the Test Definition dialog box's Steps area, click **Manage...**
- 3 In the Manage Test dialog box, click **Add...**
- 4 In the Select Step Type dialog box, select **Display Message**.

- 5 In the Define User Message dialog box:
 - a If you choose **Tell**, select the **Message type**:



Info, **Warning**, and **Error** messages can be filtered by users of the generated application (see "[To set the display preferences](#)" on page 290).

Internal Errors cannot be filtered by users.

Event messages are not displayed. They appear in the event log only and can be used for troubleshooting problems with the generated application.

- b Enter the message to be displayed. At any time, type ALT+I to insert a variable reference.
- c If you choose **Ask**, select the buttons you want displayed with the message (**Buttons**), the button that is defaulted as being highlighted when the message appears (**Default**), and the button that should be executed when the SuppressMessages property in the remote interface is set to "True" (**Remote Suppression Autoclick**). Then enter the message to be displayed.
- d If you choose **Ask For Data**, select the variable to store the user's entry (**Store in variable**), the numeric rule to use (**Validation**), and the value that should be entered when the SuppressMessages property in the remote interface is set to "True" (**Remote Suppression Autotype**). If you want the prompt to contain a

file browse button, select **Show Browse Button**. Then, enter the message to be displayed.

- e Optionally, to help remote client developers manage the prompt, add information to be included in the generated app's help system:
 - **Summary Description** – A short summary of the purpose of the prompt.
 - **Button actions** – When using **Ask** or **Ask for Data**, enter a short summary of what will happen when each button is clicked. For example: OK = Value is used, Cancel = Test is aborted.
 - **Message ID** – Automatically generated value that the generated app's remote clients can use to uniquely identify this prompt so they will know which button to click.

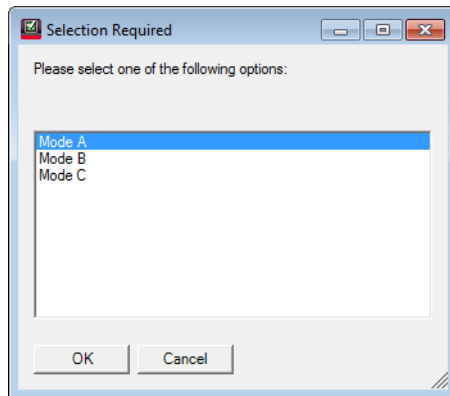
Later, if you edit this prompt and make minor changes to the message, leave the ID unchanged so remote clients will be unaffected.

However, if you change which buttons will be shown, or the basic meaning of the message, you should click **Change** to get a prompt for a new ID to force remote clients to notice the change. Also, if you copy an existing DisplayMessage step, click **Change** to give the copy a unique ID.

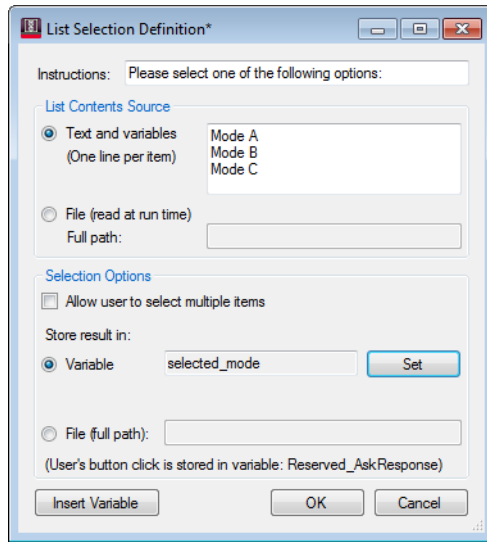
6 Click **OK**.

Displaying List Selection

Using the Display List Selection step, you can prompt the user to select one or more items from a list you define. At runtime, when the generated app executes this step, it will display a list selection prompt. For example:



- 1 In the application generator Tests tab, either add or edit a test.
- 2 In the Test Definition dialog box's Steps area, click **Manage...**
- 3 In the Manage Test dialog box, click **Add...**
- 4 In the Select Step Type dialog box, select **Display List Selection**.



Instructions tell the generated app's user what to do with the list.

If you choose **Text and variables**, you will be able to enter the items you want to appear in the list. Each line will result in one list entry in the generated app. You may enter text or insert variables. For example:

```
Option 1
%var:MyVariable%
Option %var:MyOtherVariable%
```

If you choose **File (read at runtime)**, then when the step executes the generated will create the list from the contents of the file you specify. Each line in the file will result in one list entry.

Check **Allow** if you want the user to be able to select multiple items from the list; otherwise they will only able to select one.

If you choose **Variable**, then the user's selection will be stored in the variable you select. Click the **Set** button to see available variables.

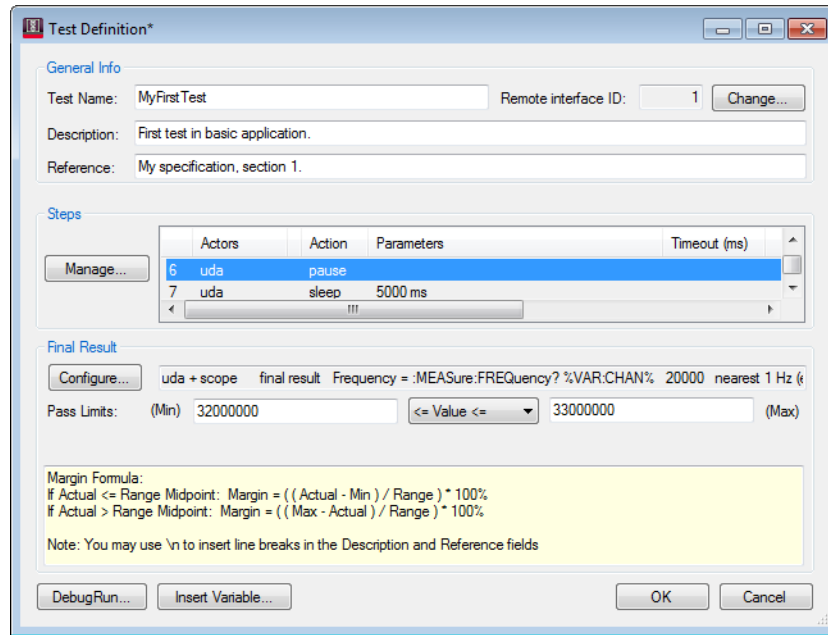
If you choose **File (full path)**, then the user's selection will be stored in a file created when the step executes.

The **Insert Variable...** button reminds you that you can use ALT+I to insert a variable whenever your cursor is in any of the text boxes.

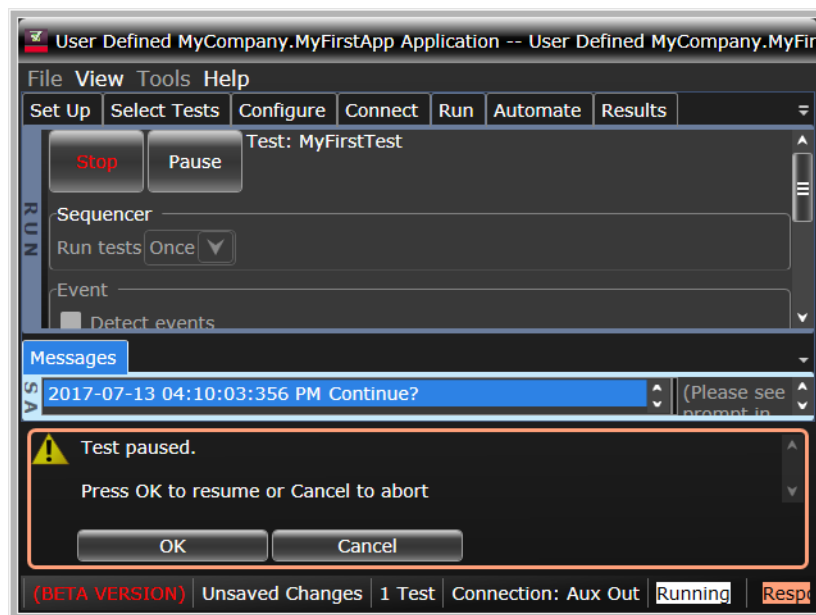
Pausing

- 1 In the application generator's Tests tab, either add or edit a test.
- 2 In the Test Definition dialog box's Steps area, click **Manage...**
- 3 In the Manage Test dialog box, click **Add...**
- 4 In the Select Step Type dialog box, select **Pause**.

The pause step is added to the Test Definition dialog box.



When running tests, a pause causes a "Continue?" dialog box to appear.



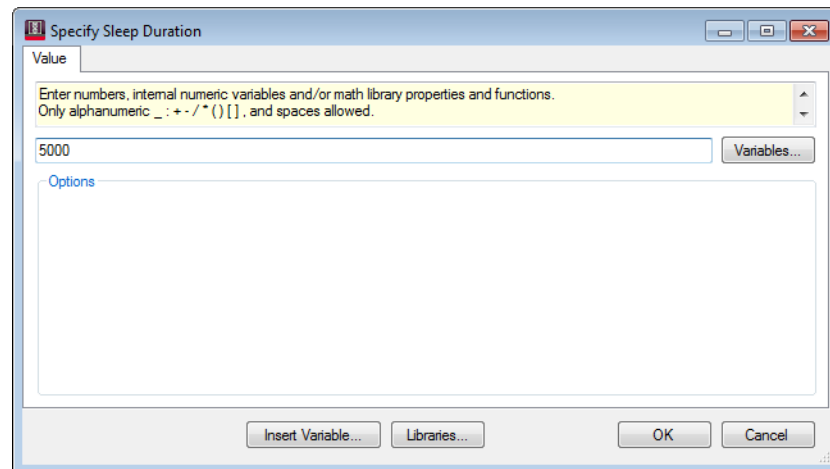
Click **OK** to continue, or click **Cancel** to abort.

Pauses can be useful when debugging tests.

Sleeping

You can halt a test run for a specified number of milliseconds or use a math expression to define the duration. This can be useful for letting an external instrument's output settle down.

- 1 In the application generator's Tests tab, either add or edit a test.
- 2 In the Test Definition dialog box's Steps area, click **Manage...**
- 3 In the Manage Test dialog box, click **Add...**
- 4 In the Select Step Type dialog box, select **Sleep**.
- 5 In the Specify Sleep dialog box, enter the number of milliseconds to sleep or a math expression that provides this value.

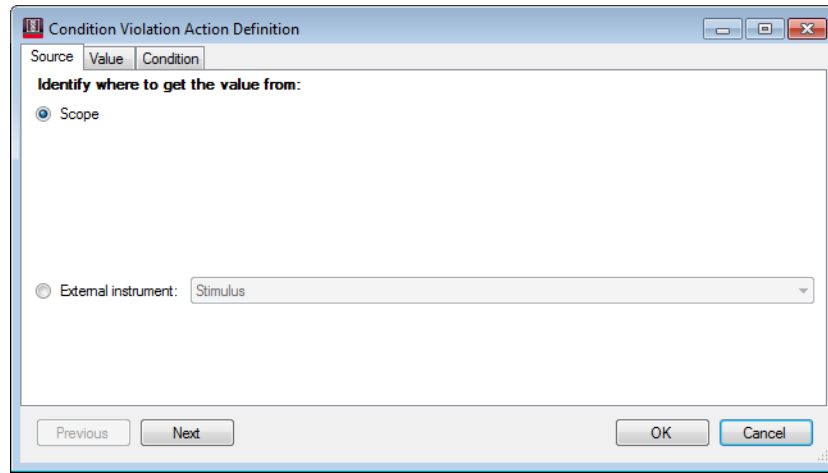


- 6 Click **OK**.

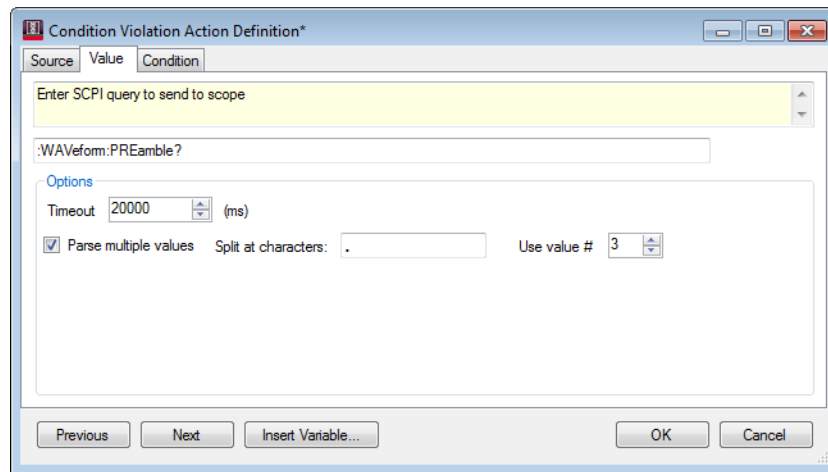
Sometimes, a "verify condition" step is more reliable than a "sleep" step when waiting for an external instrument's output to settle down. See ["Verifying a Condition"](#) on page 121.

Verifying a Condition

- 1 In the application generator's Tests tab, either add or edit a test.
- 2 In the Test Definition dialog box's Steps area, click **Manage...**
- 3 In the Manage Test dialog box, click **Add...**
- 4 In the Select Step Type dialog box, select **Verify condition**.
- 5 In the Condition Violation Action Definition dialog box, if external instruments are defined, select either the oscilloscope or the instrument as the source of the value.



- 6 In the Condition Violation Action Definition dialog box, select the Value tab and enter the SCPI query whose return value you want to check. At any time, type ALT+I to insert a variable reference.

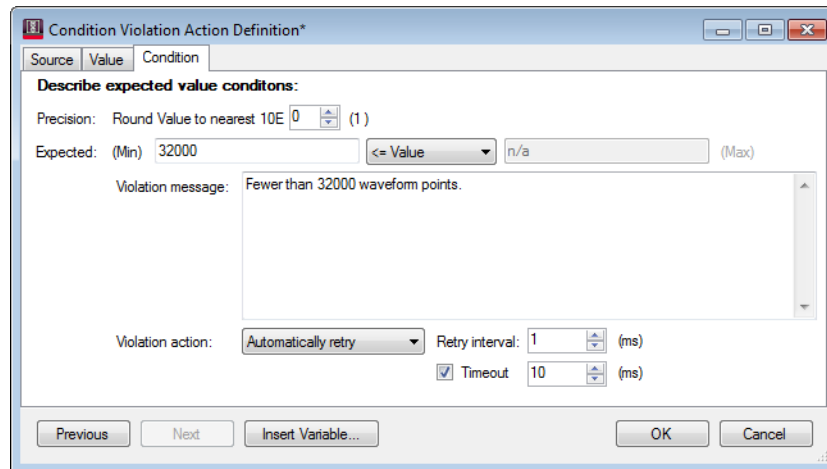


- 7 Change the **Timeout** value if desired.
- 8 In the Condition Violation Action Definition dialog box, select the Condition tab and:
 - a Specify the **Precision** of the returned value.
 - b Enter the **Expected** value or value range.

NOTE

You may enter either a number or an internal variable. When using a variable, do not enter expressions (such as "x+1").

- c Enter the **Violation message** that appears if the value is not as expected. At any time, type ALT+I to insert a variable reference.
- d Select the **Violation action**. You can:
 - Abort Run.
 - Abort Test. With this action, the run may abort or continue, depending upon the generated app's user preference setting: Run:Test Abort Action.
 - Automatically retry. With this action, you can also enter the **Retry interval** and specify a **Timeout** value.
 - Offer to retry.
 - Offer to retry or ignore.



- 9 Click **OK**.

Manually Managing SCPI Errors

By default, all Single SCPI Command steps will check the instrument's error queue after execution. If it contains any messages, the step will abort and report the message.

However, some instruments set messages in their error queue for non-fatal reasons. To handle these cases, you can disable the default Single SCPI Command step behavior and manually handle instrument error queue checking.

After each Single SCPI Command step in which you have checked the "Ignore SCPI Errors" box, you should execute the following steps to manually handle any errors that may have been suppressed:

```
GetScpiError
While (Reserved_ScpiError <> "")
  {Optionally check Reserved_ScpiError variable to see what error it contains}
  GetScpiError
```

```
EndWhile
ClearScpiErrorCache
```

After a Single SCPI Command with the Ignore setting turned on executes, the step moves all of the messages found in the instrument's error queue into a cache inside the UDA.

Each GetScpiError step takes one message out of the cache and puts it into the Reserved_ScpiError variable.

The ClearScpiErrorCache step will simply discard all messages in the cache. In the example above, this step is optional if you allow the while loop to completely empty the cache.

Reporting an Intermediate Value

An intermediate value is some interesting measurement or status value (besides the final value) that you can report in the test result.

NOTE

You cannot execute Intermediate Value steps in an event because these steps add values to the results of specific tests, whereas events execute in-between tests.

- 1 In the application generator's Tests tab, either add or edit a test.
- 2 In the Test Definition dialog box's Steps area, click **Manage...**
- 3 In the Manage Test dialog box, click **Add...**
- 4 In the Select Step Type dialog box, select **Intermediate Value**.
- 5 In the Value Definition dialog box, enter:
 - **Label** – this is a caption displayed in the results.
 - **Value** – click **Set** to specify whether the value is to be obtained by querying an instrument, from a config variable, from a math expression, or by reading a file (such as one created by an external application). See "**Getting Values**" on page 198.
 - **Units** – this is displayed in the results.

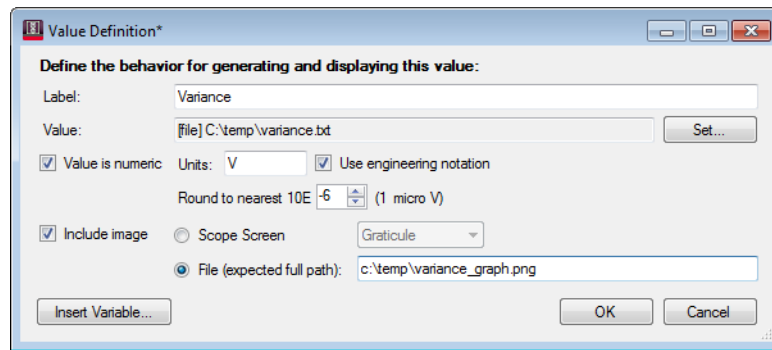
NOTE

Wherever you can enter "units", never include an engineering prefix. For example, if you are expecting nanoseconds, use "s" for units. The application will affix the appropriate engineering prefix to the "s" based upon the actual value. Use the precision control to determine desired rounding.

- **Use engineering notation** – formats numbers automatically. For example, 0.3 V becomes 300 mV.
- **Round to nearest 1E** – specifies the precision applied to the value.
- **Include image:**

- **Scope Screen** – specifies that either an oscilloscope **Graticule** or **FullScreen** image be included. (For Add-Ins, the interface level must be 1.10 or higher to use **FullScreen** – see **Chapter 4**, “Determining Add-In Compatibility,” starting on page 81.)
- **File** – specifies that an image generated by an external application be included. Again, this is the path the generated app will expect the file to exist in.

The **Insert Variable...** button reminds you that you can use ALT+I to insert a variable in the file's path.

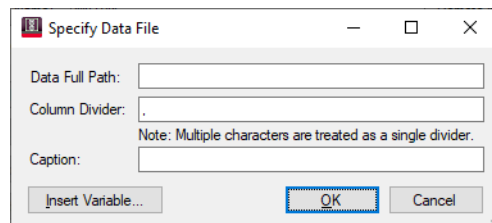


- 6 Click **OK**.

Reporting Tabular Data

You can add table-formatted data to the generated app's HTML Report by using an HTML Table step. The step works by reading a CSV file that you create, splitting the data at each comma and newline, and formatting the data into rows and columns.

- 1 Enter the **Data Full Path** to the data file to be formatted as a table.
- 2 Enter the **Column Divider** character(s) used to determine where to split each line of the data file.



- 3 Optionally enter a **Caption** for the table.
- 4 Click **OK**.

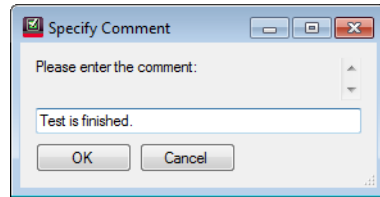
NOTE

The minimum table size is one column and two rows.

Adding a Comment

This step adds a comment to the test.

- 1 In the application generator's Tests tab, either add or edit a test.
- 2 In the Test Definition dialog box's Steps area, click **Manage...**
- 3 In the Manage Test dialog box, click **Add...**
- 4 In the Select Step Type dialog box, select **Comment**.
- 5 Enter the comment.



- 6 Click **OK**.

Return Steps

NOTE

Not available in tests.

You can use 'Return' steps to completely exit the current subroutine or event before its last step (this is not available in tests - only in subroutines or events). For example,

```
Subroutine MySub:
  ...steps...
  While (predicate)
    ...steps...
    If (predicate)
      Return // immediately exit MySub without executing any of the
steps below
    EndIf
  EndWhile
  ...steps...
```

If, Else, and EndIf Steps

You can use "If", "Else", and "EndIf" steps to create conditional blocks of steps. Add these steps around existing steps in tests, subroutines, and events to control execution flow. For example:

```
If (predicate)
    ...steps... //These steps execute only if the predicate is true
EndIf

If (predicate)
    ...steps...//These steps execute only if the predicate is true
Else
    ...steps...//These steps execute only if the predicate is false
EndIf
```

If/EndIf blocks may be embedded inside other If/EndIf blocks and inside While/EndWhile loops.

- 1 In the application generator's Tests tab, either add or edit a test.
- 2 In the Test Definition dialog box's Steps area, click **Manage...**
- 3 In the Manage Test dialog box, click **Add...**
- 4 In the Select Step Type dialog box, select either the **If()** radio button, **Else -** radio button, or **End()** radio button.
- 5 If you select **If()**, you will be prompted to define the predicate. A predicate is a set of comparisons that evaluates to true or false. It may be as simple as a single term (such as $x < 7$) or it may be a complex combination of multiple terms, such as $x < 7$ AND $(y=8$ OR $z > 9)$.
- 6 To define a single term predicate, click the **Add** button and then specify the variable type in the Predicate Term Definition dialog box. Then select the variable to be compared.

NOTE

Reserved variables are predefined by the application generator. These are read-only text variables whose names begin with "Reserved_". For more information, select the **Tools > Manage > Reserved Variables** menu item.

Once you have the variable selected, select the comparison type (unless only one is available for the variable you have chosen) and operator. Click **Set** and enter the value to compare the variable to. Depending on the above choices and what has been defined in the current project, you may be able to choose from config variables, math expressions, and/or text.

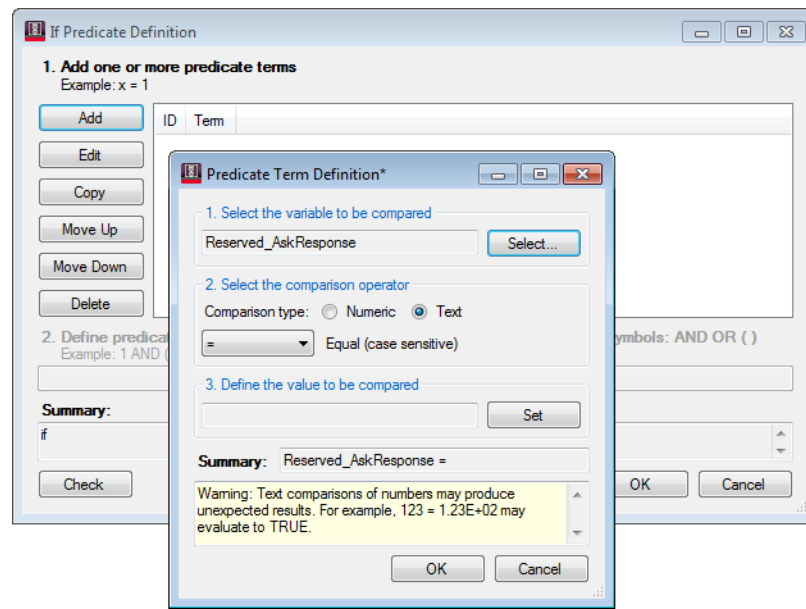
NOTE

You may not include library functions or properties in predicates.

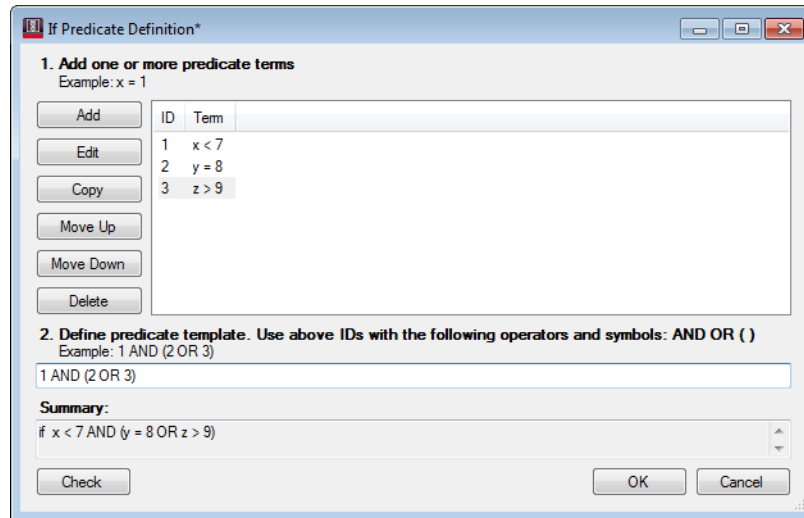
Once you have all of this set up, the **Summary** field will show you what the current term looks like.

CAUTION

Text comparisons of numbers may produce unexpected results. For example, $123 = 1.23E+02$ will evaluate to FALSE.



To define a multi-term predicate, follow the steps above to define each of the terms. After you define a term, it will be added to the If Predicate Definition dialog box and an ID will be listed to the left of each term. Also in the If Predicate Definition dialog box, you can define a predicate template that uses these ID values (see screen shot below).



So, after you define all of the terms you need, define a predicate template using ID numbers as placeholders for your terms. For example, if you have defined three terms:

```
1      x < 7
2      y = 8
3      z > 9
```

then the following template:

```
1 AND (2 OR 3)
```

will result in a predicate of:

```
x < 7 AND (y = 8 OR z > 9)
```

The **Summary** field will show you what the current predicate looks like after these substitutions take place.

7 Click **OK**.

While, Break, Continue, and EndWhile Steps

You can use "While", "Break", "Continue", and "EndWhile" steps to repeat blocks of steps. Add these steps around existing steps in tests, subroutines, and events to control execution flow. For example:

```
While (predicate)
```

```
    ...steps... //These steps execute repeatedly until the predicate becomes false
```

```
EndWhile
```

```
While (predicate)
```

```

...steps...
Break //Causes the point of execution to jump to the step after the nearest EndWhile
...steps...

EndWhile

While (predicate)

...steps...
Continue //Causes the point of execution to jump back to the nearest While step
...steps...

EndWhile

```

While/EndWhile loops may be embedded inside other While/EndWhile loops and inside If/EndIf blocks.

- 1 In the application generator's Tests tab, either add or edit a test.
- 2 In the Test Definition dialog box's Steps area, click **Manage...**
- 3 In the Manage Test dialog box, click **Add...**
- 4 In the Select Step Type dialog box, select either the **While {}** radio button, **Break** radio button, **Continue** radio button, or **End{}** radio button.
- 5 If you select **While{}** , you will be prompted to define the predicate. A predicate is a set of comparisons that evaluates to true or false. It may be as simple as a single term (such as $x < 7$) or it may be a complex combination of multiple terms (such as $x < 7$ AND $y = 8$ OR $z > 9$).
- 6 To define a single term predicate, Click the **Add** button and then specify the variable type in the Predicate Term Definition dialog box. Then select the variable to be compared.

NOTE

Reserved variables are predefined by the tool. These are read-only text variables whose names begin with "Reserved_". For more information, select the **Tools > Manage > Reserved Variables** menu item.

Once you have the variable selected, select the comparison type (unless only one is available for the variable you have chosen). Click **Set** and enter the value to compare the variable to. Depending on the above choices and what has been defined in the current project, you may be able to choose from config variables, math expressions, and/or text.

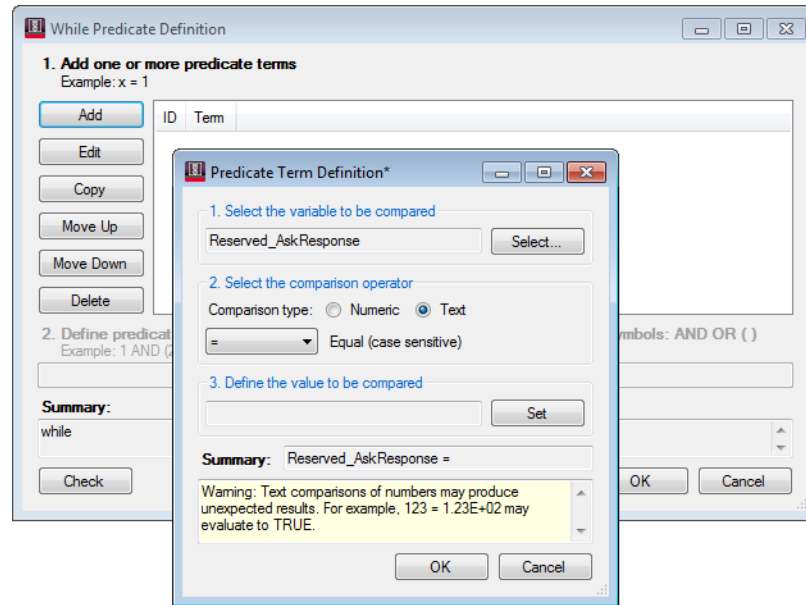
NOTE

You may not include library functions or properties in predicates.

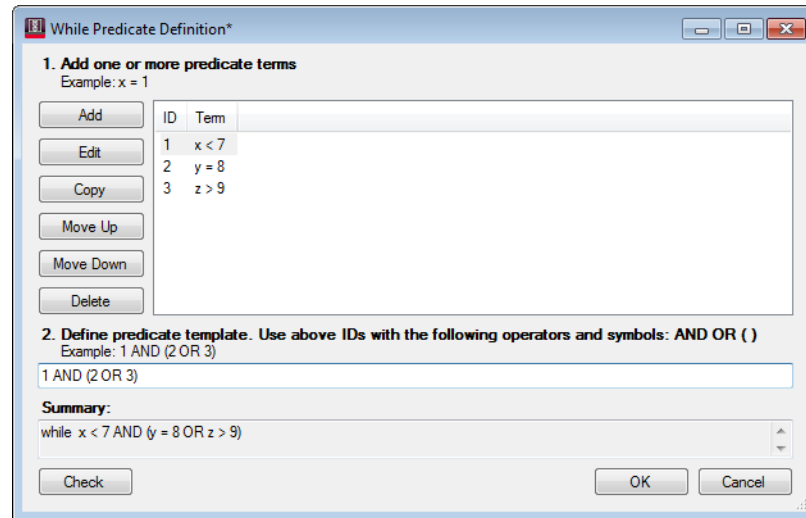
Once you have all of this set up, the **Summary** field will show you what the current term looks like.

CAUTION

Text comparisons of numbers may produce unexpected results. For example, $123 = 1.23E+02$ will evaluate to FALSE.



To define a multi-term predicate, follow the steps above to define the terms. After you define a term, it will be added to the While Predicate Definition dialog box and an ID will be listed to the left of each term. Also in the While Predicate Definition dialog box, you can define a predicate template that uses these ID values (see screen shot below).



So, after you define all of the terms you need, define a predicate template using ID numbers as placeholders for your terms. For example, if you have defined three terms:

```
1      x<7
2      y=8
3      z>9
```

then the following template:

```
1 AND (2 OR 3)
```

will result in a predicate of:

```
x < 7 AND (y = 8 OR z > 9)
```

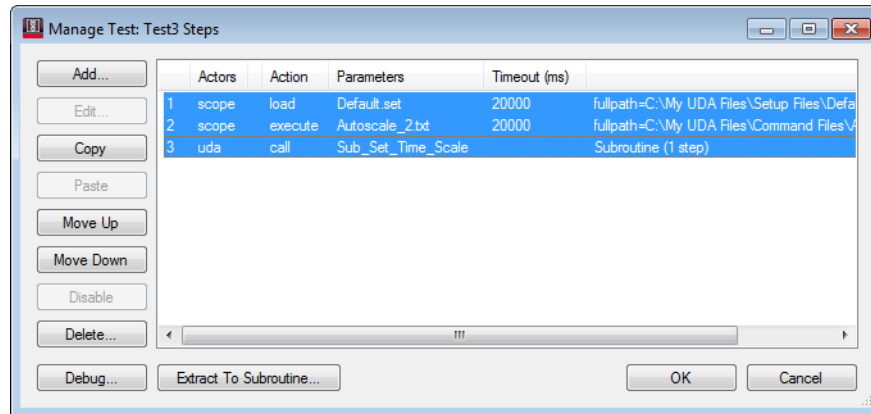
The **Summary** field will show you what the current predicate looks like after these substitutions take place.

- 7 Click **OK**.

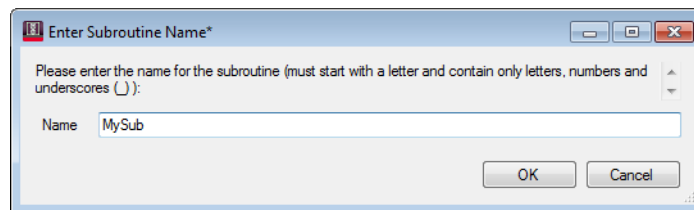
Refactoring: Extracting Steps to a Subroutine

From any Manage Steps dialog box (Test, Subroutine, or Event), you can extract one or more steps to a subroutine.

- 1 In the Manage Steps dialog box (Test, Subroutine, or Event), use Shift-click or Ctrl-click to select one or more steps.

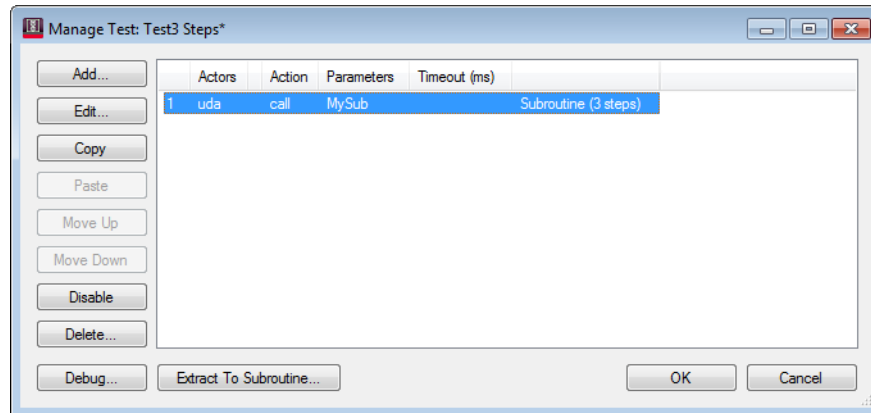


- 2 At the bottom of the dialog box, click **Extract to Subroutine...**
- 3 In the Enter Subroutine Name dialog box, enter the name and click **OK**.



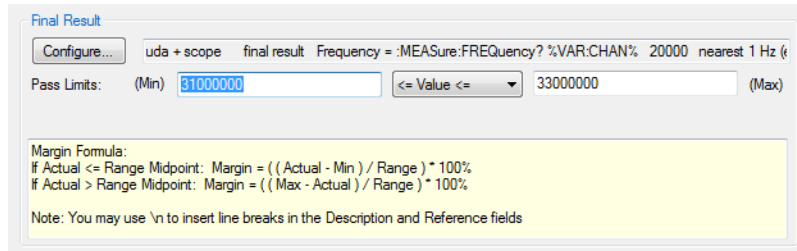
Back in the Manage Steps dialog box, the steps are replaced by a single Call Subroutine step.

6 Adding/Exporting/Importing Tests



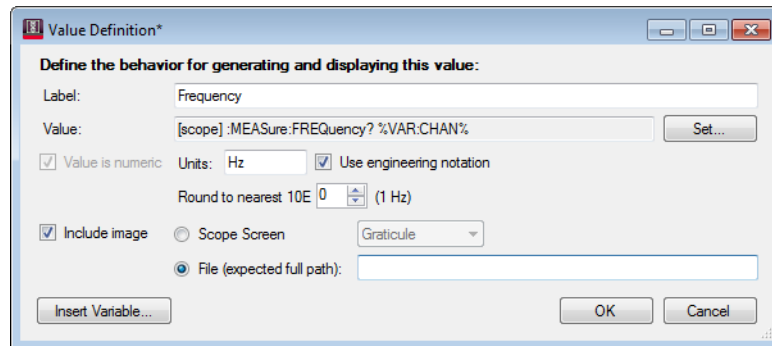
Defining Final Result Display Behavior

When adding or editing tests, the Test Definition dialog box's Final Result area lets you define the result evaluation behavior.



- 1 In the application generator's Tests tab, either add or edit a test.
- 2 In the Test Definition dialog box's Final Result area, click **Configure**.

There is a **Set** button for specifying whether the value is to be obtained by querying an instrument, from a config variable, from a math expression, or by reading a file (such as one created by an external application). See "[Getting Values](#)" on page 198.



- 3 In the Value Definition dialog box, enter:
 - **Label** – this is a caption displayed in the results.
 - **Value** – click **Set** to specify whether the value is to be obtained by querying an instrument, from a config variable, from a math expression, or by reading a file (such as one created by an external application).
 - **Units** – this is displayed in the results.

NOTE

Wherever you can enter "units", never include an engineering prefix. For example, if you are expecting nanoseconds, use "s" for units. The application will affix the appropriate engineering prefix to the "s" based upon the actual value. Use the precision control to determine desired rounding.

- **Use engineering notation** – formats numbers automatically. For example, 0.3 V becomes 300 mV.
- **Round to nearest 10E** – specifies the precision applied to the value.
- **Include image:**
 - **Scope Screen** – specifies that either an oscilloscope **Graticule** or **FullScreen** image be included. (For Add-Ins, the interface level must be 1.10 or higher to use **FullScreen** – see **Chapter 4**, “Determining Add-In Compatibility,” starting on page 81.)
 - **File** – specifies that an image generated by an external application be included. Again, this is the path the generated app will expect the file to exist in.

The **Insert Variable...** button reminds you that you can use ALT+I to insert a variable in the file's path.

- 4 Click **OK**.
- 5 Back in the Final Result area, enter the **Limits**.

The generated application applies the entered limits to the final result to determine the pass/fail margin.

If you use a single-ended limit (for example, **Value <**, **<= Value**, etc.) of value 0 or internal variable, you may enter a "typical" value in the **Nominal** field at which you want the margin to be 100%. The info box at the bottom of the dialog box describes how margin is calculated for the currently-selected limit type.

Exporting/Importing Tests or Subroutines

You can also export a test or subroutine in order to build up a library of UDA "fragments" that can be reused (imported) on other projects and shared with other users. In order to export a fragment, the current project must be free of build errors. This ensures fragments meet a minimum quality standard. Similarly, to import a fragment, the current project must be free of build errors. This is a double-check to enable the tool to determine if the imported fragment has introduced any errors.

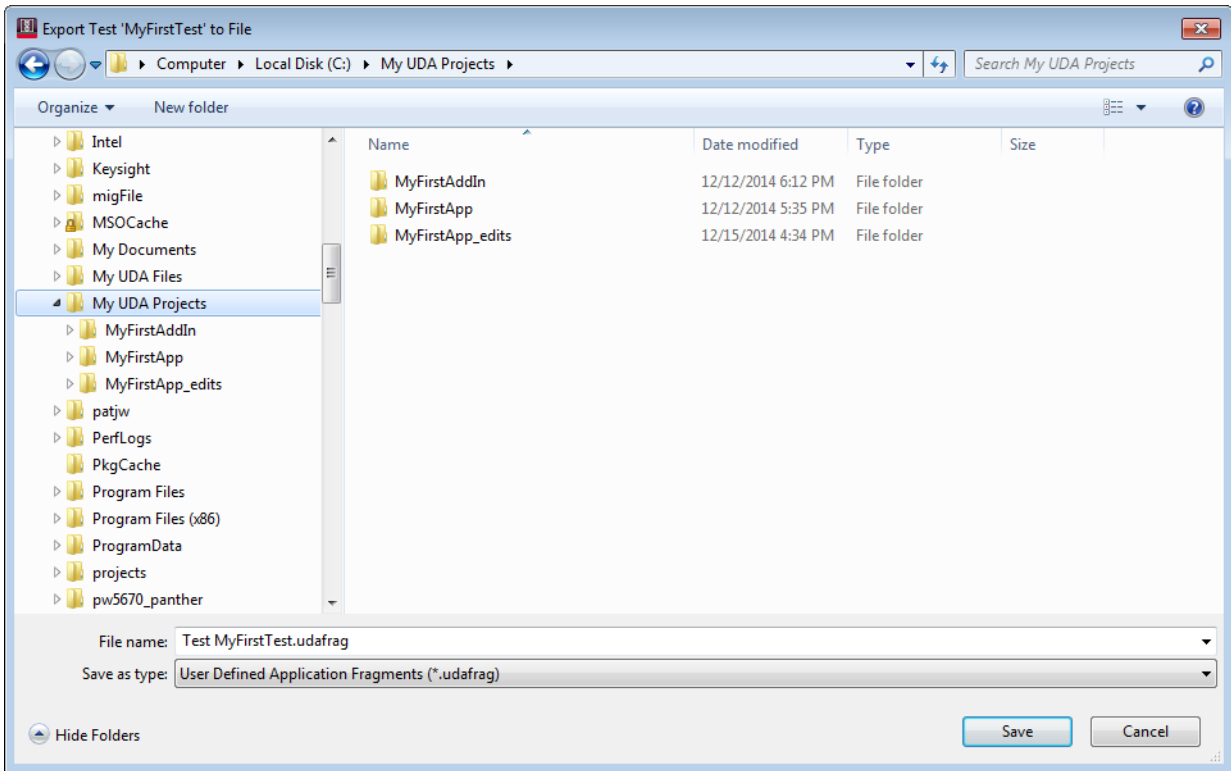
When you export a test, consider that no steps from any Run Action (Event) are ever exported. Therefore, if the test depends upon steps that execute in a Run Action, clients who import the test fragment will need to manually add those steps to the appropriate Run Action (Event).

When you import a fragment, name conflicts with existing items are resolved by renaming the items being imported. The exception to this is for User Files. When opening a fragment that contains User Files (for example, Command Files), you will be prompted for instructions on how to handle them. If you choose to "Automatically locate these files" and a file with the same name and time stamp already exists at the specified location, the importing item will simply use the existing file.

To export a test or subroutine, follow these steps:

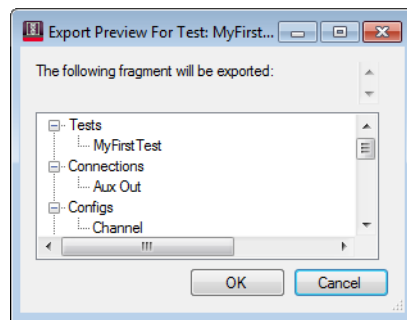
- 1 In the Tests tab, select the test you want to export. Or, in the Subroutines tab, select the subroutine you want to export.
- 2 Choose **File > Export > Test** or **File > Export > Subroutine**. The following dialog box will appear.

6 Adding/Exporting/Importing Tests



Specify the location to save the .udafraq file.

- 3 An export preview dialog box will show you all of the elements of the current project that are referenced by the test or subroutine. These will also get exported.



NOTE

When importing a fragment not created on the same machine, if the fragment contains user files, you may want to export these files instead of letting the tool "automatically" find them because it may find same-named files on your PC, which may have different contents.

To import a test or subroutine, follow these steps:

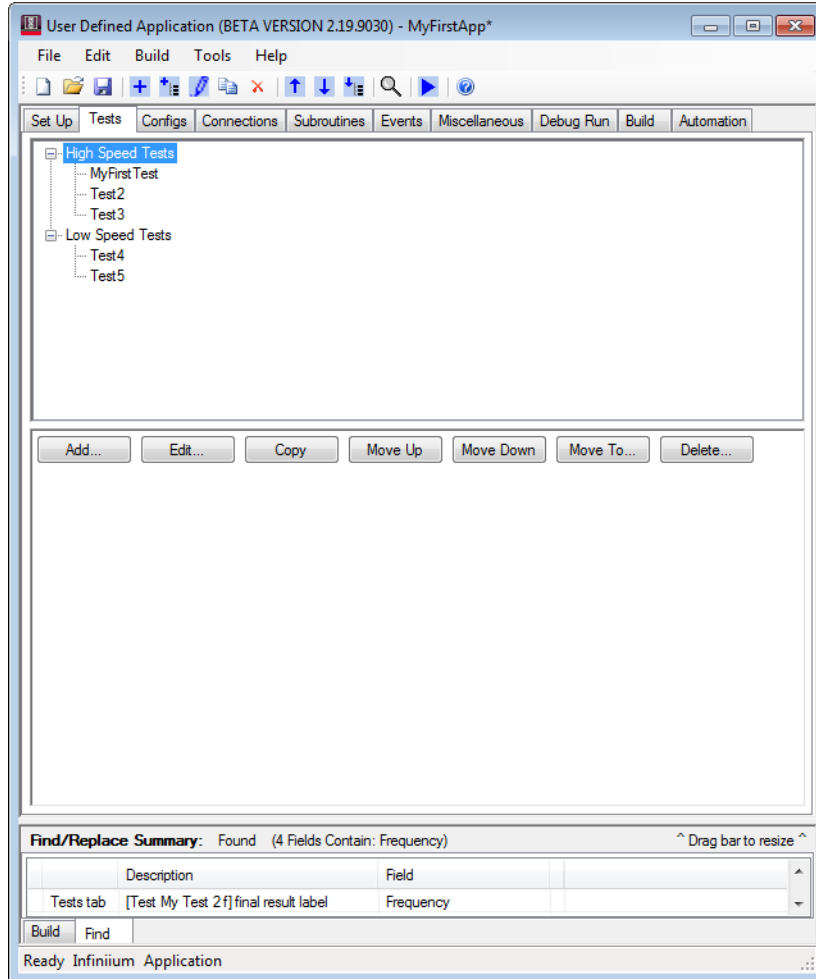
- 1 Choose **File > Import > Test** or Click **File > Import > Subroutine**.
- 2 Specify the location of the .udafrag file containing the test or subroutine.
- 3 An export preview dialog box will show you all (if any) of the referenced elements that will also get imported with the test or subroutine. After the elements are imported, the project is built. If there are errors, you will have the opportunity to abort the import.



NOTE






When importing a fragment not created on the same machine, if the fragment contains user files, you may want to export these files instead of letting the tool "automatically" find them because it may find same-named files on your PC.

Grouping Tests

The Tests tab lets you group tests. When running your application, test groups let you select and perform grouped subsets of tests.



To perform this test group action:	Click this button:	Or choose this from the main menu:	Or use this keyboard shortcut:	Or click this toolbar button:
To add a test group		Edit > Add Group...	Ctrl+P	
To move the selected group up in the hierarchy	Move Up	Edit > Move Up	Ctrl+Up	

To perform this test group action:	Click this button:	Or choose this from the main menu:	Or use this keyboard shortcut:	Or click this toolbar button:
To move the selected group down in the hierarchy	Move Down	Edit > Move Down	Ctrl+Down	
To move the selected group into another group in the hierarchy To move the selected test into a group	Move To...	Edit > Move To Group	Ctrl+Right	
To copy the selected group (including contents)	Copy	Edit > Copy	Ctrl+C	
To edit the selected group name	Edit...	Edit > Edit...	Ctrl+E	
To delete the selected group	Delete...	Edit > Delete...	Ctrl+D	

Debug Run Button

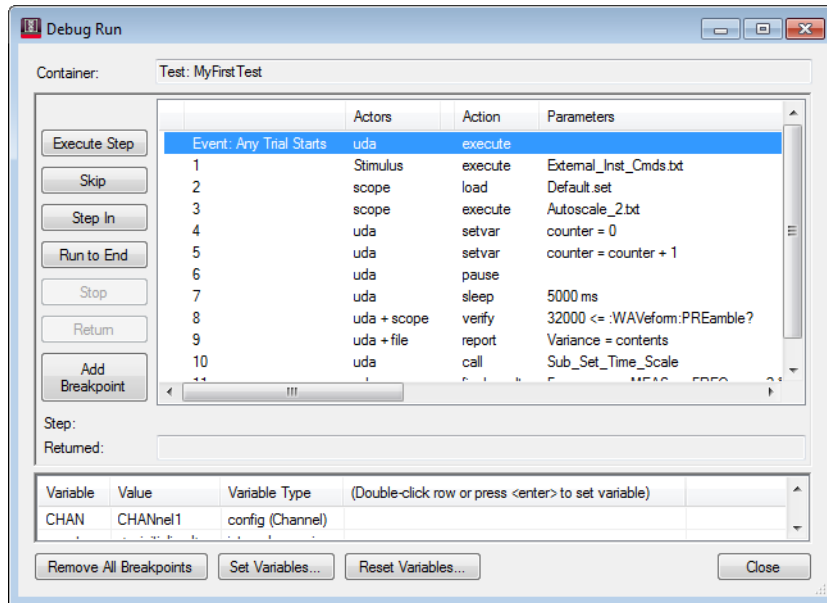
The Debug Run button in the Test Definition dialog box opens a dialog box that lets you specify which, if any, event scripts you want to execute with the test. This allows you to, in one run, execute all of the steps that would execute when you run a test from within a launch.

Debug Run buttons are different from Debug buttons (found by clicking **Manage** in a Test Definition dialog box, for example). A Debug button only lets you debug the steps you see in the current dialog box. Debug Run buttons, on the other hand, enable you to combine the steps with the final result step in one Debug Run and, if you have defined event actions, you can also include them in the order they would execute relative to the tests' steps.

During Debug Run, user files are referenced in their original locations. Miscellaneous files marked for installation in the standard location "Application" are an exception as they are copied to the internal cache at the start of every Debug Run (and during a Debug Run Reset). This enables you to use the variable `RESERVED_OtherFilePath` in Debug Run as well.

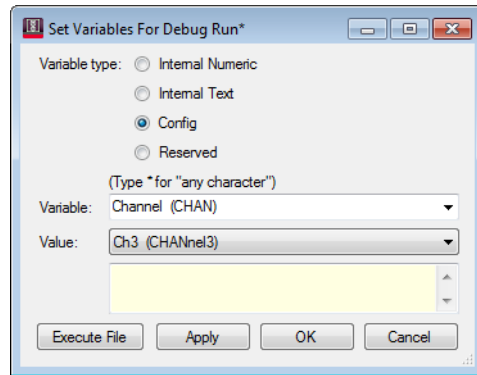
The Debug Run dialog box displays the current values of all internal and config variables directly referenced by the steps in the list (underneath the **Returned** field). When a DebugRun or Debug button is pressed, the run engine resets all internal and reserved variables to an uninitialized state and all config variables to their default values. You can prevent this autoreset using the **Maintain** option in **Tools > Options > Debug** from the main menu. You can also manually perform these resets by selecting the **Debug Run** options from the main menu.

The Debug Run dialog box that appears after pressing the button displays all of the steps you have configured for the test.



The following buttons are available in this dialog box.

- **Execute Step** – Pressing this button fully executes the currently highlighted step in one action. You can also just double-click on each step to get it to execute.
- **Skip** Pressing this button skips the currently highlighted step.
- **Step In** – When the highlighted step is an Execute Script step, subroutine, or run event, clicking this button will open a new Debug Runner with that script's steps. When the highlighted step is a Call Subroutine step, clicking this button will open a new Debug Runner with that subroutine's steps.
- **Run to End** – Pressing this button executes the steps sequentially until the end or until a breakpoint.
- **Stop** – Pressing this button stops the debug run.
- **Return** – After using Step In, this button will return you to the calling context.
- **Add/Remove Breakpoint** – Pressing this button adds/removes a breakpoint to the currently highlighted step. A "B" is placed next to the step in the list to signify that there is a breakpoint at that location.
- **Set Variables...** – Lets you change the current value of the variable selected in the bottom pane. Alternatively, you can double-click a row to initiate modification. Either action displays this dialog box:



To set a variable's value:

- a Choose the type of variable.
- b Select the variable.
- c Enter the desired value.

For Configs defined to "Allow user to add new choices", you may type in any arbitrary text in the value field; for all other configs, you may only choose one of the choices in the Value combo box.

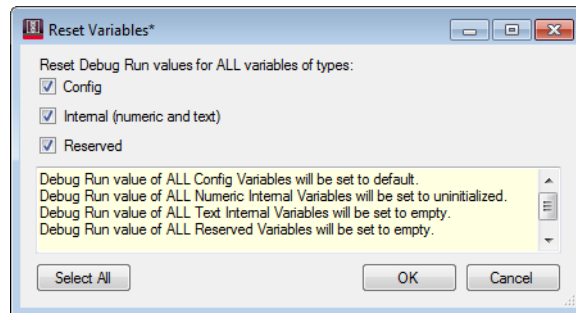
- d Click **OK** to set the variable and close the dialog box.

Or, click **Apply** to set the variable and leave the dialog box open. The **Cancel** button will not undo assignments made using **Apply**.

Or, click **Execute File** to execute a CSV file containing variable name/value pairs, formatted one pair per line. For example:

```
MyNumericVariable,123
MyTextVariable,"abc"
```

- **Reset Variables...** – Lets you reset the Debug Run values of selected types of variables to their default, uninitialized, or empty values.



The lower pane is a status window showing you the values of all internal and config vars directly referenced by the steps above.

NOTE

When this screen displays, all variables are reset to defaults. To change this behavior, see the **Tools > Options > Debug** tab.

NOTE

Debug run variable values are saved with the project and loaded when you open a project. If you want to use the loaded values, be sure the "Maintain" option (**Tools > Options > Debug**) is checked before you start a Debug Run session.

7 Adding Config Variables

Using Variable Substitutions / 148

Grouping Config Variables / 149

Config variables let users of your generated application select or enter variable values when running the application.

Config variables are optional. Your application can have no Config variables. However, in practice Config variables can be convenient for letting users specify the oscilloscope channels used for different input signals, or probing options (if they affect test execution), etc.

Config variable names are case-sensitive. If you change a config's label, all steps referring to that particular config will be updated automatically.

For instructions on adding Config variables, see: "[Step 4: Add Config variables \(optional\)](#)" on page 38

You can group Config variables into test groups if you want them to appear only if the test group is visible. See: "[Grouping Config Variables](#)" on page 149

Using Variable Substitutions

In SCPI strings, connection instructions, and SCPI command files (in the file itself, not in the full path to the command file), use the following syntax to tell the generated application to use the current value for the specified variable:

```
%VAR:MyVariableName%
```

Where MyVariableName is the text you typed in the Variable field of the Config Definition dialog box. At runtime, the generated application will substitute the value of that Config variable, replacing the entire %...% field.

Examples:

- SCPI command :ACQ:POIN %VAR:Depth%
- Connection instructions: Connect input to scope channel %VAR:Channel%

For a full list of the places where you can use variable substitutions, see "[Variable Substitution](#)" on page 209.

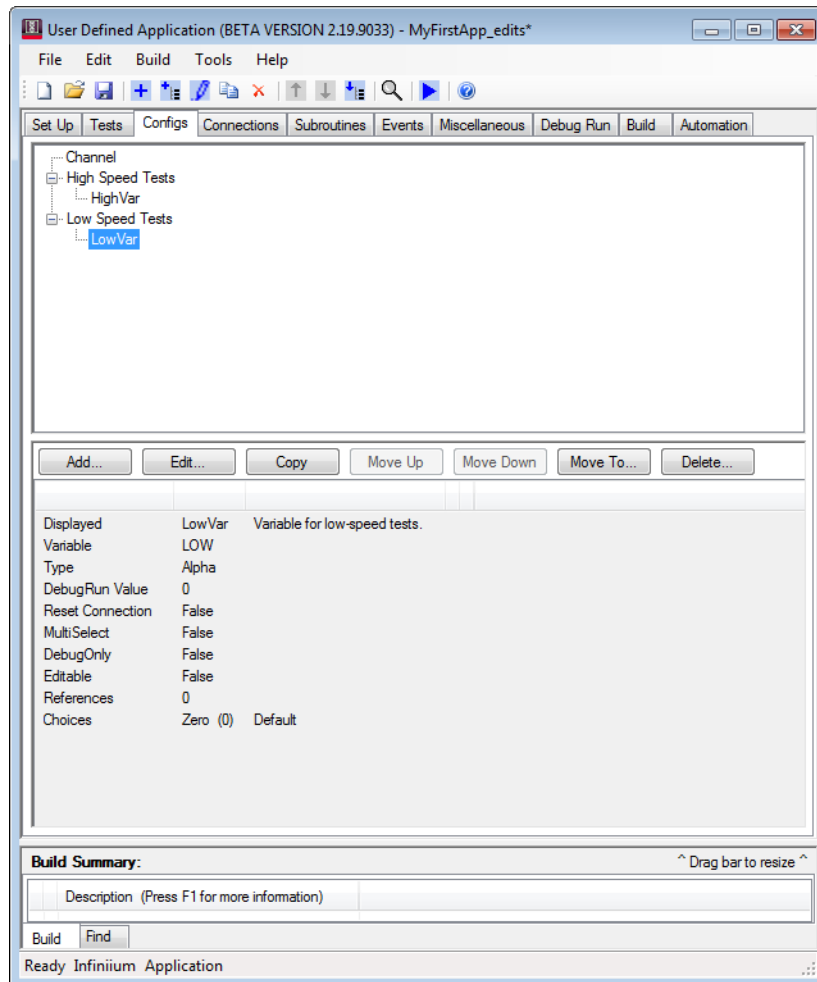
NOTE

In all text boxes where you can enter this syntax, you can also type ALT+I to automatically generate and insert the variable substitution.

Grouping Config Variables

The Configs tab lets you place configuration variables into the same test groups as specified in the Tests tab.

When running your application, the configuration variables in a certain group are only displayed when the test group is visible. Configuration variables at the root level are always displayed.



The commands to add, move, move a Config variable into a group, copy, edit, and delete are the same as when working with test groups. See "[Grouping Tests](#)" on page 140.

8 Adding Connection Instructions

Connection instructions are an optional feature that give users running your application instructions on how to make the proper connections (for example, between the oscilloscope and device under test) for a certain test.

Your connection instructions can be a single image file and simple instructions, or you can use an HTML file with a single image and formatted instruction text, or you can just have instructions (with no image).

For a description of how to add connection instructions, see "[Step 5: Add connection instructions \(optional\)](#)" on page 43.

9 Adding Switch Matrix Control

Step 1: Identify Test Points / 154

Step 2: Identify Compatible Switch Instruments / 156

Step 3: Configure Options / 157

Step 4: Assign Test Points to Connections / 159

Step 5: (Optional) Include Sample Switch Settings / 160

Assigning Test Points to Connections / 161

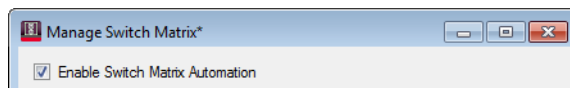
Simulators / 168

If your application uses connection instructions that involve changing the physical connections between test points (on your device under test) and the oscilloscope, you can optionally use a switch matrix to automatically implement the connection changes.

NOTE

Switch matrix control is only available for Applications.

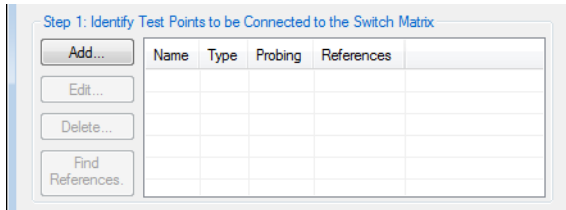
To begin, display the switch matrix configuration screen: **Tools > Manage > Switch Matrix**:



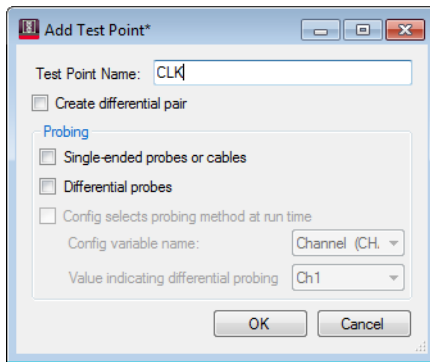
Enable – Master switch for this feature.

Step 1: Identify Test Points

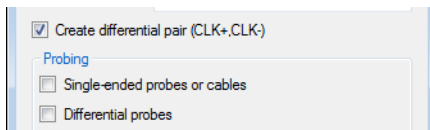
First you need to identify all of the test points on your device under test that will be routed through the switch matrix.



Add... – Prompts you to describe a test point on your device under test:



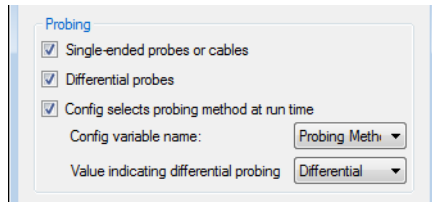
Create differential pair – Check this if your test point is differential. Two test points will automatically be created. You will also need to specify which probing methods your application will support for this test point:



Single-ended – Check this if at least one connection involving this test point specifies use of cables or single-ended probes for this test point.

Differential – Check this if at least one connection involving this test point specifies use of a differential probe for this test point.

Config – If you check both of the above and you have a configuration variable for the user to select which probing method to use at run-time, then select the variable and which of its choices corresponds to differential:



Here is an example of an added differential signal that can be probed either way, with a user-selectable probing method:

Name	Type	Probing	Referen
CLK+	Differential	Probing Method (PRBTYPE)	0
CLK-	Differential	Probing Method (PRBTYPE)	0

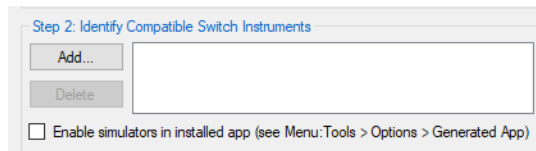
Edit... – View/Modify details of the selected test point. When you change a differential test point, its pair is automatically updated as well.

Delete... – Remove the selected test point from the list. When you remove a differential test point, its pair is automatically removed as well.

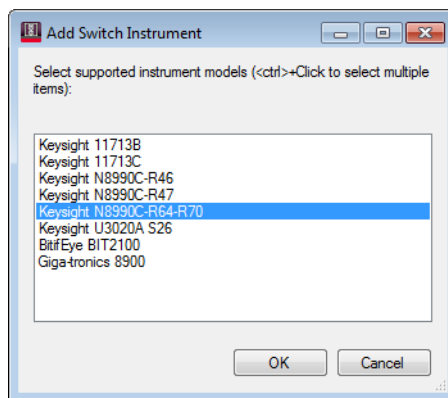
Find References – Find all uses of this test point in your connection definitions.

Step 2: Identify Compatible Switch Instruments

Next, specify which of the switch instrument models supported by UDA you want the user to be able to use for this application.



Add... — Prompts you to select one or more switch instrument models:

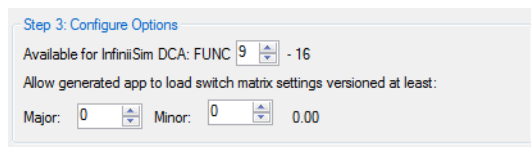
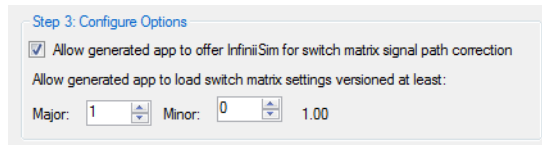


Delete — Removes the selected model from the list.

Check **Enable simulators in installed app** to include a switch instrument simulator with the generated app installer. To generate the required simulator, see **"Simulators"** on page 168.

Step 3: Configure Options

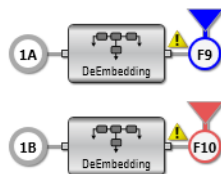
Next, specify a couple of miscellaneous options.



Allow – (Infiniium Real-Time only) Check this box to allow the generated app to offer InfiniiSim for signal path correction. Uncheck this box if your app uses InfiniiSim in its tests, events, or subroutines.

Available – (FlexDCA only) Identify the lowest function number to make available to the switch matrix module for use in path correction. All numbers below this are safe to use in your steps, while all remaining function numbers may be used by the switch matrix feature.

InfiniiSim DCA's de-embedding feature works by creating a function. So when the generated app user enables switch matrix and uses de-embedding for path correction, the switch matrix module will automatically create a de-embedding function for every oscilloscope channel in the current connection that is connected to the switch matrix. For example, in a connection that uses FlexDCA channels 1A and 1B, the following would be created:



In this situation, all of your SCPI commands that reference these channels (for example, :CHANnel1A:DISPlay ON) would need to instead reference the functions associated with them (for example, :FUNctIon9:DISPlay ON). The generated app makes this substitution automatically before sending each SCPI command or query, whenever switch matrix is enabled and path correction has been assigned.

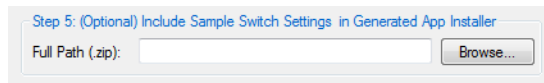
Minimum compatible version – This is similar to the saved project compatibility setting found in **Tools > Generated App Compatibility**, except this one applies to the settings created when the generated app user saves/loads switch matrix settings separately. Initially, set the version to equal the initial version you assign to your application (on the Set Up tab). Later, if you change your connections in a way that invalidates previously saved switch matrix signal paths, bump the version controls to your new application version. This will prevent users from loading old (and now incompatible) switch matrix settings.

Step 4: Assign Test Points to Connections

Now you need to declare which connections your test points are used in. This step is performed in the Connection editors, which are accessed from the Connection tab, described in "[Assigning Test Points to Connections](#)" on page 161.

Step 5: (Optional) Include Sample Switch Settings

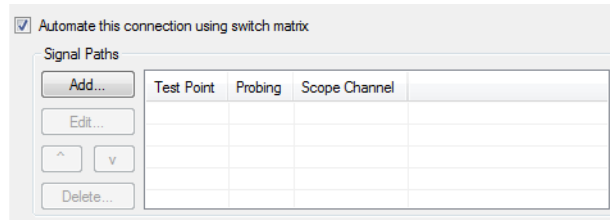
To help generated app users quickly configure their switch signal paths, you can include saved switch settings (created by running the generated app) in the installer. Then, when the generated app is run and its switch matrix module is opened, clicking **Load** will display a directory where the installed switch settings can be found. You can include multiple many saved switch settings in the .zip file.



Browse... – Navigate to the location of a compressed set of switch matrix settings files.

Assigning Test Points to Connections

When the switch matrix master switch (described above) is enabled, each connection's editor (see Connections tab) will display an automation option:



Automate – Check this to enable automation of this connection using a switch matrix.

Add... – Define a test point-to-scope channel signal path, described in "[Defining a Signal Path](#)" on page 161.

Edit... – Edit the selected signal path.

Up/Down – Rearrange the signal paths in the list (for personal preference use only, does not affect the generated app).

Delete... – Remove the selected signal path.

Defining a Signal Path

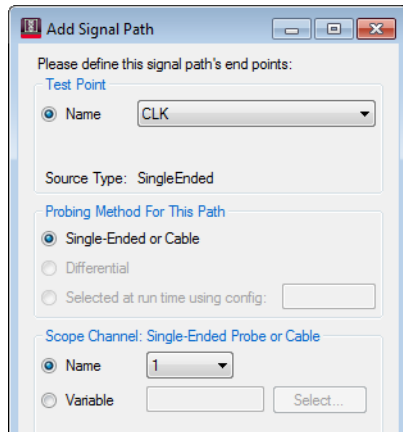
A signal path can be fully named (named test point and named oscilloscope channel) or can use config variables to specify either end. Depending upon how the test point was defined (see above), you may be able to select the probing method as well.

- "[Named Test Point](#)" on page 161
- "[Variable Test Point](#)" on page 163
- "[Variable Scope Channel](#)" on page 165
- "[Scope Channel Assignment Summary](#)" on page 166

Named Test Point

When you select a test point by name, what gets displayed depends on the test point's type.

Here is an example of a single-ended test point:

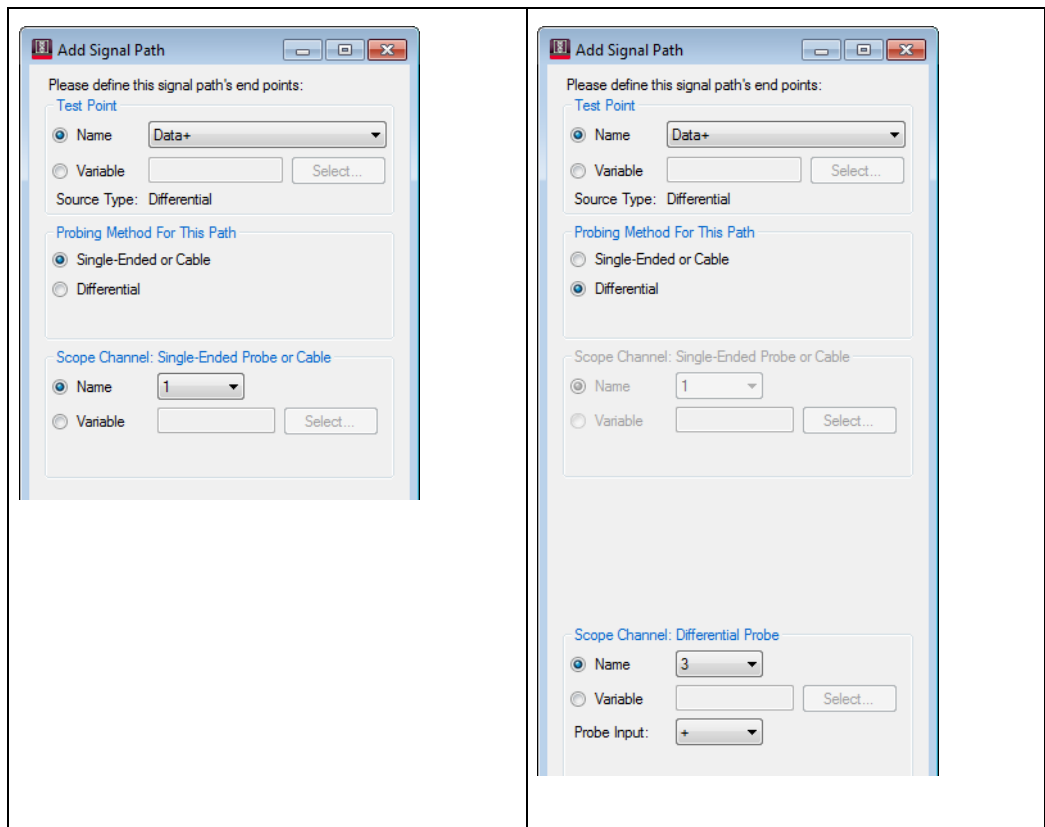


Test Point Name – List of all test points defined in **Tools > Manage > Switch Matrix**.

Probing Method – Displays each probing method supported by the test point.

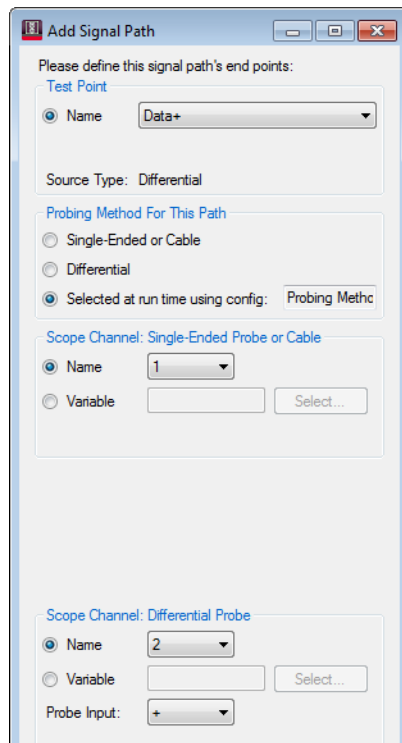
Scope Channel Name – Lists all oscilloscope channels.

Here is an example of a differential test point that allows either probing method:



The probing method you select determines which group below it is used to specify the oscilloscope channel.

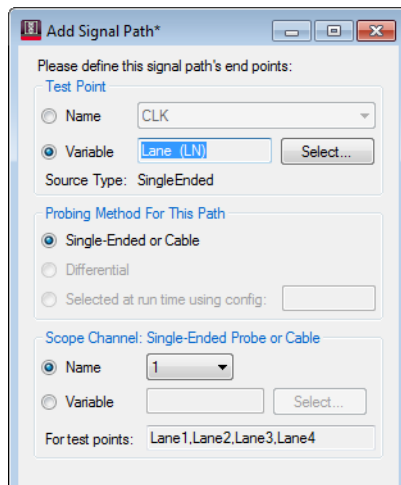
When a differential test point uses a config variable to select its probing method and you select that option, you will need to specify two oscilloscope channels to handle both of the choices that user can make:



Variable Test Point

When you use a config variable to select related test points (for example, Lane1, Lane2, ...), what gets displayed again depends upon the type of the test point.

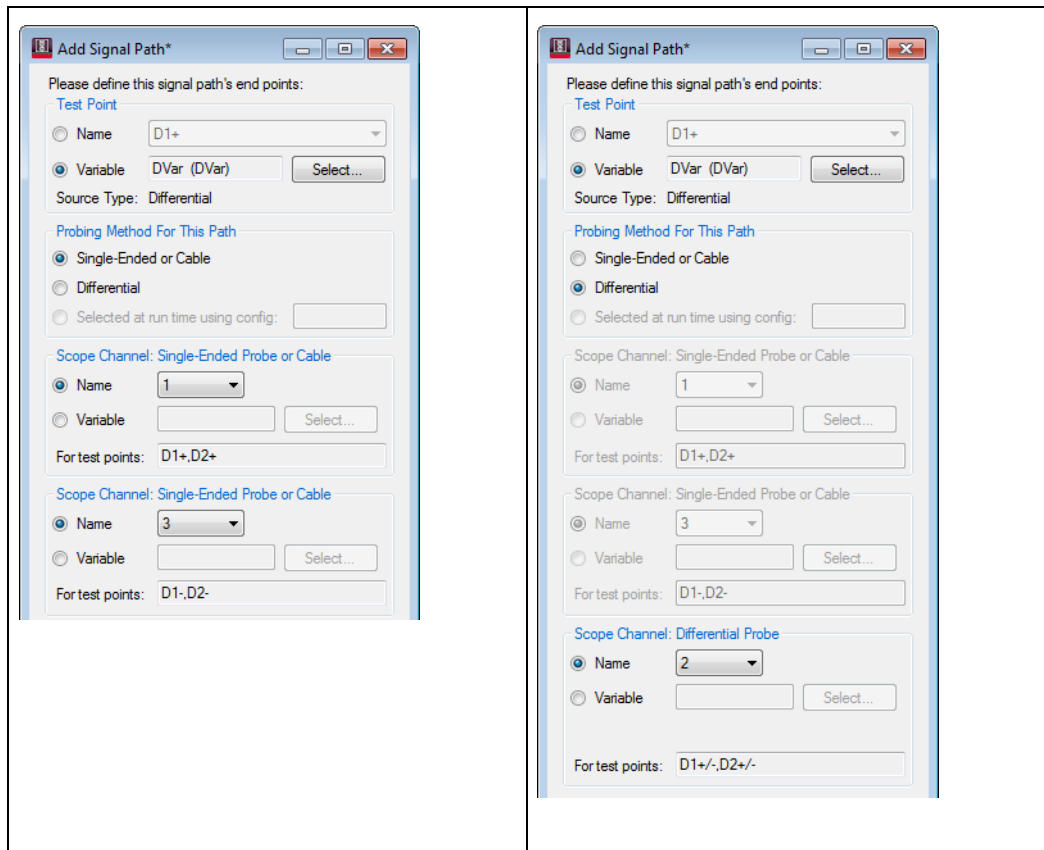
Here is an example of a group of single-ended test points being selected by a config variable:



Test Point Variable – If any config variables exist whose choice values are each identical to a test point name, these variables will be available for selection here.

For test points – When a variable is used to select test points, this field will display the individual test point names that will be selectable in the generated app. These correspond to the choice values of the config variable selected in the **Test Point** section of this screen.

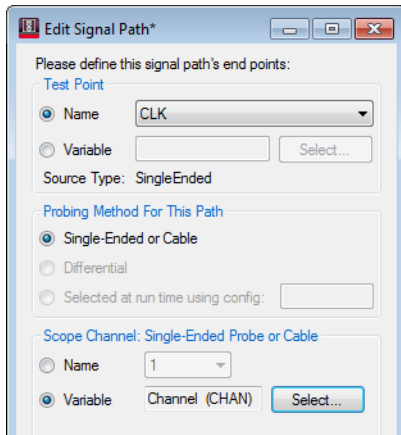
Here is an example of a group of differential test points being selected by a config variable:



Again, the probing method you select determines which group below it is used to specify the oscilloscope channel(s).

Variable Scope Channel

In all of the examples above, the oscilloscope channel was specified by name. If you have one or more config variables for allowing the generated app user to select the oscilloscope channel, you can use them in a signal path as well. Here is an example of a config variable being used to select a oscilloscope channel:



Scope Channel Variable – Any config variables that exist will be available for selection here (no checking is done to verify the config's choices match oscilloscope channel names).

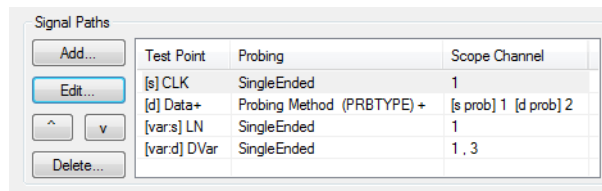
Scope Channel Assignment Summary

This table summarizes which of the three oscilloscope channel groups will need to be specified, depending on:

- 1 How the test point is selected (by name or variable).
- 2 The type of test point (single-ended, differential).
- 3 The probing method used in the connection.

	Oscilloscope Channel		
	Single-Ended	Single-Ended	Differential
Named, single-ended test point	X		
Variable, single-ended test point	X		
Named, differential test point			
- Probing Method = single-ended	X		
- Probing Method = differential			X
- Probing Method = variable	X		X
Variable, differential test point			
- Probing Method = single-ended	X	X	
- Probing Method = differential			X
- Probing Method = variable	X	X	X

Signal Paths are summarized in the table in the Connections editor:



Test Point – Includes a prefix indicating:

[s] = Named, Single-Ended

[d] = Named, Differential

[var,s] = Variable selecting single-ended test points

[var,d] = Variable selecting differential test points

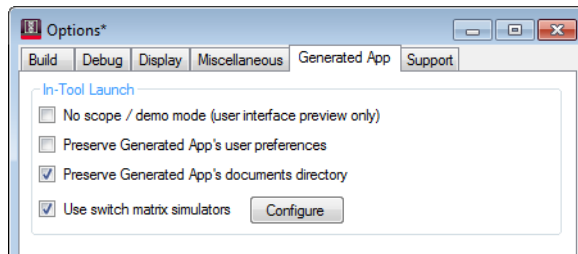
Probing – Lists all probing methods that can be used in this connection for the test point.

Scope Channel – Name or variable.

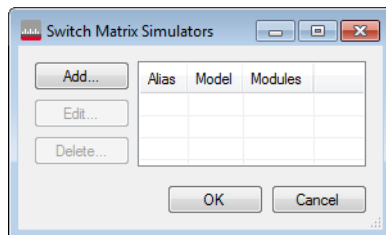
Simulators

If you do not have a physical switch instrument while developing the generated app, you can create and use simulators available for each supported model. Simulation can only be used for in-tool launch debug sessions.

To enable simulation, choose **Tools > Options...** In the Options dialog box's Generated App tab, check this user preference:



To create simulators, click the **Configure** button:

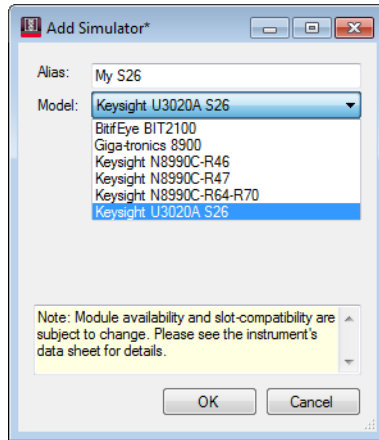


Add... – Create a single simulator. See "[Creating a Simulator](#)" on page 169.

Edit... – Edit the selected simulator.

Delete... – Delete the selected simulator.

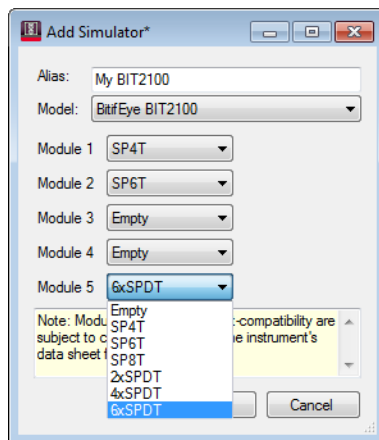
Creating a Simulator



Alias – Enter a nickname for this simulator. In the generated app, when you need to connect to a switch instrument, these alias names will be displayed.

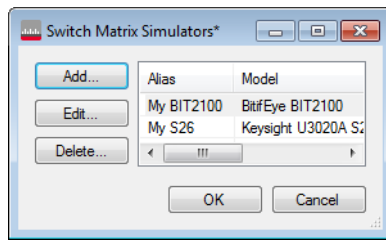
Model – Select a model to be simulated.

If the model has a variable number of modules, there will be options for selecting the switch contents:



Simulators are summarized in the preferences screen:

9 Adding Switch Matrix Control



The **Modules** column summarizes the contents using instrument-specific names.

10 Adding Run Actions on Events

Run actions are an optional feature. They let you execute steps between tests, on these types of events:

Application Mode:

- **App Shown** – these steps are executed when the generated app is launched and is completely initialized and displayed.
- **App Shown - Demo Mode** – same as the "App Shown" event, except these steps are executed when the generated app is launched in "no scope / demo mode".
- **Run Starts** – these steps are executed once at the start of the run (no matter how many trials are selected).
- **Any Trial Starts** – these steps are executed once at the start of each trial. For example, in a 2-trial run, the steps would execute twice.
- **Any Test Group Starts** – these steps are executed every time the test flow enters a new group, at any level, including the RootGroup.
- **Current Connection Changes** – these steps are executed once every time a test requires a different connection than the previously-completed test.
- **Group XX Starts** – these steps are executed once when starting a test from a different group than the previously completed test.
- **Any Test Starts** – these steps are executed once at the start of each test.
- **Any Test Ends** – these steps are executed once at the end of each test.
- **Any Trial Ends** – these steps are executed once at the end of each trial.
- **Run Ends** – these steps are executed once when the run ends.
- **App Exits** – these steps are executed when the user exits the generated app.

NOTE

When the generated app is launched in "no scope / demo mode", only the "App Shown - Demo Mode" event steps will execute.

Add-in Mode:

- **Group User Defined Starts** – these steps are executed once before the first selected add-in test begins.

- **Group XX Starts** – these steps are executed once when starting a test from a different group than the previously completed test.
- **Group User Defined Ends** – these steps are executed once after the last selected add-in test completes.

You are prevented from adding a call to a subroutine directly or indirectly containing an IntermediateValue step. This is because these steps add values to the results of specific tests while Run Events execute in-between tests.

For a description of how to add run actions, see "[Step 7: Add run actions \(optional\)](#)" on page 51.

11 Setting Miscellaneous Options

Add Online Help / 174

Add Logos/Images / 175

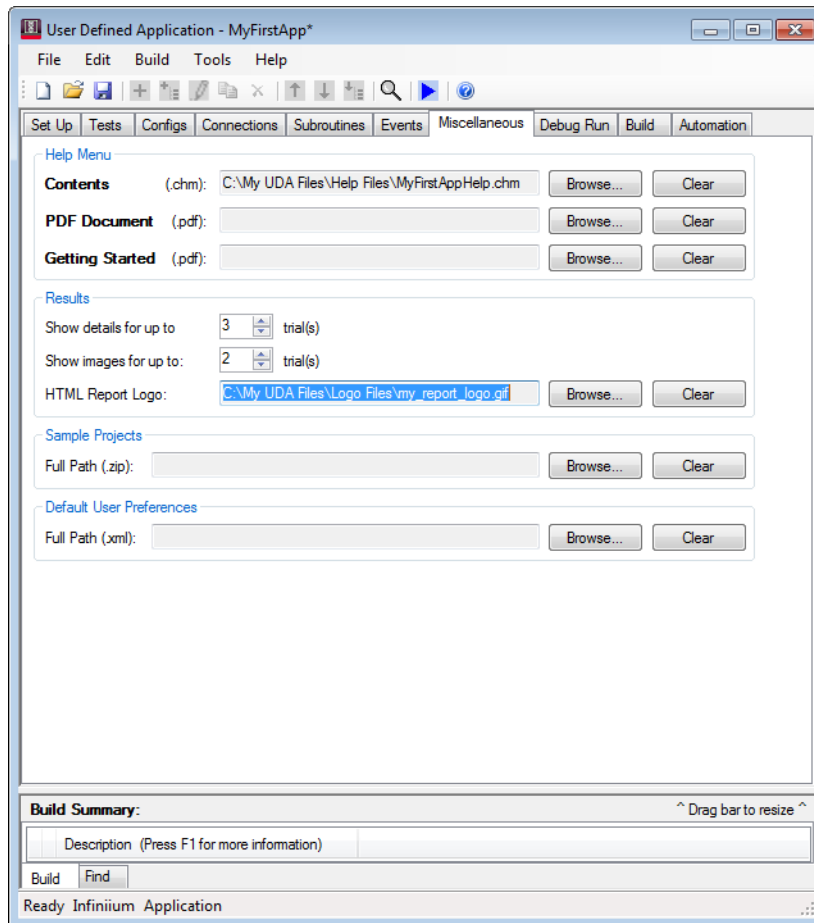
Specify Default User Preferences / 176

Add Online Help

The Miscellaneous tab lets you add an online help file to your generated application. This is a Windows HTML Help format .chm file and/or PDF format files.

You can develop HTML Help format .chm files using the HTML Help Workshop, available as a free download from Microsoft's web site, or you can use a number of other tools that are available for generating HTML Help files.

For a description of how to add an online help file, see "[Step 8: Add logos, images, help files \(optional\)](#)" on page 53.

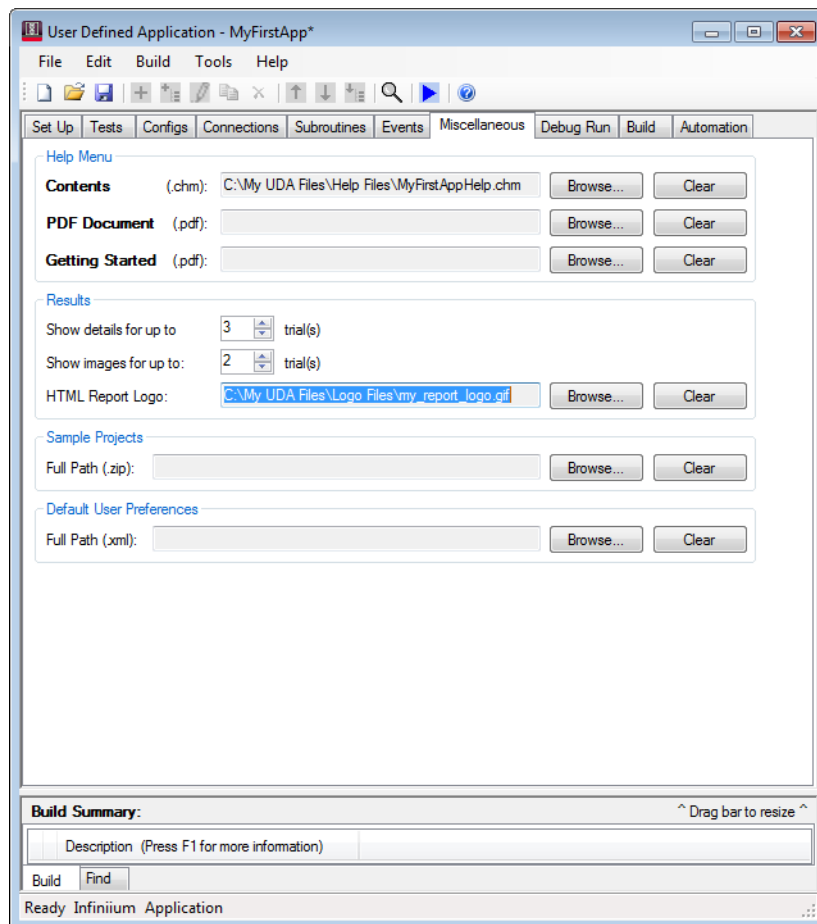


Add Logos/Images

The Miscellaneous tab lets you add a logo image that is displayed at the top of your generated application's HTML reports. This is typically a company logo.

- A logo image that is displayed at the top of your applications HTML reports. This is typically a company logo.

For a description of how to add logos and images, see "[Step 8: Add logos, images, help files \(optional\)](#)" on page 53.



Specify Default User Preferences

You can configure your generated app installer to deliver a different configuration to be used as the *initial* user preferences file to affect the *first launch* of the app only. After that, users are free to make changes using the generated app's **View > Preferences** window and those settings will then become the active user preferences.

For a description of how to specify default user preferences, see "[Step 10: Specify Default User Preferences \(optional\)](#)" on page 57.

12 Setting Debug Run Options

Oscilloscope SICL Address and Setup/Mask/Transfer Function File Paths / 178

External Instruments SICL Address Setting/Testing / 181

Oscilloscope SICL Address and Setup/Mask/Transfer Function File Paths

When you have enabled and added external instruments, you can set and test the external instrument SICL addresses in the Debug Run tab.

The following applies only to Infiniium real-time apps when connected to an oscilloscope running software versions < 4.10.

The SCPI command to load a setup file takes a path to the file. When you create a Load Setup step, you specify the path to the setup file. Whether the file resides on the oscilloscope or PC, this path is relative to the PC and might not be reusable in the SCPI command when you execute the step from a PC. The way the application generator manages this is to have you enter a directory where the oscilloscope will look for the setup file. The tool will take the .set's filename, combine it with the path entered in the Debug Run tab, and send that to the oscilloscope in the SCPI command.

NOTE

The generated app installer automatically modifies the step so it is not an issue for oscilloscope-based launches or for Debug Run and In-Tool launches when the Generator is running on the oscilloscope.

Therefore, when specifying the Setup File Path in the Debug Run tab, you have the following options:

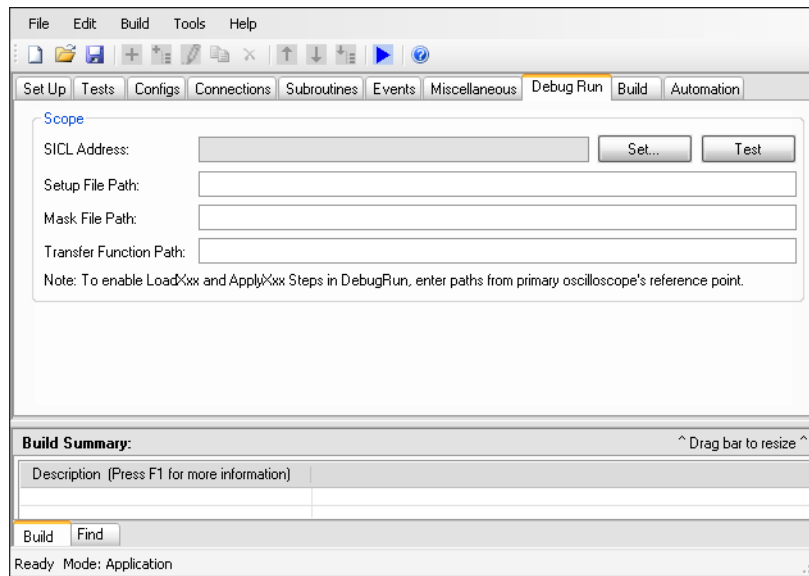
- **Option 1: Put the setup file on the oscilloscope** (in C:\UDA Inputs\Setups, for example):
 - a First, share the oscilloscope's directory where you placed the setup file (as "UDA Setups", for example).
 - b Next, in the Load Setup Step, have the source be the full network path to the shared directory on the oscilloscope (for example, "\\myscope\UDA Setups\SomeSetup.set").
 - c In the Debug Run tab, enter the local path on the oscilloscope to the directory where you placed the file (for example, C:\UDA Inputs\Setups).
 - d Thus, in this example, the SCPI command that gets sent to the oscilloscope would be :DISK:LOAD 'C:\UDA Inputs\Setups\SomeSetup.set'.

- **Option 2: Put the setup file on the PC** (in C:\MyUDAFiles\Setups, for example):
 - a First, share the PC's directory where you placed the setup file (as "UDA Setups", for example).
 - b Next, in the Load Setup Step, have the source be the local path to the file (for example, "C:\MyUDAFiles\Setups\SomeSetup.set").
 - c In the Debug Run tab, enter the network path to the share on the PC (for example, \\mypc\UDA Setups).
 - d Thus, in this example, the SCPI command that gets sent to the oscilloscope would be :DISK:LOAD '\\mypc\UDA Setups\SomeSetup.set'.
- **Option 3: Put the setup file on a shared network drive visible to both the scope and the PC:**
 - a In the Load Setup Step, have the source be the full network path to the shared network drive (for example, "\\myserver\UDA Setups\SomeSetup.set").
 - b In the Debug Run tab, enter the network path to the share (for example, \\myserver\Uda Setups).
 - c Thus, in this example, the SCPI command that gets sent to the oscilloscope would be :DISK:LOAD '\\myserver\Uda Setups\SomeSetup.set'.

For the Mask File Path, if you have a 90000A or 9000A Series oscilloscopes running software 2.10 through 4.10, you will have the same options available as the setup files described above (file path located on the oscilloscope or network path). For oscilloscopes running an Infiniium software version less than 2.10, you must use a path located on the oscilloscope when defining the Mask File Path.

For Infiniium software versions 4.10 and higher, these text entry fields are not provided because setup, mask, and transfer function files are automatically transferred to the oscilloscope and loaded during a Debug Run.

12 Setting Debug Run Options

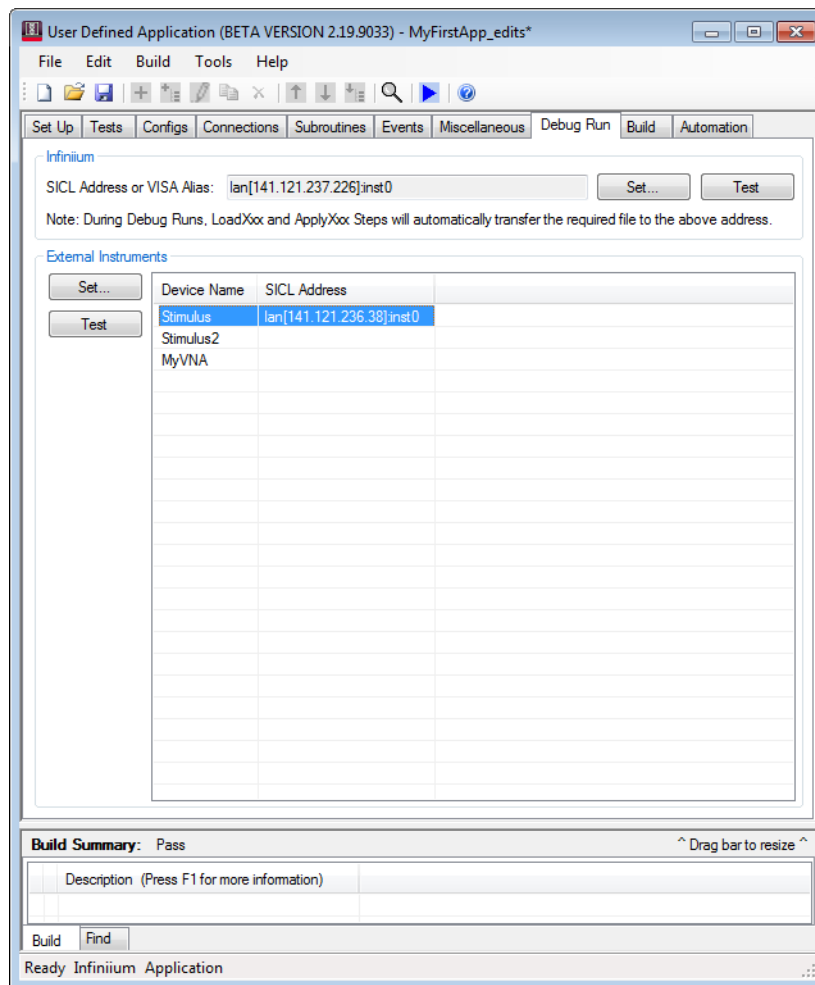


External Instruments SICL Address Setting/Testing

When you have enabled and added external instruments, you can set and test the external instrument SICL addresses in the Debug Run tab.

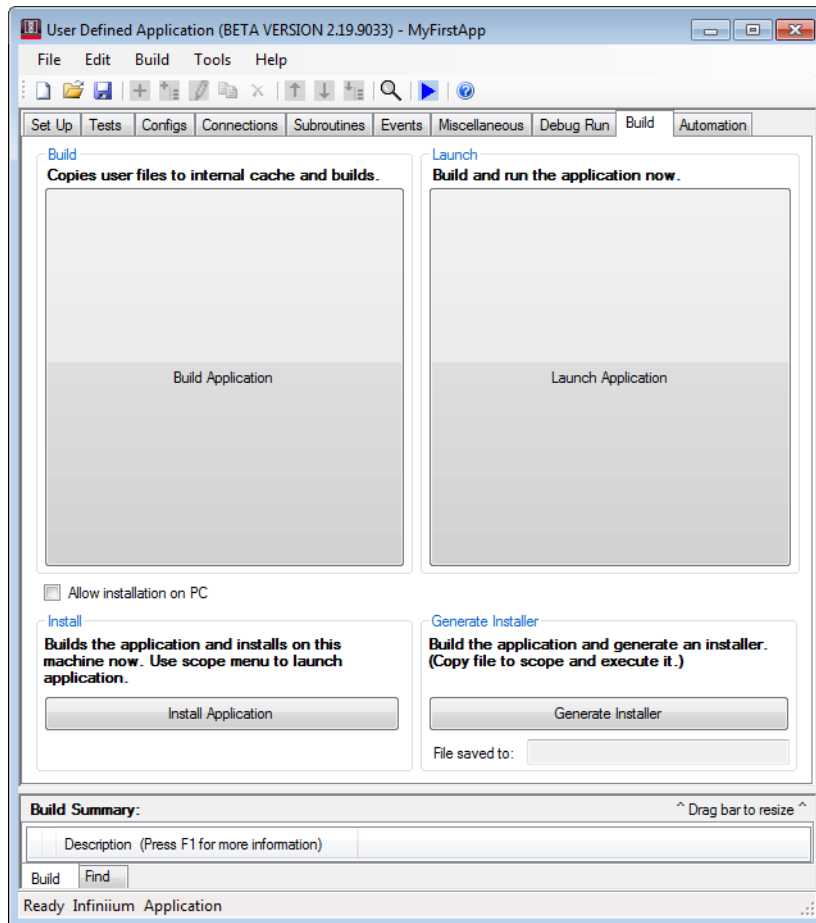
To set or change an external instrument's SICL address, select the device, then click **Set...**


To test an external instrument's SICL address, select the device, then click **Test**.



13 Generating the Application Software Installer

The application generator's Build tab has these options:



To perform this build action:	Click this:	Or choose this from the main menu:	Or use this keyboard shortcut:	Or click this toolbar button:
To copy user files to the internal cache and build (makes sure all files can be located)	Build Application	Build > Build application	Ctrl+B	
To build and run the application (tests the application before installing)	Launch Application	Build > Launch application	Ctrl+L	
To build the application and generate an installer	Generate Installer	Build > Generate installer		
To build the application and install	Install Application	Build > Install application		
To allow installers to run on a PC	Allow installation on PC	N/A		

When you generate an installer, the location of the executable appears in the **File saved to** field.

- For applications, you can copy this file to your oscilloscope and execute it.
- For Add-Ins, copy the file to the oscilloscope and execute it from within a compliance app (see "**Step 14b: Installing/Uninstalling and Running an Add-In**" on page 61).

For Infiniium real-time applications:

- When the installation is finished, restart the Infiniium oscilloscope user interface software, and you will find a new menu entry: **Analyze > Automated Test Apps > User Defined > (Application Prefix) > (Application Name)**

Remember the Infiniium oscilloscope requires the Keysight D9010UDAA User Defined Application (UDA) license in order to run your applications. (You can install and launch your application without a license, but you cannot run tests.)

For FlexDCA applications:

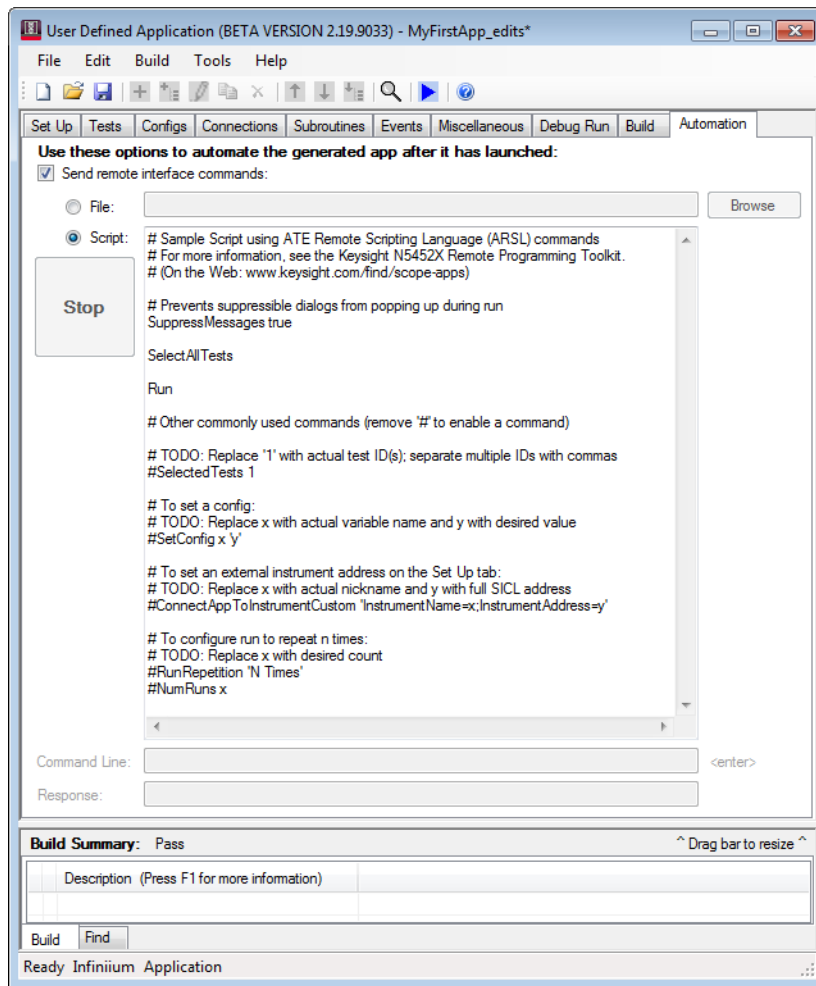
- When the installation is finished, you will find a new menu entry in FlexDCA: **Apps > Automated Test Apps > User Defined > (Application Prefix) > (Application Name)**

Remember FlexDCA requires the Keysight D9010UDAA User Defined Application (UDA) license in order to run your applications. (You can install and launch your application without a license, but you cannot run tests.)

14 Automating the Generated Application

When you launch the generated application from within the application generator, you may choose to manually interact with it through its graphical user interface to select its tests and run it. Alternatively, you can use the 'Automation' tab in the application generator to automate the generated application.

The commands used for automation are written in the ATE Remote Scripting Language (ARSL), described in the Keysight N5452A Remote Toolkit.



To automate the generated application:

- 1 Click 'Send remote interface commands'.
- 2 Specify the source commands to be sent.
 - a Check File: Select this to specify an ARSL file to execute. This is simply a text file ending in .arsl that contains ARSL commands (one per line).
 - b Check Script: Select this to specify an ARSL script to execute. The sample script contains actual ARSL commands although some have TODO actions required to make them executable.

NOTE

ARSL comments start with a '#' character in the first column.

NOTE

If your script includes the "Exit" command, all steps after it will be skipped.

- 3 Launch the generated application. When it is ready, the application generator will automatically send the commands you specified above to the application.

15 Working with Variables and Values

Managing Internal Variables / 190

Managing Math Libraries / 192

Managing Text Libraries / 194

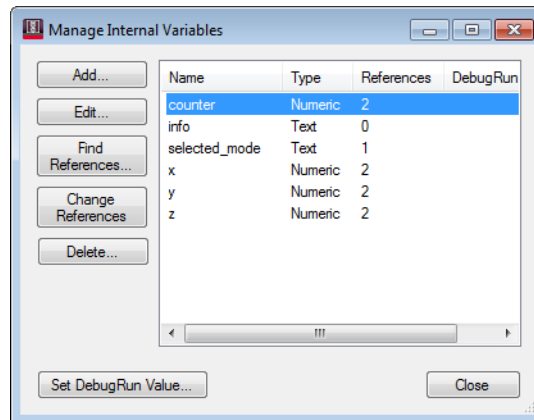
Managing Reserved Variables / 196

Getting Values / 198

Variable Substitution / 209

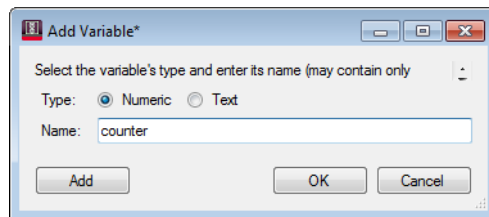
Managing Internal Variables

- 1 From the application generator's main menu, choose **Tools > Manage > Internal Variables...**
- 2 In the Manage Variables dialog box:



To add variables:

- a Click **Add...**
- b In the Add Variable dialog box, select the type of values the variable will hold, and enter the variable name.



You can quickly enter multiple variables by clicking **Add** after typing in each variable name. On the **Tools > Options > Display** tab, you can also configure the **<Enter>** key to behave like **Add**. The **Cancel** button will undo additions made using **Add**.

- c Click **OK**.

To edit a variable name:

- a Select the variable whose name you want to edit.
- b Click **Edit**.
- c In the Edit Variable dialog box, enter the new variable name.

If the variable is referenced by any test or event steps, those references will be updated with the new name.

d Click **OK**.

To remove variables:

a Select the variable you want to remove.

b Click **Remove**.

c In the confirmation dialog box, click **OK**.

To set variables to values for use in a Debug Run:

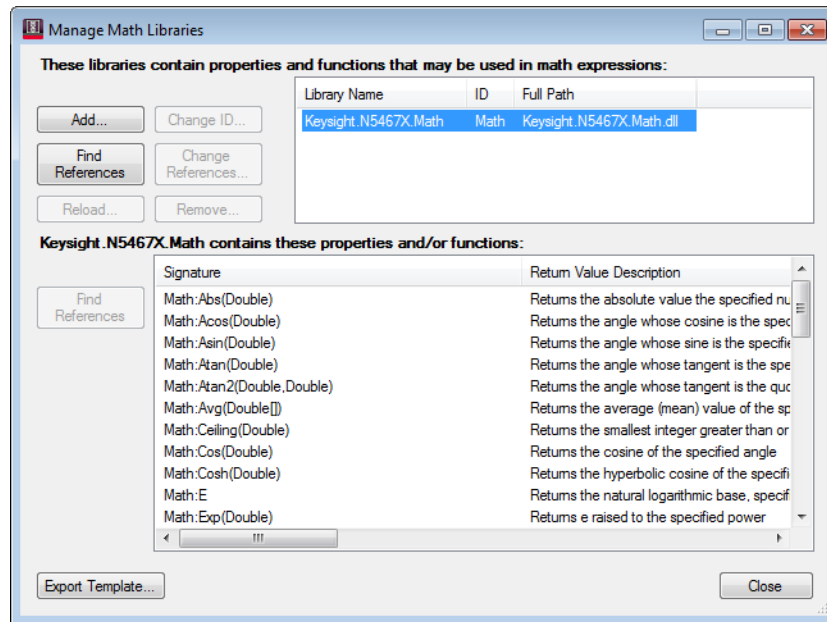
a (optionally) Select the variable you want to set.

b Click **Set DebugRun Value...**

In the Set Variables For Debug Run dialog box, you can set the variable values used in a Debug Run. See "[Debug Run Button](#)" on page 142.

Managing Math Libraries

- 1 From the application generator's main menu, choose **Tools > Manage > Math Libraries....**
- 2 In the Manage Math Libraries dialog box:



The upper half of the dialog box displays general information about the currently loaded libraries.

- Add:** Loads a UDA-compatible math library to enable steps to reference its functions and properties. You will be prompted to select a prefix for this library. This prefix is how steps will specify which library a particular function call is to be made from.
- Change ID:** Change the prefix of the selected library, updating all references in steps.
- Find References:** Find all references in steps to any function or property in the selected library.
- Change References:** Change all references in steps to any function or property in the selected library to those of another library.

CAUTION

For each existing reference, the new library should have a corresponding same-named property or a function with the same signature. Otherwise, a modified step will end up referring to a non-existent property or function.

After using Change References, build the project to ensure the new library was adequately substituted for the old.

- e **Reload:** Unloads the selected library and prompts you to reload it. Convenient when you are developing the math library and need UDA to unlock it so it can be recompiled.
- f **Remove:** Unloads the selected library and removes it from the list.

The lower half of the dialog box displays detailed information about the selected library.

- a **Find References:** Finds all references in steps to the selected function or property.
- b **Export Template:** Exports a sample C# source file that can be used to create a custom UDA-compatible math library. Your project will need to have a reference to a UDA-provided assembly in order to compile. To make this reference:
 - i Install D9010UDAA version 1.30 or newer on the PC you are developing the custom math libraries on. This will install the required UDA assembly in the Global Assembly Cache.
 - ii If using Microsoft Visual Studio, you can find it in the .NET tab of the Add References dialog box ("Keysight.N5467A.MathLibraryUtilities").

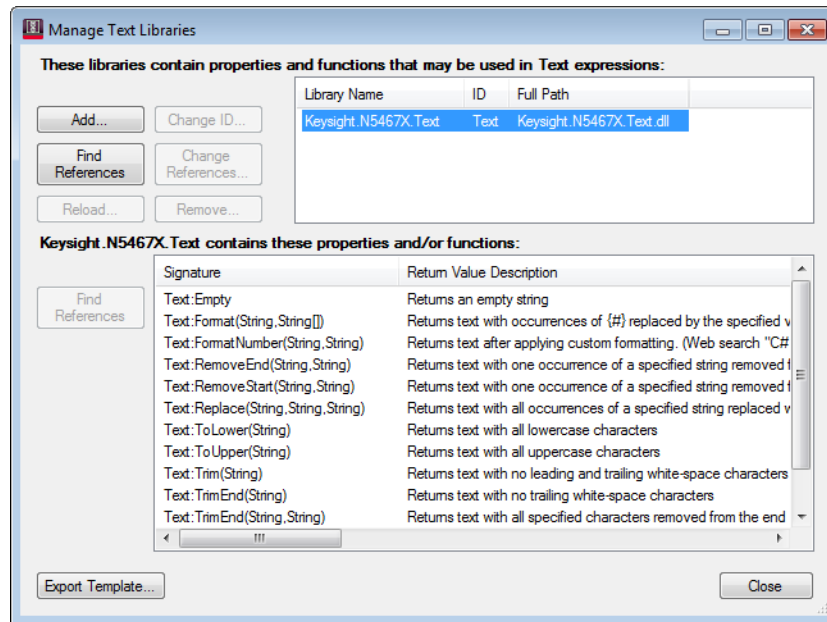
NOTE

If you have custom libraries compiled against the 1.22 or older (non-GAC) version of Keysight.N5467A.MathLibraryUtilities.dll, you must recompile them using the 1.30 version in order to use them in 1.30 or newer UDA projects.

See Also · ["To get a value from a Math Expression source"](#) on page 201

Managing Text Libraries

- 1 From the application generator's main menu, choose **Tools > Manage > Text Libraries....**
- 2 In the Manage Text Libraries dialog box:



The upper half of the dialog box displays general information about the currently loaded libraries.

- a **Add:** Loads a UDA-compatible text library to enable steps to reference its functions and properties. You will be prompted to select a prefix for this library. This prefix is how steps will specify which library a particular function call is to be made from.
- b **Change ID:** Change the prefix of the selected library, updating all references in steps.
- c **Find References:** Find all references in steps to any function or property in the selected library.
- d **Change References:** Change all references in steps to any function or property in the selected library to those of another library.

CAUTION

For each existing reference, the new library should have a corresponding same-named property or a function with the same signature. Otherwise, a modified step will end up referring to a non-existent property or function.

After using Change References, build the project to ensure the new library was adequately substituted for the old.

- e Reload:** Unloads the selected library and prompts you to reload it. Convenient when you are developing the text library and need UDA to unlock it so it can be recompiled.
- f Remove:** Unloads the selected library and removes it from the list.

The lower half of the dialog box displays detailed information about the selected library.

- a Find References:** Finds all references in steps to the selected function or property.
- b Export Template:** Exports a sample C# source file that can be used to create a custom UDA-compatible text library. Your project will need to have a reference to a UDA-provided assembly in order to compile. To make this reference:
 - i** Install D9010UDAA version 1.60 or newer on the PC you are developing the custom text libraries on. This will install the required UDA assembly in the Global Assembly Cache.
 - ii** If using Microsoft Visual Studio, you can find it in the .NET tab of the Add References dialog box ("Keysight.N5467A.TextLibraryUtilities").

See Also · ["To get a value from a Text source"](#) on page 203

Managing Reserved Variables

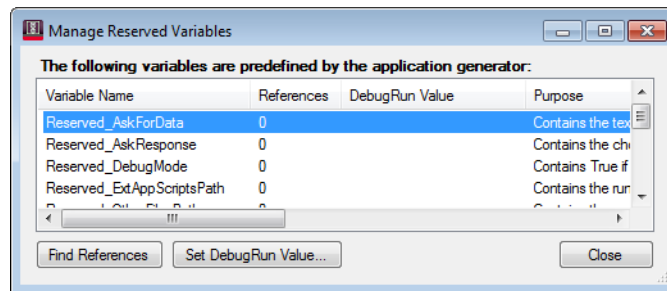
Several predefined variables are included with the application generator.

Table 1 Reserved Variables

Variable	Description
Reserved_AskForData	Contains the user's entry from the most recent Ask For Data message (if the Display Message step was configured to use this variable).
Reserved_AskResponse	Contains the user's response to the most recent Ask message.
Reserved_ChannelCount	Contains the number of channels available in the connected scope.
Reserved_DebugMode	Contains True if the generated app is in Debug Mode; otherwise, this variable contains False.
Reserved_ExtAppScriptsPath	Contains the path to the directory where any additional external application scripts you designated (in Tools > Manage > External App Scripts) to be installed will be placed.
Reserved_OtherFilesPath	Contains the path to the directory where any "miscellaneous files" you designated to be installed to the Standard Location "Application" will be placed. For Debug Run and In-Tool Launch, this location is an internal cache. For oscilloscope Launch, this location is inside the generated app's installation directory.
Reserved_ProjectAppPath	If your application creates files at runtime, placing them in the directory specified by the reserved variable Reserved_ProjectAppPath will ensure they get saved with the GeneratedApp projects.
Reserved_OperationComplete	After execution of a SCPI command with Query Operation Complete enabled, this variable contains the value returned by the *OPC? query.
Reserved_ScpiError	After execution of a Get SCPI Error step, this variable contains the text of an error retrieved from an instrument's error queue.
Reserved_InfiniiumAddress	Contains the current SICL address or VISA alias of the Infiniium application controlling the primary oscilloscope.
Reserved_StimulusAddress	Contains the external instrument address entered by the generated app's user; if no address is entered, this variable contains '<none>'.
Reserved_xxPresent	Set to "True" if the Scope Option Requirement optional alternative named "xx" is present on the oscilloscope.
Reserved_yyAddress	Contains the address entered by the generated app's user for external instrument named "yy".

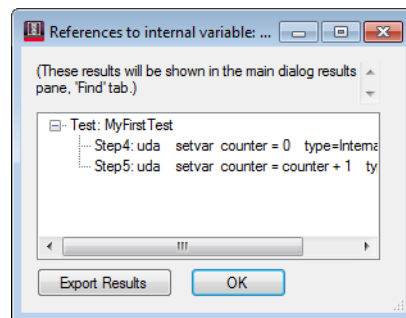
To manage reserved variables:

- 1 From the application generator's main menu, choose **Tools > Manage > Reserved Variables...**
- 2 In the Manage Reserved Variables dialog box, you can:



- **Find References** – To display where the variable is being used:
 - i Select the variable whose usage you want to display.
 - ii Click **Find References**.

The References to Internal Variable dialog box shows where the variable is used. You can export these results by clicking on the **Export Results** button.



- iii Click **Close** to close the Usage dialog box.

NOTE

Usage results are copied to the main dialog box in the lower pane Find tab.

- **Set DebugRun Value...** – Opens the Set Variables For Debug Run dialog box where you can set the variable values used in a Debug Run. See "**Debug Run Button**" on page 142.

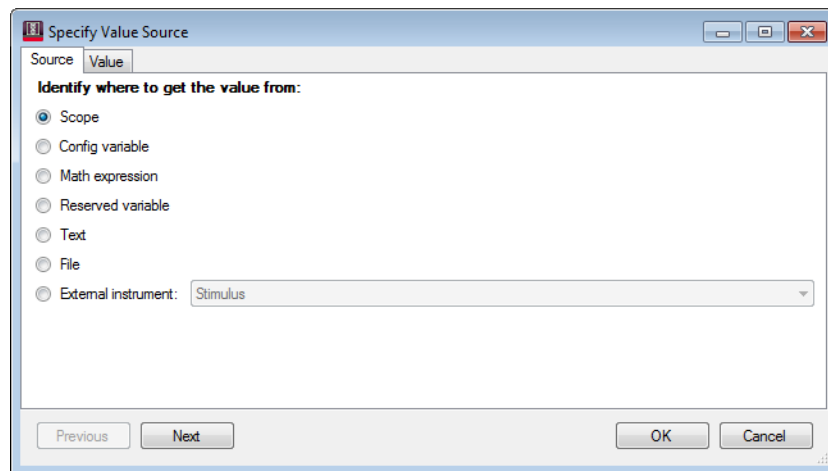
Getting Values

- "To get a value from a Scope source" on page 198
- "To get a value from a Config Variable source" on page 200
- "To get a value from a Math Expression source" on page 201
- "To get a value from a Text source" on page 203
- "To get a value from a File source" on page 206
- "To get a value from an External Instrument source" on page 207

To get a value from a Scope source

When assigning values to a variable or reporting an intermediate or final value, you can get the value from an oscilloscope SCPI query.

- 1 In the Specify Value Source dialog box, select the Source tab; then, select the **Scope** option.



- 2 Select the Value tab; then:
 - a Enter the SCPI query. At any time, type ALT+I to insert a variable reference.
 - b Change the **Timeout** value if desired.
 - c If the query returns multiple values, check **Parse multiple values**; then, enter the value separator character. If you enter more than one character, the value text will be split every time any of the characters is found.

Special characters can be parsed using the escape (\) character, including:

Enter this	To parse this
\n	end of line character
\t	tab character
\\	backspace character

To use only one of the parsed values, select the **Use Value** radio and enter the index of the value to use. You may use a variable to specify the index. For Write File Steps, you may alternatively select the **Use all values** radio to cause each value to be written to a separate line in the output file.

TIP

To place multiple values into multiple variables without having to repeat the query (for performance reasons), try this test step sequence:

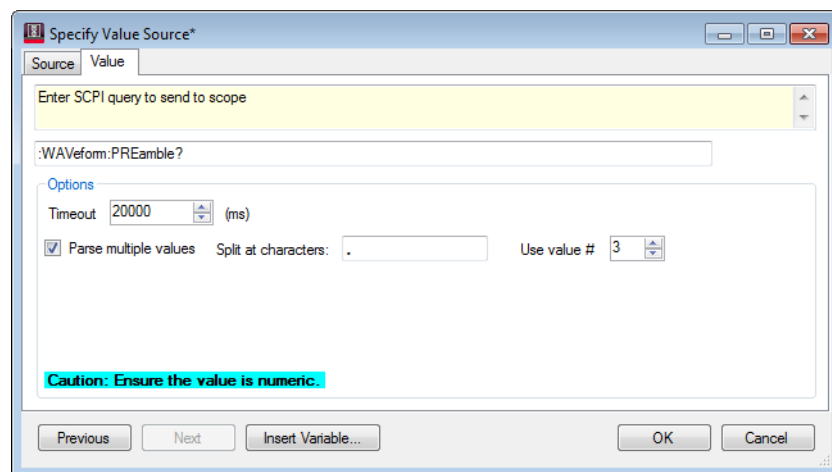
Step n: (Set Variable step) Write the text returned by the query to a text variable.

Step n+1: (Set Variable step) Parse the first value found in the text variable into another variable.

Step n+2: (Set Variable step) Parse the second value found in the text variable into another variable.

etc.

After splitting a line, the parser removes leading and trailing whitespaces.

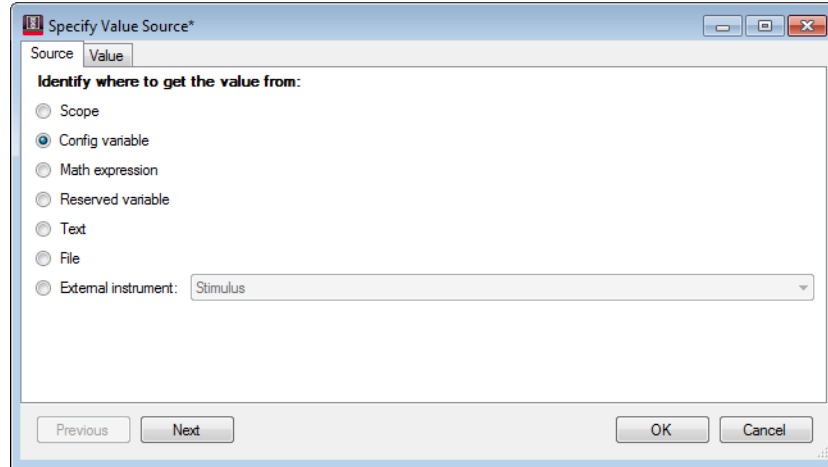


3 Click **OK**.

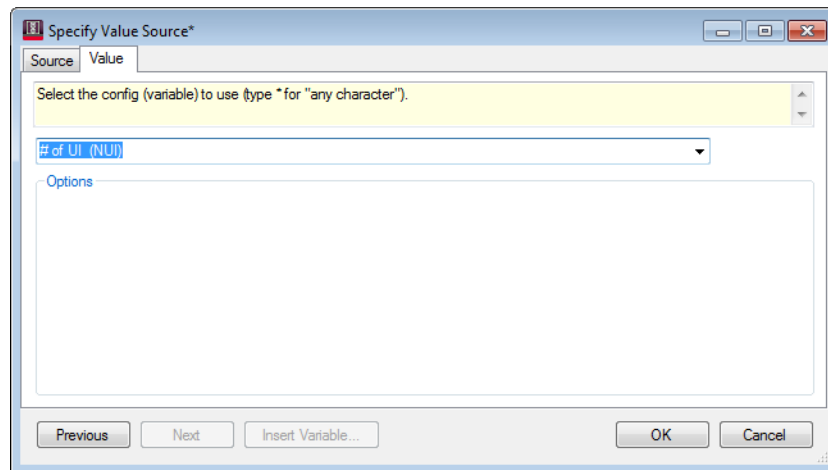
To get a value from a Config Variable source

When assigning values to a variable or reporting an intermediate or final value, you can get the value from a Config variable set in the user interface.

- 1 In the Specify Value Source dialog box, select the Source tab; then, select the **Config variable** option.



- 2 Select the Value tab; then select the Config variable.

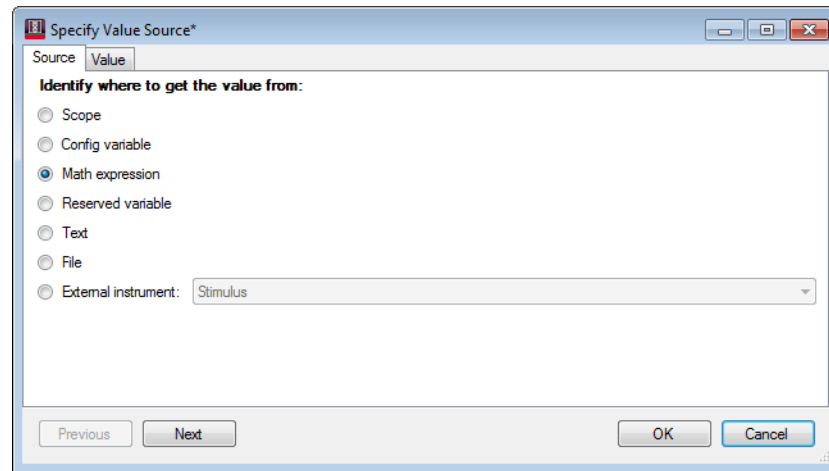


- 3 Click **OK**.

To get a value from a Math Expression source

When assigning values to a variable or reporting an intermediate or final value, you can get the value from a math expression.

- 1 In the Specify Value Source dialog box, select the Source tab; then, select the **Math expression** option.



- 2 Select the Value tab; then, enter an expression using constants, internal variables, and/or library properties and methods. At any time, type ALT+I to insert a variable reference.

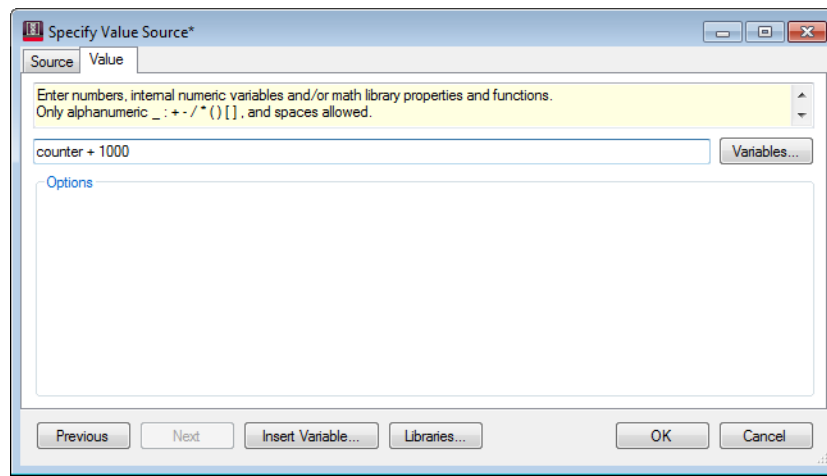
You can click **Variables...** to see the internal variables that are defined.

You can click **Libraries...** to access external libraries or the core math library. The field in the Specify Value Source Value tab can include references to properties and functions in those expressions. These properties and functions must be defined in an external library (dll) and loaded into the tool (**Tools > Manage > Math Libraries...**). The list of libraries is project-specific. A core math library is included with the application generator. To handle name collisions across multiple libraries, all references to their properties and functions must be preceded by a unique user-selected prefix for that library, chosen at the time the library is loaded. The core library is automatically assigned the prefix "Math".

For example, to access the value Pi defined in the core library, an expression would be: `2*Math:Pi`.

Properties and functions are case-sensitive. You can create custom math libraries and load them as well. However, there are conventions you must follow in order to make this work. The easiest way to discover these rules is to use the Manage Math Libraries dialog box's "Export Template" feature. This does two things: creates a heavily-documented sample source file that can immediately be compiled and copies a DLL that must be referenced by your DLL. The compiler must support at least NET 2.0.

When you call library functions, each of its arguments may only be a single number, internal variable, and/or library property, or a single array containing these. For example, `Math:WorstForRange(1,Math:Pi,[2,MyIntervalVar,Math:E])` has three arguments: the number 1, math library property Pi, and an array containing three values (the number 2, internal variable MyIntervalVar, and math library property E). If you need to pass a multiple-term argument, such as "x+y", first store its result in a temporary internal variable using a SetVariable step and then pass the temporary variable as the argument in the function call.

**NOTE**

Math expressions containing literal floating point values cannot start with a decimal (for example, use "0.123", not ".123").

NOTE

Whenever you enter numbers using engineering or scientific notation, for example 123E+09, always include the sign character (whether it is a "+" or a "-"). Also, never start a number with a decimal; use 0.1 instead of .1 to specify 1E-01.

Do not use variable substitution syntax, for example, `%Var:x%`, in math expressions.

You should not place Config variables in math expressions, even if the Config variable value is numeric. You can, however, set an internal variable equal to a Config variable value and then use the internal variable in the math expression.

3 Click OK.**Math Expression Operators**

You can use these operators in math expressions:

- + (plus).
- - (minus).
- * (multiply).

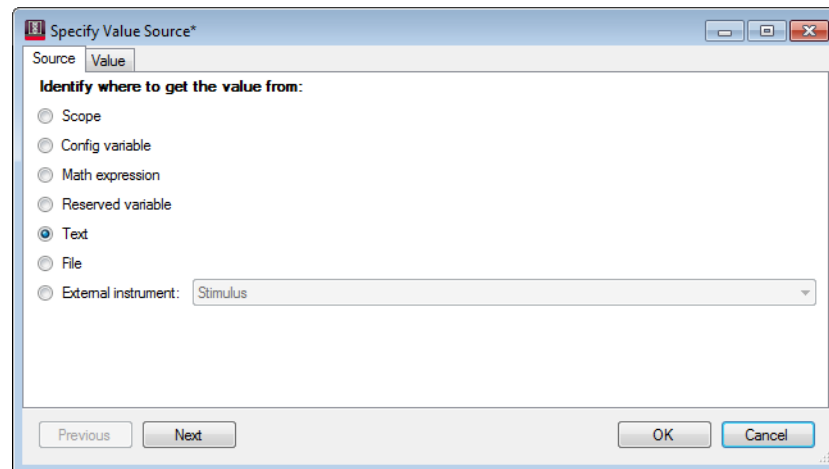
- / (divide).
- () – parentheses for grouping.
- [] – square brackets for holding arrays.

See Also • **"Managing Math Libraries"** on page 192

To get a value from a Text source

You can set a user-editable config to arbitrary text.

- 1 In the Specify Value Source dialog box, select the Source tab; then, select the **Text** option.



- 2 Select the Value tab; then:
 - a Enter an expression using quoted text, variables of any type, and/or library properties and methods. At any time, type ALT+I to insert a variable reference.

You can click **Variables...** to see the internal variables that are defined.

You can click **Libraries...** to access external libraries or the core text library. The field in the Specify Value Source Value tab can include references to properties and functions in those expressions. These properties and functions must be defined in an external library (dll) and loaded into the tool (**Tools > Manage > Text Libraries...**). The list of libraries is project-specific. A core text library is included with the application generator. To handle name collisions across multiple libraries, all references to their properties and functions must be preceded by a unique user-selected prefix for that library, chosen at the time the library is loaded. The core library is automatically assigned the prefix "Text".

For example, to get the contents of a text variable and convert them to lower case, an expression would be: `Text:ToLower(MyTextVar)`.

Properties and functions are case-sensitive. You can create custom text libraries and load them as well. However, there are conventions you must follow in order to make this work. The easiest way to discover these rules is to use the Manage Text Libraries dialog box's "Export Template" feature. This does two things: (1) creates a heavily-documented sample source file that can immediately be compiled and (2) copies a DLL that must be referenced by your DLL. The compiler must support at least .NET 2.0.

When you call library functions, each of its arguments may only be quoted text, a variable of any type, and/or library property, or a single array containing these. For example, `Text:Format("{0} {1}", ["Hello", MyTextVar])` has two arguments: the text `"{0} {1}"` and an array containing two values (the text `"Hello"` and internal variable `MyTextVar`). If you need to pass a multiple-term argument, such as `"abc" + MyTextVar`, first store its result in a temporary internal variable using a `SetVariable` step and then pass the temporary variable as the argument in the function call.

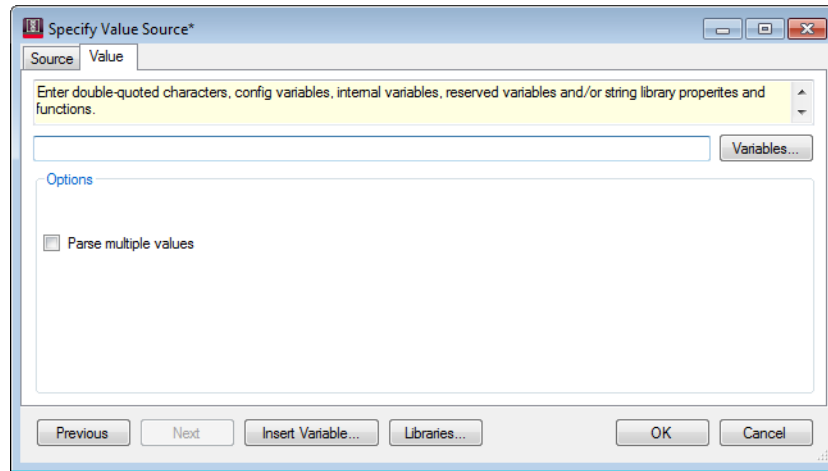
Do not use variable substitution syntax, for example, `%Var:x%`, in text expressions.

- b** If the text source contains multiple values, check **Parse multiple values**; then, enter the value separator character and the index of the value to use. If you enter more than one character, the value text will be split every time any of the characters is found. Special characters can be parsed using the escape (`\`) character, including:

Enter this	To parse this
<code>\n</code>	▪ end of line character
<code>\t</code>	▪ tab character
<code>\\</code>	▪ backspace character

To use only one of the parsed values, select the **Use Value** radio and enter the index of the value to use. You may use a variable to specify the index. For Write File Steps, you may alternatively select the **Use all values** radio to cause each value to be written to a separate line in the output file.

After splitting a line, the parser removes leading and trailing whitespaces.



3 Click **OK**.

Text Expression Operators

You can use these operators in text expressions:

- " (double quote) to surround quoted text.
- \ (backslash) to embed a double quote or backslash in quoted text.
- + (plus) to connect terms such as quoted text and variables.
- () – parentheses for function calls.
- [] – square brackets for holding arrays.

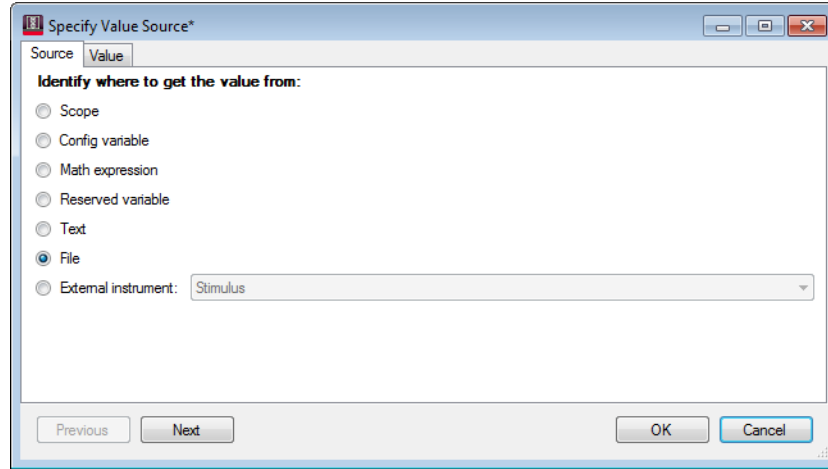
See Also

- **"Managing Text Libraries"** on page 194

To get a value from a File source

When assigning values to a variable or reporting an intermediate value, you can get the value from a file.

- 1 In the Specify Value Source dialog box, select the Source tab; then, select the **File** option.

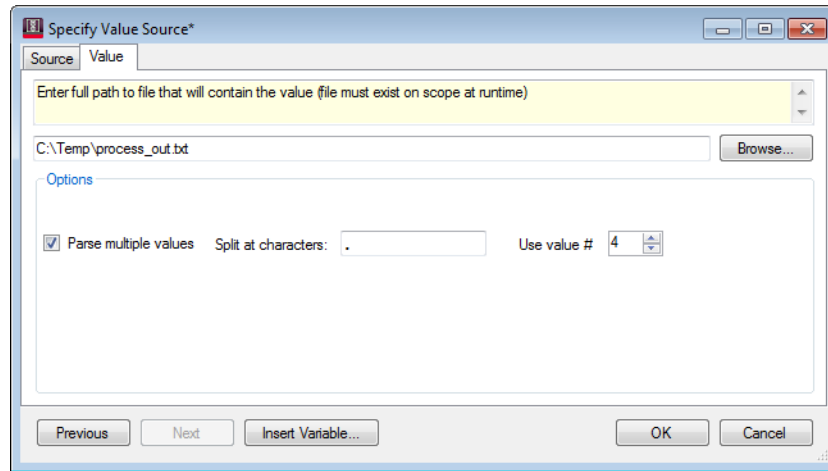


- 2 Select the Value tab; then:
 - a Enter the full path of the file that will contain the value.
The file must exist on the oscilloscope at run time.
 - b If the file contains multiple values, check **Parse multiple values**; then, enter the value separator character and the index of the value to use. If you enter more than one character, the value text will be split every time any of the characters is found. Special characters can be parsed using the escape (\) character, including:

Enter this	To parse this
\n	▪ end of line character
\t	▪ tab character
\\	▪ backspace character

To use only one of the parsed values, select the **Use Value** radio and enter the index of the value to use. You may use a variable to specify the index. For Write File Steps, you may alternatively select the **Use all values** radio to cause each value to be written to a separate line in the output file.

After splitting a line, the parser removes leading and trailing whitespaces.

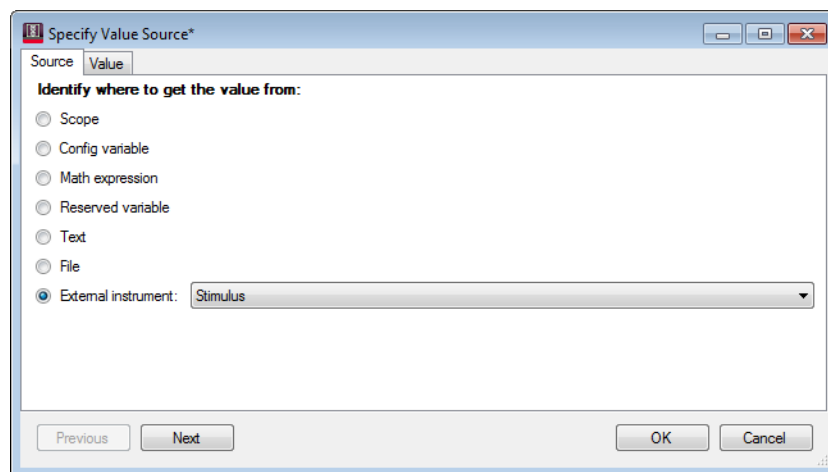


- 3 Click **OK**.

To get a value from an External Instrument source

When assigning values to a variable or reporting an intermediate value, you can get the value from an external instrument SCPI query.

- 1 In the Specify Value Source dialog box, select the Source tab; then:
 - a Select the **External instrument** option.
 - b Select the external instrument from the drop-down list.



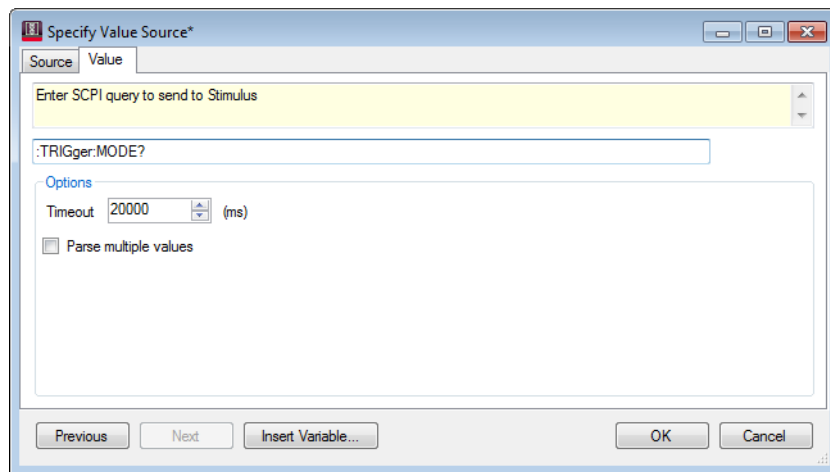
- 2 Select the Value tab; then:
 - a Enter the SCPI query. At any time, type ALT+I to insert a variable reference.
 - b Change the **Timeout** value if desired.
 - c If the query returns multiple values, check **Parse multiple values**; then, enter the value separator character and the index of the value to use. If you enter

more than one character, the value text will be split every time any of the characters is found. Special characters can be parsed using the escape (\) character, including:

Enter this	To parse this
\n	end of line character
\t	tab character
\\	backspace character

To use only one of the parsed values, select the **Use Value** radio and enter the index of the value to use. You may use a variable to specify the index. For Write File Steps, you may alternatively select the **Use all values** radio to cause each value to be written to a separate line in the output file.

After splitting a line, the parser removes leading and trailing whitespaces.



3 Click **OK**.

Variable Substitution

You can use %VAR:VariableName% substitution to include Config, Internal, and Reserved Internal variables in the following properties:

Element	Substitution Allows In	Supported Variable Types
Abort Step	<ul style="list-style-type: none"> ▪ Message 	Config, Internal, Reserved
Apply Transfer Function	<ul style="list-style-type: none"> ▪ Full Path 	Config, Internal, Reserved
Connection Definition	<ul style="list-style-type: none"> ▪ Instructions 	Config
Display Message Step	<ul style="list-style-type: none"> ▪ Message 	Config, Internal, Reserved
Final Result Step	<ul style="list-style-type: none"> ▪ Image file full path 	Config, Internal, Reserved
	<ul style="list-style-type: none"> ▪ SCPI query 	Config, Internal, Reserved
	<ul style="list-style-type: none"> ▪ File full path 	Config, Internal, Reserved
If Step	<ul style="list-style-type: none"> ▪ Predicate term text value 	Config, Internal, Reserved
Intermediate Value Step	<ul style="list-style-type: none"> ▪ SCPI query ▪ Text value ▪ File value 	Config, Internal, Reserved
Launch External Application Step	<ul style="list-style-type: none"> ▪ Program full path ▪ Args ▪ Script full path ▪ Working directory ▪ Output 	Config, Internal, Reserved
Load Scope Setup File	<ul style="list-style-type: none"> ▪ Full Path 	Config, Internal, Reserved

Element	Substitution Allows In	Supported Variable Types
Load Scope Mask File	<ul style="list-style-type: none"> Full Path 	Config, Internal, Reserved
SCPI Command Files	<ul style="list-style-type: none"> Commands contained in file 	Config, Internal, Reserved
Single SCPI Command Step	<ul style="list-style-type: none"> Command 	Config, Internal, Reserved
Use Console Application	<ul style="list-style-type: none"> Args Command Success text Failure text 	Config, Internal, Reserved
Verify Condition Step	<ul style="list-style-type: none"> SCPI query Violation message 	Config, Internal, Reserved
While Step	<ul style="list-style-type: none"> Predicate term text value 	Config, Internal, Reserved
Write File Step	<ul style="list-style-type: none"> Output Full Path 	Config, Internal, Reserved

On the screens where you enter values for these properties, you will have access to a variable inserter utility, which gives you easy access to a list of current variable names and automatically constructs the "%var" syntax when it is required.

NOTE

The "VAR" prefix is not case sensitive, but the variable name is.

16 Managing Other Resources

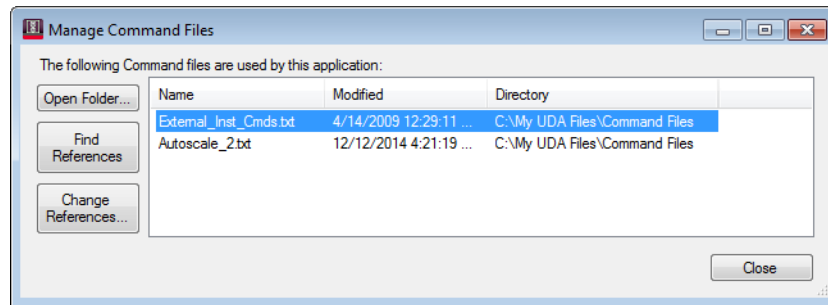
Managing Command Files / 212
Managing Connection Diagrams (HTML) / 214
Managing Connection Diagrams (Images) / 216
Managing Console Applications / 218
Managing External Application Scripts / 221
Managing Other Files / 223
Managing Repository / 226
Managing Scope Setup Files / 228
Managing Scope Mask Files / 230
Managing Transfer Function Files / 231
Managing Generated App Compatibility / 232
Scope Option Requirements / 234

- See Also
- ["Managing Internal Variables"](#) on page 190
 - ["Managing Math Libraries"](#) on page 192
 - ["Managing Reserved Variables"](#) on page 196

Managing Command Files

The Manage Command Files dialog box shows you where command files are located, shows you where they are used, and lets you replace test references to command files.

- 1 From the application generator's main menu, choose **Tools > Manage > Command Files...**
- 2 In the Manage Command Files dialog box:



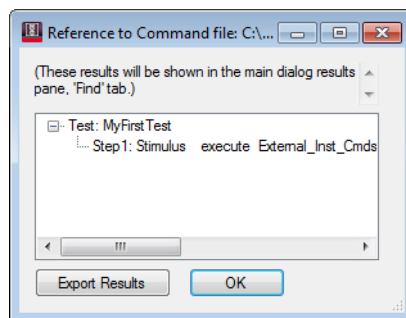
To open a folder where a command file is located:

- a Select the command file whose location you want to open a folder at.
- b Click **Open Folder...**
- c In the Explorer window, you can quickly edit the command file, copy it, etc.

To display command file usage:

- a Select the command file whose usage you want to display.
- b Click **Find References**.

The Usage dialog box shows the tests and steps where the command file is used. You can export the results to a file using the **Export Results** button.



- c Click **Close** to close the Usage dialog box.

NOTE

Usage results are copied to the main dialog box in the lower pane Find tab.

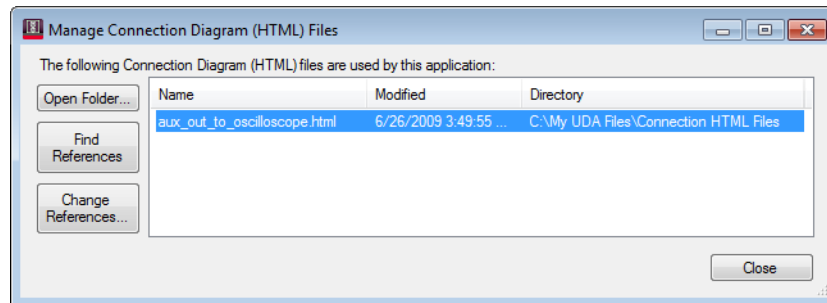
To replace a command file:

- a** Select the command file you want to replace.
- b** Click **Change References...**
- c** In the file browser dialog box, select the new command file; then, click **OK**.

Managing Connection Diagrams (HTML)

The Manage Connection Diagram (HTML) Files dialog box shows you where HTML connection diagrams are located, shows you where they are used, and lets you replace test references to them.

- 1 From the application generator's main menu, choose **Tools > Manage > Connection Diagrams (HTML)...**
- 2 In the Manage Connection Diagram (HTML) Files dialog box:



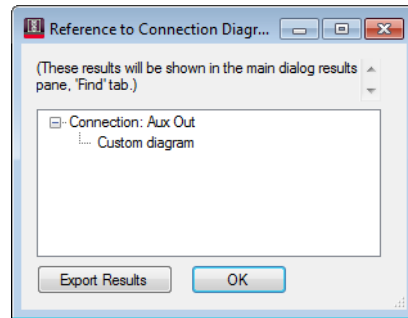
To open a folder where a connection diagram is located:

- a Select the connection diagram file whose location you want to open a folder at.
- b Click **Open Folder...**
- c In the Explorer window, you can quickly edit the connection diagram file, copy it, etc.

To display connection diagram usage:

- a Select the connection diagram file whose usage you want to display.
- b Click **Find References**.

The Usage dialog box shows the connections where the connection diagram file is used. You can export the results to a file using the **Export Results** button.



- c Click **Close** to close the Usage dialog box.

NOTE

Usage results are copied to the main dialog box in the lower pane Find tab.

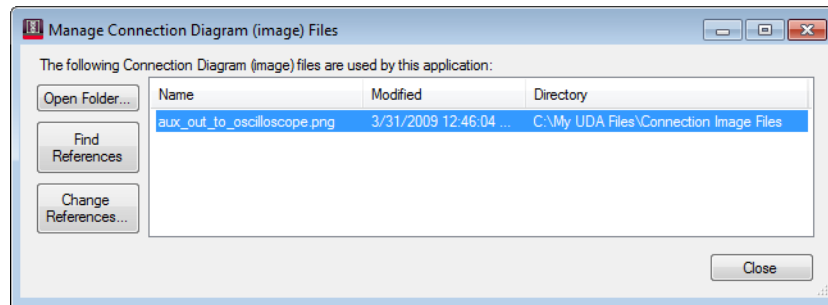
To replace a connection diagram file:

- a Select the connection diagram file you want to replace.
- b Click **Change References...**
- c In the file browser dialog box, select the new connection diagram file; then, click **OK**.

Managing Connection Diagrams (Images)

The Manage Connection Diagram (image) Files dialog box shows you where connection diagram image files are located, shows you where they are used, and lets you replace connection references to connection diagram images.

- 1 From the application generator's main menu, choose **Tools > Manage > Connection Diagrams (images)...**
- 2 In the Manage Connection Diagram (image) Files dialog box:



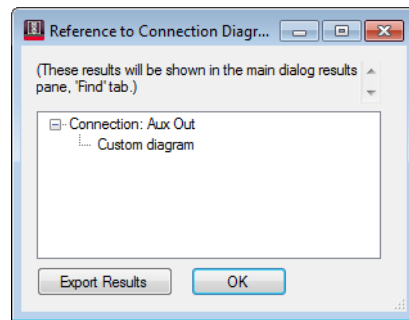
To open a folder where a connection diagram image file is located:

- a Select the connection diagram image file whose location you want to open a folder at.
- b Click **Open Folder...**
- c In the Explorer window, you can quickly edit the connection diagram file, copy it, etc.

To display connection diagram image file usage:

- a Select the connection diagram image file whose usage you want to display.
- b Click **Find References**.

The Usage dialog box shows the connections where the connection diagram image file is used. You can export the results to a file using the **Export Results** button.



- c Click **Close** to close the Usage dialog box.

NOTE

Usage results are copied to the main dialog box in the lower pane Find tab.

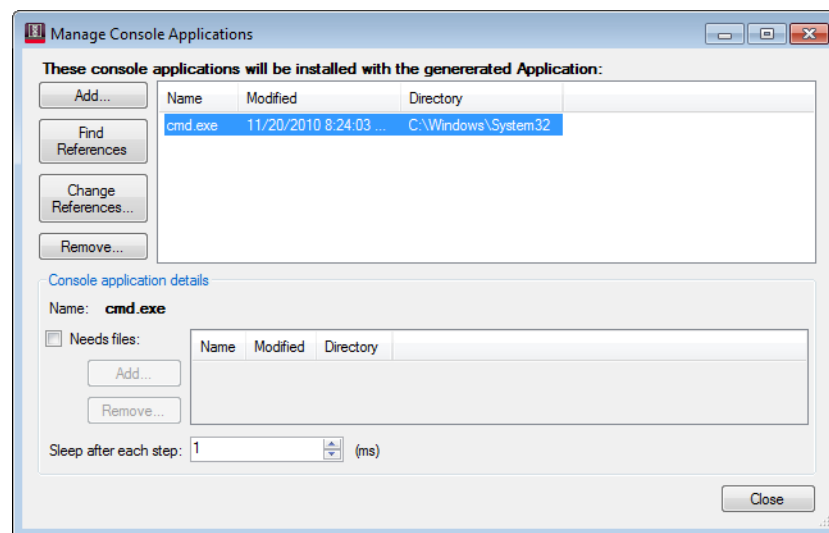
To replace a connection diagram image file:

- a Select the connection diagram image file you want to replace.
- b Click **Change References...**
- c In the file browser dialog box, select the new connection diagram image file; then, click **OK**.

Managing Console Applications

The Manage Console Applications dialog box lets you add console applications, see where they are used, change references, substitute, and remove them. Also, you can add files to be included with the console application, specify a sleep time, and specify whether the application should be included in the installer.

- 1 From the application generator's main menu, choose **Tools > Manage > Console Applications...**
- 2 In the Manage Console Applications dialog box:



To add a console application:

- a Click **Add...**
- b In the Open dialog box, browse to select the console application executable; then, click **Open**.

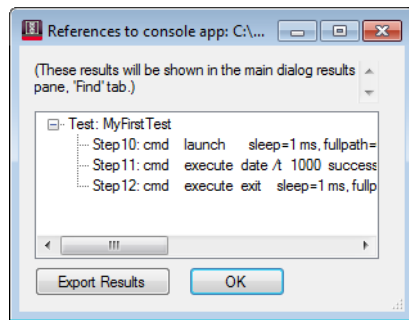
You cannot add two console applications with the same name.

- c Click **OK**.

To display console application usage:

- a Select the console application whose usage you want to display.
- b Click **Find References**.

The Usage dialog box shows where the console application is used. You can export the results to a file using the **Export Results** button.



- c Click **Close** to close the Usage dialog box.

NOTE

Usage results are copied to the main dialog box in the lower pane Find tab.

To change test references to console applications (when multiple console applications are managed):

- a Select the console application whose references you want to change.
- b Click **Change References...**
- c In the Select Console Application dialog box, select the one to use instead.

The console application you select should support the same commands that test steps specified with the first application; otherwise, you need to change those test steps.

- d Click **OK**.

To substitute a new console application for an existing one (this removes the old console application and replaces it with a new one – it can be used when the old and new console applications have the same name but are in different directories):

- a Select the console application you want to substitute.
- b Click **Change References...**
- c In the file browser dialog box, select the new console application.
- d Click **Open**.

To remove console applications:

- a Select the console application you want to remove..
- b Click **Remove...**
- c In the confirmation dialog box, click **OK**.

To specify console application details:

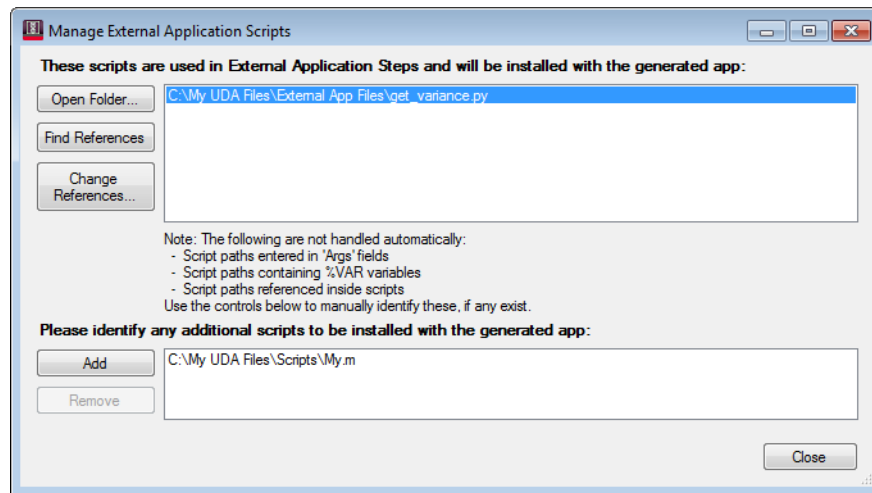
- You can add or remove any files that should be included with the console application.
- You can specify a sleep time after each console application step.

Managing External Application Scripts

The Manage External App Files dialog box shows you where external application files are located, shows you where they are used, and lets you replace test references to them. TIP: Using the predefined variable `Reserved_ExtAppScriptsPath` makes the use of this feature much simpler.

This dialog box also lets you add (to the generated application installer) any script files that are not automatically added. These include script files included with external application program arguments, script files that are identified using `%VAR` variables, and scripts referenced inside other scripts.

- 1 From the application generator's main menu, choose **Tools > Manage > External Application Scripts...**
- 2 In the Manage External App Files dialog box:



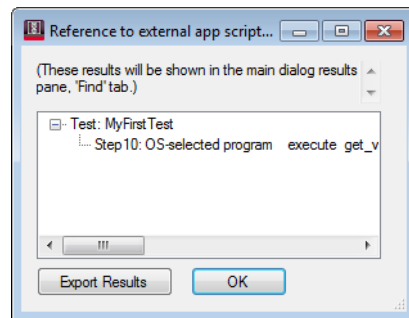
To open a folder where an external application file is located:

- a Select the external application file whose location you want to open a folder at.
- b Click **Open Folder...**
- c In the Explorer window, you can quickly edit the external application file, copy it, etc.

To find the test references to the external application file:

- a Select the external application file.
- b Click **Find References**.

The Usage dialog box shows the tests and steps where the external application file is used. You can export the results to a file using the **Export Results** button.



- c Click **Close** to close the Usage dialog box.

NOTE

Usage results are copied to the main dialog box in the lower pane Find tab.

To change references to an external application file:

- a Select the external application file you want to replace.
- b Click **Change References...**
- c In the file browser dialog box, select the new external application file; then, click **OK**.

To add scripts to be installed with the generated application:

- a Click **Add**.
- b Select the script file you want to add, and click **Open**.

Managing Other Files

The Manage Other Files dialog box allows you to specify User Defined Function files and other miscellaneous files that are to be installed (copied, not executed) with the generated app.

User Defined Function Files

You can specify User Defined Functions files to be installed into Infiniium's UDF directory. This is done using a generic mechanism for adding miscellaneous files to a UDA project (**Tools > Manage > Other Files**). By adding **<Option>MYC</Option>** to the UDF .xml file, you ensure the function is only loaded when the generated app is running. Below is an example of a UDF .xml file with this line added:

```
<?xml version="1.0" encoding="utf-8"?>
<Functions xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="Functions.xsd" Version="1">
  <Function>
    <Name> Butterworth</Name>
    <Abbreviation> BTR</Abbreviation>
    <MATLABName>Butterworth</MATLABName>
    <Option>MYC</Option>
    <FunctionType>1 Source</FunctionType>
    <SourceWindow>All</SourceWindow>
    <XUnits>Same</XUnits>
    <YUnits>Same</YUnits>
    <Control Name = "Data Rate">
      <Double>
        <Min>1e9</Min>
        <Max>10e9</Max>
        <Resolution>1000</Resolution>
        <Units>bitspersec</Units>
        <Default>2.5e9</Default>
      </Double>
    </Control>
    <Control Name = "Low Pass">
      <Enumeration>
        <Default>Off</Default>
        <Selection>Off</Selection>
        <Selection>On</Selection>
      </Enumeration>
    </Control>
  </Function>
</Functions>
```

It is generally a good idea to use a naming convention such as "CompanyName.UserDefinedAppName.YourFileName" for any file not being installed inside the app's directory to help prevent name collisions in external directories. Please refer to the Infiniium help system, *Programmer's Reference*, and the *User Defined Function User's Guide* for more information on remote programming using UDFs.

NOTE

You must compile your UDF to target the same version of the MATLAB MCR that is used by the version of Infiniium that your app will be running with.

Miscellaneous Files (in general) Besides UDF files discussed above, you may also use the Manage Other Files mechanism to install (copy, not execute) any other file that is not already getting installed by some other UDA mechanism.

You can specify the destination of these files as:

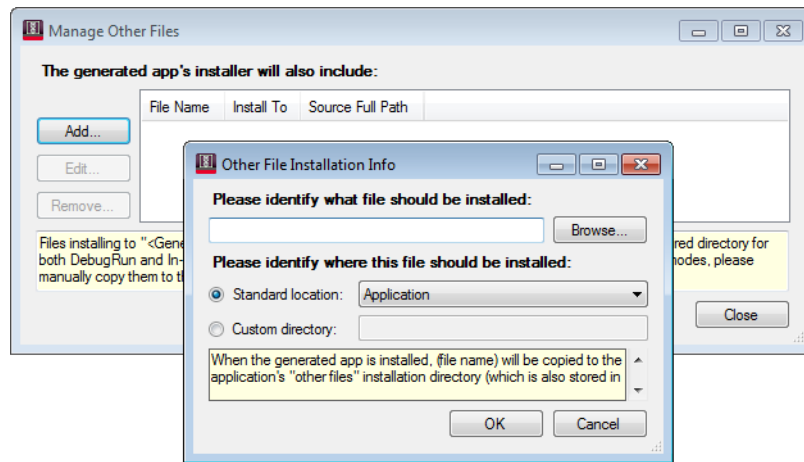
- **Application** – Gets installed inside the generated app's install directory. Will get uninstalled by the generated app uninstaller.
- **User Defined Function (32-bit MCR 7.5), User Defined Function (32-bit MCR 7.15), User Defined Function (64-bit MCR 7.15)** – (as discussed above). The generated app installer will choose the UDF required by the version of Infiniium found on the machine. Does not get uninstalled by the generated app uninstaller.
- **Custom** – You can specify any target directory. Does not get uninstalled by the generated app uninstaller.

Accessing These Files at Runtime

- **Application** – Although Debug Run usually executes user files from their working location (c:\uda inputs\etc., for example), "Application" miscellaneous files get copied to the internal cache normally used only by Launched apps. This is because the step that references these files needs to be able to specify their location using a stable syntax. For example, a console app may have as a parameter the path to some miscellaneous input file. To specify the path to this file in a way that will work for both the Debug Run and In-Tool Launch, you should use one of the predefined variable to specify the path (for example, "arg1, %VAR:Reserved_OtherFilesPath%\MyFileName.txt%, arg3").
- **UDF** – These need to be manually placed in the required Infiniium directory for both Debug Run and In-Tool Launch. No references to these files will exist in steps.
- **Custom** – These need to be manually placed in their respective target directories for both Debug Run and In-Tool Launch. Therefore, the reference to these files should be the literal target path. Repeating the console app arguments example, you would have "arg1, C:\temp\MyFileName.txt, arg3".

To add other files:

- 1 To access the Manage Other Files dialog box, choose **Tools > Manage > Other Files...** from the application generator's main menu dialog box.
- 2 In the Manage Other Files dialog box, click **Add...** to open the Other File Installation Info dialog box:



To identify the file that should be installed:

- a Click **Browse....**
- b Navigate to the location of the desired file, highlight it, and click **Open**.

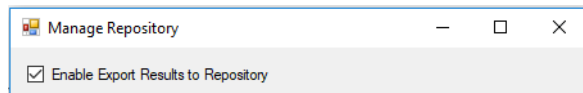
To select where the file should be installed:

- a Select either Application or User Defined Function from the Standard Location dropdown menu or click the radio button next to Custom Directory and manually type the desired file path location into the field next to it.
- b Click **OK**.

Managing Repository

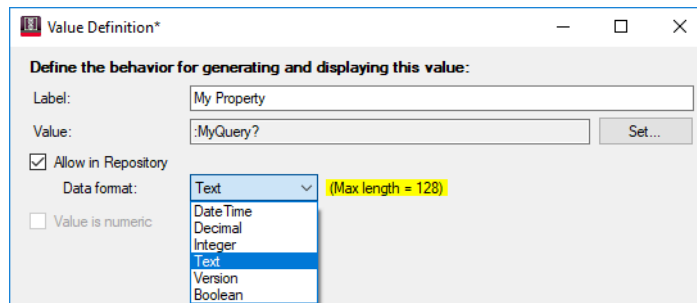
The Manage Repository dialog box allows you to enable properties and results to be eligible for upload to Keysight KS6800A Data Analytics software.

- 1 From the application generator's main menu, choose **Tools > Manage > Repository...**
- 2 In the Manage Repository Files dialog box:

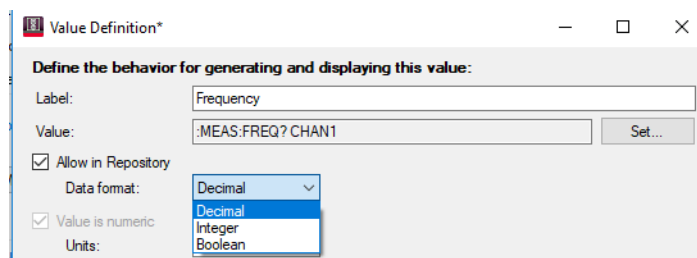


- a To enable items to be uploaded, select the **Enable Export Results to Repository** check box and click **OK**.

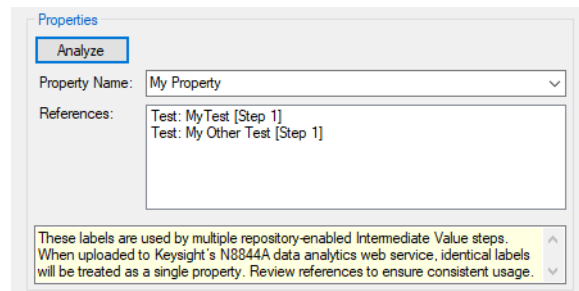
In tests and subroutines, Intermediate Value steps will now display an **Allow in Repository** option along with a drop-down list for declaring the value's data type (used by the repository):



Final Result steps will also display the same option, though with fewer available data types:



- b During test development, return to this dialog box and click the **Analyze** button to review usage to ensure that eligible property names are unique where you intend them to be. For example:

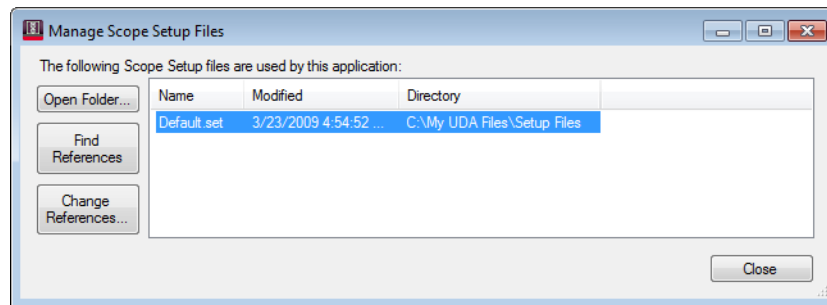


In this example, two different tests are reporting a value named "My Property". If the property has different meaning between the tests, consider renaming one to prevent them from being combined by the repository.

Managing Scope Setup Files

The Manage Scope Setup Files dialog box shows you where setup files are located, shows you where they are used, and lets you replace setup files.

- 1 From the application generator's main menu, choose **Tools > Manage > Scope Setup Files....**
- 2 In the Manage Scope Setup Files dialog box:



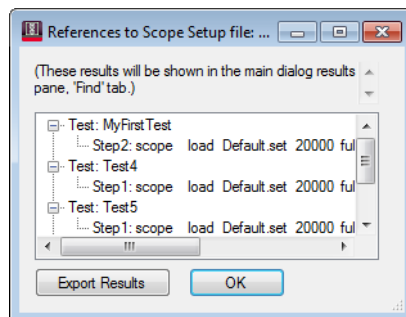
To open a folder where a setup file is located:

- a Select the setup file whose location you want to open a folder at.
- b Click **Open Folder....**
- c In the Explorer window, you can quickly copy the setup file, etc.

To display setup file usage:

- a Select the command file whose usage you want to display.
- b Click **Find References**.

The Usage dialog box shows the tests and steps where the command file is used. You can export the results to a file using the **Export Results** button.



- c Click **Close** to close the Usage dialog box.

NOTE

Usage results are copied to the main dialog box in the lower pane Find tab.

To replace test references to a setup file:

- a** Select the setup file you want to replace.
- b** Click **Change References...**
- c** In the file browser dialog box, select the new setup file; then, click **OK**.

Managing Scope Mask Files

The Manage Scope Mask Files dialog box shows you where scope mask files are located, shows you where they are used, and lets you replace them.

1 From the application generator's main menu, choose **Tools > Manage > Scope Mask Files...**

2 In the Manage Scope Mask Files dialog box:

To open a folder where a mask file is located:

- a** Select the mask file whose location you want to open a folder at.
- b** Click **Open Folder...**
- c** In the Explorer window, you can quickly copy the mask file, etc.

To display mask file usage:

- a** Select the command file whose usage you want to display.
- b** Click **Find References**.

The Usage dialog box shows the tests and steps where the command file is used.

- c** Click **Close** to close the Usage dialog box.

NOTE

Usage results are copied to the main dialog box in the lower pane Find tab.

To replace test references to a mask file:

- a** Select the mask file you want to replace.
- b** Click **Change References...**
- c** In the file browser dialog box, select the new mask file; then, click **OK**.

(InfiniiVision apps only) To download a mask to a file on the PC:

- a** Display the mask on the oscilloscope.
- b** Ensure you are connected to the oscilloscope (see the Debug Run tab).
- c** Click **Download From Scope...**
- d** In the file browser dialog box, choose a location and file name to save the mask to; then, click **OK**.

Now you can use this file in a Load Scope Mask File step.

Managing Transfer Function Files

The Manage Transfer Function Files dialog box shows you where transfer function files are located, shows you where they are used, and lets you replace them.

1 From the application generator's main menu, choose **Tools > Manage > Transfer Function Files...**

2 In the Manage Transfer Function Files dialog box:

To open a folder where a transfer function file is located:

- a** Select the transfer function file whose location you want to open a folder at.
- b** Click **Open Folder...**
- c** In the Explorer window, you can quickly copy the transfer function file, etc.

To display transfer function file usage:

- a** Select the transfer function file whose usage you want to display.
- b** Click **Find References**.

The Usage dialog box shows the tests and steps where the transfer function file is used.

- c** Click **Close** to close the Usage dialog box.

NOTE

Usage results are copied to the main dialog box in the lower pane Find tab.

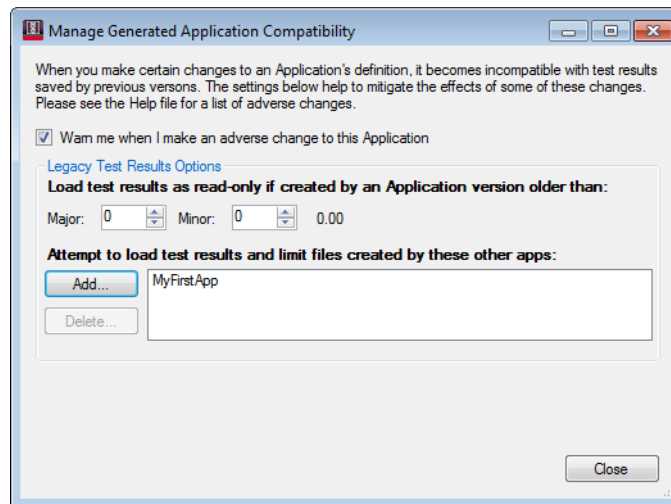
To replace test references to a transfer function file:

- a** Select the transfer function file you want to replace.
- b** Click **Change References...**
- c** In the file browser dialog box, select the new transfer function file; then, click **OK**.

Managing Generated App Compatibility

The Manage Generated App Compatibility dialog box contains options to help you manage a generated application's backward compatibility.

- 1 From the application generator's main menu, choose **Tools > Generated App Compatibility...**
- 2 In the Manage Generated App Compatibility dialog box:



The **Warn me when I make an adverse change to this application** option gives you warnings when making project changes that break generated application backward compatibility:

- Deleting a test (the new generated application will fail to open legacy projects).
- Changing a test's ID (new generated application will fail to open legacy projects).
- Changing a test's Intermediate Value steps:

These Intermediate Value changes will cause a run to abort when the new generated application appends results to a legacy project:

- Adding a new Intermediate Value.
- Changing the label of an existing Intermediate Value.
- Changing the order of existing Intermediate Values.
- Deleting an Intermediate Value.

This Intermediate Value change will cause a multi-trial summary table to produce bad statistics when the new generated application appends results to a legacy project:

- Changing the **Value is numeric** property.

- Changing a test's pass limits (when the new generated application appends results to a legacy project, a multi-trial summary table will produce bad statistics).
- Changing a Config variable (when the new generated application opens a legacy project, the Config variable will not restore to saved value).
- Changing a Config variable choice's value (when the new generated application opens a legacy project, the Config variable will not restore to this saved value).

The **Load generated app projects as read-only if version is older than** area lets you identify when to load a generated application's project file as read-only based on the major and minor version of the generated application that saved the project file.

In the case of generated application projects, read-only projects may be opened, and the saved results can be reviewed, but you cannot re-run tests.

If you are unsure whether a test generated by the old version is compatible with the new version, use the old version to create a project that contains a result for that test. Then, use the new version to load that project and run the test again (using append option). Look at the outputs of the two trials for this test and look at the summary statistics for this test. Does everything look consistent?

If you are unsure whether Config variable changes are compatible, use the old version to create a project with various settings. Then use the new version to load that project and look at the Config variable options. Are they set as you would expect?

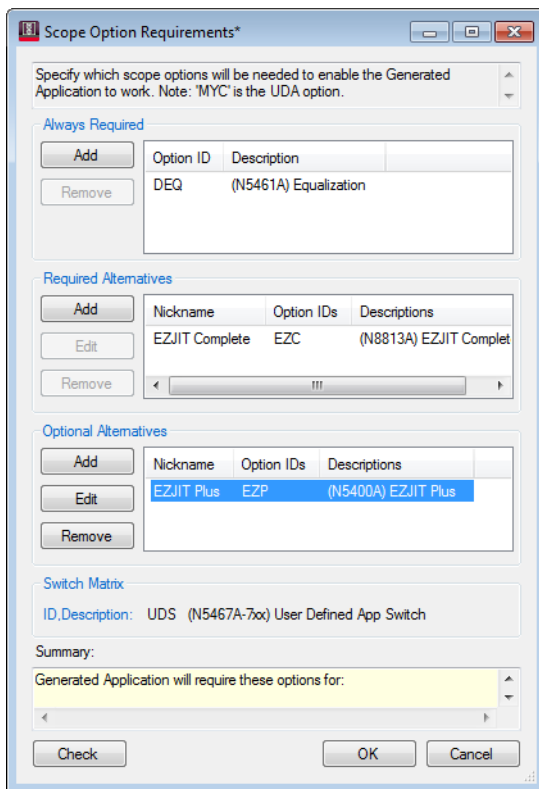
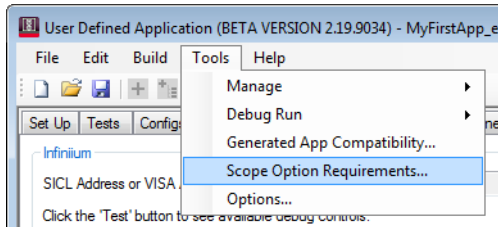
The **Attempt to load generated app project and limit files created by these other apps** list lets you specify when a generated application can open project or limit files created by another generated application. This can be useful, for example, when you have two compatible versions of an application with different names.

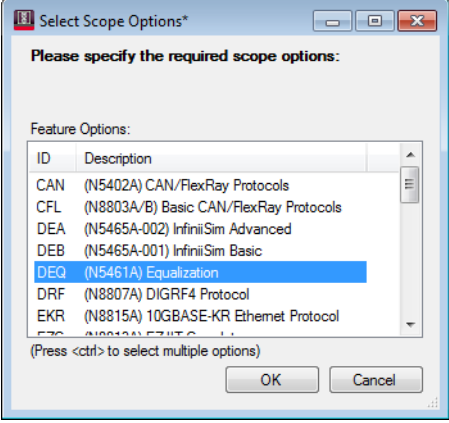
- Click **Add** to add the names of compatible applications.
- Click **Delete** to remove the names of incompatible applications.

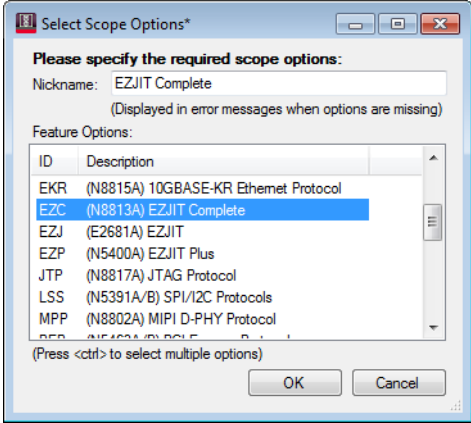
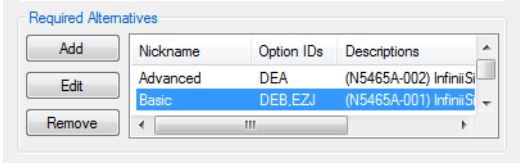
- 3 Click **Close** to close the dialog box.

Scope Option Requirements

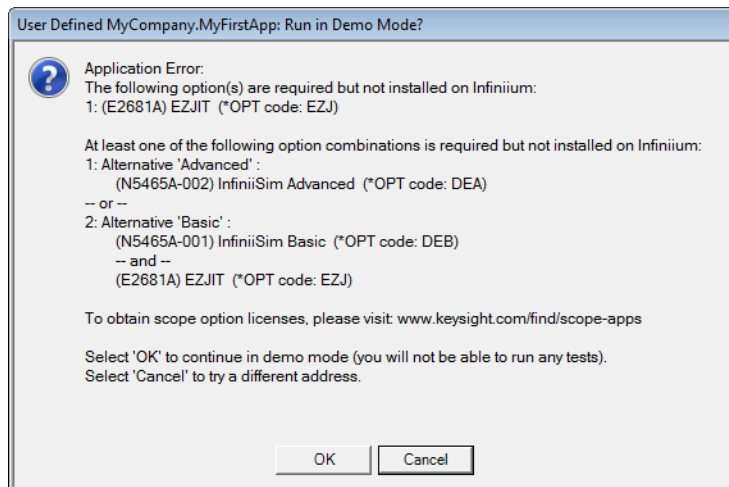
You can specify which scope options your app or add-in requires, so that when the generated app/add-in launches you will be warned if any are missing.



(FlexDCA apps only) Query	Select which hardware you want the requirements to apply to.
Always Required	<p>In the "Always Required" group, click Add to specify scope options that must be present:</p>  <p>Every option you specify must be present on the scope. Otherwise, the generated app will not allow you to run its tests.</p>

<p>Required Alternatives</p>	<p>In the "Required Alternatives" group, click Add to specify groups of scope options that must be present:</p>  <p>Every option of at least one alternative you specify must be fully present on the scope. Otherwise, the generated app will not allow you to run its tests. In this example, two alternatives are specified:</p>  <p>The example generated app will require the scope have the DEA license, or that the scope have both the DEB and EZJ licenses.</p>
<p>Optional Alternatives (Generated Apps, only)</p>	<p>For each alternative you specify here, if it is fully present on the scope then a Reserved variable (see Summary field) will be set to True. You can use this variable in your tests to determine if it is safe to execute optional steps that require these scope options.</p>
<p>Summary</p>	<p>This text describes how the generated app/add-in will combine your entries in the above groups into a single, overall specification.</p>

When the generated app launches, it verifies your "Always Required" and "Required Alternatives" specifications. If the requirements are not met, it will display a message such as:



IP Protection

Note that the preceding scope option requirements actions simply enable you to prevent a user from running your generated app when the scope is missing features your app requires, in order to prevent run-time aborts. In those cases, the user can then simply install the missing options to enable full control of the generated app. If you want to tightly control which oscilloscopes your generated application is authorized to run on, please contact your Keysight sales representative for information on custom UDA licensing.

17 Saving / Converting / Opening Projects

Saving Application Generator Projects / 240

Opening Previously Saved Application Generator Projects / 242

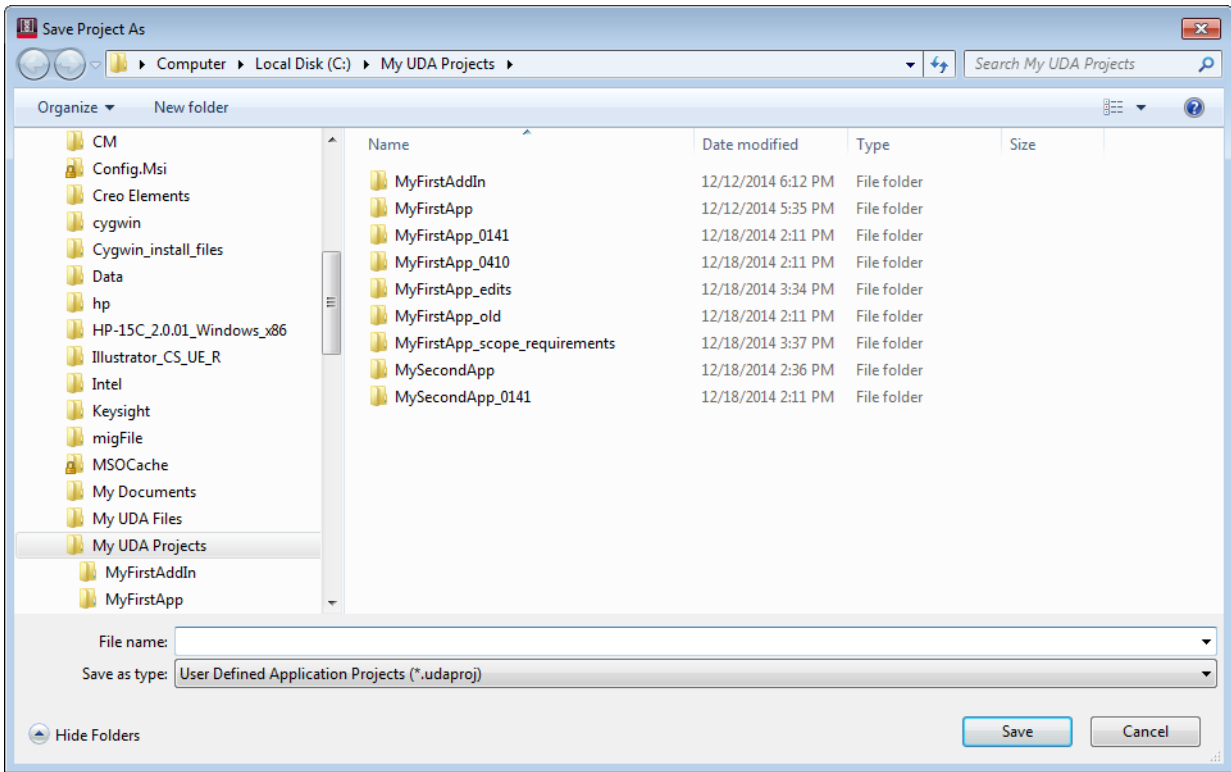
Converting an Add-in Project to an Application Project / 244

Converting an Application Project to an Add-In Project / 245

Application Generator Project Sharing/Collaboration / 247

Saving Application Generator Projects

- 1 From the application generator's main menu, choose **File > Save Project As....**
- 2 In the Save Project As dialog box:



- a Enter your project name.
- b Click **Save**.

A directory will be created for your project. It will contain the .udaproj file as well as other files used to define the application.

A complete UDA project consists of a parent folder containing:

- A .udaproj file.
- A "Studio" folder.
- (Optionally) A "UserFiles" folder.

When you first save a project, the application generator creates this parent folder, giving it the same name you entered for the .udaproj file.

NOTE

Important: The parent folder and the .udaproj file need to maintain the same name. For example:

```
...\MyProject\  
    MyProject.udaproj  
    Studio\  
    User Files\  

```

CAUTION

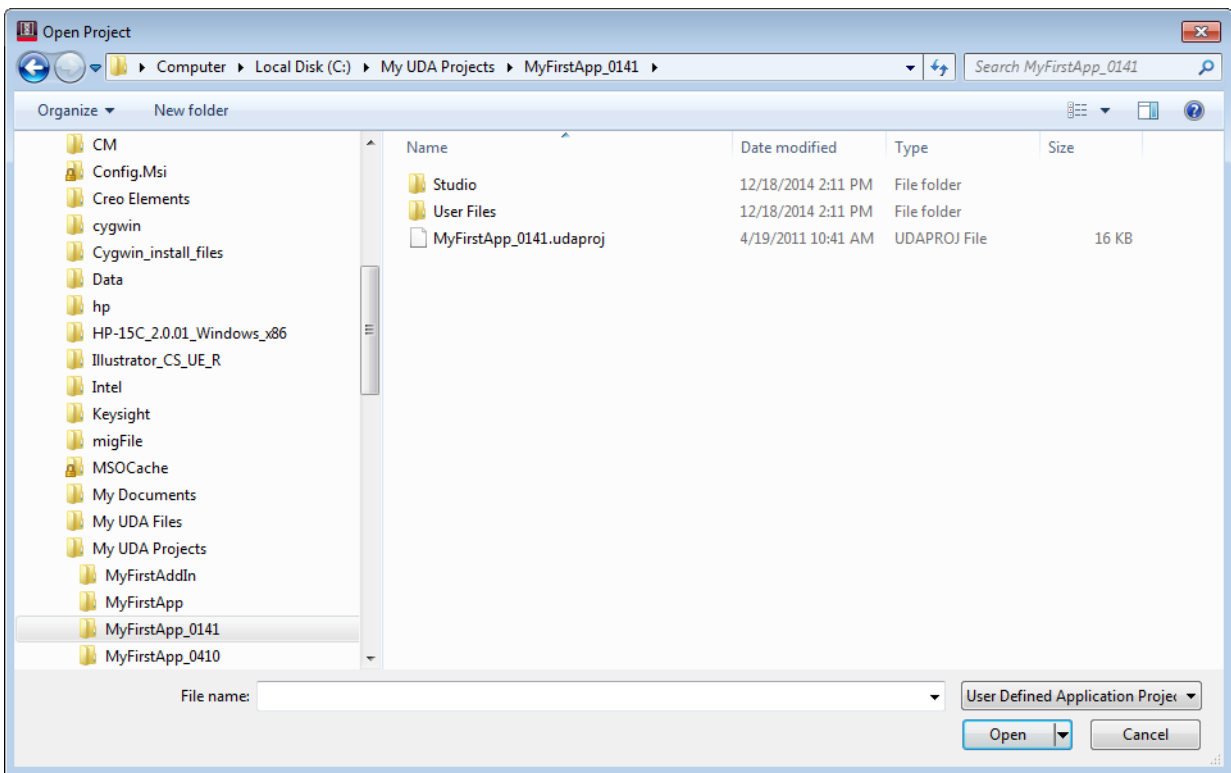
To move or share a UDA project, be sure to include the parent folder (In example above, MyProject) and all of its contents; otherwise, the project may be corrupted.

CAUTION

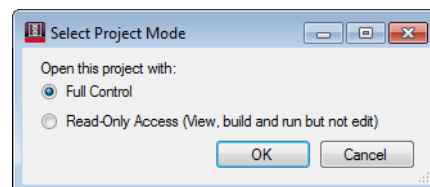
Do not save a project inside a directory tree that contains your working files (otherwise, you may lose working files).

Opening Previously Saved Application Generator Projects

- 1 From the application generator's main menu, choose **File > Open Project...**
- 2 In the Open Project dialog box:

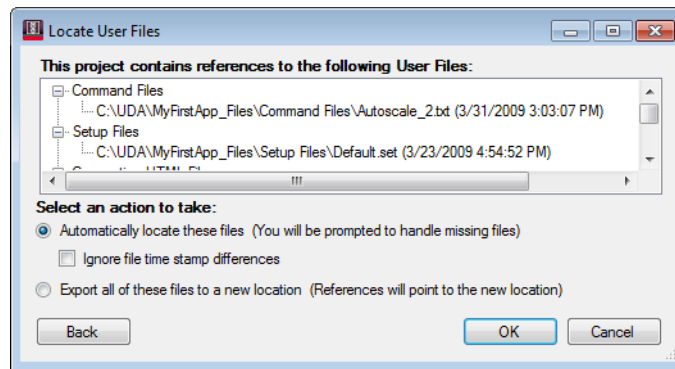


- a Browse to the project's .udaproj file and select it.
 - b Click **Open**.
- 3 In the Select Project Mode dialog box, choose whether you want to open the project with read-only or edit (full control) access.



Then, click **OK**.

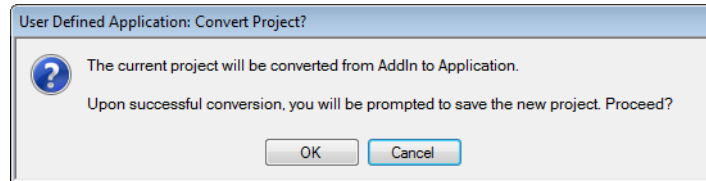
- 4 In the Locate User Files dialog box, select whether you want to automatically locate project user files or export project user files to a new location.



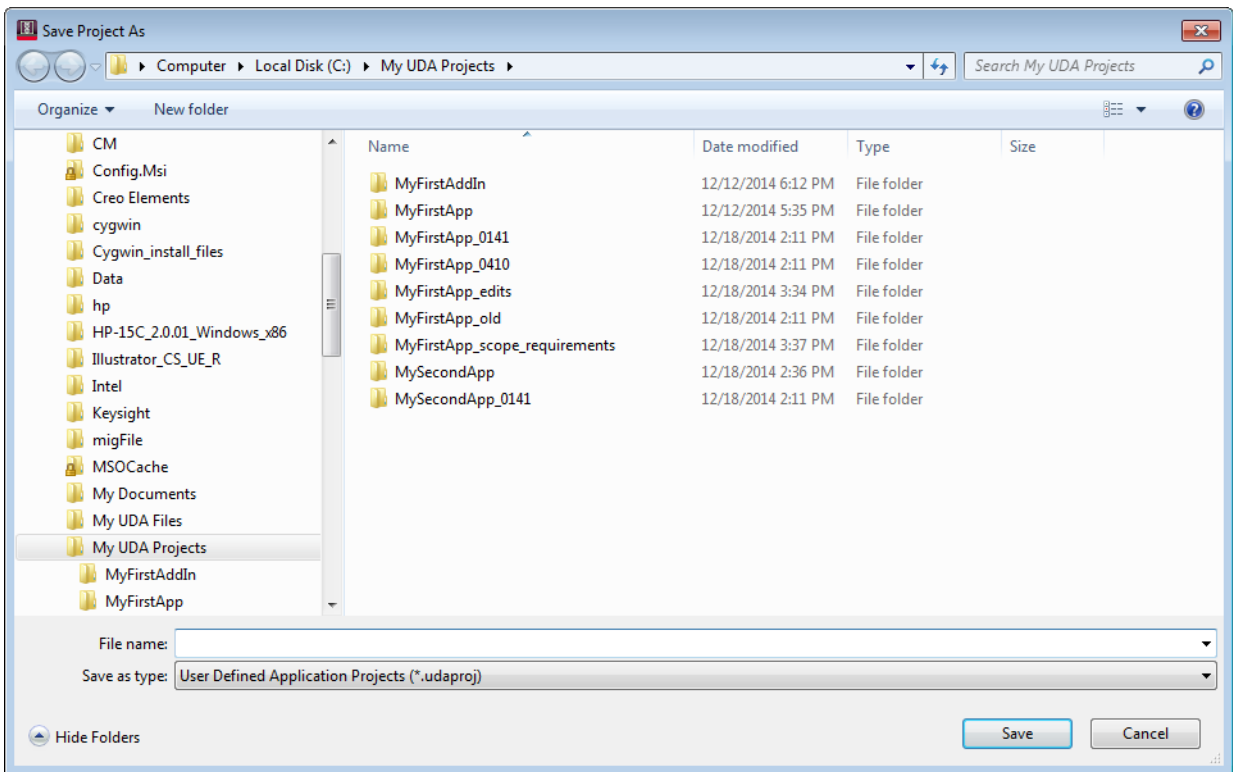
Then, click **OK**.

Converting an Add-in Project to an Application Project

- 1 Open the add-in project to be converted. If already open, be sure it has been saved.
- 2 Choose **File > Convert Project**. The following prompt will display.



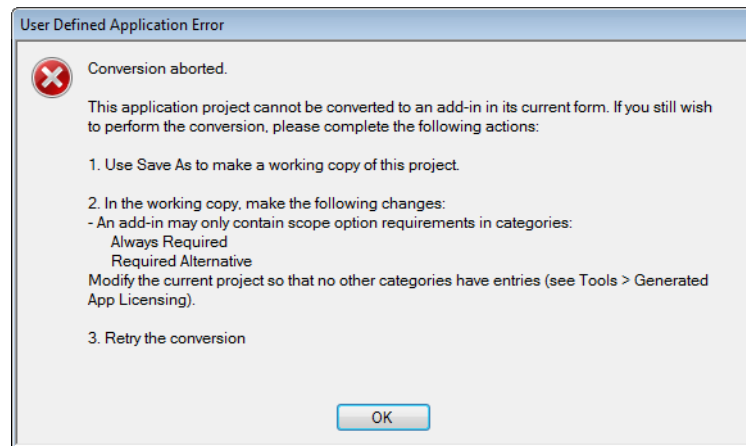
- 3 Click **OK**. The following prompt will display.



- 4 Browse to the location where you want to save the converted project, give it a name, and click **Save**.
- 5 The currently-opened project is now an Application.

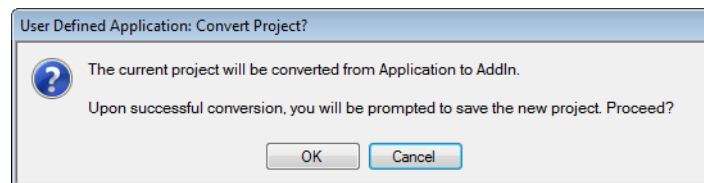
Converting an Application Project to an Add-In Project

- 1 Open the application project to be converted. If already open, be sure it has been saved.
- 2 Choose **File > Convert Project**.
 - If the application contains any disallowed features, you will be prompted to make certain changes to the project before converting. For example,



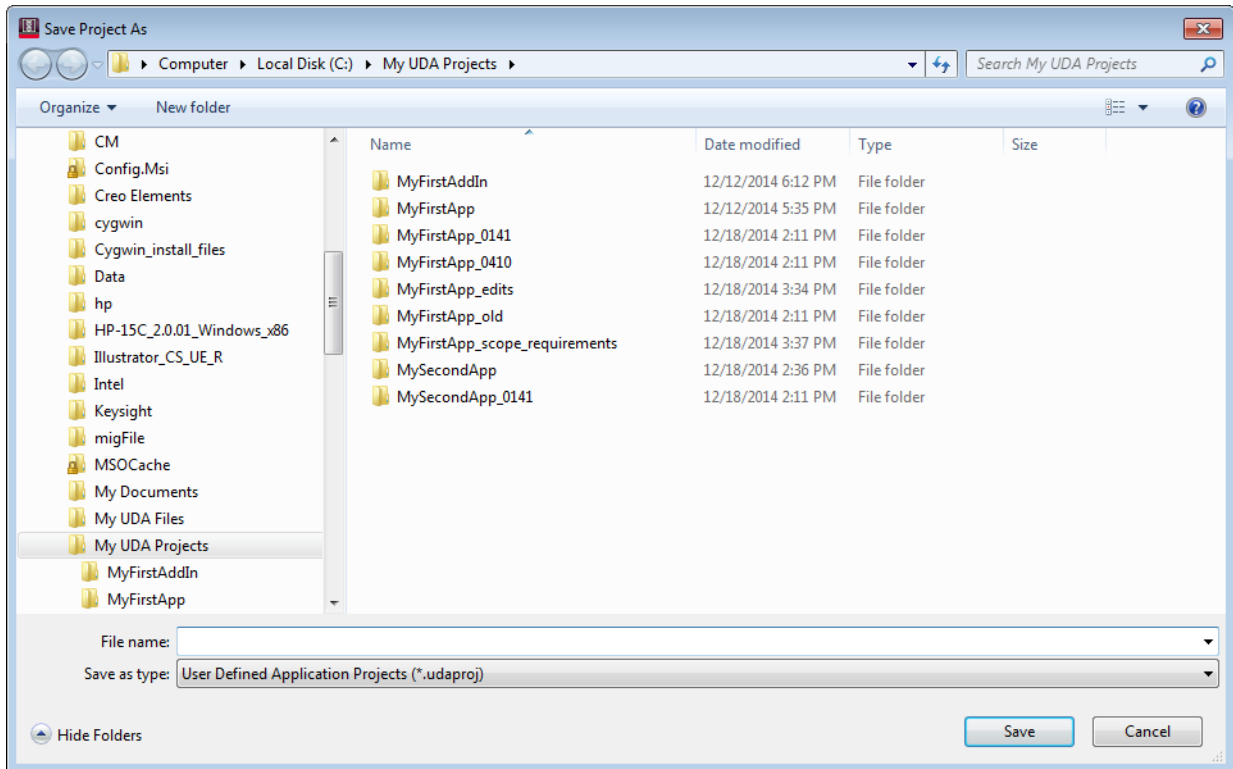
Click **OK**, perform the directed actions, and return to Step 1.

- If the application does not contain any disallowed features, this prompt will display.



- 3 Click **OK**. The following prompt will display.

17 Saving / Converting / Opening Projects



- 4 Browse to the location where you want to save the converted project, give it a name, and click **Save**.
- 5 The currently-opened project is now an Add-in.

Application Generator Project Sharing/Collaboration

You can share your application generator projects with others by sending them a .zip file of the project directory. A project's files are all saved within a directory whose name is the same as the project.

When collaborating on projects, we especially recommend organizing your working files in a directory structure similar to the one used in project files (see "**Organizing Your Working Files**" on page 23).

When opening project files that others have worked on, you are prompted to export files that are missing from your working files directory or have a newer time stamp. This way, you can tell what has been added or changed, and you can move the exported files back into your working files directory to keep your project files up-to-date.

Note that there is no merge facility, so if your collaborator adds Test A and you add Test B, you cannot merge these into an application that has both new tests. Thus either:

- 1 Have one person managing the application and one or more collaborators working on user files.
- Or -
- 2 Have one person modify the application at a time, and send the project to the next person to work on.

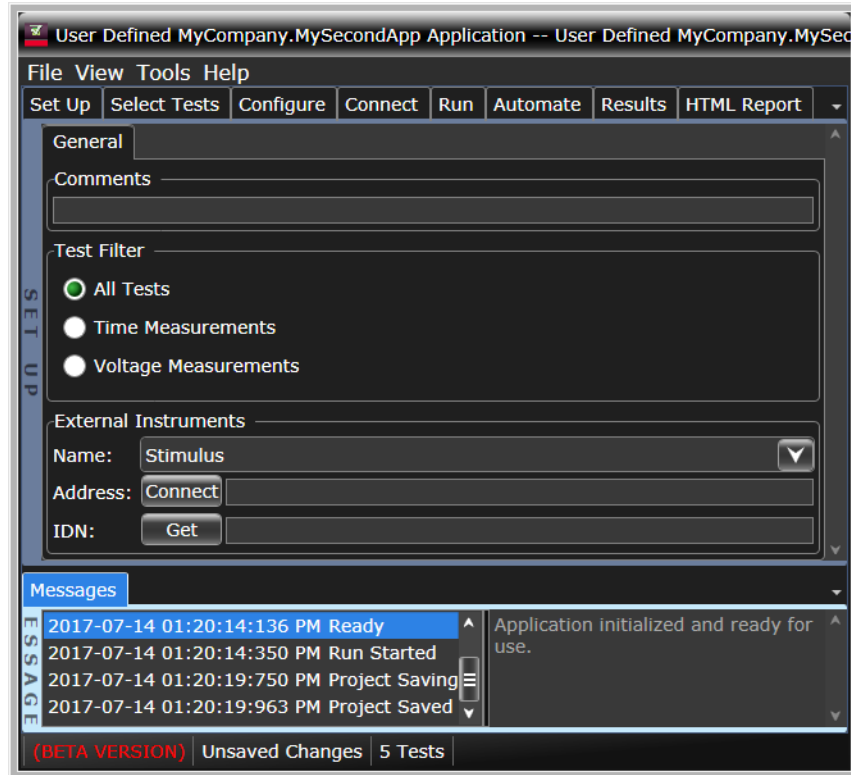
18 Using Generated Applications

Starting the Automated Test Application /	250
Creating or Opening a Test Project /	252
Setting Up the Test Environment /	253
Selecting Tests /	254
Configuring Tests /	256
Connecting the Oscilloscope to the DUT /	267
Running Tests /	268
Config Variable Auto-Iteration /	292
Automating the Application /	295
Executing Miscellaneous Scripts /	301
Viewing Results /	303
Viewing/Exporting/Printing the Report /	308
Exporting Measurement Results to Web Repository /	314
Saving Test Projects /	380
Executing Scripts /	382
Controlling the Application via a Remote PC /	383
Switch Matrix Automation /	386

Starting the Automated Test Application

- 1 From the Infiniium oscilloscope's main menu, choose **Analyze > Automated Test Apps > User Defined > (NamePrefix) > (Application Name)**.

The automated test application window appears.



NOTE

If **(Application Name)** does not appear in the Automated Test Apps menu, the automated test application has not been installed (see "[Installing and Running the Application](#)" on page 61).

The tabs in the main pane show the steps you take when running the automated tests:

Set Up	Lets you add user comments about the test environment setup, select a test filter, and set up any external instruments that are part of the test setup.
Select Tests	Lets you select the tests you want to run. The tests are organized hierarchically so you can select all tests in a group. After tests are run, status indicators show which tests have passed, failed, or not been run, and there are indicators for the test groups.

Configure	Lets you configure test parameters (like oscilloscope channel). This information appears in the HTML report.
Connect	Shows you how to connect the oscilloscope to the device under test for the tests that are to be run.
Run	Starts the automated tests. If the connections to the device under test need to be changed while multiple tests are running, the tests pause, show you how to change the connection, and wait for you to confirm that the connections have been changed before continuing.
Results	Contains more detailed information about the tests that have been run. You can change the thresholds at which marginal or critical warnings appear.
Html Report	Shows a test report that can be printed.

Next · ["Creating or Opening a Test Project"](#) on page 252

Creating or Opening a Test Project

To create a new test project:

- 1 Choose **File > New Project...** from the menu.
A new, empty project, with all the default settings is created.

To open an existing test project:

- 1 Choose **File > Open Project...** from the menu.
- 2 In the Open dialog box, browse to a test project directory and select the desired ".proj" file.
- 3 Click **Open**.

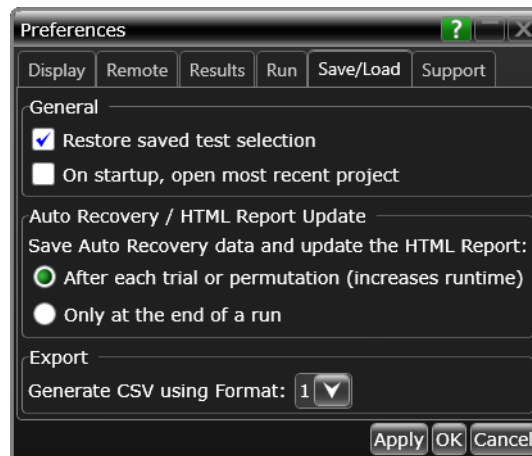
NOTE

Applications generated by UDA 1.70 or newer will open as readonly test projects created by UDA 1.60 or older applications.

- See Also · ["To set load preferences"](#) on page 252
- Next · ["Setting Up the Test Environment"](#) on page 253

To set load preferences

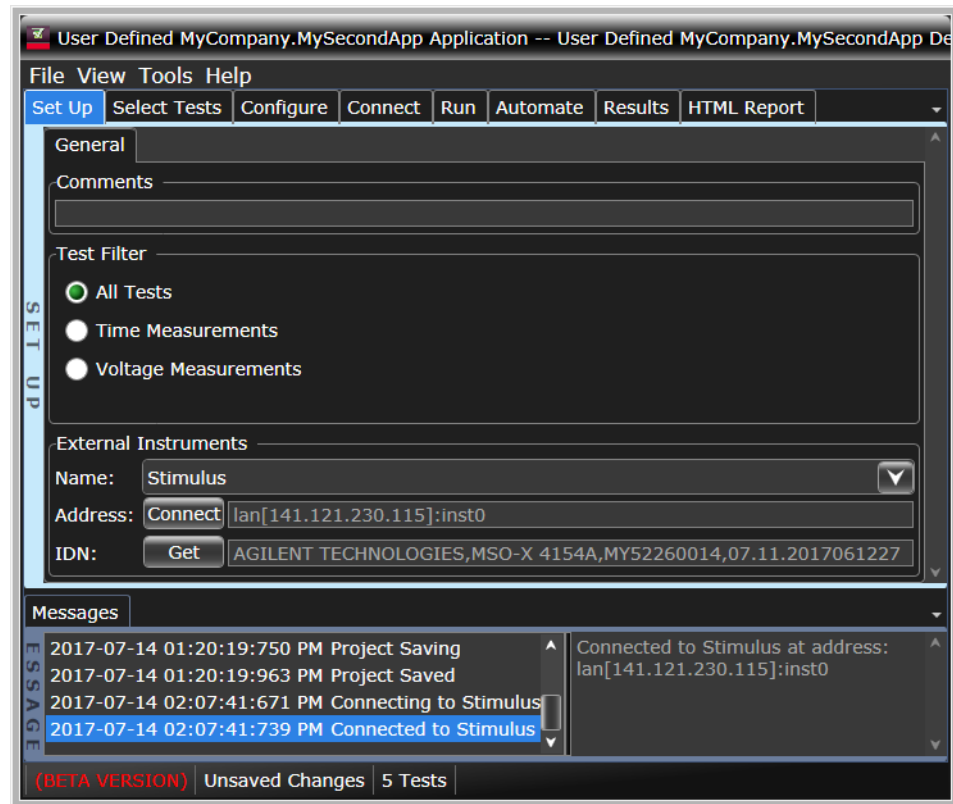
- 1 From the generated application's main menu, choose **View > Preferences....**
- 2 In the Preferences dialog box, select the **Save/Load** tab.



- 3 In the Save/Load tab, you can choose to restore saved test selections when loading a project.
- 4 Click **Apply** to save the changes and click **OK** to close the Preferences dialog box.

Setting Up the Test Environment

- 1 Click the **Set Up** tab.

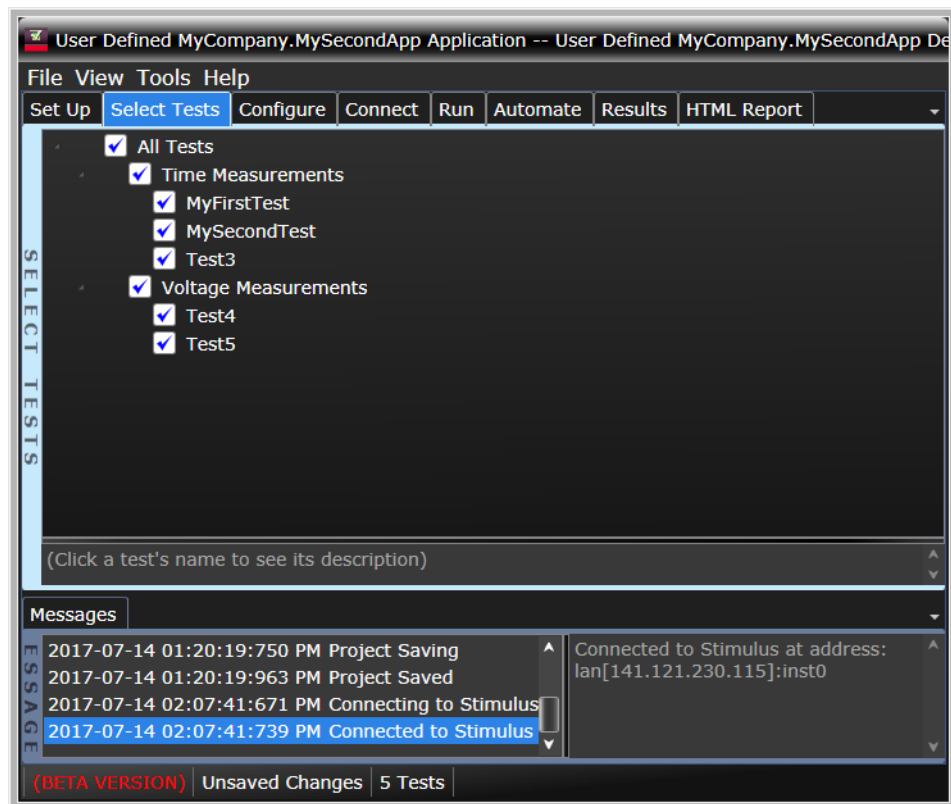


- 2 Enter or select test environment setup information:
 - **User Comments** – enter any relevant information about the test setup. This information is copied to the HTML report.
 - **Test filter** – lets you filter the tests that appear in the Select Tests tab.
 - **External Instruments** – lets you enter and test SICL addresses of external instruments that are part of the test setup.

Next • **"Selecting Tests"** on page 254

Selecting Tests

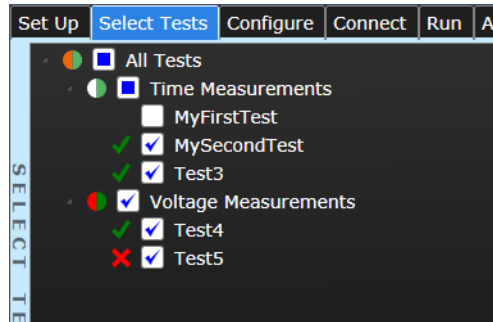
- 1 Click the **Select Tests** tab.
- 2 Check the tests you want to run.



Some things to note:

- Checking a parent node/group will check all available sub-groups/tests.
- Unchecking a parent node/group will uncheck all sub-groups/tests.
- A parent node is checked if all subgroups are checked.
- A parent node is unchecked if ANY subgroup is unchecked.

When Tests Have
Already Been Run



The marks have the following meanings:

✓	The test passed.
✗	The test failed.
○	The test has not been run, or no tests in the group have been run.
●	The test is currently running.
◐	Some tests in the group have run and passed.
◑	Some tests in the group have run and failed.
◒	Some tests in the group have passed and some have failed; not all of the tests have been run.
◓	Some tests in the group have passed and some have failed; all of the tests have run.
●	All tests in the group have run and passed.
●	All tests in the group have run and failed.

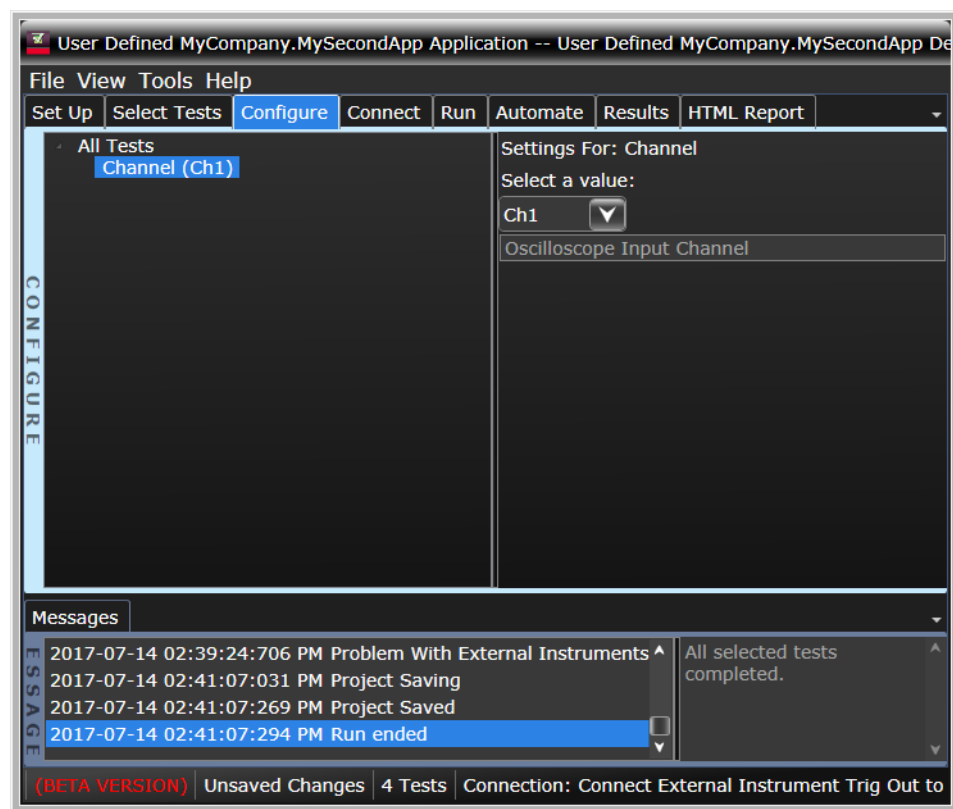
Next · ["Configuring Tests"](#) on page 256

Configuring Tests

- 1 Click the **Configure** tab.
- 2 Select the bulleted item for the settings you want to configure; then, select or enter your settings.

A description of the selected configuration item appears in the lower, right part of the application window.

Note that you can also enter values in some of the drop-down selection fields. Entered values are checked for validity.



TIP

A quick way to reset all configuration options and delete all test results is to create a new project (see [page 252](#)). The new project will have default configuration options.

- See Also
- ["To activate/refresh limit sets"](#) on page 257
 - ["To create/edit limit sets"](#) on page 258
 - ["To access InfiniiSim and PrecisionProbe"](#) on page 263
 - ["\(Infiniium Only\) To access Acquisition Setup in Debug Mode"](#) on page 265

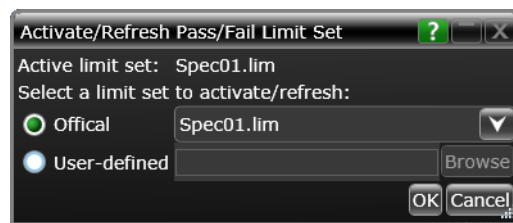
Next · ["Connecting the Oscilloscope to the DUT"](#) on page 267

To activate/refresh limit sets

Limit sets are the values you test your device against. Official limit sets are provided with the generated application. You can also define your own limit sets to test against.

To refresh the current limit set or activate a new limit set:

- 1 From the generated application's main menu, choose **Tools > Compliance Limits > Activate/Refresh limit set....**
- 2 In the Activate/Refresh Pass/Fail Limit Set dialog box, select one of the official limit sets or a user-defined limit set.



- 3 Click **OK**.

NOTE

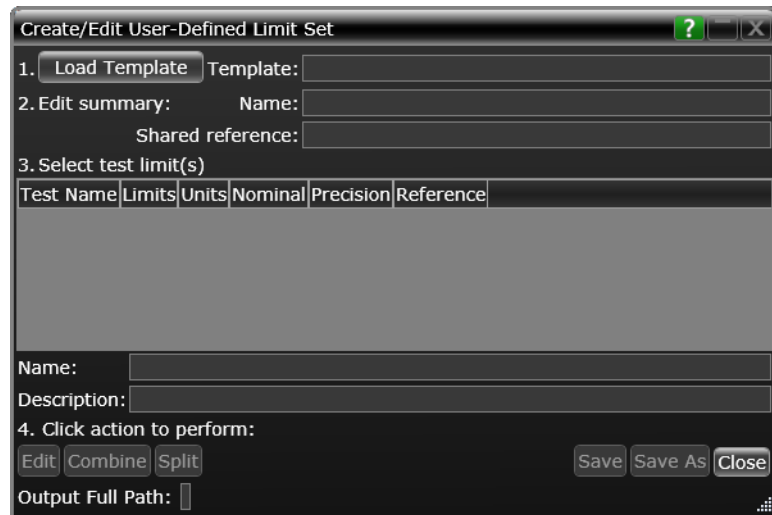
If you have existing test results when you activate a different limit set, the application examines your results to see if any of them would experience a limit change when the different limit set is loaded. If any results would be affected in this way, the application tells you which ones they are and warns that they must be deleted.

See Also · ["To create/edit limit sets"](#) on page 258

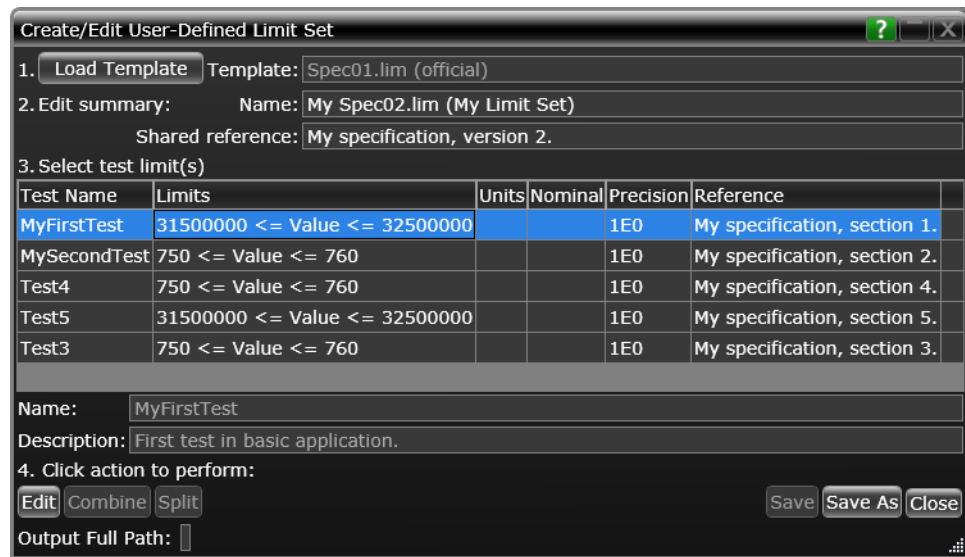
To create/edit limit sets

You can create new limit sets by modifying existing limit sets and saving them to new files.

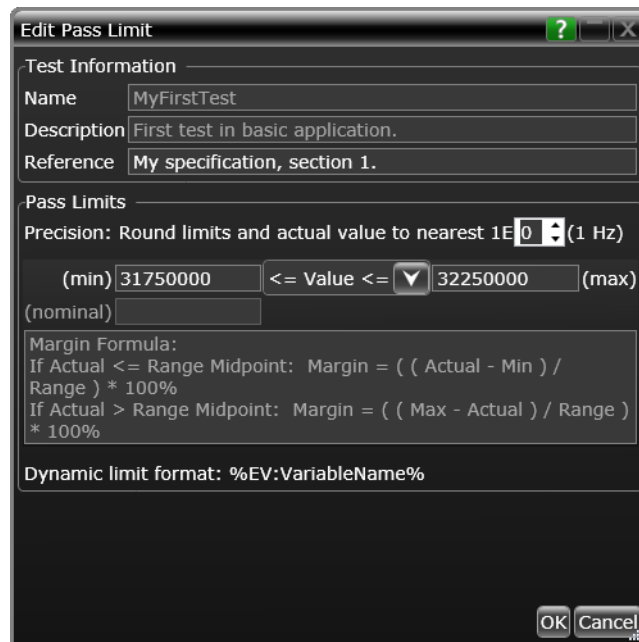
- 1 From the generated application's main menu, choose **Tools > Compliance Limits > Create/Edit limit set...**



- 2 In the Create/Edit User-Defined Limit Set dialog box, click **Load Template** to pre-load the dialog box with an existing official or user-defined limit set.
- 3 Give the new limit set a unique name (different from any official limit set's name). If all of the tests come from the same reference, you can enter a base description (for example, document name) in the **Shared Reference** field and then add test-specific references (for example, page number) down below.
- 4 Select a limit to modify; then, click **Edit**.



- 5 In the Edit Pass Limit dialog box, modify the limit as desired. When finished, click **OK**.

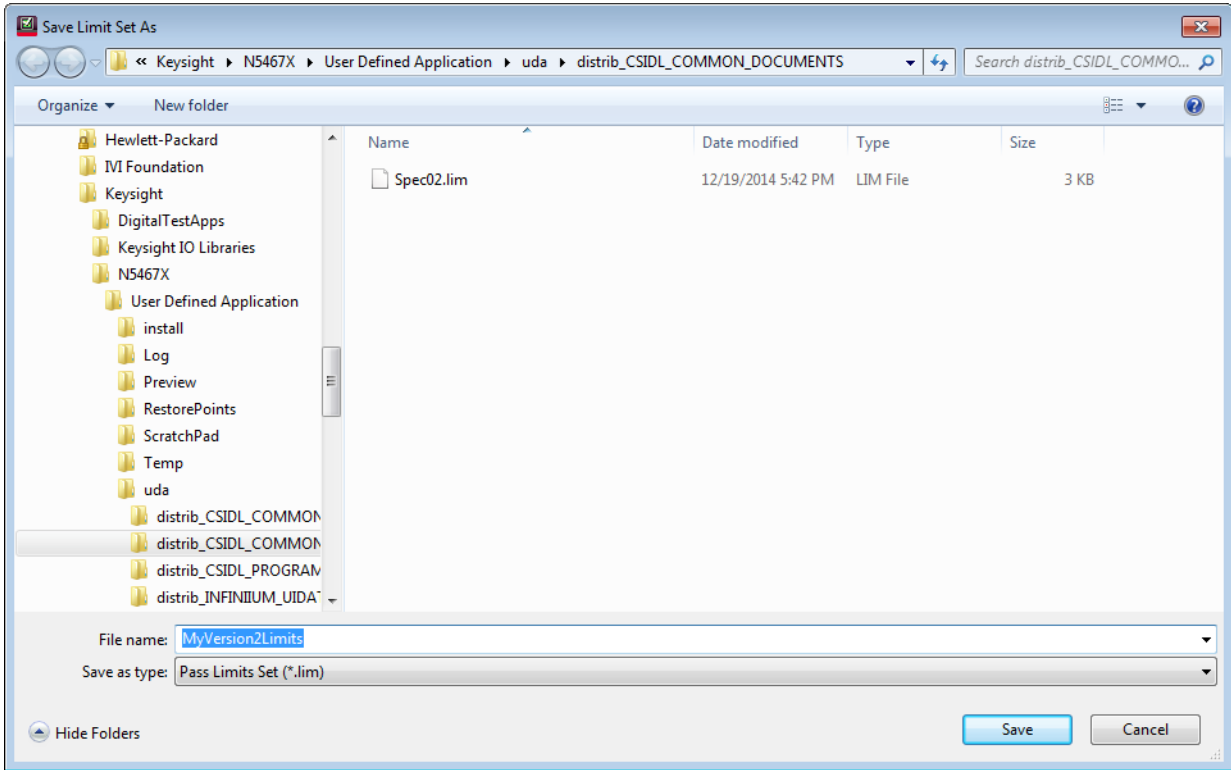


See also:

- ["To combine limits"](#) on page 261
- ["To split a combined limit"](#) on page 262

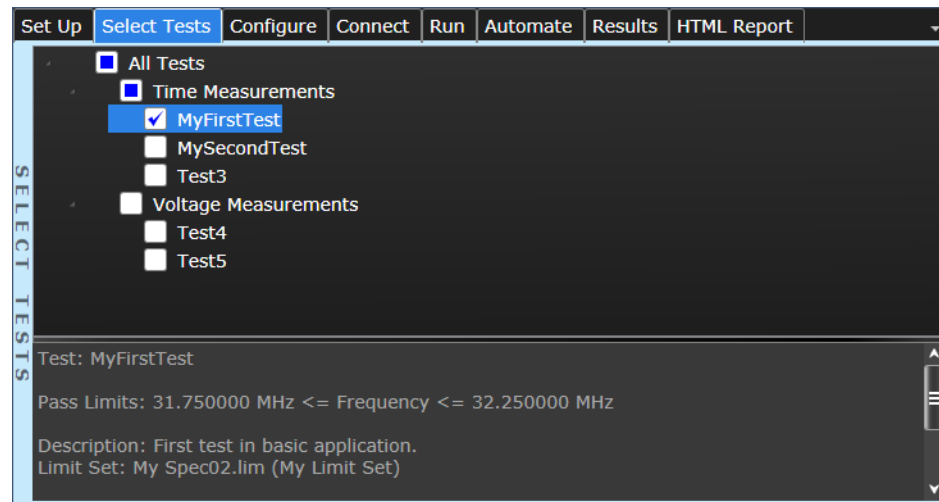
- 6 Repeat the last two steps until all limits requiring change are modified.

- 7 If you have loaded a custom limit set file, click **Save**. Otherwise, click **Save As...** to save your custom limit set to a file. Enter the file name in the Save File As dialog box.



Now, you can activate your newly-created limit set for use in the next run. See ["To activate/refresh limit sets"](#) on page 257.

You can confirm your new limit set is active by viewing test descriptions of any of the tests whose limits you modified in the Select Tests tab or by checking the Activate/Refresh Pass/Fail Limit Set dialog box (see ["To activate/refresh limit sets"](#) on page 257).

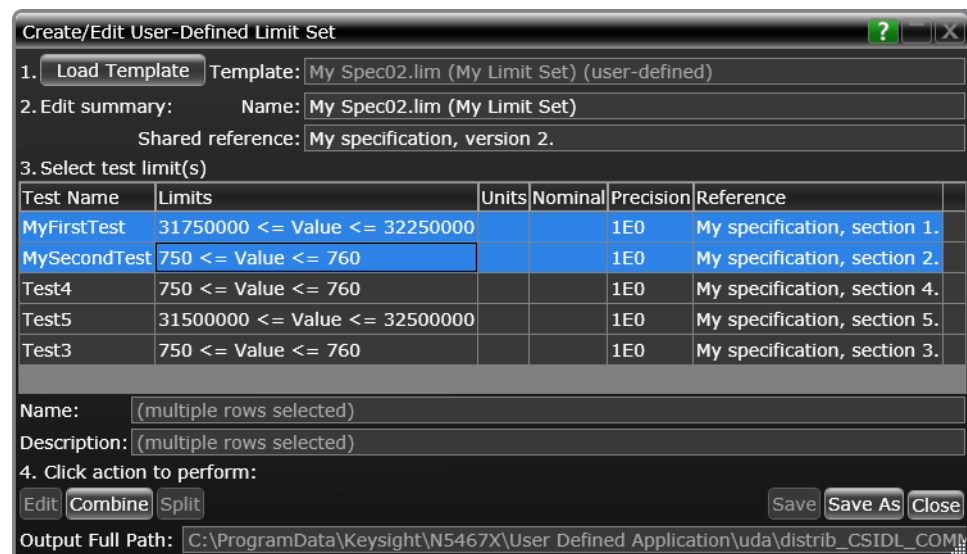


When Loading Projects

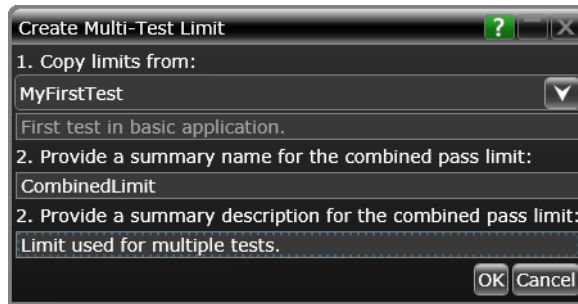
When you load a project, the application will attempt to restore the limit set that was in use at the time the project was saved. For legacy projects, which do not include this information, the application will examine the results being loaded to see if any of them would experience a limit change due to the limit set currently active in the application. If any results would be affected in this way, the application will load the project as read-only.

To combine limits

- 1 In the Create/Edit User-Defined Limit Set dialog box (see **"To create/edit limit sets"** on page 258), select the limits you want to combine, and click **Combine....**



- 2 In the Combine Multiple Test Limits dialog box, select which limit to copy values from and provide summary names and descriptions.



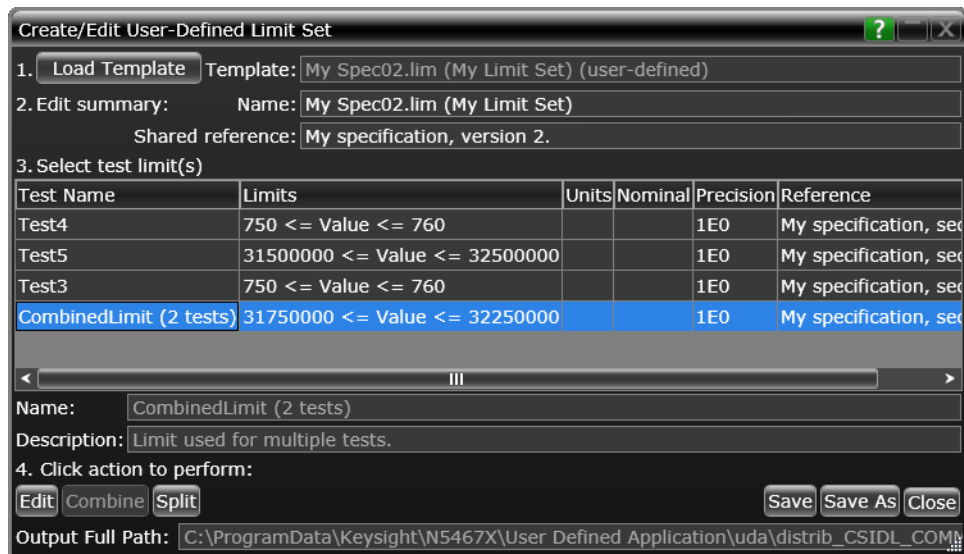
3 Click **OK**.

See Also · ["To split a combined limit"](#) on page 262

To split a combined limit

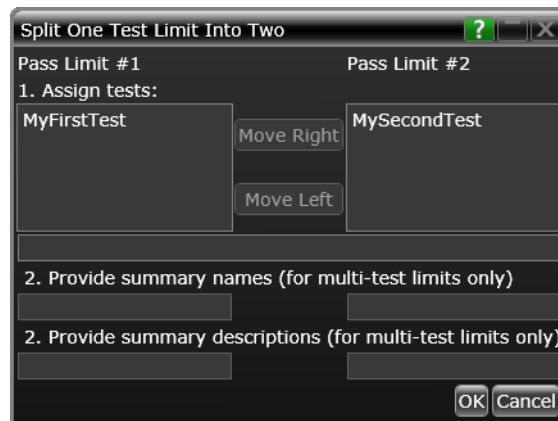
If a limit covers more than one test ID, you can split it into two limits.

1 In the Create/Edit User-Defined Limit Set dialog box (see ["To create/edit limit sets"](#) on page 258), select the limit that covers multiple tests, and click **Split...**



In this case, we are splitting a 2-test limit into two single-test limits.

2 In the Split Test Limit dialog box, assign one of the tests to the new limit by selecting it and clicking the **Move Right** button.

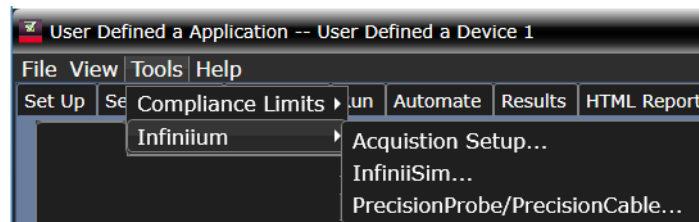


3 Click **OK**.

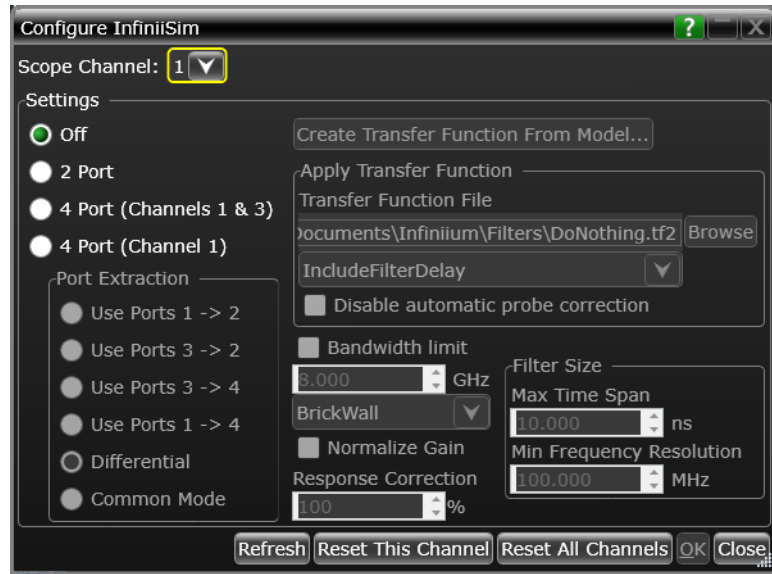
See Also · ["To combine limits"](#) on page 261

To access InfiniiSim and PrecisionProbe

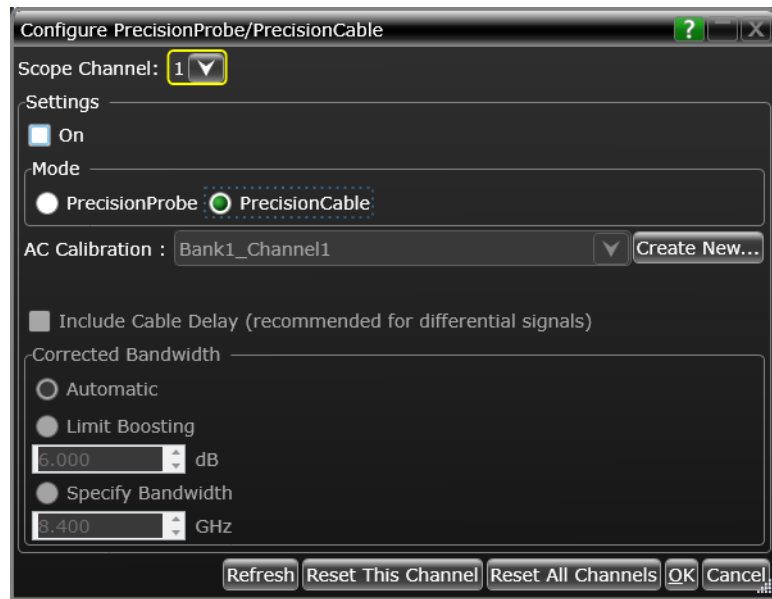
- (Infiniium only) Use the **Tools > Infiniium** menu to access Infiniium features from within the generated app:



When you choose **InfiniiSim**, the following dialog box appears:

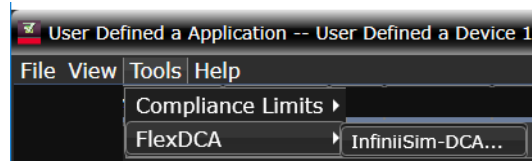


When you choose **PrecisionProbe/PrecisionCable**, the following dialog box appears:

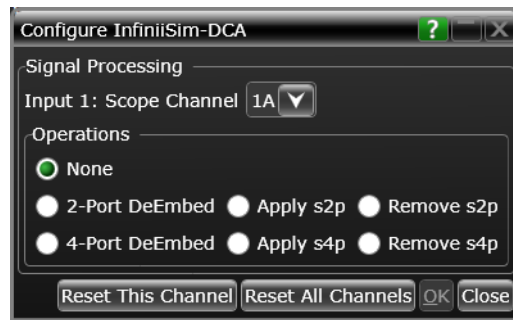


Please refer to the Infiniiium Help system for more information on the controls found in the InfiniiSim and PrecisionProbe dialog box screens.

- (FlexDCA only) Use the **Tools > FlexDCA** menu to access FlexDCA features from within the generated app:



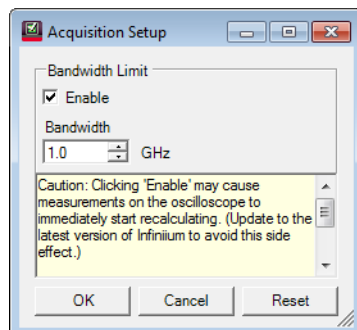
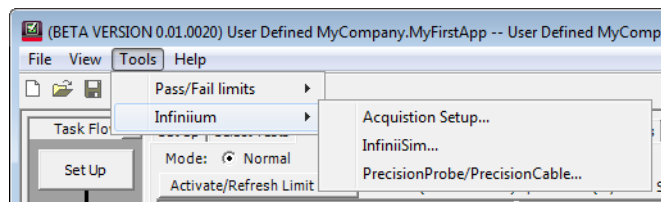
When you choose InfiniiSim-DCA, the following dialog box appears:



Please refer to the FlexDCA Help system for more information on the controls found in the InfiniiSim-DCA dialog box screens.

(Infiniium Only) To access Acquisition Setup in Debug Mode

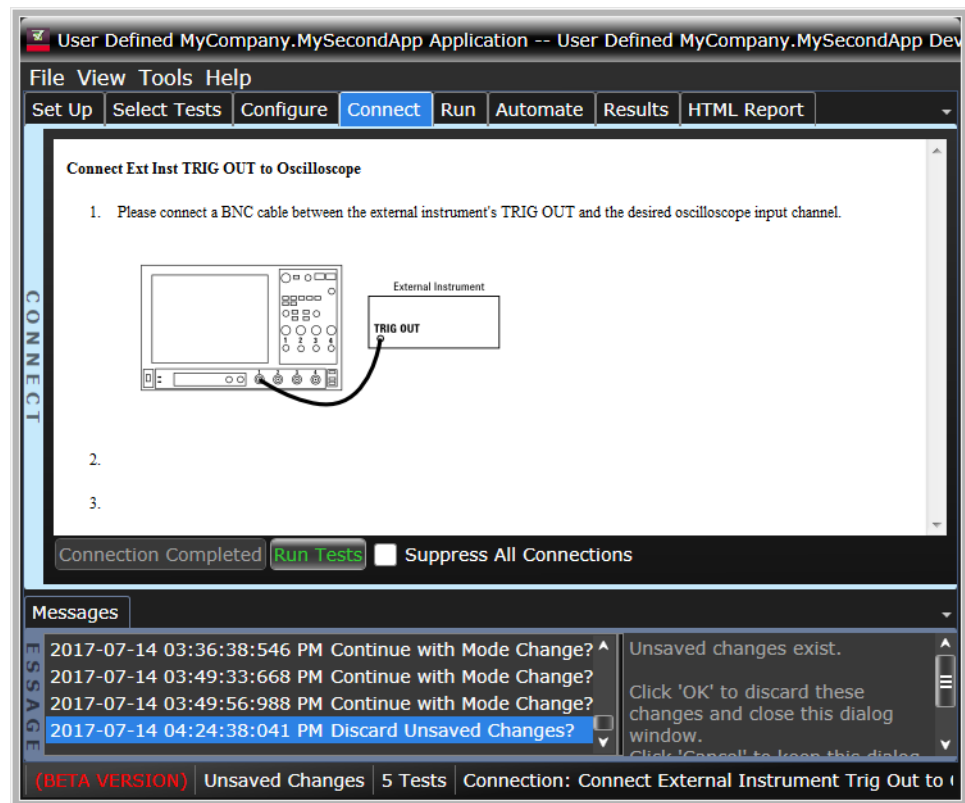
If you place your app in Debug Mode, you will have access to additional Infiniium features via the Tools menu:



Check **Enable** and select a frequency to include Bandwidth Limiting in your next run.

Connecting the Oscilloscope to the DUT

- 1 Click the **Connect** tab.
- 2 Follow the displayed instructions for connecting the oscilloscope to the device under test.
- 3 When connections to the device under test have been made, click **Connection Completed**.



Next · ["Running Tests"](#) on page 268

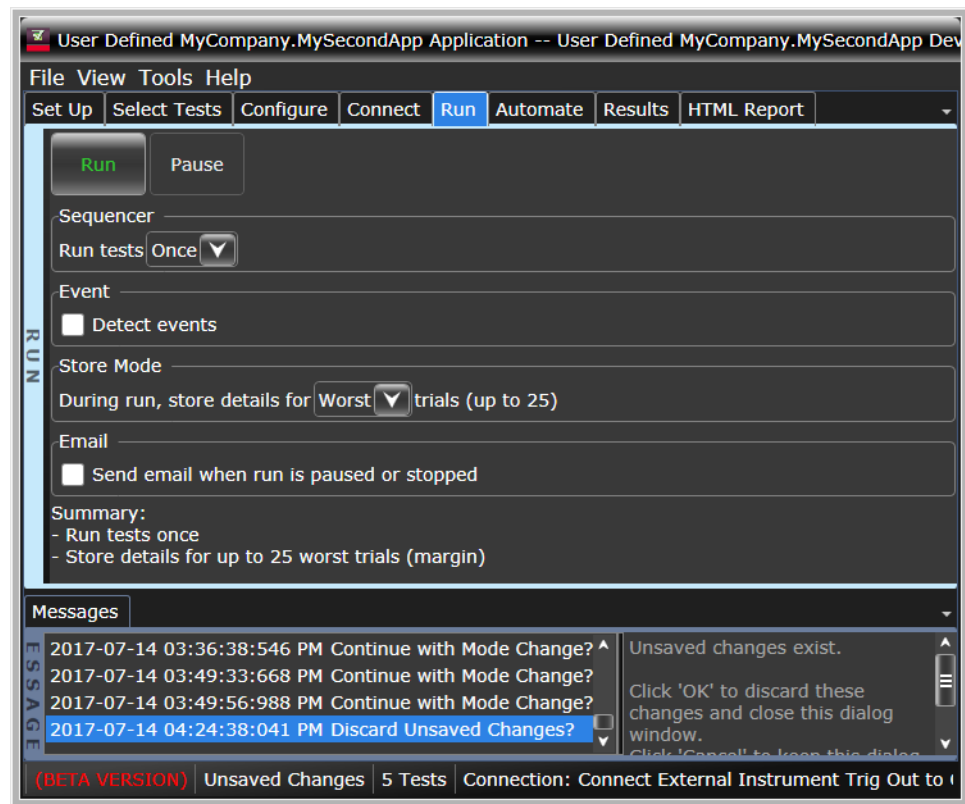
Running Tests

The Run tab's settings let you run the selected tests once or multiple times. When you run tests multiple times, there are options for selecting which trials are stored and how long tests are run.

To run the selected tests once:

- 1 Start the test run.

Select the Run tab, make sure the **Once** "run until" option is selected, and click the big **Run** button.



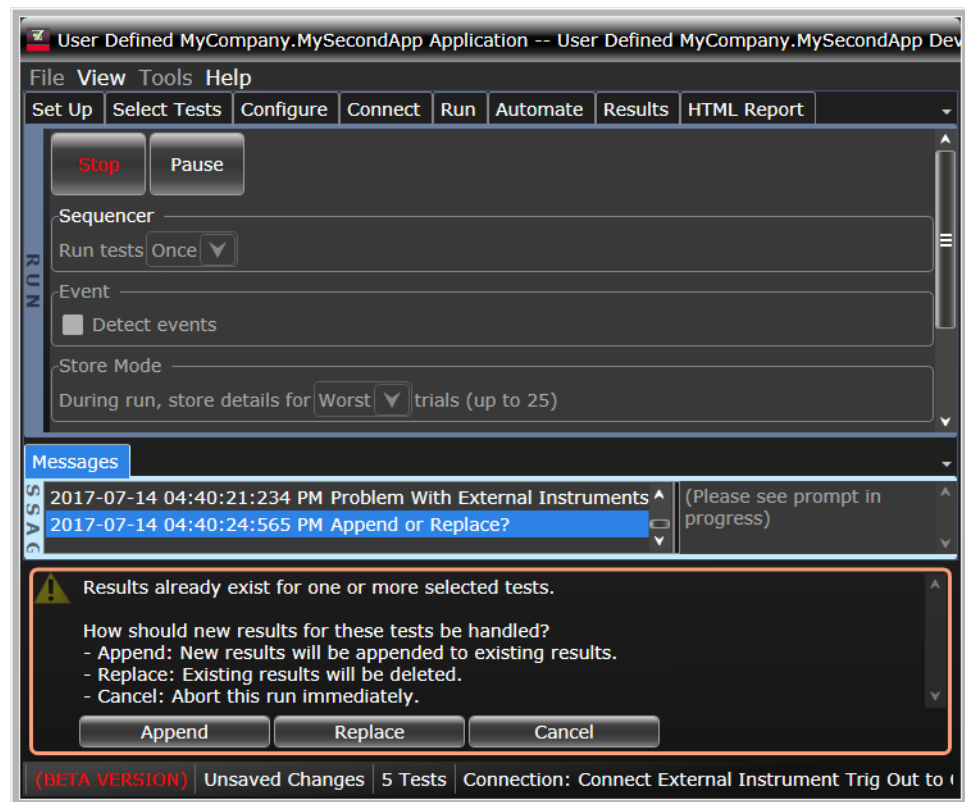
For more information on additional run options, see:

- ["To select the "store mode""](#) on page 270
- ["To run multiple times"](#) on page 271
- ["To send email on pauses or stops"](#) on page 273
- ["To configure result tags"](#) on page 273
- ["To pause or stop on events"](#) on page 288
- ["To specify the event"](#) on page 289

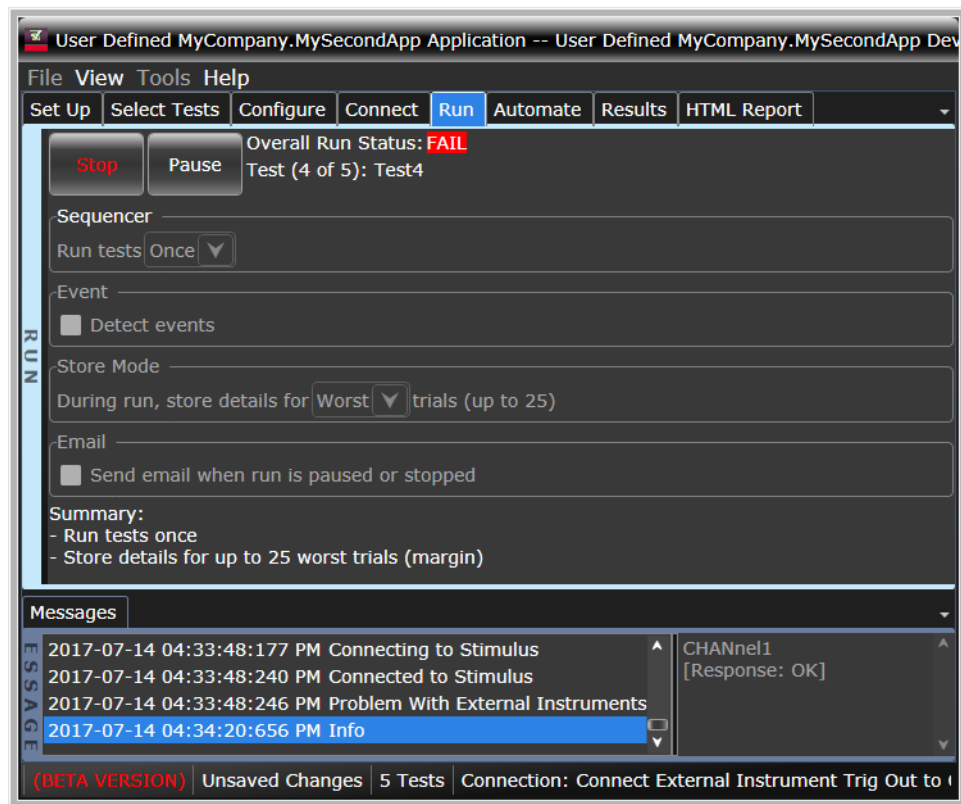
- 2 If there are existing test results, you are asked if you would like to append them or replace them.

If you would like to keep the existing test results to compare against new results, click **Append**.

Click **Replace** if you would like to delete the existing test results.



- 3 While the tests are running, run status appears in the Run tab.



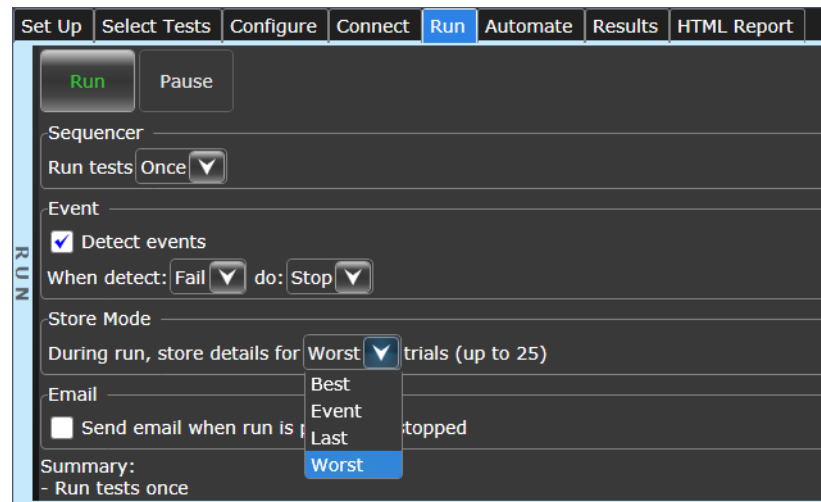
See Also · ["To set the display preferences"](#) on page 290

Next · ["Viewing Results"](#) on page 303

To select the "store mode"

When running tests multiple times, you can select which trials are stored.

- 1 Select the Run tab.
- 2 In the Store Mode area, select:



- **Best** – stores the results of the best N trials.
- **Event** – When the **Detect events** check box is selected, this option stores the results of N trials in which the event is detected. The event is determined in the Event area. See ["To specify the event"](#) on page 289.
- **Last** – stores the results of the last N trials.
- **Worst** – stores the results of the worst N trials.

Up to 80 trials can be stored.

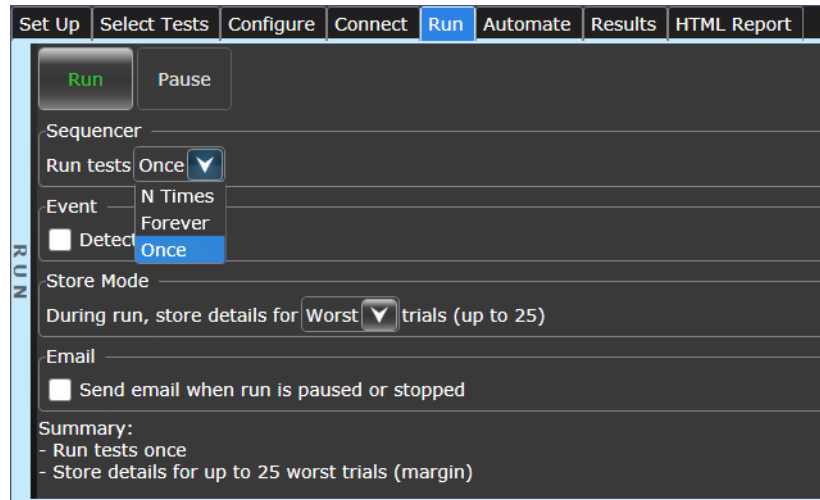
If you change the Store Mode when test results exist, the existing results will be deleted.

The Store Mode selection affects the trial display options in the Report tab of the Preference dialog box. See ["To set trial display preferences"](#) on page 305.

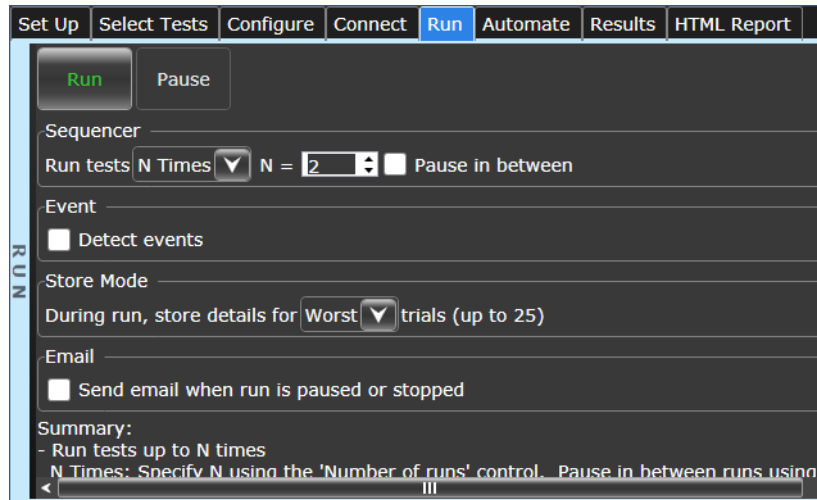
To run multiple times

The "run until" option lets you specify whether tests are run once or multiple times.

- 1 Select the Run tab.
- 2 In the Sequencer area, select:



- **Forever** – runs the tests repeatedly until you click the **Cancel** button.
- **N Times** – runs the tests N times. When this option is selected, you can specify the number of runs and whether pauses occur between each run.



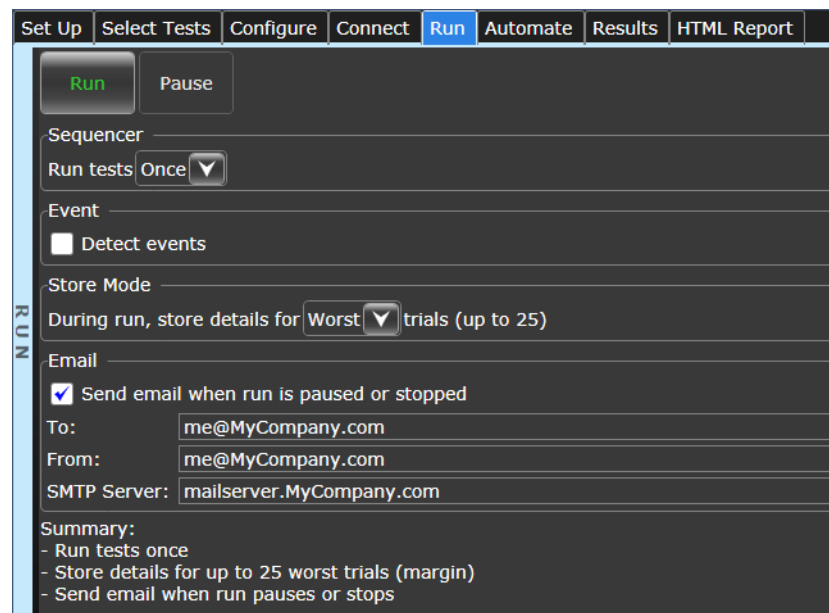
- **Once** – runs the tests only once. This is the default setting.

When multiple runs are selected, you can use the trial display options in the Report tab of the Preference dialog box to specify how many trials are displayed in the test report. See ["To set trial display preferences"](#) on page 305.

To send email on pauses or stops

You can configure the test application to send email whenever a run pauses or ends.

- 1 Select the Run tab.
- 2 In the Email area, check **Send email**.
- 3 Enter your **To** and **From** email addresses and the hostname of the **SMTP Server**.



Pauses can occur between runs when running a specific number of times (see ["To run multiple times"](#) on page 271) or when pausing on an event (see ["To pause or stop on events"](#) on page 288).

To configure result tags

In the **Run** tab, the **Result Tags** section lets you:

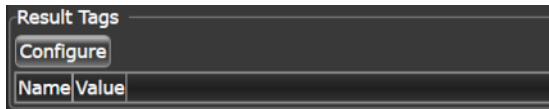
- ["To add result tags"](#) on page 273
- ["To modify result tags"](#) on page 285
- ["To delete result tags"](#) on page 287

To add result tags

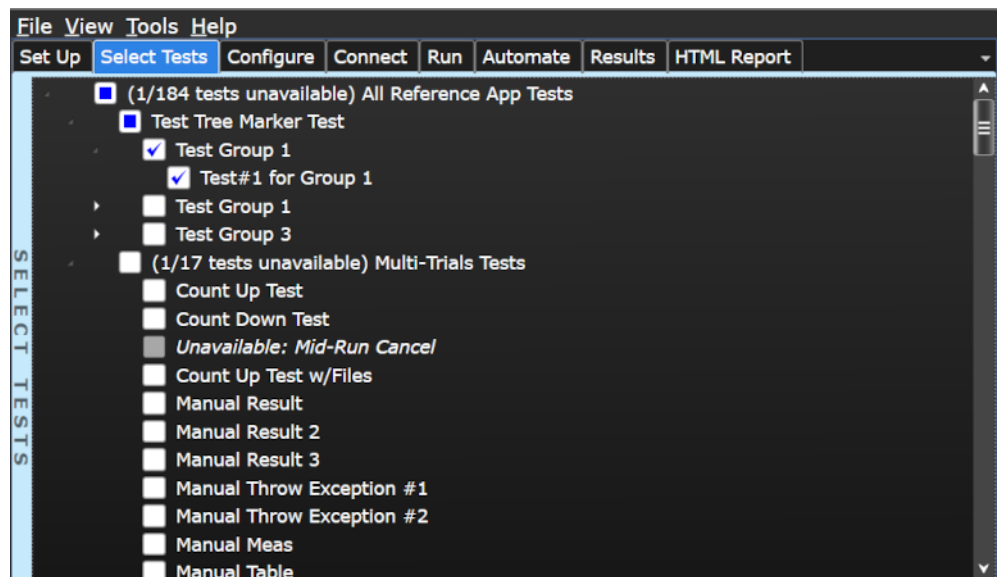
Tags for the test results can be added in the test application:

- Before starting a test run or
- After the completion of a test run

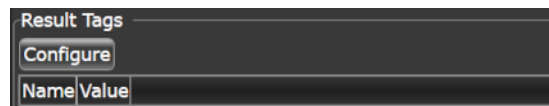
The tags added before starting a test run will be applied automatically to the test results achieved after the test run completion. However, the tags added in the application after a test run completion need to be manually applied to the test results. For more details, please refer to ["To add result tags before starting a test run"](#) on page 274 and ["To add result tags after the completion of a test run"](#) on page 282 sections.

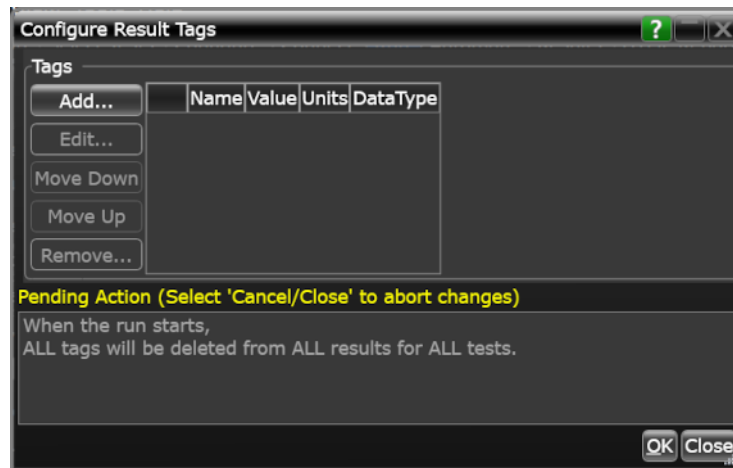


To add result tags before starting a test run 1 In the **Select Tests** tab, please select the required test(s) to run. For example, in the image below, the **Test#1 for Group1** test has been selected.

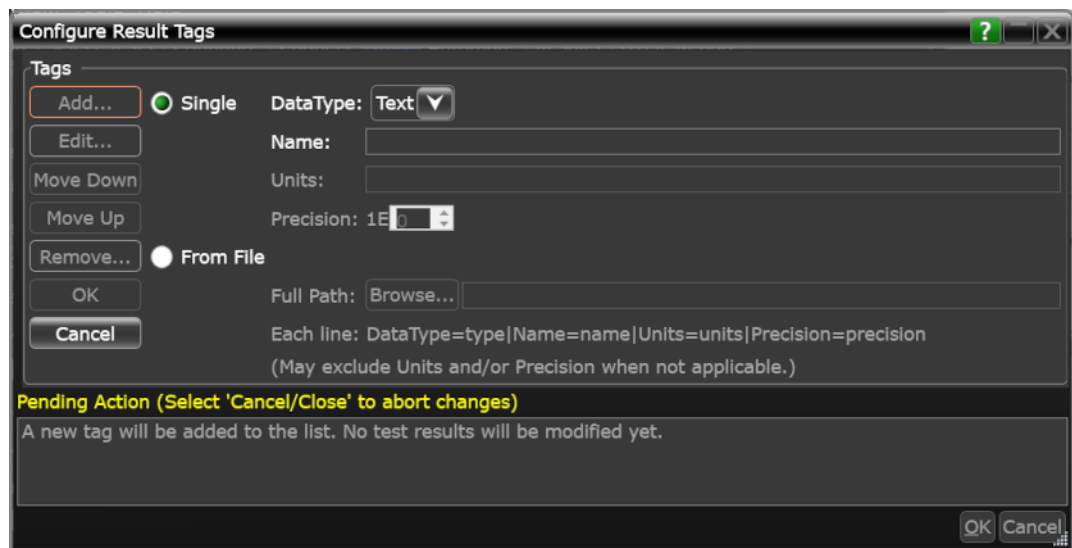


2 In the **Run** tab, under the **Result Tags** section, click the **Configure** button. The **Configure Result Tags** window will appear with the **Tags** section. By default, there are no tags listed under the **Tags** section.



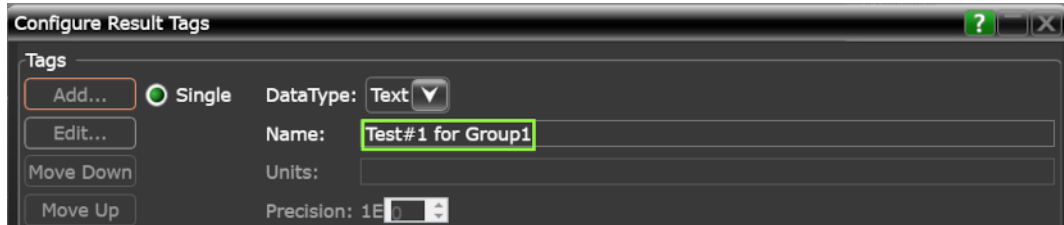


- Under the **Tags** section, click the **Add...** button. The **Single** and **From File** options will appear.

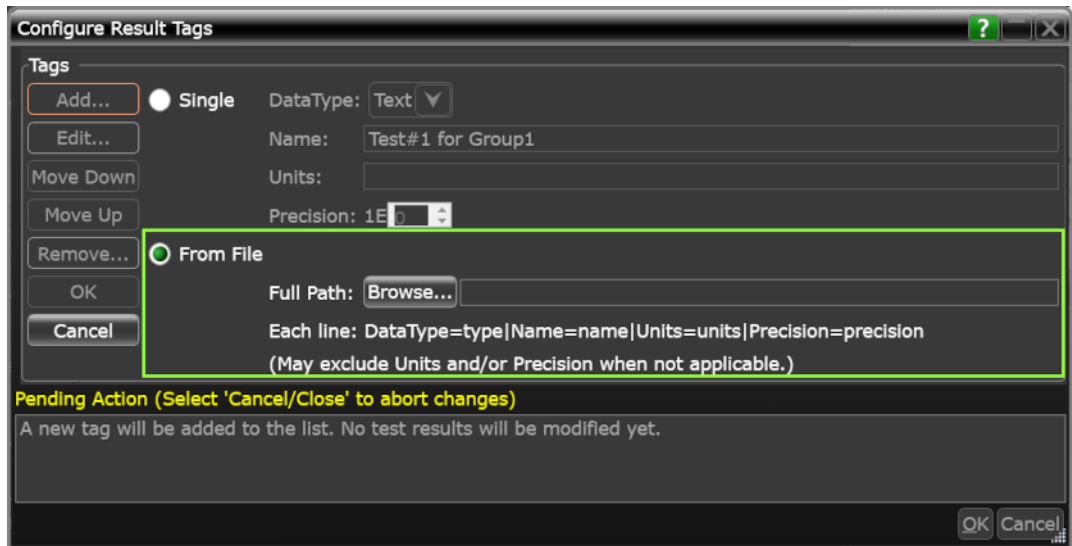


- To add tags, select either **Single** or **From File** option.
 - Single:** This option allows you to add one tag at a time. By default, this option is selected. You can define the following tag properties:
 - DataType:** Defines the type of the tag. By default, the **Text** type is selected. From the drop-down options, select any one of the following option:
 - DateTime** – date and time to the tag, in YYYY-MM-DD hh:mm:ss::nnn format
 - Decimal** – decimal value
 - Integer** – integer value
 - Text** – alphanumeric text value

- **Version** – version number value
- **Boolean** – boolean value (TRUE / FALSE or YES / NO)
- **Name:** Name of the tag. In the text field, enter an appropriate alphanumeric value as the tag name.



- **Units:** This text field gets enabled only if **Decimal** or **Integer** **DataType** is selected. Enter the required unit type in the field.
- **Precision:** This text field gets enabled only for the **Decimal** **DataType**. Select the required numeric precision value from the drop-down box.
- **From File:** For bulk operation, please select the **From File** option, and upload the tags file (containing all the required tags) from the system. It is a .txt or text file.



- In the **Full Path** text field, please type/paste the entire tags text file path or use the **Browse...** button to navigate to the file location, and select the file.

NOTE

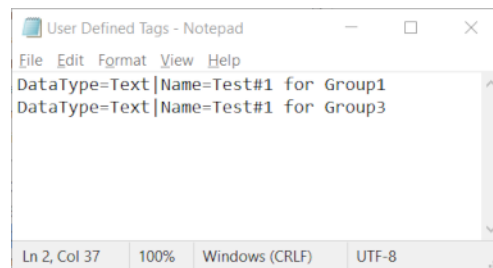
You may use Notepad or any other program to create a text file in the following format:

```

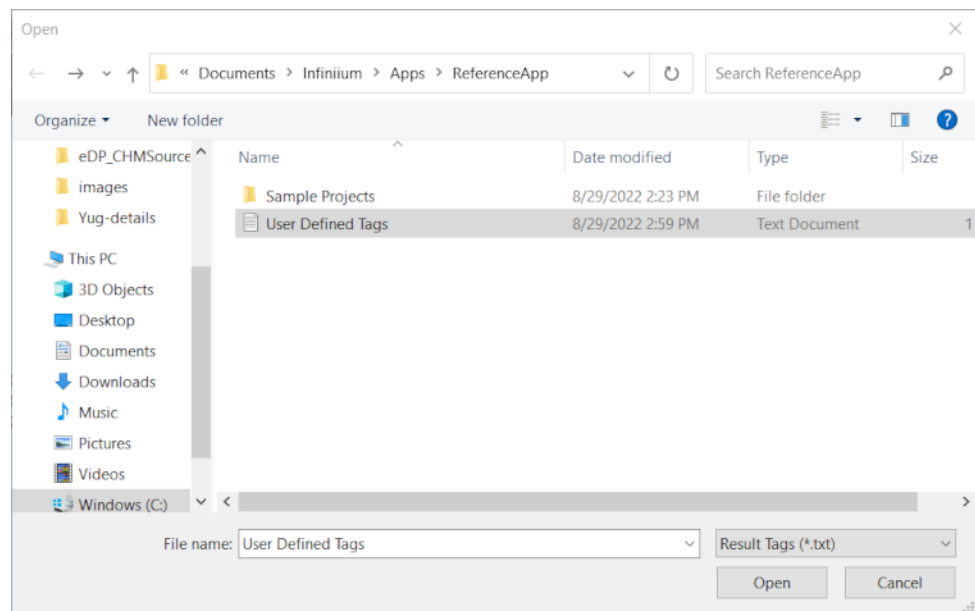
DataType=<type> | Name=<name> | Units=<units> | Precision=<precision>
DataType=<type> | Name=<name> | Units=<units> | Precision=<precision>
.
.
.
DataType=<type> | Name=<name> | Units=<units> | Precision=<precision>

```

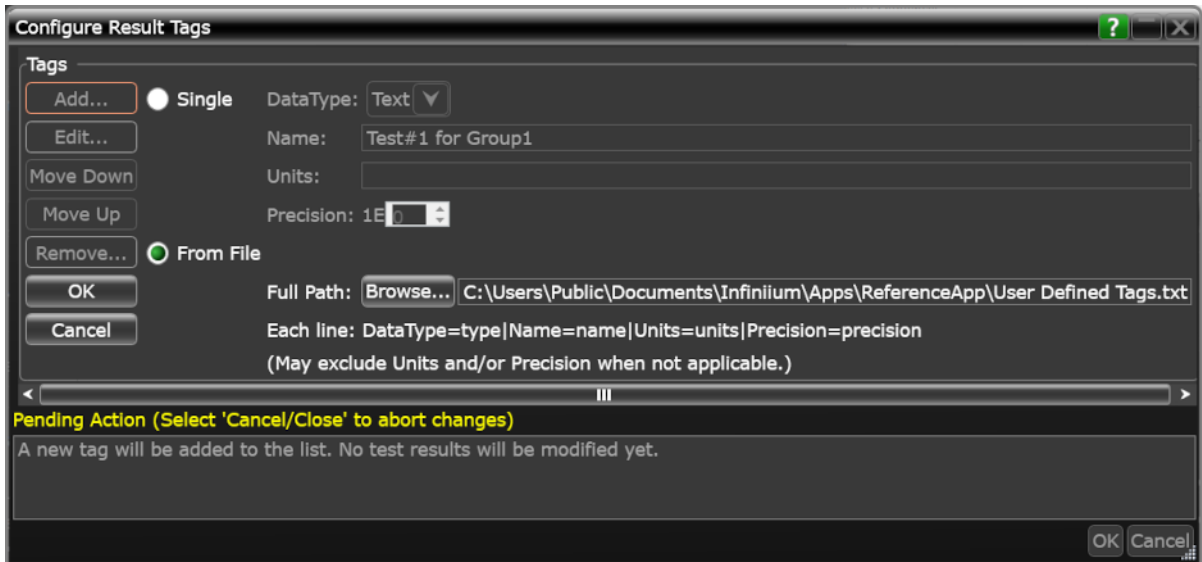
Please remember that each tag property must begin with a new line.



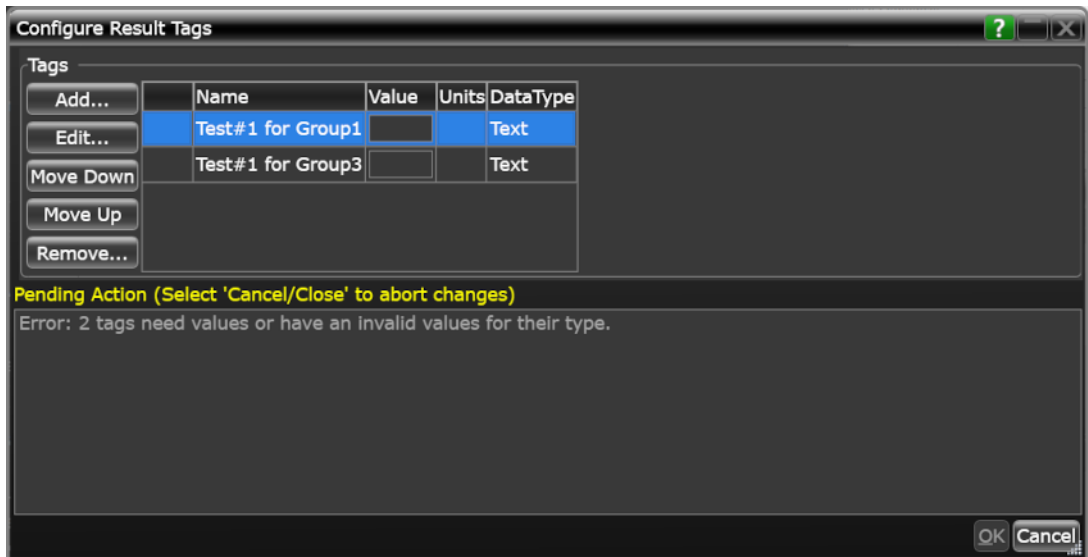
- If you have used the **Browse...** button to navigate to the file location, in the **Open** dialog box, please select the desired file, and click **Open**.



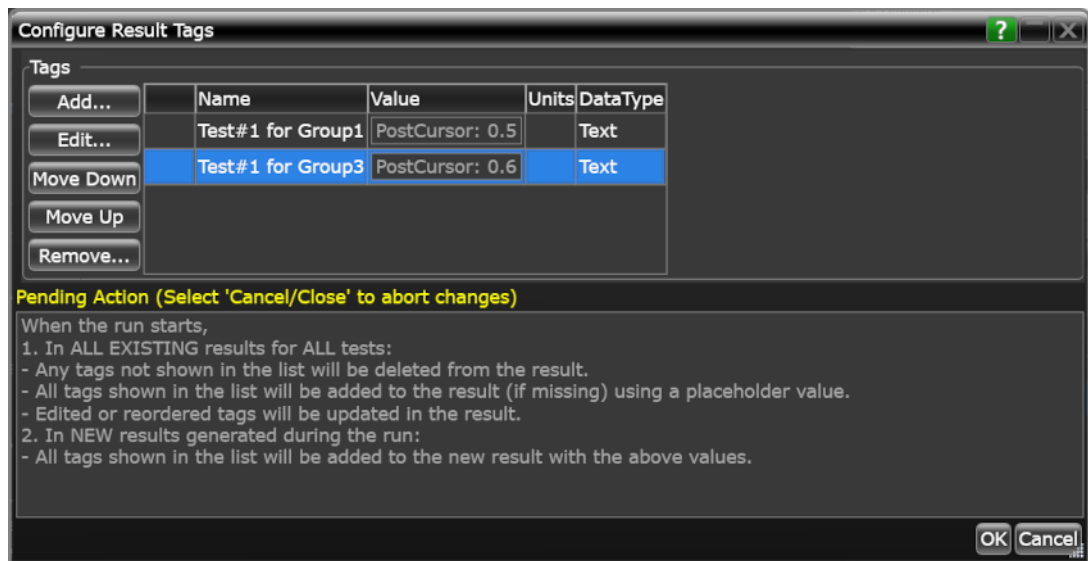
- The **Full Path** text field is populated with the selected file location.



- Once the tags are added using either **Single** or **From File** option, under the **Tags** section, click the **OK** button to save the tags. The tag names are added to the tabular view as shown in the image below.

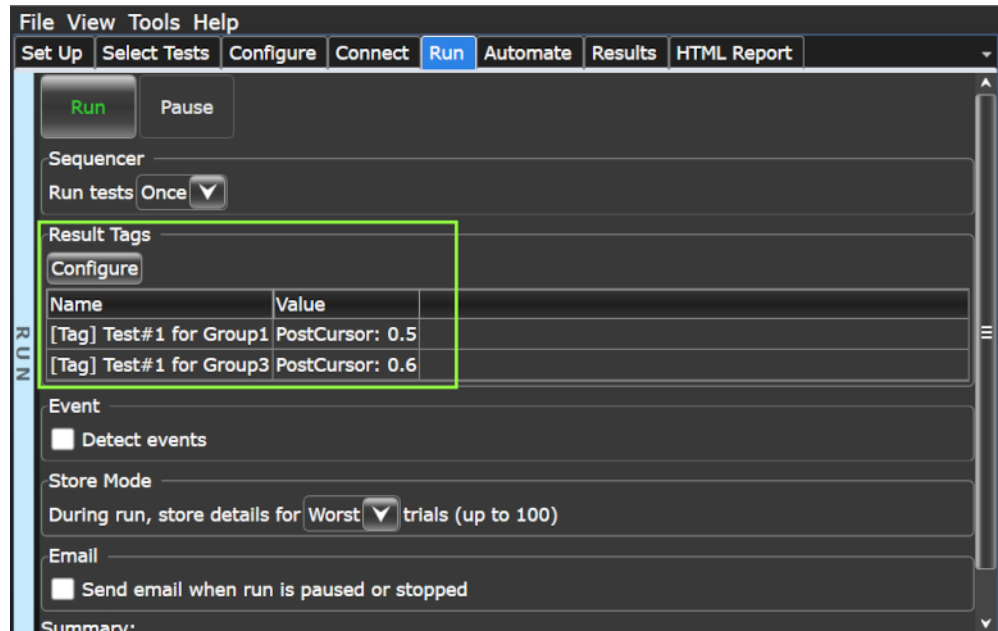


- Define values for the added tag names. Click in the **Value** text box and enter the value in alphanumeric or numeric format.

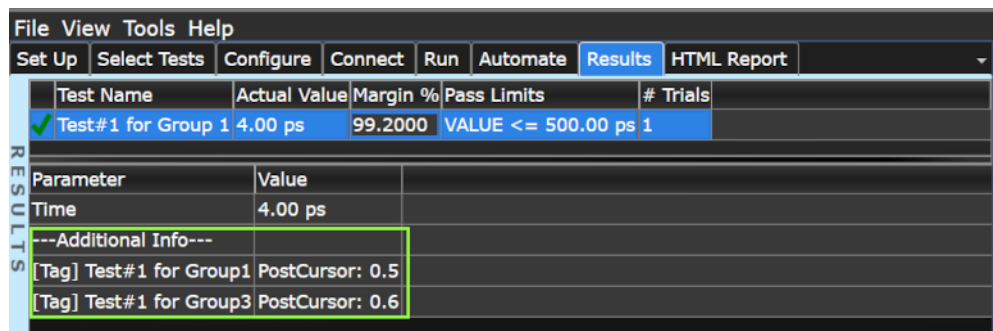


At this point, you may optionally perform one of the following actions in the **Tags** section:

- Click the **Add** button if you wish to add more tags using **Single** or **From File** options. Repeat steps 3 to 7 in this section to add more tags.
 - Click the **Edit** button if you wish to modify the added tags. The tabular view will change to edit mode. Refer to the **"To modify result tags"** on page 285 section.
 - Click the **Remove** button if you wish to remove one or more tags. Refer to the **"To delete result tags"** on page 287 section.
 - Click the **Move Up...** or **Move Down...** to change the order of added tags, in the sequence you wish to apply tags to the test results. For a single tag, these buttons will remain disabled.
- 7 Click the **OK** button in the bottom-right corner of the **Configure Result Tags** window. The window will close and the added tag names and associated values are listed under the **Result Tags** section in the **Run** tab.



- 8 Once the tags are added in the **Configure Result Tags** window using the **Configure** button, then on performing a test run, the test application will apply the tags to the test results in the following manner:
- For the **new test results** that were generated after the test run completion:
 - The tags available in the **Configure Result Tags** window will be applied to the test results along with their defined values. For example, in step 1, the **Test#1 for Group 1** test was selected for a test run, therefore the tag is applied to its results (as shown in the image below).



- For all the test results existing in the **Results** tab before starting a test run:
 - The tags not available in the **Configure Result Tags** window will be deleted from all the test results.

- The tags available in the **Configure Result Tags** window will be added to all the test results, with **N/A** values indicating that tags are not applicable. You can modify these tags later, if required. To modify tags, refer to **"To modify result tags"** on page 285 section.

Test Name	Actual Value	Margin %	Pass Limits	# Trials
Test#1 for Group 1	4.00 ps	99.2000	VALUE <= 500.00 ps	4
Test#1 for Group 3	0 mV	50.0000	-2 mV < VALUE < 2 mV	4
Multi-Trials Measurements Table	Pass	100.000	Pass/Fail	2

Trial Summary	Actual Value	Margin	Parameter	Value
Completed: 4	Min 4.000 ps	99.20 %	Time	4.00 ps
Passed: 4	Max 4.000 ps	99.20 %	--- <td></td>	
Failed: 0	Sum 16.00 ps	396.8 %	[Tag] Test#1 for Group1	N/A
Worst: 4	Trial 4 4.00 ps	99.2000%	[Tag] Test#1 for Group3	N/A
Max Displayed: 10	Trial 3 4.00 ps	99.2000%		
	Trial 2 4.00 ps	99.2000%		

- Additionally, the tags applied to the test results are also displayed in the test results available in the **HTML Report** tab. The highlighted section in the image below shows an example of a tag added to the test results.

Report Detail

[Next](#)

✓ **Test#1 for Group 1** Reference:

Test Summary: **Pass** Test Description: Connection page test

Pass Limits: <= 500.00 ps Time: 4.00 ps

Result Details

[Tag] Test#1 for Group1 PostCursor: 0.5 [Tag] Test#1 for Group3 PostCursor: 0.6

[Top](#) [Previous](#)

[Next](#)

✓ **Test#1 for Group 3** Reference:

Test Summary: **Pass** Test Description: Connection page test

Pass Limits: (-2 mV to 2 mV) Test#1forGroup3Result: 0 mV

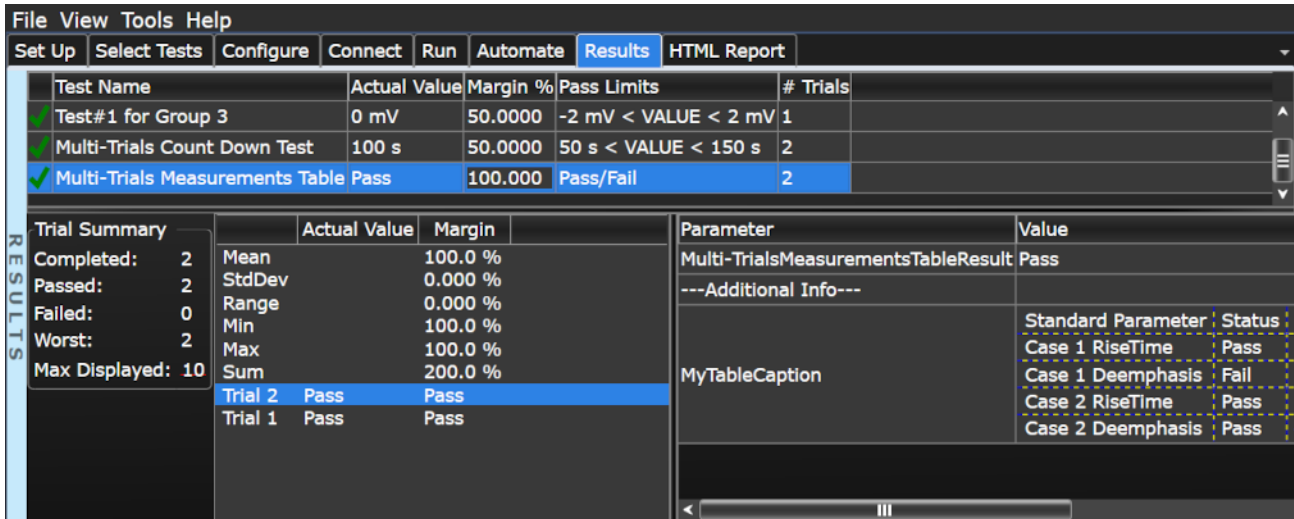
Result Details

[Tag] Test#1 for Group1 PostCursor: 0.5 [Tag] Test#1 for Group3 PostCursor: 0.6

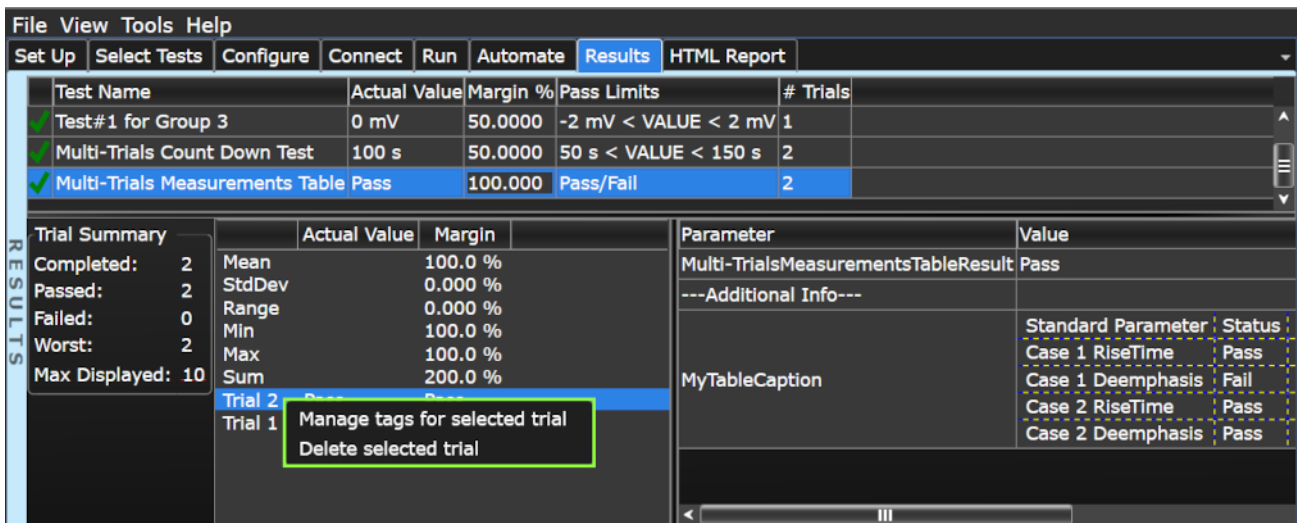
[Top](#) [Previous](#)

To add result tags after the completion of a test run If you add tags after a test run completion, then you need to manually apply tags to the test results. To do so, perform the following steps:

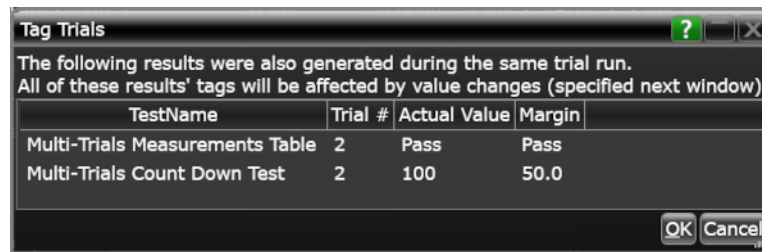
- 1 In the **Results** tab's upper pane, select the test result for which you wish to add a tag. For example, in the image below, the **Multi-Trials Measurement Table** result is selected in the upper pane and its associated results (trials) are displayed in the lower pane.



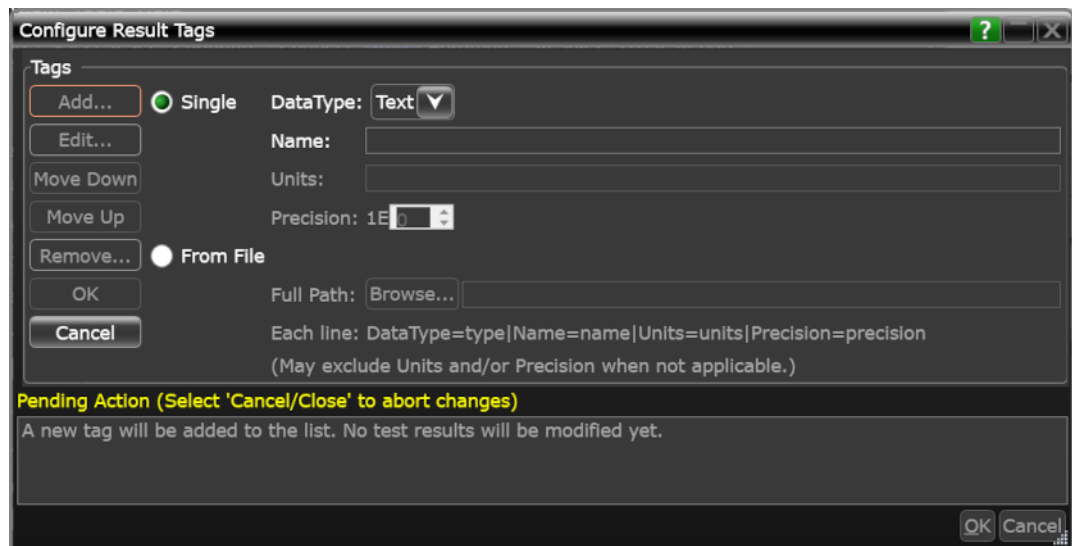
- 2 To add a tag to a specific trial, right-click in the lower pane after selecting the required trial. Click **Manage tags for selected trial** option.



- 3 The **Tag Trials** window appears which shows the tests that are a part of that trial.



- 4 Click **OK** to open the **Configure Result Tags** dialog box.



- 5 To add tags, repeat steps 3 to 7 of the **"To add result tags before starting a test run"** on page 274 section, and click **OK**. The application will apply the tags to the test results in the following manner:
- For all the tests that are a part of the selected tests' trial:
 - The tags available in the **Configure Result Tags** window will be added to the test result along with the associated values.

Test Name	Actual Value	Margin %	Pass Limits	# Trials
Test#1 for Group 3	0 mV	50.0000	-2 mV < VALUE < 2 mV	1
Multi-Trials Count Down Test	100 s	50.0000	50 s < VALUE < 150 s	2
Multi-Trials Measurements Table	Pass	100.000	Pass/Fail	2

Trial Summary	Actual Value	Margin	[Tag] Test#1 for	Parameter	Value	
Completed: 2	Mean	100.0 %		Multi-TrialsMeasurementsTableResult	Pass	
Passed: 2	StdDev	0.000 %		---Additional Info---		
Failed: 0	Range	0.000 %		[Tag] Test#1 for Group1	PreCursor: 0.1	
Worst: 2	Min	100.0 %		[Tag] Test#1 for Group3	PreCursor: 0.2	
Max Displayed: 10	Max	100.0 %				
	Sum	200.0 %				
	Trial 2	Pass	Pass		Standard Parameter	
	Trial 1	Pass	N/A		Case 1 RiseTime	Pass
					Case 1 Deemphasis	Fail
					Case 2 RiseTime	Pass
					Case 2 Deemphasis	Pass

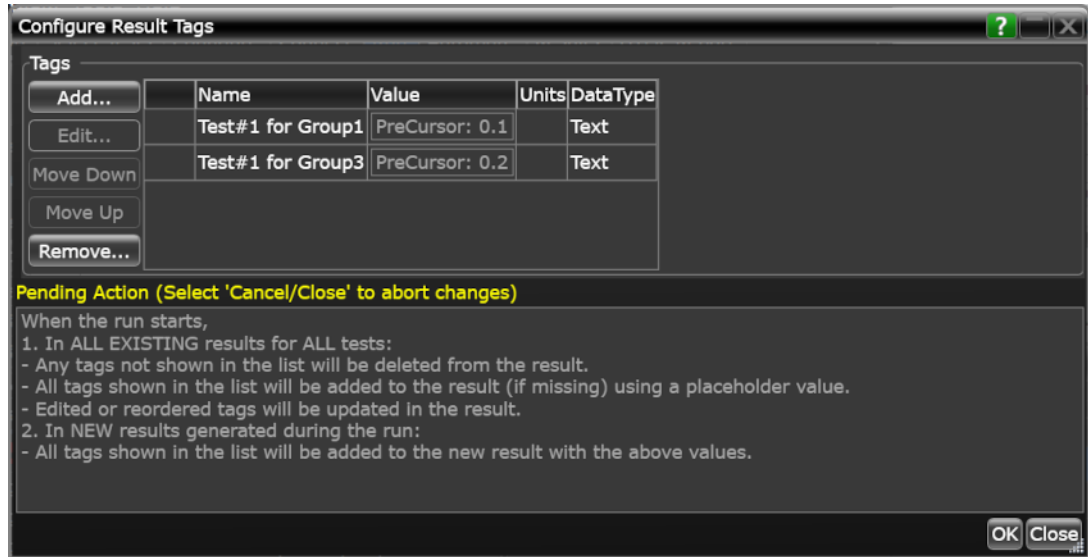
- For all other results/trials:
 - The tags not available in the **Configure Result Tags** window will be deleted from all the test results.
 - The tags available in the **Configure Result Tags** window will be added to all other test results/trials with N/A values indicating that the tags are not applicable to the test results/trials.

Test Name	Actual Value	Margin %	Pass Limits	# Trials
Test#1 for Group 3	0 mV	50.0000	-2 mV < VALUE < 2 mV	1
Multi-Trials Count Down Test	100 s	50.0000	50 s < VALUE < 150 s	2
Multi-Trials Measurements Table	Pass	100.000	Pass/Fail	2

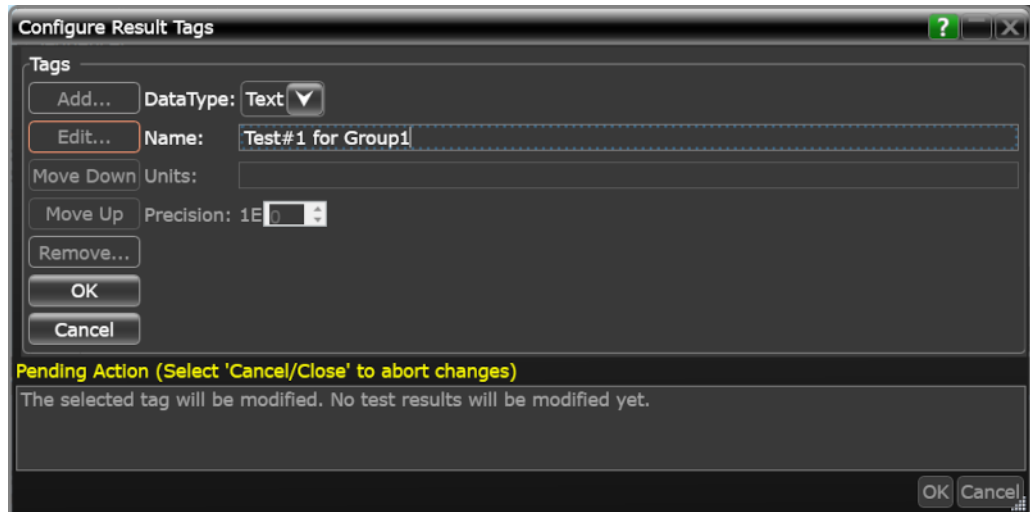
Parameter	Value
Test#1forGroup3Result	0 mV
---Additional Info---	
[Tag] Test#1 for Group1	N/A
[Tag] Test#1 for Group3	N/A

To modify result tags

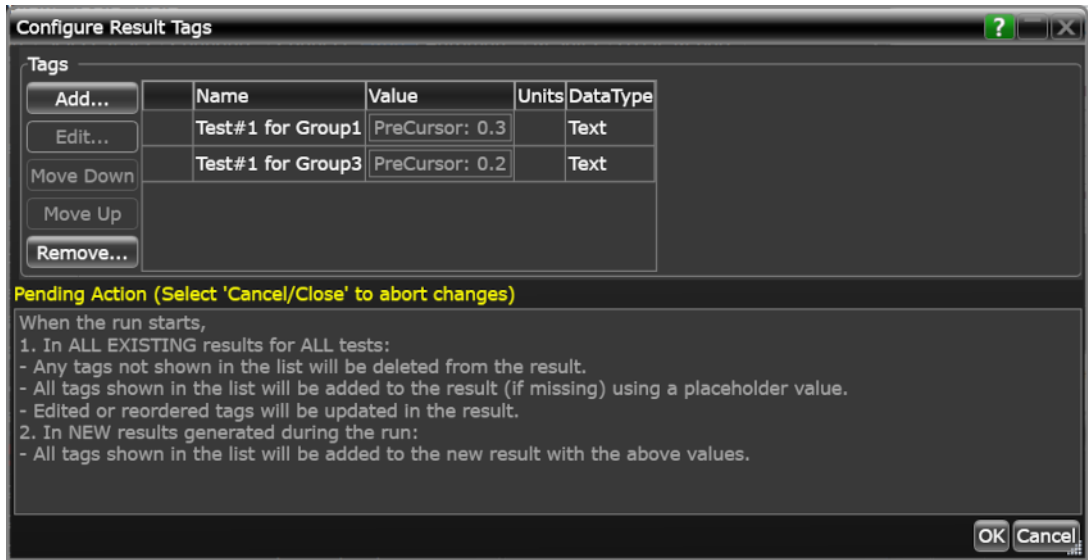
- 1 In the **Run** tab, under the **Result Tags** section, click the **Configure** button. The Tags section will appear with the list of available tags.



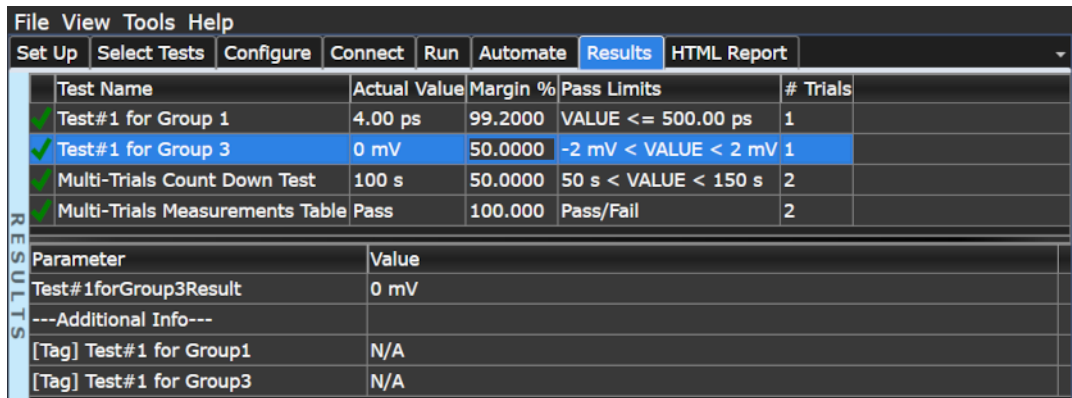
- 2 Click on the tag row that you wish to edit (as highlighted in the above image) and click the **Edit** button. The selected tag will open in the edit mode.



- 3 Edit the desired tag properties and click the **OK** button to save the changes or click **Cancel** button to discard the changes. The **Tags** section will appear with the list of updated tags. You can modify the tag values by clicking in the Value text box, if required.



- Alternatively, you can modify the applied tags in the **Results** tab, to do so:
 - In the **Results** tab's upper section, select the test name. The tag applied to that test result will display in the Additional Info section (as shown in the image below).



- To modify the applied tag, in the lower pane, right-click on the tag. The **Manage tags for selected trial** and **Delete selected trial** menu options will display.

Test Name	Actual Value	Margin %	Pass Limits	# Trials
Test#1 for Group 1	4.00 ps	99.2000	VALUE <= 500.00 ps	1
Test#1 for Group 3	0 mV	50.0000	-2 mV < VALUE < 2 mV	1
Multi-Trials Count Down Test	100 s	50.0000	50 s < VALUE < 150 s	2
Multi-Trials Measurements Table	Pass	100.000	Pass/Fail	2

Parameter	Value
Test#1forGroup3Result	0 mV
---Additional Info---	
[Tag] Test#1 for Group1	N/A
[Tag] Test#1 for Group3	Manage tags for selected trial Delete selected trial

- Click the **Manage tags for selected trial** option. Tag Trials dialog box appears, showing a list of test results that are a part of the same trial as the selected test. The tags and subsequent tag values will be applied to all these tests listed in the Tag Trials dialog box.

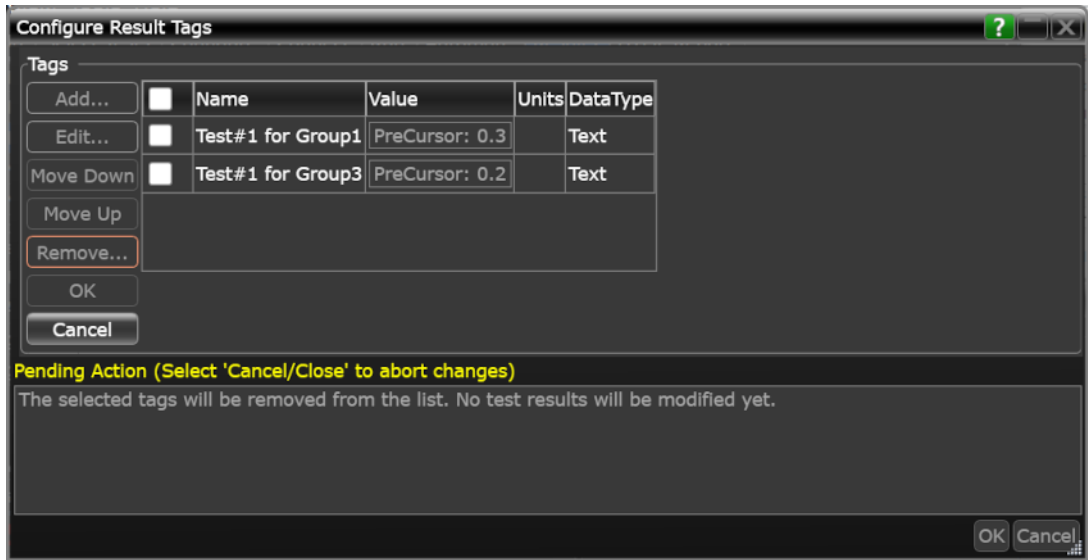
TestName	Trial #	Actual Value	Margin
Test#1 for Group 3	1	0	50.0
Test#1 for Group 1	1	4E-12	99.2

- Click **OK** to proceed. The **Configure Result Tags** window will appear with the list of available tags. To modify tags, repeat steps 2-3 of the "**To modify result tags**" on page 285 section. The modified tags will be updated for the selected test trial.

To delete result tags

To delete the result tags that you added in the application:

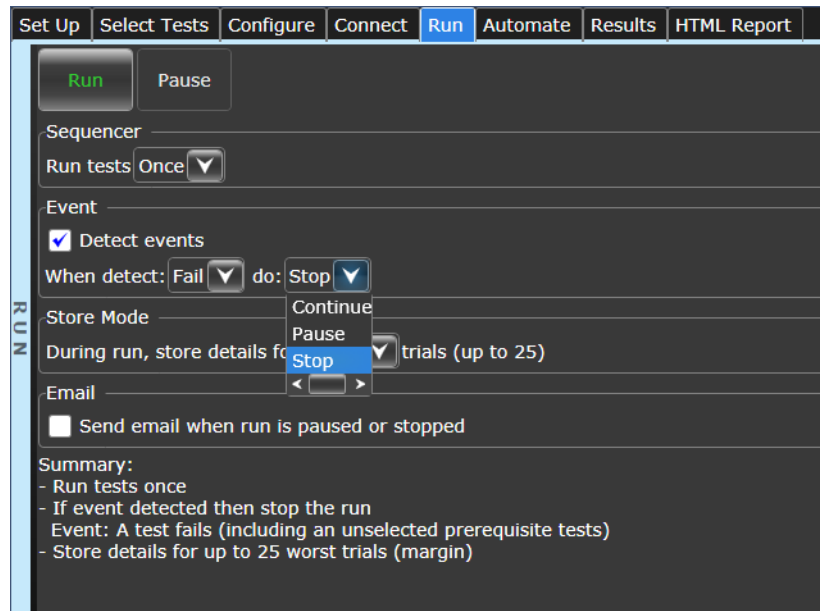
- In the **Run** tab, under the **Result Tags** section, click the **Configure** button. The **Configure Results Tags** window will appear the list of available tags.
- Under the **Tags** section, click the **Remove...** button. Select one or more tags using the available check boxes. Click **OK** to delete the selected tags.



To pause or stop on events

You can set up test runs to pause or stop on events which are checked at the end of each test..

- 1 Select the Run tab.
- 2 In the Run Until area, check **On event**.
- 3 In the drop-down selection field that appears, select either:



- **Continue** – causes the run to continue on to the next selected test.

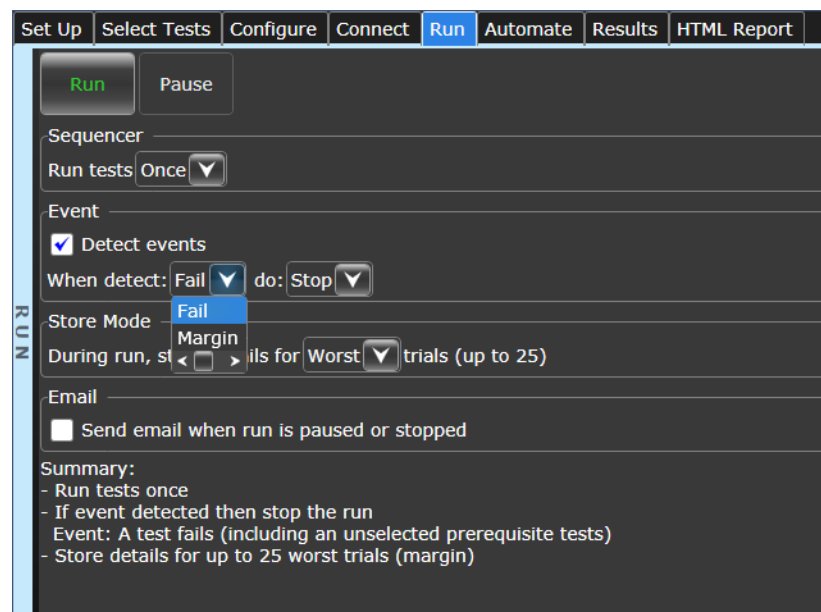
- **Pause** – causes the run to pause when the event is detected.
 - **Stop** – cause the run to stop when the event is detected.
- 4 In the Event area, specify the type of event. See **"To specify the event"** on page 289.

Pauses or stops can be set up to automatically send email (see **"To send email on pauses or stops"** on page 273).

To specify the event

In the Store Mode area when you have selected Event (see **"To select the "store mode"** on page 270) or in the Run Until area when you have selected to pause or stop on an event (see **"To pause or stop on events"** on page 288), the Event area appears so that you can specify the event.

- 1 In the Event area, select the type of event:



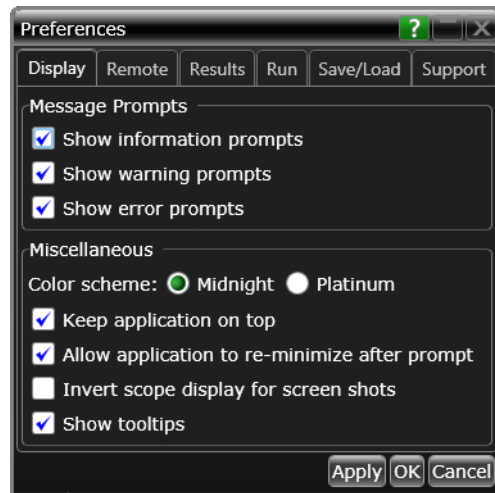
- **Fail** – causes the event to fire when a prerequisite test or selected test fails..
- **Margin < N** – causes the event to fire when a test generates a margin < specified. When this option is selected, enter the minimum required margin percentage.
- **Pass** – causes the event to fire when a test passes (excluding prerequisite tests).

A tilde "~" character in the event selection drop-down shows that the event is unavailable. If you select an event type that is not available, a dialog box tells you why.

To set the display preferences

Information, warning, and error conditions can occur while running tests. The display preferences let you choose whether message dialog boxes are shown. And, there are other display preferences that affect what happens as tests are run.

- 1 From the generated application's main menu, choose **View > Preferences....**
- 2 In the Preferences dialog box, select the **Display** tab.



- 3 In the Display tab, you can choose to show the following types of message dialog boxes:
 - Information dialogs.
 - Warning dialogs.
 - Error dialogs.

NOTE

Messages that require you to make a choice, such as "OK/Cancel" and "Yes/No" are always enabled.

- 4 Also, you can choose to:
 - Change the **Color scheme** – You can choose between the darker **Midnight** color scheme or the lighter **Platinum** color scheme.
 - **Keep application on top** – Always keep the application's main window on the top of the Infiniium application. Note that the mid-run dialog boxes are always displayed on the top.
 - **Allow application to re-minimize after prompt** – When running, the application will minimize except when there are prompts.
 - **Invert scope display for screen shots** – (white background) when the application captures the screen shots.

- **Show tooltips** – By enabling this option, the tooltips appear as you move the pointer over various controls in the application.
- 5 Click **Apply** to save the changes, or click **OK** to save the changes and close the Preferences dialog box.

Config Variable Auto-Iteration

Some automated test applications have configuration variables that allow you to simultaneously select more than one choice.

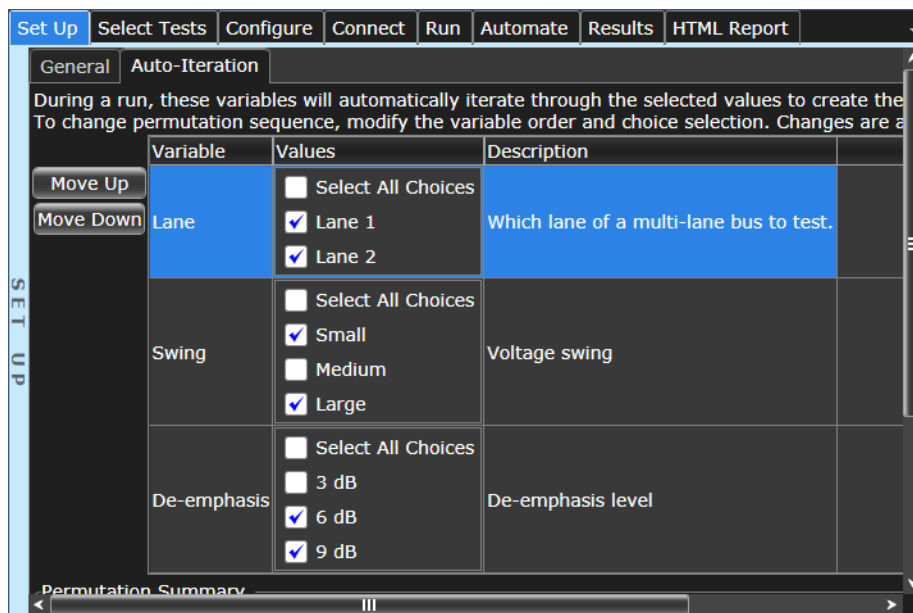
NOTE

Add-Ins do not support auto-iteration.

When these types of variables exist, the Set Up tab has these sub tabs:

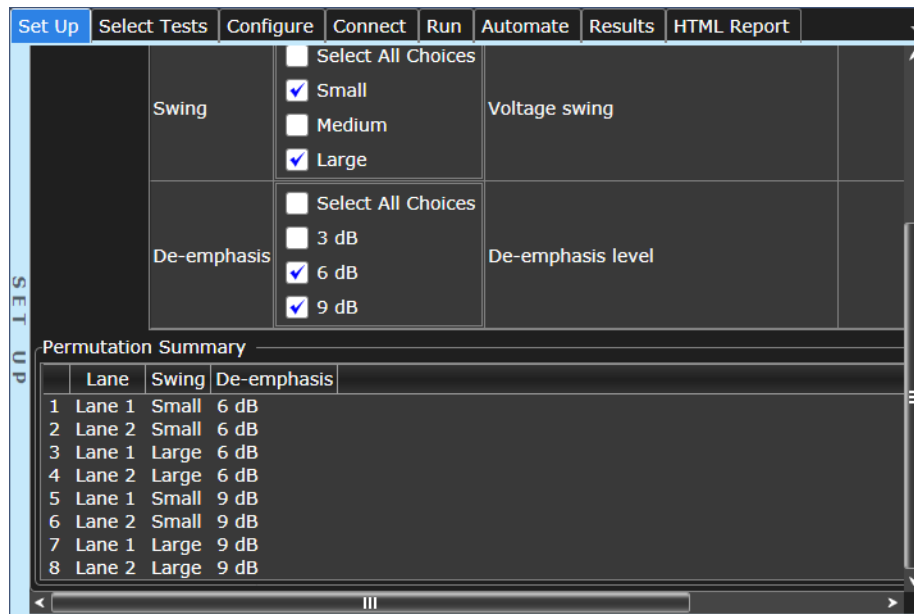
- **General** – This sub tab will contain user comments and, if supported by the application, test filters and external instrument controls.
- **Auto Iteration** – This sub tab is for managing what happens at run time when you select more than one option for each multi-select configuration variable.

In the **Auto Iteration** sub tab, each multi-select configuration variable is presented in a table.



In the above example, the application has three multi-select configuration variables: Lane, Swing and De-emphasis. The choices selected for the current run are shown.

The **Permutation Summary** shows how the run engine will iterate through the selected choices:



When you start a run, the selected tests will be run N times. In this example, for the first trial the configuration variables will be set to these values:

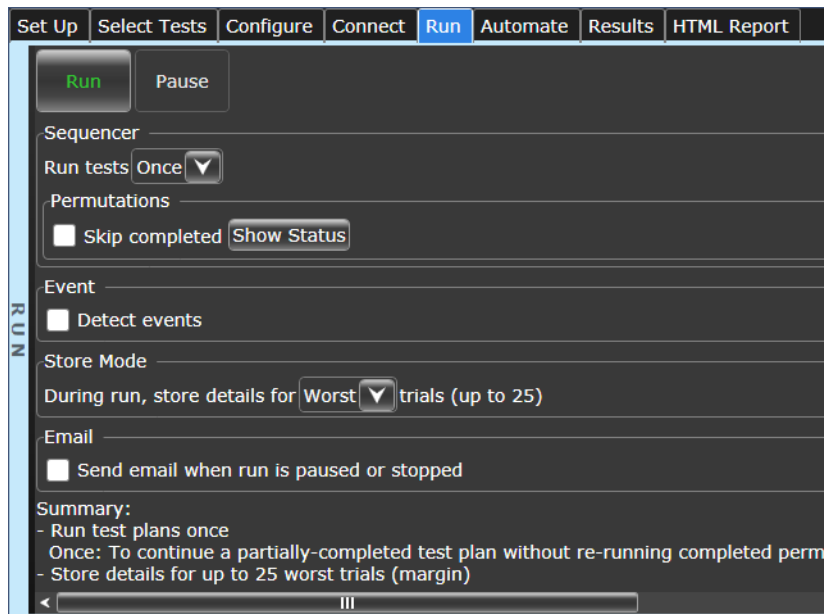
```
Lane = Lane 1
Swing = Small
De-emphasis = 6 dB
```

Then for each subsequent trial, the variables are updated using a counter algorithm in which the furthest right config is iterated first and the furthest left config is iterated last.

You can change the iteration sequence by selecting a variable and clicking the **Move Up** and **Move Down** buttons. When you move a selected variable up or down in the sequence, the rows in the variable table will re-order and the permutation summary will be recalculated.

The iteration sequence may be important if some of the multi-select configs affect physical connections, such as Lane. By sequencing appropriately, you can minimize the number of connection prompts that occur during the run.

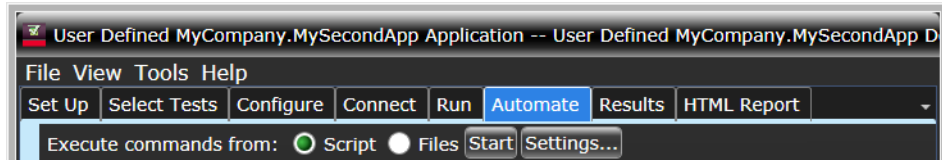
When auto-iteration is enabled, the Run tab has a **Permutations** group with controls for skipping completed permutations and showing their status.



Automating the Application

The Automation tab lets you construct command scripts that drive execution of the application.

You can select from two modes, **Script** and **Files**:

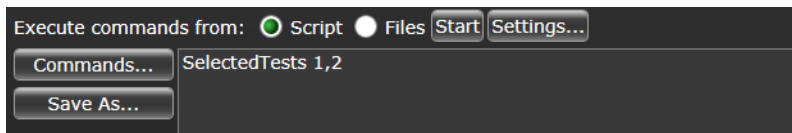


See:

- ["To enter commands in Script mode"](#) on page 295
- ["To enter commands in Files mode"](#) on page 298
- ["To begin Script or Files execution"](#) on page 298
- ["To display automation settings status"](#) on page 299
- ["To try a command line"](#) on page 300

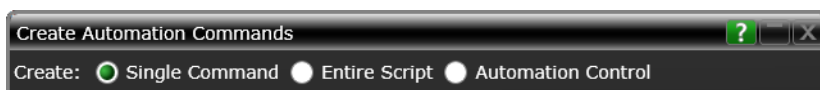
To enter commands in Script mode

In Script mode, the application will execute the commands it finds in the script text box:

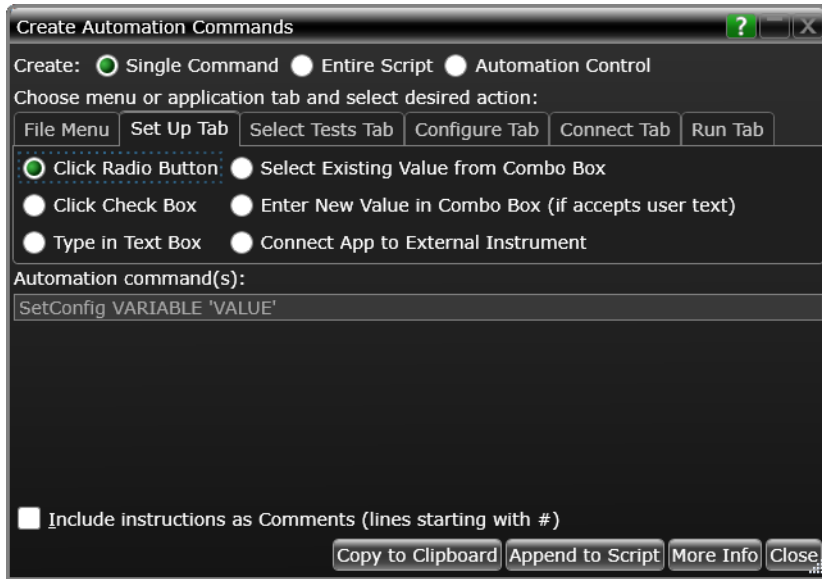


You may either type in commands (one per line) or use the **Commands...** generator to construct them. A script may be saved to a text file to make it easy to switch to the **Files** automation method.

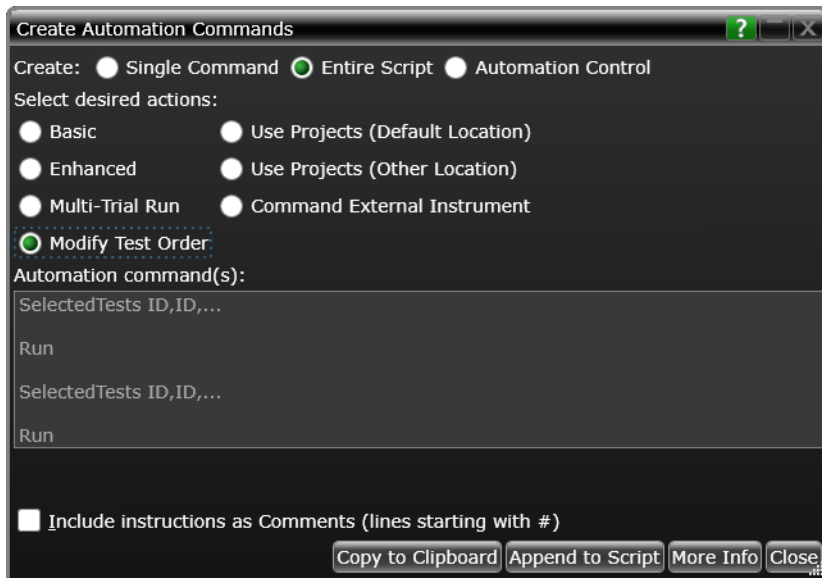
The command generator is opened by clicking **Commands...** It can generate single commands or sample scripts that demonstrate how commands work together.



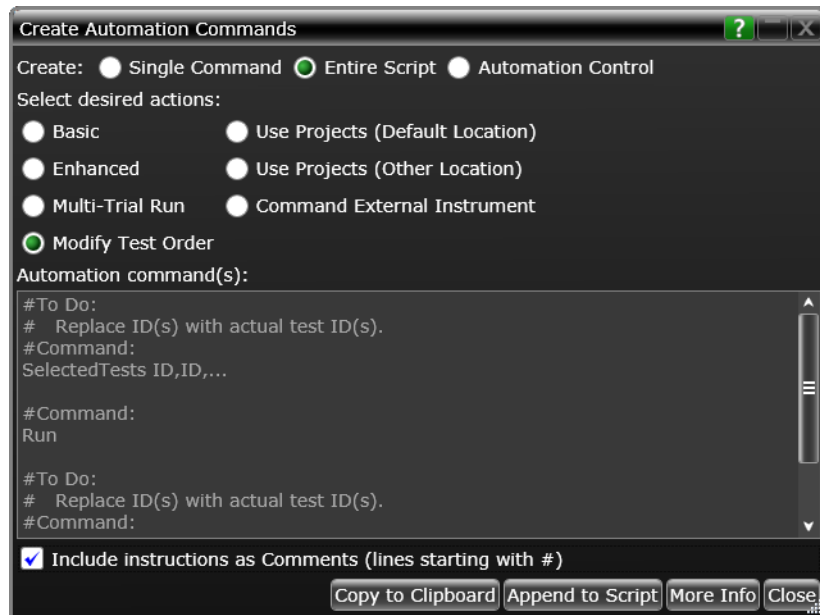
In **Single Command** mode, select the tab that corresponds to the real application tab containing the controls you want to affect. Then, select which action you want to perform. The screen will then show you what automation command performs that action. Most commands are shown with placeholder parameters along with instructions on how to customize them:



In **Entire Script** mode, select a script that describes a task you would like to know more about:



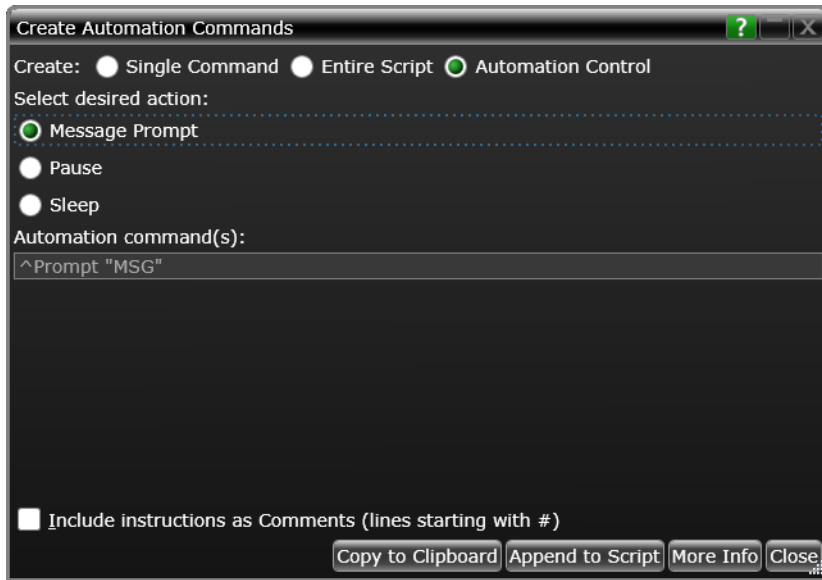
Once you have found the command or script you want, you may copy it to the clipboard or automatically append it to the script in the Automation tab. You may optionally include instructions to guide you in customizing the commands:



NOTE

In the Automation tab script, lines starting with the pound symbol (#) do not execute.

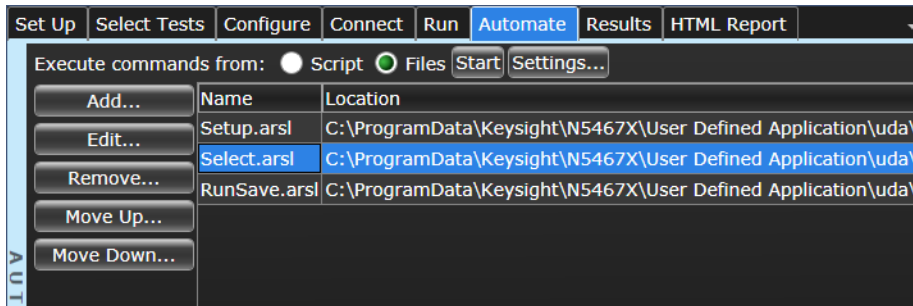
In **Automation Control** mode, select a control command to see its syntax:



See Also · ["To begin Script or Files execution"](#) on page 298

To enter commands in Files mode

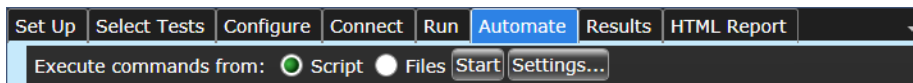
In Files mode, the application will execute the commands it finds in the files shown in the list. The list is constructed and managed via the buttons on the left:



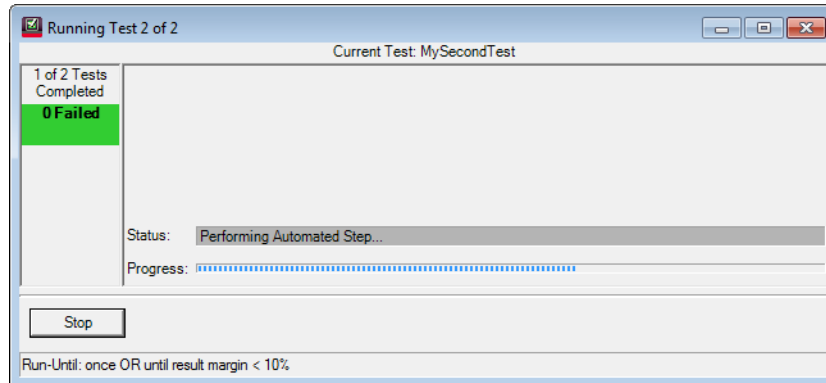
See Also · ["To begin Script or Files execution"](#) on page 298

To begin Script or Files execution

Click **Start** to begin Script or Files execution:

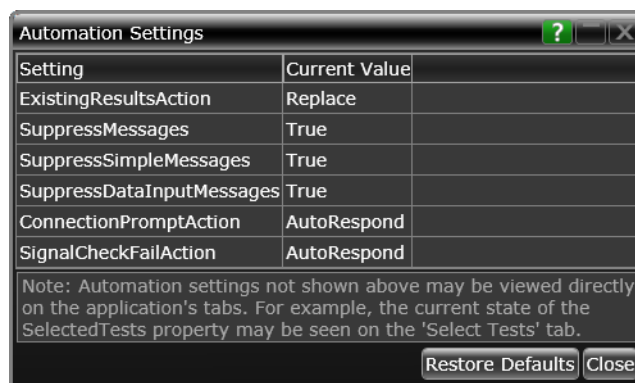
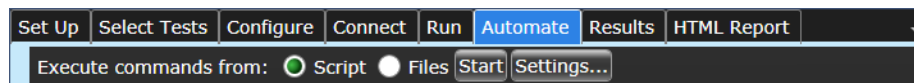


Before the first test executes, you may abort by clicking the same button (now labeled **Stop**) again. Once the run begins, the main window goes away so you would use the Running Test dialog box's **Stop** button instead:



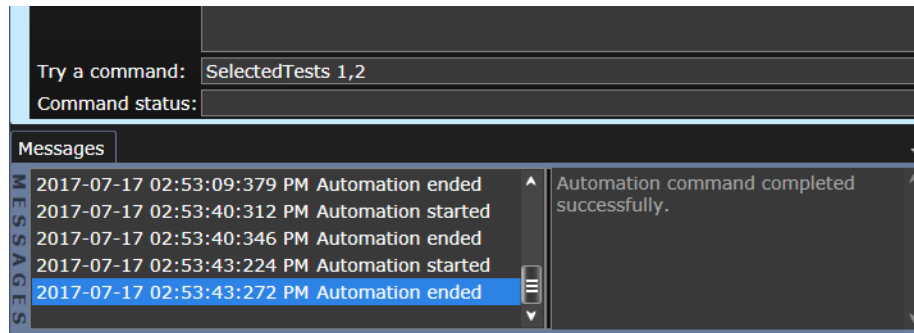
To display automation settings status

Some commands will make visible changes to the application's tabs, such as selecting a test. Other commands only modify invisible settings, such as "SuppressMessages". The Automation Settings status dialog box displays the current value of the invisible properties:



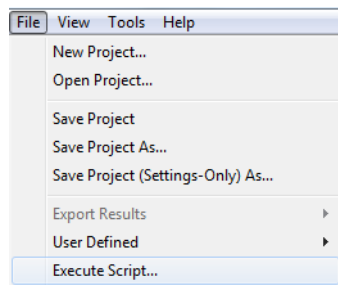
To try a command line

To experiment with a command, type it in the **Try a command** field and press <Enter> to execute it. If it has a visible effect, such as selecting a test, you may then go to another tab to see if it behaved as expected.



Executing Miscellaneous Scripts

If the task you want to perform cannot be done using the Automation tab / ARSL language alone, you can instead execute arbitrary scripts from the **File > Execute** menu:



This menu item appears whenever files exist in this application-specific location:

- Windows XP – C:\Documents and Settings\All Users\Application Data\Keysight\Infiniium\Apps\\app\scripts\
- Windows 7 – C:\ProgramData\Keysight\Infiniium\Apps\\app\scripts\

The script will automatically execute when you select it using this menu action. In order for this feature to work, the script you place in this directory must have a file extension that is registered with Windows so the operating system will know which installed program to use to execute it. This is the same mechanism that enables you, for example, to double-click a .txt file to open it using Notepad.

You may also perform this menu action from a remote client using the ExecuteScript command (see the Keysight N5452A Remote Toolkit for details).

As an example, consider this scenario: You can write a Python script that saves completed tests results in a custom XML format you require for offline data processing.

The prerequisites for this scenario are:

- 1 Python 2.7.x installed on the oscilloscope.

To see if it is already present, check for this file: *C:\python27\python.exe*.

If not, you may get it from here: <http://www.python.org/download>

- 2 Python for .NET installed on the oscilloscope.

To see if it is already present, check for these files:

```
C:\python27\DLLs\clr.pyd
C:\python27\DLLs\Python.Runtime.dll
```

If not, you may get them from here: <http://pythonnet.sourceforge.net> (Choose CLR 2.0 package for Python 2.7)

3 Python enabled for compliance application remote control.

To see if it is already enabled, check for this file: *C:\python27\DLLs\Keysight.DigitalTestApps.Framework.Remote.dll*

If not, you may get it from here: www.keysight.com/find/scope-apps Search page for N5452A and install the toolkit.

The file will be located in the toolkit installation location, "Tools" subdirectory.

Here is an example Python script that reads results of test ID 123 and saves its pass/fail status to a text file:

```

"""Import the compiled Python for .Net module"""
import clr

"""Import the Keysight automated test app remote library DLL"""
clr.AddReference("Keysight.Infiniium.AppFW.Remote")
from Keysight.Infiniium.AppFW.Remote import *

"""Connect to the automated test application"""
scopeIpAddress = "localhost"
remoteObj = RemoteAteUtilities.GetRemoteAte(scopeIpAddress)
remoteApp = IRemoteAte(remoteObj)

"""Get results for test ID 123"""
resultOptions = ResultOptions()
resultOptions.TestIds = [123]
resultContainer = remoteApp.GetResultsCustom(resultOptions)
worstTrial = resultContainer.ExtremeResults[0]
passed = worstTrial.Passed

"""Write pass/fail result to a text file"""
f = open('c:\\temp\\Example.txt', 'w');
f.write(str(passed))
f.close();

```

See the Keysight N5452A Remote Toolkit for more information on using Python to interact with an automated test application.

Viewing Results

- 1 Click the **Results** tab.

Test Name	Actual Value	Margin %	Pass Limits	# Trials
MyFirstTest	757 Hz	-315E+01	31.500000 MHz <= VALUE <= 32.500000 MHz	1
MySecondTest	0 Hz	-750E+01	750 Hz <= VALUE <= 760 Hz	1
Test3	757 Hz	30.0	750 Hz <= VALUE <= 760 Hz	1
Test4	757 Hz	30.0	750 Hz <= VALUE <= 760 Hz	1
Test5	757 Hz	-315E+01	31.500000 MHz <= VALUE <= 32.500000 MHz	1

Parameter	Value
Frequency	757 Hz

Messages

- 2017-07-17 03:57:26:160 PM Problem With External Instruments
- 2017-07-17 03:57:49:716 PM Info
- 2017-07-17 04:00:05:908 PM Project Saving
- 2017-07-17 04:00:06:318 PM Project Saved
- 2017-07-17 04:00:06:337 PM Run ended

(BETA VERSION) Unsaved Changes 5 Tests Connection: Connect External Instrument Trig Out to

The Results tab contains three resizable panes for test results information. If you select one of the tests in the top pane, details and reference images (if any) are shown in the lower panes.

To delete results, right-click and choose to delete the results for the selected test or all tests.

Test Name	Actual Value	Margin %	Pass Limits	# Trials
MyFirstTest	757 Hz	-315E+01	31.500000 MHz <= VALUE <= 32.500000 MHz	1
MySecondTest	Run checked tests		VALUE <= 760 Hz	1
Test3	Delete all results for selected test...		VALUE <= 760 Hz	1
Test4	Delete all results for ALL tests...		VALUE <= 760 Hz	1
Test5	757 Hz	-315E+01	31.500000 MHz <= VALUE <= 32.500000 MHz	1

(BETA VERSION) Unsaved Changes 5 Tests Connection: Connect External Instrument Trig Out to

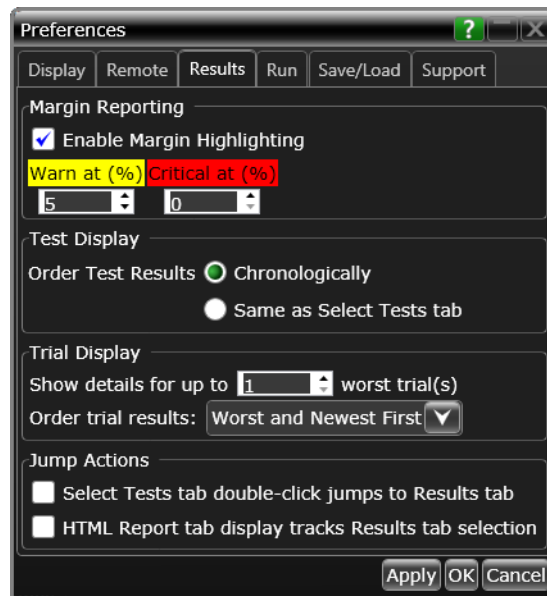
TIP

A quick way to reset all configuration options and delete all test results is to create a new project (see [page 252](#)). The new project will have default configuration options.

- See Also
- ["To change margin thresholds"](#) on page 304
 - ["To set test display preferences"](#) on page 305
 - ["To set trial display preferences"](#) on page 305
 - ["To select jump actions"](#) on page 306
- Next
- ["Viewing/Exporting/Printing the Report"](#) on page 308

To change margin thresholds

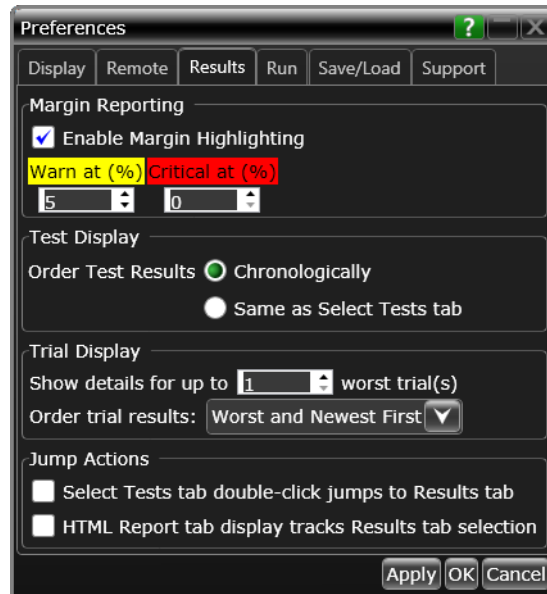
- 1 From the generated application's main menu, choose **View > Preferences...**
- 2 In the Preferences dialog box, select the **Report** tab.



- 3 In the **Margin Reporting** area, you can:
 - Enable or disable margin highlighting.
 - You can change the percent of margin at which to give warnings or critical failures.
- 4 Click **OK** to close the Preferences dialog box.

To set test display preferences

- 1 From the generated application's menu, choose **View > Preferences...**
- 2 In the Preferences dialog box, select the **Results** tab.



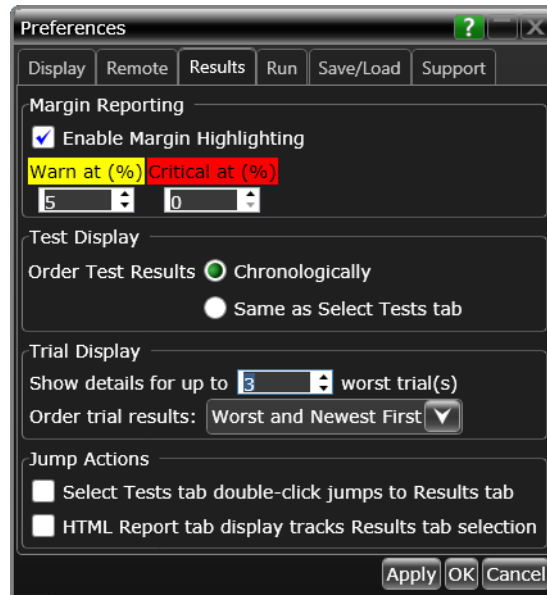
- 3 In the **Test Display** area, you can choose:
 - **Chronologically** – the test results table is arranged in the order of which test was run first.
 - **Same as Select Tests tab** – the test results table is arranged in the order seen in the test selection list.
- 4 Click **Apply** to save the changes and click **OK** to close the Preferences dialog box.

NOTE

These settings only affect the viewing of results and not their capture. Therefore, a change can be made to either before or after running the tests.

To set trial display preferences

- 1 From the generated application's main menu, choose **View > Preferences...**
- 2 In the Preferences dialog box, select the **Report** tab.



3 In the **Trial display** area, you can:

- Select the maximum number of trials, up to 25, whose details are displayed at one time.
- Order trial details chronologically or by "best", "worst", or "last" trial first.

Note that the "worst", "best", or "last" trials depends on the "store mode" setting in the Run tab. See **"To select the "store mode" on page 270.**

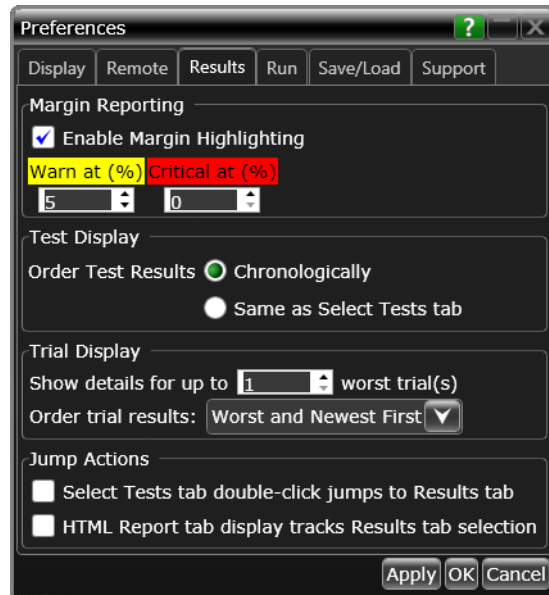
4 Click **Apply** to save the changes and click **OK** to close the Preferences dialog box.

NOTE

These settings only affect the viewing of results and not their capture. Therefore, a change can be made to either before or after running the tests.

To select jump actions

- 1 From the NFC Test Application's menu, choose **View > Preferences....**
- 2 In the Preferences dialog box, select the **Results** tab.



- 3 In the **Jump Actions** area, you can select:
 - **Select Tests tab double-click jumps to the Results tab** – when there are results for a test, double-clicking a test in the Select Tests tab goes to those results.
 - **HTML Report tab display tracks Results tab selection** – after selecting a test in the Results tab, clicking the HTML Report tab will display the report at the point where those test results appear.
- 4 Click **Apply** to save the changes and click **OK** to close the Preferences dialog box.

NOTE

These settings only affect the viewing of results and not their capture. Therefore, a change can be made to either before or after running the tests.

Viewing/Exporting/Printing the Report

- To view the HTML test report, click the **Html Report** tab.

The screenshot displays the Keysight HTML Report interface. The window title is "User Defined MyCompany.MySecondApp Application -- User Defined MyCompany.MySecondApp D". The menu bar includes "File", "View", "Tools", and "Help". The main menu contains "Set Up", "Select Tests", "Configure", "Connect", "Run", "Automate", "Results", and "HTML Report". The Keysight Technologies logo is visible in the top left. The main content area is titled "Test Report" and shows an "Overall Result: FAIL". Below this is a table of "Test Session Details":

Test Configuration Details	
Test Session Details	
Infinium SW Version	6.10.409
Infinium Model Number	DSOX91304A
Infinium Serial Number	MY52260102
Application SW Version	0.01.0029
Debug Mode Used	No
Pass Limits	(user-defined)
Last Test Date	2017-07-17 16:00:01 UTC -06:00

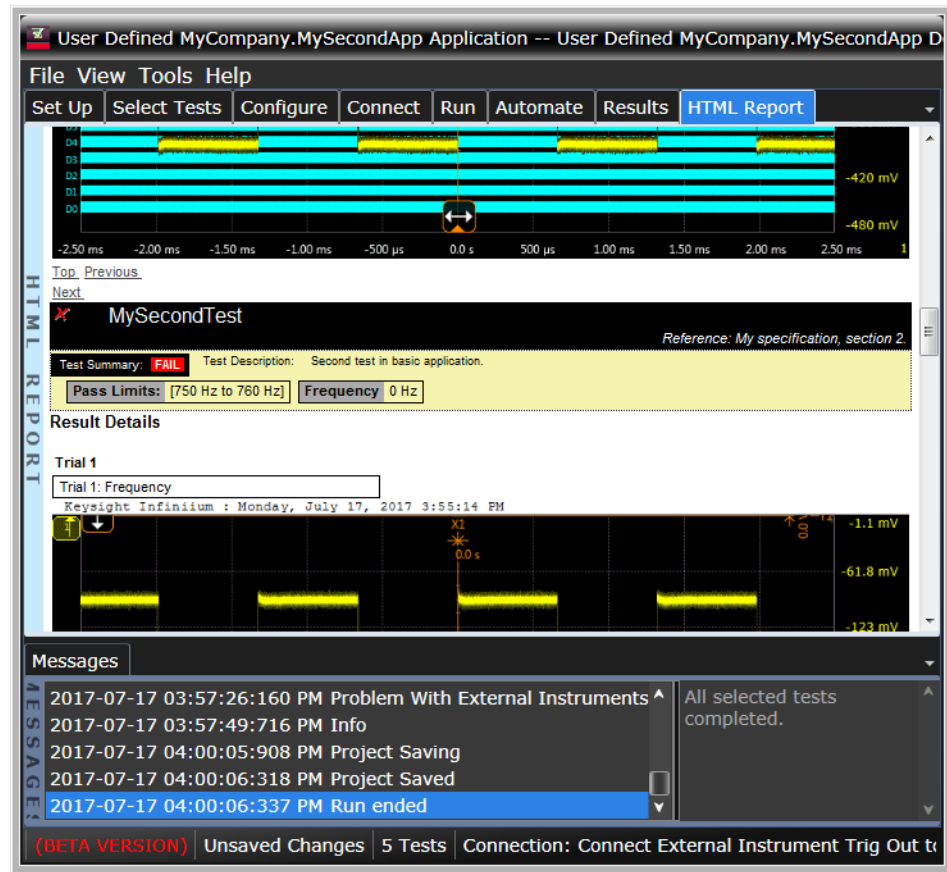
Below the table is a "Summary of Results" section. At the bottom, there is a "Messages" panel with a list of log entries:

- 2017-07-17 03:57:26:160 PM Problem With External Instruments
- 2017-07-17 03:57:49:716 PM Info
- 2017-07-17 04:00:05:908 PM Project Saving
- 2017-07-17 04:00:06:318 PM Project Saved
- 2017-07-17 04:00:06:337 PM Run ended

The status bar at the bottom indicates "(BETA VERSION)", "Unsaved Changes", "5 Tests", and "Connection: Connect External Instrument Trig Out to".

The Report Detail area provides these links:

- **Top:** Jump to test result summary table.
- **Previous:** Jump to start of previous test result.
- **Next:** Jump to start of next test result.



- See Also
- ["To export the report"](#) on page 309
 - ["To print the report"](#) on page 312
- Next
- ["Saving Test Projects"](#) on page 380

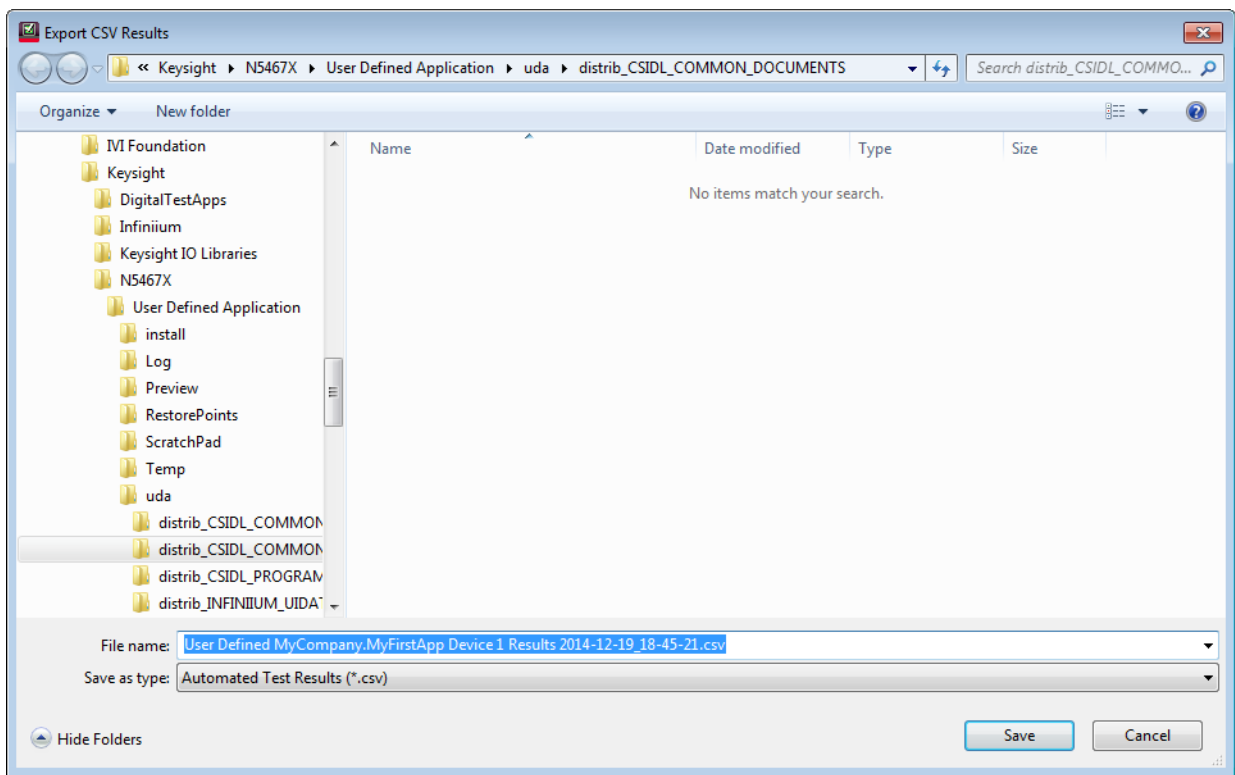
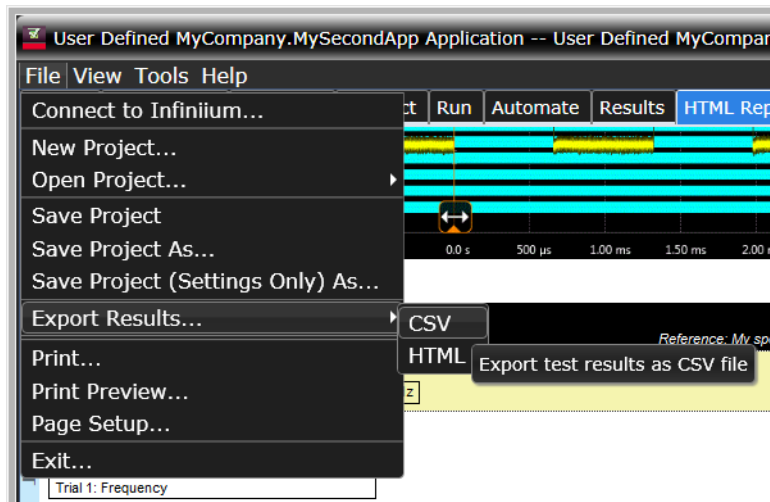
To export the report

- 1 From the generated application's main menu, choose **File > Export Results >** from the menu.

There are two options for exporting the HTML test report: CSV or HTML.

To export results
in CSV
(comma-separated
values) format

Select the CSV option to export the results as a comma-separated list of values.



The data format is shown in the first line of the exported *.csv file.

```

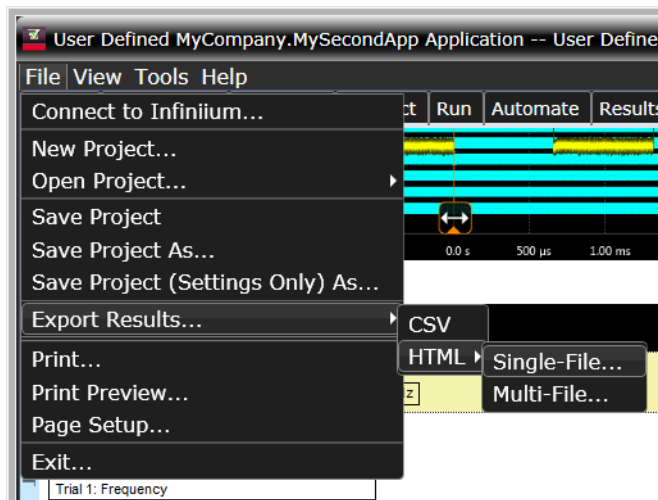
Test ID, Test Name, Measured Item, Trial 1 value
1,"MyFirstTest",Actual Value,"31972448"
1,"MyFirstTest",Margin,"47.2"
2,"MySecondTest",Actual Value,"31611161"
2,"MySecondTest",Margin,"11.1"
5,"Test3",Actual Value,"31880789"
5,"Test3",Margin,"38.1"
3,"Test4",Actual Value,"31908235"
3,"Test4",Margin,"40.8"
4,"Test5",Actual Value,"31631468"
4,"Test5",Margin,"-97.4"

```

TIP

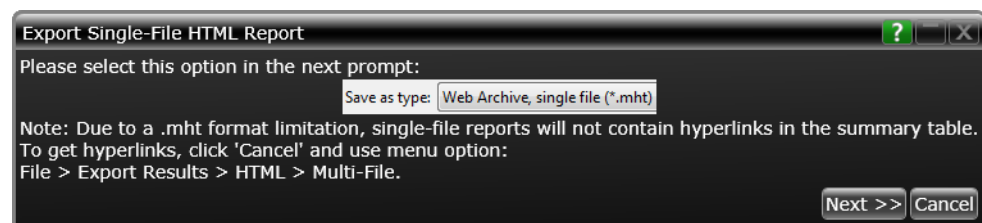
You can add a pass/fail summary row and units by changing the CSV export preference. See ["To set CSV export preferences"](#) on page 312.

To export the report in HTML format



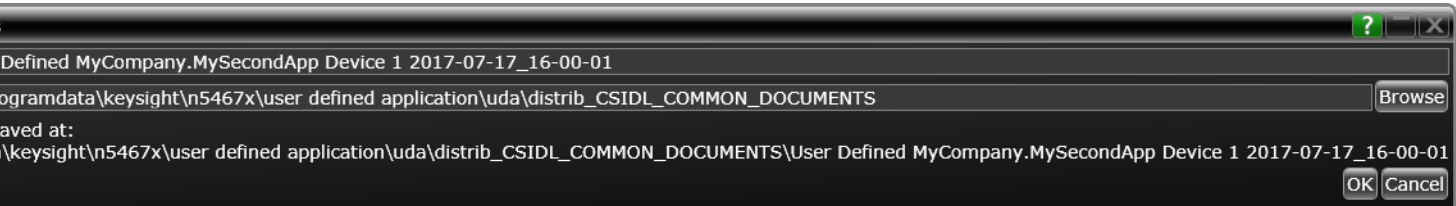
There are two options for exporting HTML format test reports:

- **Single-File** – To save a single-file report, use the "save as" type "Web Archive, single file (.mht)".

**NOTE**

Single-file reports will not contain hyperlinks in the summary table (due to a .mht format limitation). If you want these hyperlinks, use the multi-file format.

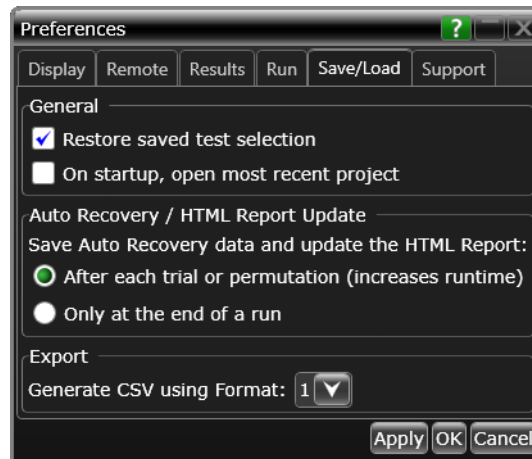
- **Multi-File** – If your report is large and you would like to use links within the report, select the **HTML > Multi-File** option. Selecting the multi-file option exports the results as a set of separate image and HTML files. It creates a folder with the specified name that may be copied to any computer.



To view the exported report, open the HTML file stored in the folder.

To set CSV export preferences

- 1 From the generated application's main menu, choose **View > Preferences....**
- 2 In the Preferences dialog box, select the **Save/Load** tab.



- 3 In the Save/Load tab's **Export** group box, you can select the desired CSV export format:
 - **1** – Selects the legacy CSV format.
 - **2** – Adds a pass/fail summary row and adds units.
- 4 Click **Apply** to save the changes and click **OK** to close the Preferences dialog box.

To print the report

- To preview the HTML test report printout, choose **File > Print Preview...** from the menu.

- To print the HTML test report, choose **File > Print...** from the menu.

Exporting Measurement Results to Web Repository

The **Upload Results To Repository** feature is an add-on to the Keysight Test Application, where it expands the boundaries of storing and analyzing the measurement results to a wider audience, who may be based in multiple sites across various geographical locations. Along with the feature of exporting test results from the Test Application into your local disk in a CSV or HTML file format, you have the option to upload the test results to a Dataset on a Web Repository. Based on your requirements, you may either upload only a single measurement trial or upload huge volumes of measurement results to any Dataset.

Not only can remote users with an active internet connection access these Datasets and the corresponding test results on the Web Repository, but they have the option to add and delete Datasets on the Web Server. In the **Upload Results To Repository** feature, you can even modify the Dataset properties, which are helpful especially when performing a graphical analysis of the uploaded data.

In combination with the Keysight KS6800A Data Analytics software, the **Upload Results To Repository** feature provides a comprehensive solution to export, view and perform analysis of the measurement results, thereby resulting in qualitative data to ensure that the Device Under Test (DUT) is compliant to the industry standards.

To understand the functionality of the **Upload Results To Repository** feature, see:

- ["Launching the Upload Results to Repository window"](#) on page 314
- ["Connecting to Dataset Web Server"](#) on page 316
- ["Accessing Service Account Credentials for Test Application"](#) on page 327
- ["Accessing the Web Repository for Datasets"](#) on page 331
- ["Selecting a Dataset"](#) on page 333
- ["Selecting and Uploading Measurement Results"](#) on page 336
- ["Defining and Modifying Dataset Properties"](#) on page 352

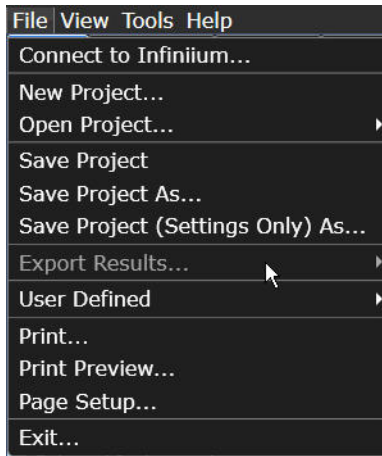
See Also • [Keysight KS6800A Data Analytics Software Online Help](#)

Next • ["Launching the Upload Results to Repository window"](#) on page 314

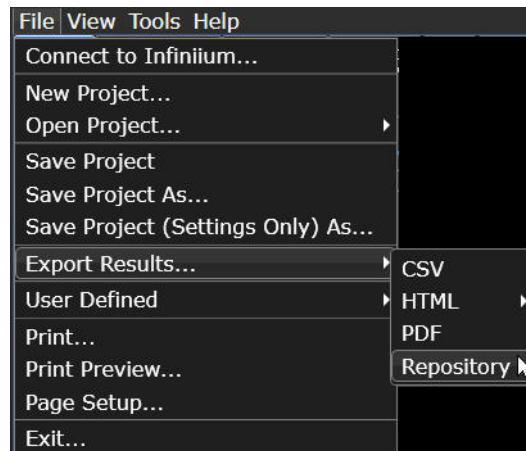
Launching the Upload Results to Repository window

To launch the **Upload Results To Repository** window, you must ensure that some or all tests in the Test Application have been run at least once and that the test results are displayed within the **Results** tab.

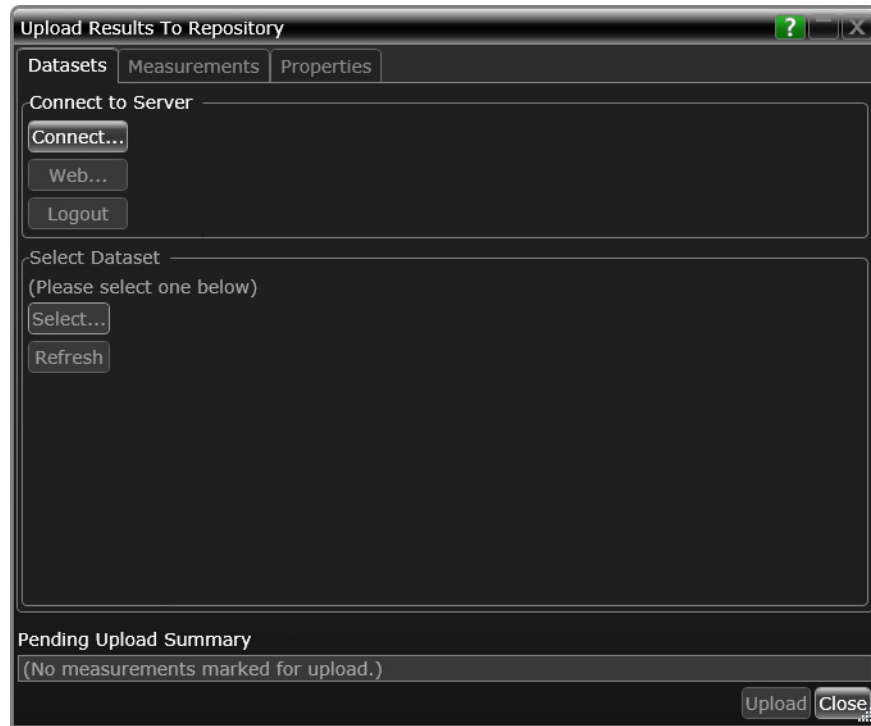
By default, for a new project, the **Export Results...** option is grayed in the **File** menu.



- 1 If you have already run one or more tests, click **File > Export Results... > Repository**.



The **Upload Results to Repository** window appears.



- 2 Click **Close** if you wish to return to the Test Application view.

Next · **"Connecting to Dataset Web Server"** on page 316

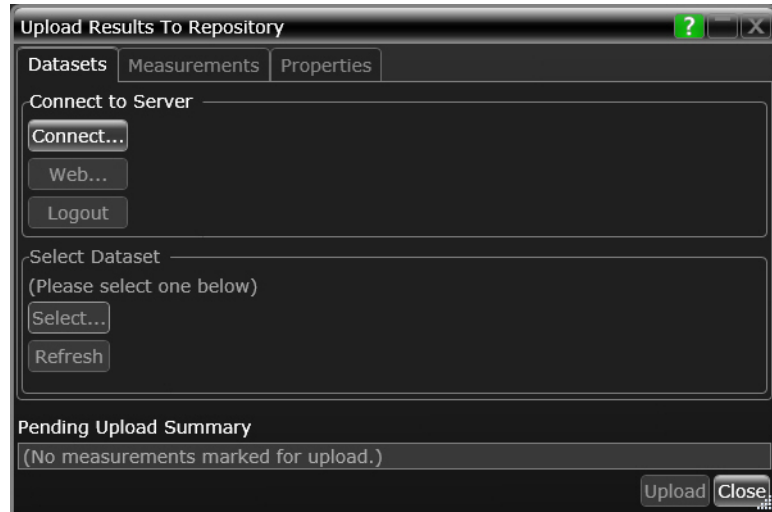
Connecting to Dataset Web Server

To export the measurement results to one or more Datasets from the Keysight Test Application, you must connect to the Web Server, where the Dataset is located. The connection to this repository requires authentication at the application level as well as on the Web Server, which can be accessed through a browser on your PC. Note that the user credentials that are used on the Test Application differ from those required to the login to the Web Server via a browser.

Refer to the *Keysight KS6800A Data Analytics Software Online Help* to know about how to login and view the contents of the Dataset web repository.

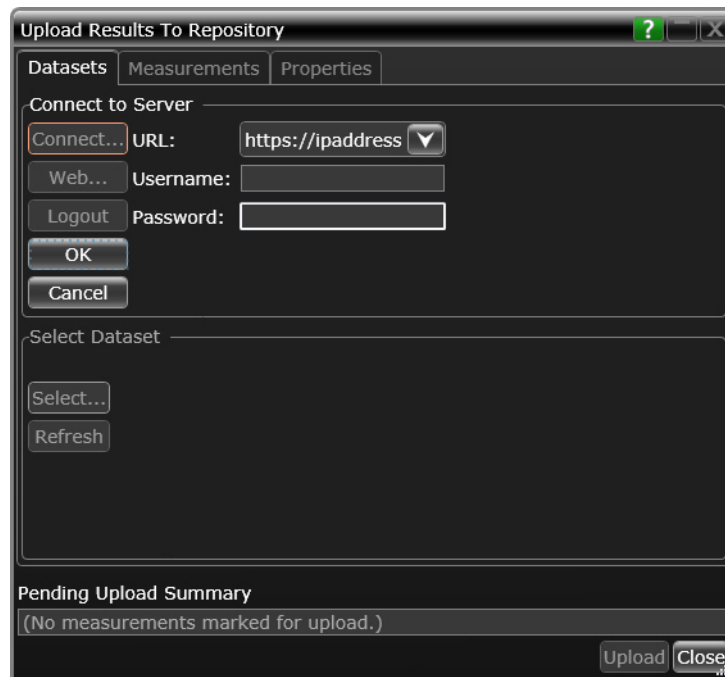
To connect to the Dataset Web server from the **Upload Results to Repository** window:

- 1 Launch the Upload Results to Repository (see [page 314](#)) window. The **Upload Results to Repository** window appears.



- 2 In the **Connect to Server** area, click the **Connect...** button.

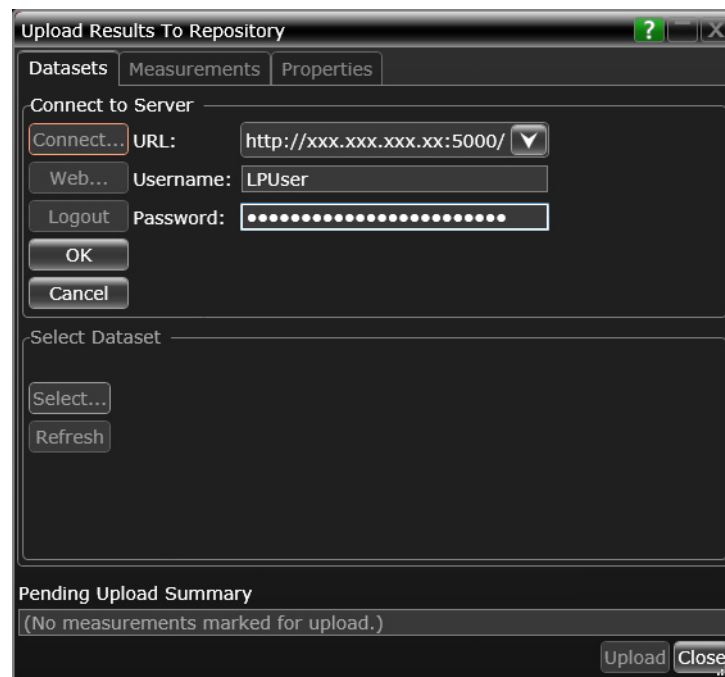
An **URL:** drop-down text field appears with a default entry **https://ipaddress**, along with the **Username:** and **Password:** fields.



- 3 Replace the default text with the actual IP address or the URL (along with the port number, if applicable) of the KS6800A Data Analytics software. Note that the URLs accessed via this window appear as a drop-down list in the URL: field.
- 4 Enter the Service Account username and password in the respective fields, which are required for authentication to access those Datasets that have been created on the web server you are connecting to.

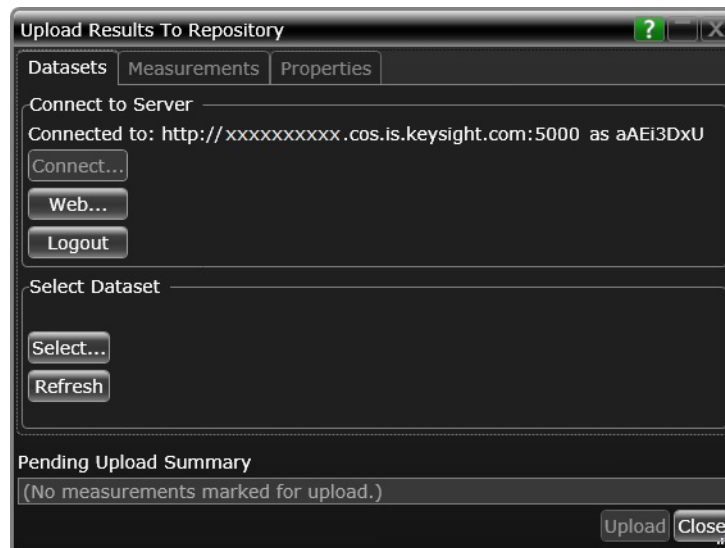
NOTE

If you do not have the Service Account credentials to access the Datasets via the Test Application, contact your local administrator for the KS6800A Data Analytics software to generate the username and password. If you are an administrator or have administrative privileges to generate the Service Account credentials, see "[Generating Service Account Credentials for Test Application](#)" on page 320. You may also refer to the *Keysight KS6800A Data Analytics Software Online Help* to know about how to create Service Account credentials.



- 5 Click **OK** to connect to the entered URL/IP address.

The **Connect to Server** area displays the connection status along with the username.



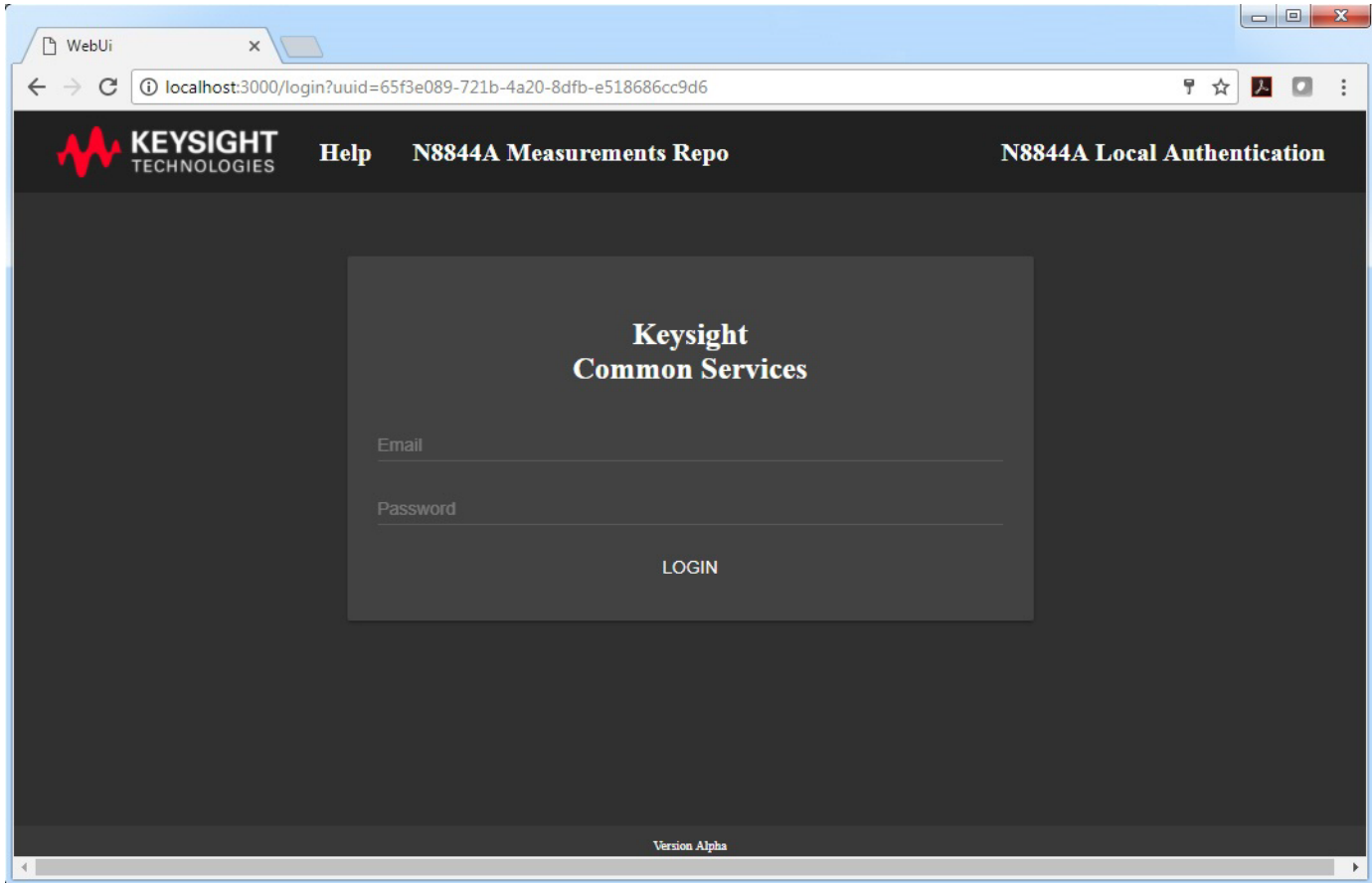
- 6 Click **Logout** to log off the current web repository.
- 7 Repeat steps 1 to 4 if you wish to connect to another web server or if you want to login as another user, that is, using a different username and password.
- 8 Click **Close** if you wish to return to the Test Application view.

Next · ["Accessing Service Account Credentials for Test Application"](#) on page 327

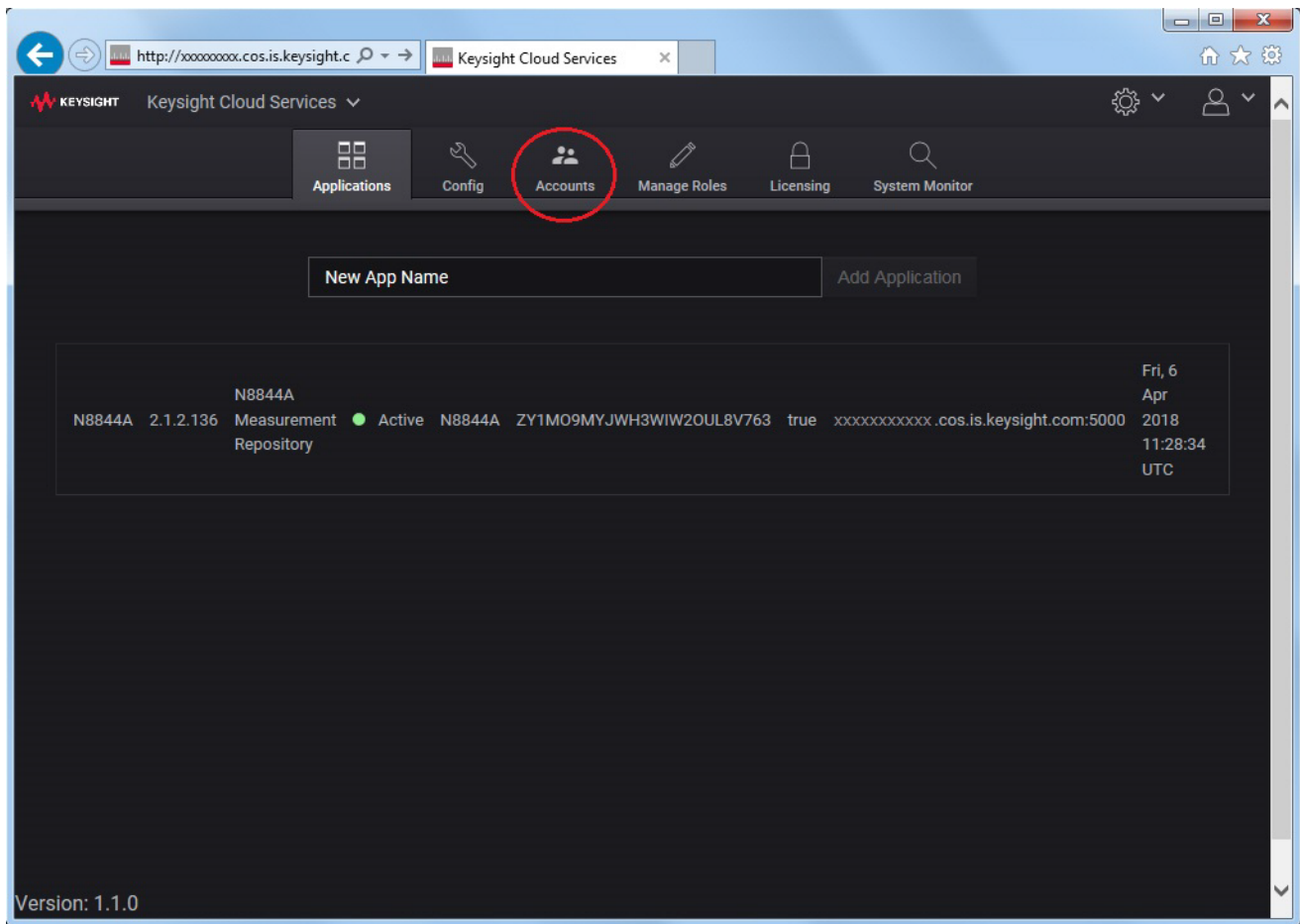
Generating Service Account Credentials for Test Application

To connect to the KS6800A Web Server that contains the Datasets you wish to access via the Test Application, you must have a Service Account username and password. These credentials are different from those required to access the KS6800A Web Server.

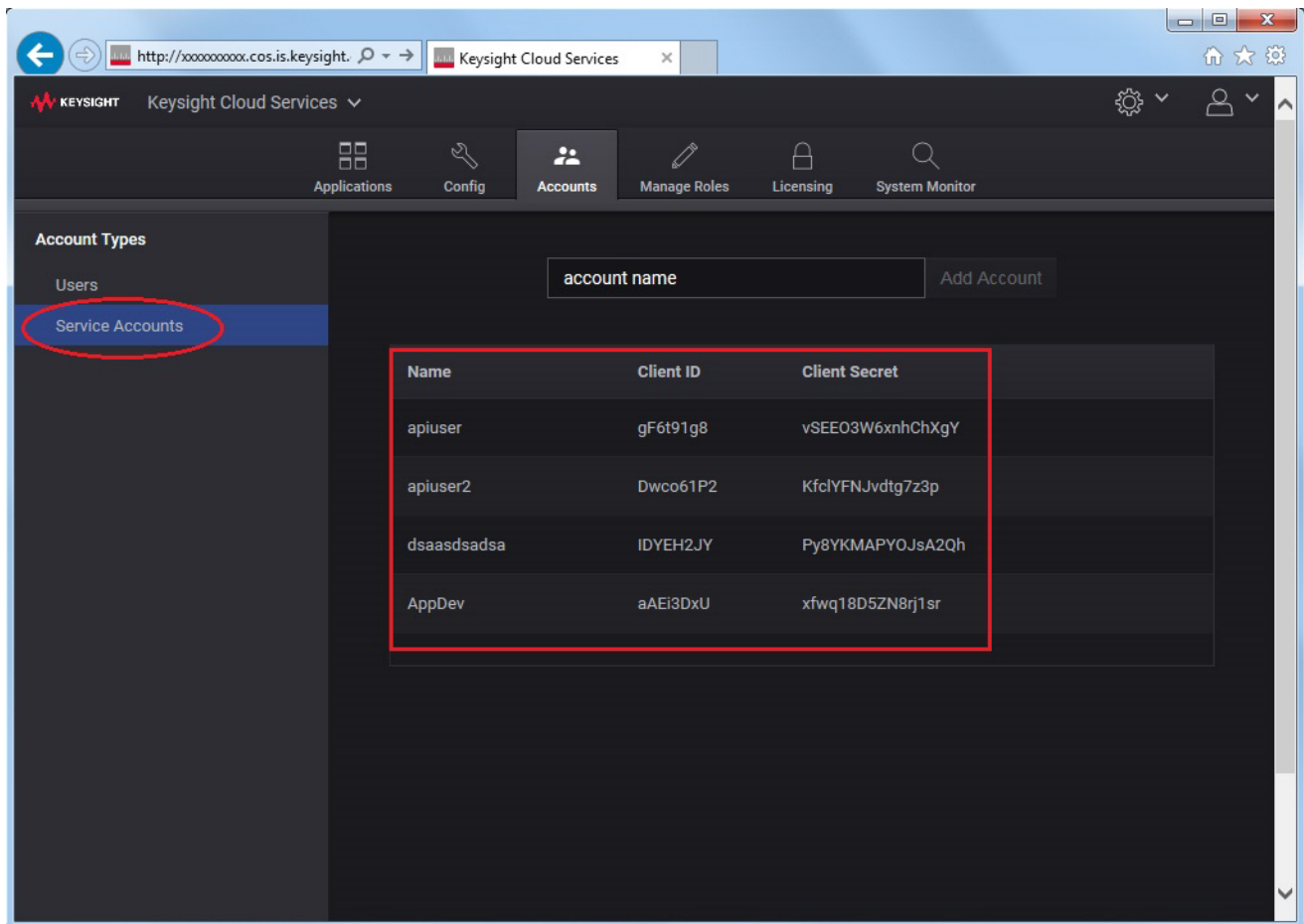
- 1 Launch the KS6800A Web Server on a web browser on your PC.




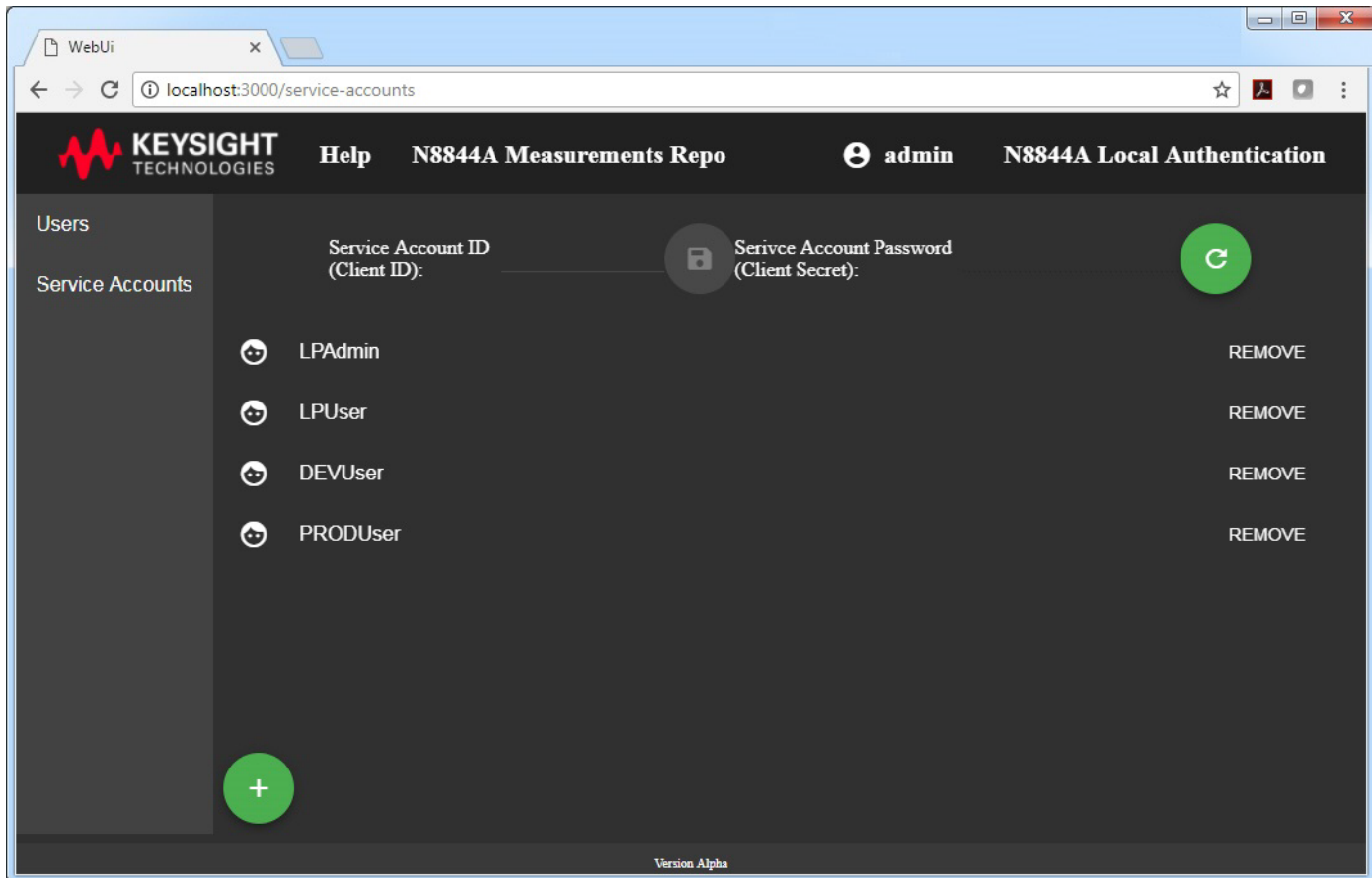
- 2 Login to the KS6800A Web Server with credentials that have administrator privileges.
- 3 By default, the **Users** page is displayed. Click **Service Accounts** on the left pane.




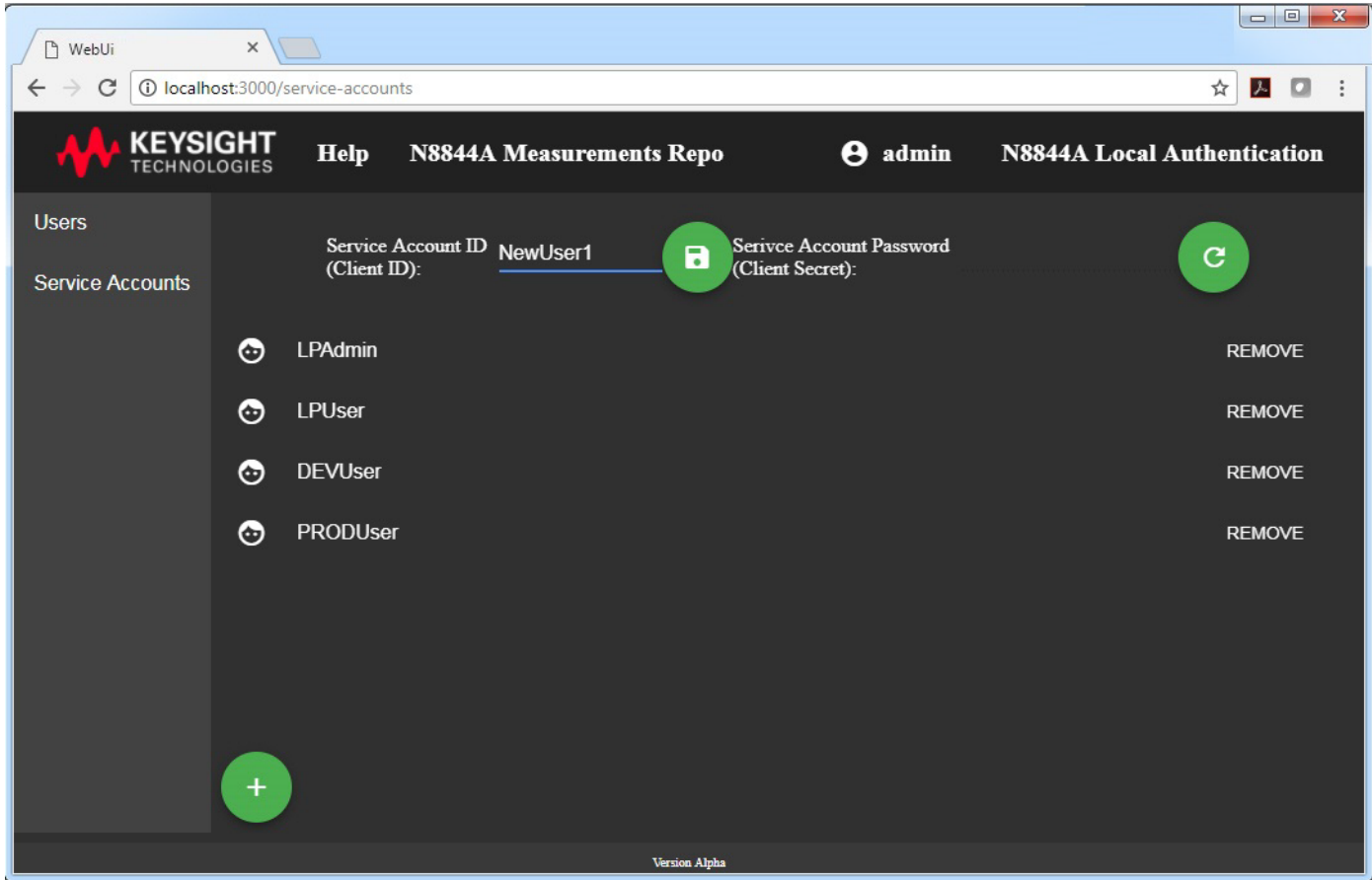
Each username created as a Service Account is listed here.



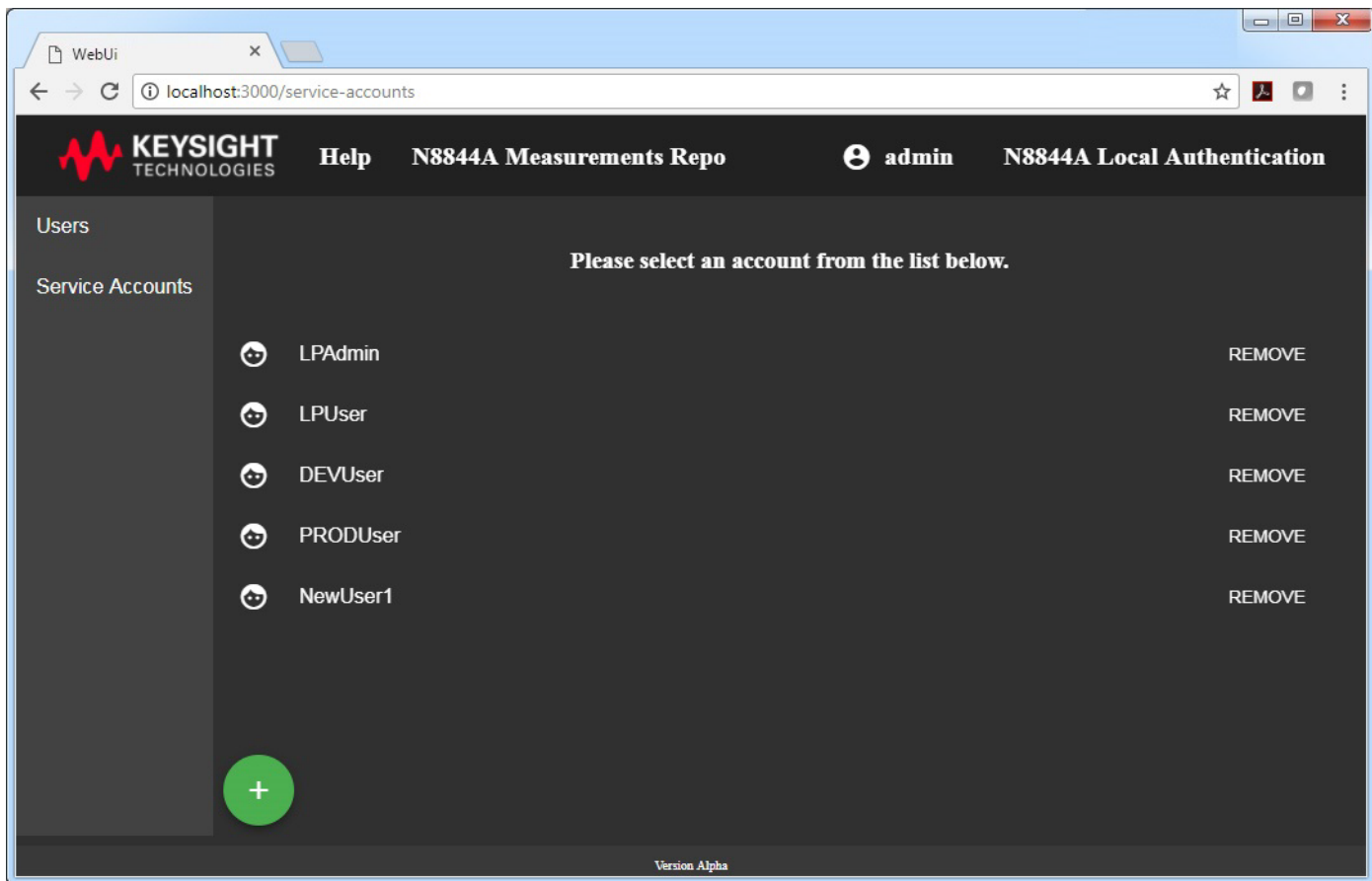
- Creating a new Service Account username
- 1 To create a new username that is not displayed under Service Accounts, click the  button.



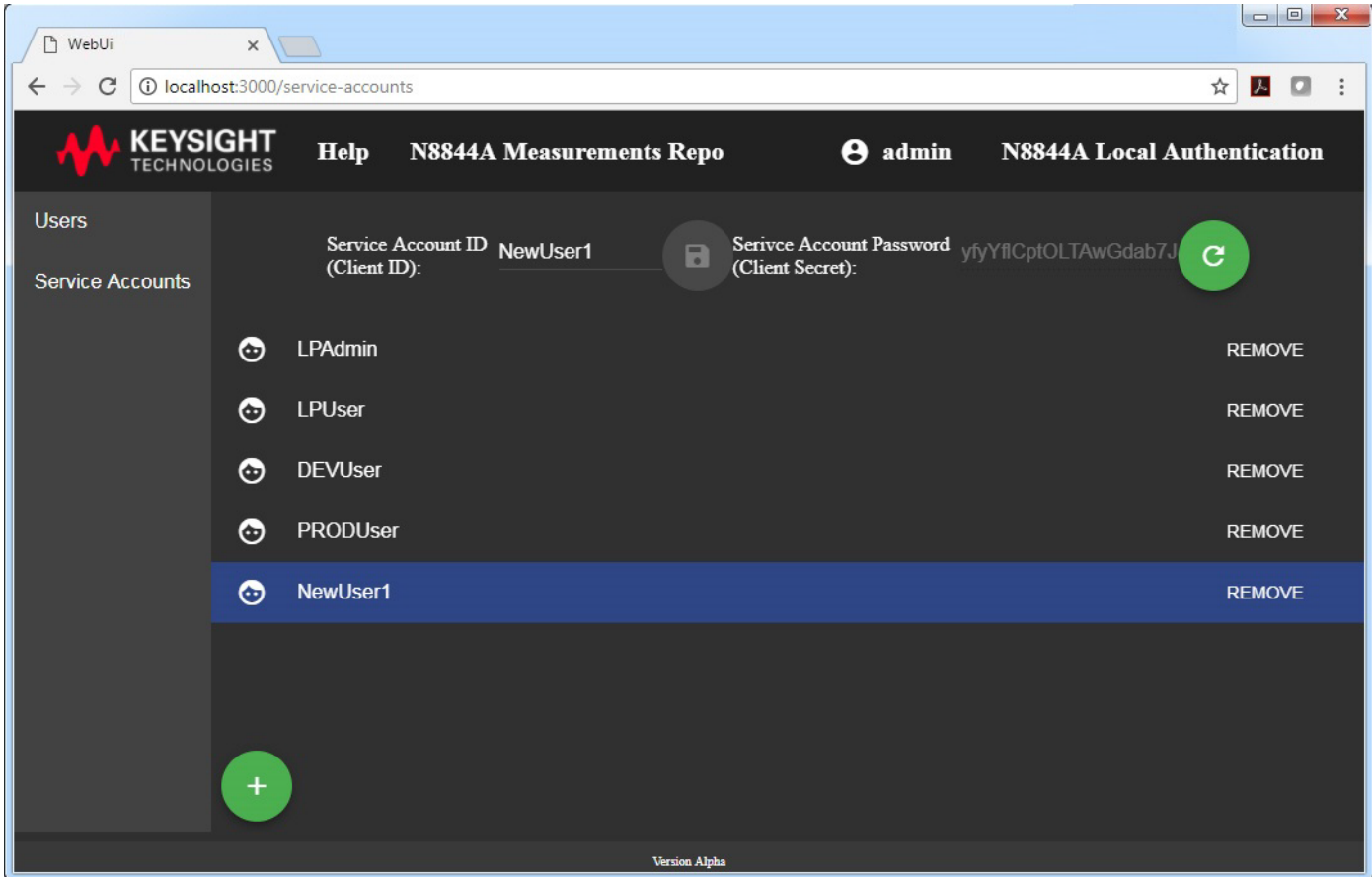
- 2 In the **Service Account ID (Client ID):** field, enter an alphanumeric value.
- 3 Click the  button.




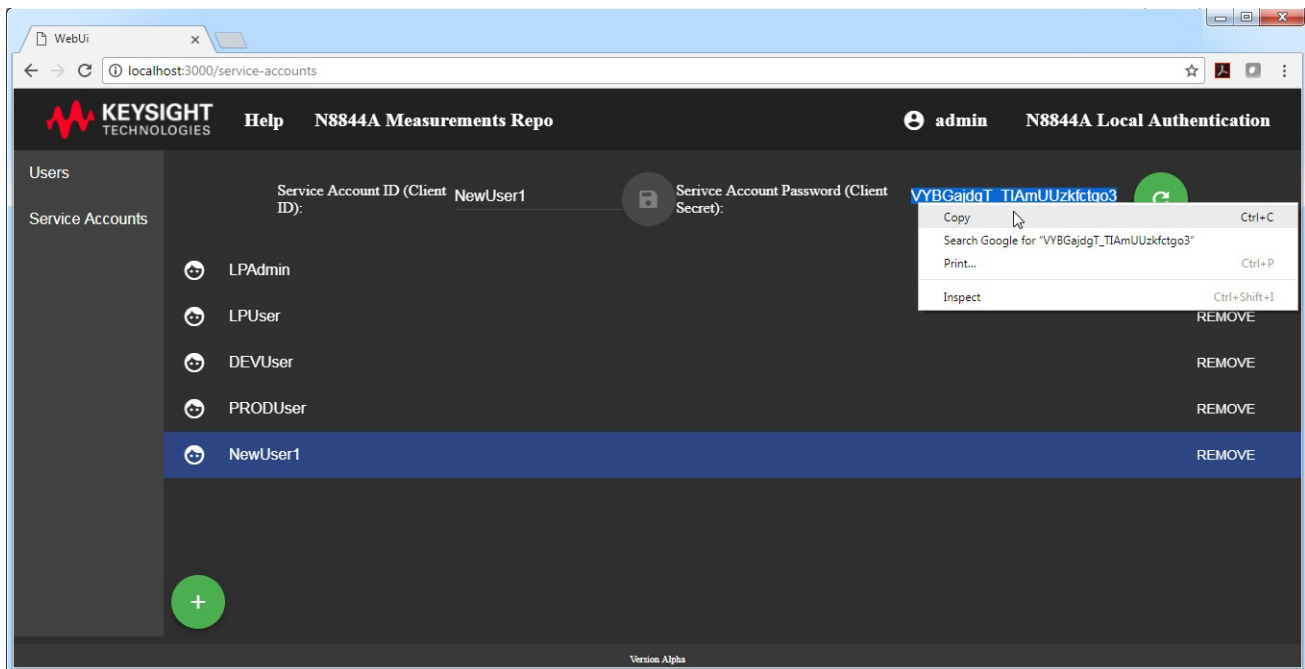
The new username appears in the list of Service Accounts.



- Generating password for a Service Account username
- 1 Select the Service Account username that requires a password.



- 2 By default, an auto-generated password is displayed on the **Service Account Password (Client Secret):** field. You may click the  button to generate a fresh password. Notice that each password that the KS6800A Web Service Software generates is unique and has jumbled alphanumeric values.
- 3 Drag the mouse cursor to highlight the password. Either press **Ctrl + C** keys or right-click and select **Copy** from the context-sensitive menu.



For ease-of-access, e-mail the Service Account username and password to the user, who requires to login to the Web Server via the Test Application. The KS6800A Web Service Software generates a password that can be used in only one instance of the Test Application along with the respective username.

Accessing Service Account Credentials for Test Application

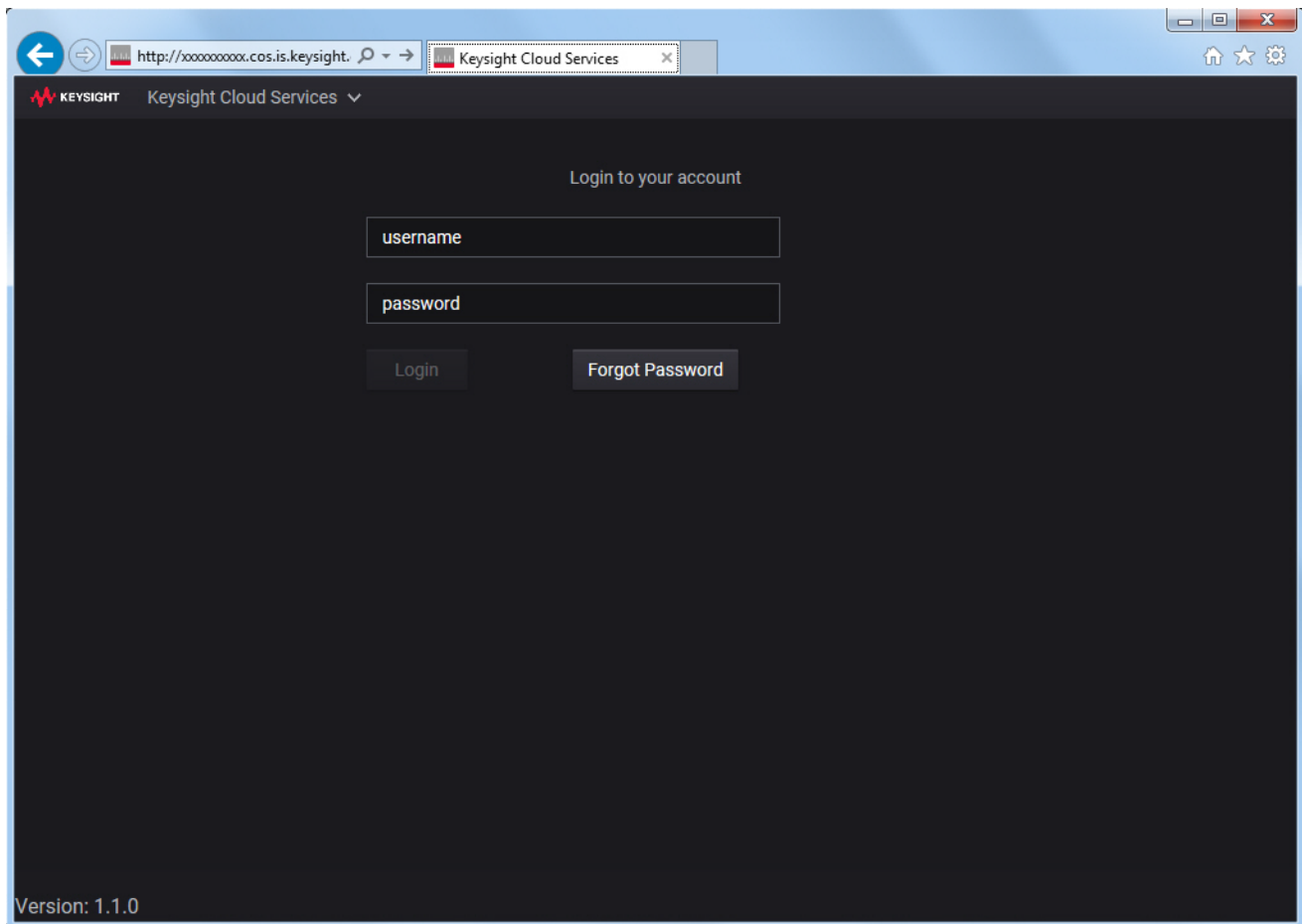
To connect to the KS6800A Web Server that contains the Datasets you wish to access via the Test Application, you must have a Service Account username and password. These credentials are different from those required to access the KS6800A Data Analytics software. The authentication for the KS6800A Data Analytics software is controlled by Keysight Cloud Services.

If you are administrator or have administrative privileges to manage user accounts via the Keysight Cloud Services, this section describes how to access the Service Account credentials only, which you may provide to individual users/user-groups, who must upload test results to the Web Repository, using the Test Application. Refer to the *Keysight KS6800A Data Analytics Software Online Help* to know about how to manage Service Account credentials using Keysight Cloud Services.

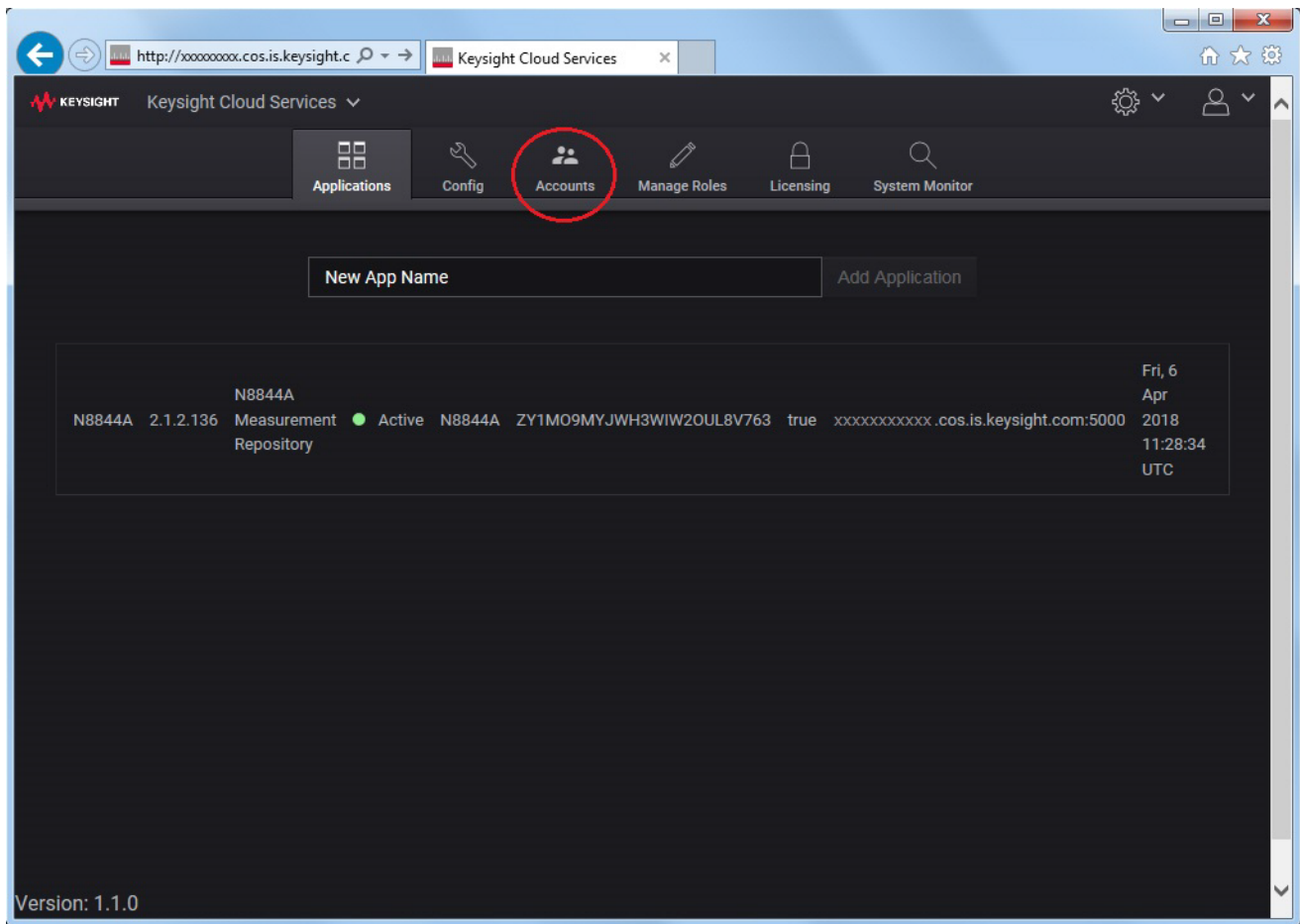
- 1 Launch the Keysight Cloud Services on a web browser on your machine.

NOTE

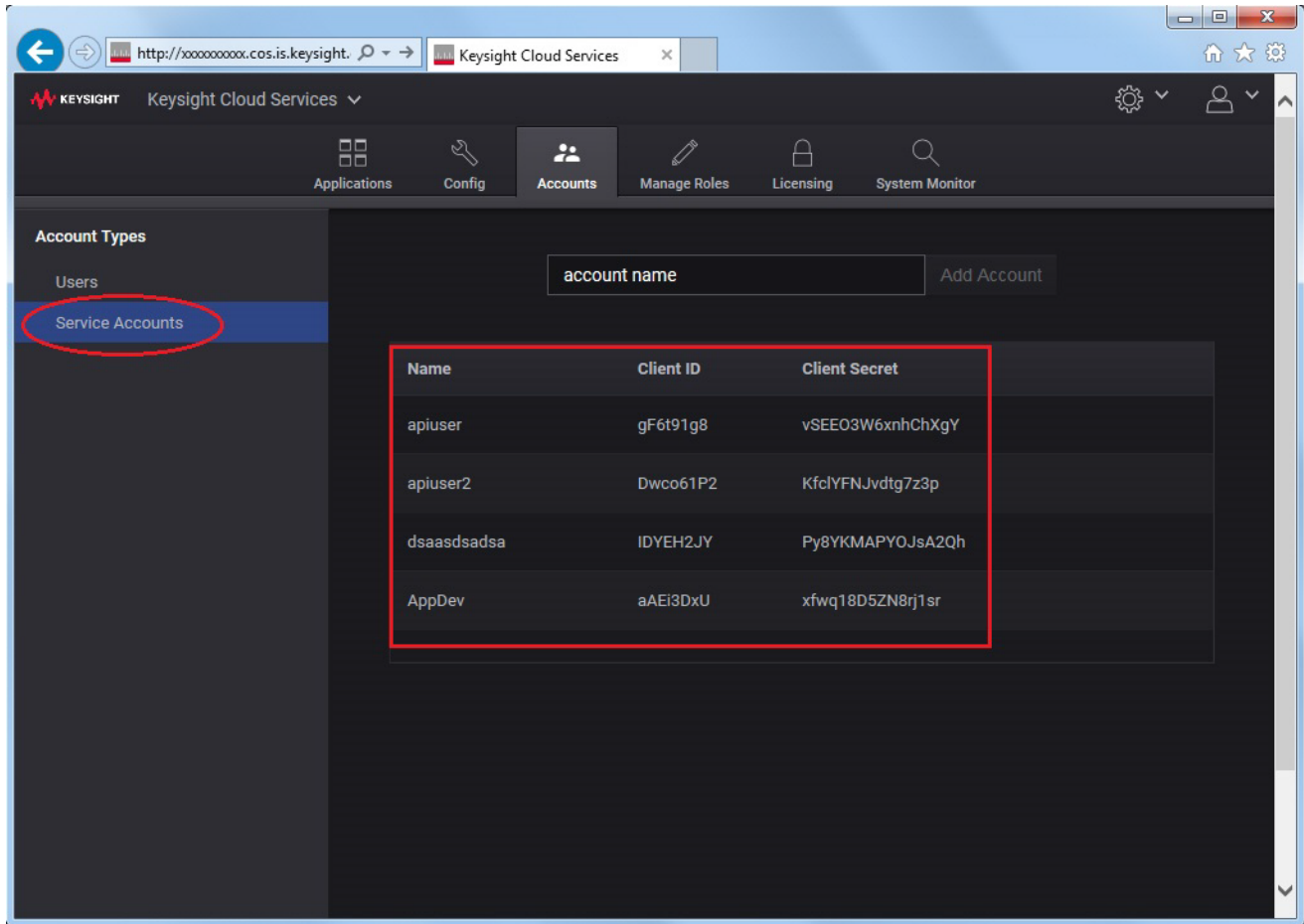
The authentication for the KS6800A Data Analytics software is controlled by Keysight Cloud Services. Therefore, the login screen is common for both Web Services. However, ensure that the port number corresponding to the Keysight Cloud Services is specified in the URL.



- 2 Login to the Keysight Cloud Services with credentials that have administrator privileges.
- 3 By default, the **Applications** tab is displayed. Click the **Accounts** tab.



- By default, the entries created under **Users** are displayed. Click **Service Accounts** on the left pane.



- 5 Under Service Accounts, three columns appear that contain the user credentials corresponding to a user/user-group, who are required to access the KS6800A Data Analytics software via the Test Application.
- **Name** – Actual name of a user or a user-group for whom the service account was created.
 - **Client ID** – the value that corresponds to the Username in the **Upload Results to Repository** window.
 - **Client Secret** – the value that corresponds to the Password in the **Upload Results to Repository** window.

NOTE

Despite their jumbled appearance, the Client ID and Client Secret are generated once and do not change each time a user wishes to access the KS6800A Data Analytics software via the Test Application.

For ease-of-access, a good practise is to e-mail the Service Account username and password to the user, who requires to login to the KS6800A Data Analytics Web Server via the Test Application.

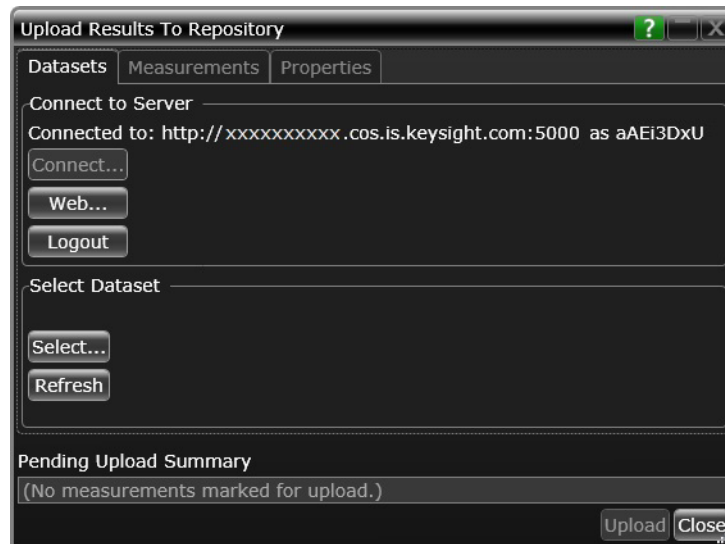
Next · ["Accessing the Web Repository for Datasets"](#) on page 331

Accessing the Web Repository for Datasets

To access the Web Repository for Datasets, you may enter the URL / IP address in the web browser on your PC. You can access the KS6800A Data Analytics software either locally or remotely. When you install the KS6800A Data Analytics software on your PC, it works as a self-hosted server and you can access the Database locally on your web browser using localhost in the URL. When you install the KS6800A Data Analytics software on a remote PC, you can access the Database remotely on your web browser using the IP address of the remote PC in the URL. For more information on installing the KS6800A Data Analytics software, refer to the *Keysight KS6800A Data Analytics Software Online Help*.

To access the Web Repository from the Test Application:

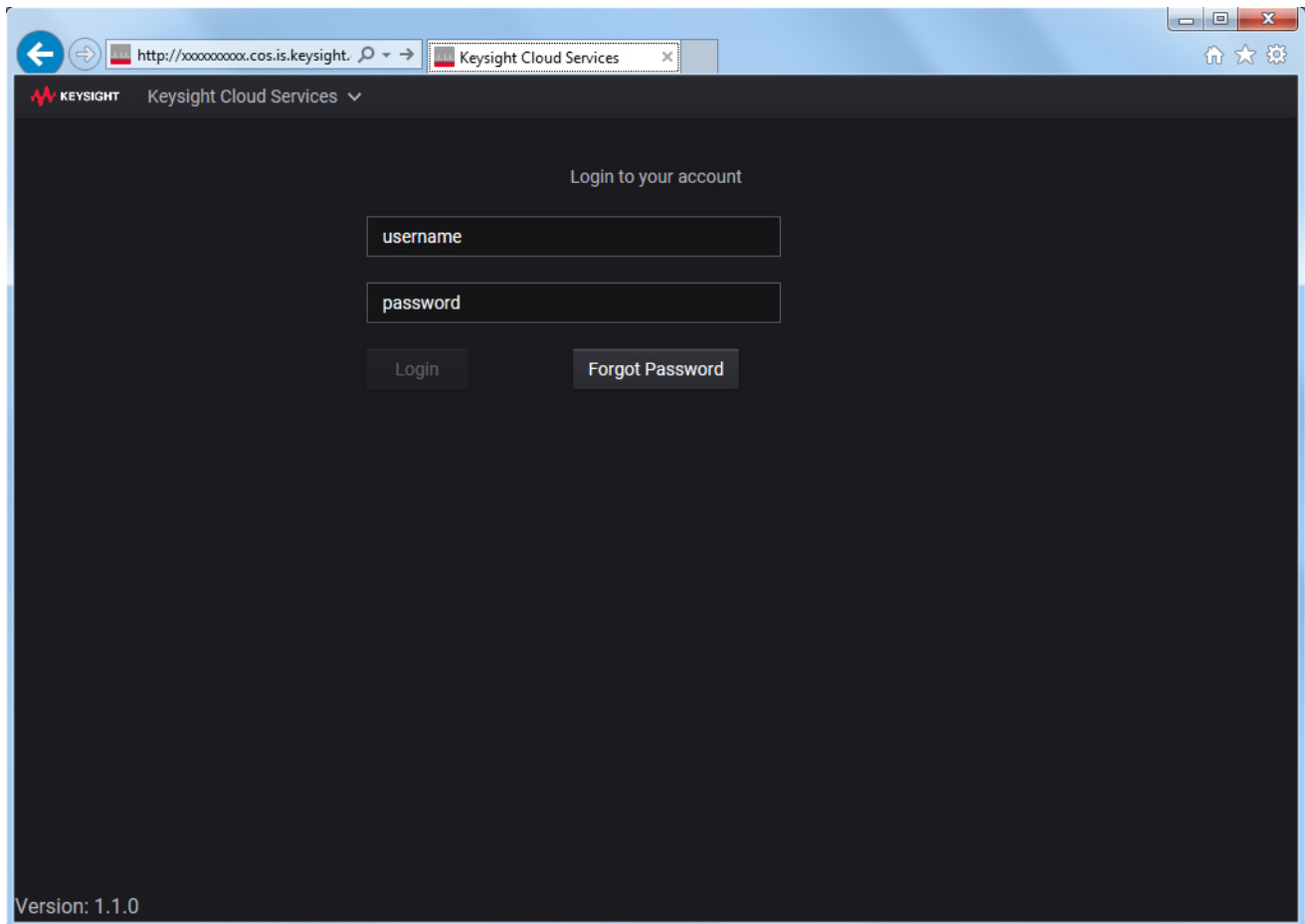
- 1 Ensure that the Test Application is connected to the web server. As show in the image below, the status in the **Connect to Server** area must display **Connected to:** <URL / IP address> **as** <username>. See Connecting to Dataset Web Server (see [page 316](#)) if the Compliance Test Application is not connected to the Web Repository for Datasets.



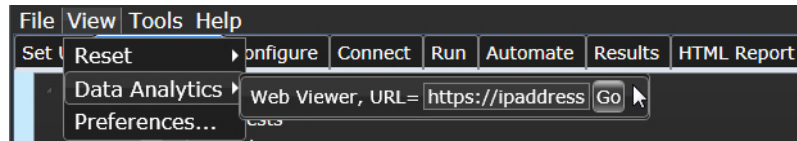
- 2 Click the **Web...** button. If you had already logged in and closed the previous instance of the KS6800A Data Analytics software without properly logging out, the default home page for the Web Repository appears on your default web browser.

NOTE

The authentication for the KS6800A Data Analytics software is controlled by Keysight Cloud Services. If you are a first time user or if you had logged off from the previous session, clicking the **Web...** button launches the login screen, which pertains to Keysight Cloud Services. Refer to the *Keysight KS6800A Data Analytics Software Online Help* to know about how to manage credentials using Keysight Cloud Services to login to the KS6800A Data Analytics software.



- 3 An alternate way to connect to the desired web server directly is using the **View** menu of the Test Application:
 - a Click **View > Data Analytics**.
 - b In the **Web Viewer, URL =** drop-down text field, replace the default text with the actual IP address or the URL of the web server where the Dataset Repository is located.



- c Click **Go**. The default browser on your PC launches the Web Repository for Datasets.

Next · **"Selecting a Dataset"** on page 333

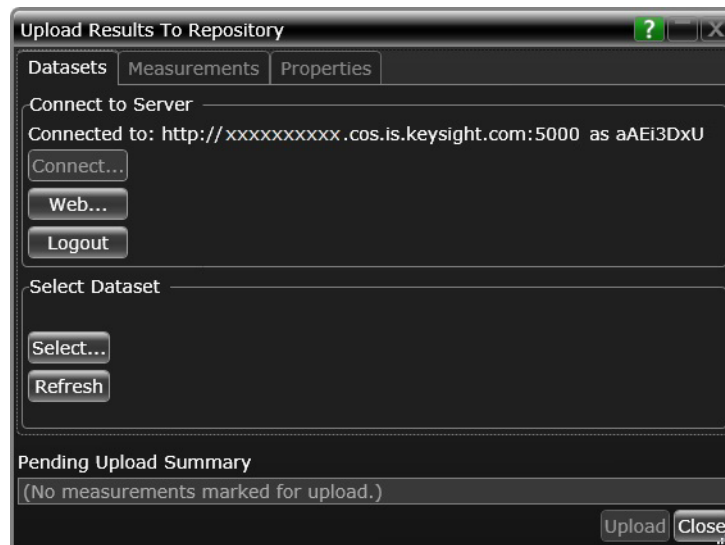
Selecting a Dataset

A Dataset is a user-defined folder on the Web Repository that holds measurement results from the Test Application.

Each Dataset can be assigned a unique name and certain user-defined properties for easy identification on the Web Repository and the Test Application. Refer to the *Keysight KS6800A Data Analytics Software Online Help* to know about how to add and delete a Dataset on the web server. The Dataset names from the connected web server appear within the **Upload Results to Repository** window such that you can select a specific Dataset name to upload measurement results.

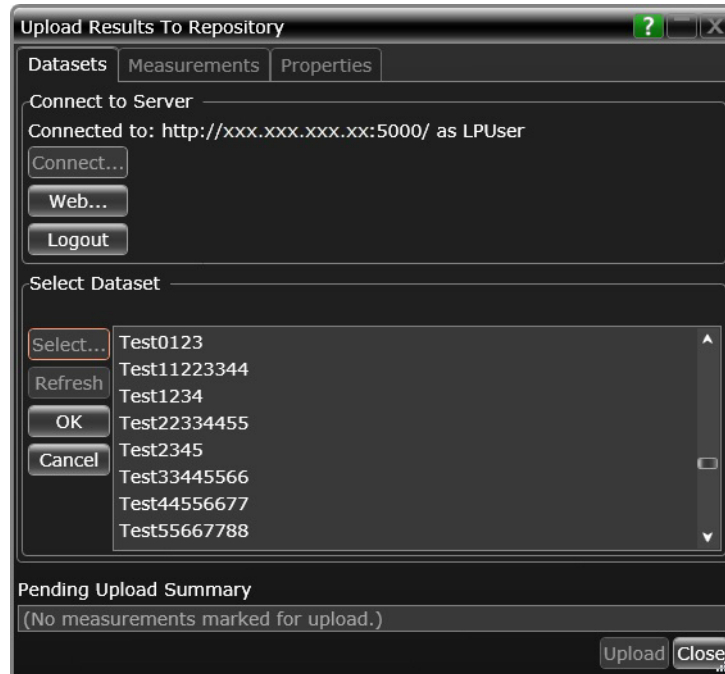
To select a Dataset, perform the following steps:

- 1 Ensure that the **Upload Results to Repository** window indicates that the Test Application is connected to a specific web repository. See Connecting to Dataset Web Server (see [page 316](#)) to know how to establish a connection with a specific URL.

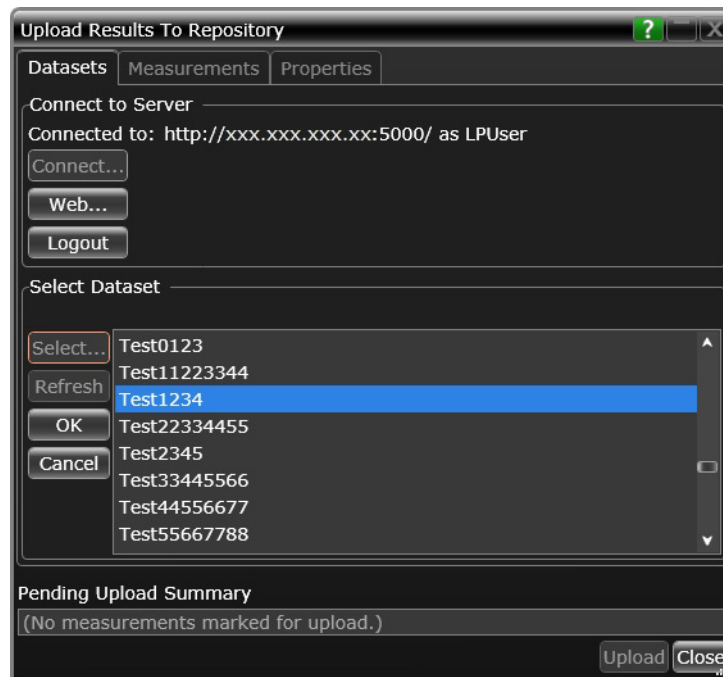


- 2 In the **Select Dataset** area, click **Select...** to view a list of Datasets that have been added in the connected Web Server. If the Dataset that you wish to select does

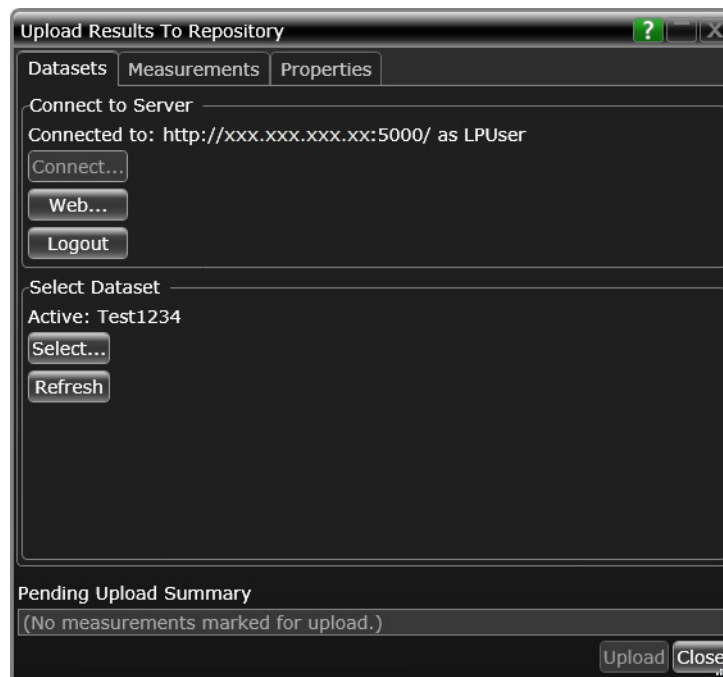
not appear in the list, click the **Cancel** button to return to the default view of the **Select Dataset** area and click the **Refresh** button. Repeat step 2 to check if the required Dataset appears in the list or not. If the Dataset still does not appear, check the Web Repository to verify whether or not the required Dataset has been added to the server. Refer to *Keysight KS6800A Data Analytics Software Online Help* if you are required to add a new Dataset.



- 3 If found, click to highlight the specific Dataset name where you wish to upload the measurement results to and click the **OK** button.



The **Select Dataset** area displays the selected Dataset as **Active**. Notice that selecting a Dataset enables the **Measurements** and **Properties** tabs.



- 4 Repeat steps 1 to 3 if you wish to select any other Dataset from the Web Repository that the Test Application is connected to.

Next · ["Selecting and Uploading Measurement Results"](#) on page 336

Selecting and Uploading Measurement Results

Once you select the Dataset (see [page 333](#)) where you wish to upload the measurement results, the **Measurements** tab of the **Upload Results to Repository** window is enabled. The **Measurements** tab displays one or more groups of measurement results and the corresponding attributes for each test result. It gives you a wide range of options to select from and upload, that is, you may select and upload:

- one or more trials of a single measurement result
- one or more trials across various measurement sets / measurement types
- single, several or all measurement sets / measurement types

By default, the **Measurements** tab displays all sets (one or more) of measurement results that you may have run in an active project on the Test Application.

NOTE

In the active project on the Test Application, when you run tests for the first time, the measurement set and its associated trials are displayed under the **Measurements** tab and the **Trial#** column on the right pane displays **1** for each trial run. However, if you run another instance of the tests in the Test Application for the same project, there is a prompt from the Test Application to either **Append** or **Replace existing trials**.

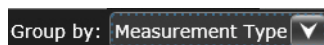
If you select **Append**, the trials that are appended in the **Results** tab of the Test Application are simultaneously appended to **Measurements** tab of the **Upload Results To Repository** window. The new measurement set contains a combination of the next trial of any test that is run again and the first trial of a new test that may have been run.

If you select **Replace existing trials**, the trials that are removed from the **Results** tab of the Test Application are simultaneously removed from the **Measurements** tab of the **Upload Results To Repository** window. The new measurement set contains only the replaced results.

- ["Grouping the Measurement Results"](#) on page 336
- ["Selecting and Uploading Results"](#) on page 344

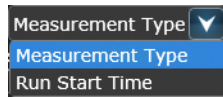
Next · ["Defining and Modifying Dataset Properties"](#) on page 352

Grouping the Measurement Results



Using the **Group by:** feature in the **Measurements** tab, you may view and select the measurement results in two different ways.

To select a grouping method, click the drop-down arrow and select one of the entries.



- **Run Start Time** – Displays the measurement results based on the date and time (in YYYY-MM-DD hh:mm:ss::nnn format, where nnn is milliseconds) when the measurement set had started running on the Test Application. Note that the measurement sets are displayed in the increasing order of time. Each measurement set includes one or more trials of either one or several measurement types.
- **Measurement Type** – Displays the measurement results based on the measurement type name. Note that the measurement type names are listed in alphabetical order. Each measurement type includes the start time and duration of one or more its associated trials that have already been run in an active project of the Test Application.

Even though the manner of selecting and uploading the measurement items is the same for both the groups, each group has its unique way of displaying the measurement results.

Run Start Time

Upload Results To Repository

Datasets Measurements Properties

Group by: Run Start Time

Select Run Start Times* to include:

<input type="checkbox"/>	Run Start Time	Run Duration	Pending	In Repository
<input type="checkbox"/>	2016-11-15 14:40:28::588	00:00:03.0483048		None
<input type="checkbox"/>	2016-11-15 15:05:02::576	00:00:02.7182718		None
<input type="checkbox"/>	2016-11-15 15:06:24::062	00:00:02.0362036		None

(Click on a Run Start Time to see measurements.)

*Each row contains results from a single execution of all checked tests; an N-Times or TestPlan execution produces multiple rows.

Pending Upload Summary
(No measurements marked for upload.)

Upload Close

The left pane of the **Run Start Time** group has the following columns:

- Select check boxes – Select either one, more than one or all measurement sets whose corresponding measurement types you wish to view on the right pane.
- **Run Start Time** – Displays the date and time (in YYYY-MM-DD hh:mm:ss::nnn format, where nnn is milliseconds) when the measurement set had started running on the Test Application. Note that the measurement sets are displayed in the increasing order of time.
- **Run Duration** – Displays the time taken for the tests in each measurement set to finish running.
- **Pending** – Displays either **All** or **Partial** for each row of measurement set:
 - **All** indicates that on the right pane, you have selected all tests and associated trials of the corresponding measurement set.

- **Partial** indicates that on the right pane, you have selected some (and not all) tests or associated trials of the corresponding measurement set.
- **In Repository** – Displays either **NONE**, **PARTIAL**, or **ALL** for each row of measurement set:
 - **NONE** indicates that none of the trials in the measurement set have been uploaded to the web repository.
 - **PARTIAL** indicates that some, but not all of the trials in the measurement set have been uploaded to the web repository.
 - **ALL** indicates that all of the trials in the measurement set have been uploaded to the web repository.

Highlight a specific row, without selecting the check box, if you wish to simply view the measurement types and trials associated with a measurement set. The measurement types on the right pane are grayed out until you select the check-box for one or more measurement sets on the left pane.

Upload Results To Repository

Datasets Measurements Properties

Group by: Run Start Time

Select Run Start Times* to Include:

<input type="checkbox"/>	Run Start Time	Run Duration	Pending	In Repository
<input checked="" type="checkbox"/>	2016-11-15 14:40:28::588	00:00:03.0483048		None
<input type="checkbox"/>	2016-11-15 15:05:02::576	00:00:02.7182718		None
<input type="checkbox"/>	2016-11-15 15:06:24::062	00:00:02.0362036		None

1b. Select Measurements to Include:

<input type="checkbox"/>	Measurement	Limits	Value	Trial#	In Repository
<input type="checkbox"/>	HardcodedMaxLimit	< 10.00	9.00	1	False
<input type="checkbox"/>	HardcodedMaxLimitWithUnits	<= 10.00 ub	10.00ub	1	False
<input type="checkbox"/>	HardcodedMinLimit	> 0.000	900 m	1	False
<input type="checkbox"/>	HardcodedMinLimitWithUnits	>= 0.000 ua	1.000ua	1	False
<input type="checkbox"/>	HardcodedRangeLimit	(200 to 3.000 k)	90	1	False
<input type="checkbox"/>	HardcodedRangeLimitWithUnits	[200 uc to 3.000 kuc]	100uc	1	False
<input type="checkbox"/>	HardcodedSingleValueLimit	= 40.00 k	900	1	False
<input type="checkbox"/>	HardcodedSingleValueLimitWithUnits	= 40.00 kud	1.00 kud	1	False
<input type="checkbox"/>	VariableMaxLimit	< 120.0	11.0	1	False
<input type="checkbox"/>	VariableMaxLimitWithUnits	<= 130.0 ub	12.0ub	1	False
<input type="checkbox"/>	VariableMinLimit	> 90.00	1.10	1	False
<input type="checkbox"/>	VariableMinLimitWithUnits	>= 100.00 ua	1.20ua	1	False
<input type="checkbox"/>	VariableRangeLimit	(110 to 120)	110	1	False
<input type="checkbox"/>	VariableRangeLimitWithUnits	[120 uc to 130 uc]	120uc	1	False
<input type="checkbox"/>	VariableSingleValueLimit	= 100	1.1 k	1	False
<input type="checkbox"/>	VariableSingleValueLimitWithUnits	= 200 ud	1.2 kud	1	False

*Each row contains results from a single execution of all checked tests; an N-Times or TestPlan execution produces multiple rows.

Pending Upload Summary
(No measurements marked for upload.)

Upload Close

The right pane of the **Run Start Time** group has the following columns:

- Select check boxes – Select either one, more than one or all measurement types whose trials you wish to upload.
- **Measurement** – Displays the measurement types. Note that the measurement types are displayed in alphabetical order.

- **Limits** – Displays the specification-based or user-defined pass limits configured in the Test Application before running the test trial. The value displayed here is the same as that displayed in the **Pass Limits** column of the **Results** tab in the Test Application for the respective test.
- **Value** – Displays the value attained by the Test Application after running the test trial. The value displayed here is the same as that displayed in the **Actual Value** column of the **Results** tab in the Test Application for the respective test.
- **Trial#** – Displays the trial number of a measurement type. Each trial of a measurement is included in a separate measurement set.
- **In Repository** – Displays either **TRUE** or **FALSE** for each row of measurement type:
 - **TRUE** indicates that the measurement trial has been uploaded to the web repository.
 - **FALSE** indicates that the measurement trial has not been uploaded to the web repository.

NOTE

As specified in the **Upload Results To Repository** window, each row in the **Run Start Times** group contains results from a single run of all selected tests. If you select **N-Times** in the **Run** tab of the Test Application, it produces multiple rows in this group.

Measurement Type

Upload Results To Repository

Datasets Measurements Properties

Group by: Measurement Type

Select Measurement Types* to include:

<input type="checkbox"/>	Measurement Type Name	Limits	Pending	In Repository
<input type="checkbox"/>	HardcodedMaxLimit	< 10.00		None
<input type="checkbox"/>	HardcodedMaxLimitWithUnits	<= 10.00 ub		None
<input type="checkbox"/>	HardcodedMinLimit	> 0.000		None
<input type="checkbox"/>	HardcodedMinLimitWithUnits	>= 0.000 ua		None
<input type="checkbox"/>	HardcodedRangeLimit	(200 to 3.000 k)		None
<input type="checkbox"/>	HardcodedRangeLimitWithUnits	[200 uc to 3.000 kuc]		None
<input type="checkbox"/>	HardcodedSingleValueLimit	= 40.00 k		None
<input type="checkbox"/>	HardcodedSingleValueLimitWithUnits	= 40.00 kud		None
<input type="checkbox"/>	VariableMaxLimit	< 160.0		None
<input type="checkbox"/>	VariableMaxLimit	< 120.0		None
<input type="checkbox"/>	VariableMaxLimit	< 80.0		None
<input type="checkbox"/>	VariableMaxLimit	< 10.0		None
<input type="checkbox"/>	VariableMaxLimitWithUnits	<= 20.0 ub		None
<input type="checkbox"/>	VariableMaxLimitWithUnits	<= 170.0 ub		None
<input type="checkbox"/>	VariableMaxLimitWithUnits	<= 130.0 ub		None
<input type="checkbox"/>	VariableMaxLimitWithUnits	<= 90.0 ub		None
<input type="checkbox"/>	VariableMinLimit	> 10.00		None
<input type="checkbox"/>	VariableMinLimit	> 130.00		None
<input type="checkbox"/>	VariableMinLimit	> 90.00		None
<input type="checkbox"/>	VariableMinLimit	> 50.00		None
<input type="checkbox"/>	VariableMinLimitWithUnits	>= 60.00 ua		None
<input type="checkbox"/>	VariableMinLimitWithUnits	>= 140.00 ua		None
<input type="checkbox"/>	VariableMinLimitWithUnits	>= 20.00 ua		None
<input type="checkbox"/>	VariableMinLimitWithUnits	>= 100.00 ua		None
<input type="checkbox"/>	VariableRangeLimit	(150 to 160)		None
<input type="checkbox"/>	VariableRangeLimit	(30 to 40)		None
<input type="checkbox"/>	VariableRangeLimit	(110 to 120)		None

(Click on a Measurement Type Name to see measurements.)

*Each row contains results from all executions of a single checked test.

Pending Upload Summary
(No measurements marked for upload.)

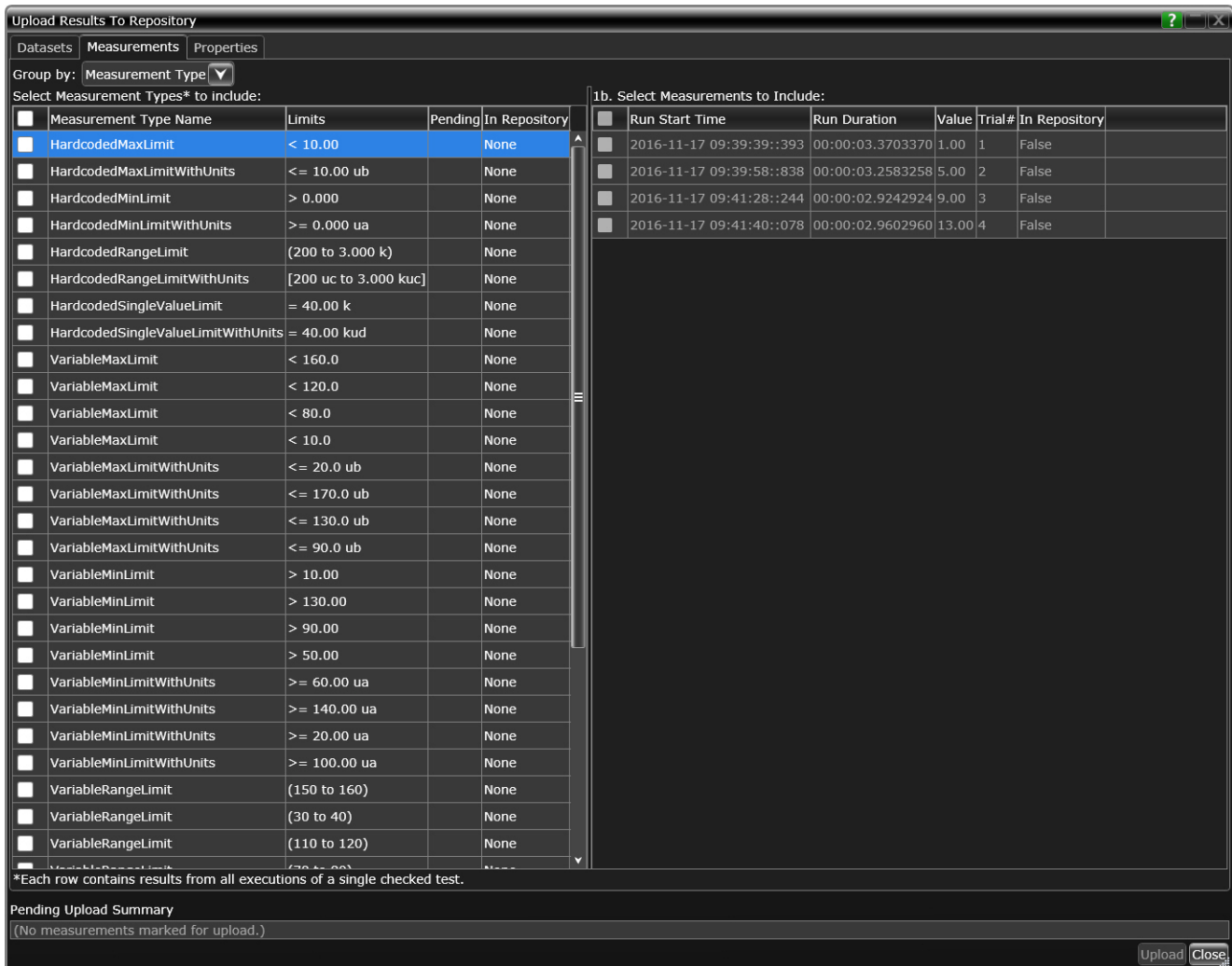
Upload Close

The left pane of the **Measurement Type** group has the following columns:

- Select check boxes – Select either one, more than one or all measurement types whose trials you wish to upload.
- **Measurement Type Name** – Displays the measurement types. Note that the measurement types are displayed in alphabetical order.
- **Limits** – Displays the specification-based or user-defined pass limits configured in the Test Application before running the test trial. The value displayed here is the same as that displayed in the **Pass Limits** column of the **Results** tab in the Test Application for the respective test.
- **Pending** – Displays either **All** or **Partial** for each row of measurement type.
 - **All** indicates that on the right pane, you have selected all trials of the corresponding measurement type.

- **Partial** indicates that on the right pane, you have selected some (and not all) trials of the corresponding measurement type.
- **In Repository** – Displays either **NONE**, **PARTIAL**, or **ALL** for each row of measurement type.
 - **NONE** indicates that none of the trials for the corresponding measurement type have been uploaded to the web repository.
 - **PARTIAL** indicates that some, but not all of the trials for the corresponding measurement type have been uploaded to the web repository.
 - **ALL** indicates that all of the trials for the corresponding measurement type have been uploaded to the web repository.

Highlight a specific row, without selecting the check box, if you wish to simply view the measurement sets and trials associated with a measurement type. The measurement sets on the right pane are grayed out until you select the check-box for one or more measurement types on the left pane.



The right pane of the **Measurement Type** group has the following columns:

- Select check boxes – Select either one, more than one or all measurement sets whose trials you wish to upload.
- **Run Start Time** – Displays the date and time (in YYYY-MM-DD hh:mm:ss::nnn format, where nnn is milliseconds) when the measurement set had started running on the Test Application. Note that the measurement sets are displayed in the increasing order of time.
- **Run Duration** – Displays the time taken for the tests in each measurement set to finish running.
- **Value** – Displays the value attained by the Test Application after running the test trial. The value displayed here is the same as that displayed in the **Actual Value** column of the **Results** tab in the Test Application for the respective test.
- **Trial#** – Displays the trial number of a measurement type. Each trial of a measurement is included in a separate measurement set.

- **In Repository** – Displays either **TRUE** or **FALSE** for each row of measurement type.
 - **TRUE** indicates that the measurement trial has been uploaded to the web repository.
 - **FALSE** indicates that the measurement trial has not been uploaded to the web repository.

NOTE

As specified in the **Upload Results To Repository** window, each row in the **Measurement Type** group contains results from all test runs of a selected test in the Test Application.

Selecting and Uploading Results

Irrespective of the grouping method you apply, the process of selecting and uploading the measurement results is the same for both groups.

- 1 Click the check box at the top if you wish to select all entries in the left pane.

Notice that the **Upload** button is enabled even if you select one trial on the right pane.

The **Pending Upload Summary:** pane displays "No measurements marked for upload" by default. The information in this pane changes as soon as you select one or more measurement sets. Notice that the **Pending Upload Summary:** pane displays a summary of the connection and the number of measurement results that are selected and ready for upload. The latter information varies if you select/deselect any measurement results or sets.

Upload Results To Repository

Datasets Measurements Properties

Group by: Run Start Time

Select Run Start Times* to include:

<input checked="" type="checkbox"/>	Run Start Time	Run Duration	Pending	In Repository
<input checked="" type="checkbox"/>	2016-11-15 14:40:28::588	00:00:03.0483048	All	None
<input checked="" type="checkbox"/>	2016-11-15 15:05:02::576	00:00:02.7182718	All	None
<input checked="" type="checkbox"/>	2016-11-15 15:06:24::062	00:00:02.0362036	All	None

1b. Select Measurements to Include:

<input checked="" type="checkbox"/>	Measurement	Limits	Value	Trial#	In Repository
<input checked="" type="checkbox"/>	HardcodedMaxLimit	< 10.00	17.00	3	False
<input checked="" type="checkbox"/>	HardcodedMaxLimitWithUnits	<= 10.00 ub	18.00ub	3	False
<input checked="" type="checkbox"/>	HardcodedRangeLimit	(200 to 3.000 k)	170	3	False
<input checked="" type="checkbox"/>	HardcodedRangeLimitWithUnits	[200 uc to 3.000 kuc]	180uc	3	False
<input checked="" type="checkbox"/>	HardcodedSingleValueLimit	= 40.00 k	1.70 k	3	False
<input checked="" type="checkbox"/>	HardcodedSingleValueLimitWithUnits	= 40.00 kud	1.80 kud	3	False
<input checked="" type="checkbox"/>	VariableMinLimit	> 170.00	1.70	3	False
<input checked="" type="checkbox"/>	VariableMinLimitWithUnits	>= 180.00 ua	1.80ua	3	False
<input checked="" type="checkbox"/>	VariableRangeLimit	(190 to 200)	190	3	False
<input checked="" type="checkbox"/>	VariableSingleValueLimit	= 200	1.9 k	3	False
<input checked="" type="checkbox"/>	VariableSingleValueLimitWithUnits	= 200 ud	2.0 kud	3	False

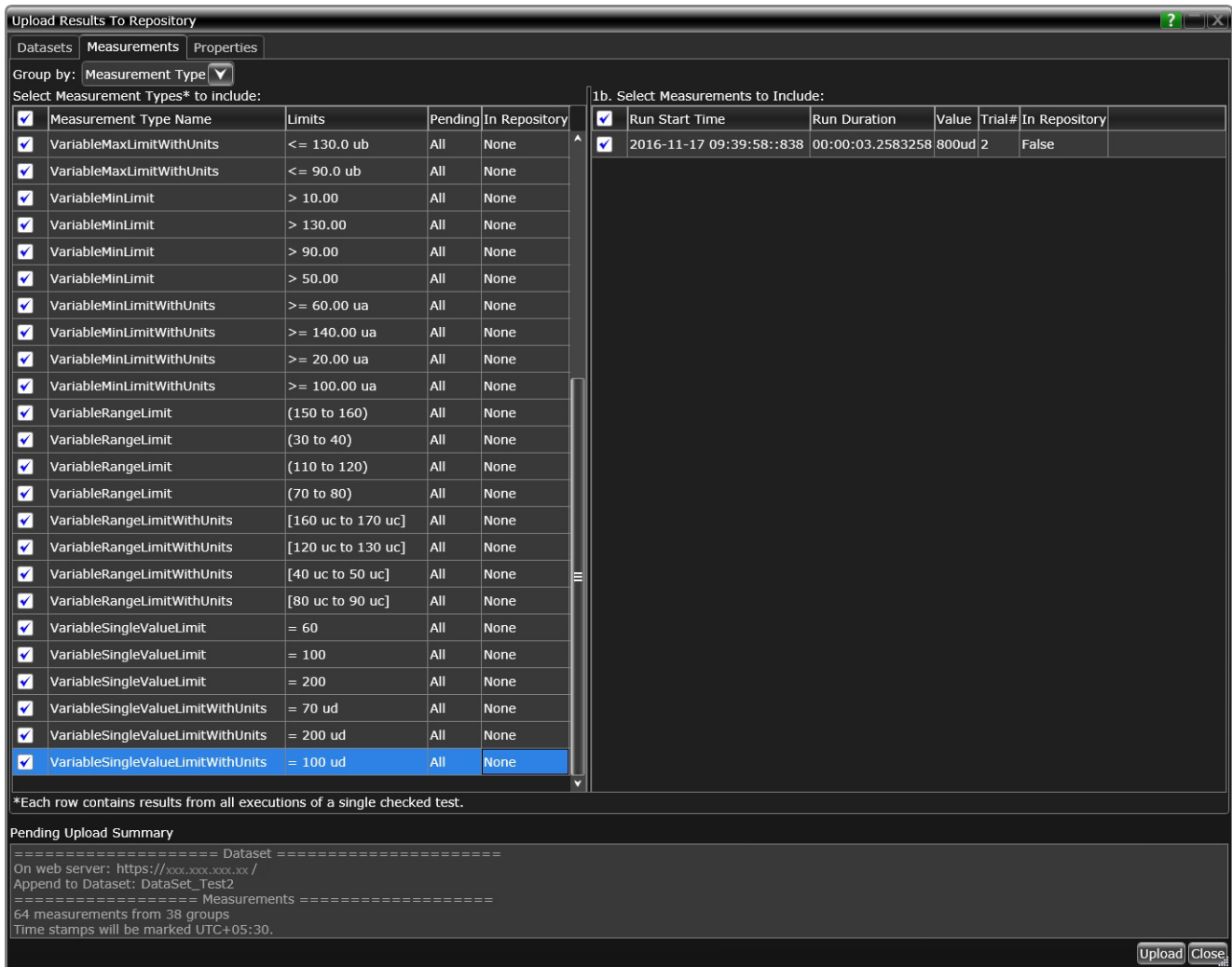
*Each row contains results from a single execution of all checked tests; an N-Times or TestPlan execution produces multiple rows.

Pending Upload Summary

```

===== Dataset =====
On web server: https://xxx.xxx.xxx.xxx/
Append to Dataset: DataSet_Test2
===== Measurements =====
43 measurements from 3 groups
Time stamps will be marked UTC+05:30.
    
```

Upload Close



- 2 You may deselect the check boxes on either side to reduce the number of measurement results to be uploaded.

Upload Results To Repository

Datasets Measurements Properties

Group by: Run Start Time

Select Run Start Times* to include:

<input type="checkbox"/>	Run Start Time	Run Duration	Pending	In Repository
<input type="checkbox"/>	2016-11-15 14:40:28::588	00:00:03.0483048		None
<input checked="" type="checkbox"/>	2016-11-15 15:05:02::576	00:00:02.7182718	Partial	None
<input type="checkbox"/>	2016-11-15 15:06:24::062	00:00:02.0362036		None

1b. Select Measurements to Include:

<input type="checkbox"/>	Measurement	Limits	Value	Trial#	In Repository
<input checked="" type="checkbox"/>	HardcodedMaxLimit	< 10.00	13.00	2	False
<input checked="" type="checkbox"/>	HardcodedMaxLimitWithUnits	<= 10.00 ub	14.00ub	2	False
<input type="checkbox"/>	HardcodedMinLimit	> 0.000	1.300	2	False
<input type="checkbox"/>	HardcodedMinLimitWithUnits	>= 0.000 ua	1.400ua	2	False
<input type="checkbox"/>	HardcodedRangeLimit	(200 to 3.000 k)	130	2	False
<input type="checkbox"/>	HardcodedRangeLimitWithUnits	[200 uc to 3.000 kuc]	140uc	2	False
<input type="checkbox"/>	HardcodedSingleValueLimit	= 40.00 k	1.30 k	2	False
<input type="checkbox"/>	HardcodedSingleValueLimitWithUnits	= 40.00 kud	1.40 kud	2	False
<input type="checkbox"/>	VariableMaxLimit	< 160.0	15.0	2	False
<input type="checkbox"/>	VariableMaxLimitWithUnits	<= 170.0 ub	16.0ub	2	False
<input type="checkbox"/>	VariableMinLimit	> 130.00	1.50	2	False
<input type="checkbox"/>	VariableMinLimitWithUnits	>= 140.00 ua	1.60ua	2	False
<input type="checkbox"/>	VariableRangeLimit	(150 to 160)	150	2	False
<input type="checkbox"/>	VariableRangeLimitWithUnits	[160 uc to 170 uc]	160uc	2	False
<input type="checkbox"/>	VariableSingleValueLimit	= 200	1.5 k	2	False
<input type="checkbox"/>	VariableSingleValueLimitWithUnits	= 200 ud	1.6 kud	2	False

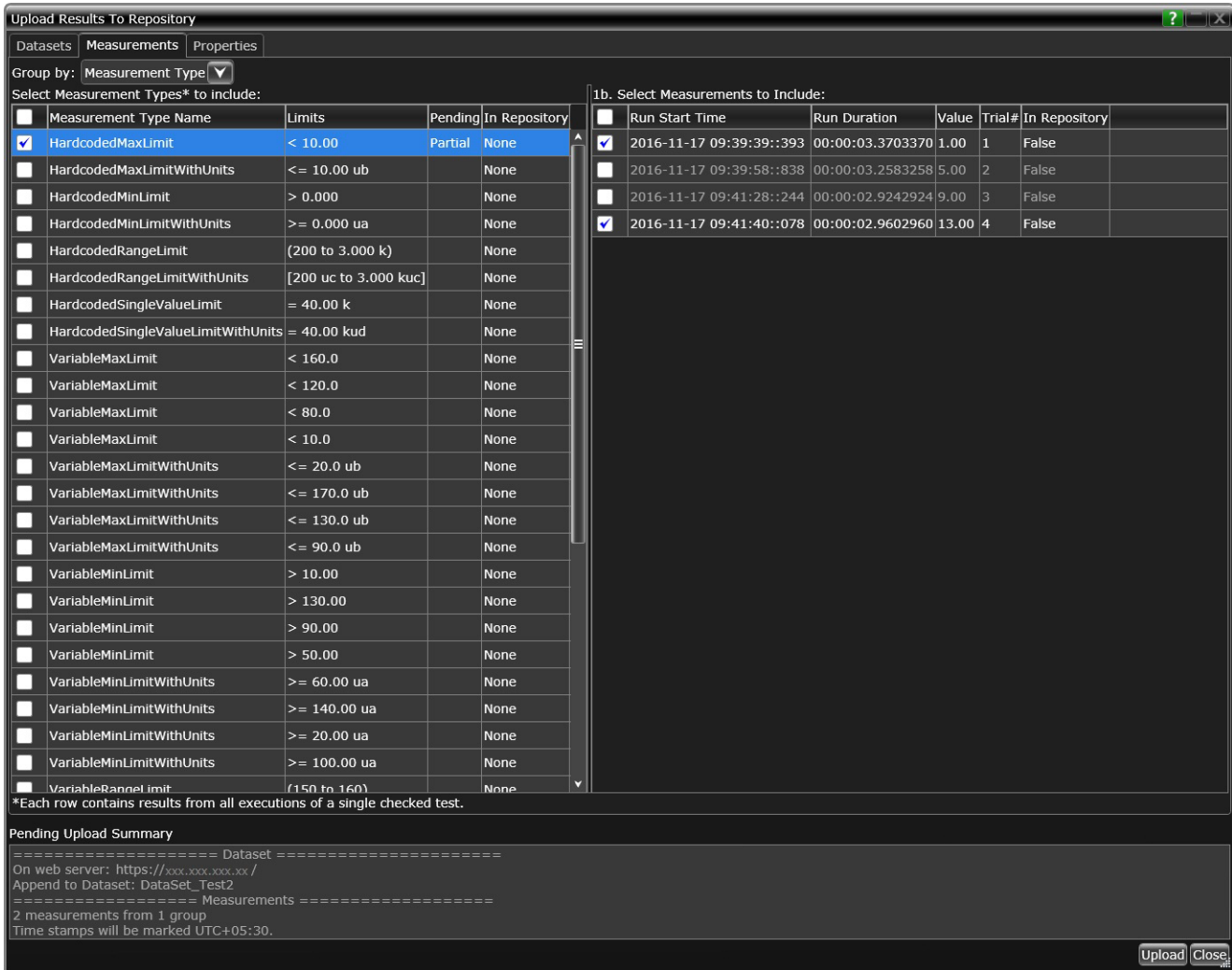
*Each row contains results from a single execution of all checked tests; an N-Times or TestPlan execution produces multiple rows.

Pending Upload Summary

```

===== Dataset =====
On web server: https://xxx.xxx.xxx.xxx /
Append to Dataset: DataSet_Test2
===== Measurements =====
2 measurements from 1 group
Time stamps will be marked UTC+05:30.
    
```

Upload Close



- 3 After you select the measurement types that you desire to export to the web repository and before you begin uploading, click the **Properties** tab to define or modify the properties of the selected Dataset (see [page 352](#)).

NOTE

It is imperative that you define or modify Dataset properties before you begin uploading the measurement results to the Dataset on the web repository. Once the upload begins, you cannot associate any properties to the selected Dataset, which directly impacts the process of data analysis later.

- 4 Click the **Upload** button to begin exporting the selected measurement results to the selected Dataset on the Web Repository that the Test Application is connected to. Notice that as soon as the Test Application finishes uploading, the **Measurements** tab returns to the default view.

Upload Results To Repository

Datasets Measurements Properties

Group by: Run Start Time

Select Run Start Times* to include:

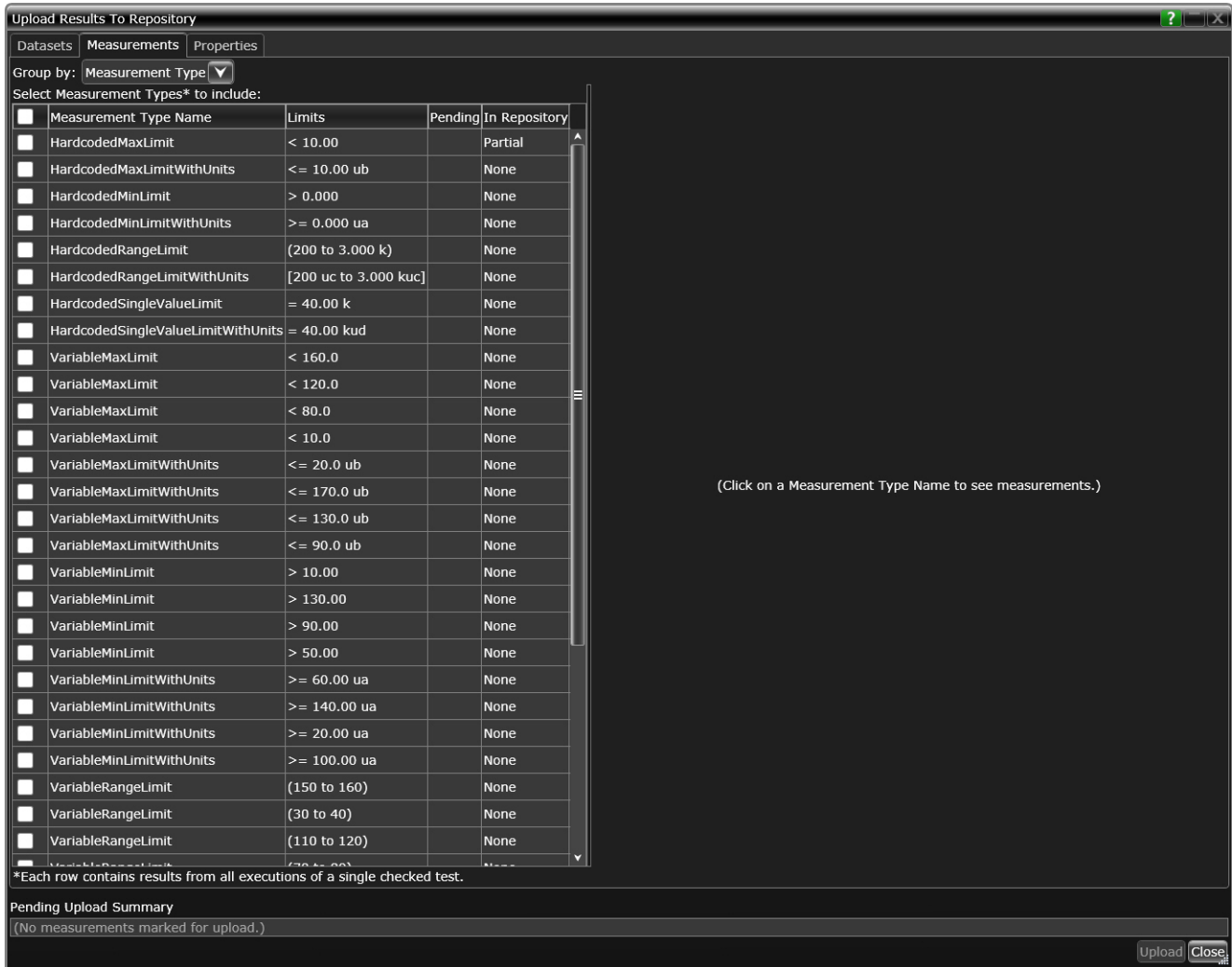
<input type="checkbox"/>	Run Start Time	Run Duration	Pending	In Repository
<input type="checkbox"/>	2016-11-15 14:40:28::588	00:00:03.0483048		None
<input type="checkbox"/>	2016-11-15 15:05:02::576	00:00:02.7182718		Partial
<input type="checkbox"/>	2016-11-15 15:06:24::062	00:00:02.0362036		None

(Click on a Run Start Time to see measurements.)

*Each row contains results from a single execution of all checked tests; an N-Times or TestPlan execution produces multiple rows.

Pending Upload Summary
(No measurements marked for upload.)

Upload Close



If you select any row, which you had selected earlier to upload measurement results, you will find that the uploaded trials are now grayed out and cannot be selected any further. Notice that the status of **In Repository** row in the right pane of the uploaded results has changed to **True**.

Upload Results To Repository

Datasets Measurements Properties

Group by: Run Start Time

Select Run Start Times* to include:

<input type="checkbox"/>	Run Start Time	Run Duration	Pending	In Repository
<input type="checkbox"/>	2016-11-15 14:40:28::588	00:00:03.0483048		None
<input checked="" type="checkbox"/>	2016-11-15 15:05:02::576	00:00:02.7182718	All	Partial
<input type="checkbox"/>	2016-11-15 15:06:24::062	00:00:02.0362036		None

1b. Select Measurements to Include:

<input checked="" type="checkbox"/>	Measurement	Limits	Value	Trial#	In Repository
<input type="checkbox"/>	HardcodedMaxLimit	< 10.00	13.00	2	True
<input type="checkbox"/>	HardcodedMaxLimitWithUnits	<= 10.00 ub	14.00ub	2	True
<input checked="" type="checkbox"/>	HardcodedMinLimit	> 0.000	1.300	2	False
<input checked="" type="checkbox"/>	HardcodedMinLimitWithUnits	>= 0.000 ua	1.400ua	2	False
<input checked="" type="checkbox"/>	HardcodedRangeLimit	(200 to 3.000 k)	130	2	False
<input checked="" type="checkbox"/>	HardcodedRangeLimitWithUnits	[200 uc to 3.000 kuc]	140uc	2	False
<input checked="" type="checkbox"/>	HardcodedSingleValueLimit	= 40.00 k	1.30 k	2	False
<input checked="" type="checkbox"/>	HardcodedSingleValueLimitWithUnits	= 40.00 kud	1.40 kud	2	False
<input checked="" type="checkbox"/>	VariableMaxLimit	< 160.0	15.0	2	False
<input checked="" type="checkbox"/>	VariableMaxLimitWithUnits	<= 170.0 ub	16.0ub	2	False
<input checked="" type="checkbox"/>	VariableMinLimit	> 130.00	1.50	2	False
<input checked="" type="checkbox"/>	VariableMinLimitWithUnits	>= 140.00 ua	1.60ua	2	False
<input checked="" type="checkbox"/>	VariableRangeLimit	(150 to 160)	150	2	False
<input checked="" type="checkbox"/>	VariableRangeLimitWithUnits	[160 uc to 170 uc]	160uc	2	False
<input checked="" type="checkbox"/>	VariableSingleValueLimit	= 200	1.5 k	2	False
<input checked="" type="checkbox"/>	VariableSingleValueLimitWithUnits	= 200 ud	1.6 kud	2	False

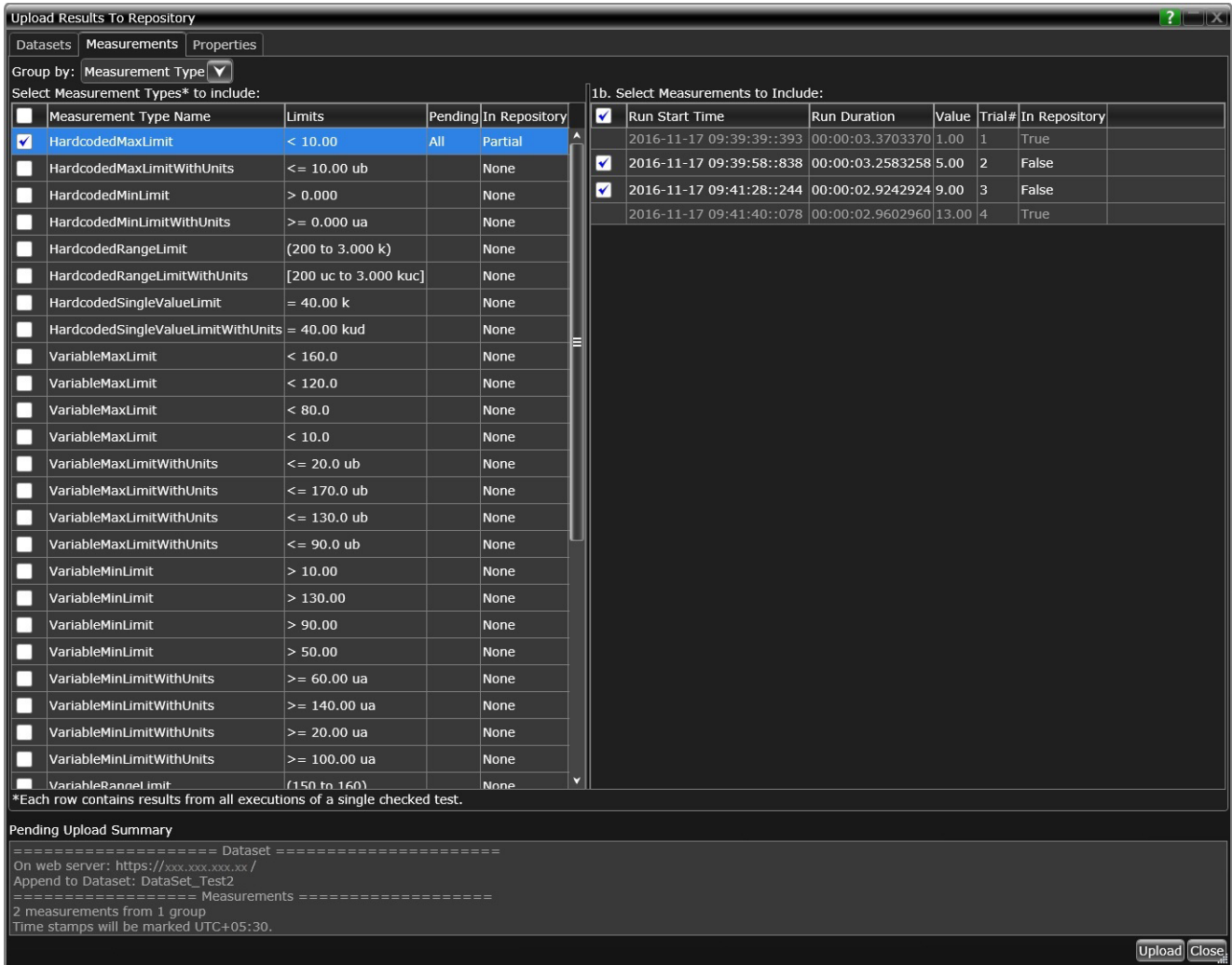
*Each row contains results from a single execution of all checked tests; an N-Times or TestPlan execution produces multiple rows.

Pending Upload Summary

```

===== Dataset =====
On web server: https://xxx.xxx.xxx.xx/
Append to Dataset: DataSet_Test4
===== Measurements =====
14 measurements from 1 group
Time stamps will be marked UTC+05:30.
    
```

Upload Close



- Repeat steps 1 to 4 if you wish to select and upload some more or all measurement results to the Web Repository.

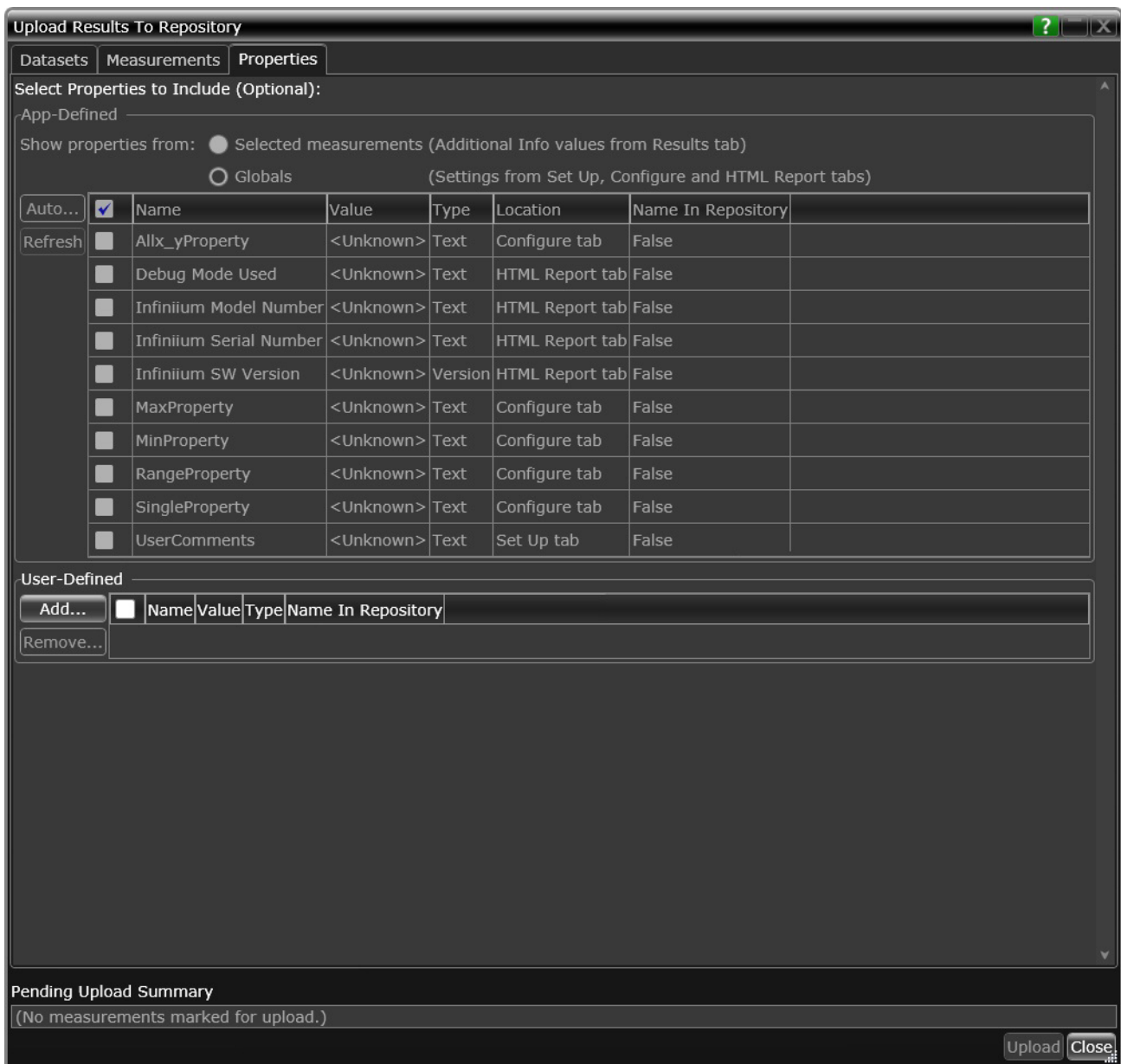
Defining and Modifying Dataset Properties

The Dataset Properties define the metadata to the Dataset where you are uploading the measurement results to. While the Test Application automatically defines certain properties to your Dataset, you have the option to also assign similar or unique properties to the Dataset. Using Dataset Properties is optional; however, Keysight recommends using this feature before uploading measurement results to your selected Dataset. These properties provide unique identification to one or more set of measurement results when you perform Data analysis, such that you can make distinctions between a range of data for a specific measurement type.

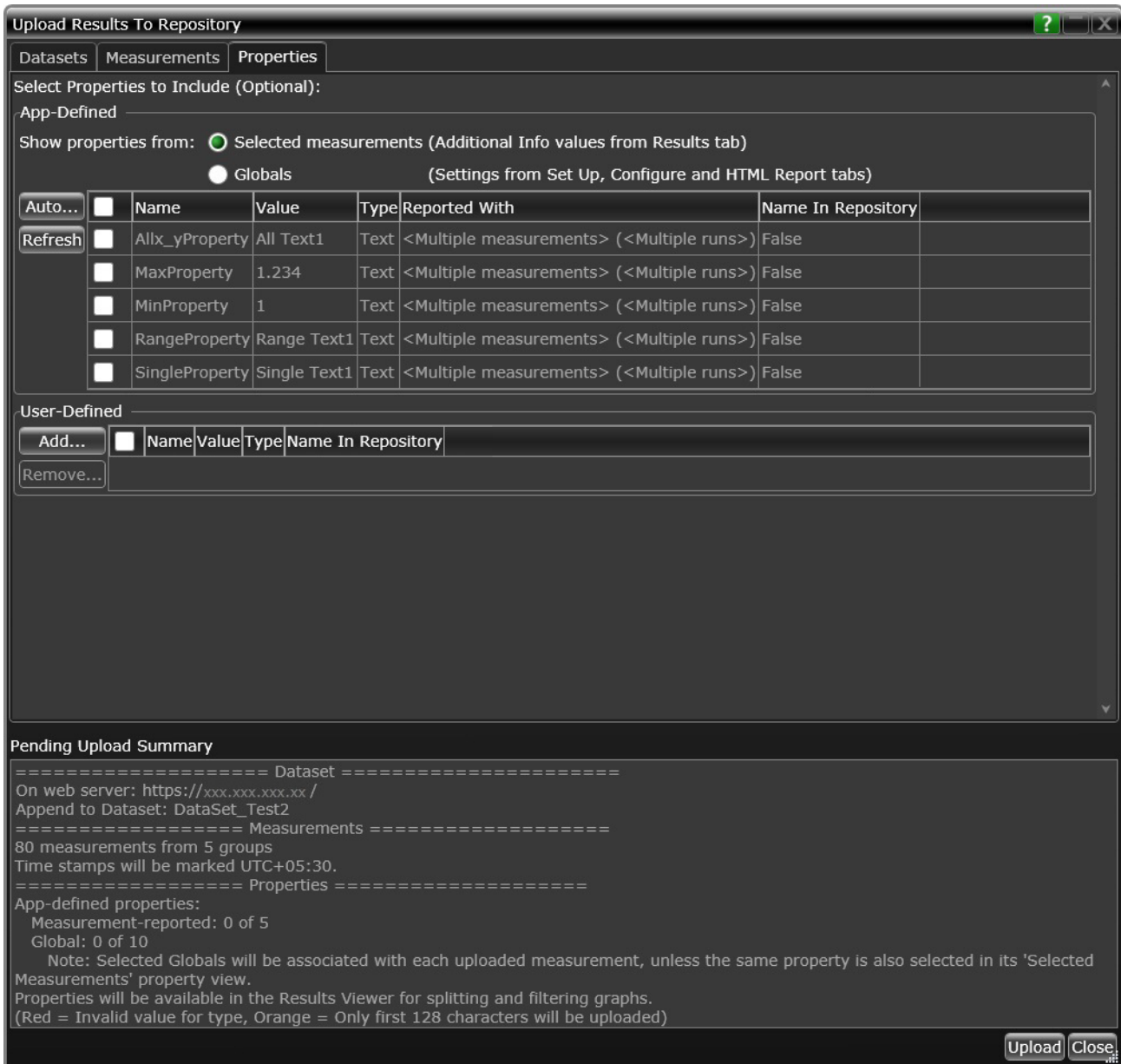
NOTE

It is imperative that you define or modify Dataset properties before you begin uploading the measurement results to the Dataset on the web repository. Once the upload begins, you cannot associate any properties to the uploaded measurements in the selected Dataset. This may impact the process of data analysis later.

To access the **Properties** tab, select a Dataset (see [page 333](#)) whose properties you wish to add or modify. By default, the **Properties** tab appears as shown in the image below, without any measurement sets or measurement types selected under the **Measurements** tab. You may add / modify only user-defined properties to the selected Dataset.



When you select measurement results (see [page 336](#)) under the **Measurements** tab, the **Properties** tab appears similar to the following image.



The **Properties** tab consists of two categories of properties that you may associate with the measurements in a Dataset.

- **App-Defined** (see [page 356](#)) – lists the options that are pre-defined or selected within the Test Application for the selected measurements.
- **User-Defined** (see [page 369](#)) – lists custom property names that one or more users or administrators may have created for the selected Dataset. These property names could be used to define certain characteristics (such as, Engineer name & designation) of the measurement results, which may be helpful when performing data analysis.

NOTE

For properties, whose values exceed 128 characters, only the first 128 characters are uploaded. Property values exceeding that character limit are highlighted and explanation text is included in the Pending Upload Summary area. In the User-Defined group, each entry field is limited to values with 128 characters only.

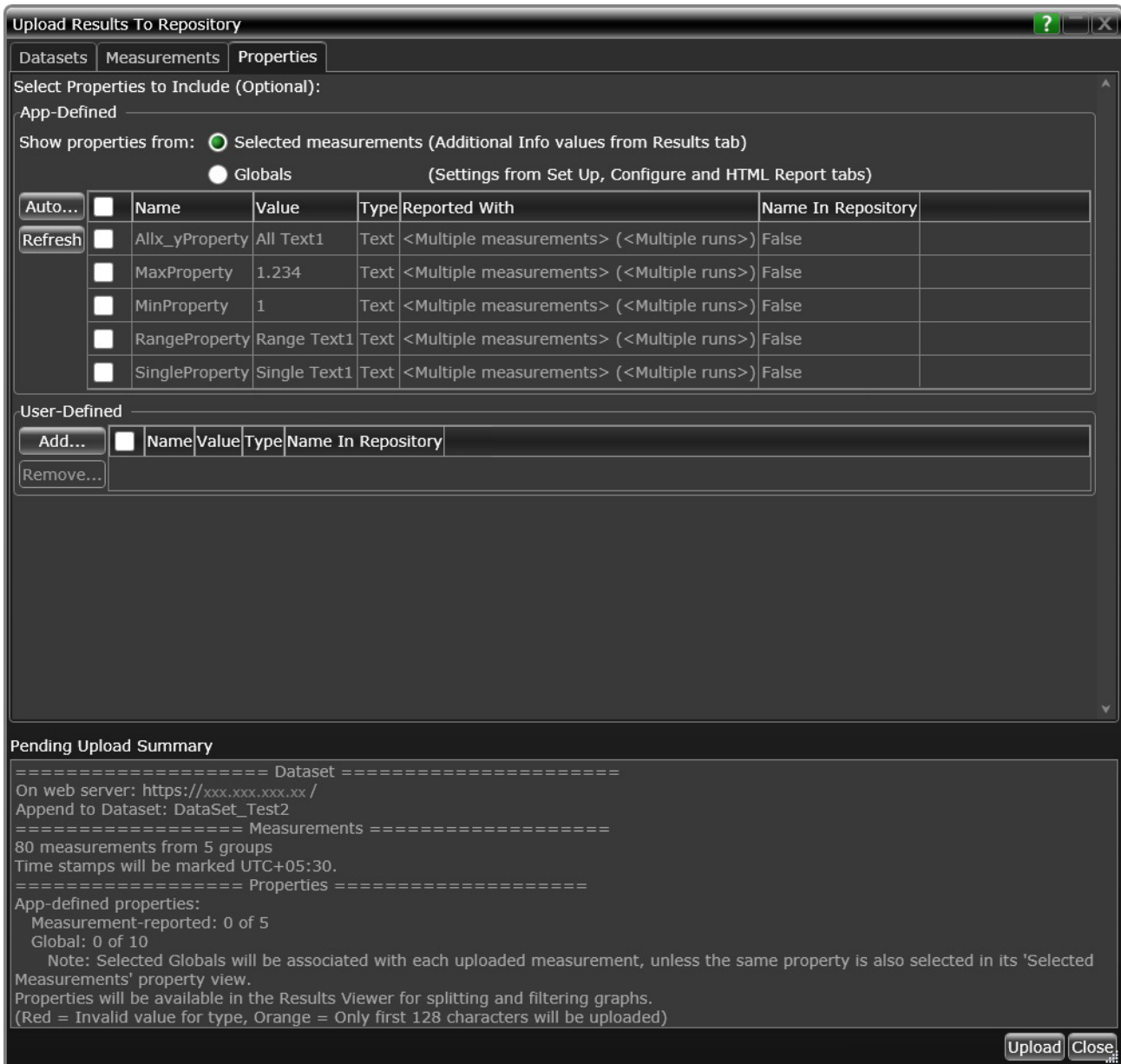
App-Defined Properties

The **App-Defined** properties consists of all such options/attributes within the Test Application, which you select/configure before running automated tests. The App-Defined properties are displayed under two sub-categories:

- **Selected measurements**
- **Globals**

Selected measurements

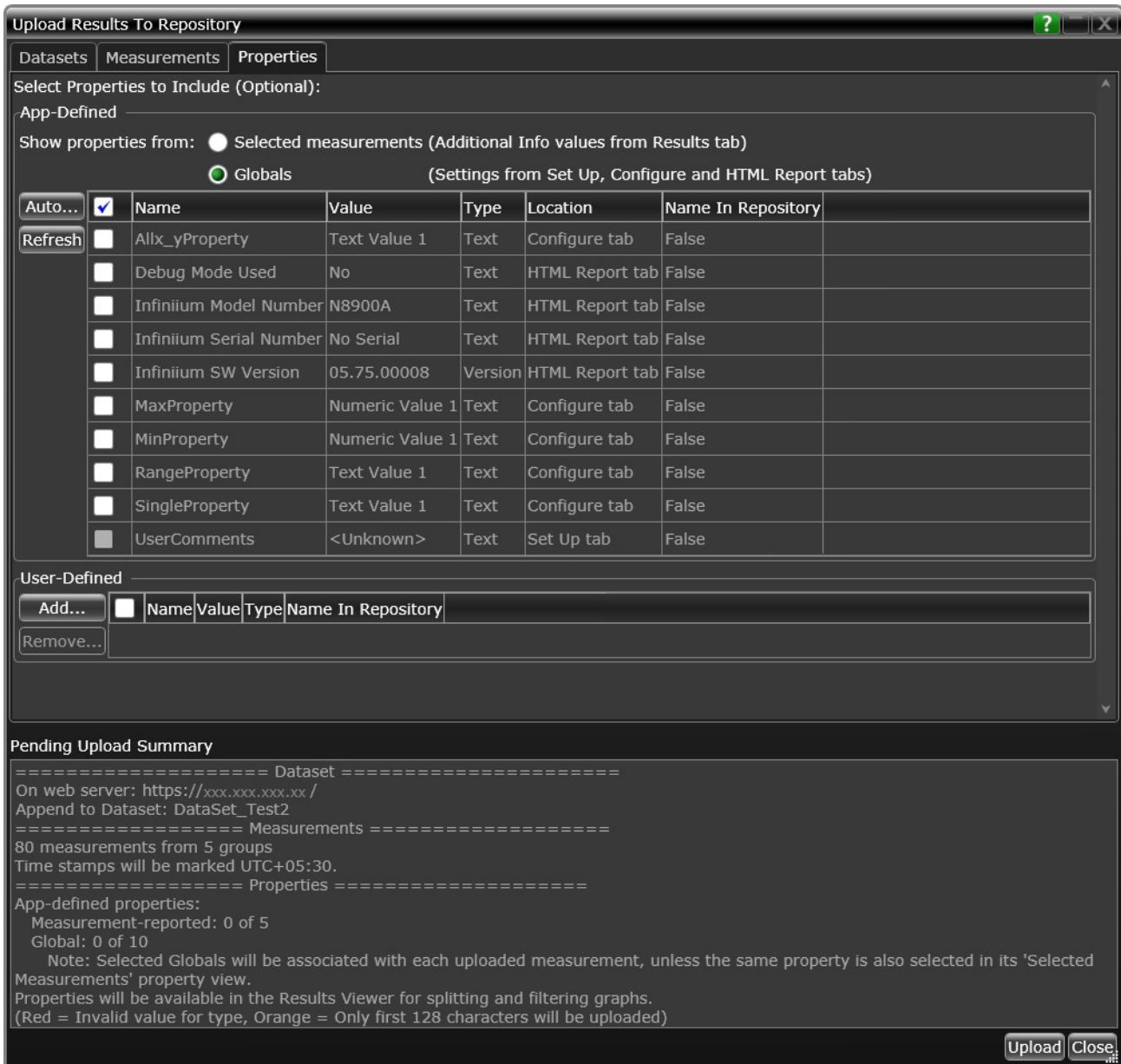
The properties displayed under **Selected measurements** are extracted from the "Additional Info" area within the **Results** tab and vary in number based on the selected number and type of measurement results.



Following image displays the attributes within the "Additional Info" area of the **Results** tab, which are extracted as properties.

Parameter	Value
VariableSingleValueLimitWithUnits	400 ud
---Additional Info---	
Allx_yProperty	All Text1
AllNoRepositoryProperty	You can type anything you want in here
SingleProperty	Single Text1
PassLimit Required (MaxLimitVar)	70 ud

Globals The properties displayed under **Globals** are extracted from various tabs within the Test Application, that is, the **Set Up** tab, **Configure** tab and the "Test Session Details" area of the **HTML** tab. Only those options/attributes that you select under these tabs prior to running the tests are displayed in this list.

**NOTE**

You may select properties listed under either **Selected measurements**, **Globals**, or both. If you select properties from both sub-categories, all selections are uploaded to the Dataset.

You may find some common property names listed under both **Selected measurements** and **Globals**. In such a scenario, even if you select the same property under both the sub-categories, the Test Application uploads the property only once for a measurement result.

**Features in the
App-Defined
section**

The **App-Defined** section, in general, consists of features common to both property types. Notice that the properties are listed in a tabular format. Other features are:

- **Show properties from:** – Select either **Selected measurements (Additional Ino values from Results tab)** or **Globals (Settings from Set Up, Configure and HTML Report tabs)**.
- **Select check boxes** – Select either one, more than one or all property names that you wish to define for the selected Dataset. Click the check box at the top if you wish to select all app-defined property names.
- **Name** – The pre-defined property names, associated with one or more measurement results, are displayed.
- **Value** – Depending on the property type, a corresponding numeric or alphanumeric value is displayed for each property name.

Upload Results To Repository

Datasets | Measurements | **Properties**

Select Properties to Include (Optional):

App-Defined

Show properties from: Selected measurements (Additional Info values from Results tab)
 Globals (Settings from Set Up, Configure and HTML Report tabs)

Auto...	<input checked="" type="checkbox"/>	Name	Value	Type	Reported With	Name In Repository
Refresh	<input checked="" type="checkbox"/>	Allx_yProperty	All Text1	Text	<Multiple measurements> (<Multiple runs>)	False
	<input checked="" type="checkbox"/>	MaxProperty	1.234	Text	<Multiple measurements> (<Multiple runs>)	False
	<input checked="" type="checkbox"/>	MinProperty	1	Text	<Multiple measurements> (<Multiple runs>)	False
	<input checked="" type="checkbox"/>	RangeProperty	Range Text1	Text	<Multiple measurements> (<Multiple runs>)	False
	<input checked="" type="checkbox"/>	SingleProperty	Single Text1	Text	<Multiple measurements> (<Multiple runs>)	False

User-Defined

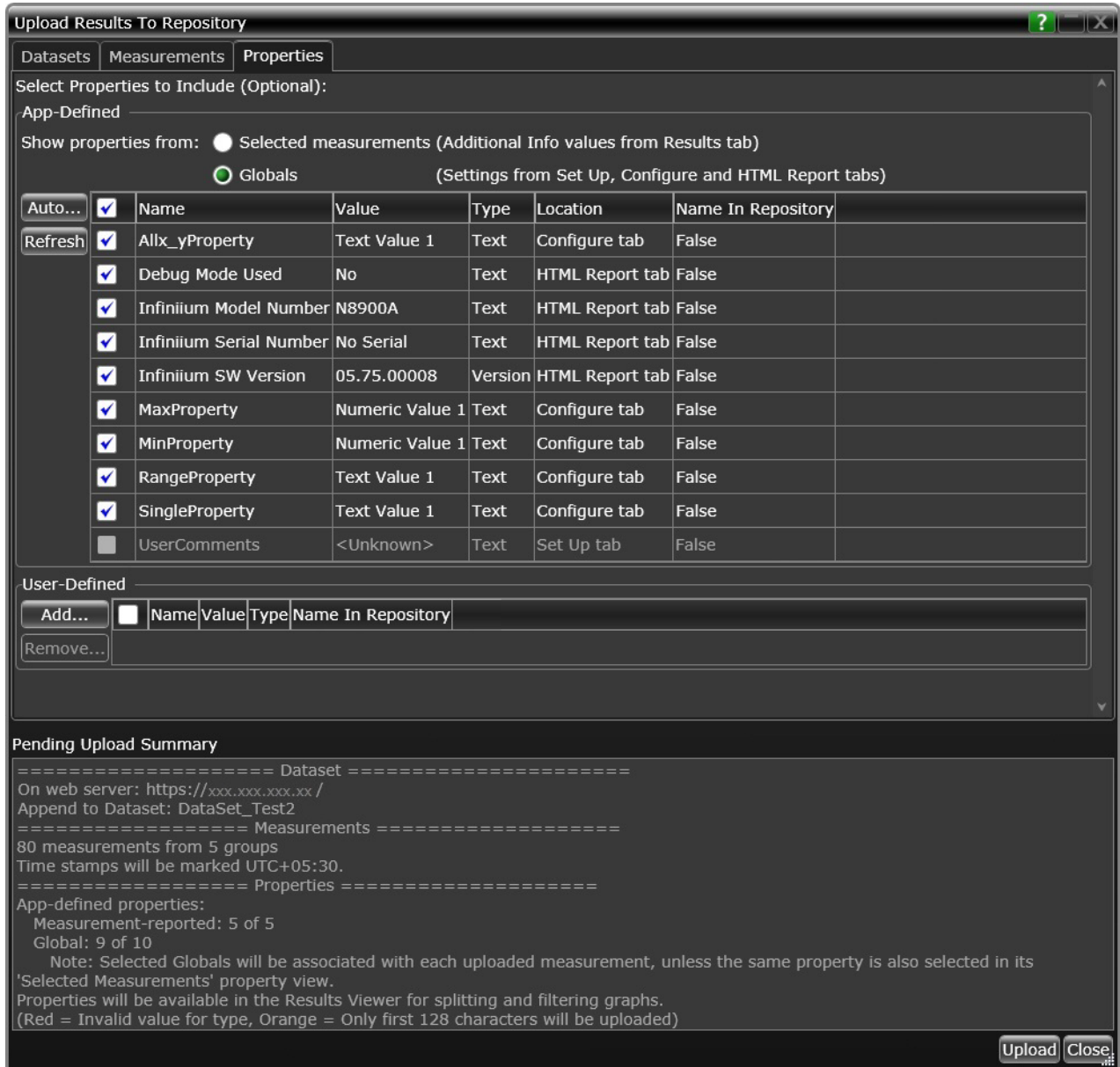
Add...	<input type="checkbox"/>	Name	Value	Type	Name In Repository
Remove...					

Pending Upload Summary

```

===== Dataset =====
On web server: https://xxx.xxx.xxx.xx /
Append to Dataset: DataSet_Test2
===== Measurements =====
80 measurements from 5 groups
Time stamps will be marked UTC+05:30.
===== Properties =====
App-defined properties:
  Measurement-reported: 5 of 5
  Global: 0 of 10
Note: Selected Globals will be associated with each uploaded measurement, unless the same property is also selected in its
'Selected Measurements' property view.
Properties will be available in the Results Viewer for splitting and filtering graphs.
(Red = Invalid value for type, Orange = Only first 128 characters will be uploaded)
    
```

Upload Close



- **Reported With** – Indicates whether the property name is associated with:
 - One measurement type and one measurement set, or

<input type="checkbox"/>	Name	Value	Reported With	Name In Repository
<input type="checkbox"/>	Allx_yProperty		HardcodedMaxLimit (2017-01-02 11:41:30::445)	True
<input type="checkbox"/>	MaxProperty	1.234	HardcodedMaxLimit (2017-01-02 11:41:30::445)	True

- One measurement type and multiple measurement sets, or

<input type="checkbox"/>	Name	Value	Reported With	Name In Repository
<input type="checkbox"/>	Allx_yProperty	All Text1	HardcodedMaxLimit (<Multiple sets>)	True
<input type="checkbox"/>	MaxProperty	1.234	HardcodedMaxLimit (<Multiple sets>)	True

- Multiple measurement types and one measurement set, or

<input type="checkbox"/>	Name	Value	Reported With	Name In Repository
<input type="checkbox"/>	Allx_yProperty	All Text1	<Multiple Measurements> (2017-01-02 11:41:30::445)	True
<input type="checkbox"/>	MaxProperty	1.234	<Multiple Measurements> (2017-01-02 11:41:30::445)	True

- Multiple measurement types and multiple measurement sets.

<input type="checkbox"/>	Name	Value	Reported With	Name In Repository
<input type="checkbox"/>	Allx_yProperty	All Text1	<Multiple Measurements> (<Multiple sets>)	True
<input type="checkbox"/>	MaxProperty	1.234	<Multiple Measurements> (<Multiple sets>)	True

- **Name In Repository** – Displays either **TRUE** or **FALSE** for each row of properties.
 - **TRUE** indicates that the property name has already been associated with a Dataset and has been uploaded to the web repository.
 - **FALSE** indicates that the property name has not yet been associated with a Dataset on the web repository.

In a real-time testing environment, the list of app-defined properties may become too long, thereby making it difficult for anyone to manage the list or even to recall whether or not certain property names have been defined for a Dataset. In order to avoid any conflicts and to avoid manual selection of properties every time, use the **Auto...** (see [page 363](#)) and **Refresh** (see [page 367](#)) buttons.

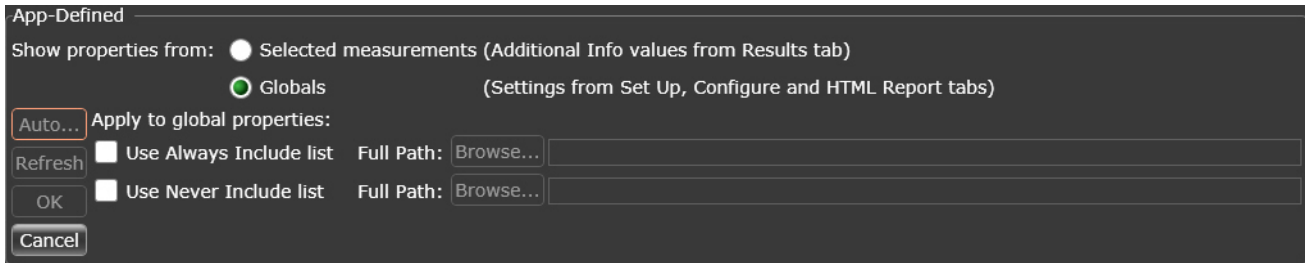
Auto... The **Auto...** button provides the ability to:

- always include only the desired properties before you upload measurement results to the selected Dataset.
- always exclude only those properties that you would never want to associate with your measurement results in the selected Dataset.

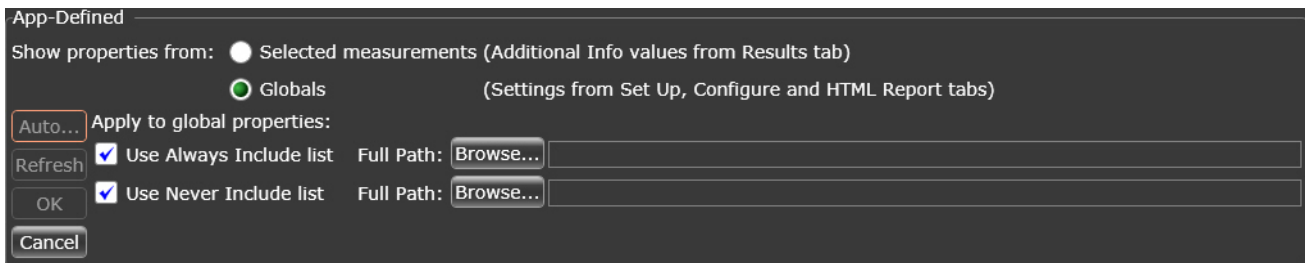
NOTE

The **Auto** feature works in the same manner for both **Selected measurements** and **Globals**. However, you cannot interchange properties between the sub-categories to mark inclusion/exclusion unless the property name is common to both sub-categories.

- 1 In the **Show properties from:** area, select either **Selected measurements** or **Globals**.
- 2 Click the **Auto...** button. The tabular view is replaced by the following options:



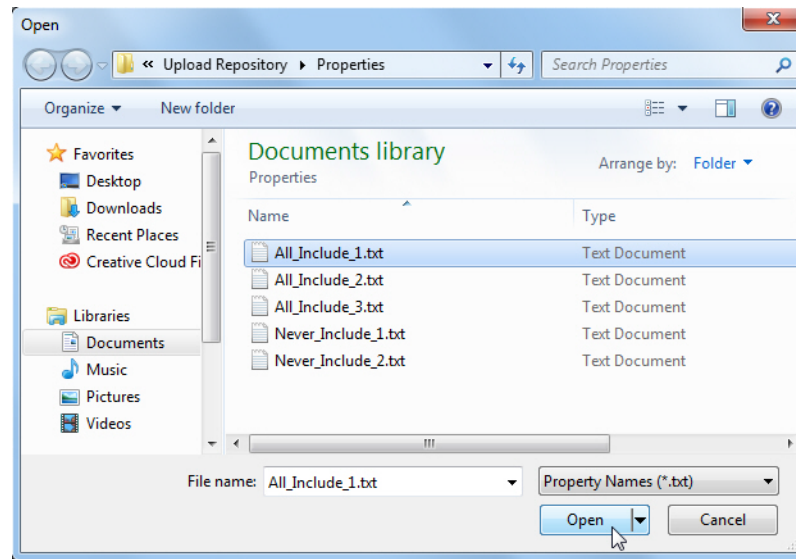
- Click the check-box either for **Use Always Include list** (to always include some properties to the selected Dataset), for **Use Never Include list** (to always exclude some properties from being associated with the selected Dataset) or for both options (to define both type of properties simultaneously. When you select either one or both the options, the **Full Path:** text field and the **Browse...** button are enabled.



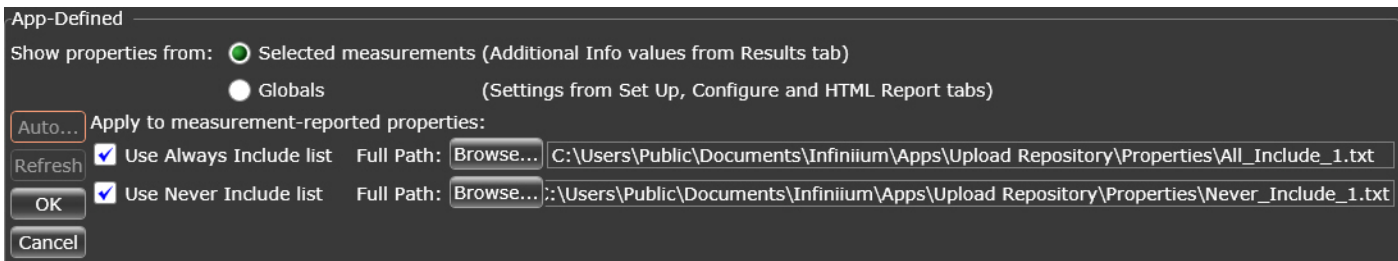
- In the **Full Path:** text field for the **Use Always Include list**, either manually type the entire file location for the property names text file (in the *.txt format) or click the **Browse...** button to navigate to the folder where the property names text file is located.

To create a property names text file, open the Notepad editor, type the name of each property you wish to include or exclude and save the file on your local disk. Each property name must begin on a new line.

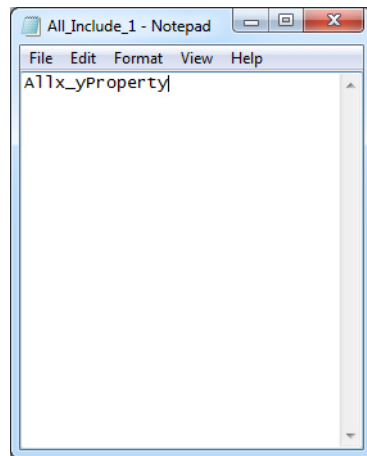
- In the **Open** dialog box, select the desired file and click **Open**. The **Full Path:** text field is populated with the file location for the property names text file.



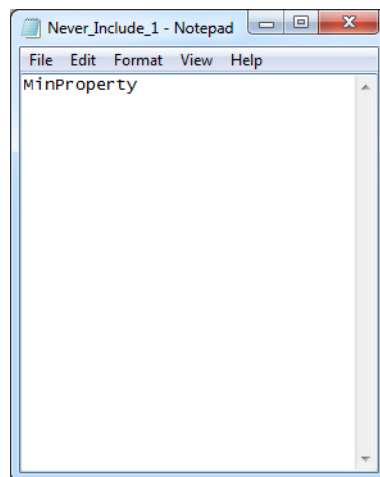
If required, repeat the previous two steps for the **Use Never Include list**, making sure that you select the correct property names text file that contains the property names that you wish to exclude from the selected Dataset.



Notice the contents of the property names text file for **Use Always Include list**. It indicates that the app-defined property named "Allx_yProperty" will always be included in the selected Dataset, whenever measurement results that have this property are uploaded to it.



Notice the contents of the property names text file for **Use Never Include list**. It indicates that the app-defined property named "MinProperty" will always be excluded from the selected Dataset, whenever measurement results are uploaded to it.



- 6 Click to save the changes. Notice the change in the table of properties, where the check-box for "Allx_yProperty" is selected but grayed out (always included) and the check-box and contents for "MinProperty" are entirely grayed out (never included) for the selected Dataset.

App-Defined

Show properties from: Selected measurements (Additional Info values from Results tab)
 Globals (Settings from Set Up, Configure and HTML Report tabs)

Auto Mode: [Meas] Always (1) + Never (1)

Auto...	<input type="checkbox"/>	Name	Value	Type	Reported With	Name In Repository
Refresh	<input checked="" type="checkbox"/>	Allx_yProperty	All Text1	Text	<Multiple measurements> (<Multiple runs>)	False
	<input type="checkbox"/>	MaxProperty	1.234	Text	<Multiple measurements> (<Multiple runs>)	False
	<input type="checkbox"/>	MinProperty	1	Text	<Multiple measurements> (<Multiple runs>)	False
	<input type="checkbox"/>	RangeProperty	Range Text1	Text	<Multiple measurements> (<Multiple runs>)	False
	<input type="checkbox"/>	SingleProperty	Single Text1	Text	<Multiple measurements> (<Multiple runs>)	False

Notice the text **Auto Mode: [Meas] Always (m) + Never (n)**, where m and n indicate the number of properties under **Selected measurements** selected to be included or excluded. In this case, $m = n = 1$.

This text indicates that Auto mode for Selected measurements is enabled. The number within the parentheses indicates that one property name has been used in the Always Include list and Never Include list, respectively.

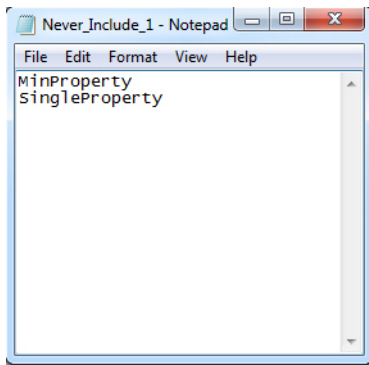
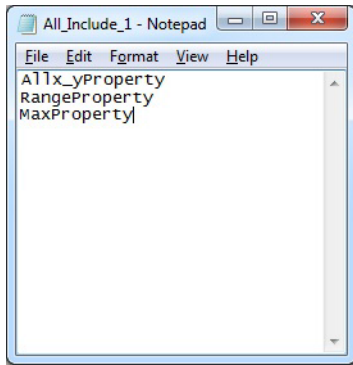
- Repeat steps 1 to 6, if you wish to include or exclude properties listed under **Globals**.

If you repeat these steps for **Globals**, the text **Auto Mode: [Meas] Always (m) + Never (n)** is modified to **Auto Mode: [Meas] Always (m) + Never (n) + [Globals] Always (x) + Never (y)**, where m and n indicate the number of properties under **Selected measurements** and x and y indicate the number of properties under **Globals** selected to be included or excluded, respectively.

At this stage, you could either continue selecting/de-selecting property names manually, you could edit the property names text files on your local disk for the Always Include list or Never Include list for Selected measurements/Globals, or begin uploading the results.

Refresh If you modify the property names text files on your local disk for the Always Include list or Never Include list, the **Properties** tab must reflect the changes.

For example, if you modify the text files, that is, the app-defined property names from **Selected measurements** "*RangeProperty*" and "*MaxProperty*" are added to the Always Include list whereas "*SingleProperty*" is added to the Never Include list, the **Properties** tab must detect these changes that you make in the **Property Names** text file.



One way is to repeat the steps described in the previous section to load the updated text files.

To avoid performing these steps every time you modify the **Property Names** text files, simply click the **Refresh** button in the **App-Defined** section under the **Properties** tab. Notice the changes in the **App-Defined** section for the selected Dataset.

App-Defined

Show properties from: Selected measurements (Additional Info values from Results tab)
 Globals (Settings from Set Up, Configure and HTML Report tabs)

Auto Mode: [Meas] Always (3) + Never (2)

Auto...	<input checked="" type="checkbox"/>	Name	Value	Type	Reported With	Name In Repository
Refresh	<input checked="" type="checkbox"/>	All_yProperty	All Text1	Text	<Multiple measurements> (<Multiple runs>)	False
	<input checked="" type="checkbox"/>	MaxProperty	1.234	Text	<Multiple measurements> (<Multiple runs>)	False
	<input type="checkbox"/>	MinProperty	1	Text	<Multiple measurements> (<Multiple runs>)	False
	<input checked="" type="checkbox"/>	RangeProperty	Range Text1	Text	<Multiple measurements> (<Multiple runs>)	False
	<input type="checkbox"/>	SingleProperty	Single Text1	Text	<Multiple measurements> (<Multiple runs>)	False

App-Defined

Show properties from: Selected measurements (Additional Info values from Results tab)
 Globals (Settings from Set Up, Configure and HTML Report tabs)

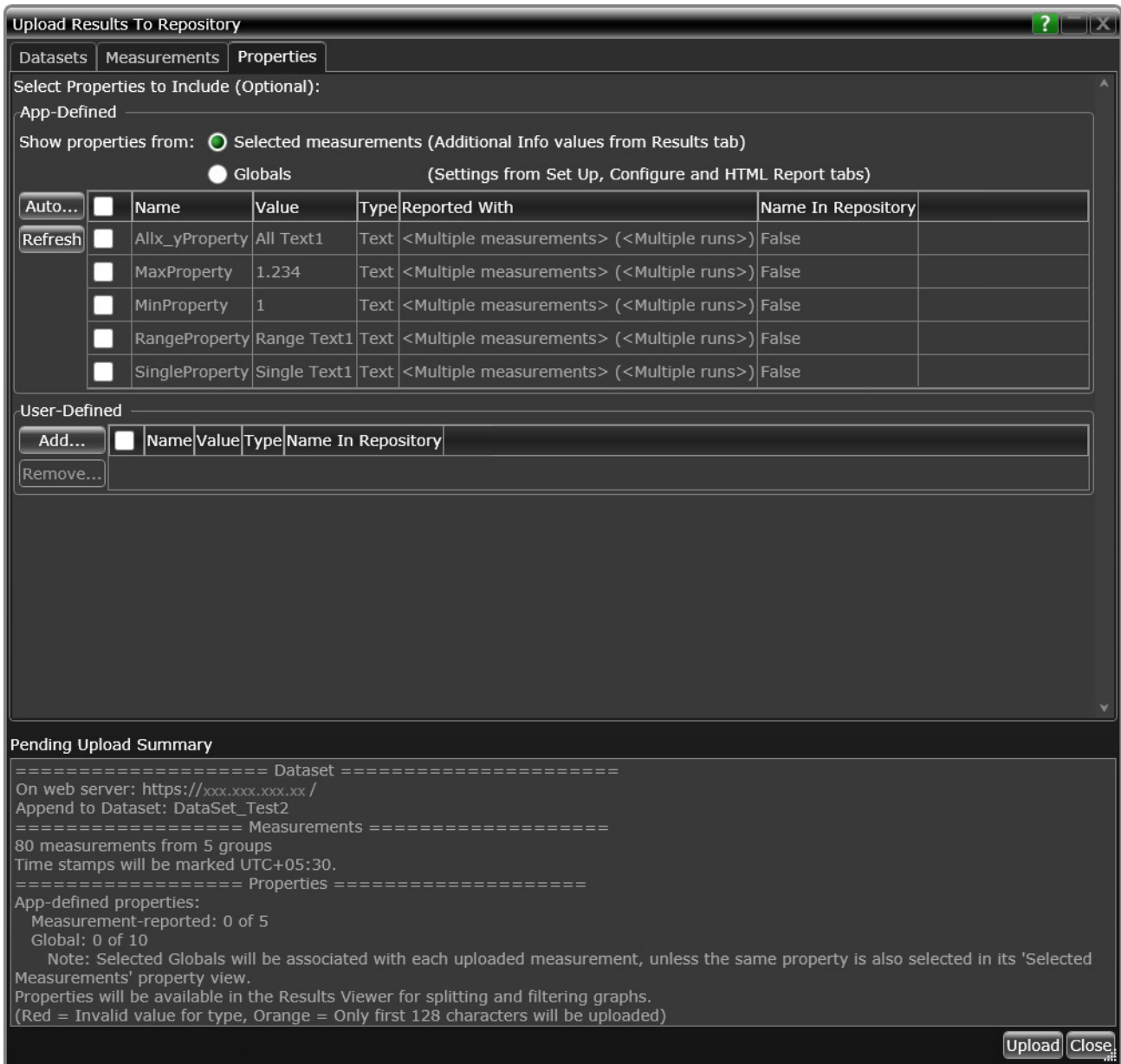
Auto Mode: [Meas] Always (3) + Never (2) + [Global] Always (2) + Never (2)

Auto...	<input type="checkbox"/>	Name	Value	Type	Location	Name In Repository
Refresh	<input type="checkbox"/>	Allx_yProperty	Text Value 1	Text	Configure tab	False
	<input checked="" type="checkbox"/>	Debug Mode Used	No	Text	HTML Report tab	False
	<input type="checkbox"/>	Infiniium Model Number	N8900A	Text	HTML Report tab	False
	<input type="checkbox"/>	Infiniium Serial Number	No Serial	Text	HTML Report tab	False
	<input type="checkbox"/>	Infiniium SW Version	05.75.00008	Version	HTML Report tab	False
	<input type="checkbox"/>	MaxProperty	Numeric Value 1	Text	Configure tab	False
	<input type="checkbox"/>	MinProperty	Numeric Value 1	Text	Configure tab	False
	<input type="checkbox"/>	RangeProperty	Text Value 1	Text	Configure tab	False
	<input type="checkbox"/>	SingleProperty	Text Value 1	Text	Configure tab	False
	<input checked="" type="checkbox"/>	UserComments	<Unknown>	Text	Set Up tab	False

Once the **App-Defined** properties are selected, you may proceed with uploading the measurement results (see [page 344](#)). If you wish to define custom property names prior to uploading measurement results, refer to the next section.

User-Defined Properties

The custom-defined properties for one or more measurement types, which you have selected under the **Measurements** tab, are listed under the **User-Defined** properties. There are no default user-defined properties within the Test Application.



Features in the User-Defined section

The **User-Defined** section consists of a table to list out all custom property names and their associated attributes, if they have already been defined, else by default, this table remains blank.

- **Name** – The property names that you assign to the selected Dataset or that have already been uploaded to the web repository, are displayed. By default, this column is blank.
- **Value** – An alphanumeric value, if defined, is displayed for each property name.

- **Type** – Displays the type of property selected when adding a new user-defined property, such that an appropriate value can be assigned to the property. The options are:
 - **DateTime** – assign date and time to the property, in *YYYY-MM-DD hh:mm:ss::nnn* format.
 - **Decimal** – assign a decimal value to the property.
 - **Integer** – assign an integer value to the property.
 - **Text** – assign an alphanumeric text to the property.
 - **Version** – assign a version number to the property.
 - **Boolean** – assign a boolean value, such as *TRUE / FALSE* or *YES / NO*, to the property.
- **Name In Repository** – Displays either **TRUE** or **FALSE** for each row of properties.
 - **TRUE** indicates that the property name has already been associated with the selected Dataset and has been uploaded to the web repository.
 - **FALSE** indicates that the property name has not yet been associated with the selected Dataset on the web repository.

To start adding custom property names, click the **Add...** button.

The **User-Defined** section displays three options for you to add or select a user-defined property name from.

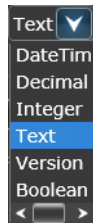
- **"New"** on page 372
- **"From File"** on page 373
- **"From Repository"** on page 377

The screenshot shows the 'User-Defined' dialog box with the following elements:

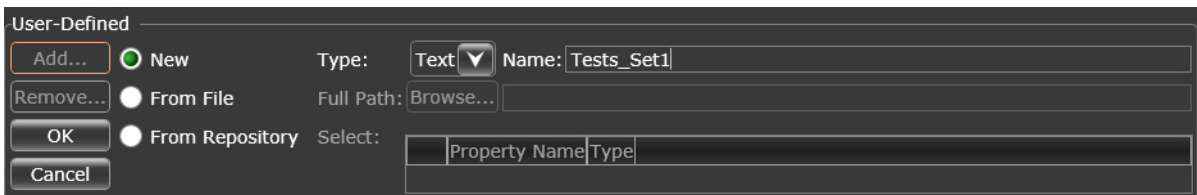
- Buttons:** Add..., Remove..., OK, Cancel.
- Radio Buttons:**
 - New
 - From File
 - From Repository
- Fields:**
 - Type:** Text (dropdown menu)
 - Name:** [Empty text field]
 - Full Path:** Browse... [Empty text field]
 - Select:** [Table with columns: Property Name, Type]

New This option lets you create a new property name of your choice. By default, this option is selected.

- 1 From the drop-down options for **Type:**, select an option to define the type of property. By default, **Text** is selected.



- 2 Enter an appropriate alphanumeric value in the **Name:** text field, which represents the property name.



- 3 Click the **OK** button. The property name is created and added to the tabular view. Notice that the **Name In Repository** column displays *False*, which indicates that this user-defined property name has not been uploaded yet to the Web Repository.

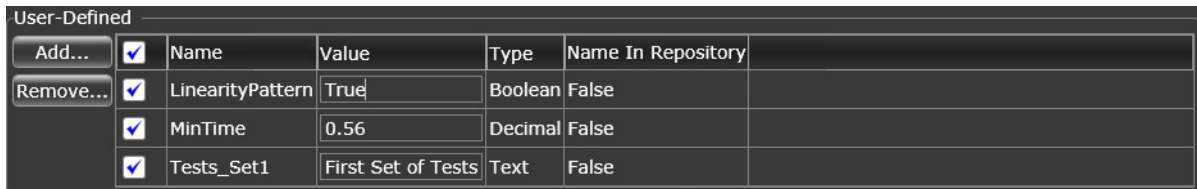


- 4 Define a value to the property name. Click within the text box in the **Value** column and type an alphanumeric text.

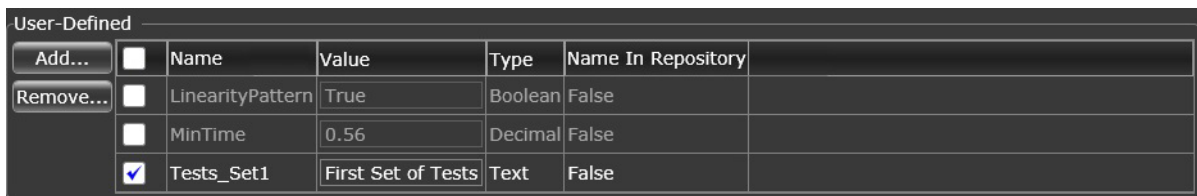


- 5 At this point, you may perform one of the following actions:
 - Click the **Upload** button if you wish to begin uploading the measurement results (see [page 344](#)) to the Web Repository.

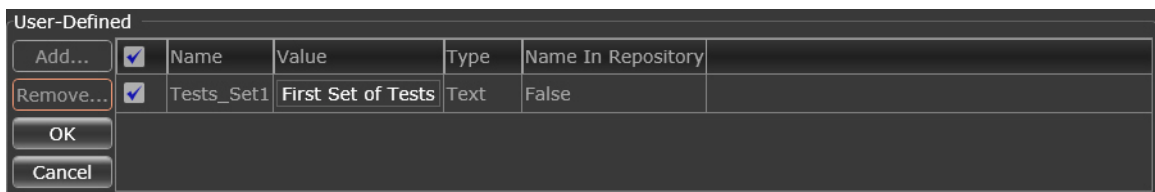
- Click the **Add...** button if you wish to create more user-defined property names. Repeat steps 1 to 3 in this section to add more properties and assign values to them.



- Click the **Remove...** button, if you wish to remove one or more custom property names. By default, all user-defined property names are selected.
 - Remove the tick from the check-box for the property name you wish to retain.



- Click the **OK** button.



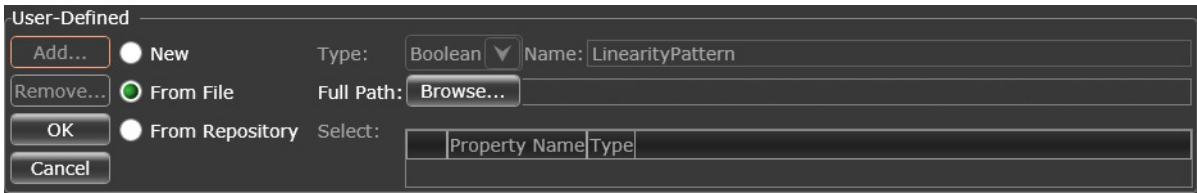
The tabular view with the remaining property names appears.

- Click the **Add...** button again if you wish to use the other options in the **User-Defined** section.

From File This option to define a new property name of your choice is similar to using the Auto... (see [page 363](#)) feature in the **App-Defined** section.

- Select **From File** to define property names saved on your local disk.

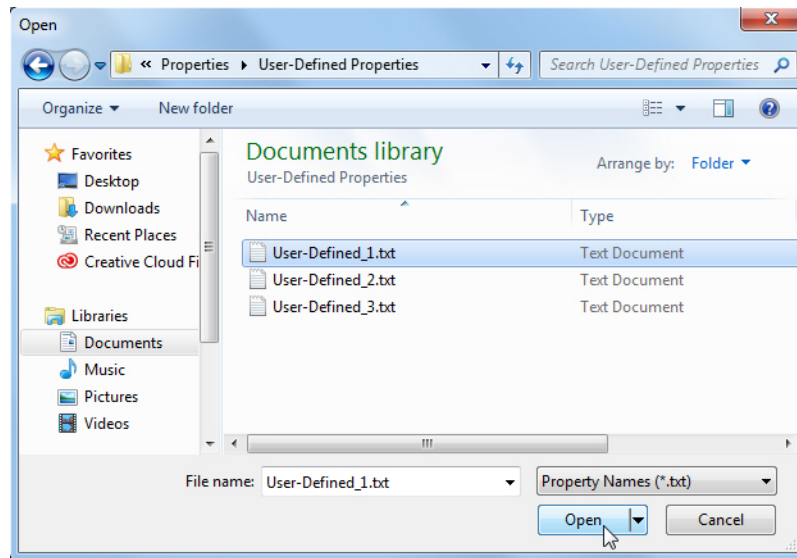
The **Full Path:** text field and the **Browse...** button are enabled.



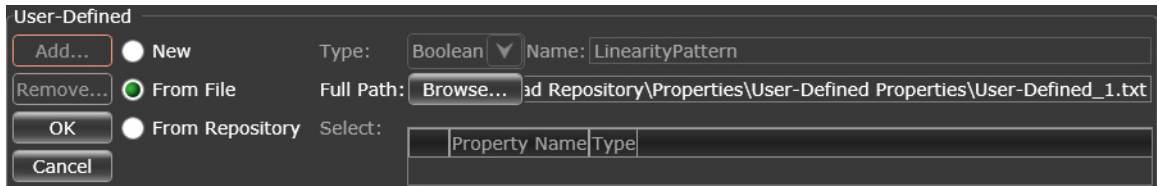
- 2 In the **Full Path:** text field, either manually type the entire file location for the **Property Names** text file (in the *.txt format) or click the **Browse...** button to navigate to the folder where the **Property Names** text file is located.

To create a **Property Names** text file, open the Notepad editor, type the name of each property you wish to include or exclude and save the file on your local disk. Each property name must begin on a new line.

- 3 In the **Open** dialog box, select the desired file and click **Open**.



The **Full Path:** text field is populated with the file location for the property names text file.



- 4 Click the **OK** button to save the changes.

The property name is created and added to the tabular view. Notice that the property type is set to text, by default and the **Name In Repository** column displays *False*, which indicates that this user-defined property name has not been uploaded yet to the Web Repository.

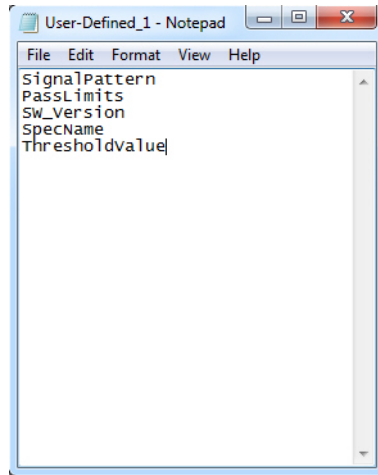
User-Defined					
Add...	<input checked="" type="checkbox"/>	Name	Value	Type	Name In Repository
Remove...	<input checked="" type="checkbox"/>	LinearityPattern	True	Boolean	False
	<input checked="" type="checkbox"/>	MinTime	0.56	Decimal	False
	<input checked="" type="checkbox"/>	PassLimits		Text	False
	<input checked="" type="checkbox"/>	SignalPattern		Text	False
	<input checked="" type="checkbox"/>	SW_Version		Text	False

- 5 Define a value to the property name. Type an alphanumeric in each text box of the **Value** column.

User-Defined					
Add...	<input checked="" type="checkbox"/>	Name	Value	Type	Name In Repository
Remove...	<input checked="" type="checkbox"/>	LinearityPattern	True	Boolean	False
	<input checked="" type="checkbox"/>	MinTime	0.56	Decimal	False
	<input checked="" type="checkbox"/>	PassLimits	10ps	Text	False
	<input checked="" type="checkbox"/>	SignalPattern	PRBS	Text	False
	<input checked="" type="checkbox"/>	SW_Version	1.03.0012	Text	False

- 6 At this point, you may perform one of the following actions:
 - Click the **Upload** button if you wish to begin uploading the measurement results (see [page 344](#)) to the Web Repository.
 - Click the **Remove...** button if you wish to remove one or more custom property names. By default, all user-defined property names are selected. This feature works in the same manner as described in the previous section.
 - Click the **Add...** button if you wish to add more user-defined property names from the property names text file. Repeat steps 1 to 5 in this section to add more properties.

If you modify the property names text file to add more user-defined property names, you must add the modified file to the **From File** option:



The tabular view displays all property names defined in the property names text file and arranges the property names in alphanumeric order.

User-Defined					
Add...	<input checked="" type="checkbox"/>	Name	Value	Type	Name In Repository
Remove...	<input checked="" type="checkbox"/>	LinearityPattern	True	Boolean	False
	<input checked="" type="checkbox"/>	MinTime	0.56	Decimal	False
	<input checked="" type="checkbox"/>	PassLimits	10ps	Text	False
	<input checked="" type="checkbox"/>	SignalPattern	PRBS	Text	False
	<input checked="" type="checkbox"/>	SpecName		Text	False
	<input checked="" type="checkbox"/>	SW_Version	1.03.0012	Text	False
	<input checked="" type="checkbox"/>	ThresholdValue		Text	False

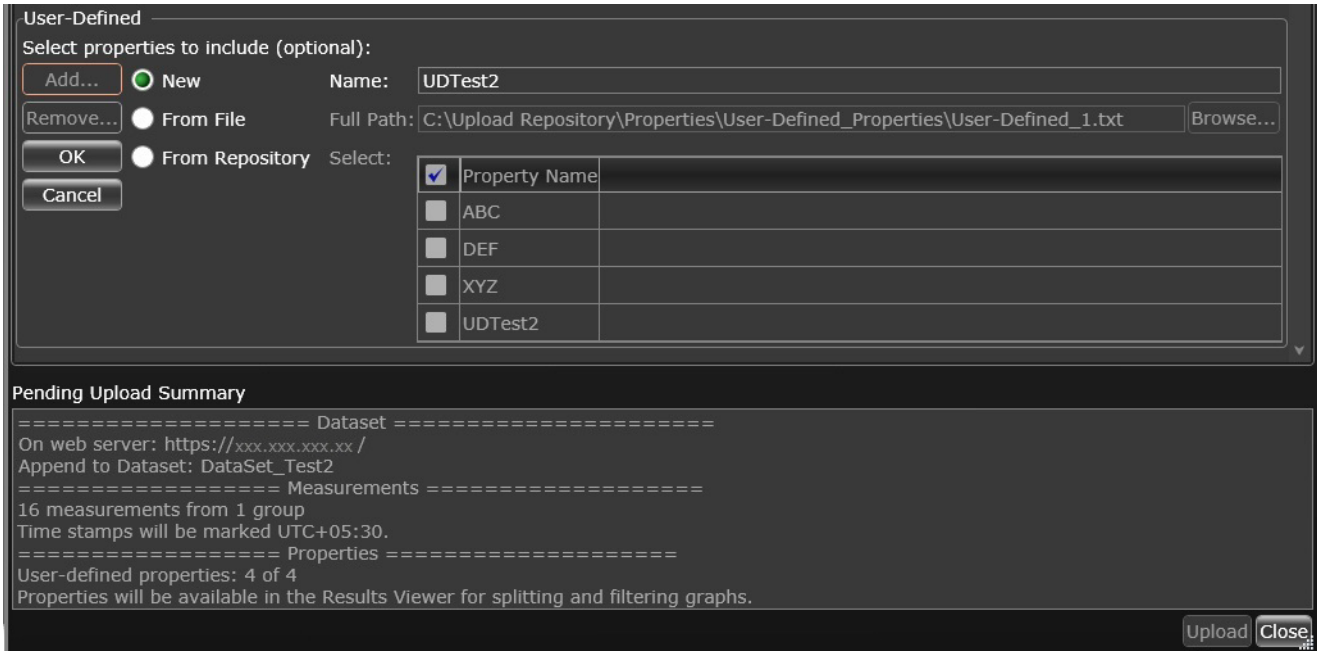
Define values to the new property names that have been added before you begin uploading the measurement results.

User-Defined					
Add...	<input checked="" type="checkbox"/>	Name	Value	Type	Name In Repository
Remove...	<input checked="" type="checkbox"/>	LinearityPattern	True	Boolean	False
	<input checked="" type="checkbox"/>	MinTime	0.56	Decimal	False
	<input checked="" type="checkbox"/>	PassLimits	10ps	Text	False
	<input checked="" type="checkbox"/>	SignalPattern	PRBS	Text	False
	<input checked="" type="checkbox"/>	SpecName	ProtocolStandards1.0	Text	False
	<input checked="" type="checkbox"/>	SW_Version	1.03.0012	Text	False
	<input checked="" type="checkbox"/>	ThresholdValue	20mV	Text	False

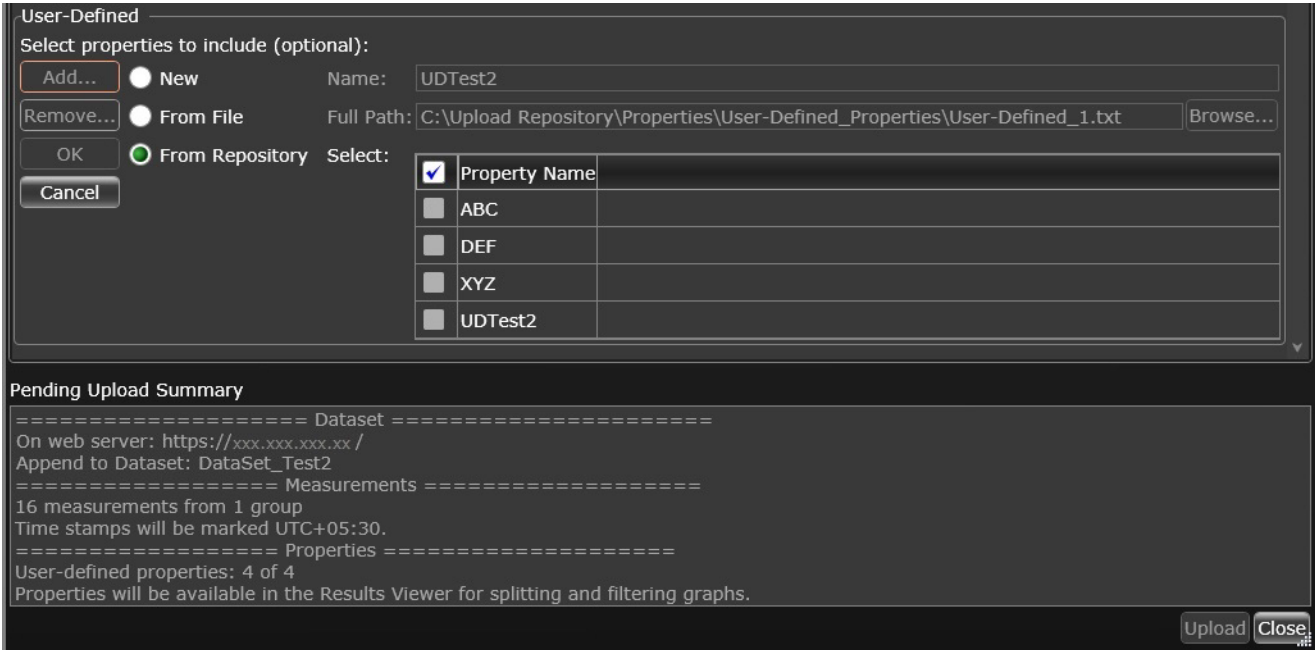
Click the **Add...** button again if you wish to use the other options in the **User-Defined** section.

From Repository This option lets you select one or more property names that have been already defined for the selected Dataset and uploaded to the Web Repository.

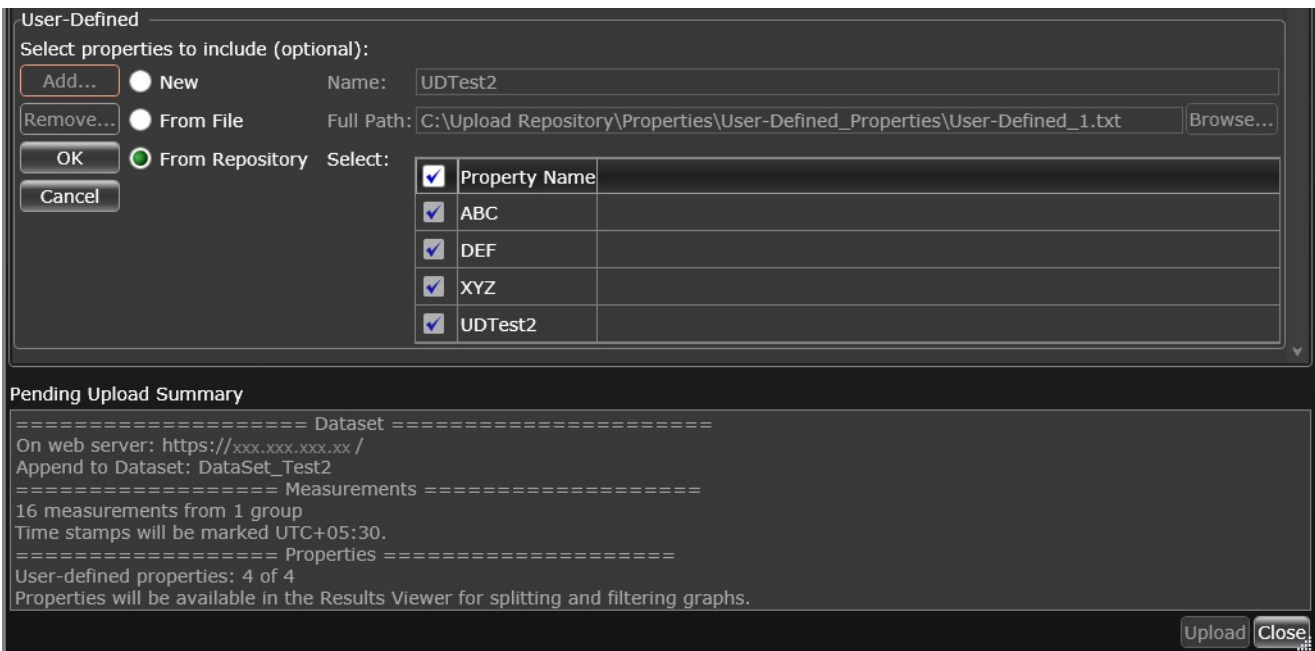
If you have already uploaded some properties using the **New** and the **From File** options for a specific Dataset, the User-Defined section for the same Dataset appears as shown below:



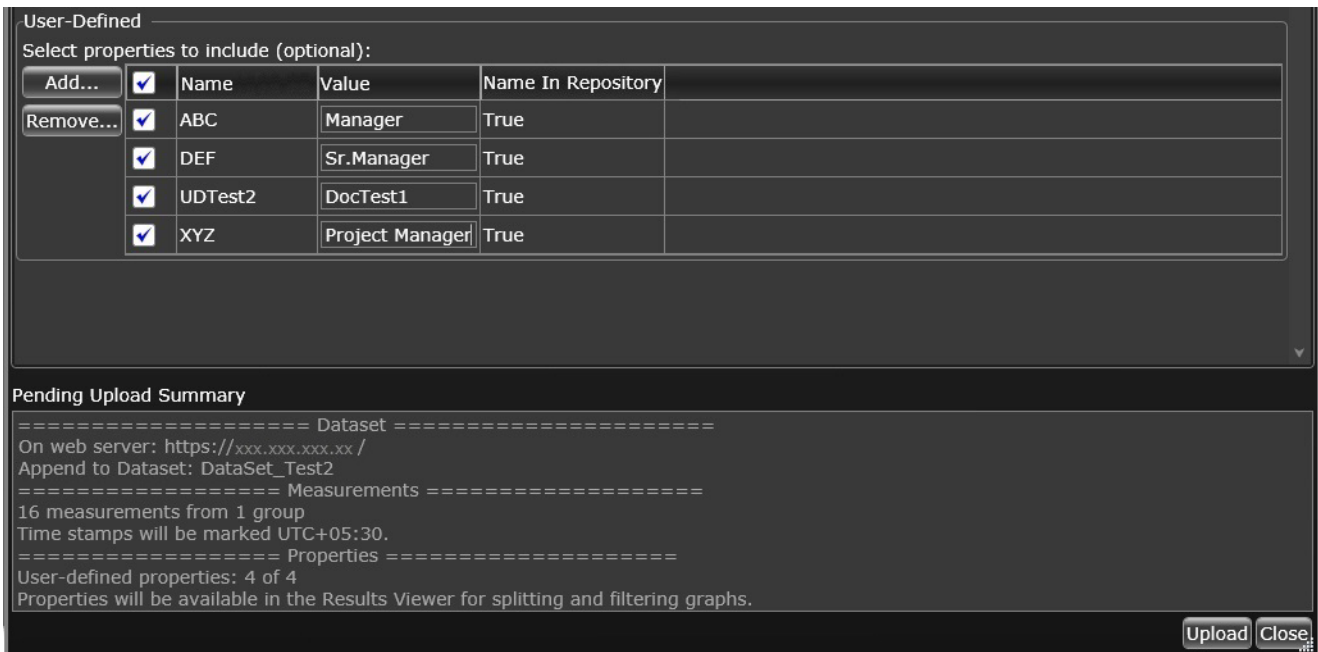
- 1 Click the **From Repository** radio-button to select a property name that may have been uploaded earlier.



- 2 By default, all property names are grayed out. De-select and select again the check-box at the top to select the check-boxes for the individual property names.



- 3 Click the **OK** button to modify the values for each property name that you want to reuse.
- 4 Define a value to the property name. Click within the text box in the **Value** column and type an alphanumeric text.



- 5 At this point, you may perform one of the following actions:
 - Click the **Upload** button if you wish to begin uploading the measurement results (see [page 344](#)) to the Web Repository.
 - Click the **Remove...** button if you wish to remove one or more custom property names. By default, all user-defined property names are selected. This feature works exactly as described in the previous section.
 - Click the **Add...** button if you wish to add more user-defined property names using any of the options described earlier.

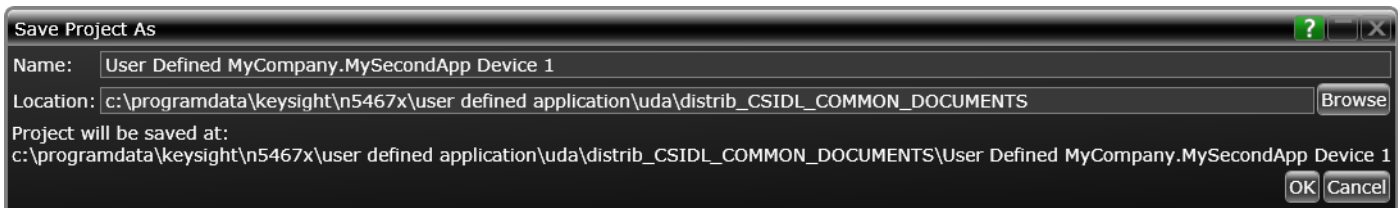
Saving Test Projects

To save test settings and results to the current project directory:

- 1 Choose **File > Save Project** from the menu.

To save test settings and results to a new project directory:

- 1 Choose **File > Save Project As...** from the menu.

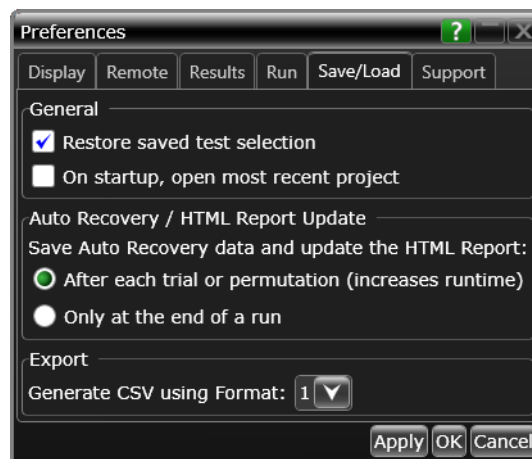


- 2 In the Save Project As... dialog box, enter the device name and location. Project files will be saved in a directory whose name is the device name.
- 3 Click **OK**.

See Also · ["To set AutoRecovery preferences"](#) on page 380

To set AutoRecovery preferences

- 1 From the generated application's main menu, choose **View > Preferences...**
- 2 In the Preferences dialog box, select the **Save/Load** tab.



- 3 In the **AutoRecovery** area, you can choose:
 - To auto-save results after each trial or permutation even if the entire multi-trial is not completed. This option enables full recovery.

- To auto-save results only upon the completion of the entire multi-trial.
- 4 Click **Apply** to save the changes and click **OK** to close the Preferences dialog box.

Executing Scripts

You can integrate arbitrary scripts you have written (in Visual Basic, Python, etc.) into your app's workflow.

- 1 Install the program needed to execute your script.
- 2 Place the script inside the automated test app's installation directory:
(Windows XP) – C:\Documents and Settings\All Users\Application Data\Keysight\Infiniium\FlexDCA\Apps\<(app name)\app\scripts
(Windows 7) – C:\ProgramData\Keysight\Infiniium\FlexDCA\Apps\<(app name)\app\scripts
- 3 When you create the "scripts" folder (above), the application will create a new **File > Execute Scripts** menu item to enable you to select and execute files from that directory.

You will also be able to execute scripts in that folder using the **Automation** tab or the app's remote interface.

Controlling the Application via a Remote PC

The generated application's Preference dialog box has a **Remote** tab for enabling the remote interface and setting remote options.

The N5452A remote interface lets you control Infiniium applications from a remote PC. It comes with ready to run executables, but it also lets you create custom programs using a .NET 2.0 programming language or the National Instruments' LabVIEW 8.5 graphical programming environment.

With the remote interface, you can:

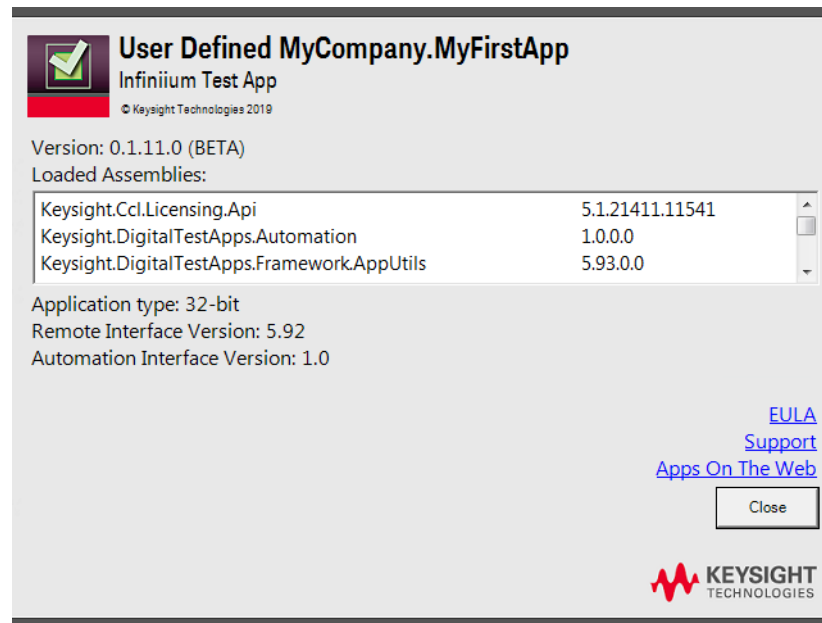
- Launch and close applications.
- Configure options.
- Run tests.
- Obtain results.
- Control when and where dialog boxes are displayed.
- Save and load projects.

For more information on the remote interface, see the [N5452A Remote Interface for DigitalTest Applications](#) on the Keysight web site.

- See Also**
- ["To identify the remote interface version"](#) on page 383
 - ["To enable the remote interface"](#) on page 384
 - ["To enable remote interface hints"](#) on page 385

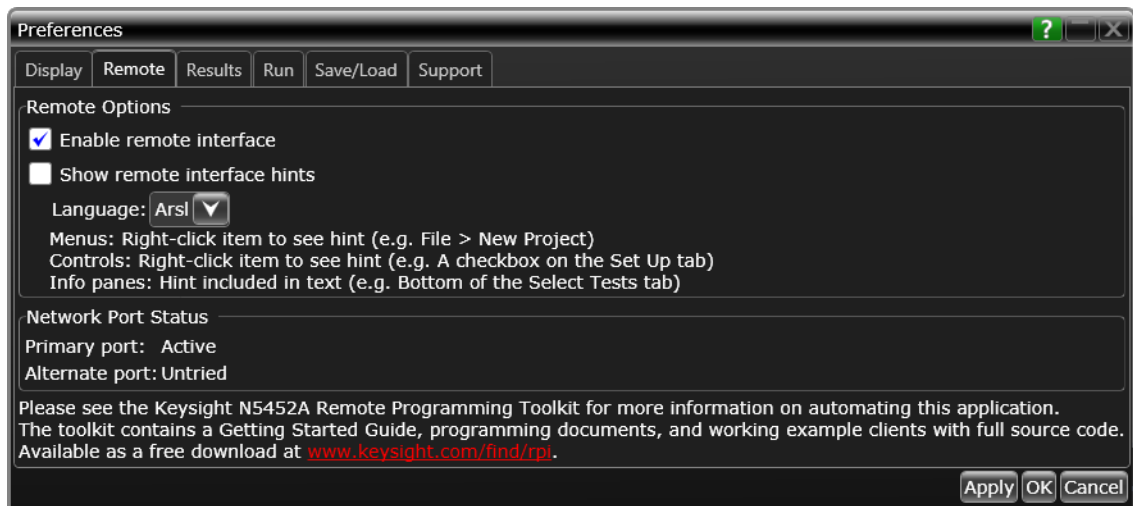
To identify the remote interface version

- 1 From the generated application's main menu, choose **Help > About...**
- 2 In the About dialog box, the remote interface version is listed.



To enable the remote interface

- 1 From the generated application's main menu, choose **View > Preferences....**
- 2 In the Preferences dialog box, select the **Remote** tab.



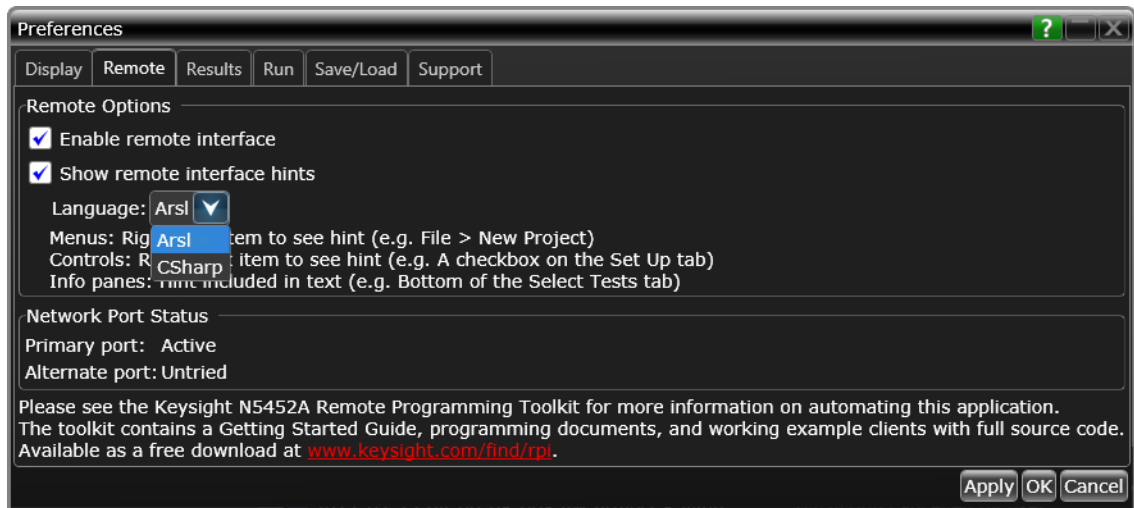
- 3 Check the **Enable remote interface** option if you need to access the application remotely.

If you are performing the tests with the application's user interface and want to ensure no remote users accidentally interfere with you, disable the remote interface by un-checking this option.

- 4 Click **Apply** to save the changes and click **OK** to close the Preferences dialog box.

To enable remote interface hints

- 1 From the generated application's main menu, choose **View > Preferences...**
- 2 In the Preferences dialog box, select the **Remote** tab.



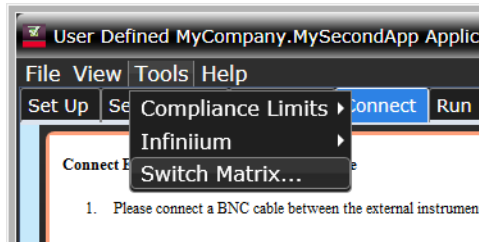
- 3 In the remote options area, check **Show remote interface hints**.

When this option is checked:

- You can select the remote programming language described in the tips.
 - Tooltips related to the remote interface commands appear when you click the toolbar.
 - Various controls in the tabs will have a context menu item added as "Remote interface hint...".
 - The **Select Tests** and **Configure** tabs will display a remote hint in their description panes at the bottom of the screen, when an item is selected.
- 4 Click **Apply** to save the changes and click **OK** to close the Preferences dialog box.

Switch Matrix Automation

Some automated test applications support a switch matrix option to enable you to use RF switches to automatically route your test points to the oscilloscope and thus avoid having to change physical connections during a run. When an application supports this option, this menu button will be enabled:



which displays the switch matrix user interface:



The **On/Off** buttons are a master switch for the entire feature. In some applications, selecting certain settings on the Configure tab will cause your switch matrix settings to be invalidated. When that occurs, you will be prompted and the master switch will be turned off. At that time, you can return to this user interface, turn the feature back on and review the tabs for necessary changes.

The switch matrix user interface has three tabs and buttons at the bottom:

- **"Controller Tab"** on page 386
- **"Signal Paths Tab"** on page 389
- **"Response Correction Tab"** on page 396
- **"Buttons"** on page 397

Controller Tab

- **"Configuration Mode"** on page 387
- **"Switch Drivers"** on page 387
- **"Signal Path Options"** on page 388
- **"Connection Options"** on page 389

Configuration Mode

Some applications support an automatic signal path routing mode. In these applications, the Configuration Mode group will be displayed, and you will be able to choose between:

Automatic mode – This mode automatically selects the required number of switch drivers and signal path routing. Typically, this option limits the switch instrument models you can select.

Manual mode – This mode enables you to manually select any number of switch drivers and to manually specify all signal path routing. This option enables you to select any of the supported switch instruments models.

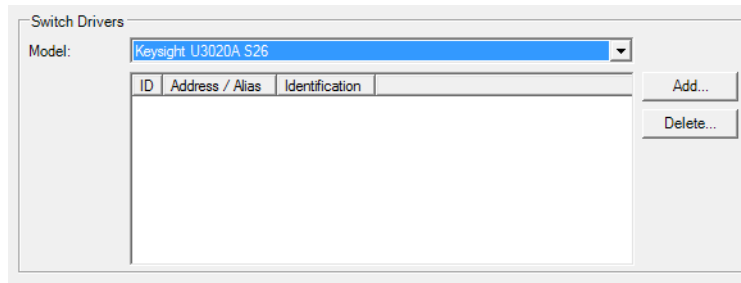
Switch Drivers

Select the model of switch instrument you are using. What appears next depends upon the active configuration mode:

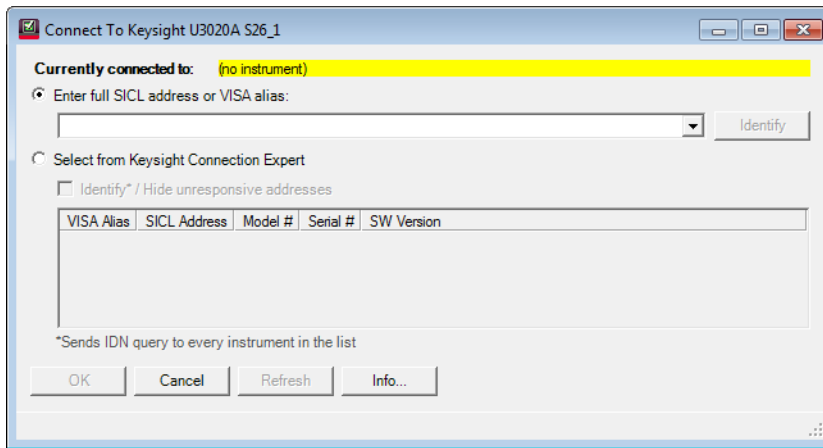
- **Automatic mode** – One or more switch drivers will be added to the list and a **Connect** button will display:

ID	Address / Alias	Identification
1	<none>	<not connected>

- **Manual mode** – **Add.../Delete...** buttons enable you to add as many switch drivers as you need. Once you have added a driver, the **Connect** button becomes available:

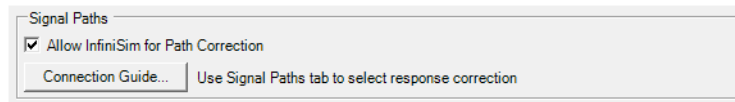


When you click the **Connect** button, you will be prompted to enter/select the address of the switch instrument currently selected in the list:



You must connect to each instrument driver in the list to be able to perform tasks required in the other two tabs.

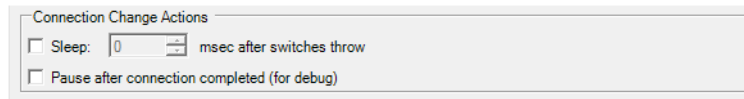
Signal Path Options



Allow InfiniiSim – If the application is not already using InfiniiSim for test-specific requirements, this option will appear and will be selected. When this setting is checked, InfiniiSim will be offered as an option in the Response Correction tab.

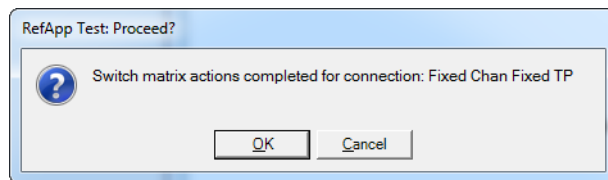
Connection Guide... – When using Automatic mode, some applications will display this button for accessing help diagrams for connecting your switch instrument to the oscilloscope and test points.

Connection Options



Sleep – If your signals require settling time after switches throw, check this option and select the desired number of milliseconds for the application to wait every time it throws switches.

Pause – To manually check signal routing, check this option to cause a prompt to appear every time the application throws switches:



At this point, you can manually verify the signal is routed as expected. Click **OK** to proceed with the test or **Cancel** to abort the test.

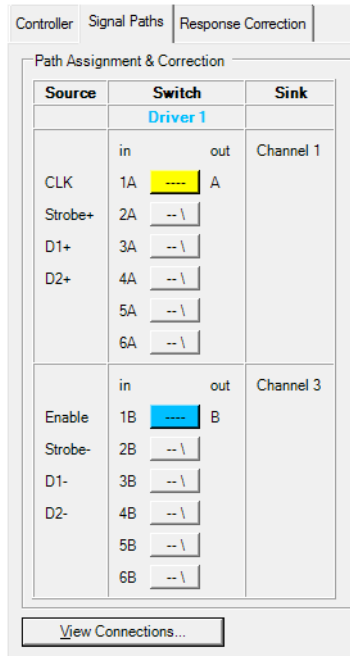
Signal Paths Tab

The Signal Paths tab shows how your test points will be routed to the oscilloscope channels. What appears depends upon the active configuration mode:

- "Automatic mode" on page 389
- "Manual mode" on page 392

Automatic mode

In automatic mode, the routing will already be assigned and will be unchangeable:

**NOTE**

The signal path examples shown here are for a fictional application and are for demonstration purposes only.

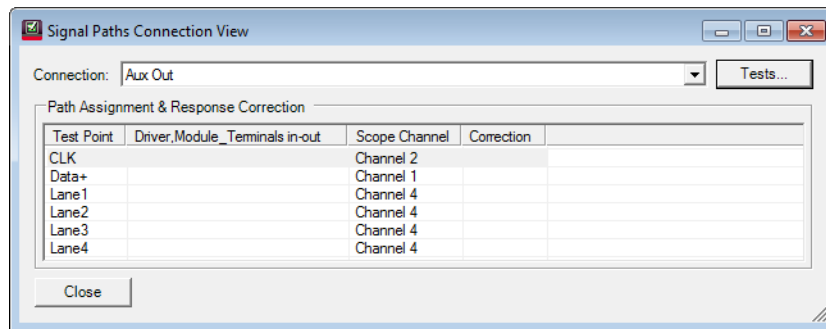
Source – This column represents your test points.

Switch – This column contains labels and buttons representing each switch position of each switch module in the driver. Each button is live and will cause the switch to throw to that position. The currently selected position will be highlighted with the associated oscilloscope channel's color.

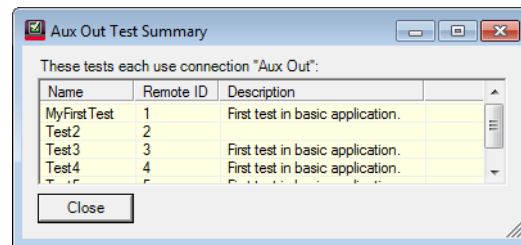
Sink – This column represents the oscilloscope channels.

Each row represents a single signal path, from test point to oscilloscope channel.

View Connections... – This button enables you to view switch connections from the point of view of the connection prompts that are being automated by the switch matrix module:



- **Connection** – This combination box contains a list of all of the connection prompts that can be automated for this application. When you select one, the list below it shows you which test points will be routed to which oscilloscope channels for that connection.
- **Tests...** – This button displays a list of tests that use the currently-selected connection:



In the above example, each signal path contained only one switch module. Some applications also support multi-switch routing, as shown below:

Path Assignment & Correction			
Source	Switch		Sink
	Driver 1		
	in	Module 1 out	Channel 1
<-> Driver 1 Module 5 Switch 1	1	----- Common	
Strobe+	2	-- \	
D1+	3	-- \	
D2+	4	-- \	
	in	Module 2 out	
	1	----- Common	
	2	-- \	
	3	-- \	
	4	-- \	
Enable	in	Module 3 out	Channel 3
Strobe-	1	----- Common	
D1-	2	-- \	
D2-	3	-- \	
	4	-- \	
<-> Driver 1 Module 5 Switch 1	in	Module 4 out	Channel 4
	1	----- Common	
	2	-- \	
	3	-- \	
	4	-- \	
CLK	in	Module 5 out	
	Common	----- 1	<-> Driver 1 Module 1
		-- \ 2	<-> Driver 1 Module 4
	in	(2) out	
	1	----- Common	
	2	-- \	

In the above example, switch-to-switch links are indicated by the "<->" symbol. This example shows that switches can be connected in an N:1 orientation (as with modules 1–4) or in a 1:N orientation (as with module 5). The example also shows that while some modules contain only one switch (as with modules 1–4), others can contain multiple switches (as with module 5, a 2xSPDT).

Manual mode

In manual mode, none of the routing will be assigned and all of it will be configurable:

Source	Switch	Sink
	Driver 1	
	in out	
	1A ---- A	
	2A -- \	
	3A -- \	
	4A -- \	
	5A -- \	
	6A -- \	
	in out	
	1B ---- B	
	2B -- \	
	3B -- \	
	4B -- \	
	5B -- \	
	6B -- \	

Start by assigning the oscilloscope channels to the module outputs. This must be done first because each application is designed to expect certain test points' signals to appear at the expected oscilloscope channels at run time.

Source	Switch	Sink
	Driver 1	
	in out	
	1A ---- A	Channel 1
	2A -- \	
	3A -- \	
	4A -- \	
	5A -- \	
	6A -- \	
	in out	
	1B ---- B	Channel 3
	2B -- \	
	3B -- \	
	4B -- \	
	5B -- \	
	6B -- \	

Now you can assign the test points to whichever switch inputs you choose:

Source	Switch	Sink
	Driver 1	
	in	out Channel 1
	1A ---- A	
	2A -- \	
	3A -- \	
	4A -- \	

When multi-switch routing is required, again assign the oscilloscope channels first, then assign output links in the **Sink** column:

Source	Switch	Sink
	Driver 1	
	in	out Channel 1
	1 ---- Common	
	2 -- \	
	3 -- \	
	4 -- \	
	in	out
	1 ---- Common	
	2 -- \	
	3 -- \	
	4 -- \	
	in	out Channel 3
	1 ---- Common	
	2 -- \	
	3 -- \	
	4 -- \	
	in	out Channel 4
	1 ---- Common	
	2 -- \	
	3 -- \	
	4 -- \	
	in	out
	Common ---- 1	<-> Driver 1 Module 1
	-- \ 2	<-> Driver 1 Module 4
	in (2) out	
	1 ---- Common	
	2 -- \	

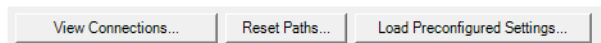
Next assign the input links in the **Source** column:

Path Assignment & Correction		
Source	Switch	Sink
	Driver 1	
	in Module 1 out	Channel 1
<-> Driver 1 Module 5 Switch 1	1 ---- Common	
	2 -- \	
	3 -- \	
	4 -- \	
	in Module 2 out	
	1 ---- Common	
	2 -- \	
	3 -- \	
	4 -- \	
	in Module 3 out	Channel 3
	1 ---- Common	
	2 -- \	
	3 -- \	
	4 -- \	
<-> Driver 1 Module 5 Switch 1	in Module 4 out	Channel 4
	1 ---- Common	
	2 -- \	
	3 -- \	
	4 -- \	
	in Module 5 out	<-> Driver 1 Module 1
	Common ---- 1	<-> Driver 1 Module 4
	-- \ 2	
	in (2) out	
	1 ---- Common	
	2 -- \	

Finally, assign the test points in the **Source** column.

Path Assignment & Correction		
Source	Switch	Sink
	Driver 1	
	in Module 1 out	Channel 1
<-> Driver 1 Module 5 Switch 1	1 ---- Common	
Strobe+	2 -- \	
D1+	3 -- \	
D2+	4 -- \	
	in Module 2 out	
	1 ---- Common	
	2 -- \	
	3 -- \	
	4 -- \	
	in Module 3 out	Channel 3
Enable	1 ---- Common	
Strobe-	2 -- \	
D1-	3 -- \	
D2-	4 -- \	
	in Module 4 out	Channel 4
<-> Driver 1 Module 5 Switch 1	1 ---- Common	
	2 -- \	
	3 -- \	
	4 -- \	
CLK	in Module 5 out	<-> Driver 1 Module 1
	Common ---- 1	<-> Driver 1 Module 4
	-- \ 2	
	in (2) out	
	1 ---- Common	
	2 -- \	

In Manual mode, you can use the **Reset Paths...** button to clear out all settings on this tab. If the application also supports Automatic mode, the application will display a **Load Preconfigured Settings...** button to enable you to manually apply the signal path routing used by automatic mode.



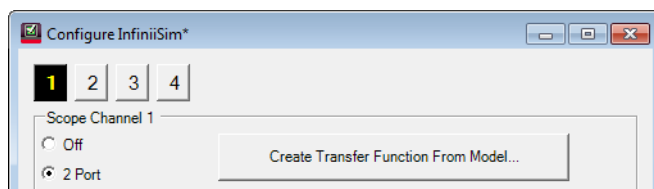
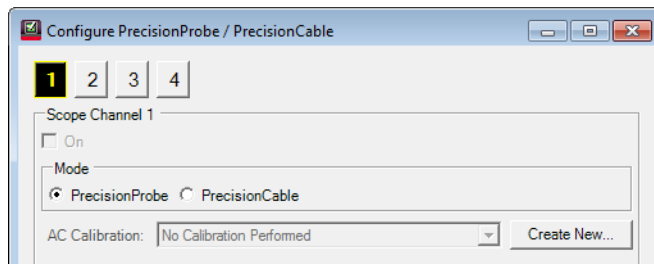
Response Correction Tab

In this tab, you can assign response correction for each signal path:

Test Point	Driver,Module,Terminals (in-out)	Scope	Response Correction
CLK	1.1.1-0	Channel 1	
D1-	1.2.3-0	Channel 3	
D1+	1.1.3-0	Channel 1	
D2-	1.2.4-0	Channel 3	
D2+	1.1.4-0	Channel 1	
Enable	1.2.1-0	Channel 3	
Strobe-	1.2.2-0	Channel 3	
Strobe+	1.1.2-0	Channel 1	

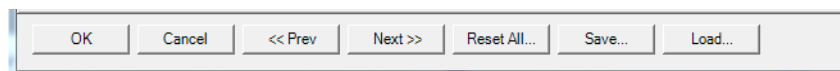
Depending on the application, you may be able to choose between Precision Probe/Precision Cable and InfiniiSim technologies for each path.

When you select a row and then select the desired path correction mode, you will be prompted with the standard Precision Probe/Precision Cable or InfiniiSim configuration screens, which are described elsewhere in this document. In those screens, either select the calibration or model you have previously created for each path, or click the **Create** button to be guided to create one:



Buttons

At the bottom of the switch matrix user interface buttons for the remaining functions:



<<Prev, Next>> – Navigation between tabs.

Reset All... – Resets all settings on all tabs.

Save... – Save all settings on all tabs to a settings file.

Load... – Restore all tabs from a settings file.

NOTE

When you save/load an entire project (**File > Save/Open**), switch matrix settings are saved/loaded with it. The save/load feature in the switch matrix user interface enables you to modify switch matrix settings without affecting the rest of the application.

19 Solving Problems

If there are problems connecting to instruments / 400

If there are problems with SCPI commands / 401

To see variable values as tests run / 407

If there are problems uninstalling a generated app / 410

Build Errors / 411

Restoring the Project / 415

If there are problems connecting to instruments

If you are unable to connect to your scope or external instrument, try any of the following:

- Ensure you are entering the full SICL address, not the VISA address. UDA does not support VISA addresses.
- Try using a VISA alias for your instrument. For details, see Keysight Connection Expert.
- Test connectivity using Keysight Connection Expert's "Send commands to this instrument" feature (send SCPI command :AUToscale). See "**Interactively Sending SCPI Commands to Instruments**" on page 404.
- Restart the instrument.
- Corrupt VISA COM installations is a common problem (VISA COM is installed with Keysight IO Libraries). The IVI Foundation has a VISA Shared Components cleanup and installation tool found here: http://www.ivifoundation.org/shared_components/Default.aspx
- Upgrade to the latest version of Keysight IO Libraries. If you are already using the latest, re-install it using the "Repair" option.

If there are problems with SCPI commands

Test the commands using the Keysight IO Libraries Suite. If the commands work via the Connection Expert and Interactive IO utilities, they should work from within the generated application.

Before you can test SCPI commands with the Interactive IO utility, you must add the instrument using the Connection Expert. See

- ["Adding Instruments Using Keysight Connection Expert"](#) on page 401
- ["Interactively Sending SCPI Commands to Instruments"](#) on page 404

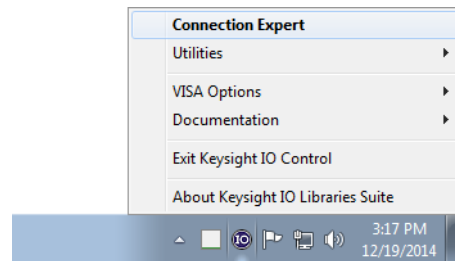
Adding Instruments Using Keysight Connection Expert

This procedure only needs to be performed once for each Infiniium oscilloscope or external instrument.

NOTE

The menus and dialog boxes shown here may differ slightly depending on the version of the Keysight IO Libraries Suite.

- 1 Choose **Start > All Programs > Keysight Connection Expert** from the Windows Start menu. Or, click on the Keysight IO Control icon in the taskbar, and choose **Connection Expert** from the popup menu.

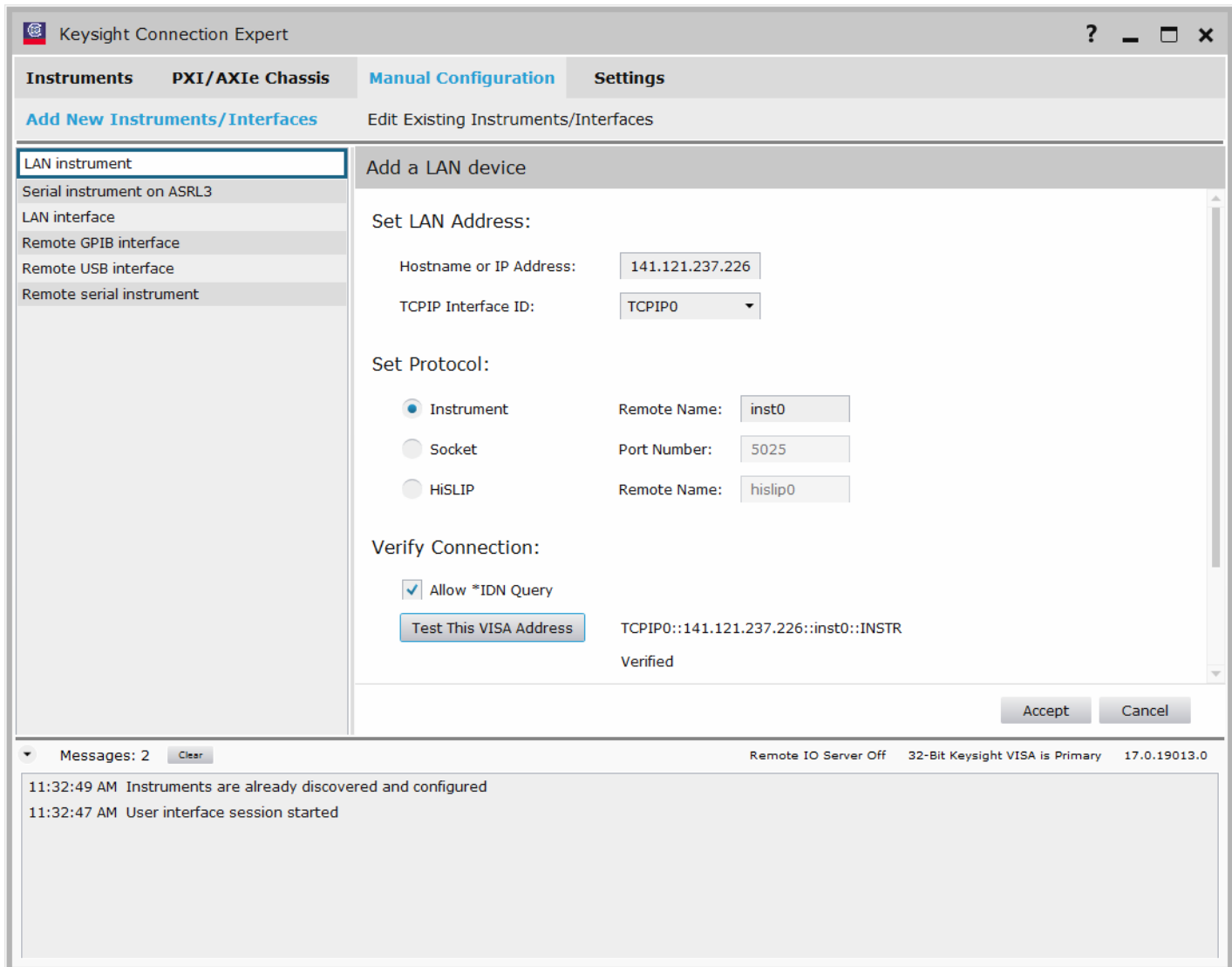


- 2 In the Keysight Connection Expert application, instruments connected to the controller's USB and GPIB interfaces as well as instruments on the same LAN subnet should automatically appear in the Instruments tab.

The screenshot displays the Keysight Connection Expert application window. The title bar reads "Keysight Connection Expert" and includes standard window controls. The main interface is divided into several sections:

- Navigation Tabs:** "Instruments", "PXI/AXIe Chassis", "Manual Configuration", and "Settings".
- Instrument List:** A list of discovered instruments, each with a star icon and a TCPIP address. The selected instrument is "PCSERNO, Agilent" with address "TCPIP0::141.121.237.226::inst0::INSTR".
- Details for Agilent PCSERNO:**
 - Manufacturer: Agilent
 - Model: PCSERNO
 - Serial Number: SR79456864
 - Firmware Version: 05.20.0010
- Connection Strings:**
 - VISA Addresses: TCPIP0::141.121.237.226::inst0::INSTR (checked)
 - VISA Aliases: (empty)
 - SICL Addresses: (collapsed)
- Installed Drivers:** A section with an "Update Drivers" button.
- Messages:** A log at the bottom showing two messages: "11:32:49 AM Instruments are already discovered and configured" and "11:32:47 AM User interface session started".
- Status Bar:** Shows "Remote IO Server Off", "32-Bit Keysight VISA is Primary", and version "17.0.19013.0".

- 3 If your instrument does not appear, you can add it using the Manual Configuration tab.

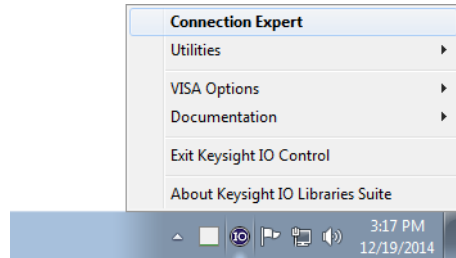


For example, to add a device:

- a Select **LAN instrument** in the list on the left.
- b Enter the oscilloscope's **Hostname** or **IP address**.
- c Select the protocol.
- d Click **Test This VISA Address** to verify the connection.

Interactively Sending SCPI Commands to Instruments

- 1 Click on the Keysight IO Control icon in the taskbar and choose **Connection Expert** from the popup menu.



- 2 In the Keysight Connection Expert, select your instrument from the list on the left.
- 3 In the Details for the selected instrument, click **Send Commands To This Instrument**.

Keysight Connection Expert

Instruments PXI/AXIe Chassis Manual Configuration Settings

Rescan Filter Instruments: Clear

PCSERNO, Agilent
TCPIP0::LAB-JENGA-PP-5.cos.is.keysight.com::his

PCSERNO, Agilent
TCPIP0::141.121.238.47::inst0::INSTR

PCSERNO, Agilent
TCPIP0::141.121.237.226::inst0::INSTR

PCSERNO, Agilent
TCPIP0::lab-trex-amc-1.cos.is.keysight.com::inst

PCSERNO, Agilent
TCPIP0::2UA3060KKZ-2.local::hislip0::INSTR (+)

PCSERNO, Agilent
TCPIP0::LAB-GRX-LP2-2.cos.is.keysight.com::his

PCSERNO, Agilent

Details for Agilent PCSERNO

Manufacturer: Agilent [View Instrument Information Online](#)
 Model: PCSERNO [Start Instrument Web Interface](#)
 Serial Number: SR79456864 [Delete User-Added Connections](#)
 Firmware Version: 05.20.0010

Connection Strings

VISA Addresses

TCPIP0::141.121.237.226::inst0::INSTR [Send Commands To This Instrument](#)
[Start IO Monitor](#)
[Start Command Expert](#)

VISA Aliases [Add or Change Aliases](#)

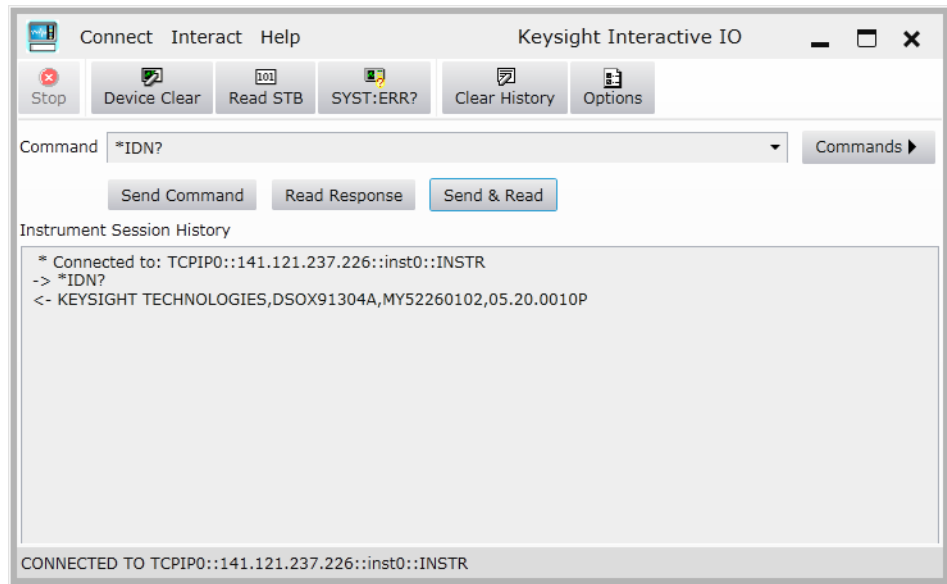
SICL Addresses

Installed Drivers [Update Drivers](#)

Messages: 4 Clear Remote IO Server Off 32-Bit Keysight VISA is Primary 17.0.19013.0

3:26:13 PM Updated connection TCPIP0::141.121.237.226::inst0::INSTR to agilent,pcserno,sr79456864
 3:26:12 PM You are updating the information for TCPIP0::141.121.237.226::inst0::INSTR. You will see a confirmation when it is processed.
 11:32:49 AM Instruments are already discovered and configured
 11:32:47 AM User interface session started

- In the Keysight Interactive IO application, enter commands in the **Command** field and press **Send Command**, **Read Response**, or **Send & Read**.



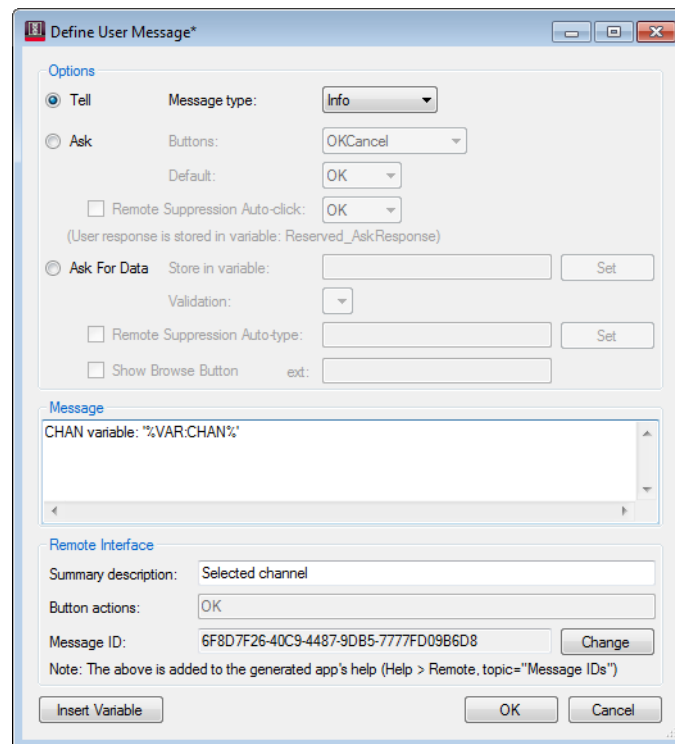
To see variable values as tests run

When you place variables into test steps and you wonder what the SCPI command looks like when it goes to the instrument (after the variables are substituted), there are a couple of options.

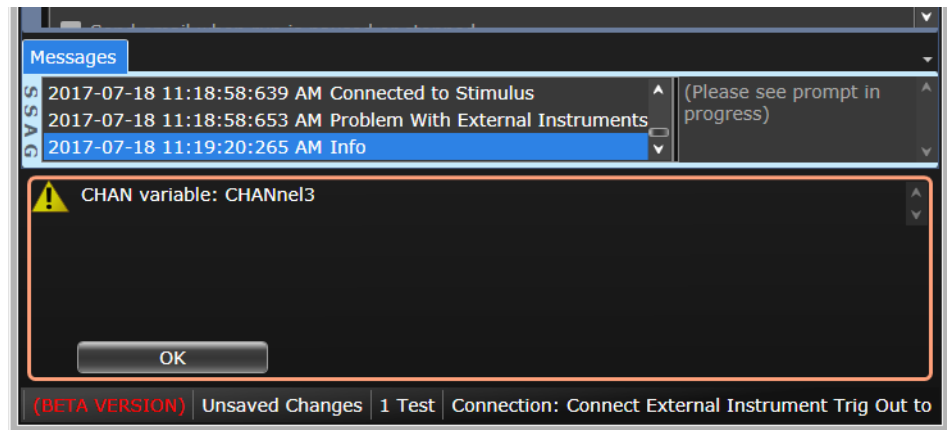
For example, if you have `":CHAN%VAR:Channel%:DISP ON"` in a command step where Channel is an internal variable, you might be surprised to find that `":CHAN:3000.e000:DISP ON"` is sent (which clearly will not work.)

There are a couple ways to see what is sent:

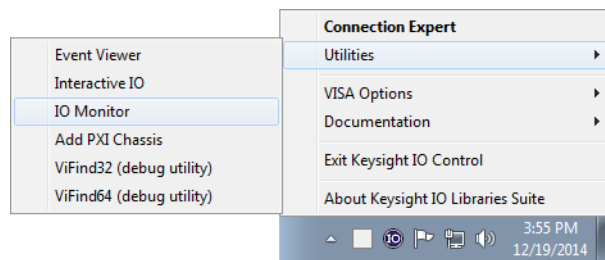
- 1 Put the variable dereference syntax in a Tell Info Display Message step.



This shows:



- 2 Better still, use the IO Monitor that comes with the Keysight IO Libraries Suite (version 15.1 and later).



When the IO Monitor opens, select **Start Capturing Messages** before you run your application.

The IO Monitor shows:

Keysight IO Monitor

File View Capture Tools Help

Stop Capturing Messages Save Messages... Open Message File... Print Clear Messages Options...

Time Stamp	Program	Address	Source	Method Call	IO Data	Return Value	Time(ms)
18:42:28.304	Keysight.DigitalTest	TCPIP0::141.121.	Keysight VISA	VISA::viSetAttribu		VI_SUCCESS	0.036782
18:42:28.304	Keysight.DigitalTest	TCPIP0::141.121.	Keysight VISA	VISA::viWrite	muxpos="1">0</val>.\n<val muxr	VI_SUCCESS	0.79693
18:42:28.305	Keysight.DigitalTest	TCPIP0::141.121.	Keysight VISA	VISA::viSetAttribu		VI_SUCCESS	0.039347
18:42:28.305	Keysight.DigitalTest	TCPIP0::141.121.	Keysight VISA	VISA::viGetAttribu		VI_SUCCESS	0.039918
18:42:28.305	Keysight.DigitalTest	TCPIP0::141.121.	Keysight VISA	VISA::viSetAttribu		VI_SUCCESS	0.036781
18:42:28.305	Keysight.DigitalTest	TCPIP0::141.121.	Keysight VISA	VISA::viWrite	uxpos="4">1</val>.\n</ctrl>.\n<c	VI_SUCCESS	0.795505
18:42:28.306	Keysight.DigitalTest	TCPIP0::141.121.	Keysight VISA	VISA::viSetAttribu		VI_SUCCESS	0.041914
18:42:28.306	Keysight.DigitalTest	TCPIP0::141.121.	Keysight VISA	VISA::viGetAttribu		VI_SUCCESS	0.039348
18:42:28.306	Keysight.DigitalTest	TCPIP0::141.121.	Keysight VISA	VISA::viSetAttribu		VI_SUCCESS	0.049327
18:42:28.306	Keysight.DigitalTest	TCPIP0::141.121.	Keysight VISA	VISA::viWrite	trl>.\n<ctrl type="f32">.\n<name>	VI_SUCCESS	0.788376
18:42:28.307	Keysight.DigitalTest	TCPIP0::141.121.	Keysight VISA	VISA::viSetAttribu		VI_SUCCESS	0.044195
18:42:28.307	Keysight.DigitalTest	TCPIP0::141.121.	Keysight VISA	VISA::viGetAttribu		VI_SUCCESS	0.041629
18:42:28.307	Keysight.DigitalTest	TCPIP0::141.121.	Keysight VISA	VISA::viSetAttribu		VI_SUCCESS	0.037352
18:42:28.307	Keysight.DigitalTest	TCPIP0::141.121.	Keysight VISA	VISA::viWrite	l type="f64">.\n<name>YgpPosn </	VI_SUCCESS	0.783815
18:42:28.308	Keysight.DigitalTest	TCPIP0::141.121.	Keysight VISA	VISA::viSetAttribu		VI_SUCCESS	0.039918
18:42:28.308	Keysight.DigitalTest	TCPIP0::141.121.	Keysight VISA	VISA::viWrite	l>0</val>.\n</ctrl>.\n<ctrl type="	VI_SUCCESS	0.733917
18:42:28.310	Keysight.DigitalTest	TCPIP0::141.121.	Keysight VISA	VISA::viClose		VI_SUCCESS	13.7654
18:42:28.326	Keysight.DigitalTest	defaultRM	Keysight VISA	VISA::viClose		VI_SUCCESS	0.068715
18:42:28.327	Keysight.DigitalTest	defaultRM	Keysight VISA	VISA::viClose		VI_SUCCESS	0.317347

Message Details:

Parameters:

Name	Type	Value

Array Data:

Index	Value

Monitoring On, Logging Disabled

If there are problems uninstalling a generated app

Note: To manually remove a generated app, delete these directories from the machine where the app is installed:

(FlexDCA App, WinXP)

- C:\Documents and Settings\All Users\Application Data\Keysight\FlexDCA\Apps\User Defined**<appname>**
- C:\Program Files\Keysight\FlexDCA\Apps\User Defined**<appname>**

(FlexDCA App, Win7)

- C:\ProgramData\Keysight\FlexDCA\Apps\User Defined**<appname>**
- C:\Program Files (x86)\Keysight\FlexDCA\Apps\User Defined**<appname>**

(Infiniiium App, WinXP)

- C:\Documents and Settings\All Users\Application Data\Keysight\Infiniiium\Apps\User Defined**<appname>**
- C:\Program Files\Keysight\Infiniiium\Apps\User Defined**<appname>**

(Infiniiium App, Win7)

- C:\ProgramData\Keysight\Infiniiium\Apps\User Defined**<appname>**
- C:\Program Files (x86)\Keysight\Infiniiium\Apps\User Defined**<appname>**

Build Errors

- "Error UDA100" on page 411
- "Error UDA101" on page 411
- "Error UDA200" on page 412
- "Error UDA201" on page 412
- "Error UDA300" on page 412
- "Error UDA301" on page 412
- "Error UDA400" on page 412
- "Error UDA401" on page 412
- "Error UDA500" on page 412
- "Error UDA501" on page 412
- "Error UDA502" on page 413
- "Error UDA503" on page 413
- "Error UDA504" on page 413
- "Error UDA505" on page 413
- "Error UDA506" on page 413
- "Error UDA508" on page 414
- "Error UDA600" on page 414
- "Error UDA700" on page 414
- "Error UDA701" on page 414
- "Error UDA800" on page 414

Error UDA100

A generated application must have at least a name and one test to be viable.

Error UDA101

When creating an Add-In, a feature was used that is not available in the requested "Add-In Interface" (see actual error for a description of the specific feature). The desired Add-In Interface is selected in the Generator Tool's "Compatibility" tab. By selecting a higher (newer) interface, you will be able to access more features (see Interface/Feature table on that tab). However, the version of the compliance application you are building the add-in for may not yet support the latest add-in interface. See the compliance app's **Help > About** screen to determine what its capabilities are.

Error UDA200

The action required a path to an existing file, but could not find the file. Check to make sure the path exists.

Error UDA201

The item was referenced by one or more steps or events, but no longer exists. Either create the item or modify the references.

Error UDA300

Two or more items were found to have the same name but require unique names. Please rename one of these items.

Error UDA301

Two or more files were found to have the same name. These files will be copied to the same directory, either for Debug Run / Launch or by the generated app's installer. Therefore, they must have unique names.

Error UDA400

A piece of required information is missing. Please go to the associated dialog box and enter it.

Error UDA401

The item was empty, but requires contents to compile. Either remove the item or use the item's editor to add contents.

Error UDA500

The math parser found an error in an expression. Please go to the associated dialog box and modify the expression.

Error UDA501

The sequence of subroutine calls shown in the error will result in a recursive subroutine call, which is not allowed. The last call in the sequence can also be found elsewhere in the sequence. Modify the sequence to remove the loop.

Error UDA502

Intermediate Value steps are used to add optional items to the results of a particular test. Because Events execute *in between* tests, it is invalid to execute an Intermediate Value step there. Review the call sequence specified in the error message to determine where the Intermediate Value step is located.

Error UDA503

The following flow control step may not appear in a step list without the specified associated step(s) also appearing in the same step list:

Step	Requires	Ordering
If	Endif	Must appear before Endif
Else	If, Endif	Must appear between If and Endif
Endif	If	Must appear after If
While	EndWhile	Must appear before EndWhile
Break	While, EndWhile	Must appear between While and EndWhile
Continue	While, EndWhile	Must appear between While and EndWhile
EndWhile	While	Must appear after While

Error UDA504

This error will be reported on one of two situations:

- A number was expected but alphanumeric text was entered.
- A variable was expected but the name entered is not a current variable.

Error UDA505

Pass limits must be internally consistent. For range-type limits (such as $1 < \text{VALUE} < 2$), the min must be less than the max. For single-ended limits of value 0 (such as $0 < \text{VALUE}$), the nominal must either be a variable or an in-limits number.

Error UDA506

The text parser found an error in an expression. Please go to the associated dialog box and modify the expression.

Error UDA508

A switch matrix signal path in the indicated Connection contains invalid and/or conflicting settings. The error message will indicate if the problem involves the test point or the oscilloscope channel.

Error UDA600

The generated app may not be able to process the specified screen shot or external image on Windows XP because the full path it will generate may exceed the file system limit of 259 characters.

The full path includes the following items (please shorten one of these to remove this error):

- The generated application name prefix.
- The generated application name.
- The test's remote interface ID.

Error UDA700

The generated app checks the version of each project it attempts to load to ensure compatibility. The minimum project version specified will prevent the generated app from opening any projects, either legacy or ones it creates itself. Please reduce the minimum required project version to be equal to or lower than the current version number, which can be found in the Set Up tab.

Error UDA701

The combination of scope option requirements specified for the project is invalid. Please modify them (**Tools > Scope Option Requirements**).

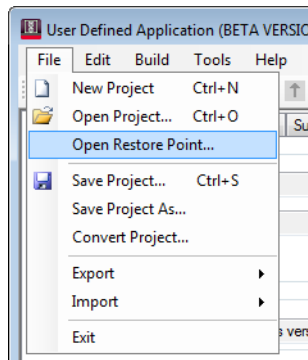
Error UDA800

Configs that are assigned to a group are only displayed when at least one test from that group is available on the generated app's "Select tests" tab. Please move this config to the "Ungrouped" selection or to a group that has at least one test assigned.

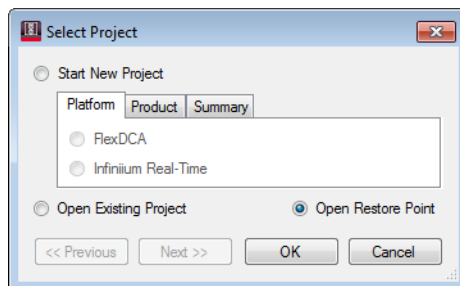
Restoring the Project

At various times during the development of your project, the application generator saves an internal snapshot of the project as a "restore point". At any time, you can load a Restore Point to return your project to the state it was in at that point in time.

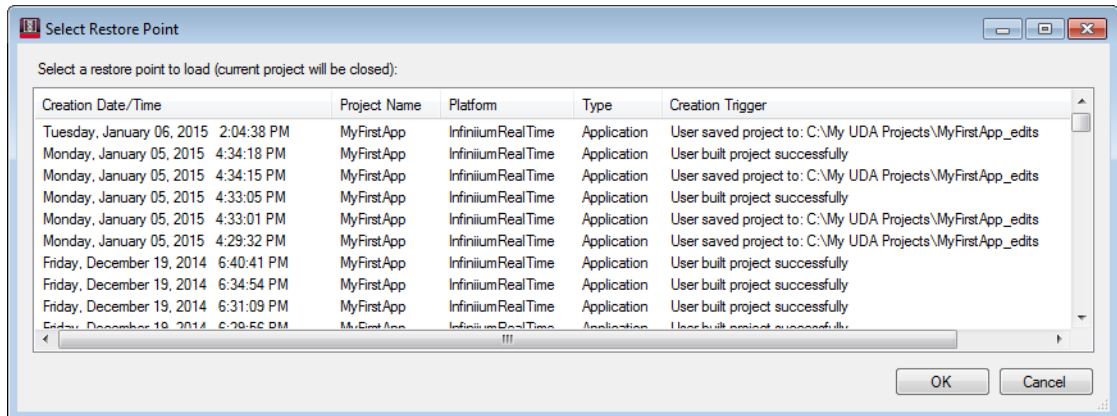
To access a restore point, use the File menu:



or use the New Project dialog box:



You will be shown the list of available Restore Points. For example:



The **Creation Trigger** describes what caused the Restore Point to be created, helping you to decide which one to load in order to "undo" one or more previous actions. In the example above, you could select the Restore Point created at 4:47:02 to return to a state right before the test "MyTest" was deleted.

To manage or learn more about Restore Point creation, use the Options dialog box (Miscellaneous tab).

20 Menu/Toolbar/Options Reference

Main Menu / 418

Toolbar Reference / 422

Options Reference / 423

Main Menu

Table 2 Menu Reference

Menu	Shortcut	Description
File > Convert Project		If the current project is for an application, it is converted to an add-in. If the current project is an add-in, it is converted to an application.
File > Export > Definitions		Exports text files containing test/subroutine/event step summaries.
File > Export > Subroutine		Exports a subroutine to a file that may be imported by other UDA projects.
File > Export > Test		Exports a test to a file that may be imported by other UDA projects.
File > Import > Subroutine		Imports a subroutine from a file previously created by a UDA Generator.
File > Import > Test		Imports a test from a file previously created by a UDA Generator.
File > New Project	Ctrl+N	Resets entire project, but leaves user preferences (Tools > Options) alone. Should prompt for loss of unsaved changes.
File > Open Project...	Ctrl+O	Opens an existing UDA project.
File > Open Restore Point		Opens an internal project snapshot.
File > Save Project...	Ctrl+S	Saves existing configuration to the project file.
File > Save Project As...		Saves existing configuration to a different project file.
File > Exit		Exit the tool.
Edit > Add Item...	Ctrl+A	Adds a new item.
Edit > Add Group...	Ctrl+P	Adds a test group.
Edit > Edit Item...	Ctrl+E	Opens a dialog box for modifying the currently selected item.
Edit > Copy Item...	Ctrl+C	Creates a new item copied from the currently-selected item.
Edit > Delete Item...	Ctrl+D	Deletes the currently selected item.
Edit > Move Item Up	Ctrl+Up	Moves the currently-selected item up one in its list.
Edit > Move Item Down	Ctrl+Down	Moves the currently-selected item down one in its list.

Table 2 Menu Reference (continued)

Menu	Shortcut	Description
Edit > Move Item To Group	Ctrl+Right	Moves the currently-selected object to the following group of objects.
Edit > Assign connection to test	Ctrl+Right	Not implemented yet.
Edit > Find...	Ctrl+F	Opens a dialog box for finding text strings used in the project.
Build > Build application	Ctrl+B	Copies user files to the internal cache and builds (makes sure all files can be located).
Build > Launch application	Ctrl+L	Builds and runs the application (tests the application before installing).
Build > Generate installer		Builds the application and generates a .zip installation file.
Build > Install application		Builds the application and installs (only available when running the application generator on the Infiniium oscilloscope).
Tools > Debug Run > Reset Variables		Displays the Reset Variables screen.
Tools > Debug Run > Set Variables		Displays the Set Variables for Debug Run screen.
Tools > Manage > Command Files...		Find and replace usage of referenced command files.
Tools > Manage > Connection Diagrams...		Find and replace usage of referenced connection diagrams (html or images).
Tools > Manage > Console Applications...		Find and replace usage of referenced console applications.
Tools > Manage > External Application Files...		Find and replace usage of referenced external application files.
Tools > Manage > External Instruments...		Find and replace usage of referenced external instruments.
Tools > Manage > Internal Variables...		Find and replace usage of referenced internal variables.
Tools > Manage > Math Libraries...		Find and replace usage of math libraries. Also can add math libraries.
Tools > Manage > Other Files...		Add or remove the installation of miscellaneous files and User Defined Function files with the generated app.
Tools > Manage > Reserved Variables...		Opens a dialog box that displays the Reserved variables.

Table 2 Menu Reference (continued)

Menu	Shortcut	Description
Tools > Manage > Scope Mask Files...		Find and replace usage of scope mask files.
Tools > Manage > Scope Setup Files...		Find and replace usage of referenced oscilloscope setup files.
Tools > Manage > Text Libraries...		Find and replace usage of text libraries. Also can add text libraries.
Tools > Manage > Transfer Function Files...		Find and replace usage of transfer function files.
Tools > Generated App Compatibility...		Opens a dialog box for managing generated application compatibility.
Tools > Options...		Displays the Options dialog box.
Tools > Scope Option Requirements...		Specify scope options to be required by the generated app/add-in.
Help > Contents		Opens the application generator's users guide in HTML Help format, with the Contents tab selected.
Help > Index		Opens the application generator's users guide in HTML Help format, with the Contents tab selected.
Help > Search		Opens the application generator's users guide in HTML Help format, with the Contents tab selected.
Help > PDF Document		Opens the application generator's users guide in PDF format.
Help > Generated App > Contents		If you have specified a help file for your generated application, this menu item lets you view that help file.
Help > Generated App > PDF		If you have specified a PDF file for your generated application, this menu item lets you view that file.
Help > Generated App > Remote Interface		If you have generated an installer for your application, this menu item lets you view the remote interface programmer's reference documentation.
Help > On the Web > Share UDA Website		Connects to the Keysight Discussion Forum for Infiniium oscilloscope user-defined applications.

Table 2 Menu Reference (continued)















Menu	Shortcut	Description
Help > On the Web > Home Page		Connects to the UDA product page on www.keysight.com .
Help > About User Defined Application		Displays version and copyright info.

Toolbar Reference



Figure 1 Toolbar

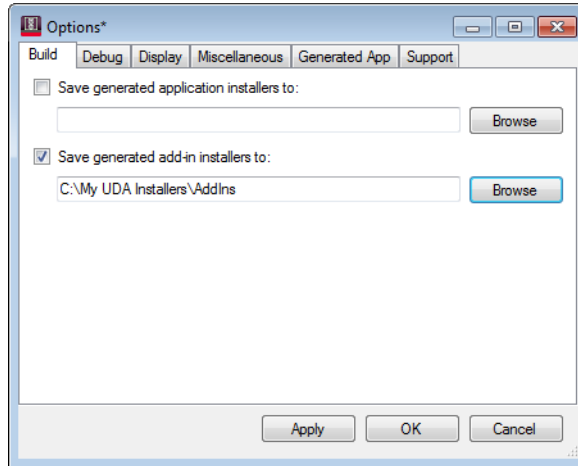
Table 3 Toolbar Button Descriptions

Toolbar Icon	Description
	New project.
	Open project.
	Save project.
	Add new item.
	Add a group.
	Edit the selected item.
	Copy the selected item.
	Delete the selected item.
	Move the selected item up in the hierarchy.
	Move the selected item down in the hierarchy.
	Move the selected item into a group.
	Finds text strings used in the project.
	Build and run the application.
	Opens the online help.

Options Reference

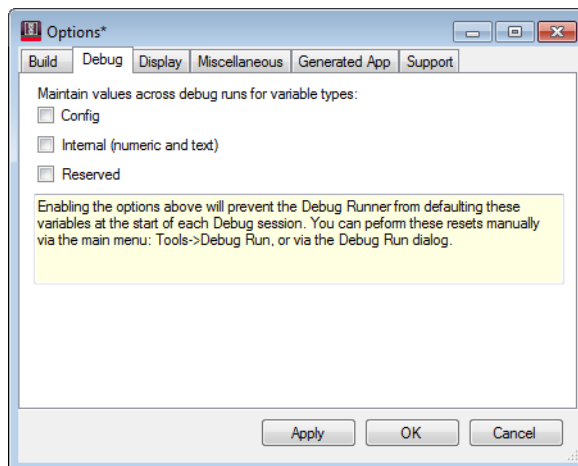
You can open the Options dialog box by choosing **Tools > Options...** from the application generator's main menu. This dialog box lets you set display, Miscellaneous, and support options.

Build Tab



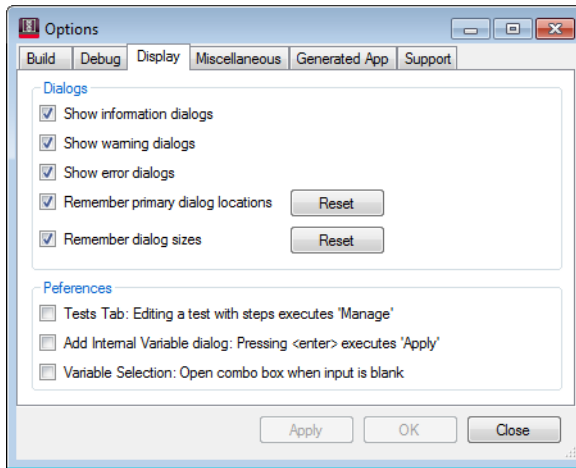
Save generated application installers to	Specifies the directory where the application generator should place newly created application installers.
Save generated add-in installers to	Specifies the directory where the application generator should place newly created add-in installers.

Debug Tab



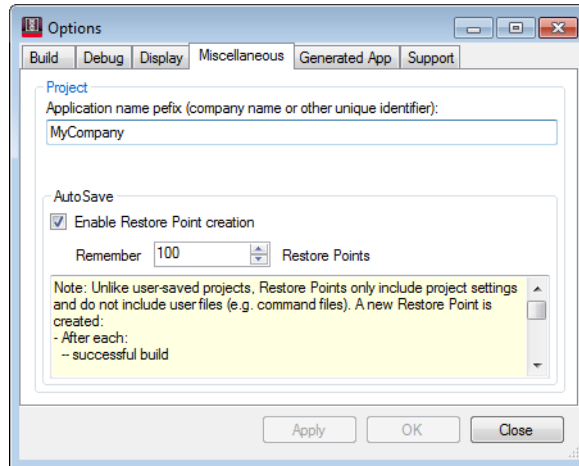
Maintain config variable values across debug runs	Prevents the Debug Runner from defaulting all config variables at the start of each Debug session.
Maintain internal variable values across debug runs	Prevents the Debug Runner from defaulting all internal variables at the start of each Debug session.
Maintain reserved variable values across debug runs	Prevents the Debug Runner from defaulting all reserved variables at the start of each Debug session.

Display Tab



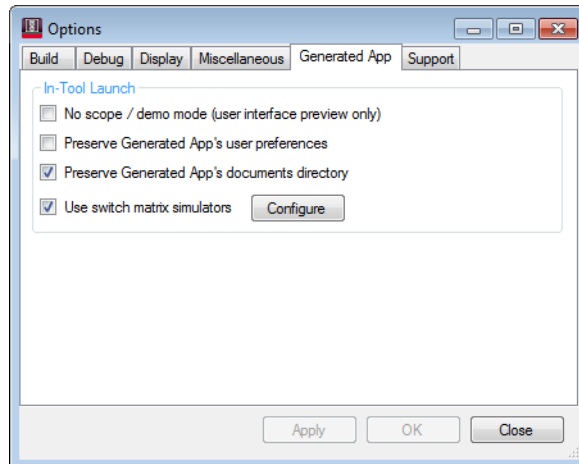
Show information dialogs	Enables/disables information dialog boxes when using the application generator.
Show warning dialogs	Enables/disables warning dialog boxes when using the application generator.
Show error dialogs	Enables/disables error dialog boxes when using the application generator.
Remember primary dialog locations	Specifies whether primary dialog box locations are saved. Click Reset to return to the default locations.
Remember dialog sizes	Specifies whether dialog box sizes are saved. Click Reset to return to the default sizes.
Tests Tab: Editing a test with steps executes 'Manage'	Specifies whether or not to automatically open the Manage Steps dialog box whenever you edit a test that has steps.
Add Internal Variable dialog box: Pressing <Enter> executes 'Apply'	Specifies whether or not pressing <Enter> will close this dialog box after creating the variable.
Variable Selection: Open combo box when input is blank	Specifies whether a variable selection combo box will automatically drop down to show its contents when the input field is blank.

Miscellaneous Tab



Application name prefix	Prevents generated application name conflicts. For more information, see "Enter Application Name and Version" on page 86.
Default...	The New Project setting to use for the Compatibility tab minimum Add-In Interface.
Enable Restore Point creation	Automatically create internal project snapshots.

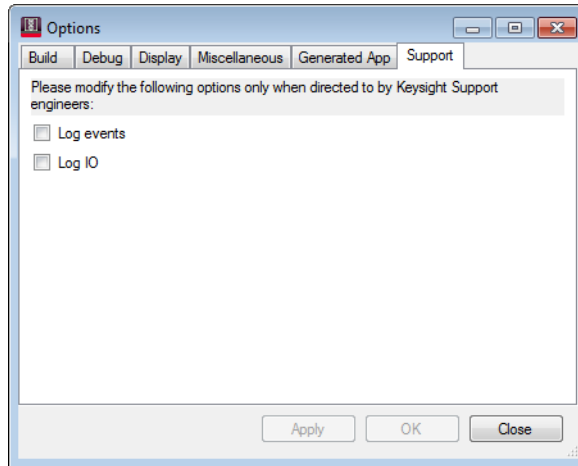
Generated App Tab



Launch with No scope / demo mode	Lets you preview the generated application's user interface.
Launch with Preserve user preferences	User preferences from the generated app's last run will be restored.

Launch with Preserve documents directory	The documents directory from the generated app's last run will be restored.
Launch with Use switch matrix simulators	Simulators will be offered when connecting to a switch instrument.

Support Tab



Log events	Only enable event logging as directed by Keysight Support engineers.
-------------------	--

Glossary

A

application generator This is the short name for this User Defined Application Generator.

C

console application This is an interactive application that is used, for example, to control a device under test (DUT). You can launch a console application in one test step and send the console application commands in subsequent test steps.

D

development PC This is a Windows operating system-based computer that runs the application generator. The application generator can also be run on an Infiniium oscilloscope.

DUT Device Under Test.

E

external application This is an application (external to the generated application) that is used to post-process oscilloscope data and generate test results. For example, MATLAB is an external application.

G

generated application This is the output of the user defined application generator. It is the application that performs automated testing.

U

UDA User Defined Application.

Index

A

- abort, calling, 112
- acquisition setup (in debug mode), 265
- activate limit set, 257
- add-in, 24
- add-in compatibility, determining, 81
- Add-In Interface, options, default, 425
- add-in, converting, 244, 245
- adding tests, 28
- any test group starts, 171
- app exits, 171
- app shown, 171
- app shown (demo mode), 171
- append, 114
- append new line, 114
- application add-in, 24
- application generator, 427
- application generator projects, 239
- application generator, installing, 18
- application name, 27, 86
- application name prefix, 86, 425
- application prefix, 24
- application version, 86
- application version number, 27
- application, creating, 24
- assignment summary, oscilloscope channel, 166
- asterisk wildcard when selecting variables, 112
- automatic mode, switch matrix signal paths, 389
- Automating the generated application, 185
- Automation Control creation, Automation, 297
- Automation tab, 295
- AutoRecovery preferences, 380

B

- backslash operator, text expression, 205
- backward compatibility, 232
- bandwidth limiting, 266
- best trials, store mode, 271
- beta versions, 28
- binary file transfers, external instrument, 91
- break, 129
- break, adding, 129
- build application, 184
- build errors, 411
- build number, 28

- buttons, switch matrix settings, 397

C

- call subroutine, 50
- calling abort, 112
- calling subroutine, 112
- channel (oscilloscope), variable, 165
- chronological test display, 305
- collaborating on projects, 23, 247
- color scheme, 290
- combine limits, 261
- command file, 31
- command files, managing, 212
- command line (Automation), trying, 300
- comma-separated value (CSV) format, export results, 309
- comment, adding, 126
- comments in command files, 31
- company logo, 175
- compatibility (backward) and remote interface IDs, 97
- compatibility, generated application, 232
- complete application, 24
- condition, verifying, 121
- config variable auto-iteration, 292
- Config variable substitution, 41
- Config variable value source, 200
- Config variables, 38
- Config variables in math expressions, 202
- Config variables, adding, 147
- Config variables, grouping, 149
- configuration mode, switch matrix controller, 387
- configuring tests, 256
- connecting oscilloscope to DUT, 267
- connection changes, 51, 171
- connection diagram HTML, managing, 214
- connection diagram images, managing, 216
- connection image file, install location, 45
- connection instructions, 43, 151
- connection options, switch matrix controller, 389
- connections, assigning switch matrix test points to, 159, 161
- console application, 3, 105, 427
- console applications, managing, 218
- continue, 129
- continue, adding, 129
- Controller tab, switch matrix settings, 386
- converting projects, 239, 244, 245

- copyright, 2
- create limit set, 258
- creating an application, 24
- creating install packages, 23
- creating test project, 252
- critical at percent of margin, 304
- CSV (comma-separated value) format, export results, 309
- CSV export preferences, 312
- custom connection diagram, 44
- custom UDA licensing, 237
- custom-formatted test results, 76

D

- D9010UDAA User Defined Application license, 19, 184
- Data Analytics software, KS6800A, 226
- Debug Run, 142
- Debug Run feature, 3
- Debug Run options, 177
- Debug Run, button, 142
- defaulting the oscilloscope, 100
- description of test, 29
- description, external instrument, 91
- details limits, 57
- development PC, 35, 427
- development PC, requirements, 16
- dialogs, message, 424
- directory separator character, 110
- display list selection, 118
- display messages, 116
- display preferences, 290
- divide operator, math expression, 203
- double quote operator, text expression, 205
- duplicate file names, 23
- DUT (Device Under Test), 427
- DUT controller programs, 3
- DUT, connecting to oscilloscope, 267

E

- edit limit set, 258
- else, 127
- else, adding, 127
- email on pause or stop, 273
- enable margin reporting, 304
- enable remote interface, 384
- Endif, 127
- Endif, adding, 127

endwhile, 129
 endwhile, adding, 129
 engineering notation, 202
 Entire Script creation, Automation, 296
 error dialogs, 290, 424
 Error messages, 117
 error query, external instrument, 91
 Event messages, 117
 event specification, 289
 event trials, store mode, 271
 events, 51
 events, actions on, 171
 events, pause or stop on, 288
 exclude this definition from ..., 27
 Exclude, external instrument, 91
 executing scripts, 382
 execution (Automation), starting, 298
 execution order, test, 75
 export preferences, 312
 export tests or subroutines, 137
 export user files, 242
 exporting the report, 309
 external application, 108, 427
 external application files, managing, 221
 external applications, 3
 external instrument value source, 207
 external instruments, 3
 external instruments, adding, 91
 external instruments, in generated applications, 253
 external measurement result, 73
 extract to subroutine, 133

F

fail event, 289
 features, 3
 features, generated applications, 3
 file value source, 206
 file, writing, 113
 files (working), organizing, 23
 Files mode Automation, 298
 filters, test group, 88
 final result, 31, 135
 find/replace text strings in project, 70
 floating point values, 202
 forever, run until setting, 272
 full control, 23
 FullScreen image, 32, 125, 136

G

generate installer, 59, 184
 generated app, annually remove, 410
 generated application, 427
 generated application compatibility, 232
 generated application, testing before install, 35
 generated applications, requirements, 16
 generated applications, testing, 23

generating the installer, 183
 glossary, 427
 Graticule screen image, 32, 125, 136
 group starts, 171, 172
 group user defined ends, 172
 group user defined starts, 171
 grouping tests, 140

H

help files, adding, 53
 hints, remote interface, 385
 host elements, 27
 how to ..., 69
 HTML format, export results, 311
 HTML report, 96
 HTML report image, 55, 175
 HTML test report, viewing/exporting/printing, 308

I

icon, application title bar, 53
 if, 127
 if, adding, 127
 images limits, 57
 images, adding, 53
 images, logo, 175
 import tests or subroutines, 137
 in this guide, 5
 InfiniiSim, accessing, 263
 Infiniium features, accessing, 263
 Infiniium oscilloscope software, restart, 65
 Info messages, 117
 information dialogs, 290, 424
 install application, 184
 install package, 3
 install packages, creating, 23
 installation, application generator, 15
 installer, generate, 59, 183
 installers on PC, allow, 184
 installing an add-in, 61
 installing the application generator, 18
 installing the generated application, 61
 instruments, connect problems, 400
 intermediate value, 124
 Internal Variable dialog box, Enter key, 424
 internal variable, setting, 111
 internal variables, managing, 190
 InternalError messages, 117
 invert scope display, 290
 IO Monitor, 408
 IP protection, 237

J

jump actions, 306

K

keep application on top, 290
 Keysight Connection Expert, 401, 404
 Keysight Interactive IO application, 405
 Keysight IO Control icon, 404
 KS6800A Data Analytics software, 226

L

LAN interface, 401
 last trials, store mode, 271
 launch application, 184
 launch the generated application, 65
 launch with no scope, 425
 launch with preserve documents directory, 426
 launch with preserve user preferences, 425
 launch with use switch matrix simulators, 426
 licensing, 19
 limit set, activate/refresh, 257
 limit set, create/edit, 258
 limit set, restore, 261
 limit, split combined, 262
 limits, 32, 136
 limits, combine, 261
 line breaks in test description/references, 96
 list selection, display, 118
 load preferences, 252
 locate user files, 242
 location, installer, 60
 log application generator events, 426
 logos, adding, 53

M

major version number, 28
 Manage Step dialog box, auto open, 424
 managing resources, 211
 manual mode, switch matrix signal paths, 392
 manually remove generated app, 410
 manually running tests, 22
 margin < N event, 289
 margin thresholds, changing, 304
 mask file, 178
 mask files, managing, 230
 math expression (internal variable) value source, 201
 math expression value source, 201
 math libraries, managing, 192
 MATLAB, 3
 MATLAB scripts, 109
 MATLAB scripts, how to execute, 78
 menu reference, 417, 418
 merging projects, 247
 messages, display, 116

- minimize after prompt, 290
- minor version number, 28
- minus operator, math expression, 202
- miscellaneous files, managing, 223
- miscellaneous options, 173
- multi-file HTML, export results, 312
- multiple return values, 198, 204, 206, 207
- multiple runs, 271
- multiply operator, math expression, 202
- multi-segment application name prefixes, 86

N

- N times, run until setting, 272
- N5452A remote interface, 383
- N5452A Remote Interface for Infiniium Compliance Applications, 18
- name, generated application, 27
- nickname, external instrument, 91
- no error string, external instrument, 91
- no scope launch, 425
- nonstandard external instrument, 91
- notices, 2

O

- once, run until setting, 272
- online help, adding, 174
- online help, oscilloscope, 5
- opening projects, 239, 242
- opening test project, 252
- operators, math expression, 202
- options reference, 417, 423
- options, switch matrix, 157
- order of test execution, 75
- organizing your working files, 23
- oscilloscope value source, 198
- oscilloscope, connecting to DUT, 267
- oscilloscope, setting based on waveform data, 72
- oscilloscopes supported, 16
- other files, managing, 223
- output full path, 114
- overview, 3
- overwrite, 114

P

- parentheses, math expression, 203
- parentheses, text expression, 205
- pass event, 289
- pause on event, 288
- pause, email on, 273
- pausing, 119
- persist config variables, 424
- persist internal variables, 424
- persist reserved variables, 424

- plus operator, math expression, 202
- plus operator, text expression, 205
- post-processing tools, 3
- PrecisionProbe/PrecisionCable, accessing, 263
- preferences, CSV export, 312
- preferences, display, 290
- preferences, remote, 384
- preferences, report, 304, 305
- preferences, save/load, 252, 380
- prefix, application name, 86
- preserve documents directory launch, 426
- preserve user preferences launch, 425
- preview print, 312
- printing HTML test report, 312
- problems, solving, 399
- programmer's reference, oscilloscope, 5
- programs, external application, 109
- project (generated application), creating or opening, 252
- project (generated application), saving, 380
- project, restoring the, 415
- projects, converting, 239
- projects, opening, 242
- projects, saving, 23, 239, 240
- pulse pattern generators, 3

R

- read-only, 23
- refactoring: extract to subroutine, 133
- reference, test, 29
- refresh limit set, 257
- release versions, 28
- remember dialog locations, 424
- remember dialog sizes, 424
- remote control, 383
- remote interface hints, 385
- remote interface ID, 97
- remote interface version, 383
- remote interface, enable, 384
- remote preferences, 384
- remove generated app manually, 410
- replace text strings in project, 70
- report (HTML), viewing/exporting/printing, 308
- report preferences, 304, 305
- report, exporting, 309
- report, printing, 312
- repository, managing, 226
- required oscilloscope software, 87
- requirements, 16
- requirements, scope option, 234
- reserved variables, 196
- Response Correction tab, switch matrix settings, 396
- restore point, enable creation, 425
- result detail limits, 57
- result tags, adding, 273
- result tags, adding after completion of test run, 282

- result tags, adding before starting a test run, 274
- result tags, configuring, 273
- result tags, deleting, 287
- result tags, modifying, 285
- result, run until, 74
- results, uploading to web repository, 314
- results, viewing test, 303
- return, 126
- return values, multiple, 198, 204, 206, 207
- return, adding, 126
- reusing host elements, 27
- run actions, 51, 171
- Run Button, 3
- run ends, 51, 171
- run multiple times, 271
- run starts, 171
- run until, 271
- run until certain result, 74
- running tests, 268
- running the generated application, 65

S

- sample projects, specifying, 56
- sample switch matrix settings, 160
- save generated add-in installers location, 423
- save generated application location, 423
- save/load preferences, 252, 380
- saving projects, 23, 239, 240
- saving test project, 380
- saving, location, 423
- scientific notation, 202
- Scope Mask File, 103
- scope mask files, managing, 230
- scope option requirements, 234
- scope value source, 198
- SCPI command file, 102
- SCPI command, single, 100
- SCPI commands, 3
- SCPI commands, interactive testing, 404
- SCPI commands, problems, 401
- SCPI errors, manually managing, 123
- SCPI query, 198
- screen shot, including, 125, 136
- Script mode Automation, 295
- scripts, executing, 382
- scripts, executing miscellaneous, 301
- scripts, external application, 110
- search for text strings in project, 70
- segment in application name prefix, 86
- Select Tests order for test display, 305
- selecting tests, 254
- set up, 85, 253
- set up oscilloscope based on waveform data, 72
- set up, application generator, 15
- set variable, 111
- setup file path, 35, 178

- setup file, add-in, 61
- setup file, generated application, 61
- setup file, loading, 100
- setup files, managing, 228
- setup files, preparing, 22
- share UDA web site, 3
- sharing application generator projects, 247
- sharing projects, 23
- show tooltips, 291
- SICL address, external instrument, 93, 181
- SICL address, oscilloscope, 35, 178
- SICL, external instrument, 91
- signal path options, switch matrix controller, 388
- signal path, switch matrix, 161
- Signal Paths tab, switch matrix settings, 389
- simple connection diagram, 43
- simulators, switch matrix, 168
- simulators, switch matrix, creating, 169
- Single Command creation,
 - Automation, 296
- single-file HTML, export results, 311
- sleeping, 121
- solving problems, 399
- splash screen, 66
- split combined limit, 262
- square brackets, math expression, 203
- square brackets, text expression, 205
- starting automated test application, 250
- starting execution, Automation, 298
- status settings, Automation, 299
- steps, extracting to subroutine, 133
- stop on event, 288
- stop, email on, 273
- store mode, 270
- subroutine, calling, 112
- subroutine, extract steps to, 133
- subroutines, 49
- subroutines, exporting/importing, 137
- substitutions, variable, 148, 209
- supported oscilloscopes, 16
- switch drivers, switch matrix controller, 387
- switch matrix automation, 386
- switch matrix control, 153
- switch matrix instruments, 156
- switch matrix simulators, launch with, 426

T

- tabular data, reporting, 125
- test description, 96
- test ends, 171
- test environment setup information, 253
- test execution order, 75
- test filter, 253
- test group filters, 88
- test name, 29, 96
- test point, named, 161
- test point, variable, 163

- test points, switch matrix, 154
- test project, creating or opening, 252
- test project, saving, 380
- test references, 96
- test report (HTML),
 - viewing/exporting/printing, 308
- test results, custom-formatted, 76
- test starts, 51, 171
- test steps, adding in Advanced mode, 98
- test the application, 35
- testing generated applications, 23
- tests, adding, 95
- tests, adding, exporting, importing, 28
- tests, chronological display, 305
- tests, configuring, 256
- tests, display in Select Tests order, 305
- tests, exporting, 95
- tests, exporting/importing, 137
- tests, grouping, 140
- tests, importing, 95
- tests, running, 268
- tests, running manually, 22
- tests, selecting, 254
- tests, viewing results, 303
- text libraries, managing, 194
- text value source, 203
- toolbar reference, 417, 422
- tooltips, show, 291
- Transfer Function, 104
- transfer function file, 178
- transfer function files, managing, 231
- transferring files to/from instruments, 115
- trial display preferences, 305
- trial ends, 171
- trial starts, 51, 171
- trials, best/worst/last, 270, 305
- troubleshooting, 399
- trying a command line, Automation, 300

U

- UDA (User Defined Application), 427
- UDA100, 411
- UDA101, 411
- UDA200, 412
- UDA201, 412
- UDA300, 412
- UDA301, 412
- UDA400, 412
- UDA401, 412
- UDA500, 412
- UDA501, 412
- UDA502, 413
- UDA503, 413
- UDA504, 413
- UDA505, 413
- UDA506, 413
- UDA508, 414
- UDA600, 414
- UDA700, 414
- UDA701, 414

- UDA800, 414
- unavailable tests, 254
- uninstalling an add-in, 61
- user comments, 253
- user preferences, specifying default, 57

V

- value, 114
- values, 189
- values, getting, 198
- variable selection combo box, 424
- variable selection filtering, 112
- variable substitutions, 148, 209
- variable, setting, 111
- variables, 189
- variables in external application script paths, 110
- variables, adding multiple, 190
- variables, viewing during test run, 407
- verify condition, 121
- verify response of console application, 106
- version number, generated application, 27
- version, remote interface, 383
- viewing HTML test report, 308
- viewing test results, 303
- VISA alias, oscilloscope, 35

W

- warn at percent of margin, 304
- warning dialogs, 290, 424
- Warning messages, 117
- warranty, 2
- waveform data, set up oscilloscope based on, 72
- web site, share UDA, 3
- while, 129
- while, adding, 129
- wildcards when selecting variables, 112
- working files, organizing, 23
- worst trials, store mode, 271
- write to file, 113