ECU Software Toolkit Help





Contents

TestStand ECU Software Toolkit	5
Getting Started with the ECU Software Toolkit	5
Overview of Test Program Components	6
Overall ECU Software Toolkit Workflow	12
InstrumentStudio to TestStand Workflow	12
ECU Action/Multi Test Step with Sequence Adapter Workflow	14
Exploring a Basic ECU Toolkit Test Program	17
Example Programs	20
Getting Started With ECU Toolkit	21
Test Steps and Flow	26
Recommended ECU Software Toolkit Test Program Structure and Filenames	27
Mapping DUT Pins to Instrument Channels	28
Natively Supported Instruments	29
Pin Map File XML Structure	30
Connecting Shared Resources in the Pin Map	39
Multiplexers and Multiplexed Connections in a Pin Map	41
Testing Multiple Sites in Parallel	41
Subsystem Considerations	43
Subsystems and Pin Maps	45
Multisite Programming with Switches	47
Switching Workflow	50
NI Switch Executive Virtual Device Configuration	52
Enforce Route/Route Group Naming Scheme	53
Exporting and Importing Test Limits with Text Files	56
Exporting Test Limits from Sequence Files	56
Test Limits Text File Structure	58
Opening Test Limits Text Files in Microsoft Excel	61
Editing Test Limits Text Files	61
Importing Test Limits from Text Files	62
Debugging	64
Reports and Data Logs	68
Debug Test Results Logs	69

Deploying ECU Software Toolkit Test Programs	71
Environment Reference	74
ECU Software Toolkit Sequence Editor UI Configuration	74
Loading the ECU Software Toolkit UI Configuration	75
Modifying the ECU Software Toolkit UI Configuration	77
Restoring ECU Software Toolkit UI Configuration to Default State	78
ECU Toolkit Toolbar Buttons	78
ECU Toolkit Menu	80
ECU Software Toolkit Steps Pane	81
Test Program Editor	82
Pin Map Panel	82
InstrumentStudio Project Panel	83
Test Limits Files Panel	83
Configuration Definition Panel	84
Configurations Panel	85
Pin Map Editor	86
Pin Map Tab	87
Instruments	88
Pins	93
Pin Groups	94
Relays	95
Relay Groups	96
Relay Configurations	97
Sites	98
Connections	98
ECU Software Toolkit 22.1 Step Types 1	.00
ECU Toolkit Group	.05
ECU Multi Test Step 1	.05
ECU Multi Test Step Execution Overview	.06
ECU Multi Test Edit Tabs 1	.08
ECU Action Step 1	.15
ECU Action Step Execution Overview	.16
ECU Action Edit Tabs 1	.17
SLSC 12251/2 Group 1	.19
SLSC FIU - Read Current Step 1	.19
SLSC FIU - Reset Step	.20

Measurement Configuration Group	120
Create Pin Sessions Step	120
Apply Pin Configuration Step	121
Close Pin Sessions Step	122
DMM - Read Measurement Step	122
FGen - Initiate Generation Step	123
FGen - Abort Generation Step	124
FGen - Enable Output Step	125
FGen - Disable Output Step	126
Scope - Initiate Acquisition Step	127
Scope - Read Measurements Step	128
Scope - Read Measurements Step Settings Tab	129
Scope - Abort Acquisition Step	132
SMU/PPS - Initiate Step	133
SMU/PPS Abort Step	134
SMU/PPS Enable Output Step	135
SMU/PPS - Disable Output Step	136
SMU/PPS - Read Measurement Step	137
SMU/PPS - Reset Device Step	139
RMXPS - Enable Output Step	140
RMXPS - Disable Output Step	141
RMXPS - Read Measurement Step	142
RMXPS - Reset Output Protection Step	144
NISE - Open Sessions Step	145
NISE - Close Sessions Step	145
NISE - Pin to Sessions Step	146
NISE - Connect Step	147
NISE - Disconnect Step	148
NISE - Disconnect All Step	149
Dialog Boxes and Windows	150

TestStand ECU Software Toolkit

Use this help file and the <u>getting started</u> content to learn the functionality the TestStand ECU Software Toolkit[™] (ECU Toolkit) adds to TestStand for developing ECUTS software.

The TestStand ECU Software Toolkit[™] (ECU Toolkit) adds the following functionality to TestStand:

- Multisite <u>pin map</u> file and <u>ECU Multi Test</u> step type for developing an <u>ECU test</u> <u>program</u> that runs on multiple test system hardware configurations with a variable number of test sites at a high parallel test efficiency.
- Support for exporting and importing test limits with text files.
- Data types, configurable callbacks, and dialog boxes for <u>specifying test program</u> <u>settings</u> and test lot information.
- Customizable operator interface for enabling or disabling test sites and configuring test lot information.

To navigate this help file, use the **Contents**, **Index**, and **Search** tabs to the left of this window.

© 2022 National Instruments Corporation. All rights reserved. Refer to the <National Instruments>_Legal Information directory for information about NI copyright, patents, trademarks, warranties, product warnings, and export compliance.

Getting Started with the ECU Software Toolkit

Use TestStand and the ECU Software Toolkit with other NI development tools to build, debug, customize, and deploy ECU characterization and production test systems.

Brief Tour of the ECU Software Toolkit

1. From the TestStand Sequence Editor, open Getting Started with ECU Toolkit.seq. Open the file from the <TestStand Public>\Examples\ NI ECUToolkit\Getting Started with ECU Toolkit directory.

The Getting Started with ECU Toolkit.seq sequence file is a simple example ECU test program that demonstrates a multisite test program that uses ECU Software Toolkit step types. <u>The example</u> is configured to test up to two DUTs in parallel, each on a separate test site.

2. Review the following ECU Software Toolkit <u>toolbar buttons</u> to use to control execution and view lot statistics while executing and debugging a sequence.



1. Edit Test Program	6. Export Test Limits from
2. Edit Pin Map File	7. Step Into
3. Configure Lot	8. Step Over
4. Active Configuration	9. Step Out
5. Import Test Limits into	10. Launch InstrumentStudio

Complete the following tasks to learn more about the ECU Software Toolkit:

- Explore the components of a test program.
- Use the TestStand and ECU Software Toolkit example programs, located in the <TestStand Public>\Examples directory, as a starting point for applications you create.

Related information:

- <u>TestStand Directory Structure</u>
- TestStand Sequence Editor

Overview of Test Program Components

An ECU test program can include a pin map file, a main sequence file, subordinate

sequence files, code modules, test limits files, and configurations.

Use the <u>Test Program Editor</u> to complete the following tasks:

- Specify the pin map, InstrumentStudio project, and test limits files
- Create and edit test program configurations
- Configure other settings for the test program

Select ECU Toolkit <u>»</u> Edit Test Program: <filename> or click the Edit Test Program: <filename> button on the ECU Software Toolkit toolbar to launch the Test Program Editor for the sequence file.

Use the TestStand Sequence Editor to complete test program development, configuration, debugging, and execution tasks.

Use the following figure and table to learn about the components of test programs and test stations.



Component	Description
TestStand	The TestStand Sequence Editor is the development environment in which you create, edit, execute, and debug sequences and the tests sequences call.
Sequence Editor	Complete the following steps to launch the sequence editor.

Component	Description	
	 (Windows 8.1/8) Click the NI Launcher tile on the Start screen and selectTestStand » TestStand Sequence Editor. (Windows 7) SelectStart » All Programs » National Instruments » TestStand » TestStand Sequence Editor. (Windows 10) SelectStart » NI TestStand. The sequence editor launches the main window and the Login dialog box. Use the default user name, administrator, in the User Name ring control. Leave the Password field empty. You can use the TestStand User Manager to customize user setting and permissions. Click OK. 	
Pin Map	A <u>pin map</u> defines the instrumentation on the tester, defines the pins on the DUT, and defines how the DUT pins are connected to the tester instrumentation for each test site. Use the <u>Pin Map Editor</u> to view, create, modify, and save pin map files. The pin map file also serves as the channel map file. Select ECU Toolkit <u>»</u> Edit Pin Map File or click the Edit Pin Map File button on the ECU Software Toolkit <u>toolbar</u> to launch the Pin Map Editor. Alternatively, you can select ECU Toolkit <u>»</u> Edit Test Program and then select Pin Map in the <u>Test</u> <u>Program Editor</u> to launch the Pin Map panel. Click the Open file for edit <u>*</u> button to launch the Pin Map Editor.	
Test Program Configurations	Test program configurations define values for conditions that a test program can reference at run time and the test limits file that loads before running a test lot. A test program can use multiple configurations to implement multiple test flows using the same sequences and code modules. Select ECU Toolkit <u>» Edit Test Program</u> and then select Configuration Definition in the <u>Test Program Editor</u> to define configuration settings.	
Test Limits Files	<u>Test limits files</u> define test limits the test program loads before running a test lot. The test program replaces test limits in test steps in the sequence file with those specified in the test limits file. You can embed test limits in the sequence file to prevent viewing or tampering with the limits.	

Component	Description	
	Select ECU Toolkit » Edit Test Program and select Test Limits Files in the Test Program Editor to specify one or more test limits files to make available to the test program configurations. The test program configuration specifies the test limits file that loads before running a test lot. You can create a test limits file by selecting ECU Toolkit » Export Test Limits from or by clicking the Export Test Limits from button on the ECU Software Toolkit teal base to available to the	
	Toolkit toolbar to export test limits from a sequence file into a tab-delimited test limits text file. You can import a test limits file by selecting ECU Toolkit » Import Test Limits to or by clicking the Import Test Limits to₊∎ button on the ECU Software Toolkit toolbar to import test limits from a tab-delimited test limits text file into a sequence file. When you import test limits from a text file, you can update limits in matching tests or replace all tests in matching steps.	
Main Sequence File	The main sequence file contains the sequences that define the <u>test flow</u> by specifying the test steps to execute and the order in which to execute them. The sequence file contains one main sequence named MainSequence and can optionally include one or more subsequences with corresponding test steps. You can use multiple sequences in a test program to keep the test code modular and organized.	
	The ProcessSetup and ProcessCleanup sequences are special sequences that TestStand calls at certain times. TestStand calls ProcessSetup once before starting execution and calls ProcessCleanup after execution completes. Initialization and cleanup of instrumentation typically occurs within these sequences.	
Test Steps	<u>Test steps</u> in the sequence called by the <u>ECU Action</u> or <u>ECU Multi Test</u> step are instances of TestStand and <u>ECU Toolkit Measurement Control</u> step types that control instrumentation on the tester, take measurements from the DUT, and pass measurement values back to ECU Multi Test steps. ECU Multi Test steps perform tests by comparing measurement values obtained by test steps inside the called sequence to test limits stored on the step. You also define test numbers and names in the <u>ECU Multi Test</u> step.	
Lot Settings	The test program can use lot information to determine how to execute tests. For example, lot settings might dictate which steps execute, what temperature to apply to a DUT, what voltage to use, and so on. You can customize how ECU Software Toolkit obtains the settings.	

Component	Description	
	Select ECU Toolkit » Configure Lot in the TestStand Sequence Editor or click the Configure Lot button in the default ECU Software Toolkit operator interface to launch the Configure Lot Settings dialog box.	
Reporting and Data Logging	You can generate ECU Software Toolkit <u>reports and data logs</u> , such as <u>Test</u> <u>Results Logs</u> . You can <u>customize the filename</u> of the report or data log file.	
Test Step Debugging Tools	TestStand includes several tools for debugging sequences and related components in a <u>TestStand test program</u> and in <u>ECU Software Toolkit test</u> <u>programs</u> . Additionally, the TestStand Sequence Editor integrates with supported application development environments to debug test steps.	
Instrument Drivers	NI provides instrument drivers to configure, customize, and implement your instrument control applications. The ECU Software Toolkit has native support for multiple NI instrument drivers.	
Soft Front Panels	Most NI modular instruments include soft front panels (SFP) to allow you to quickly configure, troubleshoot, or debug your instrument or DUT. Launch the soft front panels from MAX by selecting Tools <u>Soft Front Panels</u> . You can also use InstrumentStudio, a software-based front panel application, to monitor, control, and record measurements from supported devices.	
Measurement and Automation Explorer	 The ECU Software Toolkit pin map defines the instruments required for a test. Measurement & Automation Explorer (MAX) helps you configure the instrument connected to the system. For a test program to execute properly, the instrument names in the pin map must correspond with instrument names configured in the system. MAX helps you complete the following tasks: Configure NI hardware and software and third-party IVI hardware and software View and edit instruments names in the system Create and edit channels, tasks, interfaces, scales, and virtual instruments Execute system diagnostics and run soft front panels Update NI software 	

Related information:

- TestStand Sequence Editor
- TestStand User Manager
- TestStand UI Debug Menu

Overall ECU Software Toolkit Workflow

The following figure illustrates the workflow for utilizing and developing with the ECU Software Toolkit, as well as the relationships between the software the toolkit interacts with.



InstrumentStudio to TestStand Workflow

Use the following steps as a guide for representing your test diagram and specification in InstrumentStudio and TestStand.

InstrumentStudio

- 1. Create or open your InstrumentStudio project.
- 2. Add or create a pin map.

- 3. Create your instrument soft front panels and layout.
- 4. Configure your instruments and your instrument measurement settings.
- 5. Export the measurement configuration file (.measconfig) for your instruments to TestStand.

TestStand and ECU Software Toolkit

- 1. Configure the *station model* in the Station Options dialog box.
- 2. Configure the *number of test sockets* in the Model Options dialog box.
- 3. Create your test sequence file.
- 4. Specify the pin map file and InstrumentStudio project in the <u>Test Program Editor</u>.
- 5. Add ProcessSetup and ProcessCleanup model callback sequences.
- 6. Add a *called sequence* (or *low-level sequence*) for a test item in the test specification.
- 7. Add the Create Pin Sessions step to the ProcessSetup sequence.
- 8. Add the <u>NISE Open Sessions</u> step after the <u>Create Pin Sessions</u> step in the ProcessSetup sequence if Switch Executive devices are present in the <u>pin map</u> file.
- 9. Add the <u>Close Pin Sessions</u> step to the **ProcessCleanup** sequence.
- 10. Add the <u>NISE Close Sessions</u> step after the <u>Close Pin Sessions</u> step in the ProcessCleanup sequence if Switch Executive devices are present in the <u>pin</u> <u>map</u> file.
- 11. Configure your instrument settings by calling the <u>Apply Pin Configuration</u> step within your called sequence. Pass the measurement configuration file you exported from InstrumentStudio to the Apply Pin Configuration step.
- 12. Add the appropriate ECU Toolkit <u>Measurement Control step types</u> to the called sequence to control instruments, read measurements, and publish results.
- 13. Add an <u>ECU Multi Test</u> step with its module adapter configured as Sequence to the MainSequence sequence to perform test evaluation.
- 14. Specify the called sequence as the Sequence for the <u>ECU Multi Test</u> step and edit the test items in the <u>Tests</u> tab.

Consult the <u>Switching Workflow</u> if your test specification includes Switch Executive devices or multisite programming.

Related information:

- Station Options Dialog Box
- Model Options Dialog Box

ECU Action/Multi Test Step with Sequence Adapter Workflow

Use one of the following two methods to add an <u>ECU Action</u> or <u>ECU Multi Test</u> step to your test program and configure its module adapter as Sequence.

Method One: Directly Adding an ECU Multi Test or ECU Action Step

- 1. Launch the Adapter Configuration window by navigating to **Configure** » Adapters...
- 2. In the Adapter Configuration window, uncheck the **Hidden** checkbox next to the **Sequence** entry in the **Adapter** column.

Adapter Configuration		×
Adapter	Selected	Hidden
L LabVIEW	۲	
C LabWindows/CVI	0	
D C/C++ DLL	0	
.N.NET	0	
X ActiveX/COM	0	
Sequence	0	
Ø <none></none>	0	
N LabVIEW NXG	0	
P Python	0	
Configure		
Help		Done

3. Select the **Sequence** entry in the TestStand **Insertion Palette** drop-down list.

Insertion Palette	
Step Types	
L LabVIEW -]
LabVIEW	
C LabWindows/CVI	
D C/C++ DLL	
.N.INET	
X ActiveX/COM	
\Xi Sequence	
Ø <none></none>	_
P Python	"

4. Insert an <u>ECU Action</u> or <u>ECU Multi Test</u> step by double-clicking the appropriate step type in the **Step Types** menu.

Insertion Palette		
Step Types		
E Sequence 🔹		
⊟ ECU Toolkit		
ECU Action		
ECU Multi Test		

The resulting inserted step should look like the following example of an <u>ECU Multi</u> <u>Test</u> step.

Step Settings for ECU Multi Test		
Hodule Tests Options	Properties	
General Run Options Looping	Name: ECU Multi Test	
Post Actions Switching Synchronization	Type: NI_ECUToolkit_MultiTest	
Expressions Preconditions Requirements	Adapter:	
Additional Results Property Browser	Icon: CAdapter Icon>	

Method Two: Modifying a Sequence Call Step

1. Insert a Sequence Call step by double-clicking the Sequence Call step type in the TestStand Insertion Palette.

2. Navigate to the **Properties** tab in the Sequence Call **Step Settings**. Step Settings for SequenceCall

E Module Propertie	S	
→ General	Name:	
Run Options	SequenceCall	
Post Actions	Туре:	
Switching	SequenceCall	•
Expressions	Adapter:	
Preconditions	Sequence	~
Additional Results	Icon:	
Property Browser	SeqAdp.ico	~

3. Click **Change Step Type...** and select **ECU Toolkit** » **ECU Action** or **ECU Multi Test** depending on the step type you want to insert.

Step Sett	ings for	SequenceCall
Module	Properties	

 General Run Options 	Name: SequenceCall	Description:	
Looping Post Actions	Type:		
Switching	SequenceCall	ECU Toolkit	🕨 🔚 ECU Action
Synchronization Expressions Preconditions Requirements Additional Results Property Browser	Adapter:	Action Tests	ECU Multi Test
	Icon:	TP Files	

4. Changing the Sequence Call step type to <u>ECU Action</u> or <u>ECU Multi Test</u> adds the <u>Tests</u> and <u>Options</u> tabs to the step.

Step Settings	for	SequenceCall
---------------	-----	--------------

🗄 Module	Tests	Options	Properties
→ General Run Options Looping Post Actions Switching Synchronization			Name: SequenceCall
			Type: NI_ECUToolkit_MultiTest
Expression Precondition Requirement	ns ons ents		Adapter:
Additional Results Property Browser			Icon: CAdapter Icon>

Subsequence as Code Module Design Recommendations

NI recommends that users abide by the following test design recommendations when

creating an ECU Software Toolkit test sequence:

- The top level of your test program should be either an <u>ECU Action</u> step or an <u>ECU</u> <u>Multi Test</u> step, or a custom step derived from one of these steps, with its module adapter configured as Sequence.
- Users should use the steps provided by TestStand and the ECU Software Toolkit to implement their called sequence.
- All <u>Measurement Configuration group</u> step types must be placed within the called sequence, with the exception of the <u>Create Pin Sessions</u>, <u>Close Pin Sessions</u>, <u>NISE -</u> <u>Open Sessions</u>, and <u>NISE Close Sessions</u> steps.
- <u>ECU Action</u> and <u>ECU Multi Test</u> steps cannot be added within the called sequence. Users should use TestStand Action steps within the called sequence, and do all test evaluations in the <u>ECU Multi Test</u> step in the top-level sequence.
- The called sequence must obtain the PinMapContext from the <u>ECU Action</u> or <u>ECU Multi Test</u> step in the top-level sequence. To facilitate this, the called sequence must have a parameter of Object Reference type that NI recommends renaming to PinMapContext.
- The called sequence can contain optional parameters depending on the information the called sequence needs from the <u>ECU Action</u> or <u>ECU Multi Test</u> step in the top-level sequence. For example, called sequences typically contain a parameter of String or Array of String type that is used for passing DUT pin names from the top-level sequence to the called sequence.
- Use TestStand variables to pass data between steps in the same called sequence. To pass data from one <u>ECU Action</u> or <u>ECU Multi Test</u> step to another, use **Export Data To** in the **Step Settings** <u>Tests</u> tab.

Exploring a Basic ECU Toolkit Test Program

Complete the following steps to open an existing sequence file and configure it to create a basic ECU test program.

Note Completed solution files can be found in the <TestStand Public>\Tutorial\NI_ECUToolkit\Basic Test Program\ Solution directory.

1. Open <TestStand Public>\Tutorial\NI_ECUToolkit\Basic Test

Program\Basic Test Program.seq. This sequence file contains the
following sequences:

- a. MainSequence—Contains a single test step, Pin Impedance Test, which performs a pin impedance test on all A pins.
- b. PinImpedanceTest_Core—Sequence which serves as the module called by the Pin Impedance Test step in MainSequence. This sequence uses <u>measurement</u> <u>control steps</u> to configure instruments, perform measurements, and publish measurement data to the Pin Impedance Test step in MainSequence.
- c. ProcessSetup—Initializes instruments and instrument sessions.
- d. ProcessCleanup—Closes instrument sessions.
- 2. Specify a <u>pin map file</u> to define the instrumentation on the tester, define the pins on the DUT, and define how DUT pins are connected to tester instrumentation for each site.
 - a. Select ECU Toolkit <u>»</u> Edit Test Program: Basic Test Program.seq or click the Edit Test Program = button on the <u>ECU Toolkit toolbar</u> to launch the <u>Test</u> <u>Program Editor</u> for the sequence file.
 - b. Select the <u>Pin Map panel</u> and enter Basic Test Program.pinmap in the **Pin Map File Path** textbox. The filename you enter is a relative path from the sequence file to the pin map file, which in this case is located in the same directory as the sequence file.
 - c. Click the Edit Pin Map File 🛤 button located to the right of the Pin Map File Path display to open the Basic Test Program.pinmap file in the Pin Map Editor.
 - d. Review the pin map. Click **OK** to close the <u>Pin Map Editor</u>.
 - e. Select the InstrumentStudio Project panel and enter Basic Test Program.instudioproj in the File Path textbox. The filename you enter is a relative path from the sequence file to the InstrumentStudio project file, which in this case is located in the same directory as the sequence file.
 - f. Click **OK** to close the <u>Test Program Editor</u>.
- 3. Select the MainSequence sequence in the sequence file. This sequence contains the Pin Impedance Test step, which is an instance of the <u>ECU Multi Test</u> step type and uses the Sequence adapter.
- 4. Select the Pin Impedance Test step and select the Tests tab of the Step Settings pane. You can use the Tests tab to define tests for individual pins or pin groups. You can use ECU Toolkit <u>measurement control steps</u> that refer to pin or pin group names in the called sequence without needing to know how each pin is connected to an instrument.

- 5. Create a <u>test program configuration</u> and specify a <u>test limits file</u> to load limit values into <u>ECU Multi Test</u> step tests at run time. You can use the same test program with multiple configurations that specify different limits for the tests.
 - a. Select ECU Toolkit » Edit Test Program: Basic Test Program.seq or click the Edit Test Program = button on the ECU Toolkit toolbar to launch the Test Program Editor for the sequence file.
 - b. Select the <u>Test Limits Files panel</u> and click the **Add Test Limits File** button to add a new test limits file reference to the test program.
 - c. Enter Production Limits as the name to identify the new test limits file in the test program.
 - d. Enter ProductionLimits.txt in the Test Limits File Path textbox. The filename you enter is a relative path from the sequence file to the InstrumentStudio project file, which in this case is located in the same directory as the sequence file.
 - e. Select the <u>Configurations panel</u> and click the **Add Configurations** button to add a new configuration to the test program.
 - f. Enter Production as the name of the configuration.
 - g. Select the newly created **Production** panel. Each configuration you specify uses a corresponding <u>Configuration panel</u> that contains a table of the test program conditions and fields for configuring the <u>test limits file</u> for the configuration.
 - h. Select Production Limits in the Test Limits File listbox.
- 6. Specify a sequence for the Pin Impedance Test step.
 - a. Select the MainSequence sequence and Pin Impedance Test step.
 - b. Select the Module tab on the Step Settings pane.
 - c. Click the **Browse** button located to the right of the **File Pathname** textbox.
 - d. Browse to <TestStand Public>\Tutorial\NI_ECUToolkit\Basic
 Test Program\Basic Test Program.seq and click Open. Select the
 Use a Relative Path for the File you Selected option when prompted, and click
 OK.
 - e. Select PinImpedanceTest_Core from the **Sequence** drop-down listbox.
 - f. In the Sequence Parameter Table, configure the following parameter values:

Parameter	Value
Pin Map Context	Step.PinMapContext
Pins	{"A","B","C","D","E","F"}

- 7. Specify a measurement configuration file for the <u>Apply Pin Configuration</u> step in the PinImpedanceTest_Core sequence called by the Pin Impedance Test step.
 - a. Select the PinImpedanceTest_Core sequence and the <u>Apply Pin Configuration</u> step.
 - b. Click the <u>Step Settings tab</u> on the Step Settings pane.
 - c. Click the **Browse** button located to the right of the **Measurement Configuration Path** textbox.
 - d. Browse to <TestStand Public>\Tutorial\NI_ECUToolkit\Basic
 Test Program\BasicTestProgram.measconfig and click Open.
 Select the Use a Relative Path for the File you Selected option when
 prompted, and click OK.
 - e. In the **Parameter Table**, configure the following parameter values:

NAME	VALUE
Pin Map Context	Parameters.PinMapContext
Pins to Configure	Parameters.Pins

- 8. Save the sequence file.
- 9. Configure a lot to run.
 - a. Select **ECU Toolkit** <u>»</u> **Configure Lot** or click the **Configure Lot *** button on the <u>ECU Toolkit toolbar</u> to launch the <u>Configure Lot Settings</u> dialog box.
 - b. Select Production in the Test Program Configuration option.
 - c. Click **OK** to close the <u>Configure Lot Settings</u> dialog box.

Note You can also use the **Active Configuration** drop-down menu on the <u>ECU Toolkit toolbar</u> to set the configuration.

Related information:

• TestStand Directory Structure

Example Programs

Use ECU Software Toolkit example programs, located in the <TestStand Public>\Examples\NI_ECUToolkit directory, to learn more about how to use the ECU Software Toolkit in an ECU test program.

Related information:

• TestStand Directory Structure

Getting Started With ECU Toolkit

This example demonstrates several features of the ECU Software Toolkit in a test program that takes common measurements with the ECU Toolkit <u>Measurement</u> <u>Control</u> step types to test a simulated ECU. This example can serve as a starting point for your test programs.

This example can be located at <TestStand Public>\Examples\ NI_ECUToolkit\Getting Started with ECU Toolkit.seq.

Highlighted Features

- Pin map
- Multisites
- Limits files
- Virtual pins

Prerequisites

You must have the ECU Test System Software Suite 22.1 or later installed.

Note You can view the example program in the TestStand Sequence Editor without the required NI instrument drivers installed, but you cannot run the example program without the required drivers installed because you do not have the same system configuration.

Pin Map

Select ECU Toolkit » Pin Map File or click the Edit Pin Map File : button on the ECU Software Toolkit toolbar to open the Getting Started with ECU Toolkit pin map file in the Pin Map Editor. The pin map file defines the following information:

- Four NI Switch Executive virtual devices named J5_MATRIX_VD, J7_MATRIX_VD, J12-FIU-1-VD, and J12-FIU-2-VD.
 - J5_MATRIX_VD is created with a Matrix Switch named J5_MATRIX, which is used to build the connections between the instrument channels and DUT pins on Site 0.
 - J7_MATRIX_VD is created with a Matrix Switch named J7_MATRIX, which is used to build the connections between the instrument channels and DUT pins on Site 1.
 - J12-FIU-1-VD is created with an SLSC FIU device named J12-FIU-1, which is used to build the connection between the J24_PPS_1 channel and the DUT pin LOAD on Site 0.
 - J12-FIU-2-VD is created with an SLSC FIU device named J12-FIU-2, which is used to build the connection between the J24_PPS_2 channel and the DUT pin LOAD on Site 1.
- Two NI-DCPower instruments named J18_SMU and J20_SMU.

Note These instruments belong to two different groups, so they can be controlled by different NI-DCPower sessions. Each site has an independent NI-DCPower instrument.

- J18_SMU can be connected to the DUT pins on Site 0 using J5_MATRIX_VD.
- J20_SMU can be connected to the DUT pins on Site 1 using J7_MATRIX_VD.
- One NI-DMM instrument named J5_DMM which is shared across two sites.
 - J5_DMM can be connected to the DUT pins on Site 0 through J5_MATRIX_VD, and connected to the DUT pins on Site 1 through J7_MATRIX_VD.
- Two NI-SCOPE instruments named J22_SCOPE_1 and J22_SCOPE_2.

Note These instruments belong to two different groups, so they can be controlled by different NI-SCOPE sessions. Each site has an independent NI-SCOPE instrument.

- J22_SCOPE_1 can be connected to the DUT pins on Site 0 using J5_MATRIX_VD.
- J22_SCOPE_2 can be connected to the DUT pins on Site 1 using J7_MATRIX_VD.
- Four NI-RMX power supplies named J24_PPS_1, J24_PPS_2, J24_PPS_3, and J24_PPS_4.

- J24_PPS_3 is connected directly to the DUT pin POWER on Site 0.
- J24_PPS_4 is connected directly to the DUT pin POWER on Site 1.
- Eleven DUT pins named POWER, A_SMU, A_DMM, B, C, D, E_SMU, E_SCOPE, F_SMU, F_SCOPE, and LOAD are defined in the pin map.
 - The defined DUT pins A_SMU and A_DMM are virtual pins that refer to a single physical DUT pin A. They are used to build connections between the physical DUT pin A and NI-DCPower or NI-DMM instruments using Switch Executive devices.
 - The defined DUT pins E_SMU and E_SCOPE are virtual pins that refer to a single physical DUT pin E. They are used to build connections between the physical DUT pin E and NI-DCPower or NI-SCOPE instruments using Switch Executive devices.
 - The defined DUT pins F_SMU and F_SCOPE are virtual pins that refer to a single physical DUT pin F. They are used to build connections between the physical DUT pin F and NI-DCPower or NI-SCOPE instruments using Switch Executive devices.
- Two pin groups named AnalogPins and DigitalPins.
- Two sites on the tester.
- A series of connections for each site. Each connection specifies a DUT pin, a site number, an instrument, and an instrument channel. There are two types of connections in this pin map file:
 - **Connection**—A connection between a DUT pin and an instrument channel on one or more sites.
 - Switch Executive Connection—A switched connection between a DUT pin and an instrument channel for one site through a Switch Executive virtual device. You must specify a Switch Executive virtual device for Switch Executive Connections.

Test Program Configurations

Complete the following steps to review the Test Program Configurations that this program uses.

- 1. Select ECU Toolkit » Edit Test Program: Getting Started with ECU Toolkit.seq or click the Edit Test Program ≩ button on the ECU Software Toolkit toolbar.
- 2. Select the <u>Configuration</u> panel to review the test program configuration.

MainSequence, ProcessSetup, and ProcessCleanup Sequences

Complete the following steps to review the MainSequence, ProcessSetup, and ProcessCleanup sequences.

- 1. On the Sequences pane, select the MainSequence sequence and review the objectives each step performs. You can also review the low-level sequence called by each step. The module adapter for each step listed below is configured as Sequence:
 - a. The Pin Impedance Test step runs a pin impedance test on all pins in the AnalogPins and DigitalPins pin groups. This step is an instance of the <u>ECU Multi</u> <u>Test</u> step type.
 - b. The Power On ECU step turns on power to the DUT. This step is an instance of the ECU Action step type.
 - c. The Pin A 5V Check step verifies that the output voltage on DUT pin A is 5V. This step is an instance of the <u>ECU Multi Test</u> step type.
 - d. The Pin LOAD Current Check step tests the current value on the path from an RMX-410x power supply and DUT pin LOAD. This step is an instance of the <u>ECU Multi Test</u> step type.
 - e. The Pin E and F Signal Check step tests the amplitude, frequency, and crosspoint voltage characteristics of the signals on DUT pins E and F. This step is an instance of the <u>ECU Multi Test</u> step type.
 - f. The Power Off ECU step turns off power to the DUT. This step is an instance of the ECU Action step type.
- 2. On the Sequences pane, select the ProcessSetup sequence. TestStand calls this sequence once before starting testing. The steps in this sequence initialize the instruments and store the instrument sessions in the Pin Map Context object.
- 3. On the Sequences pane, select the ProcessCleanup sequence. TestStand calls this sequence once after testing completes. The steps in this sequence close instrument sessions and reset the instruments.

External Limits File

Use a text editor or spreadsheet software to open and review the test limits file for the Production configuration of this test program, located at <TestStand Public>\Examples\NI_ECUToolkit\Getting Started with ECU Toolkit\ProductionLimits.txt. The test limits file is loaded at run time based on the current test configuration, and specifies the values to use to evaluate whether a measurement passes or fails. The test program stores the limits loaded from the test limits file with the results.

Switch Executive Virtual Device Configuration Files

Use a text editor or spreadsheet software to open and review the virtual device configuration files located in the <TestStand Public>\Examples\ NI_ECUToolkit\Getting Started with ECU Toolkit\Switch Executive Virtual Device Configuration folder. There are four files in this folder, which correspond to the NI Switch Executive virtual devices J5_MATRIX_VD, J7_MATRIX_VD, J12-FIU-1-VD, and J12-FIU-2-VD. These files contain the defined alias for the Channel IVI name, as well as route and route group information. The route and route group names used in this test program follow the recommended naming scheme.

InstrumentStudio Files

Complete the following steps to review the files related to InstrumentStudio:

- 1. Select ECU Toolkit » Edit Test Program: Getting Started with ECU Toolkit.seq or click the Edit Test Program = button on the ECU Software Toolkit toolbar.
- 2. Select the <u>InstrumentStudio Project</u> panel to see the specified InstrumentStudio project file name and path.
- 3. Select ECU Toolkit » Launch InstrumentStudio or click the Launch InstrumentStudio III button on the ECU Software Toolkit toolbar to open the specified InstrumentStudio project file.
- 4. See that the Getting Started with ECU Toolkit.instudioproj file includes the Getting Started with ECU Toolkit.pinmap and Getting Started with ECU Toolkit.sfp files.
- 5. Double-click Getting Started with ECU Toolkit.sfp to see the soft front panels. The layout in this file consists of one large Oscilloscope panel, one small SMU/Power Supply panel, one small Digital Multimeter panel, and two small RMX Power Supply panels. You can review the device configuration from these soft front panels.
- 6. Use a text editor to open and review the measurement configuration file exported from InstrumentStudio, located at <TestStand Public>\Examples\ NI_ECUToolkit\Getting Started with ECU Toolkit\ GettingStartedWithECUToolkit.measconfig. The measurement

configuration file is used by the <u>Apply Pin Configuration</u> step and <u>Scope - Read</u> <u>Measurements</u> step.

Related information:

- TestStand Directory Structure
- TestStand Sequence Editor

Test Steps and Flow

Use the <u>ECU Multi Test</u> step type along with <u>ECU Toolkit Measurement Control</u> step types and standard TestStand flow control tools to control the operation of an ECU Software Toolkit test sequence.

A test step is an instance of the ECU Multi Test step type or a custom step type based on the ECU Multi Test step type that performs one or more parametric or functional tests. A test step calls a sequence implemented with TestStand or ECU Toolkit Measurement Control step types to control instrumentation on the tester, take measurements from the DUT, and pass measurement values back to the ECU Multi Test step. The step performs parametric and functional tests using the measurement values obtained from the code module. The Main sequence of a test program sequence file defines the test steps to execute and the order in which to execute them. You can use standard TestStand control flow features, such as preconditions and Flow Control step types, to control the test flow.

To skip subsequent tests when a test fails, enable the **Stop Performing Tests after First Failure** option on the <u>Options</u> tab of the ECU Multi Test step in the sequence editor and also set the **On Step Failure** sequence option to **Goto Cleanup** on the **General** tab in the Sequence Properties dialog box. Skipping a test step execution skips all tests associated with the test step.

Note You cannot use multiple ECU Multi Test steps or ECU Action steps configured to use multiple threads in While loops, in Do While loops, or in For loops that use the Custom Loop option when performing multisite testing. The steps report a run-time error in these situations. Use other types of loops instead, such as For loops that use the Fixed Number of Iterations option.

Related information:

- Preconditions Panel Step Settings Pane
- Flow Control Step Types
- General Tab Sequence Properties Dialog Box
- <u>Sequence Properties Dialog Box</u>
- While Step
- Do While Step
- For Step
- For Loop Edit Tab

Recommended ECU Software Toolkit Test Program Structure and Filenames

Use the following directory and file naming recommendations to organize test programs and associated files. A consistent structure can make the test program easier to maintain and <u>deploy</u>.

Directory	 Create a separate directory that contains the following files and subdirectories for each test program: InstrumentStudio project file Pin Maps—Contains <u>pin map files</u>. SFP Files—Contains InstrumentStudio soft front panel files. Measurement Configuration Files—Contains measurement configuration files exported from InstrumentStudio. Test program sequence file Switch Executive Virtual Device Configuration Files—Contains the exported NI Switch Executive Virtual Device files for deployment or configuration. Code Module—Contains tests and other necessary program-specific code. Limits—Contains the limits files for each test program configuration.
Filenames	For each test program, use a unique filename that identifies the part the test program

tests. For the corresponding pin map, use the same base filename as the test program sequence file. For example, use <code>Part123.seq</code> for a test program sequence file and <code>Part123.pinmap</code> for the associated pin map.

Mapping DUT Pins to Instrument Channels

ECU Software Toolkit test programs communicate information between tester instrumentation and DUTs by using a pin map which associates specific DUT pins with instrument channels and sessions.

An ECU test program must communicate information from the tester instrumentation to the DUT to which the instrumentation is connected. To handle this communication in a test program, test engineers must consider the following requirements for instrument sessions and channels for all resources in the test system:

- Develop test program code that uses the names of actual connections on the DUT (DUT pins) to refer to channels on the instrument.
- Manage a large number of instrument channels.
- Scale test code to test multiple sites in parallel to improve tester efficiency.
- Support multiple types of instruments because different instruments might use channels and sessions in different ways.

Typically, instrument driver software provides test engineers with software tools for communicating with the DUT in terms of the instrument channels and sessions. However, instrument drivers have no information about the actual pins on the DUT. To develop test code that uses DUT pin names, a test engineer must use a pin map to associate each DUT pin name with the name, channel, and session of the instrument connected to that particular DUT pin.

ECU Software Toolkit Implementation

Use the <u>Pin Map Editor</u> to view, create, modify, and save pin map files. The <u>pin map</u> <u>XML schema</u> defines the structure for a pin map file. Use the **Pin Map File Path** textbox on the <u>Pin Map</u> panel of the <u>Test Program Editor</u> to specify the pin map file to use with the test program. The pin map file can support <u>multiple test sites</u> and multiple instrument types.

The <u>ECU Multi Test</u> and <u>ECU Action</u> steps create the PinMapContext object that you can pass to a called sequence implemented using <u>measurement control</u> steps. The PinMapContext object describes a set of pins relays, sites, and instruments on a test system.

Natively Supported Instruments

The ECU Software Toolkit natively supports the following types of instruments:

NI Instrument	Pin Map Instrument Type ID	Instrument Driver and Pin Map API Support	Preferred Method of Interacting with Instrument
PXI Programmable Power Supply or PXI Source-Measure Unit (SMU)	niDCPower	NI-DCPower	InstrumentStudio
PXI Digital Multimeter (DMM)	niDmm	NI-DMM	InstrumentStudio
PXI Arbitrary Waveform and Function Generator	niFGen	NI-FGEN	InstrumentStudio
PXI High-Speed Digitizer	niScope	NI-SCOPE	InstrumentStudio
NI Switch Executive Virtual Device	niSwitchExecutiveVirtualDevice	NI Switch Executive	NI MAX
NI RMX-410x Power Supply Model- Based Instruments	RMX410xPowerSupplyTypeId	RMX-410x Power Supply driver	InstrumentStudio



- Consider getting instrument names to use in ECU Test programs from the System Components <u>»</u> General Information section of the ECU Test System Maintenance Software Report or from NI MAX.
- Names for NI instruments in the pin map file are not case sensitive.

Pin Map File XML Structure

The pin map XML schema defines the following structure for a pin map XML file:

I	Legend
Root element	
Element	
Attribute	

PinMap

- AschemaVersion—Specifies the version of the schema file.
- Instruments—Specifies the type of instruments connected to the tester, the name of each instrument, and the number of channels available for each instrument.

Note

- Consider getting instrument names to use in ECU Test programs from the System Components » General Information section of the ECU Test System Maintenance Software Report or from NI MAX.
- Names for NI instruments in the pin map file are not case sensitive.
- **(ENIDCPowerInstrument—Defines an NI-DCPower instrument.**
 - Aname–Name of the instrument, as defined in MAX.
 - AnumberOfChannels—Number of channels available on the instrument.
 - ChannelGroup—Defines a group of channels controlled by one session. By grouping channels into a single session, you can avoid using session loops in code modules. By default, the Pin Map Editor creates one channel group containing all instrument channels. To create multiple, custom groups, use a unique name for the set of instrument channels for which

you want to create a session. Note that channels within a group do not have to be from the same NI-DCPower instrument. Refer to the NI-DCPower Help for information about independent channels.

- Aname—Name of a group of channels. Group names are case sensitive.
- Achannels—Channel(s) that are assigned to a group. If not defined, the ECU Software Toolkit will assign all channels from the instrument. Channels can be defined as a comma-separated list (e.g., 0,1,3,..,n), a continuous range (e.g., 0:3), or as a combination of the two (e.g., 0:1,3). All channels from an instrument must be assigned to a group and a channel cannot be in multiple groups.
- **(E)**NIDAQmxTask—Defines an NI-DAQmx task, not an instrument.
 - Aname—Name of the task, as defined in test program code modules.
 - AtaskType—Category of the task. Pin queries that return tasks of more than one task type return an error.
 - AchannelList—List of physical channels associated with the task.
- IDmmInstrument—Defines an NI-DMM instrument. NI-DMM instruments define a single channel, displayed within the ECU Software Toolkit as channel 0.
 - Aname—Name of the instrument, as defined in MAX.
- **(ENIFGenInstrument—Defines an NI-FGEN instrument.**
 - Aname—Name of the instrument, as defined in MAX.
 - AnumberOfChannels—Number of channels available on the instrument.
- **(ENIScopeInstrument—Defines an NI-SCOPE instrument.**
 - Aname—Name of the instrument, as defined in MAX.
 - AnumberOfChannels—Number of channels available on the instrument.
 - Agroup—Name of the group that contains the instrument. Group names are case sensitive. By default, the <u>Pin Map Editor</u> sets this attribute to Scope when you add NI-SCOPE instruments to the pin map file. Group names are case sensitive. By using the same group name for all NI-SCOPE instruments, the ECU Software Toolkit combines all instruments into a single session so you can avoid session loops in code modules. To create multiple NI-SCOPE sessions, use a unique name for each set of instruments for which you want to create a session. Refer to the **NI-SCOPE Help** for information about hardware limitations that prevent certain instruments from operating together as a single instrument.
- **(E)**NIRelayDriverModule—Defines a PXI-2567 relay driver module.

- Aname—Name of the relay driver module, as defined in MAX.
- AnumberOfControlLines—Number of control lines available on the relay driver module.
- Instrument—Defines an instrument that the ECU Software Toolkit does not natively support.
 - Aname—String that identifies the instrument. For instruments that NI provides but that the ECU Software Toolkit does not natively support, specify the name of the instrument, as defined in MAX.
 - AinstrumentTypeId—String that identifies the instrument type, family, class, or product group. You cannot specify a value that begins with ni. This value is a string that you define in the pin map and is not a predefined value from some other source, such as a name in MAX, that you select. Use this value to identify all instances of a particular instrument type. Instruments of the same type typically have the same session data type and same driver API.
 - ChannelGroup—Defines a synchronized group of channels. Specify individual Channel elements with unique IDs within the channel group.
 - (Aid—Unique ID for the channel group. An instrument cannot contain more than one channel group with the same ID.
 - Channel—Channel within the channel group.
 - Aid—Unique ID for the channel. An instrument cannot contain more than one channel with the same ID.
 - Channel—Channel on the instrument.
 - (A)id—Unique ID for the channel. An instrument cannot contain more than one channel with the same ID.
- - Aname—Unique string that identifies the instrument.
 - AinstrumentModel—Installed model description files in the model library.
 - Acategory—String that specifies the category to which the instrument belongs. The instrument model description defines the category for the instrument model.
 - Asubcategory—(Optional) String that specifies the subcategory to which the instrument belongs. The instrument model description defines the subcategory for the instrument model.
 - Resource—(Optional) If required by the model description file, specifies the instrument resource name in Measurement & Automation Explorer (MAX).

- An owner—Specifies the instrument resource in the model description file to which the attribute values of the <UserData> element apply.
- IserData—Contains the properties from the model description file you can assign.
 - ApropertyName—Name of the property defined in the model description file.
 - ApropertyValue—Value you assign to the property.
- - Aname—Name of the Switch Executive virtual device, as defined in MAX.
 - MultiplexerTypeId—(Optional) String that identifies the switch type, family, class, or product group. You cannot specify a value that begins with ni. This value is a string that you define in the pin map and is not a predefined value from some other source, such as a name in MAX, that you select. Use this value to identify all instances of a particular switch type. Switches of the same type typically have the same session data type and same driver API.
- ISwitchExecutiveVirtualDevice—Defines a switching instrument to use as a Switch Executive virtual device on one test site or across multiple test sites. You can use one or multiple instruments switched across DUT pins on one or multiple test sites.
 - Aname—Name of the Switch Executive virtual device, as defined in NI MAX.
- Pins—Specifies the pins on the DUT and the pins on the tester that the test program associated with the pin map file references.
 - - Aname—String that identifies the DUT pin.
 - SystemPin—Defines a system pin, which is resource on the tester or DIB that is connected to an instrument.
 - Aname—String that identifies the system pin.
- PinGroups—Specifies named grouping of pins.
 - **(EPinGroup—Defines a group of pins that you can reference with a single name.**
 - Aname—String that identifies the group of pins.
 - ©PinReference—Specifies a pin or a group of pins within the pin group.
 - Apin—String that specifies the name of an existing pin or pin group.
- Relays—Specifies the relays on the site and the relays on the tester that the test

program associated with the pin map file references

- SiteRelay—Defines a site relay, which is a relay on the tester or DIB that is connected to a relay driver module and that is associated with one or more sites.
 - Aname—String that identifies the site relay.
 - AopenStateDisplayLabel—(Optional) A description of the connections when the relay is in the open state. This attribute is only for informational and display purposes for the Digital Pattern Editor.
 - AclosedStateDisplayLabel—(Optional) A description of the connections when the relay is in the closed state. This attribute is only for informational and display purposes for the Digital Pattern Editor.
- SystemRelay—Defines a system relay, which is a relay on the tester or DIB that is connected to a relay driver module and that is associated with all sites.
 - Aname—String that identifies the system relay.
 - AopenStateDisplayLabel—(Optional) A description of the connections when the relay is in the open state. This attribute is only for informational and display purposes for the Digital Pattern Editor.
 - AclosedStateDisplayLabel—(Optional) A description of the connections when the relay is in the closed state. This attribute is only for informational and display purposes for the Digital Pattern Editor.
- **E**RelayGroups—Specifies named grouping of relays.
 - - Aname—String that identifies the group of relays.
 - RelayReference—Specifies a relay or a group of relays within the relay group.
 - Arelay—String that specifies the name of an existing relay or relay group.
- **©**RelayConfigurations—Specifies a grouping of relay configurations.
 - **(ERelayConfiguration—Defines a relay configuration. A relay configuration is the name assigned to a set of relays and their positions.**
 - Aname—String that identifies the relay configuration.
 - **ERELAY POSITION**—Specifies a relay and its position.
 - Arelay—String that specifies the name of an existing relay or relay group.
 - Aposition—String that specifies the position of the relay or relay group.
 Valid values are Open or Closed.

- - **E**Site—Defines a site to test.
 - AsiteNumber—Number that identifies the site. Site numbers must start at 0 and be consecutive without gaps.
- Connections—Specifies mappings among pins, sites, instruments, and instrument channels.
 - Connection—Defines a connection between a DUT pin and an instrument channel.
 - Apin—Name of the DUT pin to connect. The value must match the value of the name attribute of a DUTPin element.
 - AsiteNumber—The site or group of sites associated with the connection. The value must match the value of the siteNumber attribute of one of the Site elements or it must be a comma-separated list of site numbers.
 - Ainstrument—Name of the instrument or DAQmx task to connect. The value must match the value of the name attribute of an Instrument element.
 - Achannel—ID of the instrument channel or physical channel ID of the DAQmx task to connect.
 - AdeembeddingFilePath—Path, relative to the path of the pin map file, to the S2P file for de-embedding an RF Port Module connection. You can manually specify an absolute path.
 - AdeembeddingOrientation—(Optional) Used with the deembeddingFilePath attribute to specify the orientation of the data in the S2P file relative to the port the channel attribute specifies. Valid values are Port1TowardDUT or Port2TowardDUT.
 - **•** SystemConnection—Defines a direct connection between a system pin and an instrument channel.
 - Apin—Name of the system pin to connect. The value must match the value of the name attribute of a SystemPin element.
 - Ainstrument—Name of the instrument to connect. The value must match the value of the name attribute of an Instrument element.
 - Achannel—(Optional) ID of the instrument channel to connect.
 - AdeembeddingFilePath—Path, relative to the path of the pin map file, to the S2P file for de-embedding an RF Port Module connection. You can manually specify an absolute path.
 - AdeembeddingOrientation—(Optional) Used with the deembeddingFilePath attribute to specify the orientation of the data in the

S2P file relative to the port the channel attribute specifies. Valid values are Port1TowardDUT or Port2TowardDUT.

- - Ainstrument—Name of the instrument to connect. The value must match the value of the name attribute of an Instrument element.
 - Achannel—ID of the instrument channel to connect.
 - MultiplexedDUTPinRoute—Specifies the route required to connect a DUT pin on a specific site to the instrument channel.
 - Apin—Name of the DUT pin to connect. The value must match the value of the name attribute of a DUTPin element.
 - AsiteNumber—Site for the DUT pin in the system. The value must match the value of the siteNumber attribute of a Site element.
 - Amultiplexer—String that identifies the multiplexer required to create the route. The value must match the value of the name attribute of a Multiplexer element.
 - ArouteName—String that identifies the multiplexer route required to connect the pin and site to the instrument and channel.
 - AdeembeddingFilePath—Path, relative to the path of the pin map file, to the S2P file for de-embedding an RF Port Module connection. You can manually specify an absolute path.
 - AdeembeddingOrientation—(Optional) Used with the deembeddingFilePath attribute to specify the orientation of the data in the S2P file relative to the port the channel attribute specifies. Valid values are Port1TowardDUT or Port2TowardDUT.
- SwitchExecutiveConnection—Specifies a switched connection between a DUT pin and an instrument channel for one site using a Switch Executive virtual device.
 - Apin—Name of a DUT pin to connect. The value must match the name attribute value of a DUTPin element.
 - AsiteNumber—Site for the DUT pin in the system. The value must match the siteNumber attribute value of a Site element.
 - Ainstrument—Name of the instrument to connect. The value must match the name attribute value of an Instrument element.
 - Achannel—ID of the instrument channel to connect.
 - AswitchExecutiveVirtualDevice—Specifies the Switch Executive virtual device used to connect a DUT pin on a specific site to the instrument
channel.

- - Arelay—Name of the site relay to connect. The value must match the value of the name attribute of a SiteRelay element.
 - AsiteNumber—The site or group of sites associated with the connection. The value must match the value of the siteNumber attribute of one of the Site elements or it must be a comma-separated list of site numbers.
 - ArelayDriverModule—Name of the relay driver module to connect. The value must match the value of the name attribute of an NIRelayDriverModule element.
 - AcontrolLine—ID of the physical control line of the relay driver module to connect.
- SystemRelayConnection—Defines a direct connection between a system relay and a control line of a relay driver module.
 - Arelay—Name of the system relay to connect. The value must match the value of the name attribute of a SystemRelay element.
 - ArelayDriverModule—Name of the relay driver module to connect. The value must match the value of the name attribute of an NIRelayDriverModule element.
 - AcontrolLine—ID of the physical control line of the relay driver module to connect.

Common XML Validation Error Messages

If the contents of the pin map XML file do not satisfy the constraints the pin map schema defines, the ECU Software Toolkit reports error messages. Some of the error messages are generic XML validation errors and can be difficult to decipher.

Refer to the following tables to interpret certain error messages.

Error Message	The key sequence ' <item>' in 'http://www.ni.com/ TestStand/SemiconductorModule/PinMap.xsd:<element>' Keyref fails to refer to some key.</element></item>
Interpretation	The following table includes the exact meaning of the error message, which depends on the value of the Element text.

Value of Element	Meaning of Error Message
PinName	The specified pin Item does not exist in the pin n
SystemPinName	The specified system pin Item does not exist in the map.
PinOrPinGroupName	The specified pin or pin group name Item does n exist in the pin map.
RelayName	The specified relay Item does not exist in the pin
SystemRelayName	The specified system relay Item does not exist in pin map.
RelayOrRelayGroupName	The specified relay or relay group name Item doe exist in the pin map.
SiteNumber	The specified site number Item does not exist in pin map.
InstrumentName	The specified instrument Item does not exist in t map.
MultiplexerName	The specified multiplexer Item does not exist in t map.
RelayDriverModuleName	The specified relay driver module Item does not in the pin map.

Error Message	There is a duplicate key the 'http://www.ni.com/Te PinMap.xsd: <element>' key</element>	<pre>sequence '<item1> [<item2>]' for stStand/SemiconductorModule/ or unique identity constraint.</item2></item1></pre>	
	The following table includes the exact meaning of the error message, which depends on the value of the Element text.Value of ElementMeaning of Error Message		
Interpretation	PinName, AllPinNames, AllPinOrPinGroupNames, AllPinAndRelayNames	The specified pin, pin group name, relay, or relay group name Item1 is defined multiple times in the pin map.	

Value of Element	Meaning of Error Message
SiteNumber	The specified site number Item1 is defined multiple times the pin map.
InstrumentName	The specified instrument Item1 is defined multiple times the pin map.
MultiplexerName	The specified multiplexer Item1 is defined multiple times in the pin map.
RelayDriverModuleName	The specified relay driver module Item1 is defined multiple times in the pin map.
ConnectionDUTPin	The pin Item2 on site Item1 is connected to multiple instrument channels.
ConnectionSiteRelay	The relay Item2 on site Item1 is connected multiple relay driver module control lines.
SystemConnectionDUTPin	The system pin Item1 is connected to multi instrument channels.
SystemConnectionSiteRelay	The system relay Item1 is connected to multiple relay driver module control lines.
ConnectionInstrumentChannel	The channel Item2 on instrument Item1 is connected to multiple pins.
RelayConnectionModuleDriver	The control line Item2 on relay driver mod Item1 is connected to multiple relays.
MultiplexedRouteName	The multiplexer route Item1 on multiplexe Item2 has duplicate connections.
UniqueChannelAndChannelGroup	The specified channel or channel group na Item1 is defined multiple times in the pin map.

Connecting Shared Resources in the Pin Map

A shared resource is a device on the tester or DIB that is connected to an instrument or relay driver module and shared by multiple sites. To specify a connection between a shared resource and an instrument channel or relay driver module control line, use the

following criteria to decide how to define a shared resource in the pin map.

- 1. If the resource is a relay:
 - a. If there is a single relay shared by all sites, define a system relay.
 - b. If there are multiple relays in which each relay is associated with multiple sites, define a site relay.
- 2. If the resource is not a relay:
 - a. If the resource is shared by all sites and you do not need to burst patterns to the resource using an NI-Digital Pattern instrument, define a system pin.
 - b. Otherwise, define a DUT pin.

Connecting a Shared Resource using System Pins and System Relays

For each system pin and system relay in the pin map, the <u>Pin Map Editor</u> displays a row in the <u>Connections table</u>. Complete the following steps in the Pin Map Editor to create a connection for a system pin or system relay that all sites share.

- 1. Select **Connections** on the Pin Map tab.
- 2. In the View Connections for drop-down menu, select All Pins and Relays.
- 3. In the Connections table, select the instrument and channel or relay driver module and control line from the Instrument and Channel column cells of the system pin or system relay for which you want to create a connection.

Connecting a Shared Resource to One or More Sites using DUT Pins and Site Relays

For each DUT pin and site relay, the Pin Map Editor assumes there is one connection per site and displays a row for each site in the Connections table. Complete the following steps in the Pin Map Editor to create a connection for a DUT Pin or site relay that multiple sites share.

- 1. Select **Connections** on the Pin Map tab.
- 2. In the View Connections for drop-down menu, select All Pins and Relays.
- 3. In the Connections table, select the instrument and channel or relay driver module and control line from the Instrument and Channel column cells for one of the rows associated with the DUT pin or site relay.
- 4. Repeat step 3 for the remaining sites, using the same instrument channel or relay driver module control line for the sites that share the resource. The Pin Map Editor

automatically combines the rows that use the same resource into a single row. Alternatively, you can enter a comma-separated list of site numbers in the Site column to specify which sites share the resource. When you save the pin map file, the Pin Map Editor automatically removes any duplicate site connections from the Connections table.

Multiplexers and Multiplexed Connections in a Pin Map

You must define the switch and route in the pin map for the test program to access a switch route.

The pin map specifies switches as <Instruments>/<Multiplexer> elements. Each connection between an instrument channel and a DUT pin that is routed through a switch must specify a <Connections>/<MultiplexedConnection> element, as shown in the following figure.



The <MultiplexedConnection> element defines a connection between a single instrument channel and the same DUT pin on multiple sites. The <MultiplexedConnection> element also contains a list of <MultiplexedDUTPinRoute> elements that each specify the route required to connect an instrument channel to a DUT pin on a specific site.

Testing Multiple Sites in Parallel

An ECU test program might need to test multiple DUTs at the same time in parallel to

improve tester efficiency. An ECU test program can test one DUT at a time on a single test site or test multiple DUTs at a time on multiple test sites.

To implement multisite testing in a test program, the test engineer must consider the following requirements:

- The test program must be able to evaluate limits on multiple DUTs.
- The test program might execute individual test sites differently depending on previous test results specific to the individual test sites.
- The test program must be able to test DUTs on multiple sites simultaneously, even when multiple sites must simultaneously communicate with the same instrument.

ECU Software Toolkit Implementation

The TestStand **Batch** and **Parallel** process models support multisite testing by creating a test socket for running a copy of the TestStand sequence in a new execution thread. However, the default TestStand behavior does not account for difficulties that a test engineer might encounter when programming for hardware shared among multiple test sites.

When you execute tests using the Batch process model, use the **Multisite Option** dropdown menu on the <u>Options</u> tab of the ECU Multi Test or the ECU Action step to configure the following multisite execution options for the test:

- One thread per subsystem—Execute tests for each subsystem in a separate thread. A subsystem is a set of sites and system resources on the tester that operate independently of other sites and resources, typically because the sites share the same instrument, which requires the test program to test the sites together in a single thread. The ECU Multi Test step or the ECU Action step identifies subsystems by using the pin map and the pins and relays shown in the PinMapContext Pins and Relays control.
- One thread only—Execute tests for all sites in a single thread.
- One thread per site—Execute tests for each site in a separate thread. Use this option only when the code module does not use hardware shared among multiple sites.

The multisite option you select determines how many copies of a called sequence to execute. The more called sequences that execute, the fewer sites the ECU Software

Toolkit tests in any one called sequence.

Multisite Programming Techniques

When you create test programs to run on multiple sites, you must account for certain <u>subsystem considerations</u>, such as <u>instrument resources</u>, the <u>relationship between the</u> <u>subsystem and the pin map</u>, and <u>using switches</u> to share a channel between the same DUT pin on multiple sites.

Note You cannot use multiple ECU Multi Test steps or ECU Action steps configured to use multiple threads in While loops, in Do While loops, or in For loops that use the Custom Loop option when performing multisite testing. The steps report a run-time error in these situations. Use other types of loops instead, such as For loops that use the Fixed Number of Iterations option.

Related information:

- Batch Process Model
- Parallel Process Model
- While Step
- <u>Do While Step</u>
- For Step
- For Loop Edit Tab

Subsystem Considerations

Executing multiple ECU Software Toolkit test programs introduces synchronization complications if hardware is shared between test sites.

By default, the TestStand Batch process model executes a copy of the test program for each site simultaneously without changing any code by creating a test socket for each site. Each test socket runs in parallel and synchronizes at the beginning and end of the batch. However, each copy of the test program executes independently on each test socket and does not synchronize over each step within the test program execution shown in the following figure.



ECU Software Toolkit site numbers do not always directly correspond to test socket indexes.

Sharing Instrumentation Resources

Executing a copy of the test program for each socket works well when you use dedicated hardware for each DUT. In reality, however, it is more likely that multiple DUTs share the same instrumentation, which requires you to write test code to account for instrumentation sharing. In the following example, Test 1 and Test 3 share instrumentation resources.



Test 1 uses instrumentation shared between Sites 0 and 1 and between Sites 2 and 3.

Test 3 uses instrumentation shared among all four sites. In this example, Sites 0 and 1 must synchronize at the first step, and Sites 2 and 3 must also synchronize at the first step. Because Sites 0 and 1 do not have to wait on Sites 2 and 3, each site group can continue testing after the first test is done. However, all sites must synchronize for Test 3 because of the dependency on instrumentation resources. After Test 3 completes, all sites can execute in parallel.

Using instrumentation multiplexed across sites or instruments that access multiple sites simultaneously introduce the following synchronization challenges:

- **Multiplexed sharing**—You can use a multiplexer to connect a single instrument to multiple pins on the same DUT or on multiple DUTs. In this case, each site or pin must wait until an instrument is free.
- Simultaneously sharing instruments—You can connect a single instrument that supports simultaneous access to its resources to multiple pins on the same DUT or multiple DUTs, but for optimal performance you must synchronize the operations to occur at the same time. In this case, one code module can take measurements for multiple sites and report the measurements back to each test socket.
- **Relay sharing**—Run any sites that share relays in the same subsystem.

Related information:

• Batch Process Model

Subsystems and Pin Maps

The <u>ECU Multi Test</u> and the <u>ECU Action</u> steps examine the <u>pin map</u> to determine which sites to synchronize. Steps can also specify required pins or pin groups and relays, relay groups, or relay configurations for a code module.

For example, if you connect a pin to a shared instrument but a code module does not require the pin, you might be able to execute the code in parallel. After determining the sites to synchronize, the ECU Multi Test and the ECU Action steps wait until those sites reach the synchronization point or become disabled. The step then calls a sequence and passes a PinMapContext object that contains the information about the sites for the called sequence.

A subsystem is a part of the test system that can operate independently. Subsystems are defined dynamically based on the active sites, the flow of each test socket, and the instrumentation required to conduct a test.

The following figure shows two NI-SCOPE instruments for which channels must operate together and one power supply instrument that can operate four independent channels. Each DUT includes an A, B, and C pin. Each pin A and pin B connects to an NI-SCOPE instrument, and each pin C connects to a power supply.



The test setup in the previous figure includes two subsystems that can operate independently: a subsystem that contains the top half of the figure, and a subsystem that contains the bottom half of the figure. If a test needs to use only pin C, you can break the test setup into four subsystems, one for each site.

By default, the ECU Multi Test and ECU Action steps assume that a code module uses every DUT pin and no site relays. The performance of the tester can be improved by specifying only the pins that the code module uses. The following figure shows a similar situation as the previous figure but replaces the two independent NI-SCOPE instruments with two NI-SCOPE instruments that are grouped together. By default, the ECU Software Toolkit groups all NI-SCOPE instruments together.



This test setup includes a single subsystem when using any of the digital pins (A and B). For test steps that use pin A or pin B, the ECU Software Toolkit executes the test code module in a single TestStand test socket because all NI-SCOPE channels must operate together. The NI-SCOPE driver performs some operations on the channels in parallel to achieve improved performance. As in the previous example, if a test uses only pin C, you can break the test setup into four subsystems.

Multisite Programming with Switches

In test layouts where users need to share an instrument channel between multiple DUTs, an instrument channel between multiple pins on the same DUT, or multiple instrument channels on a single DUT pin, a switch can be used to properly route instrument channels to DUT pins.

When you must share a channel between the same DUT pin on multiple sites, use a switch to control which pin the device is connected to when executing a test. Some examples of this type of multisite layout are shown in the following figure.



Switches can also be used to share a single test resource between multiple channels or pins within a single **subsystem**. A **subsystem** is a part of the test system which can operate independently. The following figure shows a switching architecture for sharing a single instrument channel between multiple DUT pins on a single site.



The following figure shows a switching architecture for sharing a single DUT pin between multiple instrument channels with different instrument types within a single subsystem.



<u>Switch Executive connections</u> in the <u>Pin Map Editor</u> can be used to build the connections between instrument channels and a pin on a site. However, the <u>ECU</u> <u>Action</u> or <u>ECU Multi Test</u> step doesn't know which instrument type you are using, so it may miscalculate the number of subsystems. To address this, you need to define

virtual pins to build the connections between these instrument channels. Virtual pins are pins which do not physically exist, but that can be created in the pin map for the purpose of mapping two instrument channels to a single physical pin.

In this example, Pin A is undergoing both a pin impedance test executed by an SMU device and a signal frequency test executed by a Scope device. To capture these tests correctly in the <u>Pin Map Editor</u>, you must follow one of two procedures:

- To physically define the connection between Pin A and one instrument,
 - 1. Define Pin A in the Pin Map Editor.
 - 2. Define Pin A's connection to one instrument, such as the Scope device.
 - 3. Define a virtual pin, such as SMU_A, to build the connection between the SMU device and Pin A.
- To define all of Pin A's connections using virtual pins,
 - 1. Define two virtual pins, such as Scope_A and SMU_A, to build the connections between Pin A and the Scope and SMU channels through a Matrix device.

Switching Workflow

Use the following workflow as a guide for implementing switching in a TestStand Sequence.

The steps used can be accessed at **Measurement Configuration** <u>»</u> **Measurement Control** <u>»</u> **Switch Executive**.

ProcessSetup Sequence

Ste ►I	eps: ProcessSetup	
	STEP	DESCRIPTION
	+ Setup (0)	
	- Main (2)	
	* Create Pin Sessions	
	* NISE - Open Sessions	Creates sessions for all NI Switch Executive virtual devices in the Pin Map context and store them into the Pin Map context.
	<end group=""></end>	
	+ Cleanup (0)	

- 1. Call the Create Pin Sessions step.
- 2. Call the NISE Open Sessions step.

MainSequence Sequence

STEP	DESCRIPTION		NUM TESTS	PINS	MULTISITE OPTION	SETTING
 Setup (0) 						
- Main (1)						
Pin Impedance Tes	t Call PinImpedanceTest	_Core in Basic Test Program.seq	10	AnalogPins, DigitalPins	Per subsystem	
<end group=""></end>						
+ Cleanup (0)						
ep Settings for F	Pin Impedance 1	fest				
tep Settings for F Module Tests Optic Pathname: Ba	Pin Impedance 1 ons Properties ssic Test Program.seq	fest				
tep Settings for F Module Tests Optic le Pathname: Br C:	Pin Impedance 1 ons Properties asic Test Program seq \Users\Public\Documents\	Test National Instruments\TestStand 20)21 (64-bit)\Tutoria	al/NI_ECUTookit\Basic Test Pro	gram\Solution\Basic Test Progr	am.seq
tep Settings for F Module Tests Optic e Pathname: Bu C: equence: Pi	Pin Impedance T ons Properties asic Test Program.seq Wisers (Public Documents) nimpedance Test_Core	Test	021 (64-bit)\Tutoria	al\NI_ECUTookit\Basic Test Pro	gram\Solution\Basic Test Progr	am seq
tep Settings for F Module Tests Optic e Pathname: Bi riquence: Pi riquence Comment:	Pin Impedance T ons Properties ssic Test Program seq Wsers/Public/Documents/ himpedanceTest_Core	Test National Instruments\TestStand 20	021 (64-bit)\Tutoria	al/NI_ECUTookit/Basic Test Pro	gram\Solution\Basic Test Progr	am.seq
tep Settings for F Module Tests Optic e Pathname: Bi iquence: Pi iquence Comment: VRAMETER NAME	Pin Impedance T ons Properties sisic Test Program seq Utaem Public Documents \ Impedance Test_Core	Test National Instruments\TestStand 20	021 (64-bit)\Tutoria	a'WI_ECUTookit\Basic Test Pro	gram\Solution\Basic Test Program	am.seq COMMENT
tep Settings for F Module Tests Optic le Pathname: Ba c: squence: Pr squence Comment: VRAMETER NAME PhMapContest O	Pin Impedance 1 ona Propeties asic Test Program seq (Users V-Duble \Documents\ Impedance Test_Core TYPE Digited Reference	Test National Instruments\TestStand 20 LOG DEFA. VALUE Step PriMa)21 (64-bit)\Tutoria pContext	a'WI_ECUTookit\Basic Test Pro	gram\Solution\Basic Test Progr HOW PASSED	am.seq COMMENT

Use an ECU Multi Test or ECU Action step with its module adapter configured as Sequence as the top-level step. Subsequent steps will be called as part of a SequenceCall step within the ECU Multi Test or ECU Action step.

Called Sequence

The PinImpedanceTest_Core sequence serves as the sequence called by the top-level ECU Multi Test or ECU Action step.

Steps: PinImpedanceTest_Cor	e
STEP	DESCRIPTION
- Setup (1)	
▲ Apply Pin Configuration	
<end group=""></end>	
- Main (14)	
P For Each	Locals.PinUnderTest in Parameters.Pins
NISE - Pin to Sessions	Returns the NI Switch Executive virtual device sessions, switch routes, and new Pin Map context objects
P For Each	Locals.SiteBasedInfo in Locals.SiteBasedInfoArray
Build the connection between the siwt	ched pin and the switched instrument channel.
NISE - Connect	Connects the routes specified in the connection specification.
SMU/PPS - Enable Output	
SMU/PPS - Initiate	
CONSMU/PPS - Read Measurement	
SMU/PPS - Disable Output	
SMU/PPS - Abort	
Disconnect the switched instrument ch	annel from the switched pin.
NISE - Disconnect	Disconnects the routes specified in the disconnection specification.
O End	
O End	
<end group=""></end>	
+ Cleanup (0)	

1. Call the Apply Pin Configuration step.

- 2. For each pin, call the NISE Pin to Sessions step to get the site-based route/route group names and virtual device sessions for the switched pin and the switched instrument type.
- 3. For each site:
 - a. Call the NISE Connect step to build the connection between the switched pin and the switched instrument channel on the current site.
 - b. Call the appropriate Measurement Control group step types for the instrument you're using to read and publish measurements to the top-level ECU Multi Test SequenceCall step.
 - c. Call the NISE Disconnect step to disconnect the switched pin and the switched instrument channel on the current site.

ProcessCleanup Sequence

Steps: ProcessCleanup

Þ	- <u>}</u>	
	STEP	DESCRIPTION
	+ Setup (0)	
	- Main (2)	
	X Close Pin Sessions	
	X NISE - Close Sessions	Closes all NI Switch Executive virtual device sessions in the Pin Map context.
	<end group=""></end>	
	+ Cleanup (0)	

- 1. Call the Close Pin Sessions step.
- 2. Call the NISE Close Sessions step.

NI Switch Executive Virtual Device Configuration

Users can utilize an NI Switch Executive virtual device configuration file as a baseline for creating additional NI Switch virtual devices.

For information about creating and configuring an NI Switch Executive virtual device, refer to the NI Switch Executive Virtual Device Help Manual.

Once you have configured one site or subsystem's NI Switch Executive virtual device, you can expand that configuration to multiple sites and subsystems. NI Switch Executive supports exporting a virtual device's configuration file for configuration or deployment, and importing a configuration file for creating a virtual device. Typically, each site/subsystem has an independent matrix switch device to aggregate test time and maximize the efficiency and throughput of test systems, and matrix switch devices on multiple sites/subsystems have the same model. Because of this, you can export a virtual device configuration file for a single site or subsystem and use this file as a baseline for configuration files for other virtual devices in your test system. Once the baseline file has been edited to fit a different site or subsystem, it can be imported and used as a basis to create a new virtual device using NI Switch Executive.

When you configure an NI Switch Executive Device, you must follow the <u>route/route</u> <u>group naming scheme</u>. You can use the NISE - Pin to Sessions step type to get the required route/route group names and virtual device sessions for each site based on the switched DUT pin and switched instrument type.

Enforce Route/Route Group Naming Scheme

The NISE - Pins to Sessions step type returns the NI Switch Executive virtual device sessions, switch routes, and new Pin Map context objects required to access a switched pin and a switched instrument channel. The returned switch routes' names adhere to the following naming convention:

```
<Pin Name>_Index#-<Instrument Type ID>
```

Pin Name is the DUT pin defined in the pin map file through the <u>Pin Map Editor</u>. Index# is the index of a site in the site list included in a subsystem. The ECU Software Toolkit defines subsystems dynamically based on the active sites, the flow of each test socket, and the instrumentation required to conduct a test.

The following three exampled demonstrate how to set switch route names in NI MAX for different test configurations.

Two DUTs, Two Instruments, Two Switches

The following figure illustrates the connections between two DUTs/sites, two NI-DMM instruments, and two matrix switch instruments. Each DUT has a pin A which connects to a separate, independent NI-DMM instrument through a separate, independent matrix switch instrument. Subsystem 1's site list is $\{ \texttt{Site 0} \}$ and Subsystem 2's site list is $\{ \texttt{Site 1} \}$. The two subsystems illustrated in the figure can operate independently.



The index of Site 0 in Subsystem 1 is 0 and the index of Site 1 in Subsystem 2 is 0. So, the name of the route/route group used to establish a connection between the DUT pin A and the DMM instrument on both sites should be the same. You should create a virtual device for each switch instrument, in this case named J5_MATRIX and J7_MATRIX. Then, create a route/route group named A_0-niDMM on each of the two virtual devices.

Two DUTs, One Instrument, Two Switches

The following figure illustrates the connections between two DUTs/sites, one NI-DMM instrument, and two matrix switch instruments. Each DUT has a pin A which connects to a shared NI-DMM instrument through a separate, independent matrix switch instrument. Pin A on both DUTs are contained within a single subsystem named Subsystem 1. The site list included in Subsystem 1 is {Site 0, Site 1}. The site list will always be ordered by site.



The index of Site 0 in Subsystem 1 is 0, so you should create a virtual device named J5_MATRIX and a route/route group named A_0-niDMM. This route/route group establishes a connection between the DUT pin A on Site 0 and the J5_DMM.

The index of Site 1 in Subsystem 1 is 1, so you should create a virtual device named J7_MATRIX and a route/route group named A_1-niDMM. This route/route group establishes a connection between the DUT pin A on Site 1 and the J5_DMM.

Two DUTs, One Instrument, One Switch

The following figure illustrates the connections between two DUTs/sites, one NI-DMM instrument, and one matrix switch instrument. Each DUT has a pin A which connects to a shared NI-DMM instrument through different routes in a shared matrix switch instrument. Pin A on both DUTs are contained within a single subsystem named Subsystem 1. The site list included in Subsystem 1 is {Site 0, Site 1}.



The index of Site 0 in Subsystem 1 is 0, so you should create a virtual device named J5_MATRIX and a route/route group named A_0-niDMM. This route/route group establishes a connection between the DUT pin A on Site 0 and the J5_DMM.

The index of Site 1 in Subsystem 1 is 1, so you should create a route/route group named A_1-niDMM. This route/route group establishes a connection between the DUT pin A on Site 1 and the J5_DMM.

Exporting and Importing Test Limits with Text Files

Use *test limits* files to export and import test limits from and to your test program. Create test limits files by exporting test limits from an existing sequence file.

Use a tab-delimited <u>text file</u> to <u>export</u> and <u>import</u> test limits from and to <u>ECU Multi Test</u> steps in a single sequence file at edit time or run time. During development, export all the tests in a test program to review and <u>edit</u> and then import the changes back into the test program. At run time, load and execute a different set of test limits from separate text files based on the <u>test program configuration</u> you select by exporting the test limits and creating multiple copies of the file to edit for each unique set of test limits you want to use. To protect the test limits files from viewing or editing when deploying a test program, <u>embed the test limits in the test program sequence file</u>.

Select **ECU Toolkit** <u>»</u> **Edit Test Program** and select **Test Limits Files** in the <u>Test Program</u> <u>Editor</u> to specify one or more test limits files to make available to the test program configurations. The test program configuration specifies the test limits file that loads before running a test lot.

Create a test limits file by selecting ECU Toolkit » Export Test Limits from or by clicking the Export Test Limits from button s on the ECU Software Toolkit toolbar to export test limits from a sequence file into a tab-delimited test limits text file. Import a test limits file by selecting ECU Toolkit » Import Test Limits to or by clicking the Import Test Limits to button s on the ECU Software Toolkit toolbar to import test limits from a tab-delimited test limits text file into a sequence file. When you import test limits from a text file, you can update limits in matching tests or replace all tests in matching steps.

Exporting Test Limits from Sequence Files

Select **ECU Toolkit** <u>**Export Test Limits from** *filename* to export test limits from all <u>test steps</u> in a sequence file to a new or existing tab-delimited test limits <u>text file</u> at edit time. Selecting an existing file overwrites the content of the file.</u>

When you export test limits, the text file includes the following test data for every sequence in the sequence file:

- Sequence name, Step name, UniqueStepId, Test name— Always included.
- Test number— If a step does not contain any tests, this field displays No Tests.

Note Use the ECU Action step instead of the ECU Multi Test step if the step does not contain any tests.

- Pin or pin group—The ECU Software Toolkit does not export the test data associated with the generated tests for the pins in the pin group.
- Low limit expression, High limit expression—For Pass/Fail tests, these fields are blank.
- Scaling factor—Included only when non-empty scaling factors exist for any test in the sequence file and scaling factors are the same within each test. The Units Prefix column of the <u>Supported Scaling factors</u> table lists values for exported scaling factors.
- Low limit scaling factor, High limit scaling factor, Data limit scaling factor—Included only when non-empty for any test in the sequence file and the data and limits scaling factors differ in at least one test in the sequence file. The Units Prefix column of the <u>Supported Scaling factors</u> table lists values for exported scaling factors.
- Evaluation comparison mode—Included only when at least one step contains nondefault value. For Pass/Fail tests, this field is blank.
- Base units—For Pass/Fail tests, this field is blank.
- Evaluation type— Always included.
- Test name expression, Test number expression, Published data ID, Test data source expression, Export data to expression—Included only when non-empty values exist for any tests in the sequence file.
- Test numeric display format—Included only when non-default test numeric display formats exist for any tests in the sequence file.

The file does not contain test data for external sequence files referenced in the sequence file from which you export the test limits.



• For <u>ECU Multi Test</u> steps in the sequence that have no tests, the file contains a row with the step information and the value No Tests for

the Test Name. All other columns are blank. Use the <u>ECU Action</u> step instead of the ECU Multi Test step if the step does not contain any tests.

- If a field name begins with a mathematical sign, Microsoft Excel might interpret the contents of the cell as a formula and return an error for the cell. To work around this issue, format the cell in Excel as text data.
- In some cases when a limit has a large number of digits, Microsoft Excel might truncate the decimal portion of the number. To work around this issue, format the columns that contain the limit numbers in Excel as text data.

Related information:

• Step.UniqueStepId

Test Limits Text File Structure

Each row in the test limits file corresponds to a test. The import mode and the columns in the test limits file define how the Import/Export Test Limits tool matches rows to the sequence file.

Tags enclosed in angle brackets denote the properties recognized during import or export. The Import/Export Test Limits tool ignores any column with an unknown tag and any text above or on the same line as the opening

<SemiconductorModuleTests> tag or below or on the same line as the required closing </SemiconductorModuleTests> tag. You can place comments above the <SemiconductorModuleTests> tag, below the

</SemiconductorModuleTests> tag, or to the right of the recognized data columns, as shown in the following table:

Column Tag	Description	Format
<sequencename></sequencename>	Name of the sequence. The tool uses this column, if present, to match each row in the test limits file to locations in the sequence file.	String
<stepname></stepname>	Name of the step. The tool uses this column, if	String

Column Tag	Description	Format
	present, to match each row in the test limits file to locations in the sequence file.	
<stepid></stepid>	Unique ID of the step automatically generated by TestStand. The tool uses this column, if present, to match each row in the test limits file to locations in the sequence file.	String
<testname></testname>	Name of the test.	String
<testnumber></testnumber>	Number of the test. Importing an empty string indicates the lack of a test number. When you enable the Update limits in matching tests option, the tool uses this column, if present, to match each row in the test limits file to locations in the sequence file. If a step does not contain any tests, this field displays No Tests.	Number
	Note Use the ECU Action step instead of the ECU Multi Test step if the step does not contain any tests.	
<pin></pin>	Name of the pin or pin group. For a pin group, test data for individual pins in the pin group are not included in the file.	String
<lowlimitexpression></lowlimitexpression>	Lower limit expression. Blank for Pass/Fail tests.	String
<highlimitexpression></highlimitexpression>	Upper limit expression. Blank for Pass/Fail tests.	String
	Scaling factor used to display the data, high limit, and low limit fields. The Units Prefix column of the <u>Supported Scaling factors</u> table lists valid values for scaling factors.	
<scalingfactor></scalingfactor>	Note If you use the <scalingfactor> column, you cannot use the individual field <lowlimitscalingfactor>, <highlimitscalingfactor>, and <datascalingfactor> columns.</datascalingfactor></highlimitscalingfactor></lowlimitscalingfactor></scalingfactor>	String
<lowlimitscalingfactor></lowlimitscalingfactor>	Scaling factor used to display the low limit field. The Units Prefix column of the <u>Supported Scaling factors</u>	String

Column Tag	Description	Format
	table lists valid values for scaling factors.	
<highlimitscalingfactor></highlimitscalingfactor>	Scaling factor used to display the high limit field. The Units Prefix column of the <u>Supported Scaling factors</u> table lists valid values for scaling factors.	String
<datascalingfactor></datascalingfactor>	Scaling factor used to display the data field. The Units Prefix column of the <u>Supported Scaling factors</u> table lists valid values for scaling factors.	String
<comparisontype></comparisontype>	Evaluation Comparison Mode. Included in file only when property contains non-default value. Blank for Pass/Fail tests.	String
<units></units>	Base units of the limits. Blank for Pass/Fail tests.	String
<evaluationtype></evaluationtype>	Evaluation type. Valid values are Pass/Fail or Numeric Limit.	String
<testnameexpression></testnameexpression>	Test name expression. Included in file only when non-empty test name expressions exist in the sequence file.	String
<testnumberexpression></testnumberexpression>	Test number expression. Included in file only when non-empty test number expressions exist in sequence file.	String
<publisheddataid></publisheddataid>	Published data ID. Included in file only when non- empty published data IDs exist in the sequence file.	String
<testdatasource></testdatasource>	Test data source expression. Included in file only when non-empty test data source expressions exist in the sequence file.	String
<exportlocation></exportlocation>	Expression for location to which to export data (Export Data To column in the Tests table). Included in file only when non-empty export data to expressions exist in the sequence file.	String
<numericformat></numericformat>	Test numeric display format that applies to the high limit, low limit, and data. During export, the ECU Software Toolkit retrieves the value from the Data property of the test. Included in file only when non- default numeric formats exist for any tests in the sequence file.	String

Related information:

• <u>Step.UniqueStepId</u>

Opening Test Limits Text Files in Microsoft Excel

Complete the following steps to open a <u>test limits text file</u> in Microsoft Excel.

- 1. In Excel, select **Open**. Select **All Files** in the drop-down menu of file types.
- 2. Select the test limits file you want to open and click the **Open** button.
- 3. In the Text Import Wizard, select **Delimited** in the **Original data type** option and click **Next**.
- 4. Select **Tab** in the **Delimiters** option and click **Next**.
- 5. Select General in the Column data format option and click Finish.

Alternatively, you can drag and drop a test limits text file into Excel to open the file.

When you save a modified test limits file that you opened or edited in Excel, Excel might return a message that the Excel file contains features that are not compatible with the tab-delimited text format. Click **Yes** in the prompt to save the workbook in text format. The Import/Export Test Limits tool is unable to parse any additional formatting information Excel adds to the file.

Note 🕈

- If a field name begins with a mathematical sign, Microsoft Excel might interpret the contents of the cell as a formula and return an error for the cell. To work around this issue, format the cell in Excel as text data.
- In some cases when a limit has a large number of digits, Microsoft Excel might truncate the decimal portion of the number. To work around this issue, format the columns that contain the limit numbers in Excel as text data.

Editing Test Limits Text Files

Use a text editor or Microsoft Excel to edit the test limits in a test limits file by removing and consolidating columns and rows within the file.

You can delete columns in the test limits text file to simplify the external limits file, but if you delete an identifier column, such as SequenceName, StepName, StepId, or TestName, the text file might then contain rows that use the same identifier string. When this situation occurs, you must consolidate duplicate rows or include another identifier to distinguish among tests. You must repeat the customizations you make to the text file each time you export the test limits.

For example, if you have several steps or tests in a sequence file with the same name or test number but you want to use the same limits for each step, you can delete the StepId column and possibly the SequenceName column to simplify the resulting text file. However, if you want to use different limits for steps or tests with the same name or test number, the StepId column and possibly the SequenceName column must remain in the text file to distinguish the tests.



- If a field name begins with a mathematical sign, Microsoft Excel might interpret the contents of the cell as a formula and return an error for the cell. To work around this issue, format the cell in Excel as text data.
- In some cases when a limit has a large number of digits, Microsoft Excel might truncate the decimal portion of the number. To work around this issue, format the columns that contain the limit numbers in Excel as text data.

Importing Test Limits from Text Files

Select **ECU Toolkit** <u>Import Test Limits into</u> *select ECU Toolkit* <u>Import Test Limits Into all test steps</u> in a sequence file at edit time.

When you import test limits from a text file, you can replace only the tests defined in the test limits file or you can delete all the tests in the sequence file and recreate each test from the contents of the test limits file.

The Import/Export Test Limits tool uses the SequenceName, StepName, and StepId columns in the test limits file to match tests in the sequence file. When you update the limits in matching tests from the test limits file, the tool also uses the TestNumber column to identify the test. If one of these identifier columns does not exist in the test limits file, the tool imports data into any step or test that matches the remaining identifiers and a row in the test limits file might update multiple locations in the sequence file. If more than one location in the sequence file matches the identifiers in a row in the test limits file, the ECU Software Toolkit imports data from that row into all the matching locations.

The Import/Export Test Limits tool returns an error and fails to import a test limits file if:

- A row in the test limits file does not match any location in the sequence file.
- A single test in the sequence file matches more than one row in the test limits file.
- A location in the sequence file does not match any row in the test limits file and the **Require every step to be in test limits file** option is enabled.
- A row in the test limits file matches an ECU Sequence Call step in the sequence file and the Test Data Source column contains a value.

The Import Test Limits into Sequence File dialog box contains the following options:

• Update limits in matching tests—Finds any test in the sequence file that matches the SequenceName, StepName, StepId, and TestNumber identifier columns in the test limits file and sets the property values in the sequence file to the corresponding values in the test limits file. If one of these identifier columns other than TestNumber does not exist in the test limits file, the tool sets the corresponding property values for any step that matches the remaining identifiers.

Note You cannot change the test number for tests in the sequence file when you select this mode. You must use the Replace all tests in matching steps option if you want to change the test number.

The tool modifies only the tests and properties that exist in the test limits file. Tests and properties defined in the sequence file but not in the test limits file remain in the sequence file unchanged. Select this option when you want to set only certain test properties at run time.

• Replace all tests in matching steps—Deletes all the tests that correspond to steps

in the test limits file and recreates each test from the contents of the test limits file by adding tests to any step that matches the SequenceName, StepName, and StepId identifier columns in the test limits file. If one of these identifier columns does not exist in the test limits file, the tool adds a test to any step that matches the remaining identifiers. For columns that do not exist in the test limits file, the tool sets the corresponding property to the default value in the sequence file. The test limits file must include the columns in which you use non-default values; otherwise, the tool uses default values for those properties in the sequence file. If any step in the test limits file has a value of **No Tests** for the Test Number, the tool deletes all tests for any matching steps.



Note Use the ECU Action step instead of the ECU Multi Test step if the step does not contain any tests.

Select this option when you want to use limit files with different numbers of tests. For example, you might want to include more tests in a normal testing mode than you include in a QA testing mode.

• **Require every step to be in test limits file**—Returns an error and does not import the file if a test or step exists in the sequence but does not exist in the limits file. If you do not enable this option, the Import/Export Test Limits tool returns a warning and imports only matching tests or steps in the limits file.

Debugging

Refer to the following information to learn about debugging your test programs.

- Debugging TestStand Test Programs
- <u>Debugging ECU Software Toolkit Test Programs</u>

Debugging TestStand Test Programs

Consider the following common approaches to debug the following components of a TestStand test program.

Sequence Execution

Sequences can be in a running or suspended testing state. Testing is running when

continuously testing DUTs. When testing is running, each test socket is executing tests for one DUT. Testing is suspended when one or more TestStand test socket executions are suspended at a breakpoint. One test socket can be suspended while other test sockets are running. The background of the Execution window changes to yellow to indicate that the execution is suspended. You can examine sequence variables only in executions that are suspended.

Use the following techniques to debug sequence execution:

- Set Data Breakpoints—To suspend testing when the value of a variable changes or when it has a specific value, use the Watch View pane to enter an expression for the variable and edit the watch expression to specify a break condition. TestStand evaluates these break conditions when each step executes and suspends the testing if the break conditions are met.



Note Delete watch expressions when you no longer need them because they can negatively affect execution performance.

- Control Test Program and Step Execution—When testing is suspended, use the following options in the Debug menu or toolbar or in the Steps pane context menu of the Execution window to control test program and step execution:
 - Debug Menu/Toolbar
 - **Step Into**—Enters and suspends inside the code module associated with the step to which the execution pointer points.
 - Step Over Executes the step to which the execution pointer points and suspends execution on the next step in the sequence.
 - **Step Out**—Resumes execution through the end of the current sequence and suspends execution on the next step in the calling sequence.
 - Steps Pane Context Menu

- Run Mode » Skip (temporarily skips step)—Select Skip to prevent the step from executing in the current execution. Changing the step run mode in an execution does not persist the setting to the step in the sequence file. As a result, once execution ends, the setting reverts to the original run mode before you made the change. Change the run mode on the step in the Sequence File window to persist the setting in the sequence file.
- Set Next Step to Cursor Moves the execution pointer to point to a different step to change which step executes next.
- Run Selected Steps—Select one or more arbitrary steps in the sequence that you want to execute before resuming normal execution. TestStand executes the selected steps in the order they appear in the sequence. If a breakpoint suspends execution of the selected steps, the Steps pane displays the interactive execution pointer, which differs from the normal execution pointer.
- View and Modify Variable and Expression Values —Use the following techniques to view or modify variable values and to monitor expression values:
 - When testing is suspended, use the Variables pane to view and modify the values of any of the variables and properties accessible in the current execution.
 - To monitor the value of an expression while testing, enter the expression in the Watch View pane when testing is suspended. You can also use the Edit Breakpoints/Watch Expressions dialog box to enter expressions when testing is not in progress. When you enable tracing, the sequence editor updates the values after each step executes.
- Display Debug Messages—To display messages to the Output pane while testing, use the OutputMessage expression function in a Statement step.

Related information:

- Debugging Executions in TestStand
- Watch View Pane
- Watch Expression Settings Dialog Box
- TestStand Sequence Editor Debug Menu
- <u>TestStand Toolbar Buttons and Shortcuts</u>
- <u>Steps Pane Context Menu Execution Window</u>
- Variables Pane Execution Window
- Edit Breakpoints/Watch Expressions Dialog Box
- Output Pane

- Output Message Function
- Statement Step

Debugging ECU Software Toolkit Test Programs

In addition to following common approaches to debug components of a <u>TestStand test</u> <u>program</u>, consider the following additional approaches to debug components of an ECU Software Toolkit test program:

Sequence Execution

Sequences can be in a running or suspended testing state. Testing is running when continuously testing DUTs. When testing is running, each test socket is executing tests for one DUT. Testing is suspended when one or more TestStand test socket executions are suspended at a breakpoint. One test socket can be suspended while other test sockets are running. The background of the Execution window changes to yellow to indicate that the execution is suspended. You can examine sequence variables only in executions that are suspended.

Pausing a lot does not suspend executions the way using the default TestStand Sequence Editor **Debug**.» **Break** or **Break All** menu items do. When you pause a lot with the **Pause** button, testing continues until each site completes testing its current DUT. The executions then wait for operator input before proceeding with testing the next batch of DUTs. To suspend test socket executions to examine variables or to step into code modules, set a breakpoint on the step where you want to suspend. You cannot examine variables in the sequence editor for executions that are not suspended, even if they are paused between DUTs.

For TestStand ECU Software steps under the <u>Measurement Configuration</u> category, you can follow the below procedure to debug ECU Software Toolkit sequence execution:

- 1. Set a breakpoint on a step.
- 2. Click the Launch InstrumentStudio III button in the toolbar. You must configure the InstrumentStudio project in ECU Toolkit » Edit Test Program before doing this.
- 3. The current TestStand sequence <u>pin map</u> will be active in the InstrumentStudio project. If you already have the soft front panel for the pin map created, doubleclick the soft front panel to open it. If a soft front panel is not created for the pin map:

- a. Right-click on the InstrumentStudio project.
- b. Select New <u>Auto-Create Layout</u>, which creates a soft front panel for the current pin map.
- 4. You will see instrument configurations in the soft front panel. You can check these configurations to see if their behavior is expected.

Reports and Data Logs

The TestStand ECU Software Toolkit generates a <u>Debug Test Results Log</u> by default. The Debug Test Results Log is a human-readable text file that contains the measurement values and test limits for each test that executes on each site. The Debug Test Results Log result processor generates separate files for each site in the test program.

Use the Debug Test Results Log to debug measurements and test configurations as a lot executes. Use the <u>Debug Test Results Log Options</u> dialog box to specify settings for the Debug Test Results Log.



- Keeping the Report pane open during execution negatively affects performance.
- The Debug Test Results Log contains only ECU test result data from <u>ECU</u> <u>Multi Test</u> steps. These data logs do not contain result data from other types of steps.
- You can configure the Debug Test Results Logs in the Result Processing dialog box to display in the Report pane in the sequence editor and in the reports dialog box in the default operator interface.

You can disable result recording by selecting the **Disable Result Recording for All Sequences** option on the **Execution** tab of the Station Options dialog box.

Related information:

- <u>Result Processing Dialog Box</u>
- Execution Tab Station Options Dialog Box

• <u>Station Options Dialog Box</u>

Debug Test Results Logs

When testing a lot, you can generate a Debug Test Results Log of human-readable text that contains the measurement values and test limits for tests on all <u>test steps</u> that execute on each site.

The Debug Test Results Log result processor generates a separate file for each site in the test program. You can use this data to debug test issues and to diagnose issues on the tester itself.

The Debug Test Results Log contains the following sections for each DUT:

- Header—Contains the Site Number, Batch Number, and Part ID for each DUT tested. The Batch Number refers to the loop iteration when testing multiple DUTs. The Debug Test Results Log result processing plug-in populates the Part ID field from the TestStand SerialNumber property, which ECU Software Toolkit sets to the values returned in the SitePartIds parameter.
- Step Results—Contains the Step Name and all its corresponding tests.
- Test Results—Contains the Test Number, Result, Test Name, Low Limit, Measurement Value, High Limit, and Units.

Modifying the Number of Results to Include

By default, the <u>Debug Test Results Log</u> includes all test results. You can change the default behavior.

Complete the following steps to log results only when a DUT fails.

- 1. Launch the <u>Debug Test Results Log Options</u> dialog box.
- 2. Enable the Log Results Only for DUT Failures option.

Complete the following steps to limit the number of results to display for the Debug Test Results Log.

- 1. Launch the <u>Debug Test Results Log Options</u> dialog box.
- 2. Enable the Limit Number of Results Displayed in Report View option.

3. Use the **Display Results for Last** option to specify the number of tests to display in the Report View for the Debug Test Results Log.

Changing Report Orientation

By default, the Debug Test Results Log uses portrait orientation. Landscape orientation uses wider columns for tests with long test numbers or test names. Complete the following steps to change the report orientation of the Debug Test Results Log.

- 1. Launch the <u>Debug Test Results Log Options</u> dialog box.
- 2. Use the drop-down listbox of the **Report Orientation** option to select **Portrait** or **Landscape**.

Customizing the Filename

When you customize the log filename, use unique filenames for each test site. The Debug Test Results Log result processing plug-in appends results from each DUT to the corresponding site log.

Logging Text Data

If you want to add data to the Debug Test Results Log that is not a measurement or test limit, you can include text data in the Debug Test Results Log by using the following techniques:

- Adding Text Data for a Step Using Additional Results—You can use the Additional Results panel of the Step Settings pane to add text data after all the measurements and test limits for the tests associated with the step. To configure the Additional Results panel to generate text data in the Debug Test Results Log, set the Name option of the result to NI.TestResultsLog and set the Value to Log option to the value you want to add to the log file. You can optionally add a text label for the data by adding it to the end of the Name field in the form NI.TestResultsLog.CustomTextLabel, where CustomTextLabel is the text you want to display as a label in the log file.
- Adding Text Data between Test Steps—If the text data you want to log is not associated with a specific step, you can use the Additional Results step type to add text data to the Debug Test Results Log. Use the Additional Results panel of the Step Settings pane as described above to add data to the log file.

Related information:

- Additional Results Panel Step Settings Pane
- Properties Tab Step Settings Pane
- Additional Results Step

Deploying ECU Software Toolkit Test Programs

Complete the following steps to use the TestStand Deployment Utility to deploy a test program.

- 1. Complete the following steps on the **Mode** tab of the deployment utility:
 - a. Select the **Deployable Image Only** option.
 - b. Select the **Create new Full Deployment** option. NI recommends that you create a new Full Deployment, save the deployment to increment the **Deployment Version**, and store the deployment in a source code control system each time you create or update a deployment.
- 2. Complete the following steps on the **System Source** tab:
 - a. Place a checkmark in the **From Directory** checkbox and specify the test program directory.
 - b. Place a checkmark in the **Include Subdirectories** checkbox.
 - c. In the **Location of Deployable Image** field, specify the location in which to save the deployable image.
- 3. Click the **Distributed Files** tab and select **Yes** to analyze the source files.
- 4. Click the **Build Status** tab to display the analysis results. Resolve any issues before completing the next step.
- 5. Complete the following steps on the Distributed Files tab:
 - a. Select the files to include in the distribution. Use the **Distributed Files** pulldown menu to filter the display.
 - b. If you have LabVIEW code in the sequence, click the **LabVIEW Options** button to launch the LabVIEW VI Options dialog box, which you use to specify LabVIEW options.
- 6. In the LabVIEW VI Options dialog box, enable the following options in the Packed Project Library Options section of the dialog box and use the default values for the other options in this dialog box.
 - Output VIs to a Packed Project Library ()
 - Copy files from vi.lib before Build

- Copy files from user.lib before Build
- Copy files from instr.lib before Build

Enabling these options increases the size of the distribution but allows the test program to more easily run in the LabVIEW Run-Time Engine (RTE).

- 7. Click **OK** in the LabVIEW VI Options dialog box.
- 8. Complete the following steps on the Distributed Files tab:
 - a. Click the **Save As** button if this is the first time you are configuring a deployment for the test program to save the distribution configuration for the deployment. Click the **Save** button when you modify an existing deployment to increment the version number of the deployment.
 - b. Click the **Build** button to build the distribution.
 - c. Click the **Save** button to auto-increment the **Deployment Version** field on the Mode tab.
- 9. Complete the remaining tasks in the Deployment Process Overview to transfer the deployment to a tester and validate the deployment.

Consider the NI recommendations for <u>installers</u>, and <u>protections for test programs and</u> <u>test limits</u> as you design and create deployments for an ECU test system.

Related information:

- <u>TestStand Deployment Utility</u>
- Deploying TestStand Systems
- Mode Tab TestStand Deployment Utility
- System Source Tab TestStand Deployment Utility
- Distributed Files Tab TestStand Deployment Utility
- Build Status Tab TestStand Deployment Utility
- LabVIEW VI Options Dialog Box
- Deployment Process Overview

Installer Settings for Deploying ECU Software Toolkit Test Programs

Use the following recommended settings to create a simple installer with the deployment utility:

• Use the **Deployment Version** option on the Mode tab of the deployment utility to
specify a version for the distribution. Increment the version with each release of the test program distribution.

- Do not enable the **Install TestStand Runtime** option or include any software in the Drivers and Components dialog box. Do not use the test program deployment to control the software installed on an ECU test station.
- Enable the **Do not Ask User for Installation Directory** option on the Installer Options tab of the deployment utility and set the **Default Installation Directory** option to install the test program in the same location on all test stations.

After you create a deployment, save the deployment specification file (.tsd) so you can use the same configuration to build newer versions of the installer or to create patches. Additionally, on the Build Status tab of the deployment utility, save the deployment build status log files to use to troubleshoot issues.

Related information:

- Installer Options Tab TestStand Deployment Utility
- Mode Tab TestStand Deployment Utility
- Drivers and Components Dialog Box
- Build Status Tab TestStand Deployment Utility

Protecting Test Programs and Test Limits from Editing and Viewing

You can password protect sequence files to deter editing and viewing sequence files in the sequence editor and in operator interfaces.

To protect test limits files, use the **Embed Test Limits File** option on the <u>Test limits</u> <u>Files</u> panel of the <u>Test Program Editor</u> to embed the external test limits files in the test program sequence file before you password-protect the test program sequence file.

Note NI recommends that you do not use passwords as the only way of protecting intellectual property.

Related information:

• Advanced Tab - Sequence File Properties Dialog Box

Environment Reference

The ECU Software Toolkit environment consists of dialog boxes and windows that allow you to build and modify various components of a project.

Refer to the following for information about the various components of the ECU Software Toolkit environment.

- ECU Software Toolkit Sequence Editor UI Configuration
- Test Program Editor
- Pin Map Editor
- ECU Software Toolkit Step Types
- <u>Dialog Boxes and Windows</u>

ECU Software Toolkit Sequence Editor UI Configuration

The ECU Software Toolkit contains a custom UI configuration for the TestStand Sequence Editor to streamline ECU test program development.

The default ECU Software Toolkit UI Configuration of the TestStand Sequence Editor includes the following changes:

- Simplified toolbar
- Modified Insertion Palette pane that displays the ECU Toolkit folder and the Action Step at the top of the Step Types list
- More detailed <u>Steps</u> pane

When you launch the ECU Software Toolkit for the first time or enable the ECU Software Toolkit, it loads the ECU Software Toolkit UI Configuration, named NI_ECUToolkit, and saves the most recently active UI configuration as NI_ECUToolkit_SavedLayout. When you disable the ECU Software Toolkit, TestStand loads the NI_ECUToolkit_SavedLayout UI configuration. You can modify the ECU Software Toolkit UI Configuration and restore it to the default state. UI configurations store window position, size, docked state, and other settings for panes, menus, and toolbars.

Related information:

- TestStand Sequence Editor
- Insertion Palette Pane

Loading the ECU Software Toolkit UI Configuration

You can load the ECU Toolkit UI Configuration using the following steps:

1. Open the Sequence Editor Options window by navigating to **Configure** <u>»</u>**Sequence Editor Options**.

Sequence Editor Options ×			\times
General UI Configura	tion		
Close Completed E	ecution Displays on Execution	n Start	
Disable 'View User	Manager' Command		
Allow Editing of Rea	ad-Only Files		
Make Step Names	Unique When Inserting Steps		
8 🚔 File Menu M	RU List Size for Sequence File	'S	
4 🚔 File Menu M	RU List Size for Workspace Fi	les	
-Save Before Running	r		
O Always	Prompt	O Never	
Backup Sequence Files when Resaving in Older or Newer Format:			
Always	Prompt	O Never	
Help		OK Canc	el

- 2. Click the UI Configuration tab and select NI_ECUToolkit from the Saved Configurations list.
- 3. Click the Load Selected button.
- 4. Click **OK** to apply your changes.

Sequence Editor Options	\times		
General UI Configuration 1			
Display Documents in Tabs			
Lock UI Configuration by Disabling the Following:			
Toolbars/Menus:			
Dockable Panes:			
Dragging Floating Resizing			
Docking Closing/Showing Pinning/Unpinning			
A configuration stores window position, size, docked state, and various other settings for panes, menus, and toolbars. Saved Configurations: 2 NI_ECUToolkit NI_ECUToolkit NI_ECUToolkit_SavedLayout			
Save Current			
3 Load Selected			
Delete Selected			
Automatically Load Configuration at Startup			
NI_ECUToolkit ~			
Help 4 OK Cance	əl		

If loaded correctly, the ECU Toolkit UI Configuration should look like the following figures:

Figure 1. Menus and Toolbar



Figure 2. ECU Toolkit Menu Category

ECU	Toolkit Tools Window Help		
s≞	Edit Test Program: Sequence File 1		
×	Edit Pin Map File		
::	Configure Lot		
≱∎	Import Test Limits into Sequence File 1		
₿⊁	Export Test Limits from Sequence File 1		
W	Launch InstrumentStudio		
	Disable ECU Toolkit		
	About TestStand ECU Software Toolkit		

Related information:

• UI Configuration Tab - Sequence Editor Options Dialog Box

Modifying the ECU Software Toolkit UI Configuration

Complete the following steps to modify the ECU Software Toolkit UI Configuration.

- 1. Customize the toolbars and menus and arrange the sequence editor panes to create the layout you want.
- 2. Select **Configure** » **Sequence Editor Options** and click the UI Configuration tab.
- 3. Select NI_ECUToolkit in the Saved Configurations list and click the Save Current button to overwrite the existing ECU Software Toolkit UI Configuration with the UI configuration you just created and to ensure the ECU Software Toolkit loads the modified UI configuration when you launch the ECU Software Toolkit for the first time or enable the ECU Software Toolkit.

Note When you first launch the sequence editor after installing the ECU Software Toolkit or when you enable the ECU Software Toolkit, the ECU Software Toolkit loads only the UI configuration named NI_ECUToolkit.

Related information:

- Customizing Toolbars and Menus
- TestStand Panes
- UI Configuration Tab Sequence Editor Options Dialog Box

Restoring ECU Software Toolkit UI Configuration to Default State

Complete the following steps to restore the default ECU Software Toolkit UI Configuration:

- 1. Select **Configure** » **Sequence Editor Options** and click the UI Configuration tab.
- 2. Select NI_ECUToolkit in the Saved Configurations list and click the Delete Selected button to delete the UI configuration.
- 3. Select ECU Toolkit » Disable ECU Toolkit to disable the ECU Software Toolkit.
- 4. Select **ECU Toolkit » Enable ECU Toolkit** to re-enable the ECU Software Toolkit, which loads the default ECU Software Toolkit UI Configuration **NI_ECUToolkit**.

Related information:

• UI Configuration Tab - Sequence Editor Options Dialog Box

ECU Toolkit Toolbar Buttons

The ECU Software Toolkit toolbar contains the following buttons.

Command Name	lcon	Description
Edit Test Program: <filename></filename>	÷	Launches the <u>Test Program Editor</u> in which you can edit the test program settings for the active sequence file.
Edit Pin Map File	izi	Opens the pin map file associated with the active sequence file in the <u>Pin Map Editor</u> .
Configure Lot	#	Launches the <u>Configure Lot Settings</u> dialog box.
Active Configuration		The test program configuration to use when testing. The available items correspond to the configurations in the active sequence file. The value initially corresponds to the value of the LotSettings.Standard.ActiveConfigurationName property in the lot settings. If the active sequence file does not contain a configuration that corresponds to the ActiveConfigurationName lot setting, the drop-down

Command Name	lcon	Description	
		listbox displays one of the configurations in the sequence file. Changing the selected configuration with this control does not modify the ActiveConfigurationName lot setting. You can also use the <u>Configure Lot Settings</u> dialog box to change the test program configuration.	
Import Test Limits into <filename></filename>	≯₿	Imports ECU Multi Test step test data into the active sequence file from an external test data file.	
Export Test Limits from <filename></filename>	₿≯	<u>Exports ECU Multi Test</u> step test data from the active sequence file to an external test data file.	
Step Into	ţ.	Enters and suspends within a function, VI, or sequence the step calls. If the step calls a code module TestStand cannot suspend within, TestStand suspends the execution at the next step.	
		Note When you step into a VI from TestStand and then select Return to Caller without executing the VI, any values you change in the controls or indicators of the suspended VI are not returned to TestStand.	
Step Over	(=	Executes the step to which the execution pointer points when the sequence execution is in a breakpoint state. If the step is a Sequence Call step to another sequence, Step Over executes the entire sequence. The execution then enters a breakpoint state on the step following the Sequence Call step. If the engine encounters a breakpoint within the Sequence Call step, the execution pauses at the breakpoint.	
Step Out	ڈ <u>ہ</u>	Resumes execution through the end of the current sequence and suspends/pauses on the next step in the calling sequence.	
Launch InstrumentStudio	UA	Launches InstrumentStudio, which is a pin- and site-aware, software-based front panel application you can use to monitor, control, and record measurements from supported devices.	
		Note If you launch InstrumentStudio in any other way, such as from the Microsoft Windows Start	

Command Name	lcon	Description	
		menu, InstrumentStudio is not pin and site aware.	

Related information:

- TestStand Code Modules
- <u>Sequence Call Step</u>

ECU Toolkit Menu

The **ECU Toolkit** menu in the TestStand Sequence Editor contains the following items:

- Edit Test Program: <filename>—Launches the <u>Test Program Editor</u>, in which you can edit the test program settings for the active sequence file.
- Edit Pin Map File—Opens the pin map file associated with the active sequence file in the <u>Pin Map Editor</u>.
- Configure Lot—Launches the <u>Configure Lot Settings</u> dialog box.
- Import Test Limits into <filename>—Imports ECU Multi Test step test data into the active sequence file from an external test data file.
- Export Test Limits from <filename>—Exports ECU Multi Test step test data from the active sequence file to an external test data file.
- Launch InstrumentStudio—Launches InstrumentStudio, which is a pin- and siteaware, software-based front panel application you can use to monitor, control, and record measurements from supported devices.

Note If you launch InstrumentStudio in any other way, such as from the Microsoft Windows Start menu, InstrumentStudio is not pin and site aware.

- **Disable/Enable ECU Toolkit**—Disables and enables the ECU Software Toolkit and removes ECU Software Toolkit type palettes and process model plug-ins.
- About TestStand ECU Software Toolkit—Displays version information for the ECU Software Toolkit.

Related information:

• <u>TestStand Sequence Editor</u>

ECU Software Toolkit Steps Pane

The ECU Software Toolkit Steps pane is similar to the TestStand **Steps** pane of the **Sequence File** and **Execution** windows, with some custom options.

Sequence File Window

The ECU Software Toolkit Steps pane in the Sequence File window contains the following default columns:

- **Step**—Displays the name of the step and the step icon. Click to the left of the step icon to toggle the breakpoint for the step. If you add a comment for a step, the comment appears in light text above the step name.
- **Description**—Displays a description of the step that varies according to the type of step and the adapter with which it was created. The description includes the VI name or ClassName:MethodName for the associated code module.
- Num Tests—For an ECU Multi Test step, displays the number of tests defined for the step.
- **Pins**—For ECU Action and Multi Test steps, displays the pins selected on the <u>Options</u> tab of the step.
- **Multisite Option**—For ECU Action and Multi Test steps, displays the Multisite Option selected on the <u>Options</u> tab of the step.
- Settings—Displays the properties of the step that contain non-default values. Use the Step Settings pane to specify step settings.

Execution Window

The ECU Software Toolkit Steps pane in the Execution window and contains the following default columns:

- **Step**—The name and icon of the step. Click in the space to the left of the step icon to toggle the breakpoint for the step.
- **Description**—A description of the step that varies according to the type of step and the module adapter that it uses.
- **Settings**—The properties of the step that contain non-default values.

- Module Time—The code module time for the step for the cycle of execution.
- Status—The value of the status property for the step. If the step has not yet executed, the status is an empty string. After the step executes, the status reflects the results of the execution. Possible status values can vary based on the type of step. Typical values include Passed, Failed, Done, and Error.

Related information:

- <u>Steps Pane Sequence File Window</u>
- <u>Sequence File Window</u>
- <u>Step Settings Pane</u>

Test Program Editor

Use the Test Program Editor to specify the pin map files, create and edit test program configurations, and configure other settings for the <u>test program</u>.

Select ECU Toolkit » Edit Test Program: <filename> or click the Edit Test Program: <filename> button on the ECU Software Toolkit toolbar to launch the Test Program Editor for the sequence file.

The Test Program Editor contains the following panels:

- <u>Pin Map</u>—Specifies the path of the pin map file.
- InstrumentStudio Project—Specifies the path of the InstrumentStudio project file.
- <u>Test Limits Files</u>—Specifies a list of test limits file references available for use in the test program. Each test program configuration can specify a test limits file to load when test program execution begins.
- <u>Configuration Definition</u>—Specifies a table of standard and custom test conditions available for use in the test program. Each test program configuration can specify a value for the test conditions.
- <u>Configurations</u>—Specifies a list of configurations available for use in the test program.
- <u>Configuration Panels</u>—Specifies test condition values and test limits file for the configuration.

Pin Map Panel

The Pin Map panel allows users to specify and edit a pin map file to use in a test program.

The Pin Map panel contains the following options:

• **Pin Map File Path**—Specifies the relative pathname to the <u>pin map file</u> to use in the test program.

A red exclamation point indicates that an issue exists with the file, such as the file is invalid or that the file does not conform to the <u>Pin Map XML schema</u>. A tooltip displays the error message. If errors exist in the file, you make changes in the dialog box, and click **OK**, ECU Software Toolkit commits those changes. Do not proceed without reviewing the errors for the pin map file.

• **Open file for edit** <u>■</u>—Launches the pin map file in the <u>Pin Map Editor</u>.

InstrumentStudio Project Panel

The InstrumentStudio Project panel allows users to specify and edit an InstrumentStudio project file to use in a test program.

The InstrumentStudio Project panel contains the following options:

• InstrumentStudio Project File Path—Specifies the pathname of the InstrumentStudio project file to use in the test program.

A red exclamation point indicates that an issue exists for the file. A tooltip displays the error message. Do not proceed without reviewing the errors for the file.

• Open file for edit III—Launches InstrumentStudio. If you specify an InstrumentStudio project file, this project opens in InstrumentStudio, otherwise the most recent project loads.

Test Limits Files Panel

The Test Limits Files panel allows the customization and inclusion of multiple test limits file references for a test program.

Click the **Add Test Limits File** button or the **Remove Test Limits File** button at the bottom of the panel to add or remove <u>test limits</u> file references available for use in the test program. Each test program configuration can specify a test limits file to load when test program execution begins. The Test Limits Files panel contains the following options:

- Test Limits File list—Each entry in the list contains the following fields:
 - Name—Specifies a unique name for the test limits file reference. A red exclamation point indicates that the test limits reference name is invalid. A tooltip displays the error message.
 - Test Limits File Path—Specifies the relative pathname to the test limits file. A red exclamation point indicates that the file is invalid. A tooltip displays the error message.
 - **Open file for edit** —Launches the test limits file in the default application associated with the file extension on the computer. This option is not available if the test limits file is embedded in the test program file.
- Add Test Limits File—Adds a new test limits file reference to the list.
- Remove Test Limits File—Removes the test limits file reference you select from the list.
- Embed Test Limits File—Embeds the contents of the selected test limits file in the test program sequence file. To protect limits files from modification or viewing, embed the limits file in the test program sequence file and use the TestStand password protection option to lock the sequence file. The Embed Test Limits File option is available when the test limits file path is valid and not already embedded in the sequence file.

Note NI does not recommend using passwords as the only way of protecting intellectual property.

• Extract Test Limits File—Extracts an embedded test limits file from the test program sequence file. Use this option to view or change the contents of the test limits file.

Related information:

• Advanced Tab - Sequence File Properties Dialog Box

Configuration Definition Panel

The Configuration Definition panel specifies a table of standard and custom test conditions available for use in the test program. Each test program configuration specifies a value for the test conditions.

The Configuration Definition panel contains the following options:

- **Test Condition Type**—Specifies the data type of the test condition. You can modify the type only or custom test conditions. When you modify a test condition type, each configuration resets the test condition value to a default value.
- **Test Condition Name**—Specifies the name of the test condition. Only custom test condition names can be modified. A red exclamation point indicates that the custom test condition name is invalid. A tooltip displays the error message.
- Add Condition—Launches the Specify New Condition Name dialog box, in which you can add a new test condition. You can select a standard test condition from a list or create a new custom test condition.
- **Delete Condition**—Removes the test condition you select from the test program and the test program configurations.

Configurations Panel

The Configurations panel specifies a list of configurations available for use in the test program.

When you specify a new configuration, the Test Program Editor updates with a new panel that matches the name of the configuration you added. Use the corresponding <u>Configuration</u> panel to specify test condition values and fields for configuring the test limits file for the configuration.

The Configurations panel contains the following options:

- **Configuration Name**—Specifies the name of the configuration. A red exclamation point indicates that the configuration name is invalid. A tooltip displays the error message.
- Add Configuration—Adds a configuration to the list.
- Delete Configuration—Removes the configuration you select from the list.

Configuration Panels

Each configuration you specify uses a corresponding Configuration panel that contains a table to specify test condition values and fields for configuring the test limits file for the configuration.

The Configuration panels contains the following options:

- Test Condition Name—Displays the name of the test condition.
- **Test Condition Value**—Specifies the value for the test condition for the configuration.
- **Test Limits File**—Specifies the test limits file the configuration loads when test program execution begins.
- **Open for edit** —Launches the test limits file in the default application associated with the file extension on the computer.
- Test Limits File Import Mode—Specifies whether the test limits file <u>replaces only</u> <u>matching tests or deletes and adds new tests</u> when loaded into the test program.
- **Require every step to be in test limits file**—Returns a run-time error and does not import the file if a test or step exists in the sequence but does not exist in the limits file. If you do not enable this option, the Import/Export Test Limits tool imports only matching tests or steps in the limits file.
- Import into Sequence File—Imports the test limits file into the sequence file.

Pin Map Editor

Use the Pin Map Editor to view, create, modify, and save pin map files instead of editing the XML files directly.

Use the <u>Pin Map</u> panel in the <u>Test Program Editor</u> to specify a pin map file for a test program. The pin map file also serves as the channel map file.

Select **ECU Toolkit** <u>Bedit Pin Map File</u> or click the **Edit Pin Map File** button on the ECU Software Toolkit <u>toolbar</u> to launch the Pin Map Editor. Alternatively, you can select **ECU Toolkit** <u>Bedit Test Program</u> and then select **Pin Map** in the Test Program Editor to launch the Pin Map panel. Click the **Open file for edit** <u>Bedit</u> button to launch the Pin Map Editor.

The Pin Map Editor uses a red error icon in the Errors and Warnings window to indicate that the file does not conform to the <u>Pin Map schema</u>. Click the error icon or doubleclick the <u>error message</u> to highlight the error on the XML tab. The Errors and Warnings window displays a warning in orange text when the file conforms to the Pin Map schema but will generate an error at run time.

When you close the editor with the **Cancel** button and the pin map file has been edited since you opened it in the editor, a dialog box prompts you to discard changes or return to the editor. When you close the editor with the **OK** button, the file is updated on disk with the changes you made in the editor.

Configuring Pin Map Files

The Pin Map Editor includes the following options and tabs:

- **Pin Map File**—Specifies the pin map file to load. You can manually enter a relative file path.
- Undo—Removes the last edit made.
- Redo—Reinstates the last edit removed.
- **Open**—Launches the Select Pin Map File dialog box, in which you can browse to the pin map file to load.
- Save—Displays a context menu that includes a Save option to save the file to the current path and a Save As option to launch as Save As dialog box. The Save button is disabled when errors exist in the pin map file.
- New—Creates a new pin map file. If the file currently open in the editor has unsaved changes, a dialog box prompts you to discard the changes or return to the editor to save or cancel the changes to the file before creating a new file.
- <u>Pin Map tab</u>—Displays an editable, hierarchical view of the instruments, pins, pin groups, relays, relay groups, relay configurations, sites, and connections in the pin map file. For each item, you can edit the attributes of the item, insert new items, or insert comments in the file.
- XML tab—Displays the pin map file in a text format that you can edit.
- Errors and Warnings window—Displays issues to resolve. Click the Goto Error in XML error icon to highlight the error on the XML tab.

Pin Map Tab

The Pin Map tab contains the following sections:

- <u>Instruments</u>—Use this section to specify the instruments required to execute the test program.
- <u>Pins</u>—Use this section to specify the DUT pins and other pin types connected to the tester.
- <u>Pin Groups</u>—Use this section to specify a grouping of pins that you can reference with a single name.
- <u>Relays</u>—Use this section to specify the relays on the tester that the test program associates with the pin map file references.
- <u>Relay Groups</u>—Use this section to specify a grouping of relays that you can reference with a single name.
- <u>Relay Configurations</u>—Use this section to specify a set of relays and their positions.
- <u>Sites</u>—Use this section to specify the sites on the tester.
- <u>Connections</u>—Use this section to specify mappings from pins to instrument channels for every site and for system resources.

Instruments

Use the Instruments section on the <u>Pin Map tab</u> to specify the type of instruments required to execute the test program and the name and attributes of each instrument.

You can right–click any item in the section you want to edit and use the context menu to complete common tasks.

Choose one of the following options to add an instrument to the pin map file:

- Click <Add Instruments Here> to display the Instruments pane, and click the button of the instrument type you want to add.
- Right–click <Add Instruments Here> and select the instrument type you want to add from the context menu.

The Pin Map Editor automatically adds the instrument to the **Instruments** section. Select an instrument in the **Instruments** section to display the Instruments pane, where you can edit the attributes of the instrument.

You can also cut, copy, and paste instruments, or add comments in the Instruments pane. Use the **Comment** button to specify a comment for the selected instrument. Comments display beneath the instrument they modify.



- Consider getting instrument names to use in ECU Test programs from the System Components <u>»</u> General Information section of the ECU Test System Maintenance Software Report or from NI MAX.
- Names for NI instruments in the pin map file are not case sensitive.

The ECU Software Toolkit supports the following instrument types and instrument attributes:

- **DCPower**—Defines an NI-DCPower instrument.
 - **Name**—Name of the instrument, as defined in Measurement & Automation Explorer (MAX).
 - Number of Channels—Number of channels available on the instrument.
 - Channel Group Name/Channels(s) table—Lists the channel groups and the channels assigned to each group. By default, the Pin Map Editor creates one channel group containing all instrument channels. Use the plus (+) or minus (-) buttons to add or remove channel groups.

Note A channel group is a collection of channels controlled by the same instrument session. NI-DCPower channel groups can contain channels from different physical NI-DCPower instruments.

 Channel Group Name—Name of the channel group(s). The Channel Group Name is case sensitive and must not duplicate an instrument name or a group name on another instrument type.



Note NI-DCPower channels cannot be added to groups of other instruments.

Channel(s)—Channel(s) assigned to a channel group. When you add a new channel group, the Pin Map Editor prompts you to add channels to the new group. Define the channels as a comma-separated list (for example, 0,1,3,..,n), a continuous range (for example, 0:3), or as a combination of the two (for example, 0:1,3).



Note All channels for all instruments must be assigned to a channel group and no channel can assigned to more than one group.

- **Digital Pattern**—Defines an NI-Digital Pattern instrument.
 - Name—Name of the instrument, as defined in MAX.
 - Number of Channels—Number of channels available on the instrument.
 - Group—Name of the group that contains the instrument. By default, the Pin Map Editor sets this attribute to Digital when you add NI-Digital Pattern instruments to the pin map file. By using the same group name for all NI-Digital Pattern instruments, ECU Software Toolkit combines all instruments into a single session so you can avoid session loops in code modules. To create multiple NI-Digital Pattern sessions, use a unique name for each set of instruments for which you want to create a session. Refer to the Digital Pattern Help for information about hardware limitations that prevent certain instruments from operating together as a single instrument.



Note Instrument group names must be unique and must not duplicate instrument names in the pin map file.

- **RFSA**—Defines an NI-RFSA instrument. NI-RFSA instruments define a channel named In.
 - **Name**—Name of the instrument, as defined in MAX.
- **RFSG**—Defines an NI-RFSG instrument. NI-RFSG instruments define a channel named Out.
 - **Name**—Name of the instrument, as defined in MAX.
- VST—Defines an NI-VST instrument that can hold RFSA, RFSG, and FPGA sessions. NI-VST instruments define a channel named In and another channel named Out.
 - **Name**—Name of the instrument, as defined in MAX.
 - **Custom FPGA File**—(Optional) path to the FPGA file relative to the path of the pin map file. You can manually specify an absolute file path.
- **RFPM**—Defines an RF Port Module instrument that can hold RFPM, RFmx, RFSA, RFSG, and FPGA sessions.
 - **Name**—Name of the VST instrument, as defined in MAX, that is part of the RF port module subsystem.
 - **Custom FPGA File**—(Optional) path to the FPGA file relative to the path of the pin map file. You can manually specify an absolute file path.

- **Calibration File**—Path, relative to the path of the pin map file, to the TDMS files that contain the calibration data for the RF Port Module instrument. You can manually specify an absolute file path.
- **IVI Switch Resource Name**—IVI Switch resource name associated with the port module, as defined in MAX.
- Ports List—Defines the ports available in the RF Port Module in a commaseparated list of numbers or ranges of numbers separated by a hyphen. Port number ranges are inclusive and must be in ascending order, for example, channelList="2,3,4-8".
- HSDIO—Defines an NI-HSDIO instrument.
 - **Name**—Name of the instrument, as defined in MAX.
 - **Number of Channels**—Number of channels available on the instrument.
 - PFI Lines—(Optional) Defines the PFI lines available in the NI-HSDIO instrument in a comma-separated list of numbers or ranges of numbers separated by a hyphen. PFI number ranges are inclusive and must be in ascending order, for example, PFILines="2,3,4-8".
- **DMM**—Defines an NI-DMM instrument. NI-DMM instruments define a single channel, displayed within ECU Software Toolkit as channel 0.
 - **Name**—Name of the instrument, as defined in MAX.
- **SCOPE**—Defines an NI-SCOPE instrument.
 - **Name**—Name of the instrument, as defined in MAX.
 - Number of Channels—Number of channels available on the instrument.
 - Group—Name of the group that contains the instrument. By default, the <u>Pin</u> <u>Map Editor</u> sets this attribute to Scope when you add NI-SCOPE instruments to the pin map file. By using the same group name for all NI-SCOPE instruments, ECU Software Toolkit combines all instruments into a single session so you can avoid session loops in code modules. To create multiple NI-SCOPE sessions, use a unique name for each set of instruments for which you want to create a session. Refer to the NI-SCOPE Help for information about hardware limitations that prevent certain instruments from operating together as a single instrument.



Note Instrument group names must be unique and must not duplicate instrument names in the pin map file.

- FGEN—Defines an NI-FGEN instrument.
 - **Name**—Name of the instrument, as defined in MAX.

- Number of Channels—Number of channels available on the instrument.
- **DAQmx**—Defines an NI-DAQmx task, not an instrument.
 - **Name**—Name of the task, as defined in test program code modules.
 - **Task Type**—Category of the task. Pin queries that return tasks of more than one task type return an error.
 - **Channel List**—List of physical channels associated with the task.
- Relay Driver—Defines a PXI-2567 relay driver module.
 - **Name**—Name of the relay driver module, as defined in MAX.
 - **Number of Control Lines**—Number of control lines available on the relay driver module.
- **Multiplexer**—Defines a switching instrument to use as a multiplexer across multiple test sites. You can use one instrument multiplexed across multiple test sites or multiple instruments multiplexed across multiple test sites.
 - **Name**—Name of the Switch Executive virtual device, as defined in MAX.
 - Multiplexer Type—(Optional) String that identifies the switch type, family, class, or product group. You cannot specify a value that begins with ni. This value is a string that you define in the pin map and is not a predefined value from some other source, such as a name in MAX, that you select. Use this value to identify all instances of a particular switch type. Switches of the same type typically have the same session data type and same driver API.
- Model-Based Instrument—Defines a model-based instrument.
 - Name—Unique string that identifies the instrument.
 - Instrument Model—List of installed model description files in the Instrument Model Library.
 - Property/Value Tables—Editable tables of instrument and resource properties as defined in the model description file. The first table contains properties for the entire instrument. Subsequent tables contain properties for specific resources of the instrument.
 - **Category**—Category of the ModelBasedInstrument. The category is set in the instrument model and cannot be changed.
 - **Subcategory**—Subcategory of the ModelBasedInstrument. The subcategory is set in the instrument model and cannot be changed.

Note The ECU Test System only supports the RMX-410x Power Supply as a model-based instrument. Follow these steps to add an RMX-410x power supply as a model-based instrument:

1. In the Pin Map tab, select **<Add Instruments Here>**. The Pin Map

Editor displays the instrument types that you can add.

- 2. Click Model-Based Instrument to launch the configuration window.
- 3. In the **Name** field, specify the name of the RMX-410x instrument that you are adding to the pin map.
- 4. Select **RMX-410x** from the **Instrument Model** drop-down menu. The <u>Pin Map Editor</u> displays the category and subcategory of the modelbased instrument you select.
- 5. Specify the **Resource Name** property.
- Switch Executive—Defines a switching instrument to use as a Switch Executive virtual device across one site or multiple test sites. You can use one or multiple instruments switched across the DUT pins on one or multiple test sites.
 - **Name**—Name of the Switch Executive virtual device, as defined in NI MAX.

Pins

Use the **Pins** section on the <u>Pin Map tab</u> to specify the DUT pins and other pin types connected to the tester.

You can right–click any item in the section you want to edit and use the context menu to complete common tasks.

Choose one of the following options to add a pin connection to the pin map file:

- Click <Add Pins Here> to display the Pins pane, and click the button of the pin type you want to add.
- Right–click <**Add Pins Here**> and select the pin type you want to add from the context menu.

The Pin Map Editor automatically adds the pin connection to the **Pins** section. Select a pin connection in the **Pins** section to display the Pins pane, where you can edit the pin's name.

Note Pin names are case sensitive, must begin with a letter or underscore (_), and are limited to A-Z, a-z, 0-9, or _ characters.

You can also cut, copy, and paste pins, or add comments in the Pins pane. Use the

Comment button to specify a comment for the selected pin. Comments display beneath the pin they modify.

The ECU Software Toolkit supports the following pin connection types:

- **DUT pin**—Specifies a DUT pin, which is a pin on a DUT or a resource on the tester or DIB that is associated with one or more sites.
- System pin—Specifies a system pin, which is resource on the tester or DIB that is connected to an instrument

Pin Groups

Use the **Pin Groups** section on the <u>Pin Map tab</u> to specify a grouping of pins that you can reference with a single name.

Choose one of the following options to add a pin group to the pin map file:

- Click <Add Pin Groups Here> to display the Pin Groups pane, and click the Pin Group button.
- Right–click <Add Pin Groups Here> and select Pin Group from the context menu.

The Pin Map Editor automatically adds the pin group to the **Pin Groups** section. Select a pin group in the **Pin Groups** section to display the Pin Groups pane, where you can change the pin group name and use the individual checkboxes or the **Select All** checkbox to add or remove DUT pins, system pins, or pin groups from the selected pin group.

Note Pin group names are case sensitive, must begin with a letter or underscore (_), and are limited to A-Z, a-z, 0-9, or _ characters.

Once you create a pin group, you can also add pins or other pin groups to the pin group using one of the following options:

- Click <Add Pin References Here>, and click the Pin Reference button in the Pin Groups pane to display a drop-down menu of available pins and pin groups.
- Right–click **<Add Pin References Here>**, and select Pin Reference from the context menu to display a drop–down menu of available pins and pin groups.

• Drag pins or pins groups from the **Pins** or **Pin Groups** section into a pin group.

You can also cut, copy, and paste pins, or add comments in the Pin Groups pane. Use the **Comment** button to specify a comment for the selected pin group. Comments display beneath the pin group they modify.

Relays

Use the **Relays** section on the <u>Pin Map tab</u> to specify the relays on the site and the relays on the tester that the test program associates with the pin map file references.

You can right–click any item in the section you want to edit and use the context menu to complete common tasks.

Choose one of the following options to add a relay to the pin map file:

- Click <Add Relays Here> to display the Relays pane, and click the button of the relay type you want to add.
- Right–click <Add Relays Here> and select the relay type you want to add from the context menu.

The Pin Map Editor automatically adds the relay to the **Relays** section. Select a relay in the **Relays** section to display the Relays pane, where you can edit the relay's name, open state display label, and closed state display label.



Note Relay names are case sensitive, must begin with a letter or underscore (_), and are limited to A-Z, a-z, 0-9, or _ characters.

You can also cut, copy, and paste relays, or add comments in the Relays pane. Use the **Comment** button to specify a comment for the selected relay. Comments display beneath the relay they modify.

The ECU Software Toolkit supports the following relay types:

- Site relay—Specifies a site relay, which is a relay on the tester or DIB that is connected to a relay driver module and that is associated with one or more sites.
- System relay—Specifies a system relay, which is a relay on the tester or DIB that is

connected to a relay driver module and that is associated with all sites.

Relay Groups

Use the **Relay Groups** section on the <u>Pin Map tab</u> to specify a grouping of relays that you can reference with a single name.

You can right–click any item in the section you want to edit and use the context menu to complete common tasks.

Choose one of the following options to add a relay group to the pin map file:

- Click <Add Relay Groups Here> to display the Relay Groups pane, and click the Relay Group button.
- Right–click <Add Relay Groups Here> and select Relay Group from the context menu.

The Pin Map Editor automatically adds the relay group to the **Relay Groups** section. Select a relay group in the **Relay Groups** section to display the Relay Groups pane, where you can change the relay group name and use the individual checkboxes or the **Select All** checkbox to add or remove site relays, system relays, or relay groups from the selected relay group.

Note Relay group names are case sensitive, must begin with a letter or underscore (_), and are limited to A-Z, a-z, 0-9, or _ characters.

Once you create a relay group, you can also add relays or other relay groups to the relay group using one of the following options:

- Click <Add Relay References Here>, and click the Relay Reference button in the Relay Groups pane to display a drop-down menu of available relays and relay groups.
- Right–click <Add Relay References Here>, and select Relay Reference from the context menu to display a drop–down menu of available relays and relay groups.
- Drag relays or relay groups from the **Relays** or **Relay Groups** section into a relay group.

You can also cut, copy, and paste relay groups, or add comments in the Relay Groups pane. Use the **Comment** button to specify a comment for the selected relay group. Comments display beneath the relay group they modify.

Relay Configurations

Use the **Relay Configurations** section on the <u>Pin Map tab</u> to specify a set of relays and their relay position states.

You can right–click any item in the section you want to edit and use the context menu to complete common tasks.

Choose one of the following options to add a relay configuration to the pin map file:

- Click <Add Relay Configurations Here> to display the Relay Configurations pane, and click the Relay Configuration button.
- Right–click <Add Relay Configurations Here> and select Relay Configuration from the context menu.

The Pin Map Editor automatically adds the relay configuration to the **Relay Configurations** section. Select a relay configuration in the **Relay Configurations** section to display the Relay Configuration pane, where you can change the relay configuration name, select the relay position state, view the final relay state, and change the order of relays and relay groups. Relay state positions are applied sequentially in the order in which the relay or relay group appears in the Relay Configuration pane. You can change the order of a relay or relay group by using the **Up** and **Down** buttons to move the selected relay or relay group.

Alternatively, when you create a new relay configuration, you can click **Add Relay Positions Here>**, then click the **Relay Position** button. Select the relay or relay group you want to add from the top drop–down menu, then select the relay position state from the bottom drop–down menu. Use the Relay Configuration pane to make additional changes.

Click the **Relay Configurations** section to display the Relay Configurations table, which includes a column for each relay and a column for each relay configuration. Each relay configuration column shows the final relay state of the corresponding relay(s).

You can also cut, copy, and paste relay configurations, or add comments in the Relay Configurations pane. Use the **Comment** button to specify a comment for the selected relay configuration. Comments display beneath the relay configuration they modify.

Sites

Use the Sites section on the Pin Map tab to specify the sites on the tester.

Site numbers start at 0 and must be consecutive without gaps. You can right–click any item in the section you want to edit and use the context menu to complete common tasks.

Choose one of the following options to add a site to the pin map file:

- Click <Add Sites Here> to display the Sites pane, and click the Site button.
- Right-click <Add Sites Here> and select Site from the context menu.

The Pin Map Editor automatically adds the site to the **Sites** section. Select a site in the **Sites** section to display the Sites pane.

You can also cut, copy, and paste sites, or add comments in the Sites pane. Use the **Comment** button to specify a comment for the selected site. Comments display beneath the site they modify.

Connections

Use the **Connections** section on the <u>Pin Map tab</u> to specify mappings among pins, relays, sites, instruments, instrument channels, relay driver modules, and relay driver control lines.

You can right–click any item in the section you want to edit and use the context menu to complete common tasks. To view and edit all connections and connection attributes, click the **Connections** section to display the <u>Connections table</u>.

Choose one of the following options to add a connection to the pin map file:

- Click <Add Connections Here> to display the Connections pane, and click the button of the connection type you want to add.
- Right-click <Add Connections Here> and select the connection type you want to

add from the context menu.

The Pin Map Editor automatically adds the connection to the **Connections** section. Select a connection in the **Connections** section to display the Connections pane, where you can edit the attributes of the connection.

You can also cut, copy, and paste connections, or add comments in the Connections pane. Use the **Comment** button to specify a comment for the selected connection. Comments display beneath the connection they modify.

The ECU Software Toolkit supports the following connection types:

- **Connection**—Specifies a connection between a DUT pin and an instrument channel for one or more sites.
- **System Connection**—Specifies a direct connection between a system pin and an instrument channel.
- Multiplexed Connection—Specifies a multiplexed connection between the same DUT pin on multiple sites and a single instrument channel.
- **Relay Connection**—Specifies a connection between a site relay and a relay driver module control line for one or more sites.
- **System Relay Connection**—Specifies a direct connection between a system relay and a control line of a relay drive module.
- Switch Executive Connection—Specifies a switched connection between a DUT pin and an instrument channel for one site through a Switch Executive virtual device.

Connections Table

Use the Connections table on the <u>Pin Map tab</u> to view and edit connections and connection attributes.

The Connections table includes a row for every possible connection that can be made with the available pins, relays, and sites. Use the **View Connections for** drop–down menu to filter the view by categories, such as by all pins and relays, DUT pins and site relays by site, and system pins and relays.

The Connections table includes the following sortable columns:

- **Pin**—Specifies the name of a device pin to connect to an instrument channel or the name of a relay to connect to a control line on a relay driver module.
- Site—Indicates the test site or sites to which the connection for a particular row applies. After selecting an instrument or relay driver module for the row, you can enter a comma-separated list of site numbers to allow multiple sites to <u>share the connection</u>.
- Instrument—Specifies the name of the instrument or relay driver module to connect. Select <Disconnect> to remove the association between the pin and the instrument or the relay and the relay driver module.
- **Channel**—Specifies an instrument channel number to assign to the pin or a control line on a relay driver module to assign to the relay.
- **Multiplexer/Switch**—If applicable, specifies the multiplexer required to create the route or the Switch Executive virtual device name.
- **Route**—If applicable, specifies the multiplexer route required to connect the pin and site to the instrument and channel.

ECU Software Toolkit 22.1 Step Types

The ECU Software Toolkit step types are included in the following three groups in the Step Types list on the **Insertion Palette** pane in the TestStand Sequence Editor: ECU Toolkit, SLSC 12251/2, and Measurement Configuration.

Insertion Palette

Step Types
.N .NET
ECU Toolkit
.N ECU Action
.N ECU Multi Test
± Tests
TP FTP Files
Additional Results
Sequence Call
fx Statement
→ Property Loader
📲 Label
Message Popup
····»_ Call Executable
Flow Control
Synchronization
🗄 🖓 Database
🗄 🖓 Data Streams
LabVIEW Utility
LabVIEW NXG Utility
🗄 ·· SystemLink
in Configuration
□ Instrument Control
⊡ SLSC 12251/2
SLSC FIU - Read Current
SLSC FIU - Reset
* Create Sessions and Apply Configuration
💆 Apply Configuration
Close Sessions

- Measurement Configuration - Measurement Control Digital Multimeter (DMM) DMM - Read Measurement Waveform Generator (FGen) FGen - Initiate Generation FGen - Abort Generation FGen - Enable Output Gen - Disable Output Oscilloscope (Scope) Scope - Initiate Acquisition Scope - Abort Acquisition SMU/Power Supply (SMU/PPS) SMU/PPS - Initiate SMU/PPS - Abort SMU/PPS - Enable Output SMU/PPS - Disable Output SMU/PPS - Read Measurement SMU/PPS - Reset Device RMX Power Supply (RMXPS) RMXPS - Enable Output RMXPS - Disable Output • RMXPS - Read Measurement SMXPS - Reset Output Protection - Switch Executive * NISE - Open Sessions X NISE - Close Sessions NISE - Pin to Sessions O NISE - Connect NISE - Disconnect NISE - Disconnect All Create Pin Sessions ▲ Apply Pin Configuration X Close Pin Sessions

ECU Toolkit

The <u>ECU Toolkit group</u> in the Step Types list contains the following step types:

• <u>ECU Multi Test</u>—Evaluates one or more parametric or functional tests for the DUT. A single ECU Multi Test step can specify multiple parametric or functional tests. You can configure <u>multisite</u> options directly on the step. • <u>ECU Action</u>—Performs an action, such as instrument configuration, with access to the pin map. You can configure <u>multisite</u> options directly on the step.

Note ECU Software Toolkit steps disable the Switching panel of the Properties tab in the Step Settings pane. Use <u>Switch Executive step types</u> from the ECU Software Toolkit and <u>Switch Executive connections defined in</u> <u>the pin map</u> to perform switching operations.

SLSC 12251/2

The SLSC 12251/2 group, located at **IO Configuration** <u>Instrument Control</u> **SLSC 12251/2** in the Step Types list, contains the following step types:

- <u>SLSC FIU Read Current</u>—Measures the current through the specified SLSC FIU channels.
- **<u>SLSC FIU Reset</u>**—Returns the SLSC FIU to the initialized state.

Measurement Configuration

The <u>Measurement Configuration group</u> in the Step Types list contains the following step types:

 <u>Create Pin Sessions</u>—Initializes the Pin Map Context with the driver sessions for instruments. Instruments must be defined in the pin map file, used in at least one pin connection, and used in a site enabled in the <u>Configure Lot Settings</u> dialog box for a driver session to be created.



Note The <u>Create Pin Sessions</u> step supports RMX-410x power supplies and instruments programmed using NI-FGEN, NI-DCPower, NI-DMM, or NI-SCOPE.

- <u>Apply Pin Configuration</u>—Resets the instrument to its default settings and configures the selected pins with pin settings from the corresponding measurement configuration file.
- <u>Close Pin Sessions</u>—Closes all instrument driver sessions initialized by the <u>Create</u> <u>Pin Sessions</u> step.

The Measurement Configuration group also contains the Measurement Control group, which contains groups with steps for specific instrument types.

- Digital Multimeter (DMM) Group-
 - <u>DMM Read Measurement</u>—Reads, publishes, and returns the digital multimeter measurement results for the specified DMM pins on all sites in the

Pin Map context.

- Waveform Generator (FGen) Group-
 - <u>FGen Initiate Generation</u>— Initiates waveform generation for the specified FGen pins on all sites in the Pin Map context.
 - <u>FGen Abort Generation</u>—Aborts waveform generation for the specified FGen pins on all sites in the Pin Map context.
 - <u>FGen Enable Output</u>—Enables waveform generation for the specified FGen pins on all sites in the Pin Map context.
 - <u>FGen Disable Output</u>—Disables waveform generation for the specified FGen pins on all sites in the Pin Map context.
- Oscilloscope (Scope) Group—
 - <u>Scope Initiate Acquisition</u>—Initiates waveform acquisition for the specified Scope pins on all sites in the Pin Map context.
 - <u>Scope Read Measurements</u>—Reads and publishes the Scope measurement results for the specified Scope pins on all sites in the Pin Map context.
 - <u>Scope Abort Acquisition</u>—Aborts acquisition for the specified Scope pins on all sites in the Pin Map context.
- SMU/Power Supply (SMU/PPS) Group—
 - <u>SMU/PPS Initiate</u>—Starts generation or acquisition for the specified SMU/PPS pins on all sites in the Pin Map context.
 - <u>SMU/PPS Abort</u>—Transitions the specified SMU/PPS pins on all sites in the Pin Map context from the Running state to the Uncommitted state and terminates any running sequences for those pins.
 - <u>SMU/PPS Enable Output</u>—Enables generation for the specified SMU/PPS pins on all sites in the Pin Map context.
 - <u>SMU/PPS Disable Output</u>—Disables generation for the specified SMU/PPS pins on all sites in the Pin Map context.
 - <u>SMU/PPS Read Measurement</u>—Reads, publishes and returns the voltage, current, and power measurement results for the specified SMU/PPS pins on all sites in the Pin Map context.
 - <u>SMU/PPS Reset Device</u>—Resets all instruments connected to the specified SMU/PPS pins on all sites in the Pin Map context to a known state.
- RMX Power Supply (RMXPS) Group-
 - <u>RMXPS Enable Output</u>—Enables power generation for the specified RMXPS pins on all sites in the Pin Map context.
 - <u>RMXPS Disable Output</u>—Disables power generation for the specified RMXPS pins on all sites in the Pin Map context.

- <u>RMXPS Read Measurement</u>—Reads, publishes and returns the voltage, current, and power measurement results for the specified RMXPS pins on all sites in the Pin Map context.
- <u>RMXPS Reset Output Protection</u>—Clears the latch that disables output when an overvoltage or undervoltage fault condition is detected on the specified RMXPS pins on all sites in the Pin Map context.
- Switch Executive (NISE) Group-
 - <u>NISE Open Sessions</u>—Creates sessions for all NI Switch Executive virtual devices in the Pin Map context and stores the created sessions in the Pin Map context.
 - <u>NISE Close Sessions</u>—Closes all NI Switch Executive virtual device sessions in the Pin Map context.
 - <u>NISE Pin to Sessions</u>—Returns the NI Switch Executive virtual device sessions, switch routes, and new Pin Map context objects required to access a switched pin and a switched instrument channel.
 - **<u>NISE Connect</u>**—Connects the routes specified in the connection specification.
 - <u>NISE Disconnect</u>—Disconnects the routes specified in the disconnection specification.
 - <u>NISE Disconnect All</u>—Disconnects all connections on every IVI switch device managed by NI Switch Executive virtual device sessions in the Pin Map context.

Related information:

- Insertion Palette Pane
- <u>Step Settings Pane</u>

ECU Toolkit Group

Steps in the ECU Toolkit group are used to execute tests and actions within the test program. They generally have their module adapter configured as Sequence, and call other sequences to accomplish specific tasks.

ECU Multi Test Step

Use the ECU Multi Test step to evaluate one or more parametric or functional tests for the DUT.

A single ECU Multi Test step can specify multiple parametric or functional tests. You

can configure multisite options directly on the step.

Note You cannot use multiple ECU Multi Test steps or ECU Action steps configured to use multiple threads in While loops, in Do While loops, or in For loops that use the Custom Loop option when performing multisite testing. The steps report a run-time error in these situations. Use other types of loops instead, such as For loops that use the Fixed Number of Iterations option.

Configuring the Step

You can configure a single ECU Multi Test step test to evaluate a Boolean or numeric measurement against specified limits. Each numeric limit test can have independent limits, base units, and comparison type. Configure each measurement the same way you configure an individual Numeric Limit Test step. An ECU Multi Test step passes when none of the specified tests fail.

Use the ECU Multi Test step <u>edit tabs</u> in the TestStand Sequence Editor to specify the tests, comparison, limits, and multisite options.

Related information:

- While Step
- <u>Do While Step</u>
- For Step
- For Loop Edit Tab
- Numeric Limit Test Step

ECU Multi Test Step Execution Overview

The <u>ECU Multi Test</u> step uses the following process when executing:

- Creates a PinMapContext object and stores it in the Step.PinMapContext property if the site that executes the step is the working site. The PinMapContext object describes a subset of pins, relays, sites, and instruments on a test system.
- 2. Replaces each test that specifies a pin group with a set of tests that are equivalent to the pin group test and deletes the pin group test, except for the following

differences:

- The Pin is set to the name of the pin in the pin group.
- The Test Number is computed by adding the test number specified for the pin group test to the zero-based index of the pin in the pin group.
- The Test Name is computed by appending the name of the pin in the pin group to the test name specified for the pin group test.

Note This replacement happens the first time the step executes.

- 3. Calls a sequence that completes the following actions:
 - a. Receives the Step.PinMapContext property as an input parameter.
 - b. Uses Measurement Configuration group step types to configure instruments, perform measurements, and publish measurement data. These step types automatically obtain instrument channel and session information based on your specified Pin Map context and Pins. Choose the appropriate Measurement Configuration group steps based on the instruments used in your test and your test requirements.
- 4. Performs tests. For each test, the step completes the following actions:
 - a. Obtains the measurement value from the published measurement data or from the expression, if specified, in the **Test Data Source** column on the <u>Tests</u> tab.
 - b. Compares the measurement value against the specified limits.
 - c. Sets the test status depending on the evaluation type, limits, or measurement value, as shown in the following table.

Evaluation Type	Condition	Status
Numeric Limit	You specify a low limit and high limit	Passed or Failed
Numeric Limit	You do not specify a low limit and high limit	Done
Decc/Feil	The measurement value is True	Passed
Pass/Fail	The measurement value is False	Failed
None	None	Done



- d. Stores the measurement value in the expression, if specified, in the **Export Data To** column on the Tests tab.
- e. Continues or stops performing tests depending on whether a test failed and whether you enable the **Stop Performing Tests after First Failure** option on the Options tab. The step stores the measurement value in the Export Data To expression for tests with the **None** evaluation type, regardless of whether a test failed.
- 5. Sets the step status to Passed if all tests pass. Otherwise, sets the step status to Failed or Error.

Related information:

• TestStand - Expressions

ECU Multi Test Edit Tabs

Use the <u>ECU Multi Test</u> step edit tabs in the TestStand Sequence Editor to specify the tests, limits, and multisite options for the ECU Multi Test step.

The Step Settings pane for the ECU Multi Test step contains the following tabs:

- <u>Tests</u>—Configure the tests for the step. Each test specifies the test number, test name, pin or pin group, published data ID, limits, scaling factor, base units, evaluation type, test data source, and measurement destination.
- <u>Options</u>—Configure <u>multisite</u> and other options for the step.

Related information:

• Step Settings Pane
Tests Tab

The Tests tab contains a table that lists the tests the step performs.

Changing the Layout and Entering Data

Change the layout of the Tests table in the following ways:

- Use the buttons located to the right of the table to add, remove, or reorder tests.
- Drag column headers to reorder columns.

Enter data in the table in the following ways:

- Enter data when a cell is highlighted.
- Click in a cell when it is highlighted.
- Double-click a cell.
- Select a value from a drop-down menu.
- Drag a variable or property from the Variables pane to a text or expression cell.

Copying and Pasting Data

You can copy and paste data in the Tests table in the following ways:

- Press <Ctrl-C> to copy the contents of selected table rows, columns, or cells to the clipboard.
- Press <Ctrl-V> to insert data from the clipboard into the table starting from the top left selected cell. If the number of rows of data on the clipboard is greater than the number of tests in the Tests table, the paste command adds new tests to the step to match the clipboard data.
- You can copy data to tests on the same <u>ECU Multi Test</u> step, to tests on other ECU Multi Test or ECU Sequence Call steps, or to a Microsoft Excel spreadsheet.
- When you modify the data in Excel and paste it back to an ECU Multi Test step, the step returns an error if the modified data is invalid for a test.

Columns

The Tests table contains the following columns:

- Test Number—The test number. If you do not specify a test number, the step assigns the value 0. For tests that correspond to a pin in a pin group, the step sets the test number to a value that equals the sum of the test number specified for the pin group test and the zero-based index of the pin in the pin group.
- **Test Name**—A string that identifies or describes the test. For tests that correspond to a pin in a pin group, the step sets the test name to a string that consists of the test name specified for the pin group test concatenated with the pin name.
- **Pin**—The pin or pin group to test.

When you specify a valid <u>pin map</u> file in the <u>Test Program Editor</u>, this cell contains a drop-down menu that includes the pin groups, DUT pins, and system pins the pin map defines. If the pin map file is invalid or if you do not specify a pin map file, this cell contains an editable string.

If you specify a pin group, the Tests table inserts a test for each pin in the pin group. These inserted tests are not editable and are identical to the test that specifies a pin group, with the following exceptions:

- The Pin column is set to the name of a pin in the pin group.
- The Test Number column is computed by adding the test number specified for the pin group test to the zero-based index of the pin in the pin group.
- The Test Name column is computed by appending the name of the pin in the pin group to the test name specified for the pin group test.

Although the step does not save the inserted tests in the sequence file, at run time the step adds and performs tests identical to these inserted tests. The pin group test does not appear at run time or in log files or reports.

Note To use a different numbering scheme for test numbers or a different naming scheme for the test name for pins in a pin group, copy the tests that were inserted for the pin group, delete the pin group test, and paste the copied tests into the Tests table. You can then edit the test numbers and test names directly.

- **Published Data Id**—(ECU Multi Test step) A string that identifies the measurement. This option is not available when you specify an expression in the Test Data Source cell. The published data ID string can be empty.
- Low Limit—A numeric expression that specifies the low limit. This option is available only when you select Numeric Limit in the Evaluation Type column.

If you specify a low limit, you must also specify a high limit. Use the Evaluation Comparison Mode option on the Options tab to configure how the step evaluates the limits.

If you do not specify a low limit and high limit, the step skips the comparison.

• **High Limit**—A numeric expression that specifies the high limit. This option is available only when you select Numeric Limit in the Evaluation Type column.

If you specify a high limit, you must also specify a low limit. Use the Evaluation Comparison Mode option on the Options tab to configure how the step evaluates the limits.

If you do not specify a low limit and high limit, the step skips the comparison.

- Scaling Factor—The <u>scaling factor</u> to use for the measurement and limit values. The step assumes that measurement values are in base units and that limit values are in scaled units. The Tests table displays all values in scaled units.
- **Base Units**—A string that specifies the base units associated with the limits and the measurement. This option is available only when you select Numeric Limit in the Evaluation Type column.
- Evaluation Type—The Numeric Limit, Pass/Fail, or None evaluation type. Use a Numeric Limit evaluation for a parametric test, and use a Pass/Fail evaluation for a functional test. Use the None evaluation type to publish data from a code module without evaluating the data.

When you select the Pass/Fail or None evaluation type, the Low Limit, High Limit, and Base Units cells are not available.

- Test Data Source—Optional. An expression that specifies the measurement data to use for the test. When you specify an expression, the Pin and Published Data Id cells are not available.
- Export Data To—Optional. An expression that specifies a location to copy the measurement data from the test. Use this expression when you need to refer to the measurement in subsequent steps. For tests that refer to a pin group, use an array expression to copy the measurement data for each pin test to an element in the array. The step automatically adjusts the size of the array to have enough elements to contain the data for each pin in the pin group.

The following columns are available only in the Execution window for an ECU Multi Test step suspended at a breakpoint during an execution:

- Status—The status of the test.
- **Data**—The measurement data associated with the test.

Related information:

- Variables Pane Sequence File Window
- Execution Window

Scaling Measurement and Limit Data

Use the Scaling Factor column in the Tests table on the <u>Tests</u> tab of the <u>ECU Multi Test</u> step to specify the scaling factor for the base units for test limits and measurements.

Changing the scaling factor in the Scaling Factor column in the Tests table sets the same scaling factor for the limits and the measurement.

The ECU Multi Test step uses the scaling factors in the following ways:

- Measurement Value Scaling Factor—The steps assume the measurement values that code modules publish use base units. At run time, the steps use the measurement scaling factor to display the measurement value in scaled units in the Data column of the Tests table. The Debug Test Results Log result processing plug-in uses the measurement scaling factor to format the values and units in the Measurement Value column of the Debug Test Results Log.
- Limits Scaling Factors—The steps assume the limit values in the Low Limit and High Limit columns use scaled units. At run time, the steps use the limit scaling factors to convert the limit values to base units to compare the measurement value to the limit values. The Debug Test Results Log result processing plug-in uses the limit scaling factors to add units to the values in the Low Limit and High Limit columns of the Debug Test Results Log.

Supported Scaling Factors

The ECU Software Toolkit recognizes the following scaling factors:

Scaling Factor Name	Scaling Factor Value	Units Prefix	DataScalingExponent, LowLimitScalingExponent, and HighLimitScalingExponent Property Values
femto	10 ⁻¹⁵	f	-15
pico	10 ⁻¹²	р	-12
nano	10 ⁻⁹	n	-9
micro	10 ⁻⁶	u	-6
milli	10 ⁻³	m	-3
percent	10 ⁻²	%	-2
<blank></blank>	1		0
kilo	10 ³	К	3
mega	10 ⁶	М	6
giga	10 ⁹	G	9
tera	10 ¹²	т	12

ECU Multi Test Options Tab

The ECU Multi Test Options tab contains several settings which control the execution of the ECU Multi Test step.

The Options tab contains the following options:

- Stop Performing Tests after First Failure—When you enable this checkbox and a test fails, the <u>ECU Multi Test</u> does not evaluate any subsequent tests in the step and sets the remaining test results to Skip.
- Evaluation Comparison Mode—Select how the step compares limits for all tests the step specifies. This drop-down menu is available only when you specify at least one test with a low and high limit on the <u>Tests</u> tab.
- **Test Numeric Display Format**—Displays the TestStand numeric format associated with the limits and measurement data. The numeric format determines how the Tests tab displays numeric values and how the TestStand report generators display

numeric values in reports. Click the **Edit** button to launch the Numeric Format dialog box, in which you can specify the numeric format.

- Multisite Option—Use this drop-down menu to specify how the step executes the step code module on multiple sites:
 - One thread per subsystem—Execute the code module for each subsystem in a separate thread. The ECU Multi Test step identifies subsystems by using the pin map and the pins and relays shown in PinMapContext Pins and Relays.
 - **One thread only**—Execute the code module for all sites in a single thread.
 - One thread per site—Execute the code module for each site in a separate thread. Use this option only when the code module does not use hardware shared among multiple sites.

Note

- ECU Software Toolkit does not synchronize executions when you use the Parallel process model
- ECU Multi Test steps in the ProcessSetup and ProcessCleanup sequences always execute as if you specified One thread only in the Multisite Option drop-down menu.
- Multisite Execution Diagram—When you associate a pin map file with the sequence file, this diagram shows the threads the step uses to execute code modules when executing with the Batch process model using multiple sites. Each rectangle represents a single thread. The numbers within the rectangle represent the sites tested in the same thread. Each thread corresponds to one of the test socket threads. The step determines which test socket thread executes the code module at run time. This diagram is available only when you chose Select Manually from the Specify Pins and Relays drop-down menu.
- Specify Pins and Relays—Specifies which pins and relays are included in the PinMapContext. Specify pins and relays manually or by using an expression.
- PinMapContext Pins and Relays—Shows the list of pins and relays or specifies the expression that determines the pins and relays that the PinMapContext object contains at run time. By default, the PinMapContext object contains all DUT pins in the pin map file and no system pins. If your code module uses any system pins, you must manually select the Include System Pins checkbox or include the system pins in your expression. Code modules that use system pins execute in one thread for all sites. By default, the PinMapContext object contains no relays. If your code module uses any system relays, you must manually select the Include

System Relays checkbox or include the system relays in your expression. If your code module uses site relays, you must manually select the **Specify Site Relays** checkbox and specify the relays or include the site relays in your expression. To specify the pins and relays manually, choose **Select Manually** from the **Specify Pins and Relays** drop-down menu and enable the options you want.

- Include System Pins—Include all system pins.
- Specify DUT Pins—Select the DUT pins or pin groups you want to include.
 When you select a pin group, the tab dims the DUT pins that belong to the pin group.
- Include System Relays—Include all system relays.
- **Specify Site Relays**—Select the relays, relay groups, or relay configurations you want to include. When you select a relay group or relay configuration, the tab dims the relays that belong to the relay group or configuration.

Note If you choose One thread only or One thread per site in the Multisite Option drop-down menu, all pins and relays are included in the PinMapContext and the checkboxes are disabled.

To specify the pins and relays using an expression, select **Use Expression** from the **Specify Pins and Relays** drop-down menu. At run time, the expression you use must evaluate to an array of DUT and system pin names, and site and system relay names.



Note The PinMapContext object for steps in the ProcessSetup and ProcessCleanup sequences always contains all DUT and system pins and all site and system relays.

Related information:

- Numeric Format Dialog Box
- Parallel Process Model
- <u>Batch Process Model</u>
- <u>Process Model Thread Types</u>

ECU Action Step

Use the ECU Action step to perform an action, such as instrument configuration, with

access to the pin map and per-site inputs. You can configure <u>multisite</u> options directly on the step.

Note You cannot use multiple ECU Multi Test steps or ECU Action steps configured to use multiple threads in While loops, in Do While loops, or in For loops that use the Custom Loop option when performing multisite testing. The steps report a run-time error in these situations. Use other types of loops instead, such as For loops that use the Fixed Number of Iterations option.

Configuring the Step

Use the ECU Action step <u>edit tabs</u> in the TestStand Sequence Editor to specify the multisite and per-site input options.

Related information:

- While Step
- <u>Do While Step</u>
- For Step
- For Loop Edit Tab

ECU Action Step Execution Overview

The ECU Action step uses the following process when executing:

- Creates a PinMapContext object and stores it in the Step.PinMapContext property if the site that executes the step is the working site. The PinMapContext object describes a subset of pins, relays, sites, and instruments on a test system.
- 2. Calls a sequence that completes the following actions:
 - a. Receives the Step.PinMapContext property as an input parameter.
 - b. Uses Measurement Configuration group step types to configure and control instruments. These step types automatically obtain instrument channel and session information based on your specified Pin Map context and Pins. Choose the appropriate Measurement Configuration group steps based on the instruments used in your test and your test requirements.

ECU Action Edit Tabs

Use the <u>ECU Action</u> step edit tabs in the TestStand Sequence Editor to specify the multisite and per-site input options for the ECU Action step.

The Step Settings pane for the ECU Action step contains the following tabs:

• <u>Options</u>—Configure <u>multisite</u> and other options for the step.

Related information:

• Step Settings Pane

ECU Action Options Tab

The ECU Action Options tab contains several settings which control the execution of the ECU Action step.

The Options tab contains the following options:

- Multisite Option—Select one of the following options to specify how the step executes the step code module on multiple sites:
 - One thread per subsystem—Execute the code module for each subsystem in a separate thread. The ECU Action step identifies subsystems by using the pin map and the pins shown in PinMapContext Pins and Relays.
 - **One thread only**—Execute the code module for all sites in a single thread.
 - One thread per site—Execute the code module for each site in a separate thread. Use this option only when the code module does not use hardware shared among multiple sites.

Note

- The ECU Software Toolkit does not synchronize executions when you use the Parallel process model.
- ECU Action steps in the ProcessSetup and ProcessCleanup sequences always execute as if you specified One thread only in the Multisite Option drop-down menu.
- Multisite Execution Diagram—When you associate a pin map file with the

sequence file, this diagram shows the threads the step uses to execute code modules when executing with the Batch process model using multiple sites. Each rectangle represents a single thread. The numbers within the rectangle represent the sites tested in the same thread. Each thread corresponds to one of the test socket threads. The step determines which test socket thread executes the code module at run time. This diagram is available only when you chose **Select Manually** from the **Specify Pins and Relays** drop-down menu.

- Specify Pins and Relays—Specifies which pins and relays are included in the PinMapContext. Specify pins and relays manually or by using an expression.
- **PinMapContext Pins and Relays**—Shows the list of pins and relays or specifies the expression that determines the pins and relays that the PinMapContext object contains at run time.

By default, the PinMapContext object contains all DUT pins in the pin map file and no system pins. If your code module uses any system pins, you must manually select the **Include System Pins** checkbox or include the system pins in your expression. Code modules that use system pins execute in one thread for all sites.

By default, the PinMapContext object contains no relays. If your code module uses any system relays, you must manually select the Include System Relays checkbox or include the system relays in your expression. If your code module uses site relays, you must manually select the Specify Site Relays checkbox and specify the relays or include the site relays in your expression.

To specify the pins and relays manually, choose **Select Manually** from the **Specify Pins and Relays** drop-down menu and enable the options you want.

- Include System Pins—Include all system pins.
- Specify DUT Pins—Select the DUT pins or pin groups you want to include.
 When you select a pin group, the tab dims the DUT pins that belong to the pin group.
- Include System Relays—Include all system relays.
- **Specify Site Relays**—Select the relays, relay groups, or relay configurations you want to include. When you select a relay group or relay configuration, the tab dims the relays that belong to the relay group or configuration.

Note If you choose One thread only or One thread per site in the

Multisite Option drop-down menu, all pins and relays are included in the PinMapContext and the checkboxes are disabled.

To specify the pins and relays using an expression, select **Use Expression** from the **Specify Pins and Relays** drop-down menu. At run time, the expression you use must evaluate to an array of DUT and system pin names, and site and system relay names.



Note The PinMapContext object for steps in the ProcessSetup and ProcessCleanup sequences always contains all DUT and system pins and all site and system relays.

Related information:

- Parallel Process Model
- <u>Batch Process Model</u>
- Process Model Thread Types

SLSC 12251/2 Group

Steps in the SLSC 12251/2 group are used to control current measurements on SLSC FIU channels. These steps can be found at located at **IO Configuration » Instrument Control » SLSC 12251/2** in the Step Types list.

SLSC FIU - Read Current Step

Use the SLSC FIU - Read Current step to measure the current through specified SLSC FIU channels and check limits for the current measurements.

A single SLSC FIU - Read Current step can specify multiple SLSC FIU channels. This step integrates the Multiple Numeric Limit Test step.

SLSC FIU - Read Current Step Settings Tab

The <u>SLSC FIU - Read Current Step</u> Settings tab contains the following options:

• Switch Executive Virtual Device—Specifies the Switch Executive virtual device created based on the SLSC FIU device in NI MAX. If the virtual device has not been

created, click the Launch Switch Executive button # to launch NI MAX and create the virtual device using NI Switch Executive.

- **CHANNEL and ACQUIRED DATA Table**—Table containing SLSC FIU current measurement information.
 - CHANNEL: Column which specifies the SLSC FIU channels you want to measure current on. Click the Add Channel button to add an SLSC FIU channel, and click the Remove Selected Channel(s) button to remove channels you have selected.
 - ACQUIRED DATA: Column which specifies the variables used to store current measurements. You can use these variables to reference measurement data in subsequent steps.

The Limits and Data Source tabs contain the same settings as the Limits and Data Source tabs in the TestStand Multiple Numeric Limit Test step.

SLSC FIU - Reset Step

Use the SLSC FIU – Reset step to return the SLSC FIU to the device initialization state.

A single SLSC FIU – Reset step resets one Switch Executive virtual device created based on an SLSC FIU device.

SLSC FIU - Reset Step Settings Tab

The <u>SLSC FIU - Reset Step</u> Settings tab contains the following options:

• Switch Executive Virtual Device—Specifies the Switch Executive virtual device to reset.

Measurement Configuration Group

Steps in the Measurement Configuration group accomplish tasks specific to an instrument type, or are used to configure pin settings and instrument driver sessions.

Create Pin Sessions Step

Use the Create Pin Sessions step to initialize the Pin Map Context with the driver sessions for instruments.

Instruments must be defined in the pin map file, used in at least one pin connection, and used in a site enabled in the <u>Configure Lot Settings</u> dialog box for a driver session to be created. This step automatically uses the pin map file you specified or created for the current TestStand sequence file.

Note The Create Pin Sessions step supports RMX-410x power supplies and instruments programmed using NI-FGEN, NI-DCPower, NI-DMM, or NI-SCOPE.

Apply Pin Configuration Step

Use the Apply Pin Configuration step to reset the instrument to its default settings and configure the selected pins with pin settings from the corresponding Measurement Configuration file.

Use the Apply Pin Configuration Step edit tab in the TestStand Sequence Editor to specify the pins you want to configure and the measurement configuration file you want to apply.

Related information:

• TestStand Sequence Editor

Apply Pin Configuration Step Settings Tab

The <u>Apply Pin Configuration Step</u> Settings tab contains the following options:

- Measurement Configuration Path—Specifies the path to the measurement configuration file (.measconfig) exported from InstrumentStudio to apply.
- Pins to Configure—Specifies the pins to configure using the settings from the measurement configuration file defined by the Measurement Configuration Path. The specified pins must be defined in either the Pin Map Context or the pin map file or this step will return an error.
- Pin Map Context—Specifies a reference to a Pin Map Context object. This object describes a subset of pins, sites, and instruments on a test system. The Pin Map Context object is used in this step to associate pins specified in the Pins to Configure option with pin configurations from the measurement configuration file defined by the Measurement Configuration Path option.

Close Pin Sessions Step

Use the Close Pin Sessions step to close all instrument driver sessions initialized by the <u>Create Pin Sessions</u> step.

This step uses the <u>pin map file</u> you specified for the current TestStand sequence.

DMM - Read Measurement Step

Use the DMM – Read Measurement step to read, publish and return the digital multimeter measurement results for the specified DMM pins on all sites in the Pin Map context.

This step publishes measurement data for the specified pins to the <u>ECU Multi Test</u> step type instances with their module adapter configured as Sequence for all sites in the Pin Map context.

Note The DMM - Read Measurement step must be used inside of an <u>ECU</u> <u>Multi Test</u> or <u>ECU Action</u> step with its module adapter configured as Sequence. The DMM - Read Measurement step requires the Pin Map context passed from these steps as an input.

Use the DMM – Read Measurement step edit tab in the TestStand Sequence Editor to specify the Pin Map context and the pins to query to obtain their associated NI-DMM instrument sessions.

Related information:

• TestStand Sequence Editor

DMM - Read Measurement Step Settings Tab

The <u>DMM - Read Measurement Step</u> Settings tab contains the following options:

- **Pin Map Context**—Specifies a reference to a Pin Map Context object. This object describes a subset of pins, sites, and instruments from the <u>pin map file</u> and provides data pertinent to each site executing during a test.
- **Pins**—Specifies the pins to query to obtain their associated NI-DMM instrument sessions. These instrument sessions are used to read measurement results from

the specified pins. The specified pins must be defined in the <u>pin map file</u> and included in the Pin Map context Pins and Relays list in the <u>Options</u> tab of the ECU Multi Test step.

Note The <u>DMM - Read Measurement</u> step filters the pins specified by instrument type ID. If none of the pins specified have an instrument type ID of niDMM, the step will do nothing, as none of the specified pins are connected to NI-DMM instruments.

- Published Data ID—The unique ID for a measurement. This can be used to distinguish measurements when you publish multiple measurements for a single pin within the same low-level sequence. The default published data ID is Measurement.
- Measurements (Optional)—An expression that specifies a location to store the measurement data for one or more pins on all sites in the Pin Map context. The measurement results are organized as a two-dimensional array of Number values. Each array row corresponds to a pin, with the first row representing the first valid pin in the list of pins defined by Pins. Each array column corresponds to a site, with column indexes corresponding with the site order in the Pin Map Context. Use a two-dimensional array of Number values to store the measurement results for use in subsequent steps.

FGen - Initiate Generation Step

Use the FGen - Initiate Generation step to initiate waveform generation for the specified FGen pins on all sites in the Pin Map context.

Note The FGen - Initiate Generation step must be used inside of an <u>ECU Multi</u> <u>Test</u> or <u>ECU Action</u> step with its module adapter configured as Sequence. The FGen - Initiate Generation step requires the Pin Map context passed from these steps as an input.

Use the FGen - Initiate Generation step edit tab in the TestStand Sequence Editor to specify the Pin Map context and the pins to query to obtain their associated NI-FGEN instrument sessions.

Related information:

• TestStand Sequence Editor

FGen - Initiate Generation Step Settings Tab

The FGen - Initiate Generation Step Settings tab contains the following options:

- **Pin Map Context**—Specifies a reference to a Pin Map Context object. This object describes a subset of pins, sites, and instruments from the <u>pin map file</u> and provides data pertinent to each site executing during a test.
- **Pins**—Specifies the pins to query to obtain their associated NI-FGEN instrument sessions. These instrument sessions are used to initiate waveform generation on the specified pins. The specified pins must be defined in the <u>pin map file</u> and included in the Pin Map context Pins and Relays list in the <u>Options</u> tab of the ECU Multi Test step.

Note The <u>FGen - Initiate Generation</u> step filters the pins specified by instrument type ID. If none of the pins specified have an instrument type ID of niFGen, the step will do nothing, as none of the specified pins are connected to NI-FGEN instruments.

FGen - Abort Generation Step

Use the FGen - Abort Generation step to abort waveform generation for the specified FGen pins on all sites in the Pin Map context.

Note The FGen - Abort Generation step must be used inside of an <u>ECU Multi</u> <u>Test</u> or <u>ECU Action</u> step with its module adapter configured as Sequence. The FGen - Abort Generation step requires the Pin Map context passed from these steps as an input.

Use the FGen - Initiate Generation step edit tab in the TestStand Sequence Editor to specify the Pin Map context and the pins to query to obtain their associated NI-FGEN instrument sessions.

Related information:

• TestStand Sequence Editor

FGen - Abort Generation Step Settings Tab

The FGen - Abort Generation Step Settings tab contains the following options:

- **Pin Map Context**—Specifies a reference to a Pin Map Context object. This object describes a subset of pins, sites, and instruments from the <u>pin map file</u> and provides data pertinent to each site executing during a test.
- **Pins**—Specifies the pins to query to obtain their associated NI-FGEN instrument sessions. These instrument sessions are used to abort waveform generation on the specified pins. The specified pins must be defined in the <u>pin map file</u> and included in the Pin Map context Pins and Relays list in the <u>Options</u> tab of the ECU Multi Test step.

Note The <u>FGen - Abort Generation</u> step filters the pins specified by instrument type ID. If none of the pins specified have an instrument type ID of niFGen, the step will do nothing, as none of the specified pins are connected to NI-FGEN instruments.

FGen - Enable Output Step

Use the FGen - Enable Output step to enable waveform generation for the specified FGen pins on all sites in the Pin Map context.

Note The FGen - Enable Output step must be used inside of an <u>ECU Multi</u> <u>Test</u> or <u>ECU Action</u> step with its module adapter configured as Sequence. The FGen - Enable Output step requires the Pin Map context passed from these steps as an input.

Use the FGen - Enable Output step edit tab in the TestStand Sequence Editor to specify the Pin Map context and the pins to query to obtain their associated NI-FGEN instrument sessions.

Related information:

<u>TestStand Sequence Editor</u>

FGen - Enable Output Step Settings Tab

The FGen - Enable Output Step Settings tab contains the following options:

- **Pin Map Context**—Specifies a reference to a Pin Map Context object. This object describes a subset of pins, sites, and instruments from the <u>pin map file</u> and provides data pertinent to each site executing during a test.
- **Pins**—Specifies the pins to query to obtain their associated NI-FGEN instrument sessions. These instrument sessions are used to enable waveform generation on the specified pins. The specified pins must be defined in the <u>pin map file</u> and included in the Pin Map context Pins and Relays list in the <u>Options</u> tab of the ECU Multi Test step.

Note The <u>FGen - Enable Output</u> step filters the pins specified by instrument type ID. If none of the pins specified have an instrument type ID of niFGen, the step will do nothing, as none of the specified pins are connected to NI-FGEN instruments.

FGen - Disable Output Step

Use the FGen - Disable Output step to disable waveform generation for the specified FGen pins on all sites in the Pin Map context.

Note The FGen - Disable Output step must be used inside of an <u>ECU Multi</u> <u>Test</u> or <u>ECU Action</u> step with its module adapter configured as Sequence. The FGen - Disable Output step requires the Pin Map context passed from these steps as an input.

Use the FGen - Disable Output step edit tab in the TestStand Sequence Editor to specify the Pin Map context and the pins to query to obtain their associated NI-FGEN instrument sessions.

Related information:

• TestStand Sequence Editor

FGen - Disable Output Step Settings Tab

The <u>FGen - Disable Output Step</u> Settings tab contains the following options:

- **Pin Map Context**—Specifies a reference to a Pin Map Context object. This object describes a subset of pins, sites, and instruments from the <u>pin map file</u> and provides data pertinent to each site executing during a test.
- **Pins**—Specifies the pins to query to obtain their associated NI-FGEN instrument sessions. These instrument sessions are used to disable waveform generation on the specified pins. The specified pins must be defined in the <u>pin map file</u> and included in the Pin Map context Pins and Relays list in the <u>Options</u> tab of the ECU Multi Test step.

Note The <u>FGen - Disable Output</u> step filters the pins specified by instrument type ID. If none of the pins specified have an instrument type ID of niFGen, the step will do nothing, as none of the specified pins are connected to NI-FGEN instruments.

Scope - Initiate Acquisition Step

Use the Scope - Initiate Acquisition step to initiate waveform acquisition for the specified Scope pins on all sites in the Pin Map context.

After this step is called, NI-SCOPE instruments connected to pins in the specified list leave the Idle state and wait for a trigger.

Note The Scope - Initiate Acquisition step must be used inside of an <u>ECU</u> <u>Multi Test</u> or <u>ECU Action</u> step with its module adapter configured as Sequence. The Scope - Initiate Acquisition step requires the Pin Map context passed from these steps as an input.

Use the Scope - Initiate Acquisition step edit tab in the TestStand Sequence Editor to specify the Pin Map context and the pins to query to obtain their associated NI-SCOPE instrument sessions.

Related information:

• <u>TestStand Sequence Editor</u>

Scope - Initiate Acquisition Step Settings Tab

The <u>Scope - Initiate Acquisition Step</u> Settings tab contains the following options:

- **Pin Map Context**—Specifies a reference to a Pin Map Context object. This object describes a subset of pins, sites, and instruments from the <u>pin map file</u> and provides data pertinent to each site executing during a test.
- **Pins**—Specifies the pins to query to obtain their associated NI-SCOPE instrument sessions. These instrument sessions are used to initiate waveform acquisition on the specified pins. The specified pins must be defined in the <u>pin map file</u> and included in the Pin Map context Pins and Relays list in the <u>Options</u> tab of the ECU Multi Test step.

Note The <u>Scope - Initiate Acquisition</u> step filters the pins specified by instrument type ID. If none of the pins specified have an instrument type ID of niScope, the step will do nothing, as none of the specified pins are connected to NI-SCOPE instruments.

Scope - Read Measurements Step

Use the Scope - Read Measurements step to read and publish the Scope measurement results for the specified Scope pins on all sites in the Pin Map context.

This step publishes measurement data for the specified pins to the <u>ECU Multi Test</u> step type instances with their module adapter configured as Sequence for all sites in the Pin Map context.

Note The Scope - Read Measurements step must be used inside of an <u>ECU</u> <u>Multi Test</u> or <u>ECU Action</u> step with its module adapter configured as Sequence. The Scope - Read Measurements step requires the Pin Map context passed from these steps as an input.

Use the Scope - Read Measurements step edit tab in the TestStand Sequence Editor to specify the Pin Map context and the pins to query to obtain their associated NI-SCOPE instrument sessions.

Related information:

- TestStand Sequence Editor
- InstrumentStudio Measurement Types

Scope - Read Measurements Step Settings Tab

The <u>Scope - Read Measurements Step</u> Settings tab contains the following options:

- Measurement Configuration Path—Specifies the path to the measurement configuration file (.measconfig) exported from InstrumentStudio to apply.
- **Pin Map Context**—Specifies a reference to a Pin Map Context object. This object describes a subset of pins, sites, and instruments from the <u>pin map file</u> and provides data pertinent to each site executing during a test.
- **Pins**—Specifies the pins to query to obtain their associated NI-SCOPE instrument sessions. These instrument sessions are used to read measurements on the specified pins. The specified pins must be defined in the <u>pin map file</u> and included in the Pin Map context Pins and Relays list in the <u>Options</u> tab of the ECU Multi Test step.

Note The <u>Scope - Read Measurements</u> step filters the pins specified by instrument type ID. If none of the pins specified have an instrument type ID of niScope, the step will do nothing, as none of the specified pins are connected to NI-SCOPE instruments.

- **Timeout**—Specifies the time in seconds to wait for data to be acquired.
- **Measurement Table**—Table containing measurement types obtained from the measurement configuration file. A list of all supported scope measurements can be found in the InstrumentStudio help manual.
 - **MEASUREMENT**: Column containing the measurement types obtained from the measurement configuration file.
 - **SOURCE**: Column containing the sources for measurement type entries. Three types of sources are valid:
 - Pins: Name of the source pin for a measurement type. If multiple pins are the source for a measurement type, pin names are delimited by a semicolon.
 - Math Channels: Name of the math channel source for a measurement type. If multiple math channels are the source for a measurement type, each math channel source will appear on a different row. If a math channel is the source for multiple measurement types, each measurement type sourced from that channel will appear on a different row.

- Dual Sources: dual source measurement types such as Crosspoint Voltage, Delta Time, Setup Time, and Hold Time have two sources. Dual Source names are formatted as [SourceA] / [SourceB]. Dual source measurement types can have both pins and math channels as sources, such as [Math1] / [PinA]. Additionally, if the scope channels for the measurement type are connected to different DUT pins using Switch Executive, SourceA and SourceB can be lists of pins delimited by commas, such as [PinA, PinB] / [PinC, PinD].
- **TYPE**:Column containing the type of the **PUBLISHED DATA ID**.
- PUBLISHED DATA ID: Column containing the unique ID for distinguishing measurements when you publish multiple measurements for a single pin within the same low-level sequence. The default published data ID is Measurement and is populated for every measurement type entry if a different ID is not specified.
- **COMMENT**: Column containing comments and descriptions for measurement type entries.
- Math Channel Table—Table containing math channels obtained from the measurement configuration file.
 - MATH CHANNEL: Column containing the math channels obtained from the measurement configuration file.
 - **SOURCE**: Column containing the source names for math channel entries. The format of the data in this column will change according to the math operation being conducted on the two sources for the math channel, SourceA and SourceB:

Operation	Source Displays As
A + B	[SourceA]+[SourceB]
A – B	[SourceA]-[SourceB]
A* B	[SourceA]*[SourceB]
А / В	[SourceA]/[SourceB]
(A + B) / 2	([SourceA] + [SourceB]) / 2
(<i>A</i> – <i>B</i>) / 2	([SourceA] – [SourceB]) / 2

SourceA and SourceB can be a single pin, or, if the math channels are

connected to different DUT pins using Switch Executive, can be lists of pins delimited by commas. These two formats are shown below:

Single Pins:	[PinA]+[PinB]					
Multiple Pins:	[PinA, PinB]+[PinC, PinD]					

• **COMMENT**: Column containing comments and descriptions for math channel entries.

The process for configuring the **Pin** column in the <u>ECU Multi Test</u> step varies depending on the source of the measurement type.

- For measurement types with Pin sources, select the appropriate Pin name from the **Pin** column drop-down menu.
- For measurement types with Math Channel sources, leave the **Pin** column blank.
- Measurement types with Dual Sources use sources which are formatted as
 [SourceA] / [SourceB]. If SourceA is a Pin, select the appropriate Pin name from the
 Pin column drop-down menu. If SourceA is a Math Channel, leave the Pin column
 blank.

ECU Multi Test Step Tests Tab Pin Configuration

The following figure displays an example **Pin** column configuration in the ECU Multi Test Step <u>Tests Tab</u>.

x x

Step	Settin	gs for	Pin	E and	F	Signal	Check
-				-			

	Test Number	Test Name	Pin		Published Data Id	Low Limit	High Limit	Scaling Factor		Base Units	Evaluation Type		Test Data Source	Export Data To
4	400	Amplitude CHK - E	E_SCOPE		Amplitude	17 V	21 V		٠	v	Numeric Limit	٠		
4	401	Frequency CHK - E	E_SCOPE		Frequency	9 Hz	11 Hz		٠	Hz	Numeric Limit	٠		
4	402	Amplitude CHK - F	F_SCOPE		Amplitude	17 V	21 V		٠	v	Numeric Limit	٠		
4	403	Frequency CHK - F	F_SCOPE		Frequency	9 Hz	11 Hz		٠	Hz	Numeric Limit	٠		
4	404	Amplitude CHK - Math 1 (E - F)			Amplitude_Math1	36 V	40 V		٠	v	Numeric Limit	*		
4	405	Frequency CHK - Math 1 (E - F)		٠	Frequency_Math1	9 Hz	11 Hz		٠	Hz	Numeric Limit	٠		
; 4	406	Crosspoint Voltage CHK - E / F	E_SCOPE	٠	CrosspointVoltage	13 V	18 V		٠	v	Numeric Limit	٠		
1 4	407	Crosspoint Voltage CHK - F / E	F_SCOPE		CrosspointVoltage	13 V	18 V		٠	v	Numeric Limit	٠		
4	408	Setup Time CHK - Math 1 / F			SetupTime_Math1	40 mS	60 mS	mili		S	Numeric Limit			

- Tests number 400 through 403 have Pin sources, so the **Pin** column is populated with names.
- Tests number 404 and 405 have Math Channel sources, so the **Pin** column is left blank.
- Tests number 406 through 408 have Dual Source sources. For Tests 406 and 507, SourceA is a Pin, so the **Pin** column is populated with names. For Test 408, SourceA is a Math Channel, so the **Pin** column is left blank.

Scope - Abort Acquisition Step

Use the Scope - Abort Acquisition step to abort waveform acquisition for the specified Scope pins on all sites in the Pin Map context.

After this step is called, NI-SCOPE instruments connected to pins in the specified list move to the Idle state.

Note The Scope - Abort Acquisition step must be used inside of an <u>ECU Multi</u> <u>Test</u> or <u>ECU Action</u> step with its module adapter configured as Sequence. The Scope - Initiate Acquisition step requires the Pin Map context passed from these steps as an input.

Use the Scope - Abort Acquisition step edit tab in the TestStand Sequence Editor to specify the Pin Map context and the pins to query to obtain their associated NI-SCOPE instrument sessions.

Related information:

• TestStand Sequence Editor

Scope - Abort Acquisition Step Settings Tab

The <u>Scope - Abort Acquisition Step</u> Settings tab contains the following options:

- **Pin Map Context**—Specifies a reference to a Pin Map Context object. This object describes a subset of pins, sites, and instruments from the <u>pin map file</u> and provides data pertinent to each site executing during a test.
- **Pins**—Specifies the pins to query to obtain their associated NI-SCOPE instrument sessions. These instrument sessions are used to abort waveform acquisition on the specified pins. The specified pins must be defined in the <u>pin map file</u> and included in the Pin Map context Pins and Relays list in the <u>Options</u> tab of the ECU Multi Test step.

Note The <u>Scope - Abort Acquisition</u> step filters the pins specified by instrument type ID. If none of the pins specified have an instrument type ID of niScope, the step will do nothing, as none of the specified pins are connected to NI-SCOPE instruments.

SMU/PPS - Initiate Step

Use the SMU/PPS - Initiate step to start generation or acquisition for the specified SMU/PPS pins on all sites in the Pin Map context.

This step causes the NI-DCPower instrument channels connected to the specified SMU/ PPS pins on all sites in the Pin Map context to leave the Uncommitted or Committed state and enter the Running state.

Note The SMU/PPS Initiate step must be used inside of an <u>ECU Multi Test</u> or <u>ECU Action</u> step with its module adapter configured as Sequence. The SMU/ PPS - Initiate step requires the Pin Map context passed from these steps as an input.

Use the SMU/PPS - Initiate step edit tab in the TestStand Sequence Editor to specify the Pin Map context and the pins to query to obtain their associated NI-DCPower instrument sessions.

Related information:

• TestStand Sequence Editor

SMU/PPS - Initiate Step Settings Tab

The <u>SMU/PPS Initiate Step</u> Settings tab contains the following options:

- **Pin Map Context**—Specifies a reference to a Pin Map Context object. This object describes a subset of pins, sites, and instruments from the <u>pin map file</u> and provides data pertinent to each site executing during a test.
- Pins—Specifies the pins to query to obtain their associated NI-DCPower instrument sessions. These instrument sessions are used to initiate waveform generation on the specified pins. The specified pins must be defined in the <u>pin</u> <u>map file</u> and included in the Pin Map context Pins and Relays list in the <u>Options</u> tab of the ECU Multi Test step.

Note The <u>SMU/PPS Initiate</u> step filters the pins specified by instrument type ID. If none of the pins specified have an instrument type ID of <code>niDCPower</code>, the step will do nothing, as none of the specified pins are

connected to NI-DCPower instruments.

SMU/PPS Abort Step

Use the SMU/PPS - Abort step to transition the specified SMU/PPS pins on all sites in the Pin Map context from the Running state to the Uncommitted state and terminate any running sequences for those pins.

Note The SMU/PPS Abort step must be used inside of an <u>ECU Multi Test</u> or <u>ECU Action</u> step with its module adapter configured as Sequence. The SMU/ PPS - Abort step requires the Pin Map context passed from these steps as an input.

Use the SMU/PPS - Abort step edit tab in the TestStand Sequence Editor to specify the Pin Map context and the pins to query to obtain their associated NI-DCPower instrument sessions.

Related information:

• TestStand Sequence Editor

SMU/PPS - Abort Step Settings Tab

The <u>SMU/PPS Abort Step</u> Settings tab contains the following options:

- **Pin Map Context**—Specifies a reference to a Pin Map Context object. This object describes a subset of pins, sites, and instruments from the <u>pin map file</u> and provides data pertinent to each site executing during a test.
- Pins—Specifies the pins to query to obtain their associated NI-DCPower instrument sessions. These instrument sessions are used to initiate waveform generation on the specified pins. The specified pins must be defined in the <u>pin</u> <u>map file</u> and included in the Pin Map context Pins and Relays list in the <u>Options</u> tab of the ECU Multi Test step.

Note The <u>SMU/PPS Abort</u> step filters the pins specified by instrument type ID. If none of the pins specified have an instrument type ID of <code>niDCPower</code>, the step will do nothing, as none of the specified pins are

connected to NI-DCPower instruments.

SMU/PPS Enable Output Step

Use the SMU/PPS - Enable Output step to enable generation for the specified SMU/PPS pins on all sites in the Pin Map context.

If the NI-DCPower instrument channels connected to the specified SMU/PPS pins on all sites in the specified Pin Map context are in the Uncommitted state, this step does not take effect until you call the <u>SMU/PPS – Initiate</u> step.

Note The SMU/PPS Enable Output step must be used inside of an <u>ECU Multi</u> <u>Test</u> or <u>ECU Action</u> step with its module adapter configured as Sequence. The SMU/PPS - Enable Output step requires the Pin Map context passed from these steps as an input.

Use the SMU/PPS - Enable Output step edit tab in the TestStand Sequence Editor to specify the Pin Map context and the pins to query to obtain their associated NI-DCPower instrument sessions.

Related information:

<u>TestStand Sequence Editor</u>

SMU/PPS - Enable Output Step Settings Tab

The <u>SMU/PPS Enable Output Step</u> Settings tab contains the following options:

- **Pin Map Context**—Specifies a reference to a Pin Map Context object. This object describes a subset of pins, sites, and instruments from the <u>pin map file</u> and provides data pertinent to each site executing during a test.
- **Pins**—Specifies the pins to query to obtain their associated NI-DCPower instrument sessions. These instrument sessions are used to enable generation on the specified pins. The specified pins must be defined in the <u>pin map file</u> and included in the Pin Map context Pins and Relays list in the <u>Options</u> tab of the ECU Multi Test step.

Note The <u>SMU/PPS Enable Output</u> step filters the pins specified by instrument type ID. If none of the pins specified have an instrument type ID of niDCPower, the step will do nothing, as none of the specified pins are connected to NI-DCPower instruments.

SMU/PPS - Disable Output Step

Use the SMU/PPS - Disable Output step to disable generation for the specified SMU/PPS pins on all sites in the Pin Map context.

If the NI-DCPower instrument channels connected to the specified SMU/PPS pins on all sites in the specified Pin Map context are in the Uncommitted state, this step does not take effect until you call the <u>SMU/PPS – Initiate</u> step.

Note The SMU/PPS Disable Output step must be used inside of an <u>ECU Multi</u> <u>Test</u> or <u>ECU Action</u> step with its module adapter configured as Sequence. The SMU/PPS - Disable Output step requires the Pin Map context passed from these steps as an input.

Use the SMU/PPS - Disable Output step edit tab in the TestStand Sequence Editor to specify the Pin Map context and the pins to query to obtain their associated NI-DCPower instrument sessions.

Related information:

• TestStand Sequence Editor

SMU/PPS - Disable Output Step Settings Tab

The <u>SMU/PPS - Disable Output Step</u> Settings tab contains the following options:

- **Pin Map Context**—Specifies a reference to a Pin Map Context object. This object describes a subset of pins, sites, and instruments from the <u>pin map file</u> and provides data pertinent to each site executing during a test.
- Pins—Specifies the pins to query to obtain their associated NI-DCPower instrument sessions. These instrument sessions are used to disable generation on the specified pins. The specified pins must be defined in the <u>pin map file</u> and included in the Pin Map context Pins and Relays list in the <u>Options</u> tab of the ECU

Multi Test step.

Note The <u>SMU/PPS - Disable Output</u> step filters the pins specified by instrument type ID. If none of the pins specified have an instrument type ID of niDCPower, the step will do nothing, as none of the specified pins are connected to NI-DCPower instruments.

SMU/PPS - Read Measurement Step

Use the SMU/PPS - Read Measurement step to read, publish and return the voltage, current, and power measurement results for the specified SMU/PPS pins on all sites in the Pin Map context.

This step publishes measurement data for the specified pins to the <u>ECU Multi Test</u> step type instances with their module adapter configured as Sequence for all sites in the Pin Map context.

If the DCPowerMeasurementConfiguration.MeasureWhen property is set to DCPowerMeasurementWhen.OnDemand, this step reads measurements using the DCPowerMeasurement.Measure method.Otherwise, this step reads a measurement from the NI-DCPower buffer using the

DCPowerMeasurement.Fetch method.You must call the <u>SMU/PPS – Initiate</u> step before calling this step while using the DCPowerMeasurement.Fetch method to obtain measurements.

Note The SMU/PPS - Read Measurement step must be used inside of an <u>ECU</u> <u>Multi Test</u> or <u>ECU Action</u> step with its module adapter configured as Sequence. The SMU/PPS - Read Measurement step requires the Pin Map context passed from these steps as an input.

Use the SMU/PPS - Read Measurement step edit tab in the TestStand Sequence Editor to specify the Pin Map context and the pins to query to obtain their associated NI-DCPower instrument sessions.

Related information:

• TestStand Sequence Editor

SMU/PPS - Read Measurement Step Settings Tab

The <u>SMU/PPS - Read Measurement Step</u> Settings tab contains the following options:

- **Pin Map Context**—Specifies a reference to a Pin Map Context object. This object describes a subset of pins, sites, and instruments from the <u>pin map file</u> and provides data pertinent to each site executing during a test.
- Pins—Specifies the pins to query to obtain their associated NI-DCPower instrument sessions. These instrument sessions are used to read voltage, current, and power measurements on the specified pins. The specified pins must be defined in the <u>pin map file</u> and included in the Pin Map context Pins and Relays list in the <u>Options</u> tab of the ECU Multi Test step.

Note The <u>SMU/PPS - Read Measurement</u> step filters the pins specified by instrument type ID. If none of the pins specified have an instrument type ID of niDCPower, the step will do nothing, as none of the specified pins are connected to NI-DCPower instruments.

• Timeout—Specifies the maximum time allowed for the DCPowerMeasurement.Fetch operation to complete. The default timeout is set to 5 seconds.

Note This parameter is not used when the DCPowerMeasurementConfiguration.MeasureWhen property is set to DCPowerMeasurementWhen.OnDemand.

- Published Data ID Voltage— Specifies the unique ID for distinguishing measurements when you publish multiple measurements for a single pin within the same step or low-level sequence. The published data ID is represented as a constant string value. The default published data ID for SMU/PPS voltage measurements is Voltage.
- Published Data ID Current— Specifies the unique ID for distinguishing measurements when you publish multiple measurements for a single pin within the same step or low-level sequence. The published data ID is represented as a constant string value. The default published data ID for SMU/PPS current measurements is Current.
- **Published Data ID Power** Specifies the unique ID for distinguishing measurements when you publish multiple measurements for a single pin within the same step or low-level sequence. The published data ID is represented as a

constant string value. The default published data ID for SMU/PPS power measurements is Power.

- Voltage—Specifies a location to store SMU/PPS voltage measurement results for the specified SMU/PPS pins on all sites in the Pin Map context.
- **Current**—Specifies a location to store SMU/PPS current measurement results for the specified SMU/PPS pins on all sites in the Pin Map context.
- **Power**—Specifies a location to store SMU/PPS power measurement results for the specified SMU/PPS pins on all sites in the Pin Map context.

The measurement results are organized as a two-dimensional array of Number values. Each array row corresponds to a pin, with the first row representing the first valid pin in the list of pins defined by **Pins**. Each array column corresponds to a site, with column indexes corresponding with the site order in the **Pin Map Context**. Use a twodimensional array of Number values to store the measurement results for use in subsequent steps.

SMU/PPS - Reset Device Step

Use the SMU/PPS - Reset Device step to reset all instruments connected to the specified SMU/PPS pins on all sites in the Pin Map context to a known state.

This step disables power generation, resets all properties, and clears errors such as overtemperature and unexpected loss of auxiliary power for all instruments connected to the specified SMU/PPS pins on all sites in the Pin Map context.

Note The SMU/PPS Reset Device step must be used inside of an <u>ECU Multi</u> <u>Test</u> or <u>ECU Action</u> step with its module adapter configured as Sequence. The SMU/PPS - Reset Device step requires the Pin Map context passed from these steps as an input.

Use the SMU/PPS - Reset Device step edit tab in the TestStand Sequence Editor to specify the Pin Map context and the pins to query to obtain their associated NI-DCPower instrument sessions.

Related information:

• <u>TestStand Sequence Editor</u>

SMU/PPS - Reset Device Step Settings Tab

The <u>SMU/PPS - Reset Device Step</u> Settings tab contains the following options:

- **Pin Map Context**—Specifies a reference to a Pin Map Context object. This object describes a subset of pins, sites, and instruments from the <u>pin map file</u> and provides data pertinent to each site executing during a test.
- Pins—Specifies the pins to query to obtain their associated NI-DCPower instrument sessions. These instrument sessions are used to reset all instruments connected to the specified pins to a known state. The specified pins must be defined in the <u>pin map file</u> and included in the Pin Map context Pins and Relays list in the <u>Options</u> tab of the ECU Multi Test step.

Note The <u>SMU/PPS - Reset Device</u> step filters the pins specified by instrument type ID. If none of the pins specified have an instrument type ID of niDCPower, the step will do nothing, as none of the specified pins are connected to NI-DCPower instruments.

RMXPS - Enable Output Step

Use the RMXPS - Enable Output step to enable power generation for the specified RMXPS pins on all sites in the Pin Map context.

Note The RMXPS - Enable Output step must be used inside of an <u>ECU Multi</u> <u>Test</u> or <u>ECU Action</u> step with its module adapter configured as Sequence. The RMXPS - Enable Output step requires the Pin Map context passed from these steps as an input.

Use the RMXPS - Enable Output step edit tab in the TestStand Sequence Editor to specify the Pin Map context and the pins to query to obtain their associated RMX-410x instrument sessions.

Related information:

• TestStand Sequence Editor

RMXPS - Enable Output Step Settings Tab

The <u>RMXPS - Enable Output Step</u> Settings tab contains the following options:

- **Pin Map Context**—Specifies a reference to a Pin Map Context object. This object describes a subset of pins, sites, and instruments from the <u>pin map file</u> and provides data pertinent to each site executing during a test.
- **Pins**—Specifies the pins to query to obtain their associated RMX-410x instrument sessions. These instrument sessions are used to enable power generation on the specified pins. The specified pins must be defined in the <u>pin map file</u> and included in the Pin Map context Pins and Relays list in the <u>Options</u> tab of the ECU Multi Test step.

Note The <u>RMXPS - Enable Output</u> step filters the pins specified by instrument type ID. If none of the pins specified have an instrument type ID of RMX410xPowerSupplyTypeId, the step will do nothing, as none of the specified pins are connected to RMX-410x instruments.

RMXPS - Disable Output Step

Use the RMXPS - Disable Output step to disable power generation for the specified RMXPS pins on all sites in the Pin Map context.

Note The RMXPS - Disable Output step must be used inside of an <u>ECU Multi</u> <u>Test</u> or <u>ECU Action</u> step with its module adapter configured as Sequence. The RMXPS - Disable Output step requires the Pin Map context passed from these steps as an input.

Use the RMXPS - Disable Output step edit tab in the TestStand Sequence Editor to specify the Pin Map context and the pins to query to obtain their associated RMX-410x instrument sessions.

Related information:

• TestStand Sequence Editor

RMXPS - Disable Output Step Settings Tab

The <u>RMXPS - Disable Output Step</u> Settings tab contains the following options:

- **Pin Map Context**—Specifies a reference to a Pin Map Context object. This object describes a subset of pins, sites, and instruments from the <u>pin map file</u> and provides data pertinent to each site executing during a test.
- **Pins**—Specifies the pins to query to obtain their associated RMX-410x instrument sessions. These instrument sessions are used to disable power generation on the specified pins. The specified pins must be defined in the <u>pin map file</u> and included in the Pin Map context Pins and Relays list in the <u>Options</u> tab of the ECU Multi Test step.

Note The <u>RMXPS</u> - <u>Disable Output</u> step filters the pins specified by instrument type ID. If none of the pins specified have an instrument type ID of RMX410xPowerSupplyTypeId, the step will do nothing, as none of the specified pins are connected to RMX-410x instruments.

RMXPS - Read Measurement Step

Use the RMXPS - Enable Output step to read, publish and return the voltage, current, and power measurement results for the specified RMXPS pins on all sites in the Pin Map context.

Note The RMXPS - Read Measurement step must be used inside of an <u>ECU</u> <u>Multi Test</u> or <u>ECU Action</u> step with its module adapter configured as Sequence. The RMXPS - Read Measurement step requires the Pin Map context passed from these steps as an input.

Use the RMXPS - Read Measurement step edit tab in the TestStand Sequence Editor to specify the Pin Map context and the pins to query to obtain their associated RMX-410x instrument sessions.

RMXPS - Read Measurement Step Settings Tab

The <u>RMXPS - Read Measurement Step</u> Settings tab contains the following options:

• **Pin Map Context**—Specifies a reference to a Pin Map Context object. This object describes a subset of pins, sites, and instruments from the <u>pin map file</u> and

provides data pertinent to each site executing during a test.

- Pins—Specifies the pins to query to obtain their associated RMX-410x instrument sessions. These instrument sessions are used to read voltage, current, and power measurements on the specified pins. The specified pins must be defined in the <u>pin</u> <u>map file</u> and included in the Pin Map context Pins and Relays list in the <u>Options</u> tab of the ECU Multi Test step.
 - Note The <u>RMXPS Read Measurement</u> step filters the pins specified by instrument type ID. If none of the pins specified have an instrument type ID of RMX410xPowerSupplyTypeId, the step will do nothing, as none of the specified pins are connected to RMX-410x instruments.
- Published Data ID Voltage— Specifies the unique ID for distinguishing measurements when you publish multiple measurements for a single pin within the same step or low-level sequence. The published data ID is represented as a constant string value. The default published data ID for RMXPS voltage measurements is Voltage.
- Published Data ID Current— Specifies the unique ID for distinguishing measurements when you publish multiple measurements for a single pin within the same step or low-level sequence. The published data ID is represented as a constant string value. The default published data ID for RMXPS current measurements is Current.
- Published Data ID Power— Specifies the unique ID for distinguishing measurements when you publish multiple measurements for a single pin within the same step or low-level sequence. The published data ID is represented as a constant string value. The default published data ID for RMXPS power measurements is Power.
- Voltage—Specifies a location to store RMXPS voltage measurement results for the specified RMXPS pins on all sites in the Pin Map context.
- **Current**—Specifies a location to store SMU/PPS current measurement results for the SMU/PPS specified pins on all sites in the Pin Map context.
- **Power**—Specifies a location to store RMXPS power measurement results for the specified RMXPS pins on all sites in the Pin Map context.

The measurement results are organized as a two-dimensional array of Number values. Each array row corresponds to a pin, with the first row representing the first valid pin in the list of pins defined by **Pins**. Each array column corresponds to a site, with column indexes corresponding with the site order in the **Pin Map Context**. Use a twodimensional array of Number values to store the measurement results for use in subsequent steps.

RMXPS - Reset Output Protection Step

Use the RMXPS - Reset Output Protection step to clear the latch that disables output when an overvoltage or undervoltage fault condition is detected on the specified RMXPS pins on all sites in the Pin Map context.



Use the RMXPS - Reset Output Protection step edit tab in the TestStand Sequence Editor to specify the Pin Map context and the pins to query to obtain their associated RMX-410x instrument sessions.

Related information:

• TestStand Sequence Editor

RMXPS - Reset Output Protection Step Settings Tab

The <u>RMXPS - Reset Output Protection Step</u> Settings tab contains the following options:

- **Pin Map Context**—Specifies a reference to a Pin Map Context object. This object describes a subset of pins, sites, and instruments from the <u>pin map file</u> and provides data pertinent to each site executing during a test.
- **Pins**—Specifies the pins to query to obtain their associated RMX-410x instrument sessions. These instrument sessions are used to reset the output protection on the specified pins. The specified pins must be defined in the <u>pin map file</u> and included in the Pin Map context Pins and Relays list in the <u>Options</u> tab of the ECU Multi Test step.



Note The <u>RMXPS</u> - <u>Reset Output Protection</u> step filters the pins specified by instrument type ID. If none of the pins specified have an instrument
type ID of RMX410xPowerSupplyTypeId, the step will do nothing, as none of the specified pins are connected to RMX-410x instruments.

NISE - Open Sessions Step

Use the NISE - Open Sessions step to create sessions for all NI Switch Executive virtual devices in the Pin Map context and store the created sessions in the Pin Map context.

Note Do not place this step into the sequence called by an <u>ECU Multi Test</u> or <u>ECU Action</u> step with its module adapter configured as Sequence.

Use the NISE - Open Sessions step edit tab in the TestStand Sequence Editor to specify the Pin Map context.

Related information:

• TestStand Sequence Editor

NISE - Open Sessions Step Settings Tab

The <u>NISE - Open Sessions Step</u> Settings tab contains the following options:

- **Pin Map Context**—Specifies a reference to a Pin Map Context object. This object describes a subset of pins, sites, and instruments from the <u>pin map file</u> and provides data pertinent to each site executing during a test.
- **Option String**—The option string is used to pass information to IVI devices on startup to set parameters such as simulation and range checking. Consult your driver documentation for more information about valid option string entries for your instrumentation. The default value for this parameter is an empty string.

NISE - Close Sessions Step

Use the NISE - Close Sessions step to close all NI Switch Executive virtual device sessions in the Pin Map context.



Note Do not place this step into the sequence called by an <u>ECU Multi Test</u> or

ECU Action step with its module adapter configured as Sequence.

Use the NISE - Close Sessions step edit tab in the TestStand Sequence Editor to specify the Pin Map context.

Related information:

• TestStand Sequence Editor

NISE - Close Sessions Step Settings Tab

The <u>NISE - Close Sessions Step</u> Settings tab contains the following options:

• **Pin Map Context**—Specifies a reference to a Pin Map Context object. This object describes a subset of pins, sites, and instruments from the <u>pin map file</u> and provides data pertinent to each site executing during a test.

NISE - Pin to Sessions Step

Use the NISE - Pin to Sessions step to return the NI Switch Executive virtual device sessions, switch routes, and new Pin Map context objects required to access a switched pin and a switched instrument channel.

Note The NISE - Pin to Sessions step must be used inside of an <u>ECU Multi</u> <u>Test</u> or <u>ECU Action</u> step with its module adapter configured as Sequence. The NISE - Pin to Sessions step requires the Pin Map context passed from these steps as an input.

Use the NISE - Pin to Sessions step edit tab in the TestStand Sequence Editor to specify the Pin Map context and the pins to query to obtain their associated NI Switch Executive instrument sessions.

NISE - Pin to Sessions Step Settings Tab

The <u>NISE - Pin to Sessions Step</u> Settings tab contains the following options:

• **Pin Map Context**—Specifies a reference to a Pin Map Context object. This object describes a subset of pins, sites, and instruments from the <u>pin map file</u> and provides data pertinent to each site executing during a test.

- **Pin**—Specifies the name of the pin to switch in order to obtain its associated NI Switch Executive virtual device session for each site in the Pin Map context.
- Instrument Type ID—Specifies the instrument type ID of the NI instrument to switch. The <u>NISE - Pin to Sessions</u> step uses the specified instrument type ID and the specified pin to generate the route/route group name for each site in the Pin Map context.
- NISE Site-Based Info Array—Returns an array of NI Switch Executive site-based information. Each element in the returned array includes the new Pin Map Context object, the virtual device session, and the route/route group name required to access the switched pin and switched instrument channel on a site in the above specified Pin Map context. You can create an array of

NI_ECUToolkit_SwitchExecutivSiteBasedInformation type to store the returned values.

NISE - Connect Step

Use the NISE - Connect step to connect the routes specified in the connection specification.

Note The NISE - Connect step must be used inside of an <u>ECU Multi Test</u> or <u>ECU Action</u> step with its module adapter configured as Sequence. The NISE - Connect step requires the Pin Map context passed from these steps as an input.

Use the NISE - Connect step edit tab in the TestStand Sequence Editor to specify the Pin Map context and the pins to query to obtain their associated NI Switch Executive instrument sessions.

Related information:

• TestStand Sequence Editor

NISE - Connect Step Settings Tab

The <u>NISE - Connect Step</u> Settings tab contains the following options:

• Virtual Device Session—Specifies the NI Switch Executive virtual device session returned by the <u>NISE - Pin to Sessions</u> step.

- **Connection Specification**—Specifies the routes or route groups you are connecting. The expression must be a valid route specification string as defined by the NI Switch Executive configuration for the virtual device being used.
- **Multiconnect Mode**—Specified the connection mode for this step as an integer when more than one connection operation occurs on a specific route. The mode may be one of the following:
 - **Default (-1)**: Routes default to the multiconnect mode as predefined for each route in NI Switch Executive.
 - No Multiconnect (0): Routes can connect once. Any attempt to reconnect an already connected route results in an error.
 - Multiconnect Routes (1): Routes can connect multiple times. Routes must contain the same endpoints and path, and are automatically reference counted by NI Switch Executive. If you issue multiple <u>Connect</u> operations for a specific route, the route is not physically disconnected until an equal number of <u>Disconnect</u> operations occur. You can either issue the Disconnect operation manually or use a lifetime setting for the route. The <u>Disconnect All</u> operation disconnects a route even if the route reference count is greater than one.
- Wait for Debounce—If True, the <u>NISE Connect</u> step waits for all switches to debounce before it returns. If False, it immediately returns after the switch control is completed. The Wait for Debounce action occurs after all <u>Connect</u> and <u>Disconnect</u> operations are complete. The default Wait for Debounce value in this step is True.

NISE - Disconnect Step

Use the NISE - Disconnect step to disconnect the routes specified in the disconnection specification.

Note The NISE - Disconnect step must be used inside of an <u>ECU Multi Test</u> or <u>ECU Action</u> step with its module adapter configured as Sequence. The NISE - Disconnect step requires the Pin Map context passed from these steps as an input.

Use the NISE - Disconnect step edit tab in the TestStand Sequence Editor to specify the Pin Map context and the pins to query to obtain their associated NI Switch Executive instrument sessions.

Related information:

• TestStand Sequence Editor

NISE - Disconnect Step Settings Tab

The <u>NISE - Disconnect Step</u> Settings tab contains the following options:

- Virtual Device Session—Specifies the NI Switch Executive virtual device session returned by the <u>NISE Pin to Sessions</u> step.
- **Disconnection Specification**—Specifies the routes or route groups you are disconnecting. The expression must be a valid route specification string as defined by the NI Switch Executive configuration for the virtual device being used.
- Wait for Debounce—If True, the <u>NISE Disconnect Step</u> step waits for all switches to debounce before it returns. If False, it immediately returns after the switch control is completed. The Wait for Debounce action occurs after all <u>Connect</u> and <u>Disconnect</u> operations are complete. The default Wait for Debounce value in this step is False.
- Maximum Time (ms)—Specifies the amount of time in milliseconds to wait for debounce to complete. A value of 0 checks for debouncing once and returns an error if the system is not debounced at that time. A value of -1 means to wait for an infinite period of time until the system is debounced. The default Maximum Time in this step is set to -1.

NISE - Disconnect All Step

Use the NISE - Disconnect All step to disconnect all connections on every IVI switch device managed by NI Switch Executive virtual device sessions in the Pin Map context.

Note The NISE - Disconnect All step must be used inside of an <u>ECU Multi Test</u> or <u>ECU Action</u> step with its module adapter configured as Sequence. The NISE - Disconnect All step requires the Pin Map context passed from these steps as an input.

Use the NISE - Disconnect All step edit tab in the TestStand Sequence Editor to specify the Pin Map context and the pins to query to obtain their associated NI Switch Executive instrument sessions.

Related information:

• TestStand Sequence Editor

NISE - Disconnect All Step Settings Tab

The <u>NISE - Disconnect Step</u> Settings tab contains the following options:

- Virtual Device Session—Specifies the NI Switch Executive virtual device session returned by the <u>NISE Pin to Sessions</u> step.
- Wait for Debounce—If True, the <u>NISE Disconnect Step</u> step waits for all switches to debounce before it returns. If False, it immediately returns after the switch control is completed. The Wait for Debounce action occurs after all <u>Connect</u> and <u>Disconnect</u> operations are complete. The default Wait for Debounce value in this step is False.
- Maximum Time (ms)—Specifies the amount of time in milliseconds to wait for debounce to complete. A value of 0 checks for debouncing once and returns an error if the system is not debounced at that time. A value of -1 means to wait for an infinite period of time until the system is debounced. The default Maximum Time in this step is set to -1.

Dialog Boxes and Windows

ECU Software Toolkit dialog boxes and windows are used to interact with the software and make changes to test programs.

Refer to the following for information about significant ECU Software Toolkit dialog boxes and windows.

- Configure Lot Settings Dialog Box
- Log Browser Window
- Debug Test Results Log Options Dialog Box

Configure Lot Settings Dialog Box

Use the **Configure Lot Settings** dialog box to specify settings for the current lot under test.

Select **ECU Toolkit** » **Configure Lot** in the TestStand Sequence Editor, click the **Configure Lot** button on the ECU Software Toolkit toolbar, or click the **Configure Lot** button in the default ECU Software Toolkit operator interface to launch the Configure Lot Settings dialog box.

Note If an active test configuration you select for a test program contains any of the following fields, the ECU Software Toolkit dims the field in this dialog box because the test program obtains the value from the test configuration at run time.

The Configure Lot Settings dialog box contains the following options:

- Test Program Path—Use this option to browse to a file path. If you specify a value for the Test Program Directory option, this texbox lists the sequence files in the directory you specified.
- **Test Program Configuration**—Specifies the test program configuration to execute. This option is available only when you define configurations for a test program.
- **Part Type**—**Populates the** LotSettings.Standard.PartType **property**.
- Lot ID—Populates the LotSettings.Standard.LotId property.
- Estimated Lot Size—Populates the LotSettings.Standard.LotSize property, which an inline QA algorithm can use to determine for which DUTs to enable inline QA.
- Test Flow—Populates the LotSettings.Standard.TestFlowId property.
- Test Temperature—Populates the LotSettings.Standard.TestTemperature property.
- Operator Name—Displays the value of the LotSettings.Standard.OperatorName property, if it exists. If this property is blank, this field displays the user name of the currently logged in user.
- Enabled Sites—Specifies which site numbers to use when running a test program. The ECU Software Toolkit obtains the sites to display in this list from the pin map for the test program or from the number of test sockets specified in the Model Options dialog box if the test program does not use a pin map. You must enable at least one site. If the test program uses the Sequential process model, you can enable only one site.

Related information:

- Model Options Dialog Box
- Sequential Process Model

Log Browser Window

Use the **Log Browser** window to view or compare multiple log files that represent different modifications of a test program

Click the **Log Browser** button near the upper right corner of the Test Program Performance Analyzer to open the Log Browser window.

Complete the following steps to view or compare log files.

- 1. In the **Log Directory** path box, select the top-level directory that contains the log files you want to view to compare. The column on the left displays relative subdirectories.
- 2. Complete the following steps to view file(s) in Single Data Set mode or Compare Data Sets mode.
 - a. **Single Data Set** Double-click the row of the file you want to load. The background row color turns blue.
 - b. **Compare Data Sets** Right-click the row of the base log file you want to load and chose **Select Base Log File** from the context menu. The background row color of the base log file you selected turns gray. Right-click the row of the modified log file you want to load and chose **Select Modified Log File** from the context menu. The background row color of the modified log file you selected turns blue.

Debug Test Results Log Options Dialog Box

Use the Debug Test Results Log Options dialog box to specify settings for the <u>Debug</u> <u>Test Results Log</u>.

The Debug Test Results Log Options dialog box contains the following options:

• **Debug Test Results Log Destination Directory**—Absolute path of the directory in which you want the ECU Software Toolkit to create the data log file. Leave the control blank if you want the ECU Software Toolkit to create the data log file in the same directory as the test program main sequence file.

- **Report Orientation**—Specifies the orientation of the Debug Test Results Log. The default is portrait orientation. Landscape orientation uses wider columns for tests with long test numbers or test names. Landscape orientation does not maintain report column alignment when test names exceed 101 characters.
- Log Results Only for DUT Failures—Includes results in the Debug Test Results Log only if the DUT fails testing. If a DUT passes testing, the ECU Software Toolkit does not update the Debug Test Results Log.
- Limit Number of Results Displayed in Report View—Limits the number of results to display in the report views in the ECU Software Toolkit Operator Interface and TestStand Sequence Editor to the number of DUTs you specify.