Agilent TS-5000 Functional Test System

System Software User's Guide

Agilent Technologies

Notices

Documentation History

All Editions and Updates of this manual and their creation date are listed below. The first Edition of the manual is Edition 1. The Edition number increments by 1 whenever the manual is revised. Updates, which are issued between Editions, contain replacement pages to correct or add additional information to the current Edition of the manual. Whenever a new Edition is created, it will contain all of the Update information for the previous Edition. Each new Edition or Update also includes a revised copy of this documentation history page.

Edition 1 (E8770-90000) July 1999

Edition 2 (E8770-90023) September 2000

Edition 3 (E8770-90033) January 2004

Edition 4 (E8770-90033) January 2006

Edition 4 (E8770-90033) April 2010

Edition 5 (E8770-90033) May 2011

Edition 6 (E8770-90033) August 2011

Edition 7 (E8770-90033) December 2012

Manual Part Number

E8770-90033

Printed in Malaysia

Agilent Technologies Microwave Products
(Malaysia) Sdn. Bhd.
Bayan Lepas Free Industrial Zone
11900 Penang, Malaysia

Warranty

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

Technology Licenses

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

Restricted Rights Legend

If software is for use in the performance of a U.S. Government prime contract or subcontract, Software is delivered and licensed as "Commercial computer software" as defined in DFAR 252.227-7014 (June 1995), or as a "commercial item" as defined in FAR 2.101(a) or as "Restricted computer software" as defined in FAR 52.227-19 (June 1987) or any equivalent agency regulation or contract clause. Use, duplication or disclosure of Software is subject to Agilent Technologies' standard commercial license terms, and non-DOD Departments and Agencies of the U.S. Government will receive no greater than Restricted Rights as defined in FAR 52.227-19(c)(1-2) (June 1987). U.S. Government users will receive no greater than Limited Rights as defined in FAR 52.227-14 (June 1987) or DFAR 252.227-7015 (b)(2) (November 1995), as applicable in any technical data.

Safety Notices

Caution

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

WARNING

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

Safety Summary

The following general safety precautions must be observed during all phases of operation of this system. Failure to comply with these precautions or with specific warnings elsewhere in this manual violates safety standards of design, manufacture, and intended use of the system. Agilent Technologies, Inc. assumes no liability for the customer's failure to comply with these requirements.

General

This product is provided with a protective earth terminal. The protective features of this product may be impaired if it is used in a manner not specified in the operation instructions.

WARNING

DO NOT OPERATE IN AN EXPLOSIVE ATMOSPHERE
Do not operate the system in the presence of flammable gases or flames.

If the equipment in this system is used in a manner not specified by Agilent Technologies, the protection provided by the equipment may be impaired.

Cleaning Instructions

Clean the system cabinet using a soft cloth dampened in water.

WARNING

DO NOT REMOVE ANY SYSTEM COVER
Operating personnel must not remove system covers. Component replacement and internal adjustments must be made only by qualified service personnel. Equipment that appears damaged or defective should be made inoperative and secured against unintended operation until they can be repaired by qualified service personnel.

Environmental Conditions

Unless otherwise noted in the specifications, this system is intended for indoor use in an installation category II, pollution degree 2 environment. It is designed to operate at a maximum relative humidity of 80% and at altitudes of up to 2000 meters. Refer to the specifications tables for the ac mains voltage requirements and ambient operating temperature range.

Before applying power

Verify that all safety precautions are taken. Note the external markings described in "Safety Symbols and Regulatory Markings"on page 4.

Ground the System

To minimize shock hazard, the instrument chassis and cover must be connected to an electrical protective earth ground. The instrument must be connected to the ac power mains through a grounded power cable, with the ground wire firmly connected to an electrical ground (safety ground) at the power outlet. Any interruption of the protective (grounding) conductor or disconnection of the protective earth terminal will cause a potential shock hazard that could result in personal injury.

⚠ **WARNING**

The power cable ground wire must be connected to an electrical ground (safety ground) at the power outlet. Any interruption of the protective grounding will cause a potential shock hazard that could result in personal injury.

Fuses

Use only fuses with the required rated current, voltage, and specified type (normal blow, time delay). Do not use repaired fuses or short-circuited fuse holders. To do so could cause a shock or fire hazard.

Operator Safety Information

MODULE CONNECTORS AND TEST SIGNAL CABLES CONNECTED TO THEM CANNOT BE OPERATOR ACCESSIBLE: Cables and connectors are considered inaccessible if a tool (e.g., screwdriver, wrench, socket, etc.) or a key (equipment in a locked cabinet) is required to gain access to them. Additionally, the operator cannot have access to a conductive surface connected to any cable conductor (High, Low or Guard).

ASSURE THE EQUIPMENT UNDER TEST HAS ADEQUATE INSULATION BETWEEN THE CABLE CONNECTIONS AND ANY OPERATOR-ACCESSIBLE PARTS (DOORS, COVERS, PANELS, SHIELDS, CASES, CABINETS, ETC.): Verify there are multiple and sufficient protective means (rated for the voltages you are applying) to assure the operator will NOT come into contact with any energized conductor even if one of the protective means fails to work as intended. For example, the inner side of a case, cabinet, door, cover or panel can be covered with an insulating material as well as routing the test cables to the module's front panel connectors through non-conductive, flexible conduit such as that used in electrical power distribution.

Safety Symbols and Regulatory Markings

Symbols and markings on the system, in manuals and on instruments alert you to potential risks, provide information about conditions, and comply with international regulations. Table 1 and Table 2 define the symbols and markings you may encounter.

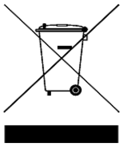
Table 1 Safety Symbols and Markings

⚡	Warning: risk of electric shock.
⚠	Caution: refer to accompanying documents.
∼	Alternating current.
≂	Both direct and alternating current.
⏚	Earth (ground) terminal
⊜	Protective earth (ground) terminal
⏚	Frame or chassis terminal
⊥	Terminal is at earth potential. Used for measurement and control circuits designed to be operated with one terminal at earth potential.
○ |	Switch setting indicator. O = Off, | = On.
⏻	Standby (supply); units with this symbol are not completely disconnected from ac mains when this switch is off. To completely disconnect the unit from ac mains, either disconnect the power cord, or have a qualified electrician install an external switch.

Table 2 Regulatory Symbols Markings

CE	The CE mark is a registered trademark of the European Community.
SP	The CSA mark is a registered trademark of the Canadian Standards Association.
N10149	The C-tick mark is a registered trademark of the Spectrum Management Agency of Australia. This signifies compliance with the Australian EMC Framework regulations under the terms of the Radio Communications Act of 1992.
ISM 1-A	This text indicates that the product is an Industrial Scientific and Medical Group 1 Class A product (CISPR 11, Clause 4).

Table 2 Regulatory Symbols Markings

	This product complies with the WEEE Directive (2002/96/EC) marking requirement. The affixed product label (see above) indicates that you must not discard this electrical/electronic product in domestic household waste. Product Category: With reference to the equipment types in the WEEE directive Annex 1, this product is classified as a “Monitoring and Control instrumentation” product. **Do not dispose in domestic household waste** To return unwanted products, contact your local Agilent office, or see **http://www.agilent.com/environment/product** for more information.

Service and Support

Any adjustment, maintenance, or repair of this product must be performed by qualified personnel. Contact your customer engineer through your local Agilent Technologies Service Center.

Agilent on the Web

You can find information about technical and professional services, product support, and equipment repair and service on the Web:

http://www.agilent.com

Click the link to **Test & Measurement**. Select your country from the drop-down menus. The Web page that appears next has contact information specific for your country.

Agilent by Phone

If you do not have access to the Internet, call one of the numbers in Table 3.

Table 3 Agilent Call Centers and Regional Headquarters

United States:	Test and Measurement Call Center (800) 829 4444(toll-free in US)
Canada:	(877) 894 4414
Europe:	(41 22) 780 8111
Japan:	Measurement Assistance Center 0120 (421) 345
Latin America:	305 269 7500
Asia-Pacific:	(85 22) 599 7777

Contents

Chapter 1

Chapter 2

Chapter 3

Chapter 4

Chapter 1
System Software Overview

This chapter contains a detailed overview of the system software. Chapter contents are:

Agilent TestExec SL

The Agilent TS-5000 System uses Agilent TestExec SL and system specific software to test the Unit Under Test (UUT). Agilent TestExec SL is a test executive designed for high-volume, high-throughput functional test applications. The other system specific software provides the communications between the test executive and system instruments.

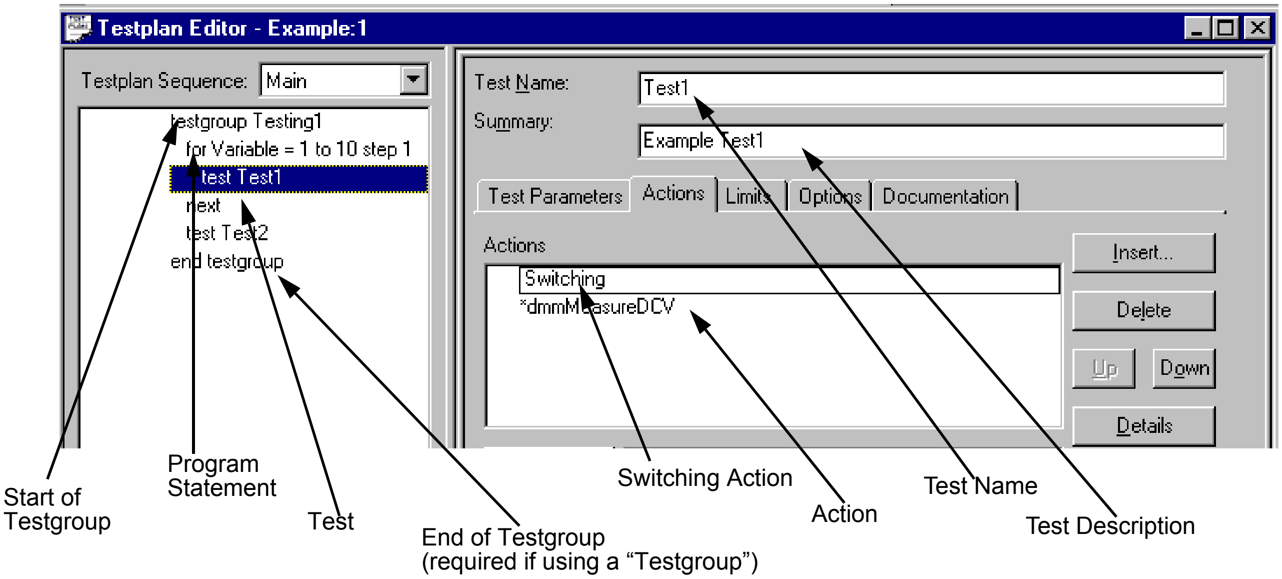
Agilent TestExec SL uses testplans (see “Testplans”) and actions (see “Actions”) to perform the tests. The actions are the building blocks from which the tests are created. The actions are called from a test which are executed in a testplan. The switching actions makes the connections from the system instruments, and/or loads on the load cards to the Unit Under Test (UUT). The switching actions are built into Agilent TestExec SL.

Both actions and switching actions are used in a testplan to run the tests. The testplan automatically closes the appropriate pin matrix and load card switches to make the connections, setup and execute the appropriate sources and detectors, and return any test results.

The switching information and the instrument types used in the system are located in the system.ust file (see “System.ust File” on page 11 for more information). This file is generated at the factory and is custom for each system. It has the necessary information for the switching actions to close the appropriate switches and for the other actions to communicate with the system instruments. Instrument data in the system.ust file can be generated and/or modified using the System Configuration Editor (see “System Configuration Editor” on page 14).

Testplans

Figure 1-1 shows a testplan and some of its components; an explanation follows the figure (see Loading a Testplan for an explanation on how to load a testplan).



Testplan Editor - Example:1

Testplan Sequence: Main

```
testgroup Testing1
 for Variable = 1 to 10 step 1
  test Test1
 next
 test Test2
end testgroup
```

Test Name: Test1

Summary: Example Test1

Actions

Switching

*dmmMeasureDCV

Figure 1-1. Typical Testplan Components

Testgroup

A named block of tests that can be executed in a predefined order. Each testgroup in a testplan must have a unique name; no duplicate names are allowed.

List of Statements

Test or flow control statements executed in the order shown.

Test

A named series of actions that can be executed as a group. A test can contain actions and switching actions. A test can have limit checking capabilities to determine if a test passes or fails. A test must have a unique name; no duplicate names are allowed.

Switching Action

Actions that make connections from the instrument and loads to the Unit Under Test. These actions are internal to Agilent TestExec SL. The switching information in these actions are determined by the data in the 'system.ust' file and the user-generated fixture.ust and uut.ust files.

Action

The smallest component in a test or testgroup that can be called to perform functions such as setting up an instrument, making measurements and prompting the user.

Actions

The system comes with a set of supplied actions. These actions are used for such things as configuring instruments, making measurements and prompting users for inputs. You can also generate custom actions using the Action Wizard and an application program such as Visual C/C++.

The action directory path is:

```
C:\Program Files\Agilent\TS-5000 System
Software\actions
```

Actions are located in the following sub-directories:

Sub-Directory	Action Type
arb	Arbitrary Waveform Generator Actions
counter	Counter Actions
dac	D/A Converter Actions
dgn	Diagnostics Actions
digitizer	Digitizer Actions
dio	Digital I/O Actions
dmm	Digital Multimeter Actions
dso	Digital Storage Oscilloscope
event	Event Detector Actions
generic	Miscellaneous Actions
mcm	Measurement Control Module Actions
power	Power Supply Actions
serial	Serial Interface Actions
SerialProtocol	Automotive Serial Protocol Actions
switch	SLU and Switching Actions
vi	Voltage / Current Source Actions
daq	Data Acquisition Actions
esa	Spectrum Analyzer Actions
esg	Signal Generator Actions

System.ust File

The 'system.ust' file is created at the factory with the appropriate module/instrument and wiring data according to the system option. The file can be edited using the System Configuration Editor (see “System

Configuration Editor” on page 14) or the Topology Editor in Agilent TestExec SL.

The data in the 'system.ust' file consists of modules/instruments and instrument nodes (i.e., connections at the 32-Pin Matrix and Instrument Multiplexer Module), and other nodes (connections at the 32-Pin Matrix Modules and load cards) using the Aliases, Wires, and Modules designations. The Module designations are used to determine the modules/instruments installed in the system, and the Aliases and Wires are used to generate switch paths.

A typical 'system.ust' file contains Aliases, Wires and Modules.

Aliases

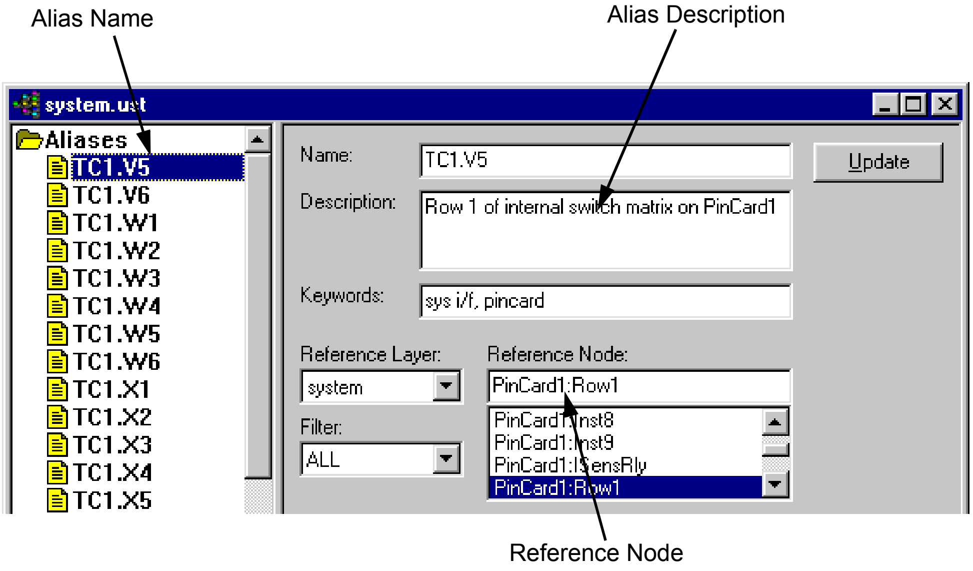
These are alternate names for reference nodes. The names are descriptive in nature to easily identify the node. For example, the node name for the high current output of the Agilent 34401 or E1411 Digital Multimeter is called "IsrcHi". Figure 1-2 shows typical Aliases in the 'system.ust' file.

Wires

These are names for wires that connect to or between nodes. In some cases, these names are the same aliases used for nodes. For example, the alias called "DVMIsrcHi" is often used for the wire name that connects to that node. Figure 1-3 shows typical wires in the 'system.ust' file.

Modules

These are the names of the instruments in the system. For example, the name "MCM" indicates that there is an Agilent E6171B Measurement control module in the system. The Figure 1-4 shows a few typical modules in the 'system.ust' file.



Aliases
- TC1.V5
- TC1.V6
- TC1.W1
- TC1.W2
- TC1.W3
- TC1.W4
- TC1.W5
- TC1.W6
- TC1.X1
- TC1.X2
- TC1.X3
- TC1.X4
- TC1.X5

Name: TC1.V5

Description: Row 1 of internal switch matrix on PinCard1

Keywords: sys i/f, pincard

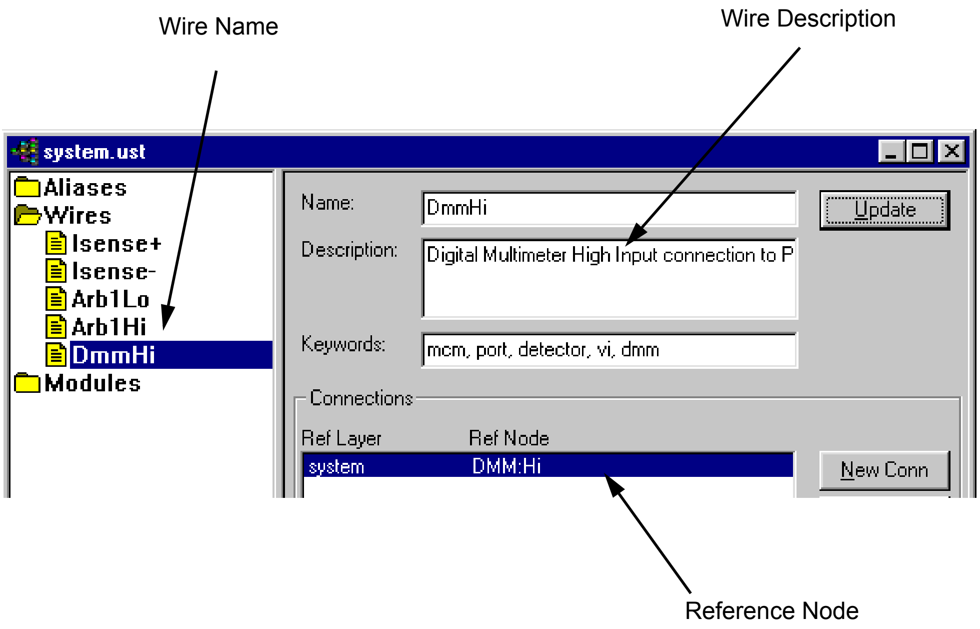
Reference Layer: system

Reference Node: PinCard1:Row1

PinCard1:Inst8
PinCard1:Inst9
PinCard1:ISensRly
PinCard1:Row1

Filter: ALL

Figure 1-2. Typical Aliases



system.ust

- Aliases
- Wires
  - Isense+
  - Isense-
  - Arb1Lo
  - Arb1Hi
  - DmmHi
- Modules

Name: DmmHi

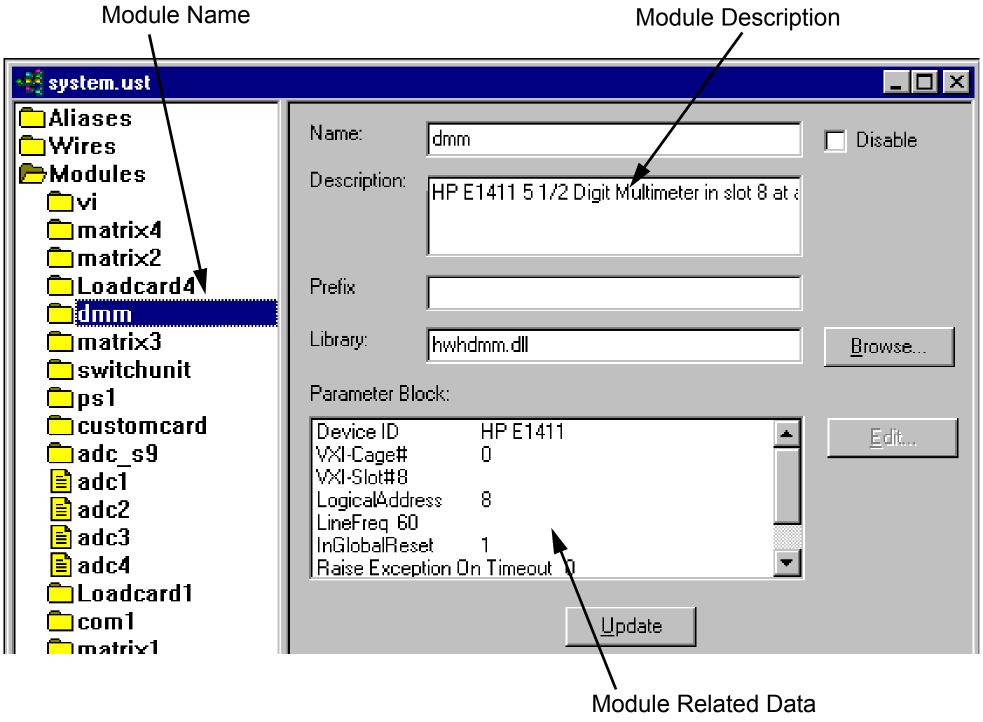
Description: Digital Multimeter High Input connection to P

Keywords: mcm, port, detector, vi, dmm

Connections

Ref Layer	Ref Node
system	DMM:Hi

Figure 1-3. Typical Wires



system.ust

- Aliases
- Wires
- Modules
  - vi
  - matrix4
  - matrix2
  - Loadcard4
  - dmm
  - matrix3
  - switchunit
  - ps1
  - customcard
  - adc_s9
  - adc1
  - adc2
  - adc3
  - adc4
  - Loadcard1
  - com1
  - matrix1

Name: dmm

□ Disable

Description: HP E1411 5 1/2 Digit Multimeter in slot 8 at a

Prefix

Library: hwhdmm.dll

Parameter Block:

```
Device ID       HP E1411
VXI-Cage#       0
VXI-Slot#8
LogicalAddress  8
LineFreq  60
InGlobalReset   1
Raise Exception On Timeout  0
```

Figure 1-4. Typical Modules/Instruments

System Configuration Editor

This editor can be used to add new modules/instruments or delete old modules/instruments to/from the system.ust file (see “System.ust File” on page 11 for information). It can add/delete any supported GPIB or VXI instruments, Pin Matrix Modules, and/or load cards.

The editor has the following functions:

- Shows all supported modules/instruments.
- Automatically detects all modules/instruments currently in the system
- Able to add new custom modules/instruments to the system.ust file
- Able to edit module/instrument parameters

Any of the supported and custom modules/instruments can be added to the system.ust file. Dependent on the system type and option (e.g., Test System Interface vs. Mass Interconnect), the System Configuration Editor automatically generates the appropriate wires and aliases (see “System.ust File” on page 11 for information). The wiring, etc. data is located in a spreadsheet that is specifically generated for your system type and option.

The System Configuration Editor is available from the TestExec SL Toolbar or from a shortcut in the Desktop on the system PC controller. Refer to the System Configuration Editor’s online help for more details.

Chapter 2
How to Use the System Software

Chapter Contents

This chapter lists the needed software to run the system and shows some system specific software operation. The chapter is separated as follows:

Required Computer Hardware and Software

The following is a list of the computer hardware and software needed to run the Agilent TS-5000 Test System. All necessary software was factory installed on your system:

- IBM-compatible PC (at least a Pentium) with 256 MB of RAM, 1024x768 graphics, 100 MB of free disk space (20 MB for Agilent TestExec SL software).
- Microsoft® Windows® XP or 7 Microsoft® Windows® 64-Bit
- Agilent TestExec SL software, Version 5.1 or later.
- Agilent TS-5000 Software version 5.1 or later.

Note For more detailed information about using the Agilent TestExec SL software, refer to the software documentation

System Software Description

The Agilent TS-5000 System uses Agilent TestExec SL and system specific software to test the Unit Under Test. Agilent TestExec SL is a test executive designed for high-volume, high-throughput functional test applications. The other system specific software provides the communications between the test executive and system instruments.

Agilent TestExec SL uses testplans and actions to perform the tests. The actions are the building blocks from which the tests are created. The actions are called from a test which are executed in a testplan. The switching actions makes the connections from the system instruments, and/or loads on the load cards to the Unit Under Test. The switching actions are built into Agilent TestExec SL.

Both actions and switching actions are used in a testplan to run the tests. The testplan automatically closes the appropriate pin matrix and load card switches to make the connections, setup and execute the appropriate sources and detectors, and return any test results.

The switching information and the instrument types used in the system are located in the `system.ust` file. This file is generated at the factory and is custom for each system. It has the necessary information for the switching actions to close the appropriate switches and for the other actions to communicate with the system instruments. Instrument data in the `system.ust` file can be generated and/or modified using the System Configuration Editor (see “Using the System Configuration Editor” in the “TS-5000 System Integrator’s Manual”).

Selecting Agilent TestExec SL

The Agilent TestExec SL software is pre-installed on your PC controller's hard drive. Start TestExec SL from this icon in the PC desktop:

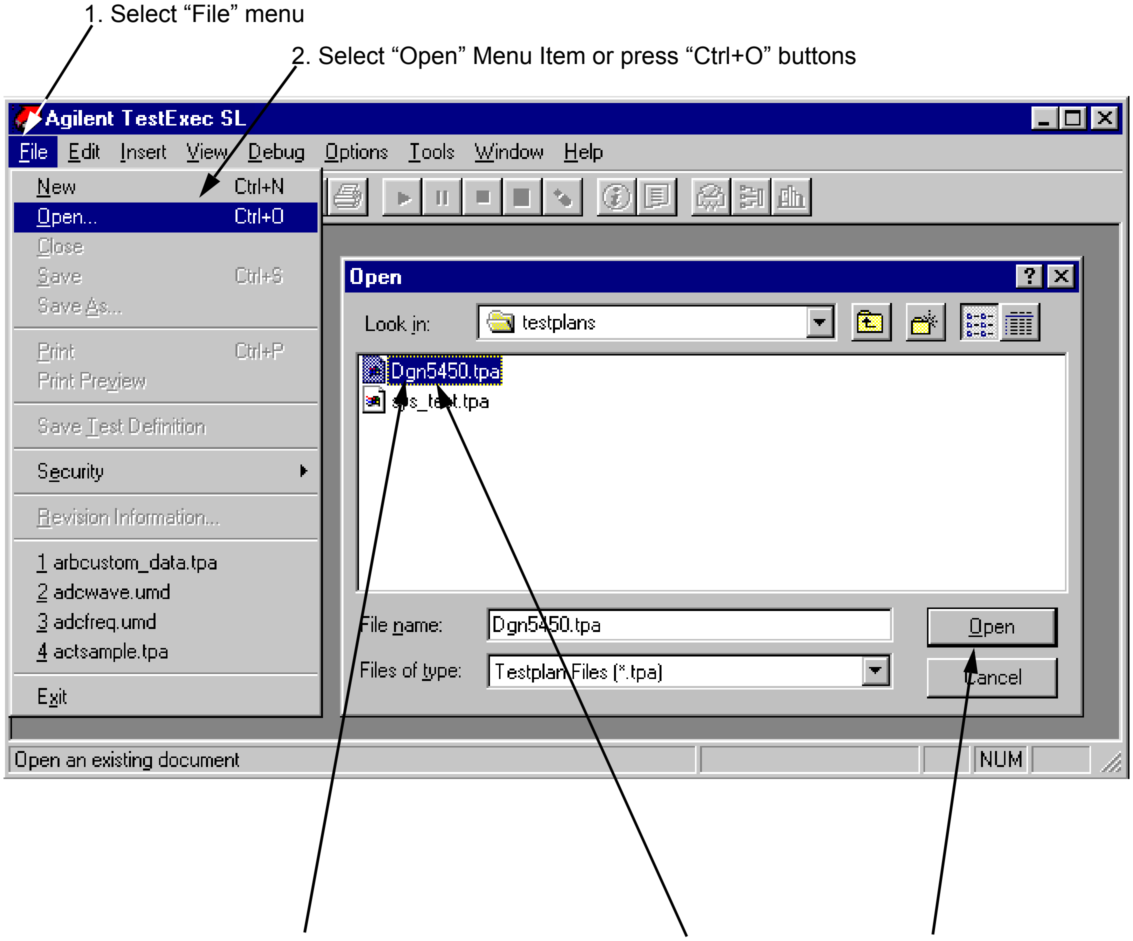
You can also run TestExec SL by clicking:

Start* | *Programs* | *Agilent TestExec SL 7.0* | *TestExec SL 7.0

Loading a Testplan

1. Select "File" menu

2. Select "Open" Menu Item or press "Ctrl+O" buttons



File name: Dgn5450.tpa

Files of type: Testplan Files (*.tpa)

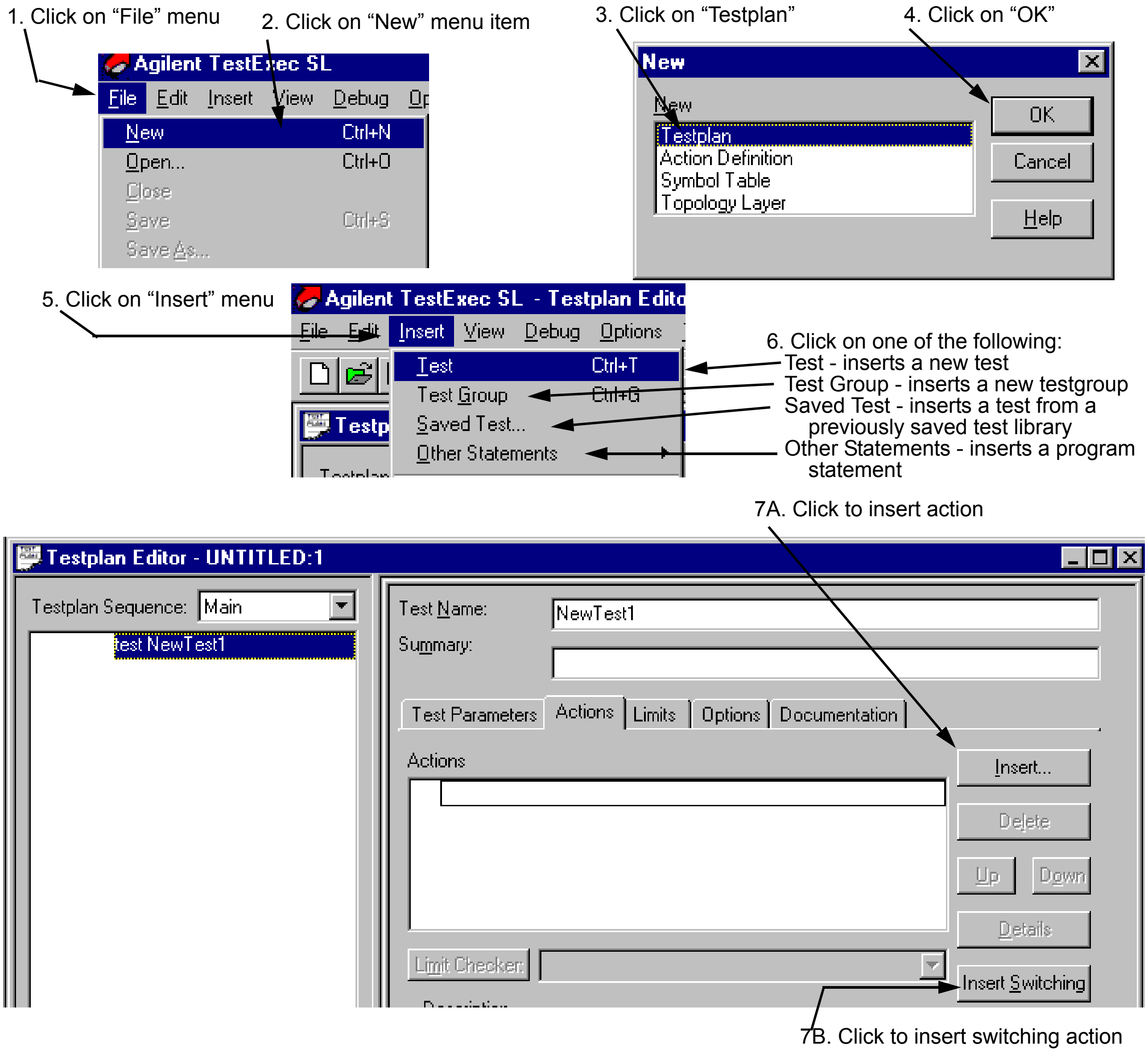
Open an existing document

3. Double Click on File Name or Select File Name and click on "Open"

Figure 2-1. Agilent TestExec SL Main Screen and File Open Box

Creating a Testplan

Figure 2-2 shows how to create a testplan.



1. Click on “File” menu

2. Click on “New” menu item

3. Click on “Testplan”

4. Click on “OK”

5. Click on “Insert” menu

6. Click on one of the following:
Test - inserts a new test
Test Group - inserts a new testgroup
Saved Test - inserts a test from a previously saved test library
Other Statements - inserts a program statement

7A. Click to insert action

7B. Click to insert switching action

Figure 2-2. Creating a Testplan

Using TS-5000 Supplied Actions

The Agilent TS-5000 System with a set of standard actions supplied with the system. The action directory path is:

```
C:\Program Files\Agilent\TS-5000 System Software\actions
```

Actions are located in the following sub-directories:

Sub-Directory	Action Type
arb	Arbitrary Waveform Generator Actions
counter	Counter Actions
dac	D/A Converter Actions
dgn	Diagnostics Actions
digitizer	Digitizer Actions
dio	Digital I/O Actions
dmm	Digital Multimeter Actions
dso	Digital Storage Oscilloscope Actions
event	Event Detector Actions
generic	Miscellaneous Actions
mcm	Measurement Control Module Actions
power	Power Supply Actions
serial	Serial Interface Actions
SerialProtocol	Automotive Serial Protocol Actions
switch	SLU and Switching Actions
vi	Voltage / Current Source Actions
daq	Data Acquisition Actions
esa	Spectrum Analyzer Actions
esg	Signal Generator Actions

Note The actions are summarized in Chapter 4 of this manual. The actions are documented in detail in the TS-5000 online help which is available from the TestExec SL Help menu.

Standard Action Types

The standard actions are generally organized around a specific instrument or module in the system. The actions are also separated into the types discussed below.

High-Level Actions

These are actions that usually perform complete tests, such as setting up a source and then making a measurement. These actions normally, but not always, use one or more of the Low-Level Actions.

Low-Level Actions

These actions perform specific configuration or measurement function on a particular instrument. The low-level actions give you more flexibility in configuring instruments and making the measurements than do the high-level actions. Low-Level actions are organized by instrument type and function. For example, an action that is to set up triggering for a voltmeter is a low-level action.

Example Testplans

Example testplans to show how to use the actions are in the following directory:

C:\Program Files\Agilent\TS-5000 System Software\testplan\examples

Adding an Action to a Testplan

Do the following:

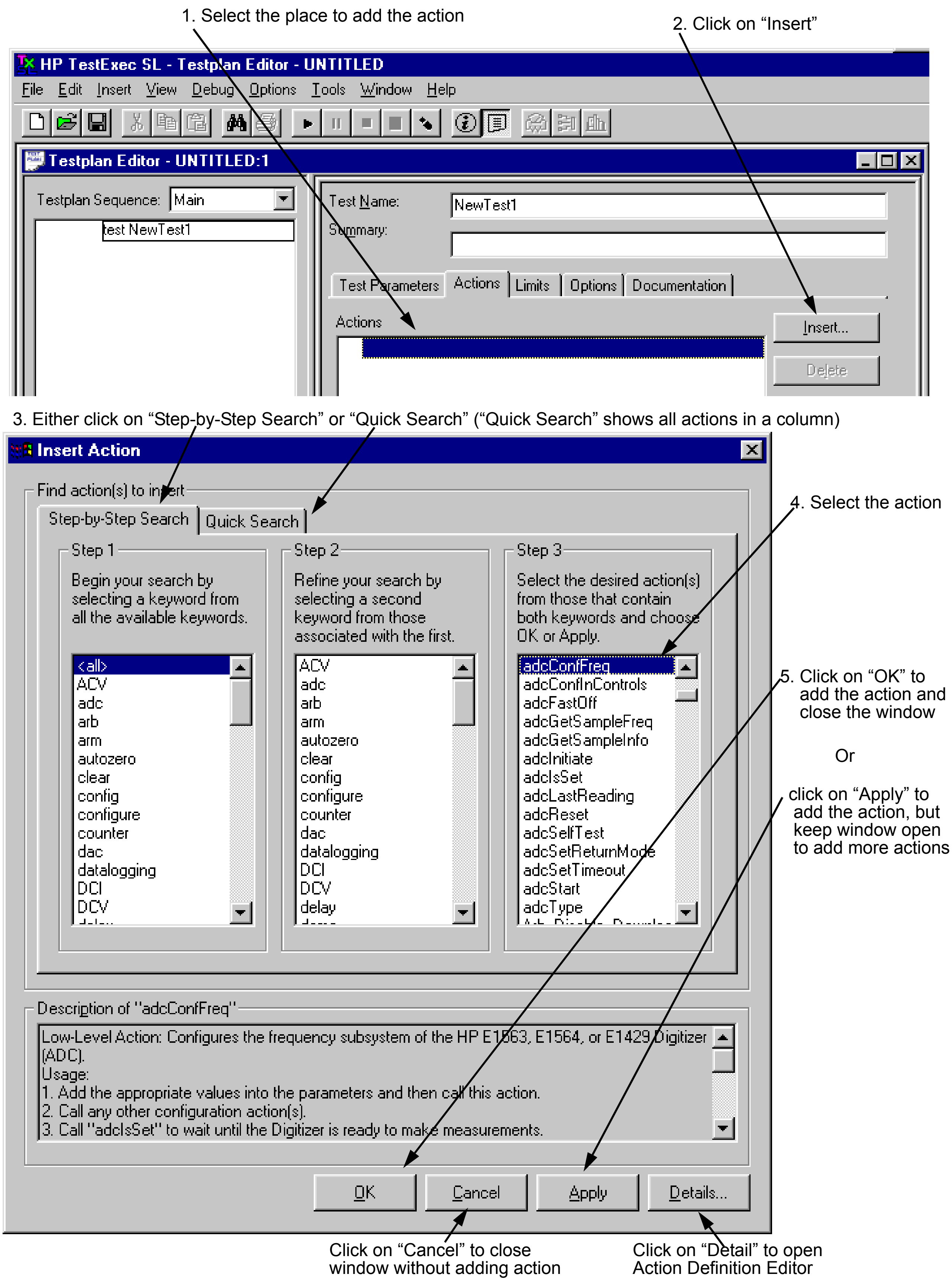
1. Open Agilent TestExec SL using the procedure in “Selecting Agilent TestExec SL” on page 17.

2. Either create a new or open an old testplan, using the procedure in “Loading a Testplan” on page 17 or “Creating a Testplan” on page 18, respectively.

3. If using a new testplan, do the procedure in Figure 2-2 on page 18 to add a test. If using an old testplan, select the test into which you wish to add an action and continue with step 4.

4. Add an action to the test, as shown in Figure 2-3 and Figure 2-4.

   There are two different actions that can be added, a regular action and a switching action. Figure 2-3 shows how to add a regular action and Figure 2-4 shows how to add a switching action.



1. Select the place to add the action

2. Click on “Insert”

HP TestExec SL - Testplan Editor - UNTITLED

Testplan Editor - UNTITLED:1

Testplan Sequence: Main

test NewTest1

Test Name: NewTest1

Summary:

3. Either click on “Step-by-Step Search” or “Quick Search” (“Quick Search” shows all actions in a column)

Insert Action

Find action(s) to insert

Step 1

Begin your search by selecting a keyword from all the available keywords.

<all>, ACV, adc, arb, arm, autozero, clear, config, configure, counter, dac, datalogging, DCI, DCV

Step 2

Refine your search by selecting a second keyword from those associated with the first.

ACV, adc, arb, arm, autozero, clear, config, configure, counter, dac, datalogging, DCI, DCV, delay

Step 3

Select the desired action(s) from those that contain both keywords and choose OK or Apply.

adcConfFreq, adcConfInControls, adcFastOff, adcGetSampleFreq, adcGetSampleInfo, adcInitiate, adcIsSet, adcLastReading, adcReset, adcSelfTest, adcSetReturnMode, adcSetTimeout, adcStart, adcType

4. Select the action

Description of "adcConfFreq"

Low-Level Action: Configures the frequency subsystem of the HP E1563, E1564, or E1429 Digitizer (ADC).
Usage:
1. Add the appropriate values into the parameters and then call this action.
2. Call any other configuration action(s).
3. Call "adcIsSet" to wait until the Digitizer is ready to make measurements.

5. Click on “OK” to add the action and close the window

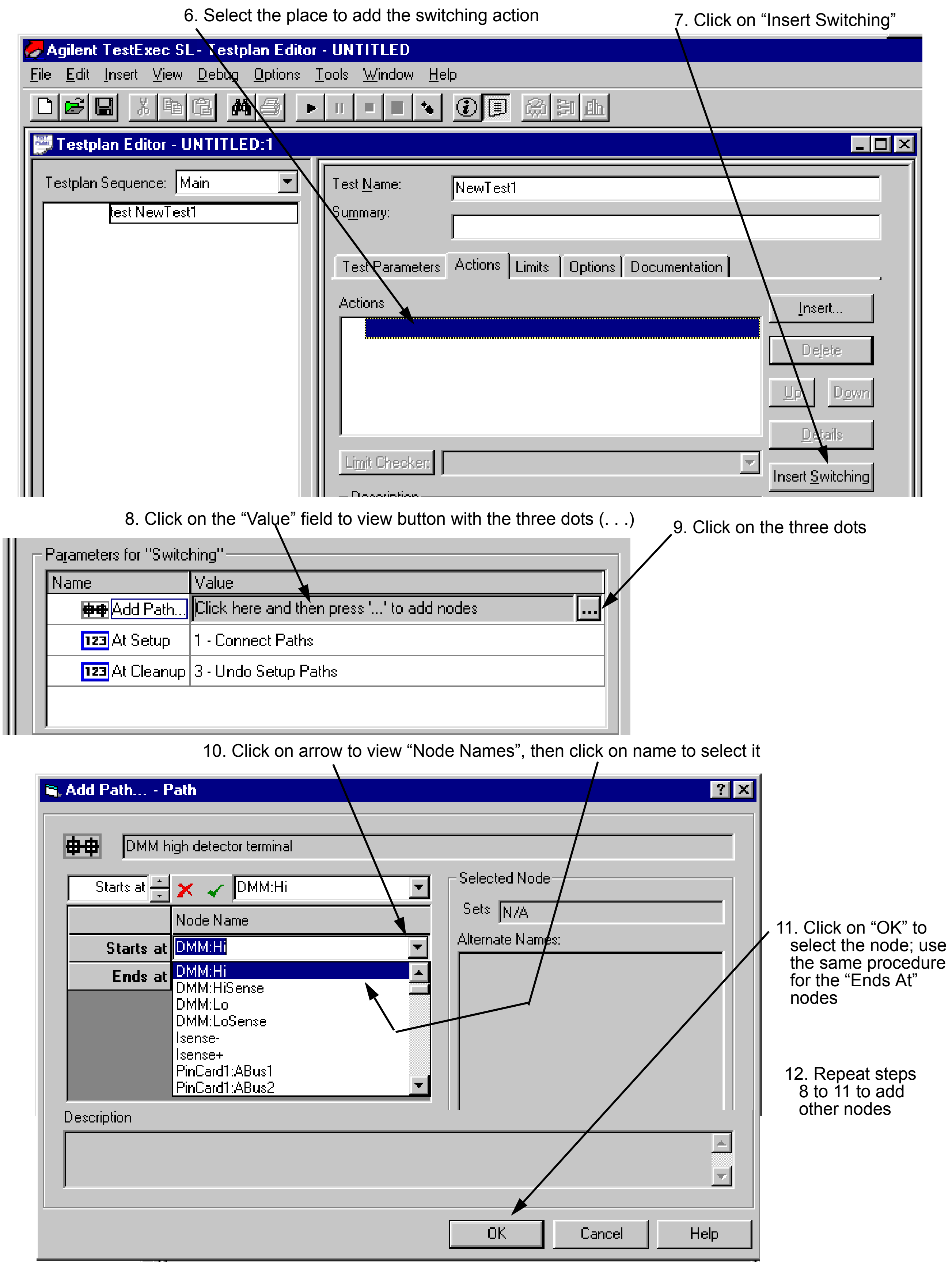
Or

click on “Apply” to add the action, but keep window open to add more actions

Click on “Cancel” to close window without adding action

Click on “Detail” to open Action Definition Editor

Figure 2-3. Adding an Action



6. Select the place to add the switching action

7. Click on “Insert Switching”

8. Click on the “Value” field to view button with the three dots (. . .)

9. Click on the three dots

10. Click on arrow to view “Node Names”, then click on name to select it

11. Click on “OK” to select the node; use the same procedure for the “Ends At” nodes

12. Repeat steps 8 to 11 to add other nodes

Agilent TestExec SL - Testplan Editor - UNTITLED

Testplan Editor - UNTITLED:1

Testplan Sequence: Main

test NewTest1

Test Name: NewTest1

Summary:

Parameters for "Switching"

Name	Value
Add Path...	Click here and then press '...' to add nodes
At Setup	1 - Connect Paths
At Cleanup	3 - Undo Setup Paths

Add Path... - Path

DMM high detector terminal

Starts at: DMM:Hi

	Node Name
Starts at	DMM:Hi
Ends at	

Sets: N/A

Figure 2-4. Adding a Switching Action

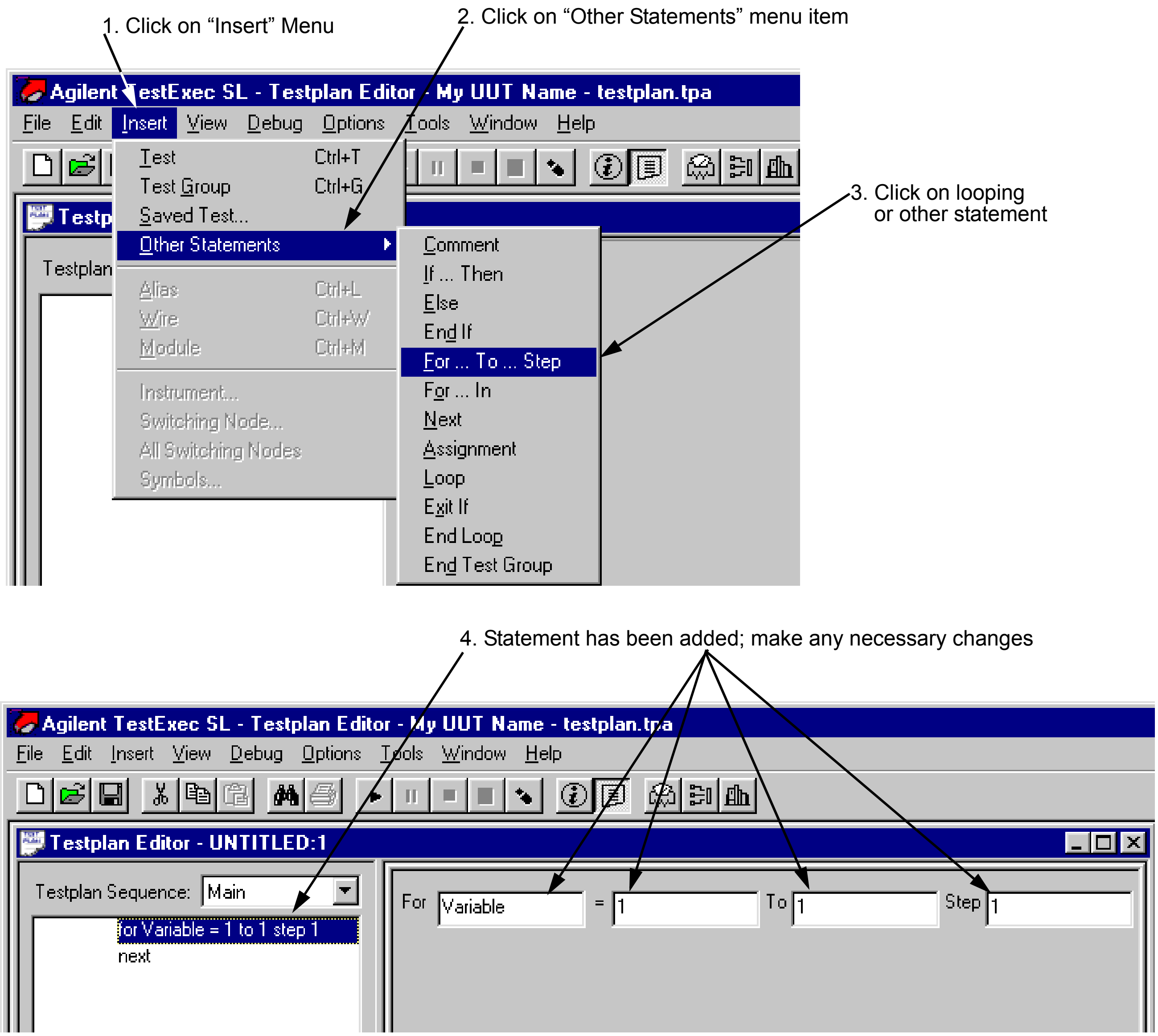
Using Software Debug Features

This section describes specific software features, unique to the Agilent TestExec SL version 2.0 and later, that you will find helpful in creating and debugging your tests and testplans. To enable Agilent TestExec SL, use the procedure in “Selecting Agilent TestExec SL” on page 17, if not enabled.

Looping

The loop constructs are `FOR / NEXT` sequencer statements. For example, a testplan developer would stop a testplan (if one is running), add `FOR` and `NEXT` lines to the testplan and then restart the testplan from the beginning. Refer to the Agilent TestExec SL documentation for additional information.

Figure 2-5 shows how to select and insert the looping statements into a testplan.



1. Click on “Insert” Menu

2. Click on “Other Statements” menu item

3. Click on looping or other statement

4. Statement has been added; make any necessary changes

Testplan Sequence: Main

```
for Variable = 1 to 1 step 1
next
```

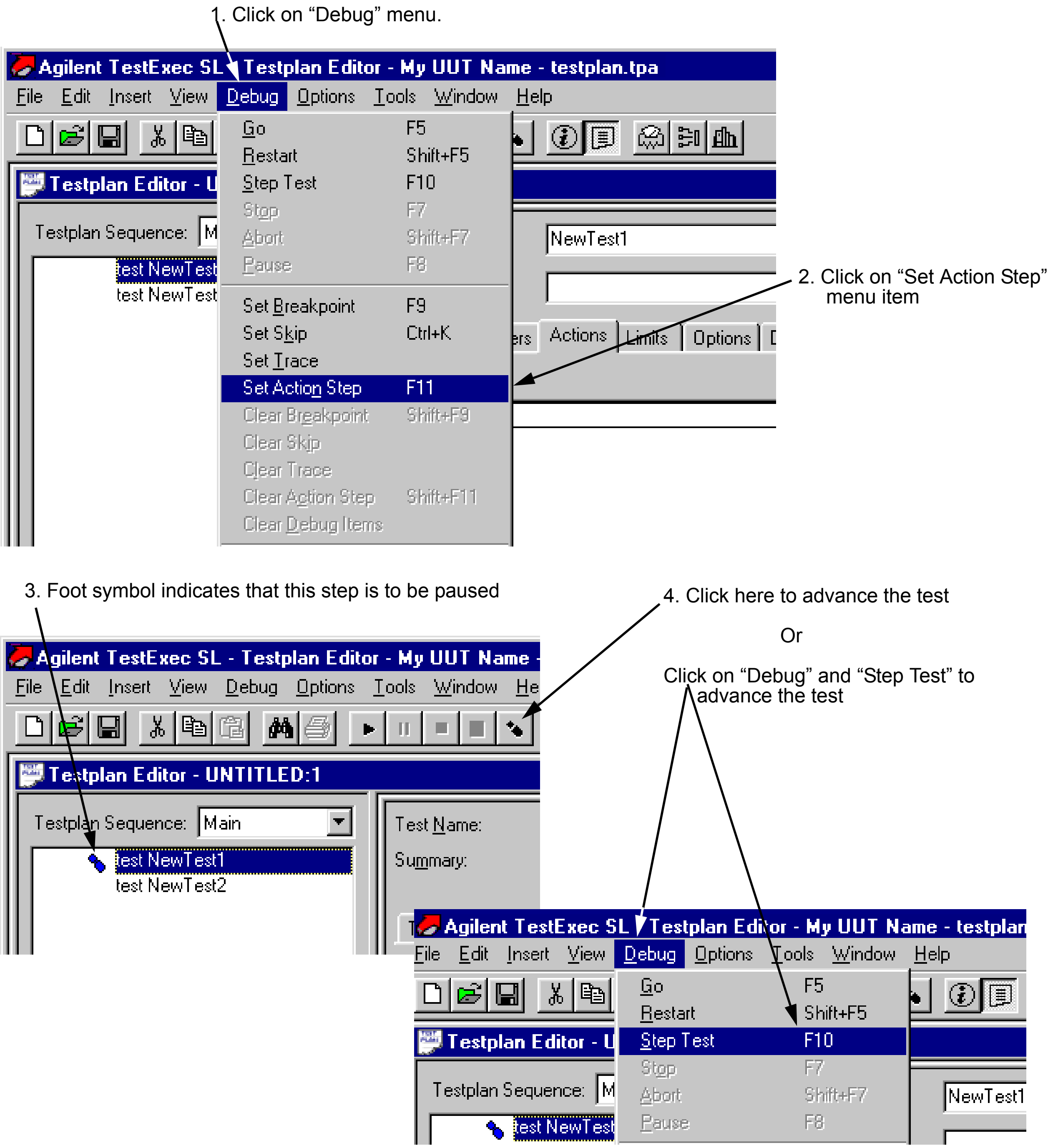
For Variable = 1 To 1 Step 1

Figure 2-5. Selecting Looping and Other Statements

Single Stepping

Agilent TestExec SL adds the ability to single step the execution of the operations that make up a test. When stepping through the actions of a test, the system pauses at entry point calls for each action routine. Action stepping can be performed any time testplan execution is paused (as a result of operator pause or test breakpoint). The action step which is currently paused is listed in the trace window. Refer to the Agilent TestExec SL documentation for additional information.

Figure 2-6 shows how to select the single stepping



1. Click on "Debug" menu.
Agilent TestExec SL - Testplan Editor - My UUT Name - testplan.tpa
File Edit Insert View Debug Options Tools Window Help
Go F5
Restart Shift+F5
Step Test F10
Stop F7
Abort Shift+F7
Pause F8
Set Breakpoint F9
Set Skip Ctrl+K
Set Trace
Set Action Step F11
Clear Breakpoint Shift+F9
Clear Skip
Clear Trace
Clear Action Step Shift+F11
Clear Debug Items
Testplan Editor - U
Testplan Sequence:
test NewTest
NewTest1
Actions
Limits
Options
2. Click on "Set Action Step" menu item
3. Foot symbol indicates that this step is to be paused
Agilent TestExec SL - Testplan Editor - My UUT Name
Testplan Editor - UNTITLED:1
Testplan Sequence: Main
test NewTest1
test NewTest2
Test Name:
Summary:
4. Click here to advance the test
Or
Click on "Debug" and "Step Test" to advance the test
Agilent TestExec SL - Testplan Editor - My UUT Name - testplan
Go F5
Restart Shift+F5
Step Test F10
Stop F7
Abort Shift+F7
Pause F8

Figure 2-6. Selecting Action Stepping

Action Debug Messages

Agilent TestExec SL allows action routines to send strings to the same debug window that the system uses for the test trace output. Use the following API:

- UtaTrace (MessageString, MessageID)

  -- MessageString is a string that will be sent to the trace window.
  -- MessageID is an optional string that allows the messages to be group (as identified by MessageID). Note: currently, this parameter does not provide any functionality.

Watch Window

Watch windows provide the ability to select 'objects' to dynamically monitor as a testplan executes. Select an item to watch from a list, the item is added to the watch window list and is updated each time the testplan pauses. Items can be added or deleted at any time (as long as the testplan is not running).

The watch window can monitor 4 types of objects:

- Instruments
- Switching Node
- All Switching Nodes
- Symbols

The following shows some typical example of the window.

Watch Window Menu Selections

Watch Window Menu Selections shows how to select one of the four Watch Window objects. The object to be selected, as shown in the figure, is the 'instrument' object. (The selection of the other objects is the same.)

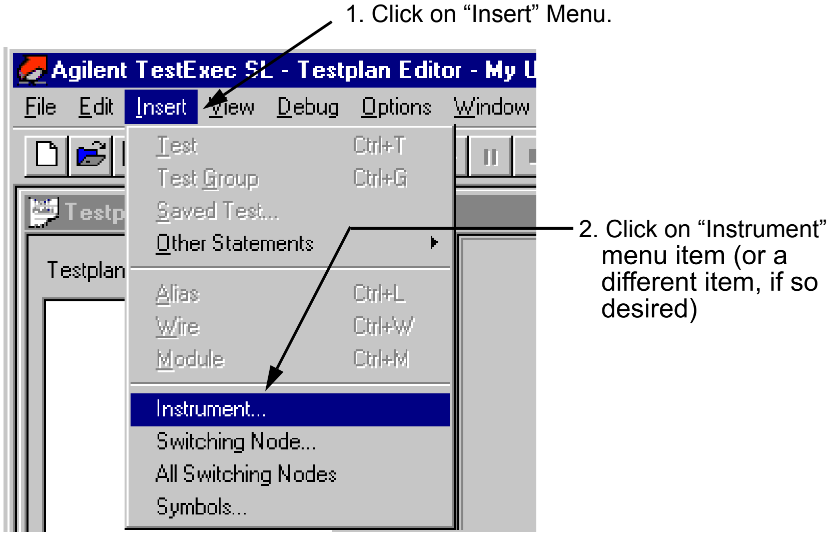


Figure 2-7. Watch Window Menu Selections

Watch Window Instrument Status

Watch windows are available for many of the instruments in an Agilent TS-5000 system. The instruments that can be monitored are:

- Agilent E1328A or E1418A Digital to Analog Converter Module (DAC)
- Agilent E1333A Counter Module
- Agilent E1411 Digit Multimeter Module (DMM)
- VXI Technology E1563A Analog to Digital Converter Module (ADC)
- Agilent E6171B Measurement Control Module (MCM)
- Agilent E6174A 32-Channel Event Detector Module
- Agilent 6642A, 6643A, 6652A, 6653A, 6673A Power Supplies

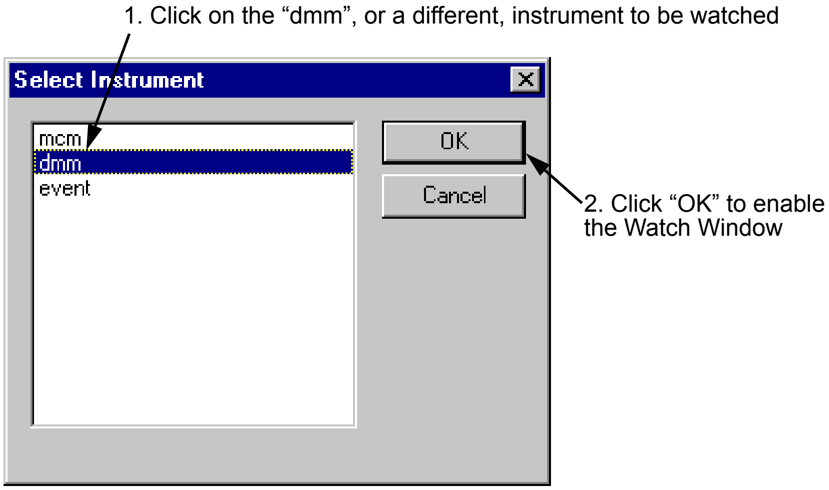
The displayed instrument states match the parameters of the instrument handler routines. For example the Agilent E1411 Digital Multimeter has the following routines which affect the state of the module:

- dmmConfFunction (dmm, Func, Range, Aperture);
- dmmConfCal (dmm, Linefreq, Autozero);
- dmmConfTrigIn (dmm, Trigselect, Trigcount, Delay);
- dmmConfSample (dmm, Sampsrc, Count, Period);

Figure 2-8 shows the menu box to select the instrument to be watched. The figure shows how to select the Digital Multimeter (dmm).

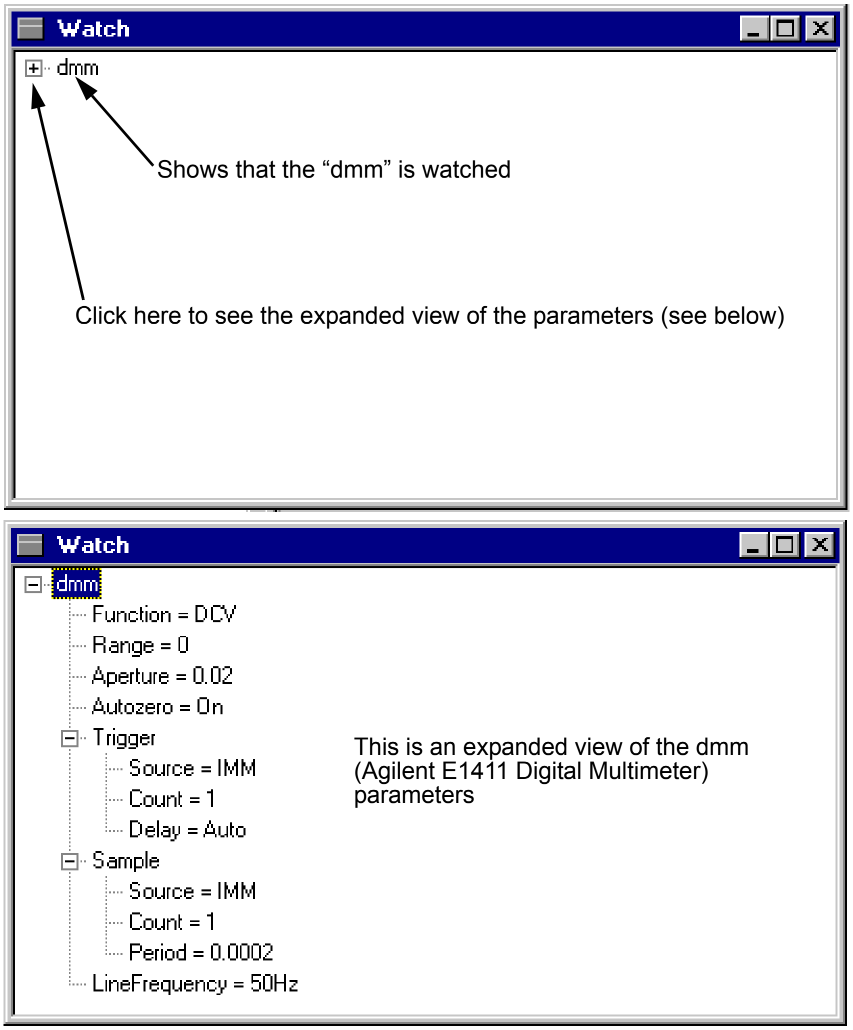
To add an instrument to the Watch Window, select the instrument type and press “OK”, as shown in Figure 2-8. The Watch Window is then displayed, as shown in Figure 2-9.

Figure 2-9 also shows the expanded tree view that shows the status of the ‘dmm’ parameters. The instruments display their state in a tree view. This allows a logical grouping of states and reduces the amount of data displayed by placing less important states in lower levels of the tree.


1. Click on the “dmm”, or a different, instrument to be watched

2. Click “OK” to enable the Watch Window

Figure 2-8. Selecting an Instrument to be Watched



```
⊞ dmm
```

Shows that the “dmm” is watched

Click here to see the expanded view of the parameters (see below)

```
⊟ dmm
   Function = DCV
   Range = 0
   Aperture = 0.02
   Autozero = On
 ⊟ Trigger
      Source = IMM
      Count = 1
      Delay = Auto
 ⊟ Sample
      Source = IMM
      Count = 1
      Period = 0.0002
   LineFrequency = 50Hz
```

This is an expanded view of the dmm (Agilent E1411 Digital Multimeter) parameters

Figure 2-9. Agilent E1411B Digital Multimeter Watch Window with Parameters

Instrument Debug Front Panels

Agilent TestExec SL provides a series of debug instrument front panels for the Agilent TS-5000 system instruments. The debug panels provide two main features:

- The ability to interactively view the current Unit Under test (UUT) state
- The ability to interactively view the current state of points within the system.

Debug Panel Types

The following debug panels are supplied with the system, dependent on the modules/instrument currently installed in the system.

- Agilent E1411B Digital Multimeter . . . . page 30
- Agilent E1328A & E1418A Digital to Analog Converter . . page 32
- Agilent E1333A Counter . . . . page 33
- VXI Technology E1563A Digitizer . . . . page 35
- Agilent E6171B Measurement Control Module . . . . page 36
- Agilent E6174A Event Detector . . . . page 37
- Agilent E6198A Switch/Load Unit. page 38
- Agilent 6642A, 6643A, 6652A, 6653A, 6673A Power Supplies page 39

Debug Panel Features

All of the debug panels provide the following features:

- Changes on the instrument front panel are not made until either:

  -- you initiate a measurement (for detectors) or
  -- you press the “Apply” button (for sources)

  The instrument state memory is modified in the instrument handler at this time (since this is when the instrument handler is called; See Instrument Handlers on page 48 for information about handlers).

- If the instrument is connected in the Agilent TS-5000 system through switching, the front panel provides a section which allows you to define the switching path. Either select a UUT pin to connect to directly or define a custom switch path.

  Connecting to the UUT pin is the easiest way to select a switch path; you select an ABus pin and the UUT pin. If the instrument has both the high and low terminals connected to the switch matrix, then both pins need to be defined. The low terminal is connected to UUTCommon by default and does not need to be selected, if you use the default. The UUT option is available if you have defined a UUT switching configuration file. In addition to the UUT pins, the system automatically adds the option to measure an ABus pin only. This will allows you to inject a signal on the front of the system and measure it using the instrumentation within the system.

- The state of the system is saved upon entering a pause state and is restored when entering the run state. This allows the panels to be used without affecting the testplan.

Selecting the Debug Panel

Debug Front Panel Selection and Select an Instrument from the Debug Front Panel Selection show how to enable the debug instrument front panels.

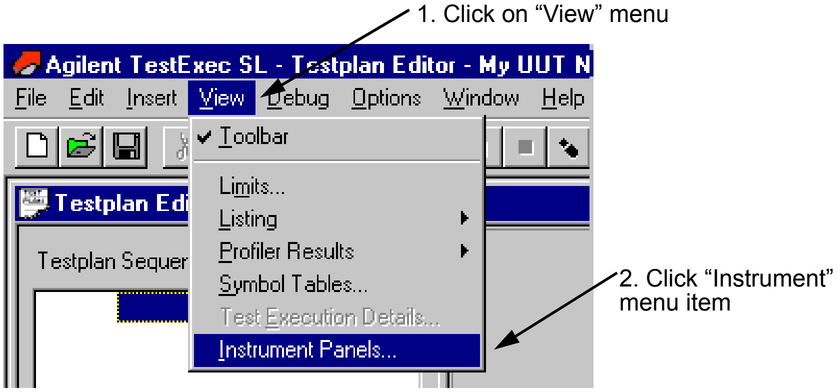

1. Click on “View” menu
Agilent TestExec SL - Testplan Editor - My UUT N
File Edit Insert View Debug Options Window Help
Toolbar
Limits...
Listing
Profiler Results
Symbol Tables...
Test Execution Details...
Instrument Panels...
Testplan Edi
Testplan Sequer
2. Click “Instrument” menu item

Figure 2-10. Debug Front Panel Selection

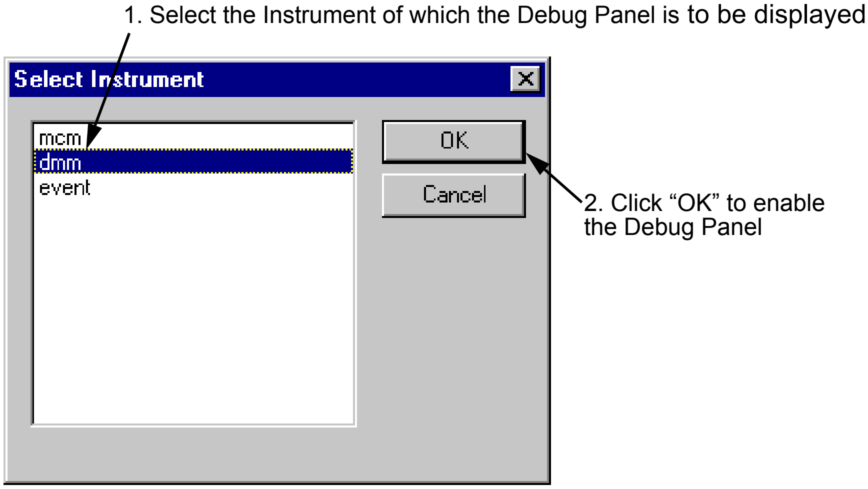
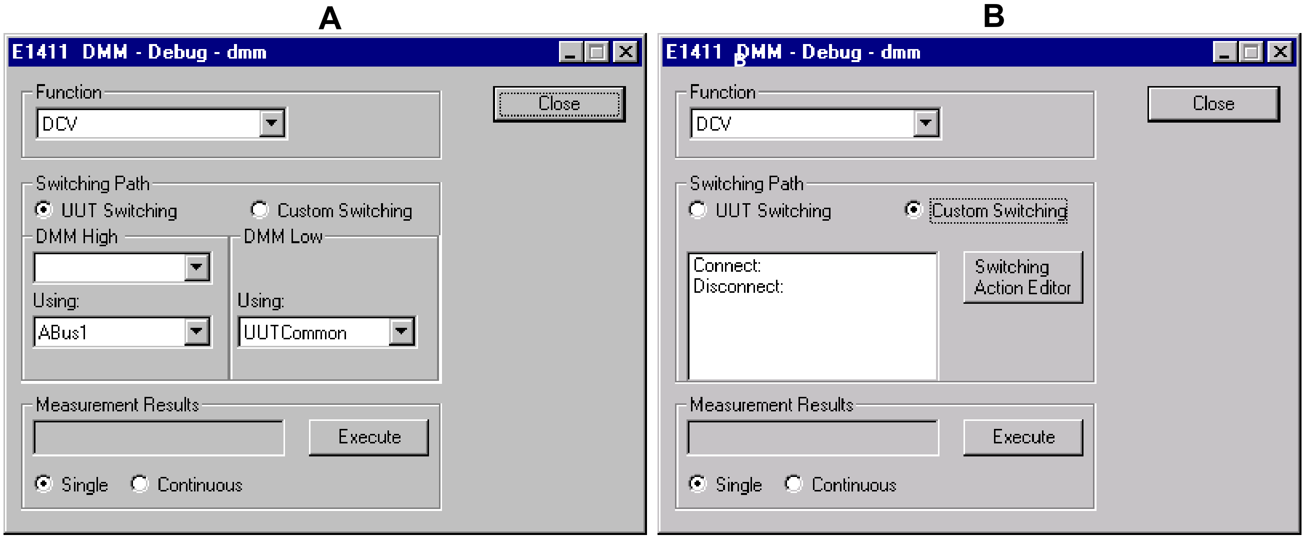

1. Select the Instrument of which the Debug Panel is to be displayed
Select Instrument
mcm
dmm
event
OK
Cancel
2. Click “OK” to enable the Debug Panel

Figure 2-11. Select an Instrument from the Debug Front Panel Selection

The following sections show and explain the different debug instrument panels.

Agilent E1411B Digital Multimeter

An instrument box which connects to the Agilent E6171 Measurement Control Module, such as the Agilent E1411 DMM, usually has two different dialog boxes which depend on the type of path selected (as shown in Agilent E1411 DMM Debug Front Panel (showing Path Selection)A and Agilent E1411 DMM Debug Front Panel (showing Path Selection)B). This allows you to either connect to the pin on the UUT (Agilent E1411 DMM Debug Front Panel (showing Path Selection)A) or select any point within the system by defining a custom switching path (Agilent E1411 DMM Debug Front Panel (showing Path Selection)B). The custom switching is applied on top of the current switching state, so to undo a switching path you need to define a 'disconnect' path.



A

E1411 DMM - Debug - dmm

Function: DCV

Switching Path: ● UUT Switching ○ Custom Switching

DMM High: Using: ABus1

DMM Low: Using: UUTCommon

Measurement Results:

● Single ○ Continuous

B

E1411 DMM - Debug - dmm

Function: DCV

Switching Path: ○ UUT Switching ● Custom Switching

Connect:
Disconnect:

Measurement Results:

● Single ○ Continuous

Figure 2-12. Agilent E1411 DMM Debug Front Panel (showing Path Selection)

The debug panel boxes and buttons do the following:

Function:

Choose ACV, DCV, Ohms, Ohms - Offset Compensated
The measurement is made in auto range and medium resolution.

Switching Path:

The debug panel supports the selection of the switching path. The panel switches both high and low connections of voltage sense and the high and low connection of the current terminals. The sense connections are connected in parallel with the high and low terminals. You specify the high or low terminal and the system automatically connects the associated sense terminal.

Measurement Results:

Press the "Single" button to initiate a single measurement; press the "Continuous" button to initiate a series of measurements. The continuous button is modified to a "Stop" button once pressed. When the dialog box is brought up, the measurement results box is blank.

Switching Action Editor:

This button enables the editor to select the switching paths for the

Digital Multimeter. See “Fast Connection Selection” on page 40 for a description on how to use the editor.

Execute:
This button starts a measurement.

Close:
This button closes the debug panel.

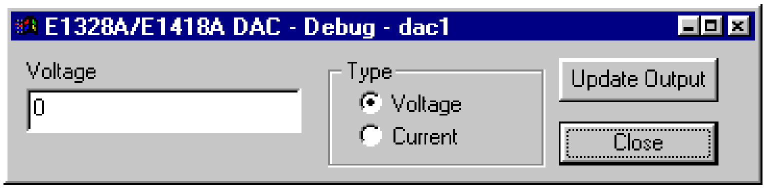
System Interface

The setting of the Digital Multimeter function uses the following actions:
dmmMeasureDCV, dmmMeasureACV, dmmMeasureOhms

The above actions also return the results of a measurement.

Agilent E1328A & E1418A Digital to Analog Converter

Agilent E1328A and E1418A DAC Debug Front Panel shows the dialog box for controlling the Agilent E1328/E1418 Digital-to-Analog Converter (DAC) Module.



E1328A/E1418A DAC - Debug - dac1

Voltage: 0

Type: ● Voltage ○ Current

Figure 2-13. Agilent E1328A and E1418A DAC Debug Front Panel

The debug panel boxes and buttons do the following:

Voltage or Current:
You have the option to specify the output voltage or current for each channel of the DAC. The ‘Type’ selection sets the voltage or current mode and defines contents of the input field.

Dependent on the type selected, enter the voltage value/current value into the Voltage/Current field.

(Note the E1328A cannot programmatically select the voltage or current option.)

Update Output:
This button changes the output setting.

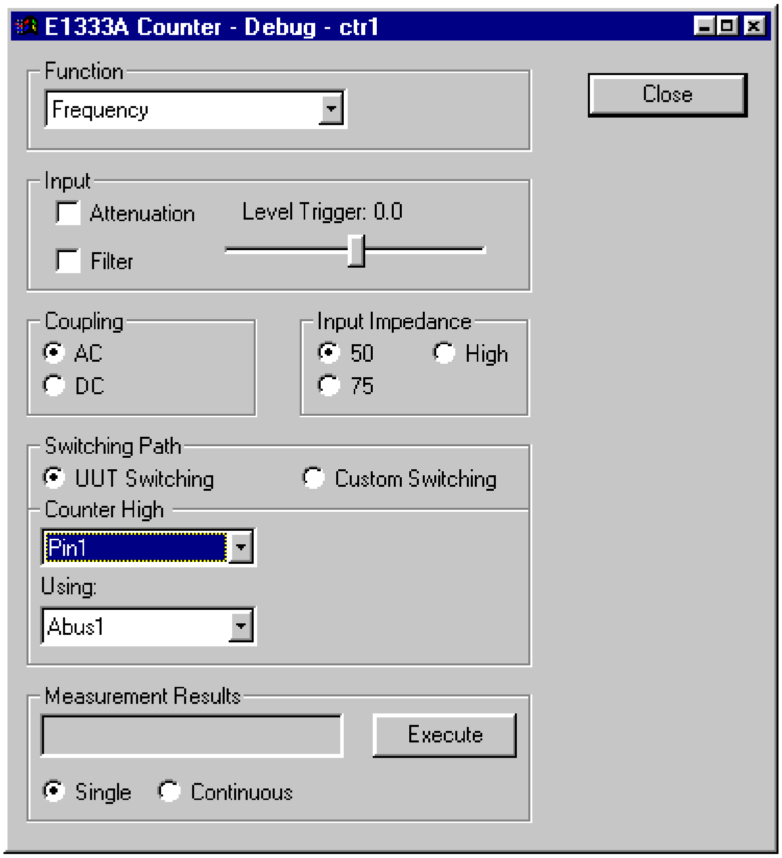
Close:
This button closes the debug panel.

System Interface

The setting of the DAC uses the following action routine:
dac16i_setup

Agilent E1333A Counter

Agilent E1333 Counter Debug Front Panel shows the dialog box for Agilent E1333A Counter. Only the UUT connection screen is shown. The custom path selection is the same used by the Digital Multimeter (see Figure 2-12 on page 30).



E1333A Counter - Debug - ctrl

Function: Frequency

Input: □ Attenuation □ Filter Level Trigger: 0.0

Coupling: ● AC ○ DC

Input Impedance: ● 50 ○ High ○ 75

Switching Path: ● UUT Switching ○ Custom Switching

Counter High: Pin1

Using: Abus1

Measurement Results: ● Single ○ Continuous

Figure 2-14. Agilent E1333 Counter Debug Front Panel

The debug panel boxes and buttons do the following:

Function:
Choose Frequency, Period, Positive Pulse Width, Negative Pulse Width, Totalize

Input, Coupling, Input Impedance:
Allows you to set input functions. The setting of each function is sent before a measurement is made.

Switching Path:
The counter panel supports the selection of a switching path. The counter switches only the high connections of the output. The low connections are automatically connected to system ground.

Measurement Results:

Select the “Single” button to initiate a single measurement or the “Continuous” button to initiate a series of measurements. The continuous button is changed to a “Stop” button, once pressed.

Note that when the dialog box is brought up, the measurement results box is blank.

Execute:

This button starts a measurement.

Close:

This button closes the debug panel.

System Interface

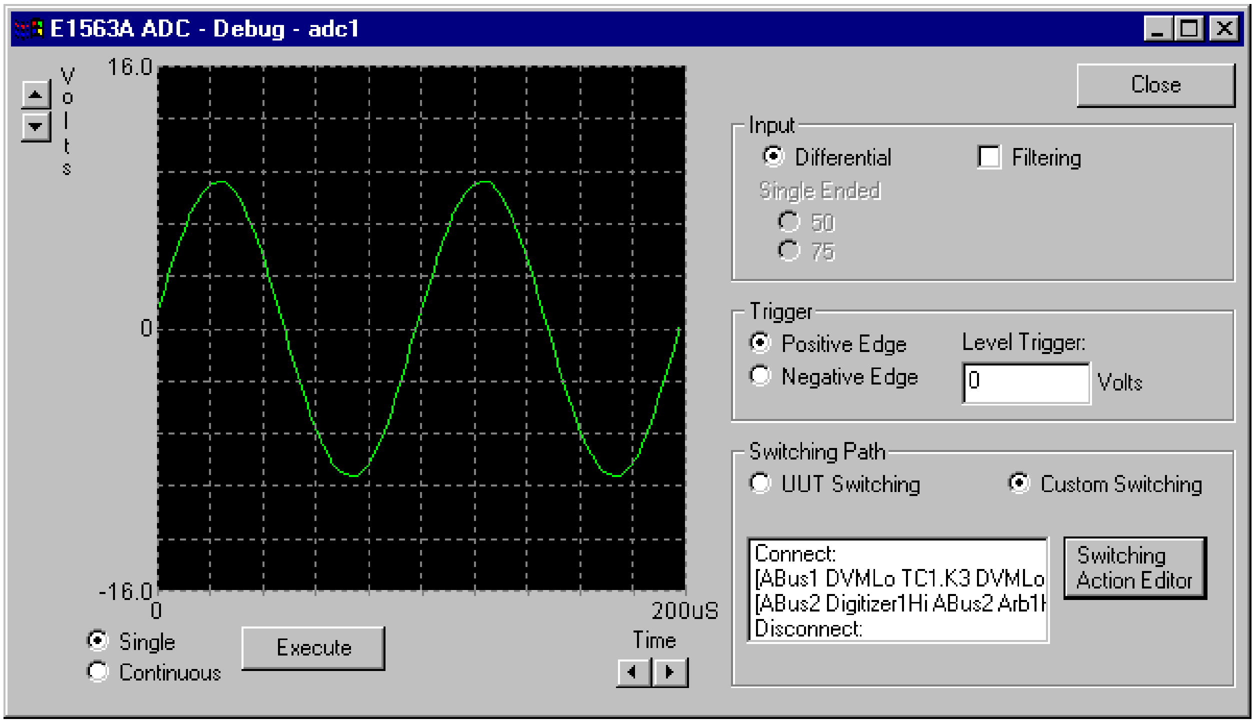
The setting of the counter function will use the following action routines:
ctrMeasureFrequency, ctrMeasurePeriod, ctrMeasurePulseWidth, ctrMeasureTimeInterval, ctrMeasureTotalize

The measurement uses the level trigger specified in the dialog box (using the slider control), a default range and default resolution.

The input section uses the following action routine:
ctrMeasureInControls

VXI Technology E1563A Digitizer

Figure 2-15 shows the debug front panel for the VXI Technology E1563A Digitizer. Only the UUT connection screen is shown. The custom path selection is the same as for the Agilent E1411B Digital Multimeter (see “Agilent E1411B Digital Multimeter” on page 30).



● Single
○ Continuous

Input
● Differential
□ Filtering
Single Ended
○ 50
○ 75

Trigger
● Positive Edge
○ Negative Edge
Level Trigger: 0 Volts

Switching Path
○ UUT Switching
● Custom Switching

Connect:
[ABus1 DVMLo TC1.K3 DVMLo
[ABus2 Digitizer1Hi ABus2 Arb1H
Disconnect:

Figure 2-15. VXI Technology E1563A Digitizer Debug Front Panel

The debug panel boxes and buttons do the following:

Time Base and Range:

The range and timebase can be modified by selecting the “arrow” keys on the side and bottom of the waveform. To update the display, take a new measurement.

The time base allows the following range of selections:
50 nS to 20 Sec in 1, 2, 5 steps
the range has the following selections:
0.1 to 100 volts in 1, 2, 5 steps.
When the dialog box is brought up, the combo box contains the current time base and range. All of the setting for the digitizer are saved until an actual sweep is initiated.

Input and Triggering:

These two sections allow you to set a variety of input and trigger functions. The setting of each function is sent when a measurement is made.

Switching Path:

The debug panel supports the selection of a switching path. The digitizer switches both the high and low connections of the output.

Waveform Display:

Select the “Single” button to generates a single waveform display or the “Continuous” button to generate a continuous update of waveforms. The display updates at a rate of >10 update/sec.

Close:

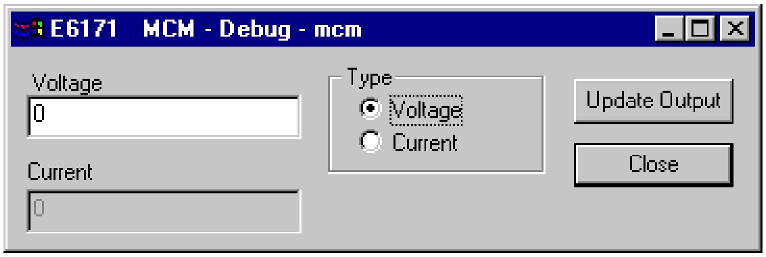
This button closes the debug panel.

System Interface

The setting of the digitizer function will use the following action routines:adcConfArm, dcConfFreq, adcConfInControls, adcIsSet, adcGetResults

Agilent E6171B Measurement Control Module

Agilent E6171B Measurement Control Module Debug Front Panel shows the dialog box for controlling the Agilent E6171B Measurement Control Module.



E6171 MCM - Debug - mcm

Voltage: 0

Current: 0

Type: ● Voltage ○ Current

Figure 2-16. Agilent E6171B Measurement Control Module Debug Front Panel

The debug panel boxes and buttons do the following:

Voltage or Current:

You have the option to specify the output voltage or current for each channel of the MCM. The ‘Type’ selection sets the voltage or current mode.

Dependent on the type selected, enter the voltage value or current value into the Voltage or Current field, respectively.

Expected Current (when enabled by the Type-Current):

Defines the expected current of the MCM (or VI, i.e., Voltage/Current) source.

Update Output:

This button changes the output source of the MCM.

Close:

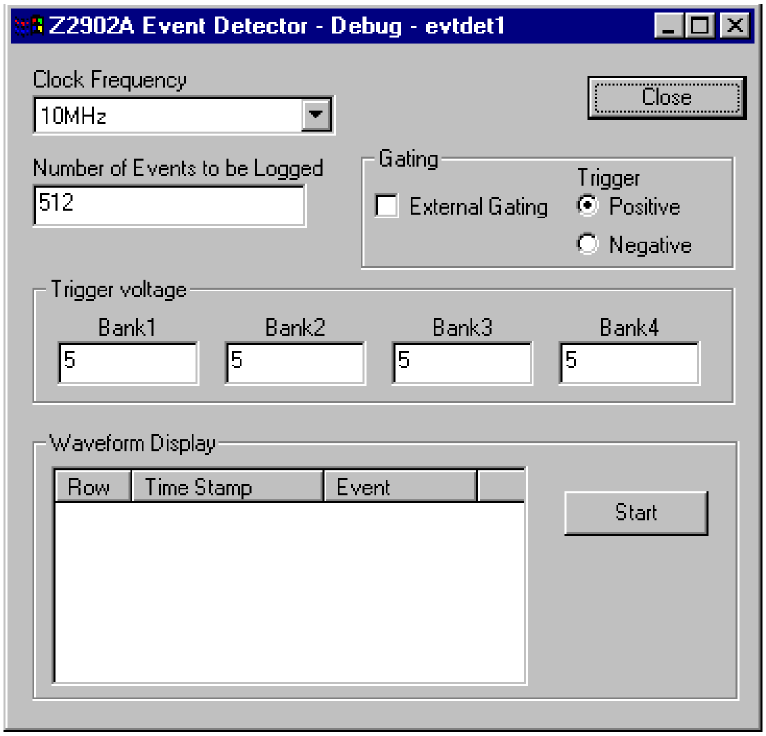
This button closes the debug panel.

System Interface

The setting of the counter function will use the following action routines:
viSetSourceDCI, viSetSourceDCV, viSet

Agilent E6174A Event Detector

Agilent E6174A Event Detector Debug Front Panel shows the dialog box for controlling the Event Detector Module.


Z2902A Event Detector - Debug - evtdet1

Clock Frequency: 10MHz

Number of Events to be Logged: 512

Gating: □ External Gating

Trigger: ● Positive ○ Negative

Trigger voltage: Bank1 5 Bank2 5 Bank3 5 Bank4 5

Waveform Display

Row	Time Stamp	Event

Figure 2-17. Agilent E6174A Event Detector Debug Front Panel

The debug panel boxes and buttons do the following:

Clock Frequency:
Select the clock rate of the event detector module.

Edge Trigger, External Gating and Number of Events:
Specifies the type of edge triggering to be used and external gating. The text box defines the number of events to be logged before a measurement is complete.

Event Display:
The event detector can generate either a single display or a continuous update of events. The table provides a simple list view of the event and it’s time stamp. The column width can be adjust by dragging the line which separates the titles.

Execute:
This button starts a measurement.

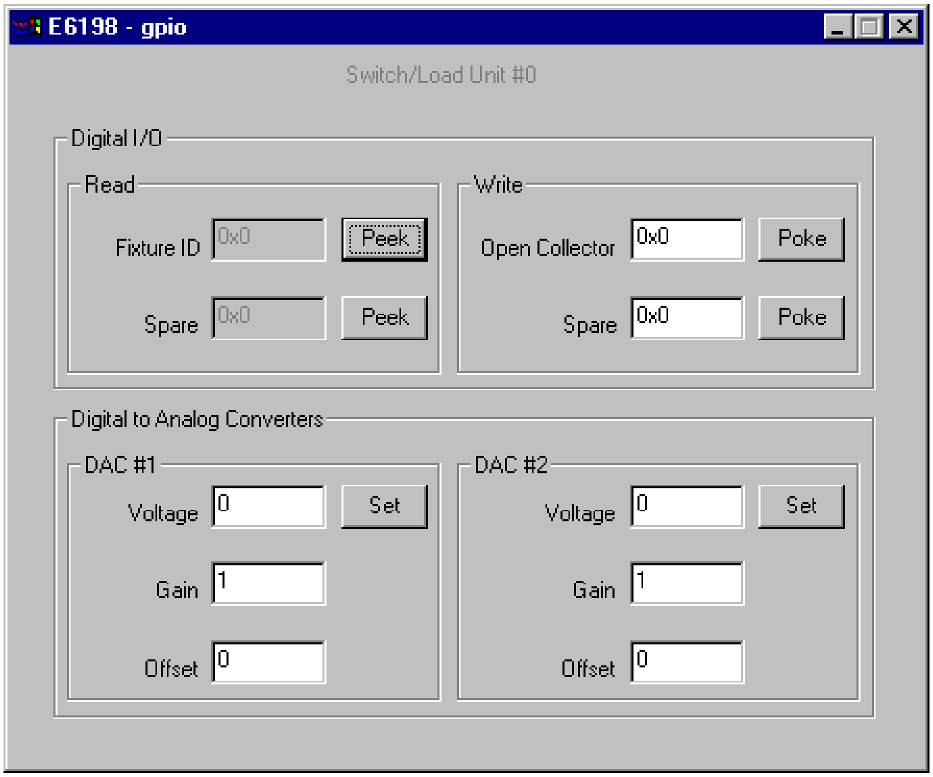
Close:
This button closes the debug panel.

System Interface

The setting of the event detector will use the following action routines:
eventMeasure

Agilent E6198A Switch/Load Unit

Agilent E6198A Switch/Load Unit Debug Front Panel shows the dialog box for controlling the Switch/Load Unit.



E6198 - gpio

Switch/Load Unit #0

Digital I/O

Read

Fixture ID 0x0

Spare 0x0

Write

Open Collector 0x0

Spare 0x0

Digital to Analog Converters

DAC #1

Voltage 0

Gain 1

Offset 0

DAC #2

Voltage 0

Gain 1

Offset 0

Figure 2-18. Agilent E6198A Switch/Load Unit Debug Front Panel

The debug panel boxes and buttons do the following:

Digital I/O Read:
Reads a value from the Agilent E6198 switch/load unit digital input ports. Fixture ID is value from Fix_ID(0..7) of access connector J104 and Spare is value from Spare_DigIn(0..7) of access connector J104.

Digital I/O Write:
Write a value to the Agilent E6198 switch/load unit digital output ports. Open Collector writes to output port on the switch/load unit backplane. Spare writes to output Spare_DigOut(0..7) on backplane connector J104.

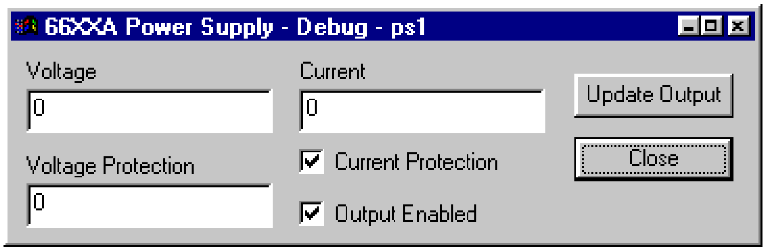
Digital to Analog Converter DAC #1/DAC #2:
Change the voltage, gain, and offset values used to set the DAC voltage in the switch/load unit for both DAC #1 and DAC #2.

System Interface

The setting of the switch/load unit will use the following action routines: digitalWriteSU, digitalReadSU, dacSetDCVSU, dacSetGainOffsetSU

Agilent 6642A, 6643A, 6652A, 6653A, 6673A Power Supplies

Power Supply Debug Front Panel shows the dialog box for controlling the Agilent 6642A, 6643A, 6653A, 6673A Power Supplies. You can specify the output voltage or current for each channel of the Power Supply. You must specify a protection voltage greater than the voltage setting.



66XXA Power Supply - Debug - ps1

Voltage: 0

Current: 0

Voltage Protection: 0

☑ Current Protection

☑ Output Enabled

Figure 2-19. Power Supply Debug Front Panel

The debug panel boxes and buttons do the following:

Voltage/Current:

Specifies the output voltage or current for each channel of the power supplies. The ‘Type’ selection sets the voltage or current mode.

Dependent on the type selected, enter the voltage value or current value into the Voltage or Current field, respectively.

Update Output:

This button changes the power supply output.

Close:

This button closes the debug panel.

System Interface

The setting of the counter function uses the following action routines:
psConfVI, psSet, psIsSet, psConnect, psDisconnect

Fast Connection Selection

You can select a switching path to a module/instrument either by selecting pins on the Unit Under Test (UUT) (see “Switching Path” box in Agilent E1411 DMM Debug Front Panel (showing Path Selection)A on page 30) or by creating a custom path using the Switching Action Editor (see “Switching Action Editor” button in Agilent E1411 DMM Debug Front Panel (showing Path Selection)B on page 30).

Specifying Unit Under Test Pins

When selecting UUT pins, you normally specify from 1 to 4 connections, depending on the module/instrument. For example, for the Digital Multimeter (DMM), you define these four connections: high, low, high current sense, and low current sense. The Counter only has only a high connection.

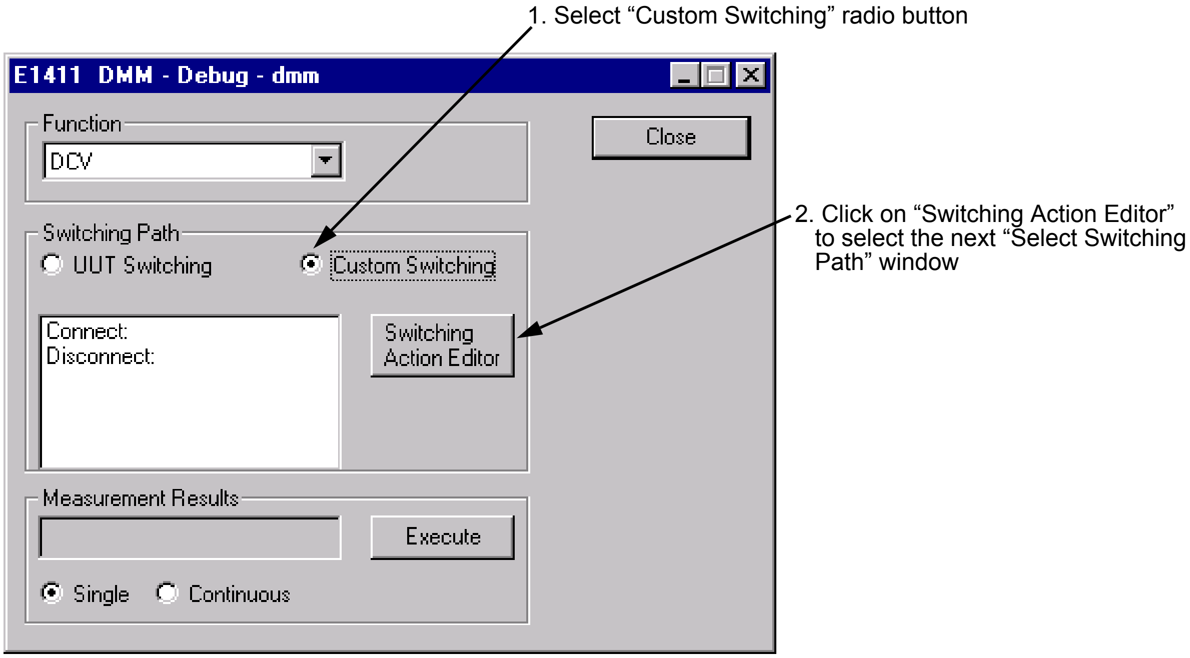
The software works backwards though the switching configuration and generates a switching path which connects to ABus*x*, depending on the option selected. For example, a typical connection from ABus1 to the high input of the Digital Multimeter would be: [DVMHi ABus1 VISrcHi]

Creating a User Defined Switching Path

User defined switching paths are created using the “Select Switching Path” screen which is enabled by the “Switching Action Editor” button (see Agilent E1411 DMM Debug Front Panel (showing Path Selection)B on page 30). The “Select Switching Path” screen shows two different fields, the ‘Connect’ and ‘Disconnect’ fields. All paths added or listed in the ‘Connect’ field are always connected. All paths added or listed in the ‘Disconnect’ field are always disconnected. Note that the fields are blank, if no paths have previously been added.

To add/edit/delete a path, first enable the “Select Switching Path” screen using an appropriate instrument panel, as shown in Figure 2-20. The figure uses the Agilent E1411B Digital Multimeter instrument panel.

The following shows how to add/edit/delete a path.



E1411 DMM - Debug - dmm

Function: DCV

Switching Path: ○ UUT Switching ● Custom Switching

Connect:
Disconnect:

Measurement Results:

● Single ○ Continuous

1. Select “Custom Switching” radio button

2. Click on “Switching Action Editor” to select the next “Select Switching Path” window

Figure 2-20. Enabling the “Select Switching Path” Screen

Adding a New Path

To add a new path, first determine the end nodes or points in the path. Then use the “Switching Action Editor” to find a path using the intermediate nodes between the end notes or points.

For example, to connect the Hi input of the Agilent E1411B Digital Multimeter (DMM) to the Hi output of the Agilent E6171 Measurement Control Module (MCM), you must know the node names for these connections. Also, since the DMM has no direct connections to the VI source or MCM, it must connect using ABus*x*. Thus, the nodes for this path would be: [DVMHi ABus1 VISrcHi].

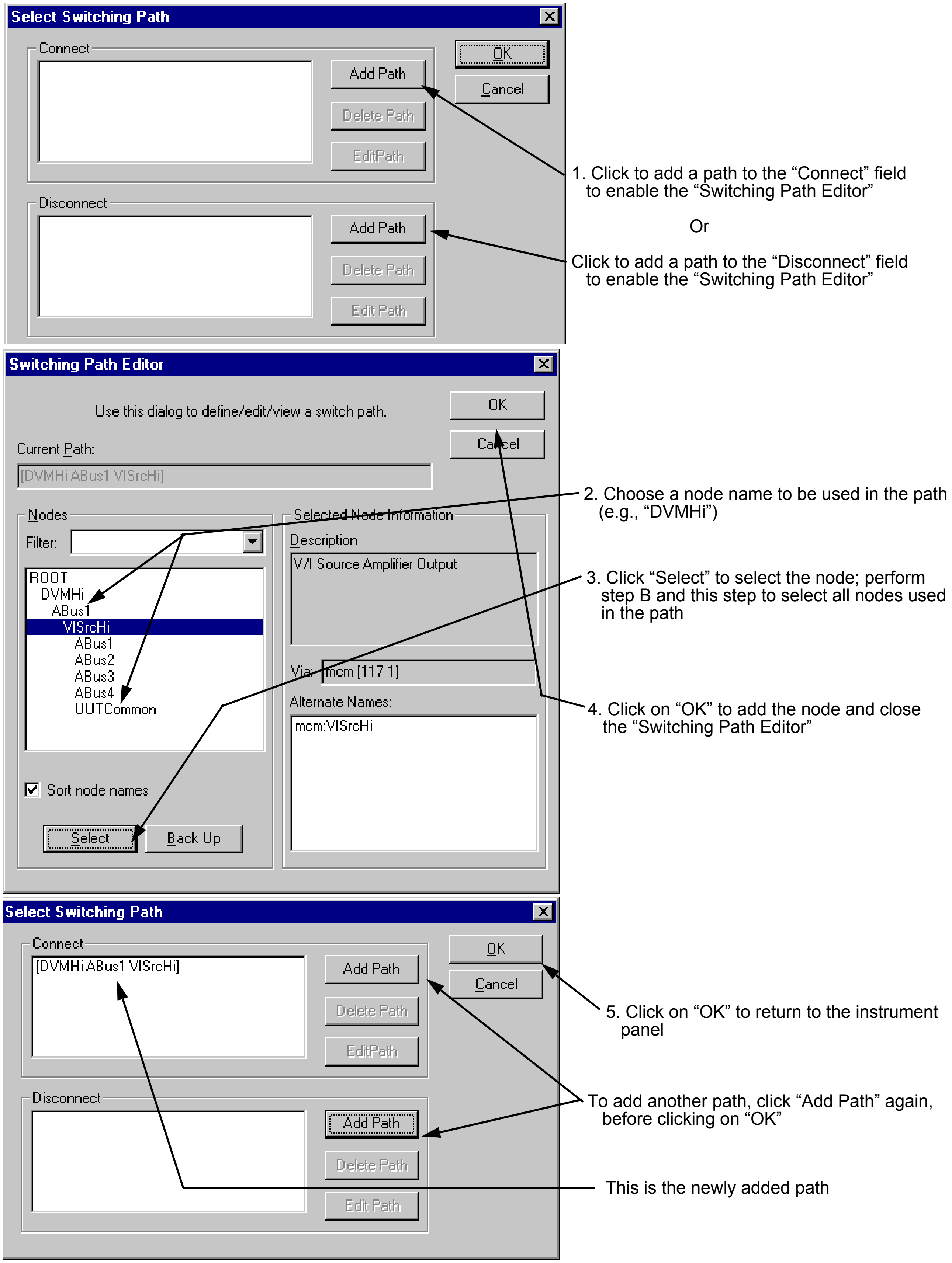
Before adding a new path, first use the procedure in Figure 2-20 to enable the “Select Switching Path” screen. Then use the procedure in Figure 2-21 to add the path.

Editing a Switching Path

To edit a path, first use the procedure in Figure 2-20 to enable the “Select Switching Path” screen, if the screen is not enabled. Then use the procedure in Figure 2-22 to edit the path.

Deleting a Path

To delete a path, first use the procedure in Figure 2-20 to enable the “Select Switching Path” screen, if the screen is not enabled. Then use the procedure in Figure 2-23 to delete the path.



1. Click to add a path to the "Connect" field to enable the "Switching Path Editor"

Or

Click to add a path to the "Disconnect" field to enable the "Switching Path Editor"

Use this dialog to define/edit/view a switch path.

Current Path: [DVMHi ABus1 VISrcHi]

```
ROOT
  DVMHi
    ABus1
      VISrcHi
        ABus1
        ABus2
        ABus3
        ABus4
        UUTCommon
```

☑ Sort node names

Description: V/I Source Amplifier Output

Via: mcm [117 1]

Alternate Names: mcm:VISrcHi

2. Choose a node name to be used in the path (e.g., "DVMHi")

3. Click "Select" to select the node; perform step B and this step to select all nodes used in the path

4. Click on "OK" to add the node and close the "Switching Path Editor"

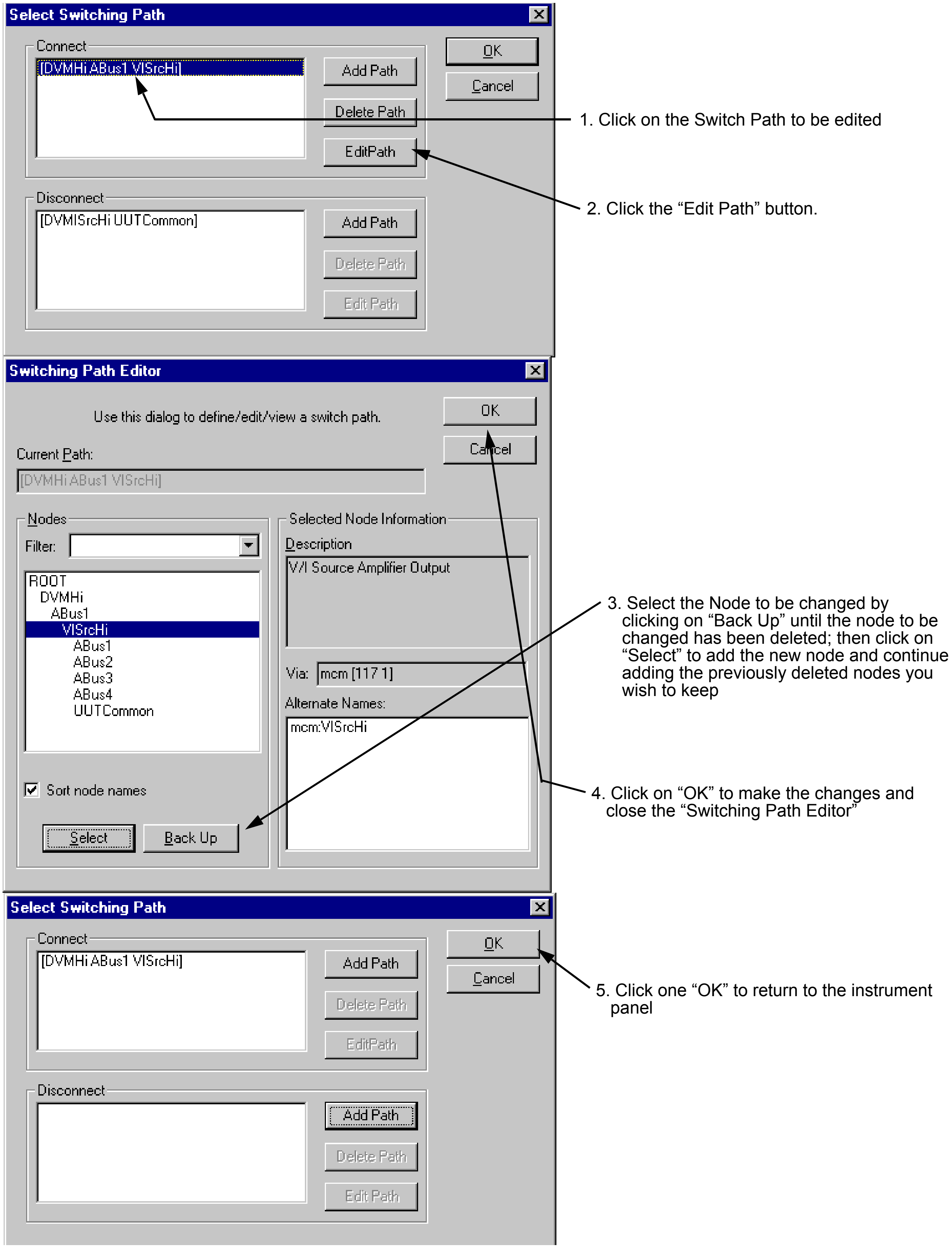
Connect: [DVMHi ABus1 VISrcHi]

5. Click on "OK" to return to the instrument panel

To add another path, click "Add Path" again, before clicking on "OK"

This is the newly added path

Figure 2-21. Adding a New Switching Path



Select Switching Path

Connect: [DVMHi ABus1 VISrcHi]

Disconnect: [DVMISrcHi UUTCommon]

Switching Path Editor

Use this dialog to define/edit/view a switch path.

Current Path: [DVMHi ABus1 VISrcHi]

```
ROOT
  DVMHi
    ABus1
      VISrcHi
        ABus1
        ABus2
        ABus3
        ABus4
        UUTCommon
```

☑ Sort node names

Description: V/I Source Amplifier Output

Via: mcm [117 1]

Alternate Names: mcm:VISrcHi

Select Switching Path

Connect: [DVMHi ABus1 VISrcHi]

Disconnect:

1. Click on the Switch Path to be edited

2. Click the “Edit Path” button.

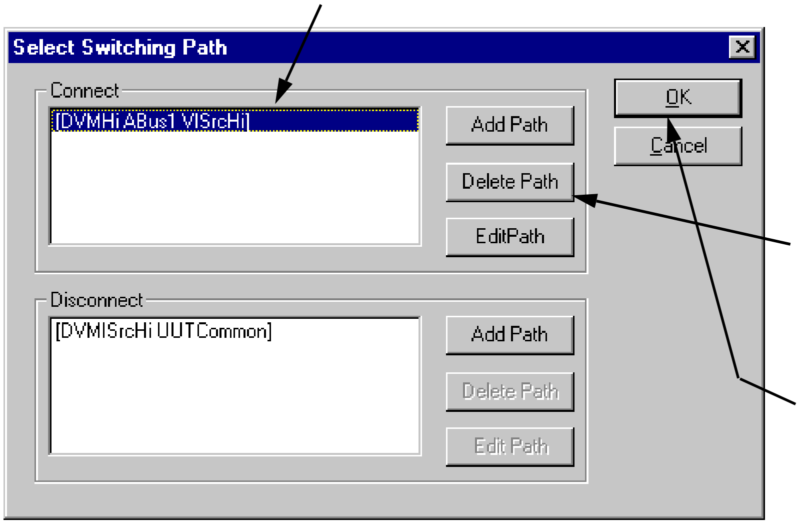
3. Select the Node to be changed by clicking on “Back Up” until the node to be changed has been deleted; then click on “Select” to add the new node and continue adding the previously deleted nodes you wish to keep

4. Click on “OK” to make the changes and close the “Switching Path Editor”

5. Click one “OK” to return to the instrument panel

Figure 2-22. Editing a Switching Path

1. Click on the Path to be deleted



[DVMHi ABus1 VISrcHi]

[DVMISrcHi UUTCommon]

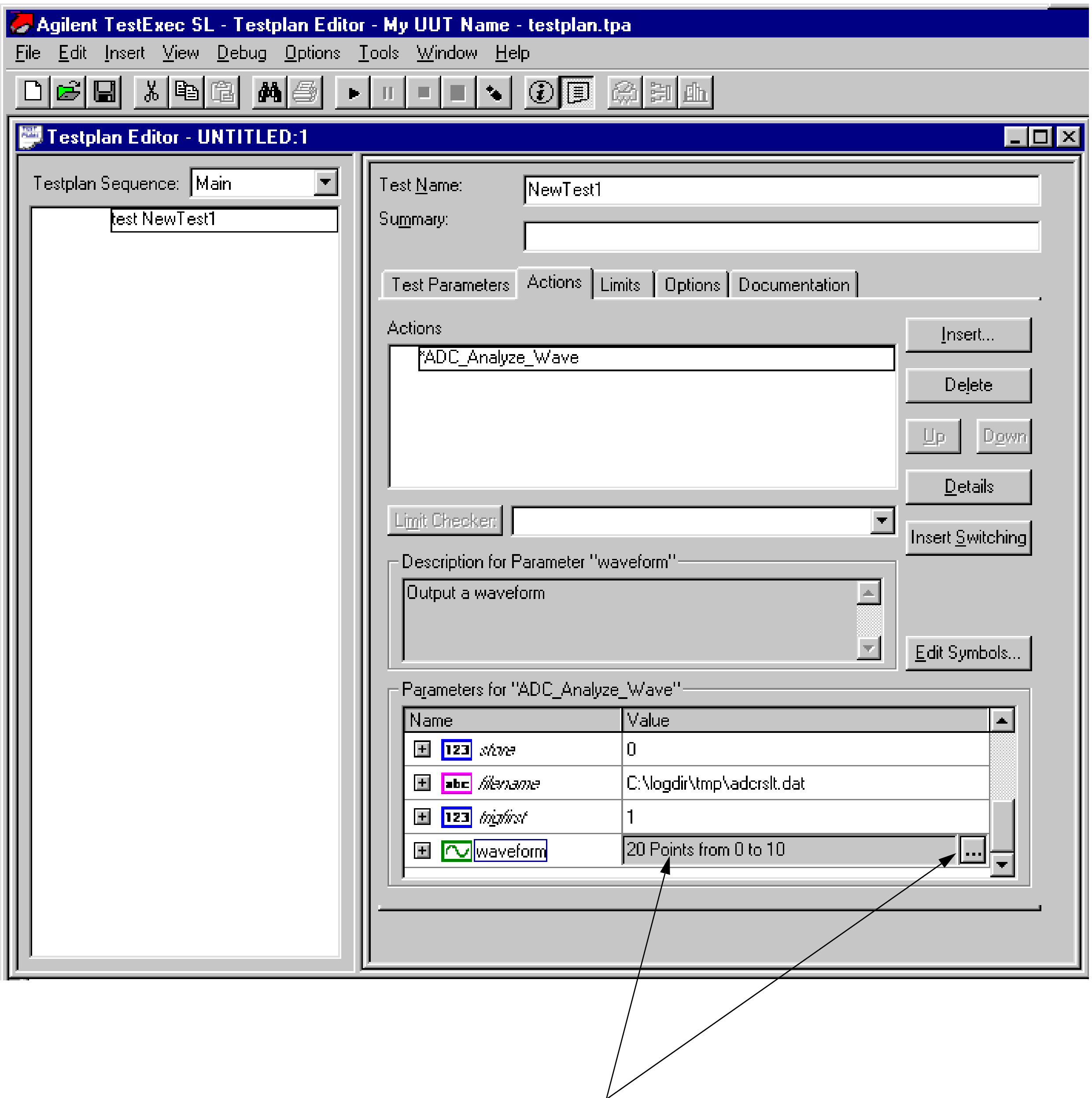
2. Click on “Delete Path” to delete the path

3. Click on “OK” to return to the instrument panel

Figure 2-23. Deleting a Switching Path

Viewing Waveforms

A ‘Waveform’ data type is available in the “ADC_Analyze_Wave” High-Level Action. This data type includes a graphical viewer for data. Sample Testplan Showing Waveform Data Type shows a sample testplan using the “ADC_Analyze_Wave” Action with the Waveform parameter. Range Data on the Waveform Display Graphical Editor and Sample Waveform Display show the waveform editor, and Figure 2-27 shows the waveform display.


Agilent TestExec SL - Testplan Editor - My UUT Name - testplan.tpa
File Edit Insert View Debug Options Tools Window Help
Testplan Editor - UNTITLED:1
Test Parameters
Actions
Limits
Options
Documentation
Insert...
Delete
Up
Down
Details
Limit Checker:
Insert Switching
Edit Symbols...

Testplan Sequence: Main

test NewTest1

Test Name: NewTest1

Summary:

Actions

*ADC_Analyze_Wave

Description for Parameter "waveform"

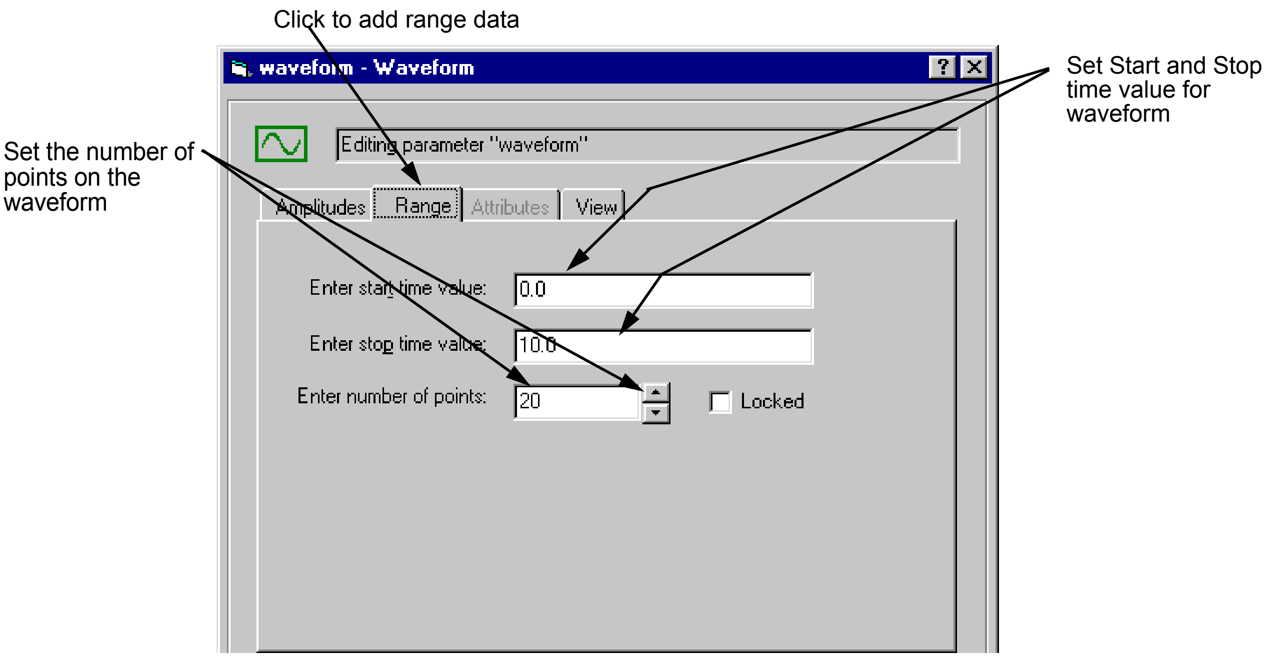
Output a waveform

Parameters for "ADC_Analyze_Wave"

Name	Value
123 slot	0
abc filename	C:\logdir\tmp\adcrslt.dat
123 trigfirst	1
waveform	20 Points from 0 to 10

Click on the parameter value to select it then click on the dots (...) to enable the waveform editor

Figure 2-24. Sample Testplan Showing Waveform Data Type



Click to add range data

Set the number of points on the waveform

Set Start and Stop time value for waveform

Editing parameter "waveform"

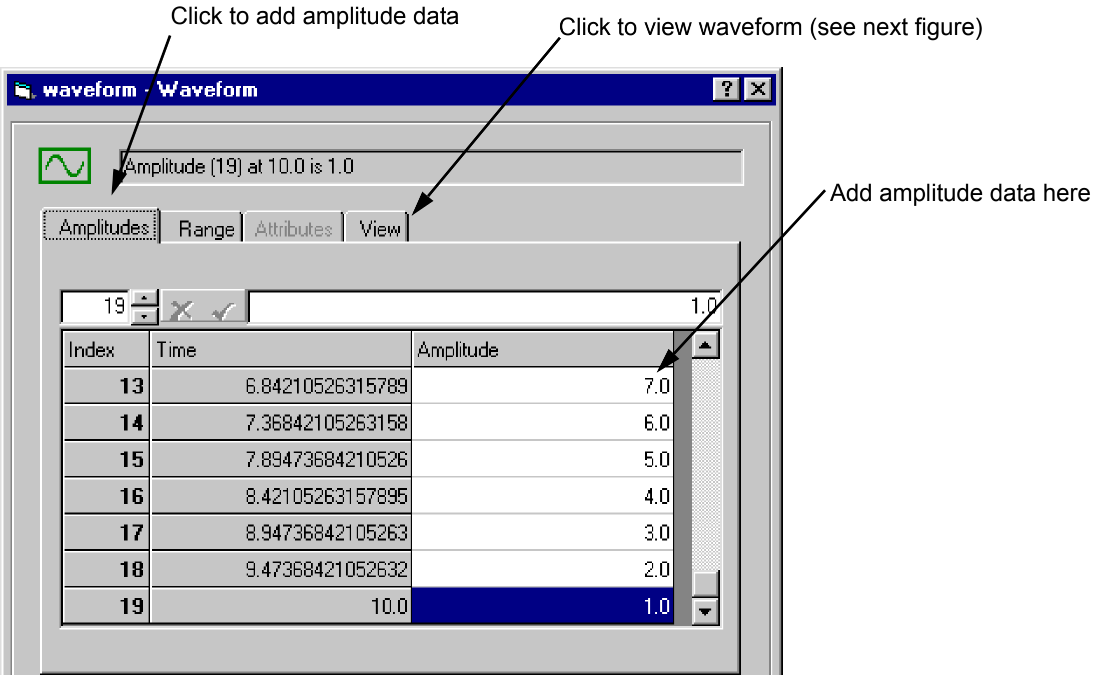
Enter start time value: 0.0

Enter stop time value: 10.0

Enter number of points: 20

□ Locked

Figure 2-25. Range Data on the Waveform Display Graphical Editor



Click to add amplitude data

Click to view waveform (see next figure)

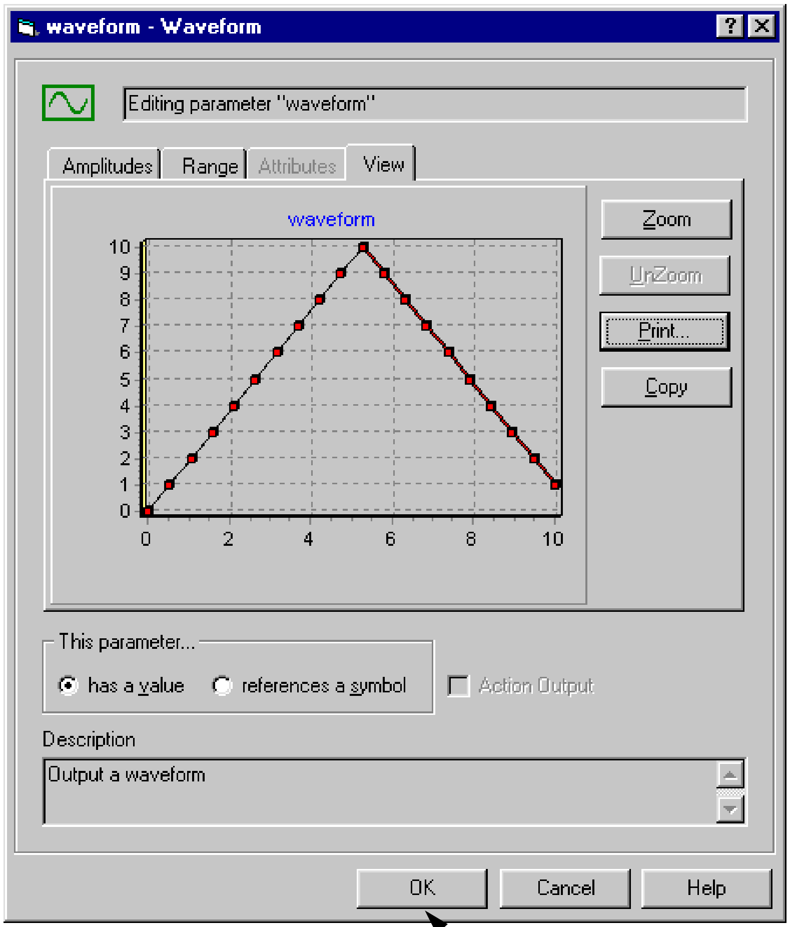
Add amplitude data here

Amplitude (19) at 10.0 is 1.0

19 1.0

Index	Time	Amplitude
13	6.84210526315789	7.0
14	7.36842105263158	6.0
15	7.89473684210526	5.0
16	8.42105263157895	4.0
17	8.94736842105263	3.0
18	9.47368421052632	2.0
19	10.0	1.0

Figure 2-26. Amplitude Data on the Waveform Display Graphical Editor



Editing parameter "waveform"

This parameter...

● has a value ○ references a symbol

□ Action Output

Description

Output a waveform

Click on “OK” to store waveform

Figure 2-27. Sample Waveform Display

Instrument Handlers

Instrument Handlers are a layer of software between Agilent TestExec SL and standard instrument drivers (see Figure 2-28). In general, Instrument Handlers are designed to be called from C/C++ code action.

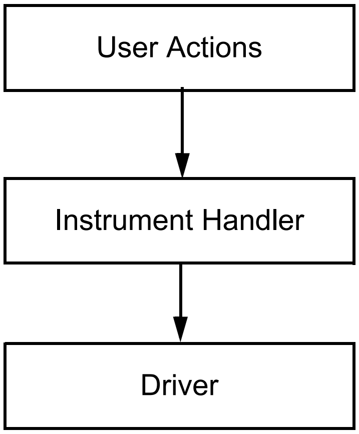


Figure 2-28. Software Layers

Instrument handlers contain functions written in C code that are organized by instrument type and function, and require parameters relevant to the function. For example, the call to set up triggering for a voltmeter is:

```
dmmConfTrigIn (dmm, trigselect, count, delay);
```

Function `dmmConfTrigIn` accepts four parameters: dmm, trigselect, count, and delay. The generic name of the function is `ConfTrigIn`, and the name of the instrument, in this case `dmm`, is added as a prefix to form the full, specific name of the function.

Note Instrument handlers are maintained constant in different TS-5000 software releases, which is not the case with the drivers. Thus, use instrument handlers for actions whenever possible.

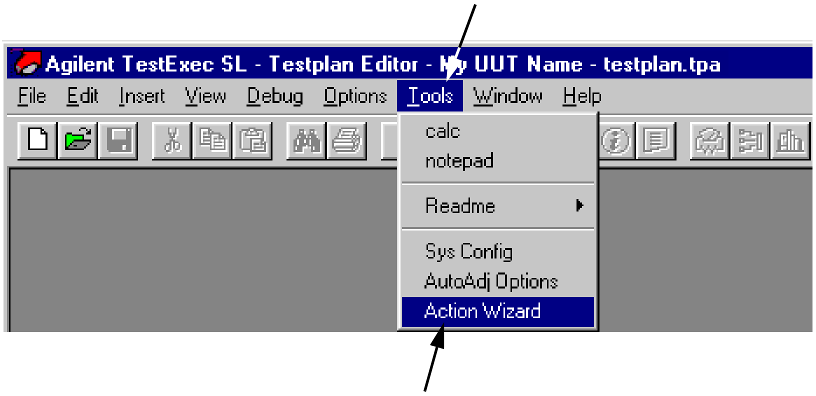
Using the Action Wizard To Develop

The Agilent TestExec SL application used in the TS-5000 System comes with a program called Action Wizard to develop actions. The Action Wizard automatically runs through the steps necessary to develop an action. The wizard also includes a help file with more detailed instructions and information on how to use the wizard.

To run the "Action Wizard", use the procedure in Figure 2-29.

Note Actions are the smallest components of a test and are used to setup and execute instruments, perform cleanup functions, and to make measurements.

1. Click on "Tools" menu


Agilent TestExec SL - Testplan Editor - My UUT Name - testplan.tpa
File Edit Insert View Debug Options Tools Window Help
calc
notepad
Readme
Sys Config
AutoAdj Options
Action Wizard

2. Click on "Action Wizard" menu item.

Figure 2-29. Selecting and Running the Action Wizard

Multithreading

Multithreading is an execution model which allows multiple threads to exist within a single process. It contains more than one unit of execution in a single process. In the context of TestExec SL, multithreading is the capability of running more than one thread. These threads share the process's resources but are able to execute independently.

A thread is a basic unit of execution. A single thread executes a series of application instructions, following a single path of logic through the application. All applications have at least one thread, but you can design your applications to use multiple threads, with each thread executing separate logic.

This capability becomes useful when a process thread stalls due to necessary data that are not yet available or when switching to another thread will yield a better result. In a single threaded program, once the main execution thread is blocked the whole application comes to a standstill. However, with multithreading, blocked programs can be moved from the main execution to a separate thread and can be concurrently executed with the main execution. This allows the application to stay responsive while executing tasks in the background. Multithreading can lead to improved performance as many threads are executed concurrently.

However, there are always disadvantages to complex application like this. Executing a multithreaded test can be more difficult than a typical test as there can be time related defects. In order for the operating system to track a large number of threads, it is going to consume processor time. If there are too many threads, then each thread may not be given enough time to execute during its time slice. In addition, each thread is scheduled for execution less frequently due to the volume and time slice committed to each thread.

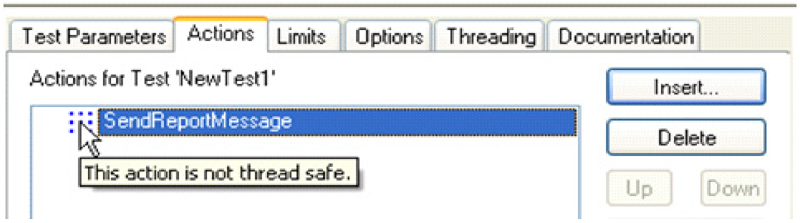
How Multithreading is working on a Testplan?

a. Creating a Thread Safe Action

Thread safe is a concept that can be applied in multithreading. A piece of code is thread-safe, if it functions correctly during simultaneous execution in multiple threads. By using thread-safe routines, the risk that one thread will interfere and modify data elements of another thread is eliminated by circumventing potential data race situations with coordinated access to shared data.

For a thread to be thread-safe, it first must behave correctly in a single-threaded environment. Furthermore, for a thread to be thread-safe, it must continue to behave correctly when accessed from multiple threads, regardless of the scheduling or interleaving of the execution threads by the runtime environment and without any additional synchronization on the part of the calling code. The effect is that operations on a thread-safe object will appear in a fixed, globally consistent order to all threads.

The thread safe declaration is introduced as an indicator. By default, an action is not thread safe and is indicated by a visual cue.



Actions for Test 'NewTest1'

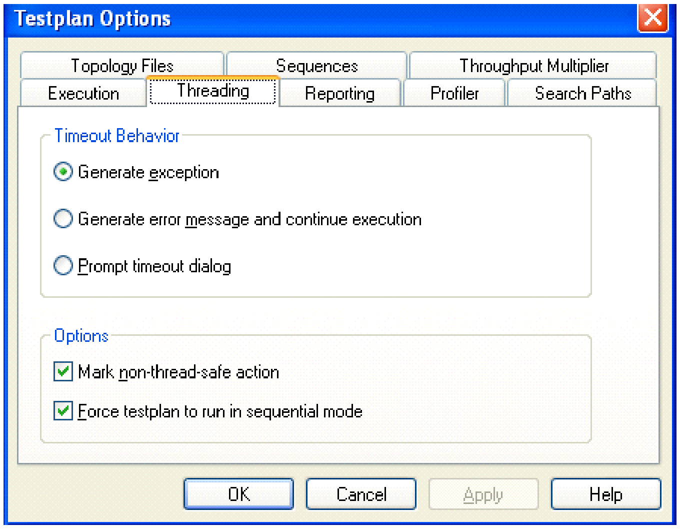
SendReportMessage

This action is not thread safe.

Figure 2-30. Symbol of thread safe declaration

If you do not wish to see a visual cue next to non-thread safe action, click Options | Testplan Options and in the Threading tab, uncheck the Mark non-thread-safe action checkbox.

Note: If check "Force testplan to run in sequential mode" checkbox, all testplans will run in sequential mode. For multithreading mode, the checkbox must be unchecked.



Timeout Behavior

● Generate exception

○ Generate error message and continue execution

○ Prompt timeout dialog

Options

☑ Mark non-thread-safe action

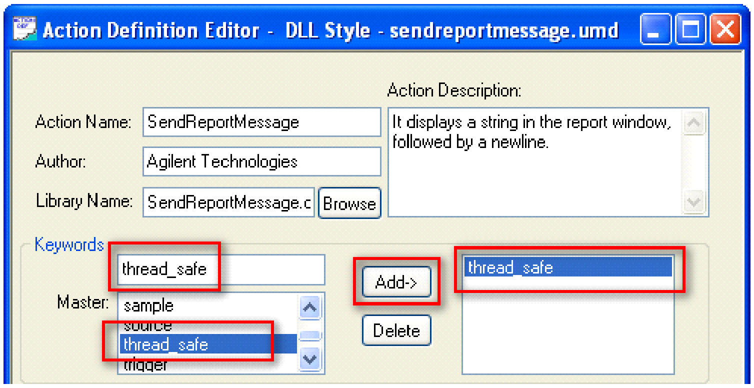
☑ Force testplan to run in sequential mode

Figure 2-31. Testplan Option for Mark non-thread-safe action

To create a thread safe action,

1. Double-click an action and the Action Definition Editor window will appear.
2. Enter "thread_safe" under Keywords.
3. Click Add.

You should now be able to see thread_safe in the right column. Take note of the highlighted areas.


Action Definition Editor - DLL Style - sendreportmessage.umd
Browse
Add->
Delete

Action Name: SendReportMessage

Author: Agilent Technologies

Library Name: SendReportMessage.c

Action Description: It displays a string in the report window, followed by a newline.

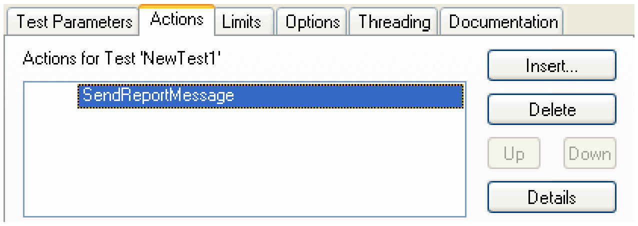
Keywords: thread_safe

Master: sample, source, thread_safe, trigger

thread_safe

Figure 2-32. Action Defination Editor

4. The visual cue will disappear once the action has been declared as thread safe.



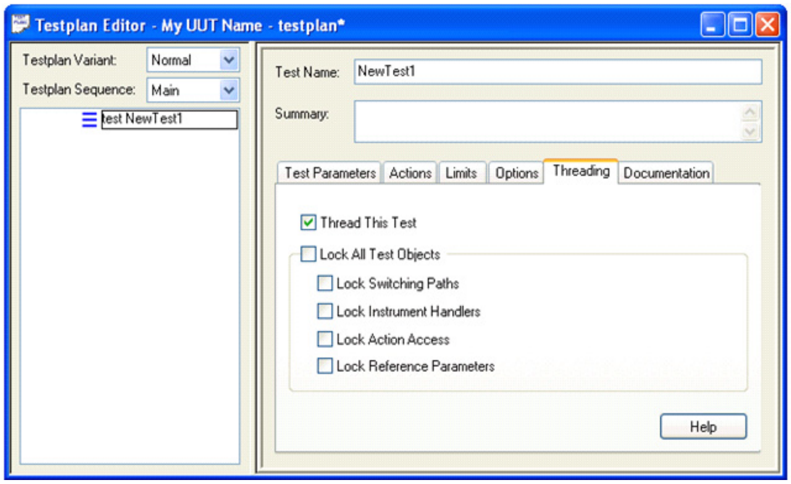
Actions for Test 'NewTest1'

SendReportMessage

Figure 2-33. Disappear Thread Safe's symbol

b. Settings for a Threaded Test

To thread a test, select the Threading tab and check the Thread This Test checkbox. A threaded visual cue will appear next to the test once the test has been threaded, as shown below.



Testplan Editor - My UUT Name - testplan*

Testplan Variant: Normal

Testplan Sequence: Main

test NewTest1

Test Name: NewTest1

Summary:

☑ Thread This Test

□ Lock All Test Objects

□ Lock Switching Paths

□ Lock Instrument Handlers

□ Lock Action Access

□ Lock Reference Parameters

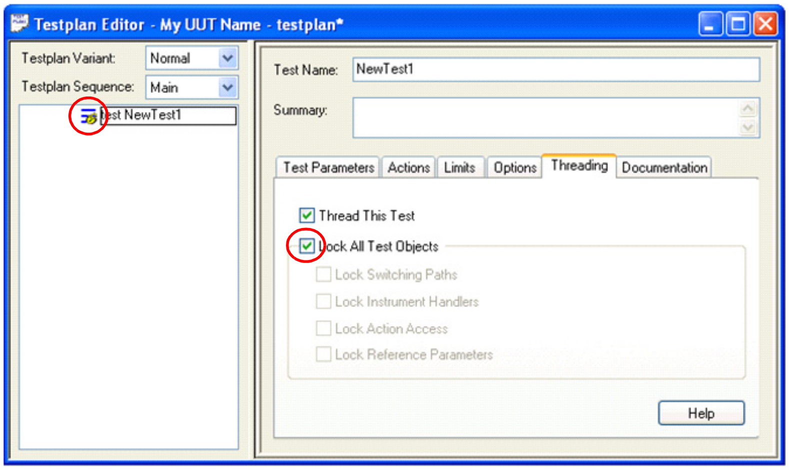
Figure 2-34. Thread This Test

c. Using Resource Locking

Resource locking is the process of protecting data in a multiprocessing environment. It means at the same point of time only one thread can access to this method of created object. Unless the lock is released after completion of the code, the next routine or iteration cannot enter the block. In the context of TestExec SL, resources that can be locked are switching paths, instrument handlers, action access, and reference parameters.

TestExec SL supports two types of resource locking which are full locking and partial locking. By implementing multithreaded tests and choosing the correct resource locking that fits your test, you can easily control the sharing of instruments and synchronize multiple UUTs. In full locking mode, TestExec SL will automatically perform locking on all objects underneath the test objects which are switching paths, instrument handlers, action access, and reference parameters. In partial locking mode, TestExec SL allows you to manually apply one or more of the above mentioned locks to the object(s).

Resource locking can be applied to each of the containing objects under the test object. Tests can be manually locked by selecting the Threading tab. Click the Lock All Test Objects checkbox for a full lock or click an individual lock to choose your preferred lock. The visual cues for full locking and partial locking slightly differ in color.



Testplan Editor - My UUT Name - testplan*

Testplan Variant: Normal

Testplan Sequence: Main

est NewTest1

Test Name: NewTest1

Summary:

☑ Thread This Test

☑ Lock All Test Objects

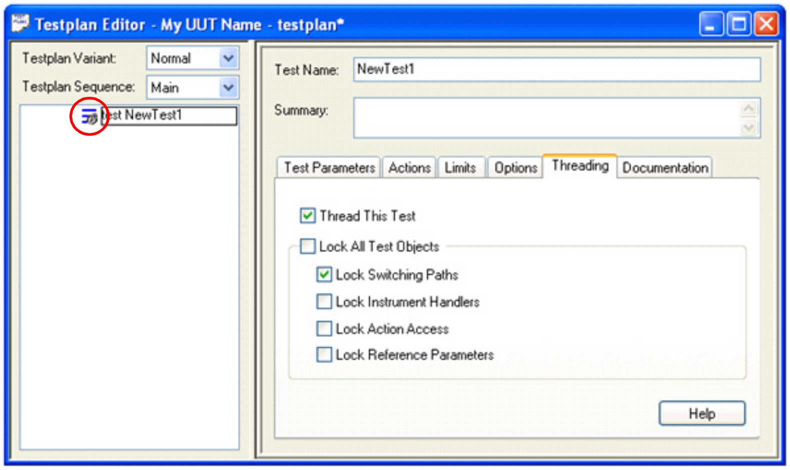
☐ Lock Switching Paths

☐ Lock Instrument Handlers

☐ Lock Action Access

☐ Lock Reference Parameters

Figure 2-35. Lock All Test Objects



Testplan Editor - My UUT Name - testplan*

Testplan Variant: Normal

Testplan Sequence: Main

est NewTest1

Test Name: NewTest1

Summary:

☑ Thread This Test

☐ Lock All Test Objects

☑ Lock Switching Paths

☐ Lock Instrument Handlers

☐ Lock Action Access

☐ Lock Reference Parameters

Figure 2-36. Lock Test Objects partially

d. Using the Wait Thread Statement

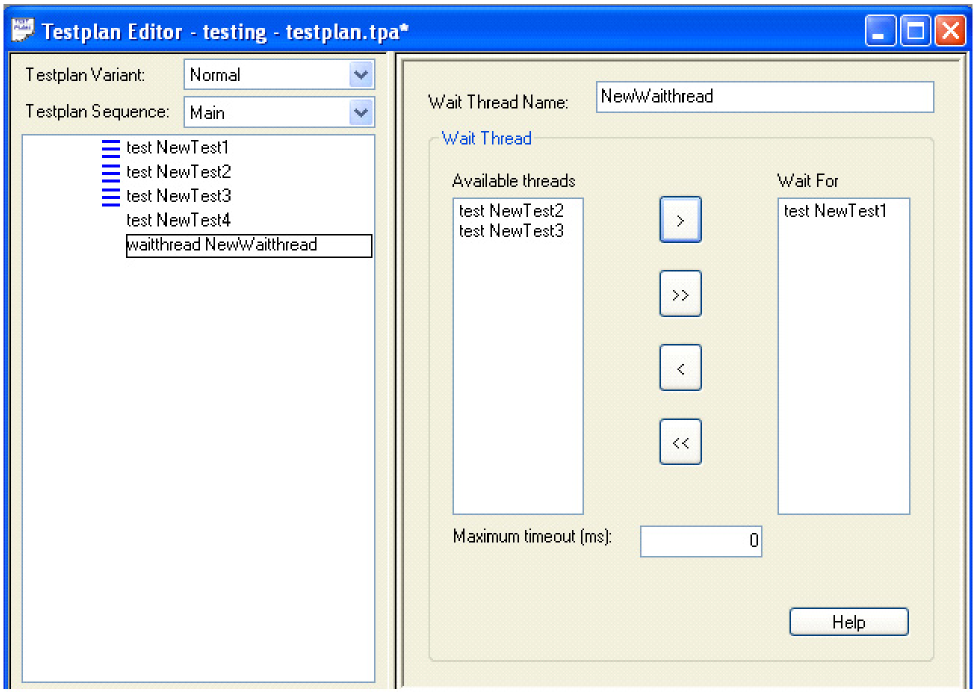
One way to think of the wait thread statement is to imagine an item of data such as an integer variable as if it were a field in a database. If you do not have some locking mechanism in the database you stand a chance of corruption to the data.

Thus, one user might retrieve the data and perform a calculation and write back the data. If in the meantime someone else has retrieved the data, performed the calculation and written it back, the second user's calculations will be lost when the first person writes back to the database. In the way that a database has to handle updates at unpredictable times, so a multithreaded program has to cater for this possibility. TestExec SL has a wait thread statement especially to resolve matters like this.

The Wait Thread statement allows you to block remaining statements from execution prior to completion of threaded tests statement in the Wait For column.

To insert a Wait Thread statement, click Insert | Wait Thread in TestExec SLs menu bar. The maximum timeout value is used to specify the maximum duration a Wait Thread statement would need to wait for the threads. A timeout of 0 ms means the program will wait forever until all the waited threaded tests have completed execution.

The below testplan shows that NewWaitthread needs to wait for NewTest1 to complete execution.



Testplan Editor - testing - testplan.tpa*

Testplan Variant: Normal

Testplan Sequence: Main

- test NewTest1
- test NewTest2
- test NewTest3
- test NewTest4
- waitthread NewWaitthread

Wait Thread Name: NewWaitthread

Wait Thread

Available threads:
test NewTest2
test NewTest3

Wait For:
test NewTest1

Maximum timeout (ms): 0

Figure 2-37. Wait Thread statment

Using Wait Thread Statement in Loop, For...In and For...To...Step Statements

It is recommended to insert Wait Thread statement right before Next statement in Loop, For...In and For...To...Step statements if threaded tests exists within these statements. The Wait Thread statement needs to wait for all threaded tests within the Loop and For Loop statements, otherwise, the results might be unexpected.

Using Wait Thread Statement for Throughput Multiplier Threaded Test Statement

It is also recommended to insert Wait Thread statement after a throughput multiplier threaded test statement to wait for all the threads running for all the UUTs to complete execution before proceeding on with next statement.

Using Wait Thread Statement in Testgroup Statement

A Wait Thread statement should be inserted right before end testgroup statement if threaded tests exist within the testgroup statement. The Wait Thread statement needs to wait for all threaded tests within the testgroup, otherwise, the results might be unexpected.

Using Wait Thread Statement in Sequence Library

A Wait Thread statement should be inserted as the last statement of a sequence library (local or external) if threaded tests exist in the sequence library. The Wait Thread statement is to ensure all the threaded tests in the sequence library have completed their execution before returning to Main sequence. If the Wait Thread statement is not inserted, threads in the sequence library might be still running upon returning to Main sequence.

e. Using Timeout

A specified period of time that will be allowed to elapse in a system before a the next specified event is to take place. TestExec SL allows you to specify a timeout value as well as the behavior of the timeout. The timeout value is set in the Wait Thread statement while it's behavior is set in Testplan Options.

In Options | Testplan Options, a new Threading tab has been added to allow users to specify the behavior of the timeout. There are three options you can choose from:

- Generate exception
- Generate error message on report window and continue execute
- Prompt timeout dialog

Threaded objects are killed when timeout is exceeded. This timeout feature is the simplest method of preventing of deadlocks.

Opening System Configuration Editor

1. Start System Configuration Editor from this icon on the PC desktop:

   You can also run System Configuration Editor by clicking:

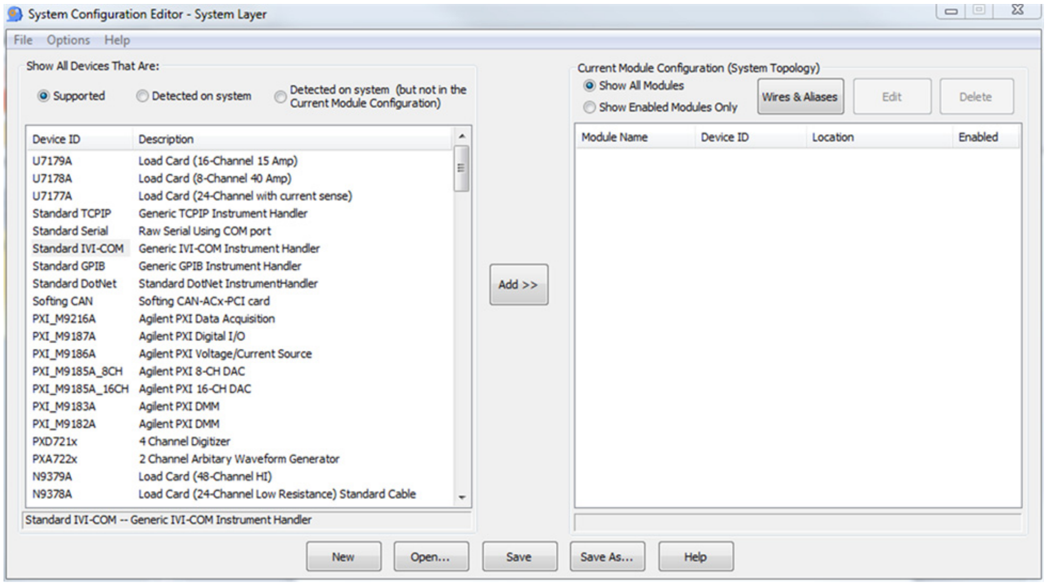
   Start | All Programs | Agilent TS-5000 System Software 7.1.2

Adding an IVI-COM Module in System Configuration Editor

Note This feature will only be available for TS-5000 7.1 Service Pack 3 or newer.

1. When System Configuration Editor opens, all devices that are supported by current configuration will be shown by default.

   To add a Standard IVI-COM module, double click its selection available under the Device ID column as show below.



Show All Devices That Are:

● Supported ○ Detected on system ○ Detected on system (but not in the Current Module Configuration)

Device ID	Description
U7179A	Load Card (16-Channel 15 Amp)
U7178A	Load Card (8-Channel 40 Amp)
U7177A	Load Card (24-Channel with current sense)
Standard TCPIP	Generic TCPIP Instrument Handler
Standard Serial	Raw Serial Using COM port
Standard IVI-COM	Generic IVI-COM Instrument Handler
Standard GPIB	Generic GPIB Instrument Handler
Standard DotNet	Standard DotNet InstrumentHandler
Softing CAN	Softing CAN-ACx-PCI card
PXI_M9216A	Agilent PXI Data Acquisition
PXI_M9187A	Agilent PXI Digital I/O
PXI_M9186A	Agilent PXI Voltage/Current Source
PXI_M9185A_8CH	Agilent PXI 8-CH DAC
PXI_M9185A_16CH	Agilent PXI 16-CH DAC
PXI_M9183A	Agilent PXI DMM
PXI_M9182A	Agilent PXI DMM
PXD721x	4 Channel Digitizer
PXA722x	2 Channel Arbitary Waveform Generator
N9379A	Load Card (48-Channel HI)
N9378A	Load Card (24-Channel Low Resistance) Standard Cable

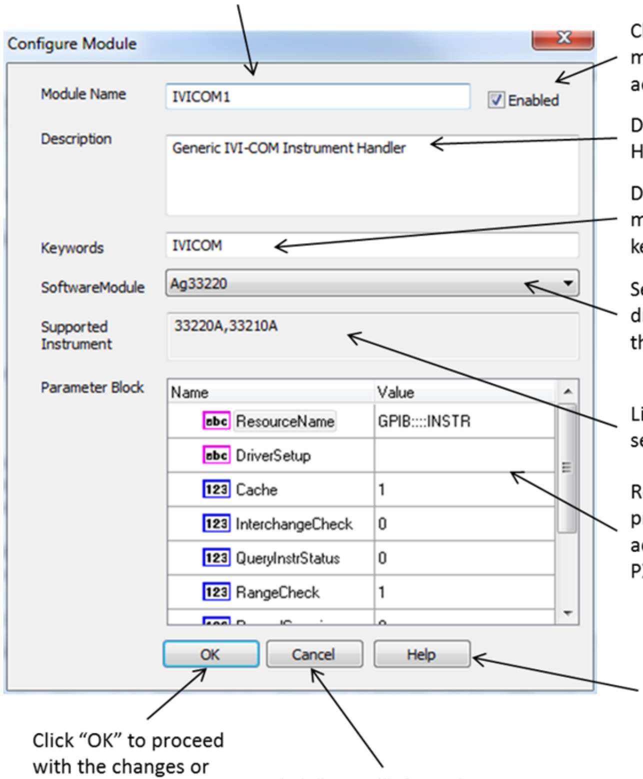
Standard IVI-COM -- Generic IVI-COM Instrument Handler

Current Module Configuration (System Topology)

● Show All Modules
○ Show Enabled Modules Only

Module Name	Device ID	Location	Enabled

2. The window Configure Module will pop-up.



Configure Module

Module Name: IVICOM1 ☑ Enabled

Description: Generic IVI-COM Instrument Handler

Keywords: IVICOM

SoftwareModule: Ag33220

Supported Instrument: 33220A,33210A

Parameter Block

Name	Value
abc ResourceName	GPIB::::INSTR
abc DriverSetup	
123 Cache	1
123 InterchangeCheck	0
123 QueryInstrStatus	0
123 RangeCheck	1

Name of the IVI module.
This is the name referenced from a testplan

Check this box to enable a module/instrument, and to make it accessible from testplan.

Default description of IVI-COM Instrument Handler. Change if needed.

Descriptive keywords to aid in finding the module in the system.ust file, separate keywords by comma.

Select the IVI software module from this drop down list. Only IVI drivers installed in the system are available for selection.

List of instruments supported by the selection of Software Module

ResourceName in Parameter block must be provided. Copy and paste instrument VISA address into this field. It can be GPIB, TCPIP, PXI, etc. Example: GPIB0::6::INSTR

Click “Help” to select this help topic.

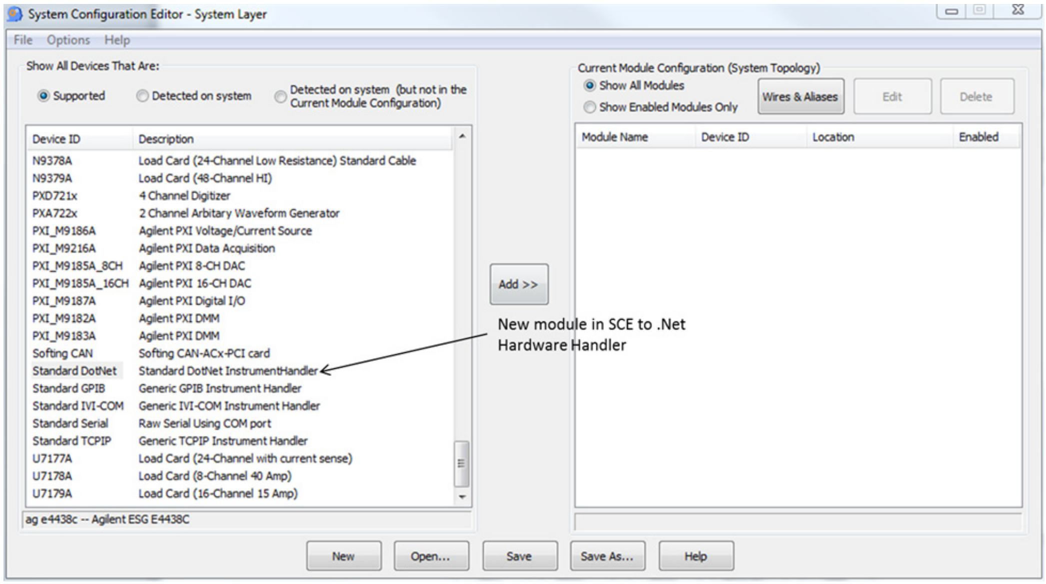
Click “OK” to proceed with the changes or adding of this module/instrument

Click “Cancel” if you do not wish to continue with the changes or adding of this module/instrument.

Adding a .Net Module in System Configuration Editor

1. When System Configuration Editor opens, all devices that are supported by current configuration will be shown by default.

   To add a Standard DotNet module, double click its selection available under the Device ID column as show below.



Show All Devices That Are:

● Supported ○ Detected on system ○ Detected on system (but not in the Current Module Configuration)

Device ID	Description
N9378A	Load Card (24-Channel Low Resistance) Standard Cable
N9379A	Load Card (48-Channel HI)
PXD721x	4 Channel Digitizer
PXA722x	2 Channel Arbitary Waveform Generator
PXI_M9186A	Agilent PXI Voltage/Current Source
PXI_M9216A	Agilent PXI Data Acquisition
PXI_M9185A_8CH	Agilent PXI 8-CH DAC
PXI_M9185A_16CH	Agilent PXI 16-CH DAC
PXI_M9187A	Agilent PXI Digital I/O
PXI_M9182A	Agilent PXI DMM
PXI_M9183A	Agilent PXI DMM
Softing CAN	Softing CAN-ACx-PCI card
Standard DotNet	Standard DotNet InstrumentHandler
Standard GPIB	Generic GPIB Instrument Handler
Standard IVI-COM	Generic IVI-COM Instrument Handler
Standard Serial	Raw Serial Using COM port
Standard TCPIP	Generic TCPIP Instrument Handler
U7177A	Load Card (24-Channel with current sense)
U7178A	Load Card (8-Channel 40 Amp)
U7179A	Load Card (16-Channel 15 Amp)

ag e4438c -- Agilent ESG E4438C

New module in SCE to .Net Hardware Handler

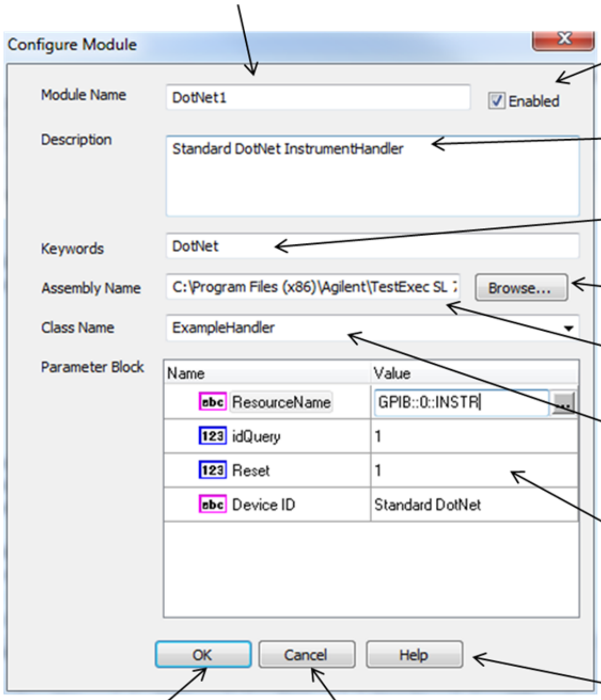
Current Module Configuration (System Topology)

● Show All Modules
○ Show Enabled Modules Only

Module Name	Device ID	Location	Enabled

2. The window Configure Module will pop-up.

Name of the .NET module.
This is the name referenced from a testplan



Configure Module

Module Name: DotNet1 ☑ Enabled

Description: Standard DotNet InstrumentHandler

Keywords: DotNet

Assembly Name: C:\Program Files (x86)\Agilent\TestExec SL ː

Class Name: ExampleHandler

Parameter Block

Name	Value
abc ResourceName	GPIB::0::INSTR
123 idQuery	1
123 Reset	1
abc Device ID	Standard DotNet

Check this box to enable a module/instrument, and to make it accessible from testplan.

Default description of .NET Instrument Handler. Change if needed.

Descriptive keywords to aid in finding the module in the system.ust file, separate keywords by comma.

Browse to select .NET Hardware Handler

Full path of the selected .NET hardware handler file.

Class Name base on the .NET hardware handler.

ResourceName in Parameter block must be provided. Copy and paste instrument VISA address into this field. It can be GPIB, TCPIP, PXI, etc. Example: GPIB0::6::INSTR

Click "Help" to select this help topic.

Click "OK" to proceed with the changes or adding of this module/instrument

Click "Cancel" if you do not wish to continue with the changes or adding of this module/instrument.

Chapter 3
Using the DMM and ARB

Chapter Contents

This chapter describes how to trigger the DMM and how to use the ARB. The chapter is separated as follows:

Triggered Voltage Measurement

The Agilent E1411B Multimeter (DMM) can be triggered by a signal on the VXI Mainframe’s backplane. This signal uses one of the TTL (i.e., TTL0 to TTL7) trigger lines.

How to Trigger the DMM

Testplan ‘`dmmtrig.tpa`’ shows how the DMM is triggered using the Agilent E6174A Arbitrary Waveform Generator (Arb). The Arb is also used to generate the signal measured by the DMM. The testplan also demonstrates how to use the Agilent E6171B Measurement Control Module’s (MCM) analog comparator. The testplan is in the following directory:

`C:\Program Files\Agilent\TS-5000 System Software\testplan\examples`

The Arb generates a ±5V 10 Hz square wave as the trigger signal. This method uses the analog comparator of the MCM to send the trigger to the DMM using the VXI Mainframe's backplane (see connections in Figure 3-1 and Figure 3-2).

The testplan configures the MCM analog comparator to trigger at 0 volts. Setting the “trigedge” parameter in the “viConfCompare” action to “0” (positive edge detection), causes negative going backplane trigger when the input signal makes a low to high (positive going) transition through the trigger threshold. Setting the “trigedge” parameter in the “viConfCompare” action set to “1” (negative edge detection), causes triggers when the input signal makes a high to low (negative going) transition.

The following is a review of the testplan

Test/Action Name	Description
test: Dmm Init	Initializes the Digital Multimeter.
dmmConfCal	Disables autozero and set the power line frequency.
dmmIsSet	Waits for setup to complete.
test: Arb 1 pgm	Configures Arb and outputs a 10 Hz +5V Square wave.
arbReset	Reset the Arbitrary Waveform Generator (Arb) to its turn-on state.
arbConfOutControls	Configures the Arb’s output circuitry.
Arb_Dl_Std_Waveform	Download a 10 Hz square wave into the Arb.
arbInitiate	Start outputting the waveform.
Arb_Select_Wave_By_Name	Select waveform downloaded above.
test: VI Config0 (+5)	Configures MCM’s analog comparator to trigger at 0V and a low to high (positive transition) trigger.
viConfCompare	Configures the comparator. The “trigedge” parameter value (0) causes a backplane trigger when the input signal makes a low to high (positive going) transition trough trigger threshold.
viIsSet	Waits for setup to complete.
Test: Meas V - Trigger	Trigger DMM and measure trigger voltage.

Test/Action Name	Description
Switching	Make the necessary switching to setup the Arb to DMM measurements and trigger signal paths.
dmmMeasureTrigVoltage	Configures DMM for a measurement using the backplane TTL trigger line 0.
dmmGetResults	Make the measurement and return the reading.
test: VI Config1 (-5)	Configures MCM’s analog comparator to trigger at 0V and a high to low (negative transition) trigger.
viConfCompare	Configures the comparator. The “trigedge” parameter value (0) causes a negative going backplane trigger when the input signal makes a high to low (negative going) transition trough trigger threshold.
viIsSet	Waits for setup to complete.
test Meas V - Expanded	Trigger DMM and measure trigger voltage.
Switching	Make the necessary switching to setup the Arb to DMM measurements and trigger signal paths.
DelayMillisecond	Wait for switching to complete.
dmmInitiate	Initiates a reading cycle.
dmmGetResults	Make the measurement and return the reading.

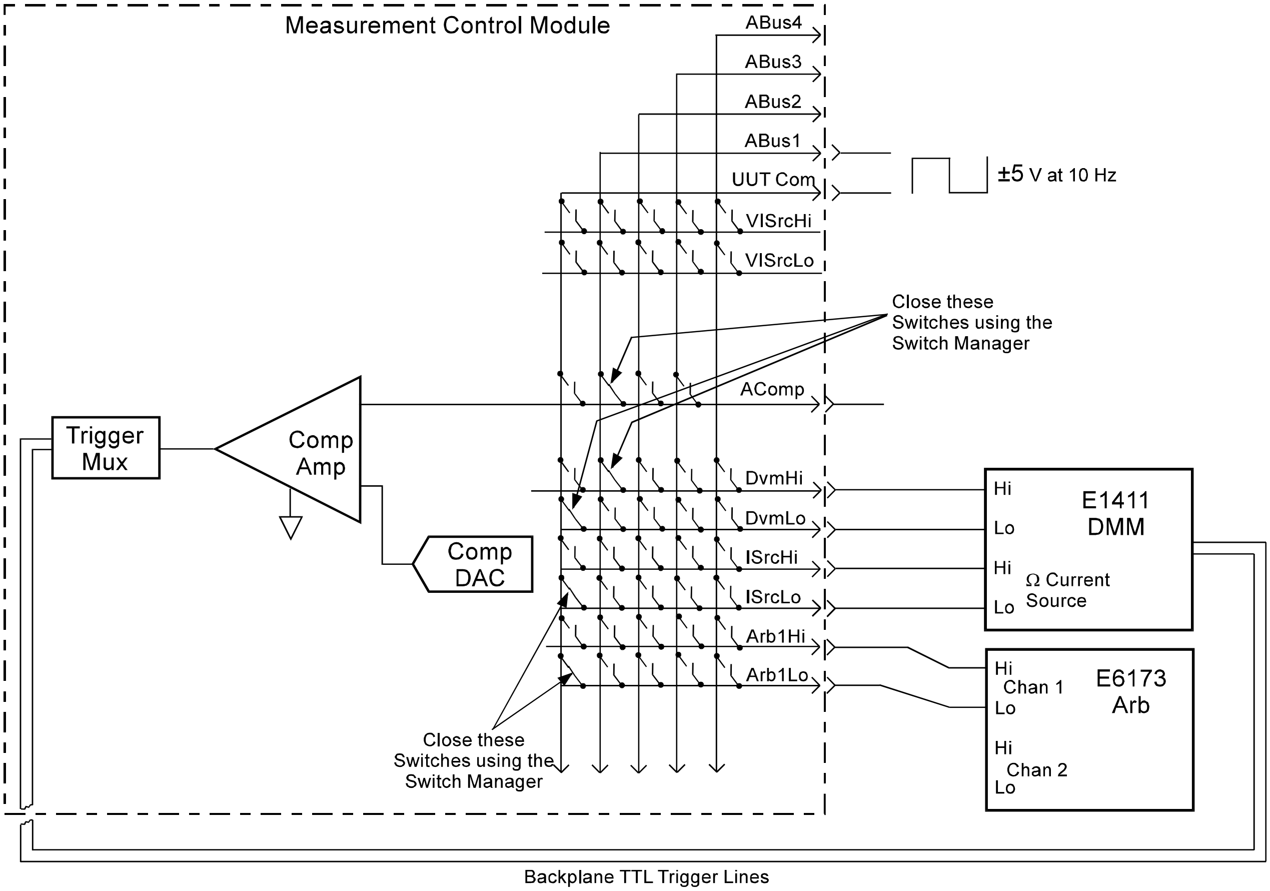
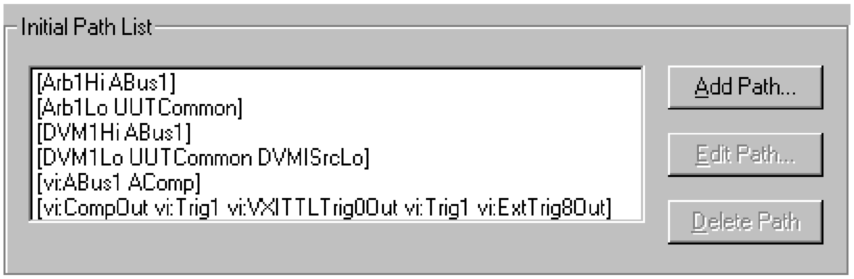


Figure 3-1. Trigger Connections



```
[Arb1Hi ABus1]
[Arb1Lo UUTCommon]
[DVM1Hi ABus1]
[DVM1Lo UUTCommon DVMISrcLo]
[vi:ABus1 AComp]
[vi:CompOut vi:Trig1 vi:VXITTLTrig0Out vi:Trig1 vi:ExtTrig8Out]
```

Figure 3-2. Switch Paths Listing

Using the Agilent E6173A Arbitrary Waveform Generator

The following describes the Agilent E6173A Arbitrary Waveform generator and how to use it.

Arbitrary Waveform Generator Description

The Agilent E6173A Arbitrary Waveform Generator (Arb), formerly the Agilent Z2471A, is a register-based, two channel signal generator whose channels are isolated from each other and from ground. The factory default configuration connects first channel to the Agilent E6171 Measurement Control Module (MCM).

You can amplify channel one's output using the MCM's V/I amplifier/current source. You can connect channel two directly to your Unit Under Test (UUT) or use one of the unassigned MCM inputs. The Arb can also switch to any UUT pin by switching to Aux ports, if using the default system pins.

The Arb can output standard waveforms, such as sine, square, triangle, knock, or crank of a specified frequency, amplitude, and offset, and custom arbitrary waveforms defined by the user.

How the Arb Generates Waveforms

The Arb generates waveforms using pre-defined or user defined segment and sequence data. Figure 3-3 shows a typical example of a waveform.

The segments of a waveform are the voltage levels of each point on the waveform. The sequence consists of a group of pre-defined or user defined number of segments. To generate a waveform, the Arb outputs the sequence, consisting of the defined number of segments, using a pre-defined time.

The number of segments output (i.e., how often they occur) and the time of the Arb’s internal clock (i.e., segment duration time) determines the time or frequency of the waveform to be output. The Arb’s internal clock time is 2 µS. Either single or multiple clock cycles can be used to set the appropriate frequency. Selecting the clock cycles is usually known as selecting the “dwell” count.

For example, a waveform consisting of 250 segments using two 2 uS clock cycles (or 2 dwell counts), outputs a 1 kHz waveform, shown as follows:

$$250 * (0.000002 * 2) = 250 * 0.000004 = 0.001 \text{ S} = 1 \text{ kHz}$$

In addition to the 2 µS clock, the Arb also has a 100 nS counter that can be used to add time in 100 nS steps to the 2 µS clock cycles. This is used to generate frequency accurate waveforms.The counter is normally set using the “clock adjust” function. Note: Segment time = (Dwell + 1) * (2 µS + (clock adjust * 0.1 µS)).

Both the segment data and sequence data is *downloaded* into the Arb’s memory using the standard actions supplied with the system. These actions can be used to download both standard (pre-defined) waveforms and user defined waveforms.

Note For more information about the waveform timing, refer to the *Agilent E6173A Arbitrary Waveform Generator User's Manual*.

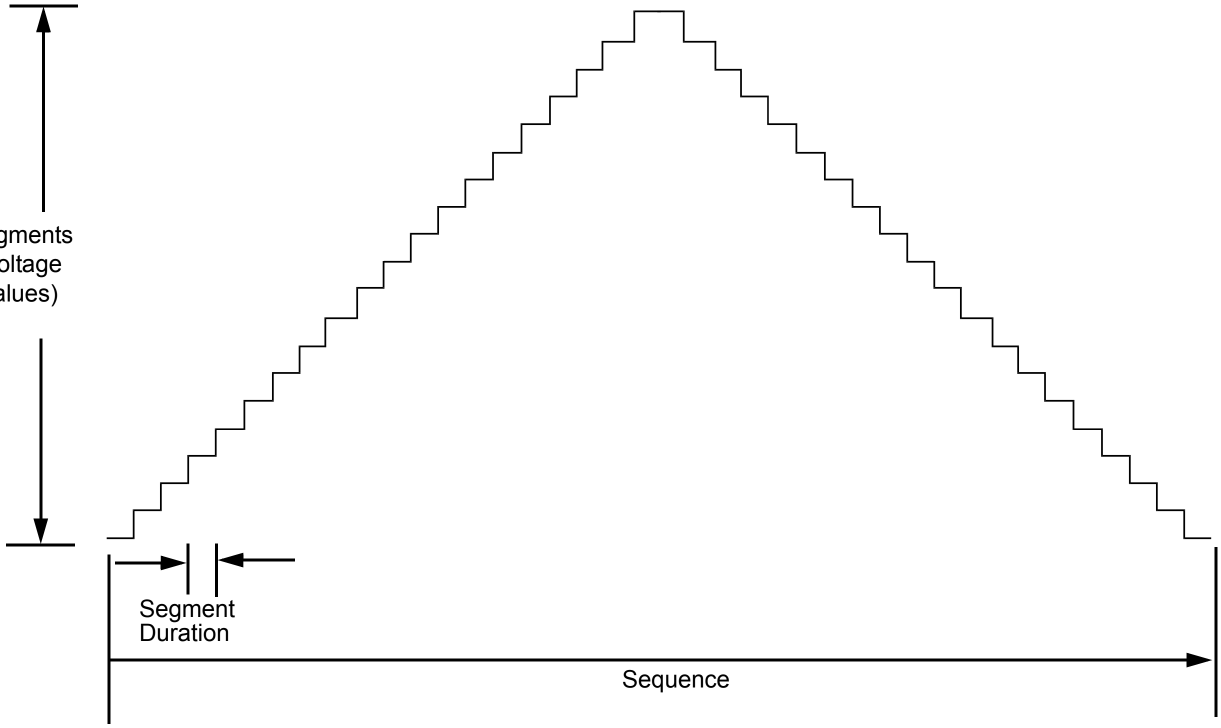


Figure 3-3. Typical Waveform using Segments and Sequence

Generating Arb Waveforms

There are several methods to generate Arb waveforms using the standard actions. You can generate standard waveforms, such as sine, pulse, and triangle waveforms, or you can create custom waveforms.

Many actions have been developed for high throughput operations. These actions are generally named "Arb_Dl_xxx" where the "Dl" indicates the actions with the high throughput capability (see "In all testplans, the Arb connects to the Analog Bus ABus1. To view the output, connect an oscilloscope to the ABus 1 connector at the front of the system. Be sure to make the necessary connection to the Mass Interconnect/Express Connect so the interlock is shorted in order to use the system (see the appropriate documentation). Instead of an oscilloscope, you can also use an VXI Technology E1563A Digitizer to read the waveforms and display them using the appropriate actions." below).

Note DO NOT use the "Arb_Dl_xxx" actions with the older "ArbSetSegment" and/or "ArbSetSequence" actions on the same Arb channel.

Note When selecting different downloaded waveforms, usually when there are major frequency differences, the first few cycles of a newly selected waveform may appear to be distorted. The reason is that the previous waveform must complete, but the new “clock adjust” is already in effect, which may change the timing of the last cycles of the previous waveform.

The following explains the methods used. For each method, there is a testplan available it. The testplans are located in the following directory:

```
C:\Program Files\Agilent\TS-5000 System
Software\testplan\examples
```

In all testplans, the Arb connects to the Analog Bus ABus1. To view the output, connect an oscilloscope to the ABus 1 connector at the front of the system. Be sure to make the necessary connection to the Mass Interconnect/Express Connect so the interlock is shorted in order to use the system (see the appropriate documentation). Instead of an oscilloscope, you can also use an VXI Technology E1563A Digitizer to read the waveforms and display them using the appropriate actions.

Downloading and Executing Waveforms Using the “Arb_Dl_xxx” Actions

Using the “Arb_Dl_xxx” actions to download standard or user defined waveforms gives you the highest throughput capability. Using these actions, the waveforms are downloaded the first time the testplan is run. The waveforms remain in memory and can then be selected any time during testplan execution. The only time new waveforms are downloaded is when a testplan is modified.

Each downloaded waveform needs a unique name to keep track of the waveform. This name can then be used to call a particular waveform in any part of the testplan as many times as needed. Since both waveform data and other parameters, such as output voltage are also included with a name, the waveform will be output the same way every time it is selected.

Note

When initializing the Arb for output using the “arbInitiate” action, the Arb outputs the last downloaded waveform, if “Arb_Select_Wave_By_Name” is not called before “arbInitiate”. Use the waveform names in the ‘waveform_name’ parameters of the of “Arb_Dl_xxx” actions to select them.

A typical sequence to generate a waveform is as follows:

1. Setup Arb using the “arbConfOutControls” action.
2. Send the setup data to the Arb using the “arbSet” action.
3. Download waveform data using the “Arb_Dl_xxx” action; use a unique waveform name in the ‘waveform_name’ parameter.
4. Initialize the Arb to output the waveform using the “arbInitiate” action.
5. Select the downloaded waveform using the “Arb_Select_Wave_By_Name” action using the waveform name in the “Arb_Dl_xxx” action.

Generating Standard Waveforms

The following shows how to download standard waveforms for output using the standard actions. For high throughput capability, use the “Arb_Dl_xxx” actions.

Downloading a Standard Waveform

Use the “Arb_Dl_Std_Waveform” action to download the waveform and the “Arb_Select_Wave_By_Name” action to select the waveform. In this case, the same waveform name is referenced by both actions to select the waveform for output.

Using this method, a waveform can be selected on-the-fly to allow fast waveform access. This is because the waveform has previously been generated and stored in memory. All that is needed is to select the waveform to output it.

An example to download and then select a waveform is as follows. In the example, a waveform is generated and stored using a particular name (i.e., “Arbwave1”). Once this waveform is generated, it can then be selected in any part of the testplan.

> Generate the waveform using Arb_Dl_Std_Waveform with the waveform name = “Arbwave1” (parameter: ‘waveform_name’ = Arbwave1).

to select the waveform, set:

> waveform name of an Arb_Select_Wave_By_Name action = “Arbwave1” (parameter: ‘waveform_name’ = Arbwave1) and then execute that action.

Note The waveform names are NOT case sensitive.

Using the above actions allow you to download and store several different waveforms into the Arb that can be selected in different parts of the testplan. One way to do this is to download several different waveforms with different names at the beginning of the testplan and then select the different waveforms throughout the testplan.

An example to use different downloaded waveforms in different parts of a testplan is as follows:

```
    Test1:    Arb_Dl_Std_Waveform: ‘waveform_name’ = sine
              Arb_Dl_Std_Waveform: ‘waveform_name’ = square
              Arb_Dl_Std_Waveform: ‘waveform_name’ = triangle
                   ●
                   ●
    Test2:    Arb_Select_Wave_By_Name: ‘waveform_name’ = triangle
                   ●
                   ●
    Test3:    Arb_Select_Wave_By_Name: ‘waveform_name’ = sine
                   ●
```

- Test4: Arb_Select_Wave_By_Name: ‘waveform_name’ = square

Testplan ‘arbdown_name.tpa’ illustrates how to download a 1 kHz sine wave, 2 kHz square wave, and 4 kHz triangle wave. A review of the testplan is below.

Test Group/Test/Action Name	Description
testgroup Arb download waveforms	Downloads standard waveforms.
Switching	Connect Arb channel 1 output to ABus1 and UUT Common.
test setup Arb	Setup arb output circuitry.
arbConfOutControls	Configures Arb’s output circuitry.
arbSet	Sends setup data to Arb.
test download sine wave	Downloads the standard sine wave to the Arb.
Arb_Dl_Std_Waveform	Downloads a sine wave into the Arb. The ‘function’ parameter is set to “0” to download a sine wave. The ‘waveform_name’ parameter is set to “sine” to give the downloaded waveform that name. The ‘frequency’ parameter is set for a 1 kHz waveform. The ‘Vpeak’ parameter is set for a 5 V peak waveform. The ‘waveform_name’ parameter is set to
test download square wave	Downloads the standard pulse wave to the Arb.
Arb_Dl_Std_Waveform	Downloads a square wave into the Arb. The ‘function’ parameter is set to “1” to download a pulse. The ‘waveform_name’ parameter is set to “square” to give the downloaded waveform that name. The ‘duty_cycle’ parameter is set to “0.5” so the pulse function downloads a square wave. The ‘frequency’ parameter is set for a 2 kHz waveform. The ‘Vpeak’ parameter is set for a 5 V peak waveform.
test download triangle wave	Downloads the standard triangle wave to the Arb.
Arb_Dl_Std_Waveform	Downloads a triangle wave into the Arb. The ‘function’ parameter is set to “2” to download a triangle wave. The ‘waveform_name’ parameter is set to “triangle” to give the downloaded waveform that name. The ‘frequency’ parameter is set for a 4 kHz waveform. The ‘Vpeak’ parameter is set for a 5 V peak waveform.
test output waveforms	This test executes the downloaded waveforms.
arbInitiate	Initialize the Arb to output the waveform. Start outputting the waveform. The Arb at this time outputs the last downloaded waveform (i.e., “triangle”).
DialogOkay	This optional action is used here to view the current waveform (i.e., triangle wave).
Arb_Select_Wave_By_Name	Selects and outputs the waveform using the name entered in the ‘waveform_name’ parameter (i.e., “sine”). Since this is the name of the ‘waveform_name’ parameter of the “Arb_Dl_Std_Waveform” action in “test download sine wave”, a sine wave is output by the Arb.
DialogOkay	This optional action is used here to view the current waveform (i.e., sine wave).

Test Group/Test/Action Name	Description
Arb_Select_Wave_By_Name	Selects and outputs the waveform using the name entered in the ‘waveform_name’ parameter (i.e., “square”). Since this is the name of the ‘waveform_name’ parameter of the “Arb_Dl_Std_Waveform” action in “test download sine wave”, a square wave is output by the Arb.
DialogOkay	This optional action is used here to view the current waveform (i.e., square wave).
Arb_Select_Wave_By_Name	Selects and outputs the waveform using the name entered in the ‘waveform_name’ parameter (i.e., “triangle”). Since this is the name of the ‘waveform_name’ parameter of the “Arb_Dl_Std_Waveform” action in “test download sine wave”, a triangle wave is output by the Arb.
DialogOkay	This optional action is used here to view the current waveform (i.e., triangle wave).

Outputting a Standard Waveform Immediately

The “Arb_Output_Std_Waveform” action can also be used to generate and output standard waveforms. This action enables the Arb to output a sine, pulse, or triangle wave. Unlike the “Arb_Dl_Waveform” action, this action does not require “arbIntiate” to enable it to output a waveform, however, it also has the high throughput capability of the “Arb_Dl_Waveform” action. “Arb_Output_Std_Waveform” can be used in any part of the testplan.

Testplan ‘`arboutputstd.tpa`’ illustrates how to output a 1 kHz sine wave, 2 kHz square wave, and a 4 kHz triangle wave. A review of the testplan is below.

Test Group/Test/Action Name	Description
testgroup Arb output std waveforms	Outputs standard waveforms.
Switching	Connect Arb channel 1 output to ABus1 and UUT Common.
test output sine	Output sine wave.
Arb_Output_Std_Waveform	Immediately outputs a sine wave; ‘function’ parameter is set to “0”.
DialogOkay	This optional action is used here to view the current waveform (i.e., sine wave).
test output square	Output square wave.
Arb_Output_Std_Waveform	Immediately outputs a square wave; ‘function’ parameter is set to “1” (pule) and ‘duty_cycle’ parameter is set to “0.5” for the “pulse” function to output the square wave.
DialogOkay	This optional action is used here to view the current waveform (i.e., square wave).
test output triangle	Output triangle wave.
Arb_Output_Std_Waveform	Immediately outputs a triangle wave; ‘function’ parameter is set to “2”.
DialogOkay	This optional action is used here to view the current waveform (i.e., triangle wave).

Downloading a Standard Phase Shifted or Burst Modulated Waveform

Use the “Arb_Dl_Ext_Waveform” action for phase shifted and burst modulated waveforms. This action has the following functionality:

1. Square wave generation
2. Positive and negative ramp generation
3. Positive and negative pulse generation
4. Burst modulation
5. Phase shift
6. Pulse width for positive and negative pulses, and burst pulses.

To correctly generate a waveform, it is important to understand how to set some of this action’s different parameters. These are explained as follows:

frequency - Specifies the output frequency if the waveform is NOT a “burst” waveform (‘F_burst’ parameter set to “0”). If “burst” is used, this is the frequency of the burst wave, as shown in Figure 3-4. (Note: “burst” can not be used in the positive or negative pulse functions; i.e., functions 5 or 6, respectively).

Tpulse_width - Sets the pulse width of the positive or negative pulses function (for functions 5 and 6, respectively) or sets the width, in seconds, of the burst wave, as shown in Figure 3-4. If used with “burst”, the pulse width must be less than 100 mS.

F_burst - If set to “0”, no “burst” is generated. Otherwise, this sets the waveform frequency that is to be modulated by the burst wave (the burst wave frequency is set by the ‘frequency’ parameter), as shown in Figure 3-4.

phase_shift - Specifies the amount of phase shift in a waveform in degrees, relative to a non-phase shifted version of itself.

force_seg_count - If set to “0”, the frequency accuracy is generated using the 2 μS clock cycles plus any number of 100 nS clock cycles needed to make the frequency accurate. This is illustrated in the following formula:

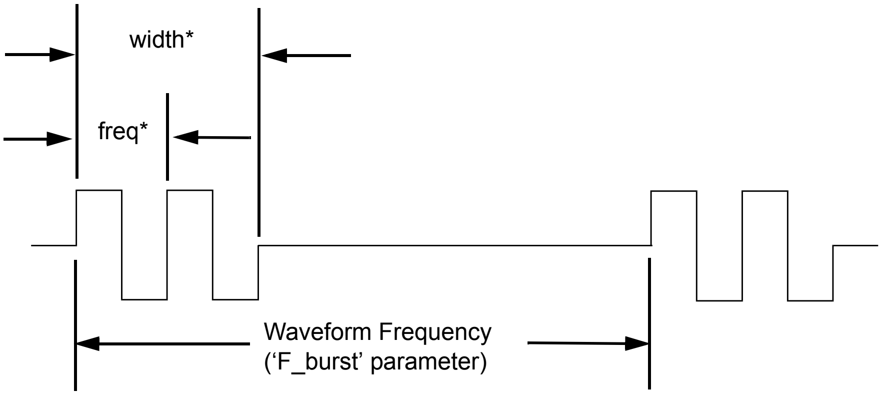
segments = (1 / frequency) / (2 μS + (100 nS * clockadj))

where ‘clockadj’ is used to add 1 or more 100 nS ticks to the 2 μS clock.

If this parameter is set to “1” (or “-1”), the number of segments closest to the frequency value will be used, without using ‘clockadj’. Only the 2 μS clock is used without the 100 nS clock. This is illustrated in the following formula:

segments = 1 / (frequency * 2 μS)

Normally, 'force_seg_count' is used (set to "1" or "-1") if you wish to have the segment counts of two waveforms be a multiple of each other to guarantee phase lock. However, this method uses more segment memory since the number of segments determines the frequency.



* width = Pulse Width in seconds ('Tpulse_width' parameter)
freq = Burst Frequency in Hz ('frequency' parameter)

Figure 3-4. Typical Burst Waveform

Testplan '`arbburst.tpa`' illustrates how to generate a burst modulated waveform, with waveform frequency set to 30 kHz, burst frequency set to 1 kHz, and burst pulse width set to 0.0001 seconds. A review of the testplan is below.

Test Group/Test/Action Name	Description
testgroup Arb burst waveform	Outputs a burst waveform.
Switching	Connect Arb channel 1 output to ABus1 and UUT Common.
test setup Arb	Setup arb output circuitry.
arbConfOutControls	Configures Arb's output circuitry.
arbSet	Sends setup data to Arb.
test download burst	Setup arb to download a burst waveform.
Arb_Dl_Ext_Waveform	Setup a burst waveform with waveform frequency set to 30 kHz ('F_burst' parameter), burst frequency of 1 kHz ('frequency' parameter), and a burst pulse width of 100 µS ('Tpulse_width' parameter).
test output waveform	Selects and outputs a burst waveform by name.
arbInitiate	Initialize the Arb to output the waveform. Start outputting the waveform. The Arb at this time outputs the last downloaded waveform.
Arb_Select_Wave_By_Name	Selects and outputs the waveform using the name entered in the 'waveform_name' parameter (i.e., "wave1").
DialogOkay	This optional action is used here to view the current waveform (i.e., forward and reverse sweep).

Downloading a Swept Sine Waveform

The “Arb_Dl_Swept_Sine” and “Arb_Dl_Swept_Sine_Ex” actions perform frequency sweeps of sine waves. The “Arb_Dl_Swept_Sine” action only sweeps in the forward direction. The “Arb_Dl_Swept_Sine_Ex” action can sweep in both forward and reverse directions with a selectable delay between sweeps (i.e., last frequency remain for the selected time).

Testplan ‘`arbsweep.tpa`’ illustrates how to output a sine wave sweep using both actions. In one test, the sine wave sweeps from 1 kHz to 2 kHz with a sweep time of 0.5 seconds. In another test, the sine wave sweeps from 1 kHz to 2 kHz in both forward and backwards directions with a sweep time of 0.2 seconds. In the The latter test, the last frequency is held for 0.5 seconds before the next sweeps is initiated. A review of the testplan is below.

Note The number of segments needed to generate the swept sine waves depend on the frequency span between the start and end frequencies, and the sweep time. Large spans and/or sweep times could exceed the maximum allowable segments.

Test Group/Test/Action Name	Description
testgroup Arb sine sweeps	Sweeps Arb sine waves.
Switching	Connect Arb channel 1 output to ABus1 and UUT Common.
test setup Arb	Setup arb output circuitry.
arbConfOutControls	Configures Arb’s output circuitry.
arbSet	Sends setup data to Arb.
test setup forward sweep	Sweeps a sine wave in the forward direction.
Arb_Dl_Swept_Sine	Downloads a sine wave for a forward sweep from 1 kHz to 2 kHz for a time of 0.5 seconds.
test setup forward & reverse sweeps	Sweeps a sine wave in both forward and reverse direction.
Arb_Dl_Swept_Sine_Ex	Downloads a sine wave for a forward and reverse sweep from 1 kHz to 2 kHz for a time of 0.2 seconds with a delay of 0.5 seconds between sweeps.
test execute sweeps	Execute both sweeps.
arbInitiate	Initialize the Arb to output the waveform. Start outputting the waveform. The Arb at this time outputs the last downloaded waveform (i.e., forward and reverse sweep).
Arb_Select_Wave_By_Name	Selects and outputs the waveform using the name entered in the ‘waveform_name’ parameter (i.e., “wave1”).
DialogOkay	This optional action is used here to view the current waveform (i.e., forward sweep).
Arb_Select_Wave_By_Name	Selects and outputs the waveform using the name entered in the ‘waveform_name’ parameter (i.e., “wave2”).
DialogOkay	This optional action is used here to view the current waveform (i.e., forward and reverse sweep).

Note Use the Watch Window to determine the amount of segment memory used by the swept sine waveforms.

Generating Custom Waveforms

Custom waveforms can be generated using the “Arb_Dl_Waveform_Data” action and or the “Arb_Dl_Custom_Waveform” actions.

Downloading a Data Type Custom Waveform

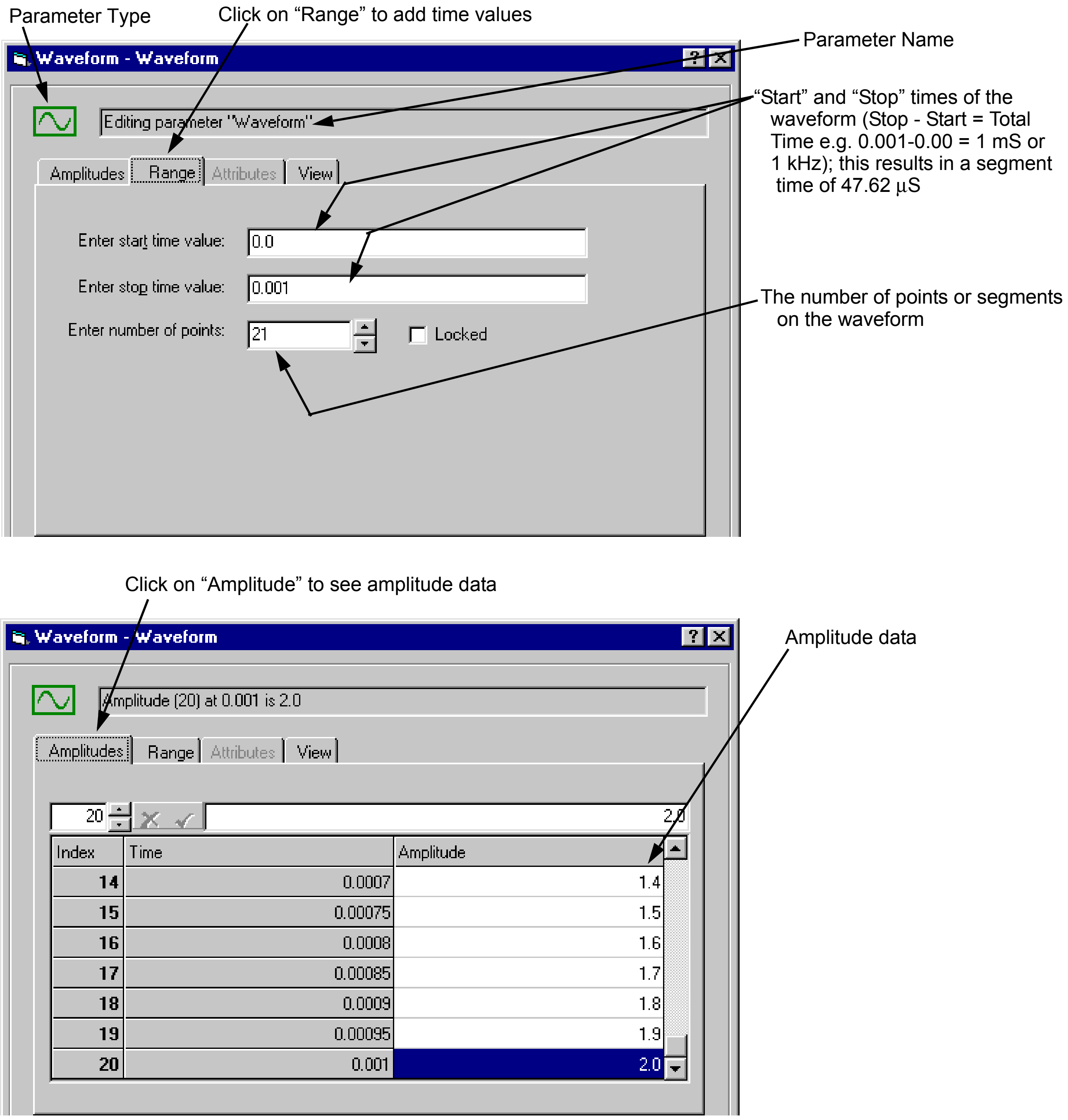
The “Arb_Dl_Waveform_Data” action uses a ‘Waveform’ type array variable called “Waveform” to generate a custom waveform. The variable stores both the segments of the waveform (i.e., amplitude) and the time all segments are to be executed (i.e., waveform frequency).

Figure 3-5 shows the data used to generate the waveform shown in the bottom part of the figure. The segment data of the waveform is stored in the array part of the “Waveform” parameter. (You can view the resultant waveform by clicking on the “View Waveform” button, as shown in Figure 3-5.)

How to download custom data type waveforms is shown in testplan ‘`arbcustom_data.tpa`’. The waveform is a 1 kHz (1 mS), 2 V peak ramp, consisting of 21 segments. A review of the testplan is below.

Test Group/Test/Action Name	Description
testgroup download custom waveform	Downloads a user generated data type custom waveform.
Switching	Connect Arb channel 1 output to ABus1 and UUT Common.
test setup Arb	Setup arb output circuitry.
arbConfOutControls	Configures Arb’s output circuitry.
arbSet	Sends setup data to Arb.
test download custom waveforms	Downloads custom waveforms to Arb.
Arb_Dl_Waveform_Data	Downloads data type waveform into Arb. The waveform data is the data stored into the ‘Waveform’ variable (see Figure 3-5 for typical waveform data). The ‘waveform_name’ parameter is set to “wave1” to give the downloaded waveform that name. The maximum value in the ‘Waveform’ array variable is “2.0000000000” which makes the peak Waveform voltage 2.0 V. The Star and /Stop values in the ‘Waveform’ array are set to “0.0000000000” and “.0010000000”, respectively. This outputs a 1 kHz waveform (Stop - Start = .0010000000 - 0.0000000000 = 1 mS or 1 kHz).
test output custom waveform	This test executes the downloaded waveform. Although the Arb outputs the waveform from the previous test (after ‘arbinitiate’ in “test download custom waveform”), this test is to show that the waveform can be selected in any part of the testplan. The Arb normally outputs the last downloaded waveform, whether a custom waveform or a standard waveform, but can output a different waveform in any part of the testplan using the action in this test.
arbInitiate	Initialize the Arb to output the waveform. Start outputting the waveform.

Test Group/Test/Action Name	Description
Arb_Select_Wave_By_Name	Selects and outputs the waveform using the name entered in the ‘waveform_name’ parameter (i.e., “wave1”) This is the same name as the name in the ‘waveform_name’ parameter of the “Arb_Dl_Waveform_Data” action.
DialogOkay	This optional action is used here to view waveform.



Parameter Type

Click on “Range” to add time values

Parameter Name

“Start” and “Stop” times of the waveform (Stop - Start = Total Time e.g. 0.001-0.00 = 1 mS or 1 kHz); this results in a segment time of 47.62 µS

The number of points or segments on the waveform

Waveform - Waveform

Editing parameter "Waveform"

Amplitudes | Range | Attributes | View

Enter start time value: 0.0

Enter stop time value: 0.001

Enter number of points: 21

□ Locked

Click on “Amplitude” to see amplitude data

Amplitude data

Waveform - Waveform

Amplitude (20) at 0.001 is 2.0

Amplitudes | Range | Attributes | View

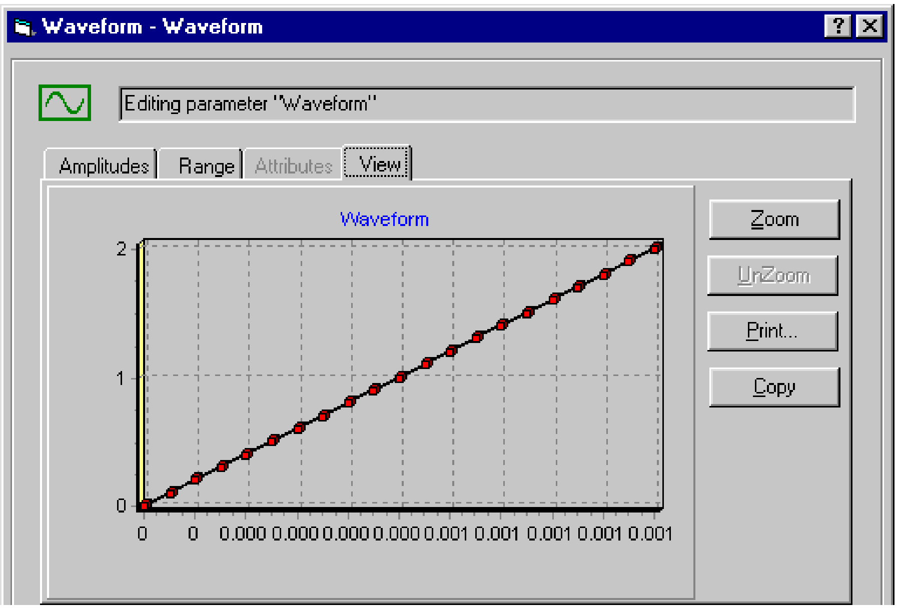
20 2.0

Index	Time	Amplitude
14	0.0007	1.4
15	0.00075	1.5
16	0.0008	1.6
17	0.00085	1.7
18	0.0009	1.8
19	0.00095	1.9
20	0.001	2.0

Click on “View Waveform” to view resultant waveform (see next figure)

Figure 3-5. Generating a Data Type Waveform

The values in this example produces a 2 Vpeak, 1 kHz ramp, as shown below; the waveform was generated using the following values



Editing parameter "Waveform"

0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
1.1
1.2
1.3
1.4
1.5
1.6
1.7
1.8
1.9
2.0
and Stop / Start values of
0.000 / 0.001

Note: The Arb output looks more “staircased” on an oscilloscpe due to the 47.62 µS/segment time.

Figure 3-6. Arb Waveform

Downloading User Defined Waveforms

This “Arb_Dl_Custom_Waveform” action uses several double and single dimensioned array type parameters to set the voltage segments, sequences, markers, dwell, next sequences, and other values to download and generate user defined custom waveforms.

Figure 3-3 shows a typical custom waveform. The figure shows a triangle waveform that shows the relationship between the segments, sequence, and segment execution time.

To generate a waveform, it is important to understand how to set the different parameters of the action. This is explained as follows:

voltage - Specifies the voltage value of the waveform segments for one or more sequences (voltage range is ±16.4 V). This is a double dimensioned array where the column is the sequence number and the rows are the segment numbers. Figure 3-7 shows a typical example.

marker - Specifies a marker on the selected segment in a valid sequence. The marker pulse will only be output if the sequence is non zero. The marker signal always appears on the Arb’s front panel BNC connector. It can also be routed to the VXI TTL backplane. Figure 3-8 shows a typical example on how to specify markers.

nr_sequence - Specifies the number of sequences to be downloaded. This value must be “1” or above, or no waveform will be downloaded. If more than one sequence is available and the parameter is set to “1”, only the waveform using the first sequence is downloaded.

next_sequence - Specifies which sequence is to be downloaded next. An example is in Figure 3-9.

sequence_repeat - Specifies if the sequence is to be repeated (i.e., value is set to “1” or more), as shown in Figure 3-10.

nr_segments - Specifies the number of segments for each sequence. If the number of specified segments is lower than the number of segments entered in the corresponding sequence of the ‘voltage’ parameter, only the number of segments specified here will be downloaded. Figure 3-11 shows a typical example.

segment_dwell - Specifies for how many 2 µS clock cycles each segment of a sequence is to be output. Normally, each segment is output in a single 2 µS clock cycle (if the parameter is set to “0”), which is a dwell count of “0”. Setting the ‘segment_dwell’ to a number above zero, the clock cycles will increase by that new number plus a single clock cycle.

Since the number of segments and the duration time of the segments determines the waveform frequency, each additional number added to ‘segment_dwell’ lowers the waveform frequency.

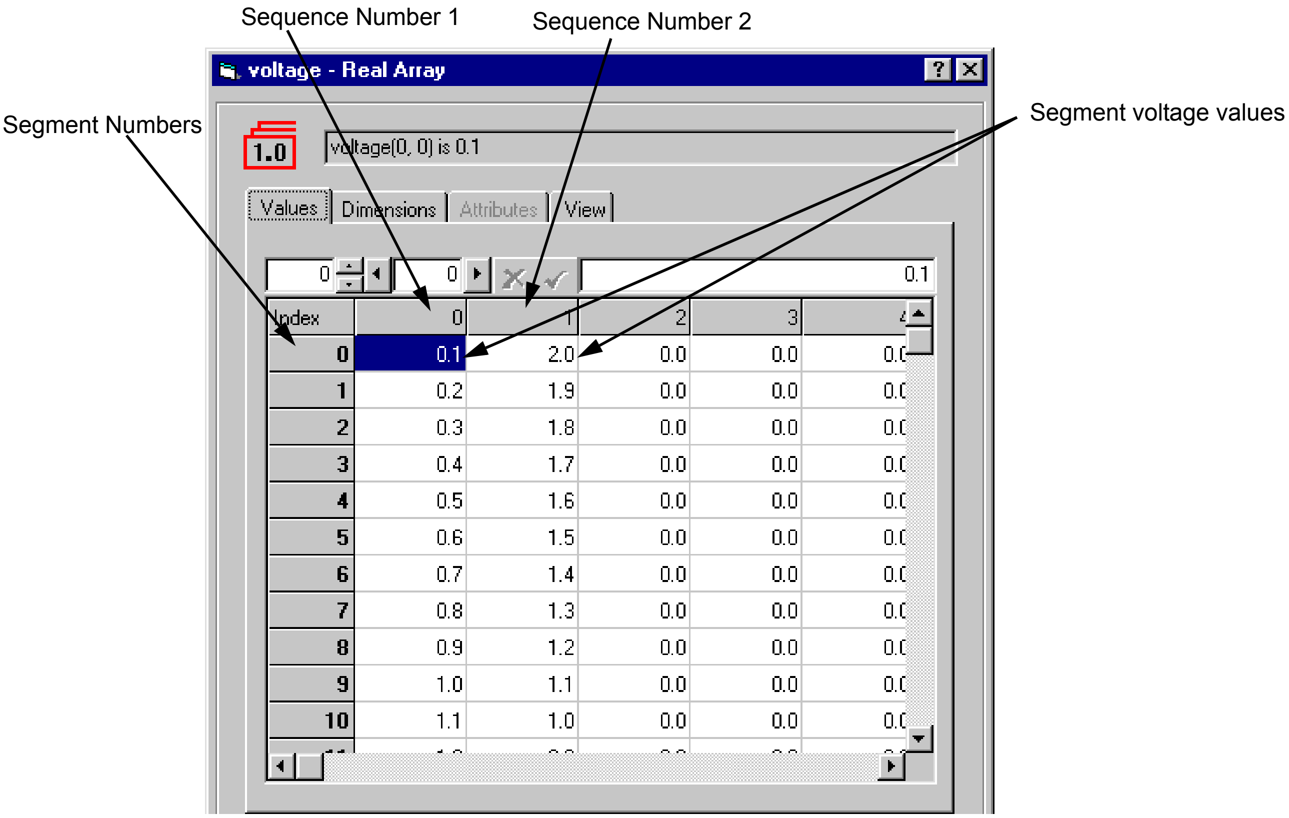
For example, if the dwell count is 0, for a 20 segment waveform, the total time the segments are output is:

20 * 2 µS = 40 µS = a frequency of 25 kHz

Changing the dwell count to 49, changes the time and resultant waveform frequency to:

20 * ((49 * 2 µS) + 2 µS) = 20 * ((98 µS) + 2 µS) = 20 * 100 µS = 0.002 S = a frequency of 500 Hz.

An example of using the ‘segment_dwell’ parameter is in Figure 3-12. In the figure, the dwell count is 49 for two 20 segment sequences which gives a total waveform frequency of 250 Hz.



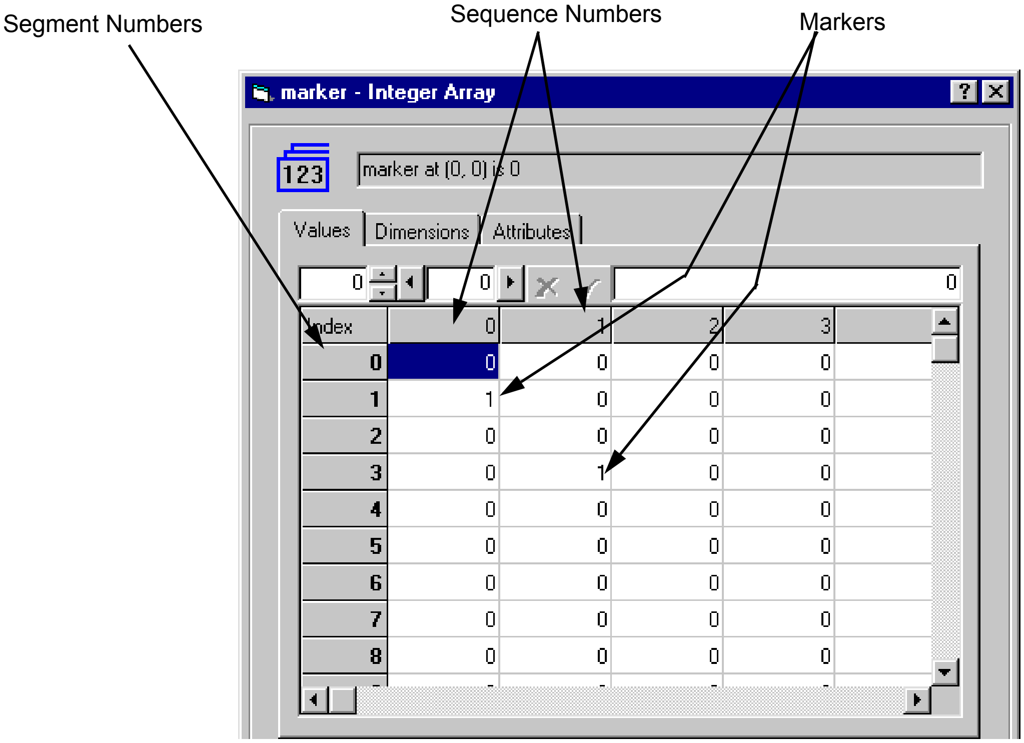
voltage - Real Array

voltage(0, 0) is 0.1

0 | 0 | 0.1

Index	0	1	2	3	4
0	0.1	2.0	0.0	0.0	0.0
1	0.2	1.9	0.0	0.0	0.0
2	0.3	1.8	0.0	0.0	0.0
3	0.4	1.7	0.0	0.0	0.0
4	0.5	1.6	0.0	0.0	0.0
5	0.6	1.5	0.0	0.0	0.0
6	0.7	1.4	0.0	0.0	0.0
7	0.8	1.3	0.0	0.0	0.0
8	0.9	1.2	0.0	0.0	0.0
9	1.0	1.1	0.0	0.0	0.0
10	1.1	1.0	0.0	0.0	0.0

Figure 3-7. Segment/Sequence Array Values



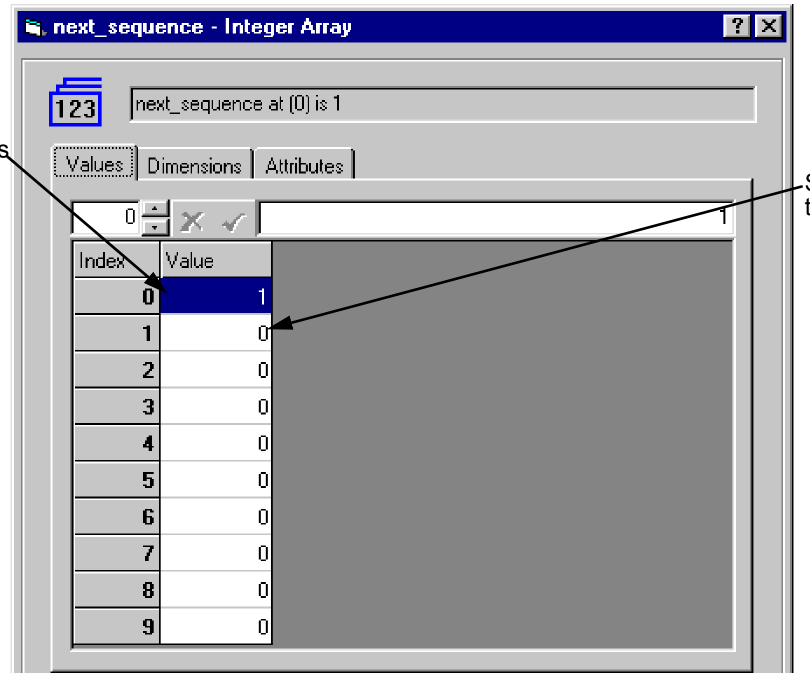
marker - Integer Array

marker at (0, 0) is 0

0 | 0 | 0

Index	0	1	2	3
0	0	0	0	0
1	1	0	0	0
2	0	0	0	0
3	0	1	0	0
4	0	0	0	0
5	0	0	0	0
6	0	0	0	0
7	0	0	0	0
8	0	0	0	0

Figure 3-8. Typical Marker Values



next_sequence - Integer Array

next_sequence at (0) is 1

0

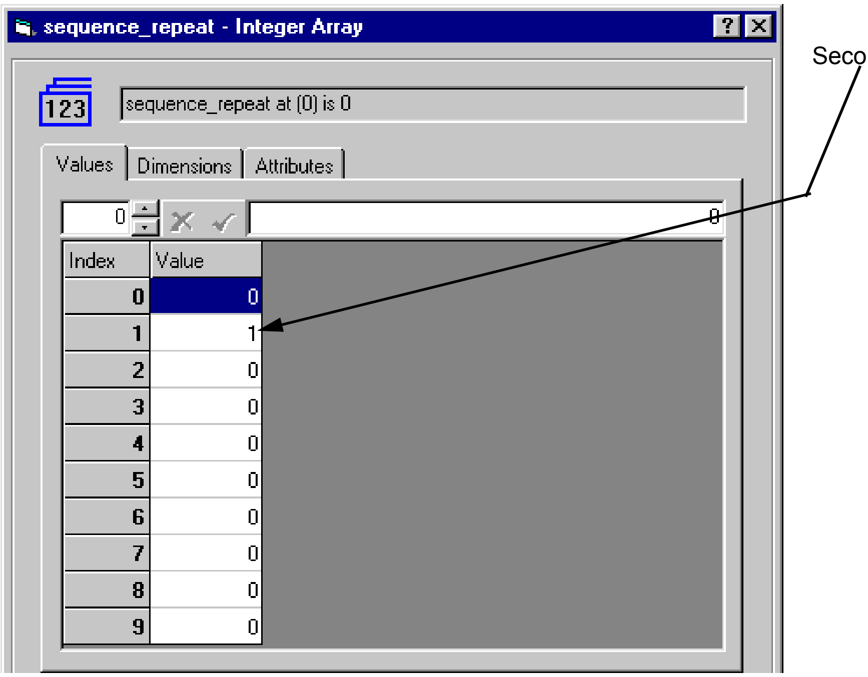
1

Index	Value
0	1
1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0

First Sequence points to Second Sequence

Second Sequence points to First Sequence

Figure 3-9. Typical Next Sequence Values



sequence_repeat - Integer Array

sequence_repeat at (0) is 0

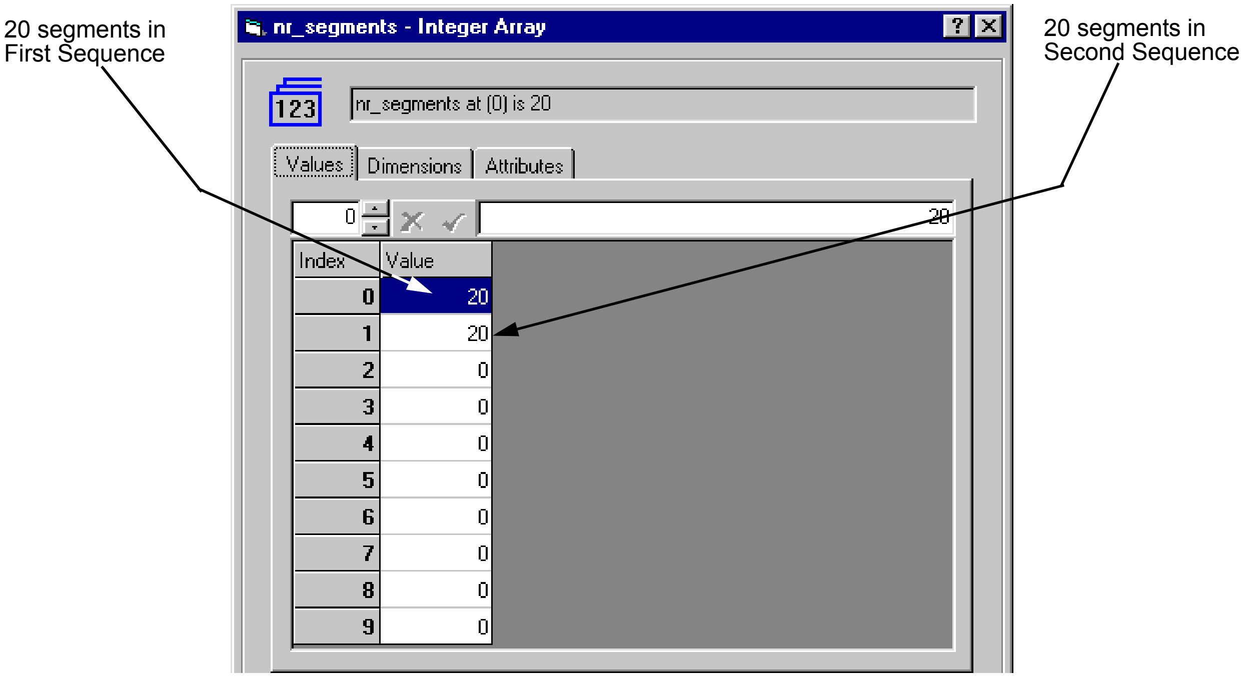
0

0

Index	Value
0	0
1	1
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0

Second Sequence is repeated 1 times

Figure 3-10. Typical Repeat Sequence Value



nr_segments - Integer Array

nr_segments at (0) is 20

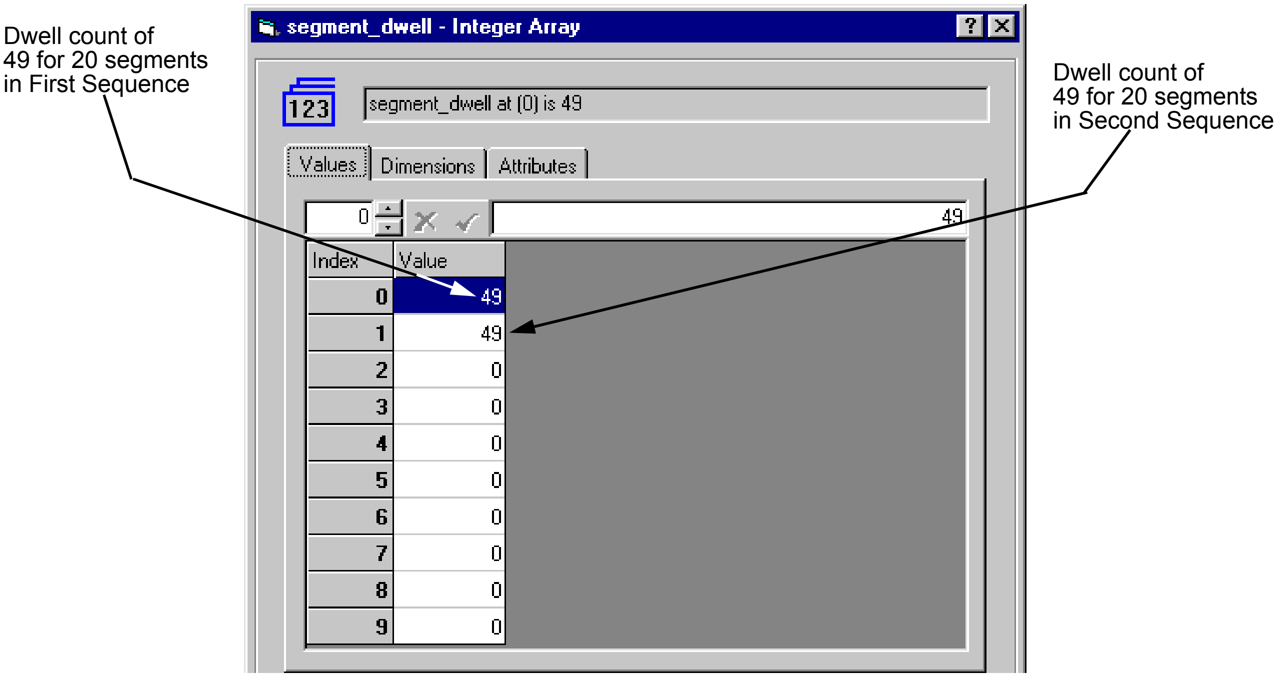
0 | 20

Index	Value
0	20
1	20
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0

20 segments in First Sequence

20 segments in Second Sequence

Figure 3-11. Typical Segment Number Value



segment_dwell - Integer Array

segment_dwell at (0) is 49

0 | 49

Index	Value
0	49
1	49
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0

Dwell count of 49 for 20 segments in First Sequence

Dwell count of 49 for 20 segments in Second Sequence

Waveform time for both sequences is: total of (segments * clock time for each sequence) =
20 * ((49 * 0.000002) + 0.000002) = 20 * (0.000098 + 0.000002) = 20 * 0.0001 = 0.002 per sequence

Since both sequence time is the same, the total time = 2 * 0.002 = 0.004 = 250 Hz

Figure 3-12. Typical Segment Number Value

How to download custom waveforms is shown in testplan ‘`arbcustom_user.tpa`’. A review of the testplan is as follows:

Test Group/Test/Action Name	**Description**
testgroup download custom waveform	Downloads user generated custom waveforms.
Switching	Connect Arb channel 1 output to ABus1 and UUT Common.
test setup Arb	Setup arb output circuitry.
arbConfOutControls	Configures Arb’s output circuitry.
arbSet	Sends setup data to Arb.
test download custom waveforms	Downloads the custom waveforms to Arb.
Arb_Dl_Custom_Waveform	Downloads two ramps (one positive the other negative going) to the Arb to output a triangle waveform. The number of segments is 20 for each waveform. The waveforms require 2 sequences. The ‘segment_dwell’ is set to 49 for both waveforms to output at 250 Hz.
test execute waveforms	Executes custom waveforms.
arbInitiate	Initialize the Arb to output the waveforms. Start outputting the waveform.
Arb_Select_Wave_By_Name	Selects and outputs the waveform using the name entered in the ‘waveform_name’ parameter (i.e., “wave1”) This is the same name as the name in the ‘waveform_name’ parameter of the “Arb_Dl_Custom_Waveform” action.
DialogOkay	This optional action is used here to view waveform.

Generating Custom Waveforms From a Data File

This is similar to generating a data type custom waveform using the “Arb_Dl_Waveform_Data” action, except in this case the waveform data comes from a data file using the “WaveformDataRead” action. The following shows how to download custom waveform data from a file and other pertinent information.

Downloading from a File

The following shows how the actions are used:

1. The “WaveformDataRead” action reads the data from a file and stores it into its ‘Waveform” variable (at first run of the testplan only).
2. The data is then stored into the ‘Waveform’ variable of the “Arb_Dl_Waveform_Data” action.
3. The data is downloaded into the Arb and output whenever the “Arb_Select_Wave_By_Name” action selects the appropriate waveform name.

To perform the correct operation, the following must take place:

- The “WaveformDataRead” action must be referenced to the data file using the file name. Figure 3-13 shows how the file is referenced.

  Normally, “WaveformDataRead” action reads the data file when the testplan is first run (i.e., the System RunCount = 0). After that, the data file will not be read.

- The waveform time/frequency must be set by the ‘SampleTime’ parameter in the “WaveformDataRead” action. Figure 3-13 shows how to set the waveform time/frequency using the ‘SampleTime’ parameter.

- The ‘Waveform’ variable in the “Arb_Dl_Waveform_Data” action must be referenced to the ‘Waveform’ variable in the “WaveformDataRead” action using symbols. Figure 3-14 shows how to create a symbol in the “WaveformDataRead” action and Figure 3-15 shows how to reference to the symbol in the “Arb-Dl_Waveform_Data” action.

- The data in the file must be in such a format that each line is a data point or segment of the waveforms. The following is a typical example on how data is stored into the file:

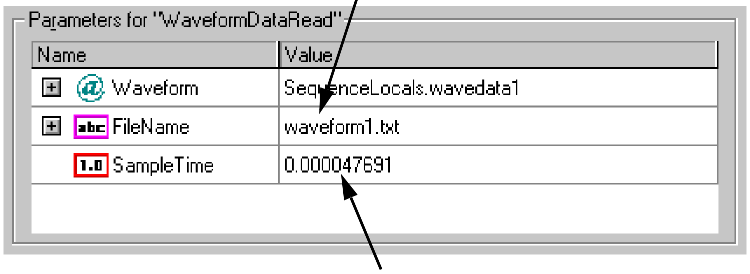
```
0.0000000000
0.1000000000
0.2000000000
    ●
    ●
    ●
0.5000000000
```

Note A data file using this format can be generated by an E1563A Digitizer using an appropriate action or the using the "WaveformDataWrite" or other supplied standard action.

How to download a custom data type waveform is shown in testplan '`arbcustom_file.tpa`'. The downloaded waveform is a 1 kHz (1.0 mS), 2 V peak ramp, consisting of 21 segments. A review of the testplan is as follows:.

Test Group/Test/Action Name	Description
testgroup Arb waveform from file	Downloads a waveform from a file.
Switching	Connect Arb channel 1 output to ABus1 and UUT Common.
test setup Arb	Setup arb output circuitry.
arbConfOutControls	Configures Arb's output circuitry.
arbSet	Sends setup data to Arb.
test get and download data	Gets download waveform data from file and downloads the data to the Arb.
WaveformDataRead	Reads data from file and store it in an array variable.
Arb_Dl_Waveform_Data	Download the data into the Arb that was read from the file to generate a data type waveform. The 'waveform_name' parameter is set to "wave1" to give the downloaded waveform that name. The waveform time or frequency is set to 1 kHz using the 'SampleTime' parameter value in the "WaveformDataRead" action where frequency = 1 / (sample time * number of samples) = 1 / (0.000047619 * 21 = 0.001 S = 1 kHz
test output custom waveform	Executes the downloaded waveform.
arbInitiate	Initialize the Arb to output the waveforms. Start outputting the waveform.
Arb_Select_Wave_By_Name	Selects and outputs the waveform using the name entered in the 'waveform_name' parameter (i.e., "wave1") This is the same name as the name in the 'waveform_name' parameter of the "Arb_Dl_Waveform_Data" action.
DialogOkay	This optional action is used here to view waveform.

Enter file name and path here

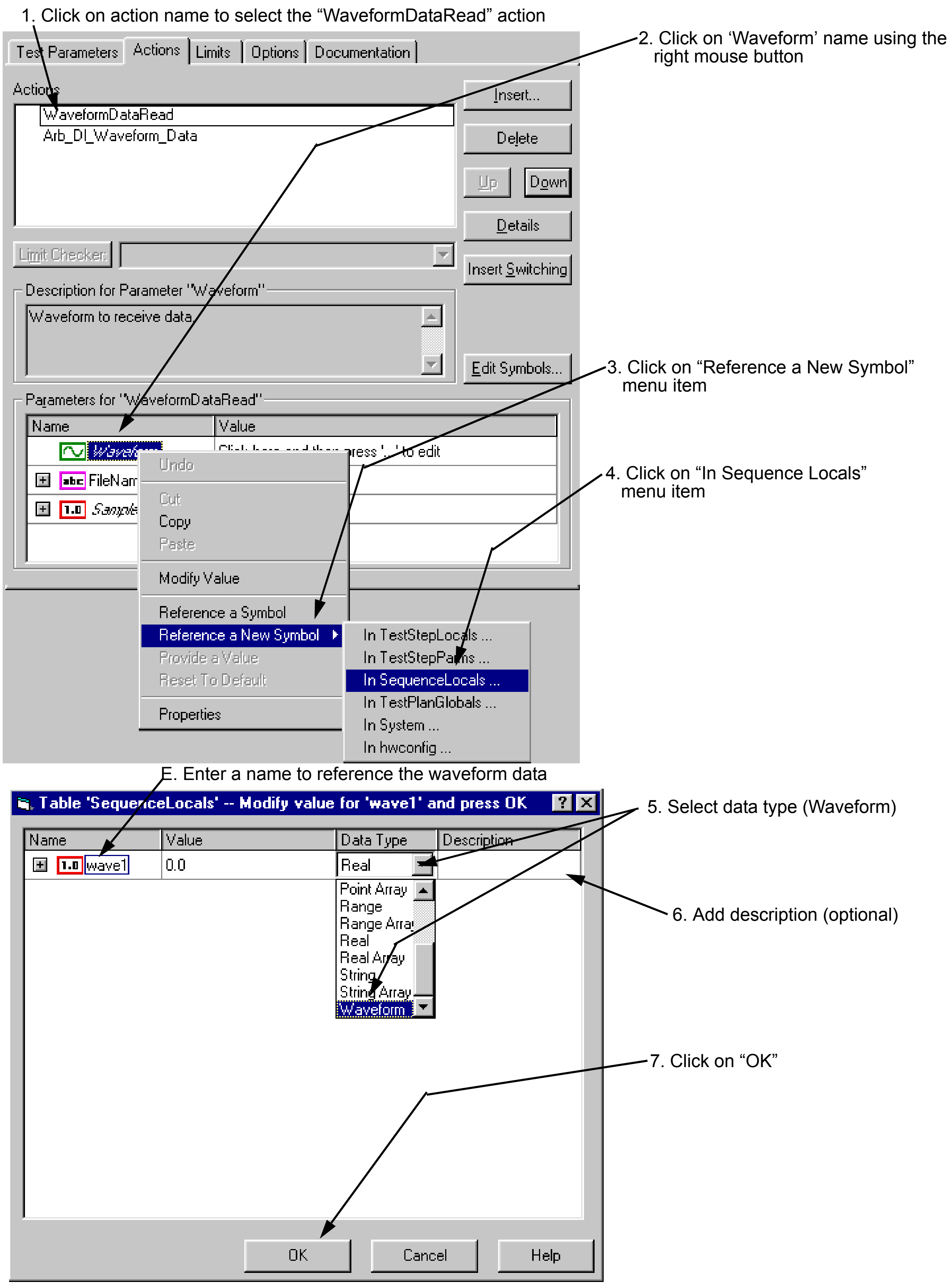


Parameters for "WaveformDataRead"

Name	Value
Waveform	SequenceLocals.wavedata1
FileName	waveform1.txt
SampleTime	0.000047691

Enter waveform time/frequency here (e.g., "SampleTime' of 0.000047619 = frequency of 1 kHz (0.000047619 * 21) = 0.001 S; 1 / 0.001 = 1 kHz)

Figure 3-13. Entering File Name and Sample Time



1. Click on action name to select the "WaveformDataRead" action

2. Click on 'Waveform' name using the right mouse button

3. Click on "Reference a New Symbol" menu item

4. Click on "In Sequence Locals" menu item

E. Enter a name to reference the waveform data

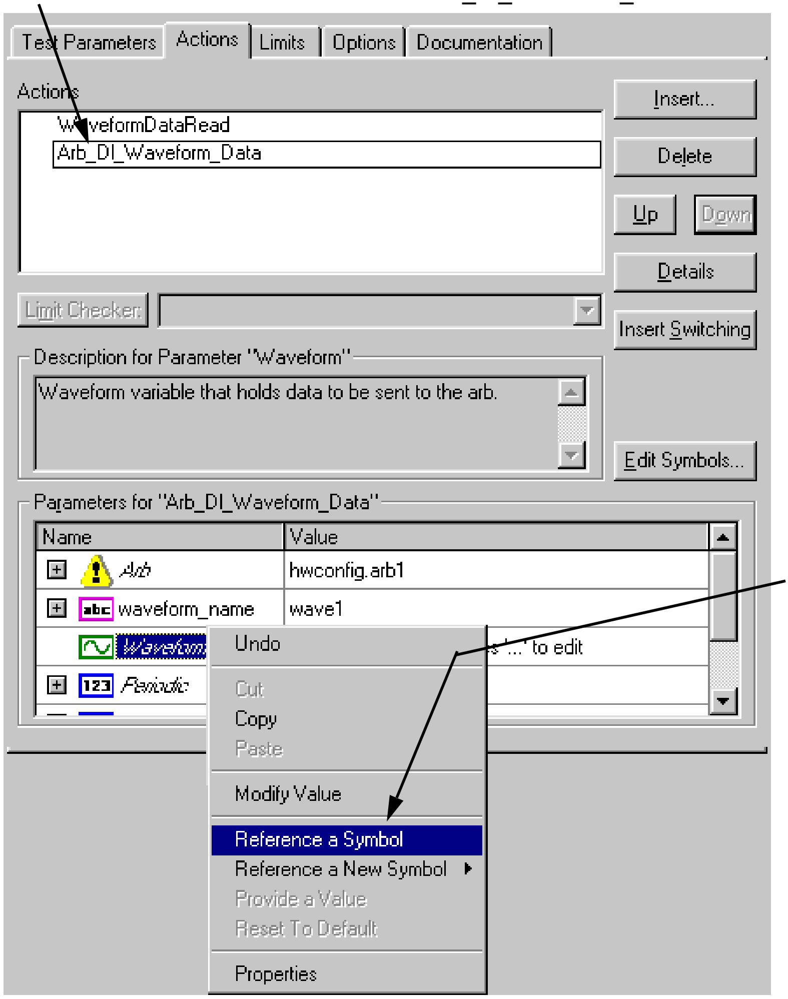
5. Select data type (Waveform)

6. Add description (optional)

7. Click on "OK"

Figure 3-14. Create a Symbol for the "WaveformDataRead" Action

1. Click on action name to select the “Arb_DI_Waveform_Data” action



Actions

WaveformDataRead
Arb_DI_Waveform_Data

Description for Parameter "Waveform"

Waveform variable that holds data to be sent to the arb.

Parameters for "Arb_DI_Waveform_Data"

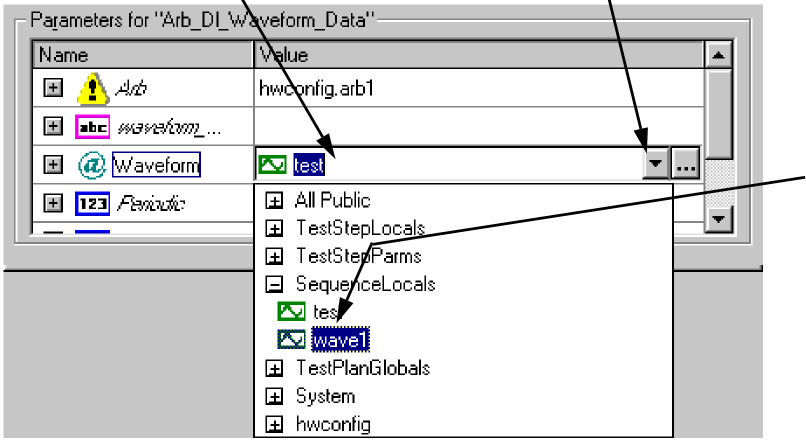
Name	Value
Arb	hwconfig.arb1
waveform_name	wave1
Waveform	...' ...' to edit
Periodic	

2. Click on “Reference a Symbol” menu item

3. The program automatically selects a symbol. If only one symbol is stored, it selects that symbol. If more than one symbol is stores, you must select the symbol, as shown in steps D to F

4. Click on parameter value to select the parameter

5. Click on the arrow to select the symbol (you can also click on the three dots to select it, but a different procedure is used)



Parameters for "Arb_DI_Waveform_Data"

Name	Value
Arb	hwconfig.arb1
waveform_...	
Waveform	test
Periodic	

6. Click on the appropriate symbol name to select it

Figure 3-15. Reference a Symbol in the “Arb_DI_Waveform_Data” Action

Creating a Data File

The data file is an ASCI file that contains waveform data. You can add data manually where each segment data point is a line in the file. You can also use the "WaveformDataWrite" action to store the data into the file. For example, you can use the action to store a waveform read by the E1563A A/D Converter into a file, then download the file to the Arb and output.

Memory Requirements for Downloading Waveforms

For the Arb to achieve high throughput, it downloads into memory all waveforms to be used in the testplan. A waveform can then be output in any part of the testplan.

The Arbis limited on how many waveforms can be stored, depending on the size of the waveforms. Normally, the Arb has sufficient memory for all the waveforms in a testplan unless the testplans use a very large number of waveforms or several long waveforms. The Arb driver then generates an error; normally an "Segment count too big during download" error (see "Agilent E6173A Arbitrary Waveform Generator Error Messages" below).

In case of a memory overflow, the only way for the Arb to receive new waveforms is to remove the current waveforms from memory. To do this, use the "arbReleaseWaveforms" action. This action will then remove all waveforms downloaded for the selected channel.

Note It it becomes necessary to use the "arbReleaseWaveforms" action, place the action both at the beginning of the testplan and at the point where more waveforms need to be added.

Agilent E6173A Arbitrary Waveform Generator Error Messages

The following lists the error numbers in hex and corresponding error messages the Arb can generate. Use the messages to troubleshoot the testplan or Arb, if needed.

Table 3-1. Agilent E6173A Error Codes and Descriptions

Error Number	Description
21	Unimplemented function call
22	Timeout waiting for arbStop
23	Illegal segment voltage
24	Segment count too big during download
25	Segment range error (setSegment)
26	Sequence range error (setSequence)
27	Illegal trigger input
28	Not channel 1 or 2
29	Sequence function without a segment fn
2A	Segment funct. without a sequence fn
2B	Couldn't find specified download ID
2C	Sequence count too big during download

Chapter 4

Action Summary

This chapter summarizes the TestExec SL actions supplied with the Agilent TS-5000 System. The action directory path is:

```
C:\Program Files\Agilent\TS-5000 System Software\actions
```

Actions are located in sub-directories and summarized in the following tables:

Sub-Directory	Table and Action Type
arb	Table 4-1, “arb (Arbitrary Waveform Generator) Actions”
counter	Table 4-2, “Counter Actions”
dac	Table 4-3, “dac (D/A Converter) Actions”
daq	Table 4-4, “daq (Digital Acquisition) Actions”
dgn	Table 4-5, “dgn (Diagnostic) Actions”
digitizer	Table 4-6, “digitizer Actions”
dio	Table 4-7, “dio (Digital I/O) Actions”
dmm	Table 4-8, “DMM Actions”
dso	Table 4-9, “DSO Actions”
eload	Table 4-10, “Electronic Load Actions”
esa	Table 4-11, “Spectrum Analyzer Actions”
esg	Table 4-12, “Signal Generator Actions”
event	Table 4-13, “Event Detector Actions”
generic	Table 4-14, “Miscellaneous Actions”
mcm	Table 4-15, “Measurement Control Module Actions”
msginst	Table 4-16, “msginst Actions”
power	Table 4-17, “Power Supply Actions”
serial	Table 4-18, “Serial Interface Actions”
SerialProtocols	Table 4-19, “Automotive Serial Protocol Actions”
switch	Table 4-20, “SLU and Switching Actions”
vi	Table 4-21, “Voltage/Current Source Actions”
esa	Table 4-22, “Spectrum Analyzer Actions”
esg	Table 4-23, “Signal Generator Actions”

Note This chapter contains action summaries. The actions are documented in detail in the TS5000 online help. This help is available from the Help menu in TestExec SL.

Table 4-1. arb (Arbitrary Waveform Generator) Actions

Actions	Descriptions
AgAwg3352xErrorQuery	Queries the instrument and returns instrument specific error information.
AgAwg3352xGetBurstProperties	This action gets the burst properties in the 33522A/33521A arbitrary waveform generator.
AgAwg3352xGetOutputProperties	This action gets the output properties in the 33522A/33521A arbitrary waveform generator.
AgAwg3352xOutputArbWaveform	This action sets up, downloads and enables an arbitrary waveform at the output of 33522A/33521A.
AgAwg3352xOutputStdWaveform	This action generates a basic waveform and enables the waveform at the output of 33522A/33521A.
AgAwg3352xPreset	This action presets the 33522A/33521A parameter.
AgAwg3352xQueryCmd	This action sends a SCPI query command to the instrument.
AgAwg3352xReset	This action sends the command string *RST to the instrument.
AgAwg3352xSelfTest	This action sets the instrument to run self test, waits for test completion, and queries the results.
AgAwg3352xSendCmd	This action sends a SCPI command to the instrument.
AgAwg3352xSendSoftwareTrigger	This action sends a software trigger to the instrument to trigger a sweep or burst from the remote interface only if the bus (software) trigger source is currently selected.
AgAwg3352xSetBurstProperties	This action sets the burst properties of the 33522A/33521A arbitrary waveform generator.
AgAwg3352xSetOutputEnable	This action configures whether the signal the 33522A/33521A produces appears at the front-panel output connector for the selected channel.
AgAwg3352xSetOutputProperties	This action sets the output properties in the 33522A/33521A arbitrary waveform generator.
AgAwg3352xSetTrigger	This action configures the trigger source and slope of the 33522A/33521A arbitrary waveform generator.
Agt33220GetBurstProperties	Returns the values of the IAgilent33220Burst interface properties.
Agt33220GetOutputProperties	Returns the values of the IAgilent33220Output interface properties.
Agt33220MsgReceive	Receives SCPI commands via the IAgilent33220System interface.
Agt33220MsgSend	Sends SCPI commands via the IAgilent33220System interface.
Agt33220MsgSendReceive	Receives SCPI commands via the IAgilent33220System interface.
Agt33220OutputArbWaveform	Receives SCPI commands via the IAgilent33220System interface.
agt33220OutputWaveform	High level action to generate a basic waveform.
Agt33220PutBurstProperties	Sets the values of the IAgilent33220Burst interface properties.
Agt33220SetOutputProperties	Sets the values of the IAgilent33220Output interface properties.
Agt33220SetTrigger	Sets up the trigger source.
Agt33220SoftTrigger	This action will trigger the 33220.

Table 4-1. arb (Arbitrary Waveform Generator) Actions (continued)

Actions	Descriptions
Arb_Download_Deluxe	**High-Level Action:** Not recommended for use in new testplans (Use "Arb_Dl_Ext_Waveform" or "Arb_Dl_Custom_Waveform"). Downloads a sine, square, or triangle waveform of specified frequency, amplitude, offset and phase to the Agilent E6173 Arb. For backwards compatibility, waveforms are numbered and must start at 0. Use "Arb_Select_Waveform" or "Arb_Select_Synced_Waveform" with this routine. **Usage:** 1. Call "arbConfTrigIn" to select trigger if syncing multiple cards. 2. Call "arbConfOutControls" to select the Arb output controls. 3. Call "arbSet" action to make output control changes take effect. 4. Call this action as many times as necessary to download the data. Be sure to increment the "waveform_nr" parameter by "1" for each download, starting with number "0". 5. Call "arbInitiate" to start waveform generation. 6. Call "Arb_Select_Waveform" to output the waveform as set by "waveform_nr". **Notes:** 1. Do all downloads for a given channel before switching to another Arb channel. 2. "waveform_nr" MUST start with 0 and increment by 1 until done; the waveforms are selected by this number. 3. The last waveform downloaded will be the one that is started when calling "arbInitiate". After that, waveforms can be selected on the fly using the number in the "waveform_nr" parameter. 4. Do not mix Arb_Download_Deluxe with "arbSetSegment" or "arbSetSequence" Action unless they program different Arb channels. 5. To use channel 2, change the "arb" parameter to "@hwconfig.arb2".

Table 4-1. arb (Arbitrary Waveform Generator) Actions (continued)

Actions	Descriptions
Arb_Download	**High-Level Action:** Not recommended for use in new testplans (use "Arb_Dl_Std_Waveform" or "Arb_Dl_Ext_Waveform"). Downloads a sine, square, triangle, knock, or crank waveform of specified frequency, amplitude, and offset to the Agilent E6173 Arbitrary Waveform Generator (Arb). Also allows duty cycle for pulse trains. For a DC level, set frequency to 0 Hz. For backwards compatibility, waveforms are numbered and must start at 0. **Usage:** 1. Call "arbConfOutControls" to select the Arb output controls. 2. Call "arbSet" action to make output control changes take effect. 3. Call this action as many times as necessary to download the data. Be sure to increment the "waveform_nr" parameter by "1" for each download, starting with number "0". 4. Call "Arb_Disable_Download" to disable the downloading of data. 5. Call "arbInitiate" to start waveform generation. 6. Call "Arb_Select_Waveform" or "Arb_Select_Synced_Waveforms" to output the waveform as set by "waveform_nr". **Notes:** 1. Do all downloads for a given channel before switching to another Arb channel. 2. "waveform_nr" MUST start with 0 and increment by 1 until done; the waveforms are selected by this number. 3. The last waveform downloaded will be the one that is started when calling "arbInitiate". After that, waveforms can be selected on the fly using the number in the "waveform_nr" parameter. 4. Do not mix Arb_Download with "arbSetSegment" or "arbSetSequence" Action unless they program different Arb channels. 5. To use channel 2, change the "arb" parameter to "@hwconfig.arb2".
arbSetClkAdjust	**Low-Level Action:** Change a channel's clock timing on the fly. **Notes:** 1. Action is channel independent. Clock timing changes immediately. It is not synchronized to any point on the waveform. Clock Adjust is always overwritten during any of the waveform select actions. 2. To use channel 2, change the "arb" parameter to "@hwconfig.arb2".
arbConfWaveform	arb

Table 4-1. arb (Arbitrary Waveform Generator) Actions (continued)

Actions	Descriptions
Arb_Disable_Download	**High-Level Action:** Not recommended for use in new testplans. Disables the "Arb_Download", "Arb_Download_Waveform" and "Arb_Download_Deluxe" actions. Use this action after all the "Arb_Download_xxx" actions in a testplan to avoid download overhead to the Agilent E6173 Arb on later testplan executions. Downloads are automatically re-enabled when a testplan is loaded or edited. **Usage:** 1. Call the "Arb_Download" Action to download the waveform. 2. Call this action to disable the download. **Notes:** Any download actions encountered in the testplan after this action executes are ignored.
Arb_Dl_Custom_Waveform	**High-Level Action:** Downloads a custom, user defined waveform consisting of up to 10 sequences with 50 segments per sequence to the Agilent E6173 Arb. Use "Arb_Select_Wave_By_Name" or "Arb_Select_Synced_Waves_By_Name" with this routine. **Usage:** 1. Call "arbConfTrigIn" to select trigger if syncing multiple cards. 2. Call "arbConfOutControls" to select the Arb output controls. 3. Call "arbSet" action to make output control changes take effect. 4. Call this action as many times as necessary to download the data. Be sure "waveform_name" parameter is unique for each download. 5. Call "arbInitiate" to start waveform generation. 6. Call "Arb_Select_Wave_By_Name" to output the waveform as set by "waveform_name". **Notes:** 1. The last waveform downloaded will be the one that is started when calling "arbInitiate". After that, waveforms can be selected on the fly using the name in the "waveform_name" parameter. 2. Do not mix this action with "arbSetSegment" or "arbSetSequence" Action unless they program different Arb channels. 3. To use channel 2, change the "arb" parameter to "@hwconfig.arb2".

Table 4-1. arb (Arbitrary Waveform Generator) Actions (continued)

Actions	Descriptions
Arb_Dl_Waveform_Data	**High-Level Action:** Downloads a waveform data type to the Agilent E6173 Arbitrary Waveform Generator (Arb). Waveform sample data is downloaded as segments with sample_time = (Stop-Start)/Sample_count, where Stop and Start are the waveform times and Sample_count is the number of samples in the waveform. Waveforms should be given a unique and descriptive name. **Usage:** 1. Call "arbConfOutControls" to select the Arb output controls. 2. Call "arbSet" to transfer configuration settings to the arb hardware. 3. Call this action or the other "Arb_Dl_xxx" Action as many times as needed to download custom or standard waveforms. 4. Call "Arb_Initiate" to start waveform generation. 5. Call "'Arb_Select_Wave_By_Name" or "Arb_Select_Synced_Waves_By_Name" to switch between waveforms. a. If Periodic = 1, the waveform will be generated as a continuos periodic waveform. b. If Periodic = 0, Repeat is used as a count of how many times to repeat the waveform. c. Marker sets the sample index for an arb marker signal for triggering (use arbConfTrigOut to configure). d. If Marker = -1, no marker is set. (Sample index starts with 0.) **Notes:** This action cannot be used with "arbSetSegement" or "arbSetSequence" on the same arb channel.

Table 4-1. arb (Arbitrary Waveform Generator) Actions (continued)

Actions	Descriptions
Arb_Dl_Ext_Waveform	**High-Level Action:** Downloads a sine, square, or triangle waveform of specified frequency, amplitude, offset and phase (with optional burst modulation) to the Agilent E6173 Arb. Use "Arb_Select_Wave_By_Name" or "Arb_Select_Synced_Waves_By_Name" with this routine. **Usage:** 1. Call "arbConfTrigIn" to select trigger if syncing multiple cards. 2. Call "arbConfOutControls" to select the Arb output controls. 3. Call "arbSet" action to make output control changes take effect. 4. Call this action as many times as necessary to download the data. Be sure "waveform_name" parameter is unique for each download. 5. Call "arbInitiate" to start waveform generation. 6. Call "Arb_Select_Wave_By_Name" to output the waveform as set by "waveform_name". **Notes:** 1. The last waveform downloaded will be the one that is started when calling "arbInitiate". After that, waveforms can be selected on the fly using the name in the "waveform_name" parameter. 2. Do not mix this action with "arbSetSegment" or "arbSetSequence" actions unless they program different Arb channels. 3. To use channel 2, change the "arb" parameter to "@hwconfig.arb2".
Arb_Dl_Swept_Sine	**High-Level Action:** Downloads a basic swept sine waveform of specified frequency, amplitude, and offset to the Agilent E6173 Arbitrary Waveform Generator (Arb). **Usage:** 1. Call "arbConfOutControls" to set the state of the Arb output amplifier parameters for this channel. 2. Call "arbSet" to transfer configuration settings to the arb hardware. 3. Call this action as many times as necessary to download the data. Be sure the "waveform_name" parameter is unique. 4. Call "arbInitiate" to start waveform generation. 5. Call "Arb_Select_Wave_By_Name" or "Arb_Select_Synced_Waves_By_Name" to output the waveform as set by "waveform_name". **Notes:** 1. The last waveform downloaded will be the one that is started when calling "arbInitiate". After that, waveforms can be selected on the fly using the name in the "waveform_name" parameter. 2. Do not mix Arb_Dl_xxx actions with "arbSetSegment" or "arbSetSequence" actions unless they program different Arb channels. 3. To use channel 2, change the "arb" parameter to "@hwconfig.arb2".

Table 4-1. arb (Arbitrary Waveform Generator) Actions (continued)

Actions	Descriptions
Arb_Dl_Swept_Sine_Ex	**High-Level Action:** Downloads a swept sine waveform of specified frequency, amplitude, and offset to the Agilent E6173 Arbitrary Waveform Generator (Arb). Allows adjustment of advanced parameters such as reverse sweep time and F_end hold time. **Usage:** 1. Call "arbConfOutControls" to set the state of the Arb output amplifier parameters for this channel. 2. Call "arbSet" to transfer configuration settings to the arb hardware. 3. Call this action as many times as necessary to download the data. Be sure the "waveform_name" parameter is unique. 4. Call "arbInitiate" to start waveform generation. 5. Call "Arb_Select_Wave_By_Name" or "Arb_Select_Synced_Waves_By_Name" to output the waveform as set by "waveform_name". **Notes:** 1. The last waveform downloaded will be the one that is started when calling "arbInitiate". After that, waveforms can be selected on the fly using the name in the "waveform_name" parameter. 2. Do not mix Arb_Dl_xxx actions with "arbSetSegment" or "arbSetSequence" actions unless they program different Arb channels. 3. To use channel 2, change the "arb" parameter to "@hwconfig.arb2".

Table 4-1. arb (Arbitrary Waveform Generator) Actions (continued)

Actions	Descriptions
Arb_Dl_Std_Waveform	**High-Level Action:** Downloads a sine, square/pulse or triangle waveform of specified frequency, amplitude, and offset to the Agilent E6173 Arbitrary Waveform Generator (Arb). Also allows duty cycle adjustment for pulse trains. **Usage:** 1. Call "arbConfOutControls" to set the state of the Arb output amplifier parameters for this channel. 2. Call "arbSet" to transfer configuration settings to the arb hardware. 3. Call this action as many times as necessary to download the data. Be sure the "waveform_name" parameter is unique. 4. Call "arbInitiate" to start waveform generation. 5. Call "Arb_Select_Wave_By_Name" or "Arb_Select_Synced_Waves_By_Name" to output the waveform as set by "waveform_name". **Notes:** 1. For a DC level, set frequency to 0 Hz. 2. The last waveform downloaded will be the one that is started when calling "arbInitiate". After that, waveforms can be selected on the fly using the name in the "waveform_name" parameter. 3. Do not mix Arb_Dl_xxx actions with "arbSetSegment" or "arbSetSequence" actions unless they program different Arb channels. 4. To use channel 2, change the "arb" parameter to "@hwconfig.arb2". 5. This action uses the same waveform generation algorithm as "Arb_Download" did.

Table 4-1. arb (Arbitrary Waveform Generator) Actions (continued)

Actions	Descriptions
Arb_Download_Waveform	**High-Level Action:** Not recommended for use in new testplans (use "Arb_Dl_Waveform_Data"). Downloads a waveform data type to the Agilent E6173 Arbitrary Waveform Generator (Arb). Waveform sample data is downloaded as segments with sample_time = (Stop-Start)/Sample_count, where Stop and Start are the waveform times and Sample_count is the number of samples in the waveform. For backwards compatibility, waveforms are numbered and must start at 0. **Usage:** 1. Call "arbConfOutControls" to select the Arb output controls. 2. Call "arbSet" action to make output control changes take effect. 3. Call this action or the "Arb_Download" Action as many times as needed to download custom or standard waveforms. 4. Call "Arb_Initiate" to start waveform generation. 5. Call "'Arb_Select_Waveform" or "Arb_Select_Synced_Waveform" to switch between waveforms.(selected by waveform number). a. If Periodic = 1, the waveform will be generated as a continuos periodic waveform. b. If Periodic = 0, Repeat is used as a count of how many times to repeat the waveform. c. Marker sets the sample index for an arb marker signal for triggering (use arbConfTrigOut to configure). d. If Marker = -1, no marker is set (Sample index starts with 0). **Notes:** This action cannot be used with "arbSetSegement" or "arbSetSequence" on the same arb channel.
Arb_Enable_Download	**High-Level Action:** Not recommended for use in new testplans. Enables the "Arb_Download" Action. Use it to re-enable the "Arb_Download" Action to download data to the Agilent E6173 Arb. This action is not commonly used during normal testplan execution. Downloads are automatically enabled when a testplan is loaded or edited. **See Also:** Arb_Download, Arb_Disable_Download **Instrument:** Arb
Arb_DL_Datafile	**High-Level Action:** Creates a waveform from a data file.
arbInitiate	**Low-Level Action:** Starts the arb sequencer to output the waveform. **Usage:** After calling this action, trigger the Arb to start the output, if different than a FREERUN trigger. **Notes:** 1. Starts both channels if synchronized by "arbConfSync". 2. To use channel 2, change the "arb" parameter to "@hwconfig.arb2".

Table 4-1. arb (Arbitrary Waveform Generator) Actions (continued)

Actions	Descriptions
arbIsSet	**Low-Level Action:** Program arb - transfers all arb settings specified by the arbConf... routines to registers in the hardware by calling the driver's "setup" function. (Action returns a true.) **Usage:** 1. Call any configuration action(s) to set up the Arb. 2. Call this action to send the setup information to the Arb. 3. Call "arbInitiate" to start the output. **Notes:** To use channel 2, change the "arb" parameter to "@hwconfig.arb2".
Arb_Source_MDA	**High-Level Action:** Programs the arb to act as the MDA source.
arbConfOutControls	**Low-Level Action:** Configures the Agilent E6173 Arb's output circuitry. **Usage:** 1. Add the appropriate values into the parameters and then call this action. 2. Call any other configuration action(s). 3. Call "arbSet" to sent the setup information to the Arb. **Requires:** "arbIsSet" or "arbSet" after this action. **Notes:** 1. Call this action after a reset (Arb or global) to re enable the output relay(s). 2. To use channel 2, change the "arb" parameter to "@hwconfig.arb2".
arbReset	**Low-Level Action:** Causes a hard reset of the arb hardware. The output connect relay is left open. Waveform data in A24 memory is not changed. **Usage:** Call this action when the Arb configuration is unknown and the Arb is to be configured into a known state. **Notes:** Use either channel to reset the Arb (both are reset; clears the synchronized state).
arbReleaseWaveforms	**Low-Level Action:** Causes the Agilent E6173 handler and driver to "forget" about all previously downloaded waveforms. This would only be useful in a testplan that required so many downloads that the arb's waveform memory overflowed. **Notes:** 1. The action executes immediately. 2. Only the channel specified is affected. 3. To use channel 2, change the "arb" parameter to "@hwconfig.arb2".

Table 4-1. arb (Arbitrary Waveform Generator) Actions (continued)

Actions	Descriptions
arbSet	**Low-Level Action:** Program arb - transfers all arb settings specified by the arbConf... routines to registers in the hardware by calling the driver's "setup" function. **Usage:** 1. Call any configuration action(s) to set up the Arb. 2. Call this action to send the setup information to the Arb. **Notes:** 1. This action only sends configuration information to the Arb and does not start operation. 2. To use channel 2, change the "arb" parameter to "@hwconfig.arb2".
arbSetSegment	**Low-Level Action:** Not recommended for use in new testplans (use Arb_DI_Custom_Waveform). Specify data for one segment of a waveform. First segment of a sequence must start at an even segment number. Last segment of a sequence must have EOS bit set. **Usage:** 1. The segment requires a segment number ("segment" parameter) and the voltage value of the segment ("voltage" parameter). At the end of the segments in a sequence, the end of sequence flag ("eos" parameter) must be set true (1). A marker for trigger output use can be placed in any segment by setting the marker flag ("marker" parameter) to true (1). 2. A typical segment action is: arbSetSegment (arb, 0, 5, 0, 1) - where:arb=instrument handle, 0=segment number, 5=voltage, 0=end of sequence flag, and 1=marker pulse. **Requires:** "arbSetSequence" to select the segments. **Notes:** 1. A sequence must start on an even segment number. 2. A sequence can have any number of segments. 3. To use channel 2, change the "arb" parameter to "@hwconfig.arb2".

Table 4-1. arb (Arbitrary Waveform Generator) Actions (continued)

Actions	Descriptions
arbSetSequence	**Low-Level Action:** Not recommended for use in new testplans (use Arb_Dl_Custom_Waveform). Specify sequence data for a waveform. **Usage:** 1. The sequence requires a sequence number ("sequence" parameter), the segment number to start with ("segment" parameter from the "arbSetSegment" action), number of times to repeat the sequence ("repeat" parameter). To execute a number of clock cycles between each segment, set the "dwell" parameter to that number. To stop after a sequence, set the stop flag ("stop" parameter) to true (1). 2. A typical sequence action is: arbSetSequence (arb, 0, 0, 1, 100, 0, 1) - where:arb=instrument handle, 0=sequence number, 0=starting segment number, 1=number of times to repeat sequence, 100=number of clock cycles between segments, 0=number of the next sequence, 1=stop flag. **Requires:** "arbSetSegment" before this action to generate the segments. **Notes:** To use channel 2, change the "arb" parameter to "@hwconfig.arb2".
arbConfTimeout	**Low-Level Action:** Sets the I/O time-out value (in milliseconds) for communication with the Agilent E6173 Arb. **Usage:** 1. Call this and any other arbConfxxx actions. 2. Call "arbSet" to send the setup information to the Arb. **Notes:** Call this action for either channel. It is common to both.
arbSetWaveform	**Low-Level Action:** Not recommended for use in new testplans. Select waveform (by its sequence number) and sets clock adjust register. This routine can be called before or after arbInitiate. If the synchronous mode was enabled (via arbConfSync) the arb is stopped before the waveform is selected and restarted afterwards. **Usage:** Add the sequence number defined in the "arbSetSequence" Action and add the number of clock counts in the "clkadj" parameter. Each count in the "clkadj" parameter adds 0.1 microsecond to the segment time. **Notes:** 1. An incorrect sequence number may cause unpredictable results. 2. To use channel 2, change the "arb" parameter to "@hwconfig.arb2".

Table 4-1. arb (Arbitrary Waveform Generator) Actions (continued)

Actions	Descriptions
Arb_Output_Std_Waveform	**High-Level Action:** Configures Agilent E6173 Arb to generate a sine, pulse, or triangle waveform of specified frequency, amplitude, and offset. Allows duty cycle selection for pulse trains. For a DC level, set frequency to 0 Hz. Downloads the waveform and initiates the Arb. **Notes:** 1. The action generates a unique waveform name based on specified parameters so the waveform is only downloaded to the arb once during testplan execution. 2. It is not necessary to call the "arbInitiate" Action to output the waveform. 3. This action will override previous output settings made by the "arbConfOutControls" action. The output relay will be enabled and the output attenuator will be disabled. 4. To use channel 2, change the "arb" parameter to "@hwconfig.arb2".
Arb_Select_Waveform	**High-Level Action:** Not recommended for use in new testplans (Use "Arb_Select_Wave_By_Name"). Allows the selection of Agilent E6173 Arb waveforms (on the fly) referenced by number assigned in the "Arb_Download", "Arb_Download_Waveform" and "Arb_Download_Deluxe" actions. **Usage:** 1. Call "Arb_Download" Action to download the data. 2. Call "arbInitiate" to start waveform generation. 3. Call this action to select and output the waveform set by "waveform_nr". **Requires:** "Arb_Download" and "arbInitiate" Action before this action.
Arb_Select_Synced_Waves_By_Name	**High-Level Action:** Allows selection of Agilent E6173 Arb waveforms (on the fly) of both channels referenced by name assigned in the "Arb_Dl_xxx" actions. **Usage:** 1. Call "Arb_Dl_xxx" Action to download the data. 2. Call "arbInitiate" to start waveform generation. 3. Call this action to select and output the waveform set by "ch1_wave_name" and "ch2_wave_name". **Requires:** "Arb_Dl_xxx" Action before this action.

Table 4-1. arb (Arbitrary Waveform Generator) Actions (continued)

Actions	Descriptions
Arb_Select_Wave_By_Name	**High-Level Action:** Allows the selection of Agilent E6173 Arb waveforms (on the fly) referenced by name assigned in the "Arb_Dl_xxx" actions. **Usage:** 1. Call "Arb_Dl_xxx" Action to download the data. 2. Call "arbInitiate" to start waveform generation. 3. Call this action to select and output the waveform set by "waveform_name". **Requires:** "Arb_Dl_xxx" Action before this action. **Notes:** 1. This action can be called before arbInitiate to force start-up of a specific waveform. 2. To use channel 2, change the "arb" parameter to "@hwconfig.arb2".
Arb_Select_Synced_Waveforms	**High-Level Action:** Not recommended for use in new testplans (Use "Arb_Select_Synced_Waves_By_Name"). Allows selection of Agilent E6173 Arb waveforms (on the fly) of both channels referenced by number assigned in the "Arb_Download", "Arb_Download_Waveform" and "Arb_Download_Deluxe" actions. **Usage:** 1. Call "Arb_Download" or "Arb_Download_Deluxe" Action to download the data. 2. Call "arbInitiate" to start waveform generation. 3. Call this action to select and output the waveform set by "ch1_wave_num" and "ch2_wave_num". **Requires:** "Arb_Download" Action before this action.
arbStop	**Low-Level Action:** Stops waveform output, at end of current sequence, on a given channel. **Usage:** Call this action whenever the output of the Arb is to stop. **Notes:** If the sync mode in the "arbConfSync" Action is set true ("sync" parameter is 1), both channels will be stopped. Output disconnect relays are not affected.

Table 4-1. arb (Arbitrary Waveform Generator) Actions (continued)

Actions	Descriptions
arbConfSync	**Low-Level Action:** Enable/disable synchronous mode on a given arb channel pair. "Synchronous mode" effects how arbInitiate and arbStop work. When enabled, both channels on the Agilent E6173A start and stop at the same time, regardless of which channel was specified in the call. **Usage:** 1. Call any configuration actions. be sure to setup each channel for non shared configurations. 2. Call this action. 3. Call "arbSet" to send the setup information to the Arbs. 4. Call "arbInitiate" to start the output. **Requires:** "arbInitiate" or "arbSet" after this action. **Notes:** 1. Call this action for either channel only once to set the operating mode for both channels. 2. Synchronous mode is reset to 0 by "arbReset".
arbConfTimebase	**Low-Level Action:** Select internal 10MHz clock or an external clock supplied through the front panel BNC. **Usage:** 1. Add the appropriate values into the parameters and then call this action. 2. Call any other configuration action(s). 3. Call "arbSet" to send the setup information to the Arb. 4. Call "arbInitiate" to start the output. **Requires:** Must call "arbSet" after this action; "arbInitiate" is optional. **Notes:** Each channel can have its own clock source. To use channel 2, change the "arb" parameter to "@hwconfig.arb2".
arbConfTrigIn	**Low-Level Action:** Set input triggering parameters. Default after reset is free run. Other options include front panel (edge or level), TTL backplane, Hold (None). **Usage:** 1. Add the appropriate values into the parameters and then call this action. 2. Call any other configuration action(s). 3. Call "arbSet" to send the setup information to the Arb. 4. Call "arbInitiate" to start the output. **Requires:** Must call "arbSet" after this action; "arbInitiate" is optional. **Notes:** 1. Channels are set independent from each other, unless synchronized by "arbConfSync". 2. To use channel 2, change the "arb" parameter to "@hwconfig.arb2".

Table 4-1. arb (Arbitrary Waveform Generator) Actions (continued)

Actions	Descriptions
arbConfTrigOut	**Low-Level Action:** Set trigger parameters for output. Default after reset is none (off). Triggers are output to the specified backplane TTL trigger line. Triggers can originate from either the front panel or a marker on the waveform. **Usage:** 1. Add the appropriate values into the parameters and then call this action. 2. Call any other configuration action(s). 3. Call "arbSet" to send the setup information to the Arb. 4. Call "arbInitiate" to start the output. **Requires:** Must call "arbSet" after this action; "arbInitiate" is optional. **Notes:** 1. Channels are set independent from each other. 2. To use channel 2, change the "arb" parameter to "@hwconfig.arb2".
awgGetBurstProperties	This action is use to get the burst properties in the 33220A/33250A arbitrary waveform generator.
awgGetOutputProperties	This action is use to get the Output properties in the 33220A/33250A arbitrary waveform generator.
awgOutputArbWaveform	This action is use to downloads an arbitrary waveform an enables the output for the 33220A/33250A.
awgOutputStdWaveform	This action is use to generate a basic waveform from 33220A/33250A. **Notes:** When selecting DC, add additional time delay before measuring/using the output.
awgPreset	This action is to preset the 33220A/33250A parameter.
awgQueryCmd	This action is use to send a SCPI query command.
awgSendCmd	This action is use to send a SCPI command.
awgSendSoftwareTrigger	This action is use to send a software trigger to the 33220A/33250A.
awgSetBurstProperties	This action is use to set the burst properties in the 33220A/33250A arbitrary waveform generator.
awgSetOutputProperties	This action is use to set the Output properties in the 33220A/33250A arbitrary waveform generator.
awgSetTimeOut	This action is use to set time out (millisecond).
awgSetTrigger	This action is use to configure the trigger source and slope in the 33220A/33250A arbitrary waveform generator.
PXA722xawgChangesequence	This action changes the sequence number during output of waveform by the device.

Table 4-1. arb (Arbitrary Waveform Generator) Actions (continued)

Actions	Descriptions
PXA722xawgConfOutputArbWaveform	This action generates an arbitrary (or user defined) waveform from the device. **Usage:** 1. Call this action to load a user defined waveform data to a specific sequence number and waveform RAM. 2. Call "PXA722x_awgSetStartSequenceNumber" action to specify the sequence number. 3. Call "PXA722x_awgSoftwareTrigger" action to output this waveform. 4. Call "PXA722x_awgStopWaveform" action to stop this waveform output.
PXA722xawgErrQuery	This action queries error queue in the device.
PXA722xawgGetDeviceOption	This action retrieves device information on hardware options/features.
PXA722xawgGetInstStatus	This action returns the device status.
PXA722xawgGetOutputArbWaveformCon f	This action retrieves settings of a waveform at a specified sequence number.
PXA722xawgLoadSequenceFile	This action loads a user defined waveform from a sequence file. The sequence file is a file in .ini format where sequences are defined and waveform RAM organization is done. (For example: C:\Program Files\VX Instruments GmbH\PXA722x Arbitrary Waveform Generator\Example\Sequenzen\Standardkurvenformen.ini) **Usage:** 1. Call this action to load a user defined waveform. 2. Call "PXA722x_awgSetStartSequenceNumber" action to specify the sequence number. 3. Call "PXA722x_awgSoftwareTrigger" action to output this waveform. 4. Call "PXA722x_awgStopWaveform" action to stop this waveform
PXA722xawgOutputStdWaveform	This action is used to generate a basic waveform from PXA722x.
PXA722xawgReset	This action resets parameters (all parameters/parameters) of the PXA722x.
PXA722xawgRevQuery	This action queries the PXA722x for Driver Revision and Hardware Revision.
PXA722xawgSelfTest	This action commands the device to perform self test.

Table 4-1. arb (Arbitrary Waveform Generator) Actions (continued)

Actions	Descriptions
PXA722xawgSetStartsequenceNumber	This action sets the sequence number when the device is not outputting any waveform. **Usage:** 1. Call "PXA722x_awgLoadSequenceFile" or "PXA722x_awgConfOutputArbWaveform" action to load a user defined waveform. 2. Call this action to specify the sequence number. 3. Call "PXA722x_awgSoftwareTrigger" action to output this waveform. 4. Call "PXA722x_awgStopWaveform" action to stop this waveform output.
PXA722xawgSoftwareTrigger	This performs a Software Trigger to output the waveform.
PXA722xawgStopWaveform	This action stops the device output waveform.

Table 4-2. Counter Actions

Actions	Description
ag53220ConfInControls	**Action Type:** An instrument action that configures the counter inputs. **Usage:** Call this action before making any measurements. Use it to configure the instrument inputs.
ag53220ErrorStatus	**Action Type:** An instrument action that gets the error status from the HP53131. **Usage:** 1. Call this action to retrieve the status register from the instrument using ErrorQuery, mask off all but error bits with ErrorMask, and return the resulting integer in Error. If there is no error, this should result in Error = 0. 2. For IEEE-488.2, the defaults retrieve the Event Status Register and mask off all but error bits for Query, Execution, Device dependent, and Command error events. 3. IEEE-488.2 Standard error bits: a. 32 (bit 5):Command Error b. 16 (bit 4):Execution Error c. 8 (bit 3):Device Dependent Error d. 4 (bit 2):Query Error **Notes:** The instrument handler must be the hwhmsginst.dll for this action to work properly. Users can create custom versions of this action by saving a copy of this UMD file and changing the ErrorQuery or ErrorMask parameters to match a particular instrument.
ag53220MeasureAgain	**Action Type:** An instrument action that repeats the last measurement (use for speed). **Usage:** Call HP53131ConfInControls to configure the counter input, use one of the other measurement actions, then call this action to repeat that measurement.
ag53220MeasureFreq	**Action Type:** An instrument action used to measure frequency. **Usage:** Call HP53131ConfInControls to configure the counter input, then call this action. **Notes:** This action supports only voltage triggered measurements.
ag53220MeasureNegPulse	**Action Type:** An instrument action that measures negative pulse width. **Usage:** Call HP53131ConfInControls to configure the counter input, then call this action. **Notes:** This action supports only voltage triggered measurements.

Table 4-2. Counter Actions (continued)

Actions	Description
ag53220MeasurePeriod	**Action Type:** An instrument action used to measure period. **Usage:** Call HP53131ConfInControls to configure the counter input, then call this action. **Notes:** This action supports only voltage triggered measurements.
ag53220MeasureVpp	**Action Type:** An instrument action that measures the peak-to-peak voltage using the HP53131. **Usage:** Call HP53131ConfInControls to configure the counter input, then call this action. **Notes:** Measurements are triggered on positive-going voltage transitions.
ag53220MeasurePosPulse	**Action Type:** An instrument action that measures positive pulse width. **Usage:** Call HP53131ConfInControls to configure the counter input, then call this action. **Notes:** This action supports only voltage triggered measurements.
ag53220MeasTimeInterval	**Action Type:** An instrument action that sends a query command to an instrument and receives its response. **Usage:** Call this action to send a query to an instrument, then read a response from it. **Notes:** The instrument handler must be the hwhmsginst.dll for this action to work properly. Users can create custom versions of this action by saving a copy of this UMD file, adding task specific parameters to it, and using String Formatting on the Command string to send parameterized Queries and on the Response string to extract parameter from the Response.
ag53220Reset	**Action Type:** An instrument action that causes a hard reset of the HP53131instrument. **Usage:** Call this action when the instrument configuration is unknown and you want to configure it to a known state. **Notes:** The instrument handler must be the hwhmsginst.dll for this reset to work properly.
ctrMeasureFrequency	An Agilent E1333 Counter action that measures frequency.

Table 4-2. Counter Actions (continued)

Actions	Description
ctrMeasurePeriod	An Agilent E1333 Counter action that measures period.
ctrMeasurePulseWidth	An Agilent E1333 Counter action that measures pulse widths.
ctrMeasureRatio	High level Agilent E1333 Counter Action to measure Ratio using the counter.
ctrMeasureTimeInterval	High level Agilent E1333 Counter action that measures time interval.
ctrMeasureTotalize	High level Agilent E1333 Counter action that starts a totalize measurement.
ctrConfFunction	**Low-Level Action:** Selects function, range, and resolution of the Agilent E1333 Counter. **Usage:** 1. Add the appropriate values into the parameters and then call this action. Call any other configuration action(s). 2. Call "ctrIsSet" to wait until the Counter is ready to make measurements (in a loop if called from a C program). 3. Call "ctrInitiate" to start the measurement. 4. Call "ctrGetResults" or trigger the Counter and then call "ctrGetResults", dependent on the "trigfirst" parameter value, to read the measurement. **Requires:** "ctrInitiate" or "ctrGetResults" after this action. **Notes:** 1. To use channel 2 or channel 3, change the "ctr" parameter to "@hwconfig.cntr2" or "@hwconfig.cntr3", respectively. 2. However, channel 3 can only be used with function 1 (Frequency). 3. Function 6 (Totalize) does not use the "range" or "resolution" parameters.
ctrConfInControls	**Low-Level Action:** Configures the input circuit of the Agilent E1333 Counter. **Usage:** 1. Add the appropriate values into the parameters and then call this action. 2. Call any other configuration action(s). 3. Call "ctrIsSet" to wait until the Counter is ready to make measurements (in a loop if called from a C program). 4. Call "ctrInitiate" to start the measurement. 5. Call "ctrGetResults" or trigger the Counter and then call "ctrGetResults", dependent on the "trigfirst" parameter value, to read the measurement. **Requires:** "ctrInitiate" or "ctrGetResults" after this action. **Notes:** 1. To use channel 2 or channel 3, change the "ctr" parameter to "@hwconfig.cntr2" or "@hwconfig.cntr3", respectively. 2. Not all parameters can be set for channel 3 ("@hwconfig.cntr3").

Table 4-2. Counter Actions (continued)

Actions	Description
ctrInitiate	**Low-Level Action:** Starts the previously configured Agilent E1333 Counter measurement. **Usage:** After calling this action, trigger the Counter to make the measurement. Use the "ctrGetResults" Action to get the reading. **Notes:** To use channel 2 or channel 3, change the "ctr" parameter to "@hwconfig.cntr2" or "@hwconfig.cntr3", respectively.
ctrIsSet	**Low-Level Action:** Waits until the Agilent E1333 Counter is ready for measurement. (Action returns a true when ready.) **Usage:** 1. Call any configuration action(s) to set up the Counter. 2. Call this action to allow the Counter sufficient time for configuration to complete (in a loop if called from a C program). 3. Call "ctrInitiate" to start the measurement. 4. Call "ctrGetResults" or trigger the Counter and then call "ctrGetResults", dependent on the "trigfirst" parameter value, to read the measurement. **Notes:** To use channel 2 or channel 3, change the "ctr" parameter to "@hwconfig.cntr2" or "@hwconfig.cntr3", respectively.
ctrReset	**Low-Level Action:** Resets all of the Agilent E1333 Counter's channels to the power-on state. **Usage:** Call this action when the Counter configuration is unknown and the Counter is to be configured into a known state.
ctrGetResults	**Low-Level Action:** Reads the Agilent E1333 Counter results. **Usage:** Call this action to wait for a trigger using the signal or other trigger modes; then reads the results. **Notes:** 1. Can use optional triggering before taking a reading by setting the "trigfirst" parameter to "1". 2. To use channel 2 or channel 3, change the "ctr" parameter to "@hwconfig.cntr2" or "@hwconfig.cntr3", respectively.
ctrSetReturnMode	**Low-Level Action:** This action allows time-outs on the specified Agilent E1333 Universal Counter to raise an exception when encountered. **Notes:** The default is not to raise an exception with time-outs.
ctrSetTimeout	**Low-Level Action:** Sets the Agilent E1333 Counter operation time out value (in milliseconds). **Notes:** All channels are set to the same time-out value. **Instrument:** Counter

Table 4-2. Counter Actions (continued)

Actions	Description
CtrSelfTest	**Low-Level Action:** Runs a self-test on the Agilent E1333 Universal Counter that is provided by its Plug and Play driver. **Notes:** Routine attempts to restore previous counter state after the self-test. **Instrument:** Counter
ctrConfTrigIn	**Low-Level Action:** Sets the input trigger of the Agilent E1333 Counter. **Usage:** 1. Add the appropriate values into the parameters and then call this action. 2. Call any other configuration action(s). 3. Call "ctrIsSet" to wait until the Counter is ready to make measurements (in a loop if called from a C program). 4. Call "ctrInitiate" to start the measurement. 5. Call "ctrGetResults" or trigger the Counter and then call "ctrGetResults", dependent on the "trigfirst" parameter value, to read the measurement. **Requires:** "ctrInitiate" or "ctrGetResults" after this action. **Notes:** 1. All channels use the same trigger source. 2. The "triglevel" and "trigedge" parameters are set separately for channels 1 and 2 only. 3. To use channel 2 or channel 3, change the "ctr" parameter to "@hwconfig.cntr2" or "@hwconfig.cntr3", respectively. 4. Not all parameters can be set for channel 3 ("@hwconfig.cntr3").
ctr_ConfInControls	This actions configures input of selected channel: 1. Input impedance 2. Range 3. Probe factor 4. Coupling 5. Low Pass Filter 6. Auto-level 7. Threshold level 8. Slope
ctr_ErrorStatus	This action is queries error status.
ctr_GetInputCoupling	This actions gets input coupling of selected channel.
ctr_GetInputImpedance	This actions gets input impedance of selected channel.
ctr_GetInputLPFilter	This actions gets input low pass filter of selected channel.
ctr_GetInputProbeFactor	This actions gets input probe factor of selected channel.
ctr_GetInputRange	This actions gets input range of selected channel.
ctr_GetInputSlope	This actions gets input slope of selected channel.
ctr_GetInputThreshold	This actions gets input threshold level of selected channel.

Table 4-2. Counter Actions (continued)

Actions	Description
ctr_MeasAgain	This action repeats the last measurement.
ctr_MeasDutyCycle	This actions measures duty cycle.
ctr_MeasFreq	This action measures frequency.
ctr_MeasFreqRatio	This action measures frequency.
ctr_MeasPeriod	This action measures period.
ctr_MeasPosNegPulse	This action measures Positive/Negative pulse width.
ctr_MeasRiseFall	This actions measures Rise/Fall time.
ctr_MeasTimeInterval	This actions measures time interval.
ctr_MeasVMinMax	This actions measures min and max voltage.
ctr_MeasVpp	This actions measures peak to peak voltage.
ctr_PhaseMeas	This action performs phase measurements.
ctr_RecallSettings	This action recall settings from counter's memory.
ctr_Reset	This action causes a hard reset of the Agilent 53131A/53220A Universal Counter.
ctr_SaveSettings	This action saves present settings to counter's memory.
ctr_SelfTest	This action sets the 53131A/53220A to run self test.
ctr_SetDisplay	This action sets display ON/OFF.
ctr_SetInputCoupling	This actions sets input coupling of selected channel.
ctr_SetInputImpedance	This actions sets input impedance of selected channel.
ctr_SetInputLPFilter	This actions sets input low pass filter of selected channel.
ctr_SetInputProbeFactor	This actions sets input probe factor of selected channel.
ctr_SetInputRange	This actions sets input range of selected channel.
ctr_SetInputSlope	This actions sets input slope of selected channel.
ctr_SetInputThreshold	This actions sets input threshold level of selected channel.
ctr_SetTimeout	Only for 53131. This action specify time-out value.
ctr_TotalizeMeasCont_Conf	This actions configure continuous totalize measurement.
ctr_TotalizeMeasCont_Meas	This actions measure continuous totalize measurement.
ctr_TotalizeMeasCont_Start	Only for 53220. This actions start continuous totalize measurement.
ctr_TotalizeMeasCont_Stop	Only for 53220. This actions start continuous totalize measurement.
ctr_TotalizeMeasTimed	This actions configure timed totalize measurement.

Table 4-3. dac (D/A Converter) Actions

Actions	Descriptions
Agt34951ConfigureExternalPacing	Configure the trigger type and external clock divisor for the specified channels. After using this action to set up the external clock and trigger type, call the following functions: • Agt34951ConfigureWaveform • Agt34951StartWaveform
Agt34951ConfigureInternalPacing	Configure the trigger type and internal clock period for the specified channels. After using this action to set up the internal clock and trigger, call the following functions: • Agt34951ConfigureWaveform • Agt34951StartWaveform
Agt34951ConfigureTrigOut	Enable or disable the external trigger output of the DAC. This will cause the DAC to output a 3.5 V trigger signal when a waveform starts. **Notes:** This line is also the Trigger Input for external triggers. Thus this function can not be used at the same time an external trigger input is expected. See ConfigureExternalPacing and ConfigureInternalPacing.
Agt34951ConfigureWaveform	Configure the specified channels for waveform output. It will not start the waveform, nor will it configure the clock or trigger for the channel. Typical usage would involve the following sequence of actions: • Agt34951ConfigureTrigger • Agt34951ConfigureClock • Agt34951ConfigureWaveform • Agt34951StartWaveform Two shortcut actions exist which do all of this but does not allow control of the clock, trigger or number of cycles: • Agt34951SetVoltWaveform • Agt34951SetCurrentWaveform
Agt34951CreateTrace	Load trace data from an array of real values. Each point should be between -1.0 and +1.0. THE NUMBER OF POINTS IN THE ARRAY MUST BE SPECIFIED BY THE NumPoints PARAMETER.
Agt34951DebugInstErrs	Setting the debug flag to On will cause every action to query the instrument for errors upon completion of the action and raise an exception if there is an instrument error.
Agt34951DeleteAll	Delete all traces. **Notes:** Traces which are currently running will not be deleted! Call Agt34951HaltWaveform with all channels set to 1 and Disable set to 1 before calling this action.
Agt34951DeleteTrace	Delete the trace specified. **Notes:** Traces which are currently running will not be deleted! Call Agt34951HaltWaveform with all channels set to 1 and Disable set to 1 before calling this action.
Agt34951Enable	Enable, disable or leave unchanged multiple DAC channels. This will open or close the output relay for the specified channels.

Table 4-3. dac (D/A Converter) Actions (continued)

Actions	Descriptions
Agt34951GetChanState	Return the output state and trace state for a given channel.
Agt34951HaltWaveform	This action will halt waveforms running on the specified channels.
Agt34951LoadTrace	Load trace data from a text file. The file should contain one point per line, each point between -1.0 and +1.0. Minimum 8 points per file. Total trace data storage for the DAC is 512,000 points for all traces.
Agt34951MsgQuery	Send an instrument query and get the results.
Agt34951MsgReceive	Read the 34980 results buffer.
Agt34951MsgSend	Send a SCPI command to the 34980. **Notes:** This action is not slot specific and may affect other cards in the 34980 besides the 34951. If slot information is required for the command it must be supplied, unlike most 34951 actions.
Agt34951SetCurrentDC	Set up a DAC channel to output DC current.
Agt34951SetCurrentWaveform	This action will apply the specified trace to the output channels selected and enable the channels. **Defaults:** • Clock: Internal • Trigger: Immediate • # Cycles: Infinite To change these defaults or to avoid enabling the channels use the ConfigureWaveform action.
Agt34951SetStandardWaveform	Create trace data from a standard waveform: Ramp, Sine, Square or Triangle. Points are generated between +1.0 and -1.0. Minimum 8 points per trace. Total trace data storage for the DAC is 512,000 points for all traces.
Agt34951SetVoltDC	Set up a DAC channel to output DC voltage.
Agt34951SetVoltWaveform	This action will apply the specified trace to the output channels selected and enable the channels. Defaults: • Clock: Internal • Trigger: Immediate • # Cycles: Infinite To change these defaults or to avoid enabling the channels use the ConfigureWaveform action.
Agt34951StandardTrace	Create trace data from a standard waveform: Ramp, Sine, Square or Triangle. Points are generated between +1.0 and -1.0. Minimum 8 points per trace. Total trace data storage for the DAC is 512,000 points for all traces.
Agt34951StartWaveform	This action will start the waveforms on the specified channels. The waveforms must have already been configured with the Agt34951ConfigWaveform action.

Table 4-3. dac (D/A Converter) Actions (continued)

Actions	Descriptions
DAC: 16 Channels	**High-Level Action:** This action programs all channels of the 16 channel Agilent E1418 DAC either as a voltage or current source. This action can also connect the DAC before programming and disconnect after the measurement, and choose the Trigger Type. All 16 channel are connected, disconnected, and have source type set together. **Usage:** This action is self contained and makes several calls to other "dacXXX" Actions. This action generates an output when run. **Notes:** 1. This action is for the Agilent E1418 DAC only. 2. The current source setup works only in the immediate (imm) trigger mode.
DAC: 16 Chan (Individual)	**High-Level Action:** This action programs each channel of the 16 channel Agilent E1418 DAC either as a voltage or current source. This action can also connect the DAC before programming and disconnect after the measurement, and choose the Trigger Type. All 16 channels are connected and disconnected together. Each channel can be individually set as a voltage or current source. **Usage:** This action is self contained and makes several calls to other "dacXXX" Actions. This action generates an output when run. **Notes:** 1. This action is for the Agilent E1418 DAC only. 2. The current source setup works only in the immediate (imm) trigger mode.
DAC: 32 Chan (Individual)	**High-Level Action:** This action programs each channel of 2 - 16 channel Agilent E1418 DAC either as a voltage or current source. This action can also connect the DAC before programming and disconnect after the measurement, and choose the Trigger Type. All 32 channels are connected and disconnected together. Each channel can be individually set as a voltage or current source. Channel 17-32 is assumed to be the second dac for connect/disconnect purposes. **Usage:** This action is self contained and makes several calls to other "dacXXX" Actions. This action generates an output when run. **Notes:** 1. This action is for the Agilent E1418 DAC only. 2. The current source setup works only in the immediate (imm) trigger mode.
dacAdvanceConfigureExternalPacing	Configure the trigger type and external clock divisor for the specified channels. After using this action to set up the external clock and trigger type, call the following functions: • dacAdvanceConfigureWaveform • dacAdvanceStartWaveform

Table 4-3. dac (D/A Converter) Actions (continued)

Actions	Descriptions
dacAdvanceConfigureInternalPacing	Configure the trigger type and internal clock period for the specified channels. After using this action to set up the internal clock and trigger, call the following functions: • dacAdvanceConfigureWaveform • dacAdvanceStartWaveform
dacAdvanceConfigureTrigOut	Enable or disable the external trigger output of the DAC. This will cause the DAC to output a 3.5 V trigger signal when a waveform starts. **Notes:** This line is also the Trigger Input for external triggers. Thus this function can not be used at the same time an external trigger input is expected. See dacAdvanceConfigureExternalPacing and dacAdvanceConfigureInternalPacing.
dacAdvanceConfigureWaveform	Configure the specified channels for waveform output. It will not start the waveform, nor will it configure the clock or trigger for the channel. Typical usage would involve the following sequence of actions: • dacAdvanceConfigureTrigger • dacAdvanceConfigureClock • dacAdvanceConfigureWaveform • dacAdvanceStartWaveform Two shortcut actions exist which do all of this but does not allow control of the clock, trigger or number of cycles: • dacAdvanceSetVoltWaveform • dacAdvanceSetCurrentWaveform
dacAdvanceCreateTrace	Load trace data from an array of real values. Each point should be between -1.0 and +1.0. THE NUMBER OF POINTS IN THE ARRAY MUST BE SPECIFIED BY THE NumPoints PARAMETER.
dacAdvanceDebugInstErrs	Setting the debug flag to On will cause every action to query the instrument for errors upon completion of the action and raise an exception if there is an instrument error.
dacAdvanceDeleteAll	Delete all traces. **Notes:** Traces which are currently running will not be deleted! Call dacAdvanceHaltWaveform with all channels set to 1 and Disable set to 1 before calling this action.
dacAdvanceDeleteTrace	Delete the trace specified. **Notes:** Traces which are currently running will not be deleted! Call dacAdvanceHaltWaveform with all channels set to 1 and Disable set to 1 before calling this action.
dacAdvanceEnable	Enable, disable or leave unchanged multiple DAC channels. This will open or close the output relay for the specified channels.
dacAdvanceGetChanState	Return the output state and trace state for a given channel.
dacAdvanceHaltWaveform	This action will halt waveforms running on the specified channels.
dacAdvanceLoadTrace	Load trace data from a text file. The file should contain one point per line, each point between -1.0 and +1.0. Minimum 8 points per file. Total trace data storage for the DAC is 512,000 points for all traces.

Table 4-3. dac (D/A Converter) Actions (continued)

Actions	Descriptions
dacAdvanceMsgQuery	Send an instrument query and get the results.
dacAdvanceMsgReceive	Read theL4451A results buffer.
dacAdvanceMsgSend	Send a SCPI command to the L44xx. **Notes:** This action is not slot specific and may affect other L44XX cards. If slot information is required for the command it must be supplied, unlike most 4451 actions.
dacAdvanceReset	This action is use for L4451A reset.
dacAdvanceSetCurrentDC	Set up a DAC channel to output DC current.
dacAdvanceSetCurrentWaveform	This action will apply the specified trace to the output channels selected and enable the channels. **Defaults:** • Clock: Internal • Trigger: Immediate • # Cycles: Infinite To change these defaults or to avoid enabling the channels use the ConfigureWaveform action.
dacAdvanceSetStandardWaveform	Create trace data from a standard waveform: Ramp, Sine, Square or Triangle. Points are generated between +1.0 and -1.0. Minimum 8 points per trace. Total trace data storage for the DAC is 512,000 points for all traces.
dacAdvanceSetVoltDC	Set up a DAC channel to output DC voltage.
dacAdvanceSetVoltWaveform	This action will apply the specified trace to the output channels selected and enable the channels. **Defaults:** • Clock: Internal • Trigger: Immediate • # Cycles: Infinite To change these defaults or to avoid enabling the channels use the ConfigureWaveform action.
dacAdvanceStandardTrace	Create trace data from a standard waveform: Ramp, Sine, Square or Triangle. Points are generated between +1.0 and -1.0. Minimum 8 points per trace. Total trace data storage for the DAC is 512,000 points for all traces.
dacAdvanceStartWaveform	This action will start the waveforms on the specified channels. The waveforms must have already been configured with the dacAdvanceConfigWaveform action.
dacConnect	**Low-Level Action:** Immediately connects the output relays on the selected Agilent E1418 DAC (Agilent E1418 only). **Usage:** Call this action to close the output relays to connect the DAC. **Notes:** To use channel 2, change the "dac" parameter to "@hwconfig.dac2", channel 3 to ""@hwconfig.dac3", and so on.

Table 4-3. dac (D/A Converter) Actions (continued)

Actions	Descriptions
dacConfSourceDCI	**Low-Level Action:** Configures the Agilent E1418 DAC to output the specified current (Agilent E1418 only). **Usage:** 1. Add the appropriate values into the parameters and then call this action. 2. Call any other configuration action(s). 3. Call "dacSet" to sent the setup information to the DAC. 4. Call "dacIsSet" to wait until the DAC is ready for output. 5. Call "dacConnect" to start the output. **Requires:** "dacSet" after this action. **Notes:** To use channel 2, change the "dac" parameter to "@hwconfig.dac2", channel 3 to ""@hwconfig.dac3", and so on.
dacDisconnect	**Low-Level Action:** Immediately disconnects the output relays on the selected Agilent E1418 DAC (Agilent E1418 only). **Usage:** Call this action to open the output relays to disconnect the DAC. **Notes:** To use channel 2, change the "dac" parameter to "@hwconfig.dac2", channel 3 to "@hwconfig.dac3", and so on.
dacConfSourceDCV	**Low-Level Action:** Configures the Agilent E1418 DAC to output the specified voltage (Agilent E1418 only). **Usage:** 1. Add the appropriate values into the parameters and then call this action. 2. Call any other configuration action(s). 3. Call "dacSet" to sent the setup information to the DAC. 4. Call "dacIsSet" to wait until the DAC is ready for output. 5. Call "dacConnect" to start the output. Requires: "dacSet" after this action. **Notes:** To use channel 2, change the "dac" parameter to "@hwconfig.dac2".
dacErrorMessage	**Low-Level Action:** Takes an error code and returns the error string.

Table 4-3. dac (D/A Converter) Actions (continued)

Actions	Descriptions
dacIsSet	**Low-Level Action:** Waits until the Agilent E1418/E1328 DAC is ready for output. (Action returns a true when ready.) **Usage:** 1. Call any configuration action(s) to set up the DAC. 2. Call "dacSet" to send the setup information to the DAC. 3. Call this action to allow the DAC sufficient time for configuration to complete. 4. Call "dacConnect" to start the output. **Notes:** 1. After a reset, the E1418 dac trigger mode is set to "hold". This action will hang until the dac is triggered (which can not happen if the mode is "hold"!) Call dacConfTrigIn to set the trigger source appropriately before calling this action. 2. To use channel 2, change the "dac" parameter to "@hwconfig.dac2", channel 3 to ""@hwconfig.dac3", and so on.
dacReset	**Low-Level Action:** Resets all channels of the Agilent E1418/E1328 DAC to the power-on state (0 V output) and disconnects the output relay. **Usage:** Call this action when the DAC configuration is unknown and the DAC is to be configured into a known state. Instrument: DAC
dacSet	**Low-Level Action:** Sends current setup information to the Agilent E1418 DAC (Agilent E1418 only). **Usage:** 1. Call any configuration action(s) to set up the DAC. 2. Call this action to sent the setup information to the DAC. 3. Call "dacIsSet" to wait until the DAC is ready for output. 4. Call "dacConnect" to start the output. **Requires:** "dacIsSet" after this action. **Notes:** 1. This action only sends configuration information to the DAC and does not start operation. 2. To use channel 2, change the "dac" parameter to "@hwconfig.dac2", channel 3 to ""@hwconfig.dac3", and so on.
dacSetDCVSU	**Low-Level Action:** Program a DAC channel on the Agilent E6198 switch/load unit to a desired voltage. DAC voltage is computed as follows: dacV=(UserVoltage + offset)/gain
dacSetGainOffsetSU	**Low-Level Action:** Change the gain/offset values used to set the DAC voltage in the switch/load unit. This is especially useful for attaching a voltage-programmable power supply to the DAC output. DAC voltage is computed as follows: dacV=(UserVoltage + offset)/gain

Table 4-3. dac (D/A Converter) Actions (continued)

Actions	Descriptions
dacSetGainSU	**Low-Level Action:** Change the gain value used to set the DAC voltage in the switch/load unit. This is especially useful for attaching a voltage-programmable power supply to the DAC output. DAC voltage is computed as follows: dacV=(UserVoltage + offset)/gain
dacSetOffsetSU	**Low-Level Action:** Change the offset value used to set the DAC voltage in the switch/load unit. This is especially useful for attaching a voltage-programmable power supply to the DAC output. DAC voltage is computed as follows: dacV=(UserVoltage + offset)/gain
dacSetPSVSU	**Low-Level Action:** Program a DAC channel voltage on the Agilent E6198 switch unit using the gain value in the module parameter block.
dacSetTimeout	**Low-Level Action:** Sets the Agilent E1328 or E1418 Digital to Analog Converter (DAC) operation time out value (in milliseconds). **Notes:** All channels are set to the same time-out value.
dacSetDCI	**Low-Level Action:** Sets the Agilent E1418/E1328 DAC to output the specified current; causes an immediate output. **Usage:** This action is self contained; there is no need to call other actions. **Notes:** To use channel 2, change the "dac" parameter to "@hwconfig.dac2", channel 3 to "@hwconfig.dac3", and so on.
dacSetDCV	**Low-Level Action:** Sets the Agilent E1418/E1328 DAC to output the specified voltage; causes an immediate output. **Usage:** This action is self contained; there is no need to call other actions. **Notes:** To use channel 2, change the "dac" parameter to "@hwconfig.dac2", channel 3 to "@hwconfig.dac3", and so on.
dacConfTrigIn	**Low-Level Action:** Selects the Agilent E1418 DAC's trigger source (Agilent E1418 only) for all channels. **Usage:** 1. Add the appropriate values into the parameters and then call this action. 2. Call any other configuration action(s). 3. Call "dacSet" to sent the setup information to the DAC. 4. Call "dacIsSet" to wait until the DAC is ready for output. 5. Call "dacConnect" to start the output. **Requires:** "dacSet" after this action. **Notes:** To use channel 2, change the "dac" parameter to "@hwconfig.dac2", channel 3 to "@hwconfig.dac3", and so on.

Table 4-3. dac (D/A Converter) Actions (continued)

Actions	Descriptions
dacSelftest	**Low-Level Action:** Runs self-test on the Agilent E1418/E1328 DAC; returns a pass/fail indication. **Notes:** This action tests all channels; call using "@hwconfig.dac1".
m9185dacGetCardInfo	This action retrieves the DAC card information. This action should be used after initializing the card. The action will retrieve the following information: • (1) Slot Number • (2) Channels detected • (3) Board version • (4) FPGA version • (5) Fundamental version • (6) Serial number • (7) Last calibration date
m9185dacGetOutput	This action returns the voltage/current output level for a single channel.
m9185dacGetRelay	This action returns the output relay for a single channel.
m9185dacReset	This action resets the DAC card.
m9185dacSetOut1Ch	This action sets the voltage/current output for a single channel.
m9185dacSetOut8Ch	This action sets the voltage/current output for 8 channels.
m9185dacSetOut16Ch	This action sets the voltage/current output for 16 channels.
m9185dacSetRelay1Ch	This action turns on/off the output relay for a single channel.
m9185dacSetRelay8Ch	This action turns on/off the output relay for 8 channels simultaneously.
m9185dacSetRelay16Ch	This action turns on/off the output relay for 16 channels simultaneously.
m9185dacSetTrig	This action sets the trigger configuration of the DAC card: • (1) Select trigger type • (2) Output pulse edge • (3) Output pulse width • (4) Input pulse edge • (5) PXI trigger line

Table 4-4. daq (Digital Acquisition) Actions

Actions	Descriptions
daqIvIConfMeasurement	daqIvIConfMeasurement: • Function #1 To set the Daq Module to Single Channel or Multiple Channel mode. • Function #2 To set the Daq Module to Immediate (acquire data immediately) or Channel (Waiting for valid trigger). Measurement mode: • 0: Single • 1: Multiple Trigger mode: • 0: Channel Mode • 1: Immediate Mode
daqIvIGetCardInfo	This action retrieves the card information. This action should be used after initializing the card. The action will retrieve the following information: • (1) Slot No. • (2) Serial No. • (3) PCBVersion • (4) FPGAVersion • (5) Driver Version • (6) Cal Date
daqIvIMultipleConfAllInputPort	This function allow user to configure all the input ports in DAQ in multiple mode. Each input port can have a unique configuration.
daqIvIMultipleConfInputPort	This function allow user to configure specific input port. Configure each input port before perform measurement.
daqIvIMultipleDCVAvg	This function initiates data acquisition in multiple channel mode and compute the DCV average value from each channel data.
daqIvIMultipleGetMeasureResult	This function measures selected input port voltage while running daqPseudoFindGlitch32Ports action.
daqIvIMultipleGlitchPassFailResult	This functions read back the glitch detection result while running daqPseudoFindGlitch32Ports action.
daqIvIMultipleMinMax	This function initiates data acquisition in multiple channel mode and compute the DCV Min/Max value from each channel data.
daqIvIMultipleModeFindGlitch8Ports	This function performs glitch detection voltage monitoring on 8 input ports simultaneously. Besides that, user can choose to use this function as DCV measurement by setting its PortFunction parameter. For example: 4 ports set as glitch detection, 4 ports set as DCV measurement.
daqIvIMultipleWave	This function initiates data acquisition in multi channel mode and presents the each channel data in waveform.
daqIvIPseudoFindGlitch32Ports	This function performs glitch detection voltage monitoring on each group of input ports sequentially. This functions also can specific the input ports as DCV measurement by setting its PortFunction parameter.

Table 4-4. daq (Digital Acquisition) Actions (continued)

Actions	Descriptions
daqIvIReset	This function resets the M9216. All ports and properties will be set to the default values. A UpDateHardware command will be sent. This method needs to be called once when the M9216A is first started up.
daqIviSingleConfInputPort	This function allow user to configure specific input port in single channel mode. The configured input port will be the active port. **Notes:** 1. Configure input port before perform any measurement. 2. Use daqIvIConfMeasurementMode to set Daq Module in Single Mode before use this action.
daqIvISingleDCVAvg	This function initiates data acquisition in single channel mode and compute the DCV average value from the data.
daqIvISingleMinMax	This function initiates data acquisition in single channel mode and compute the DCV Min/Max value from the data.
daqIvISingleWave	This function initiates data acquisition in single channel mode and presents the data in waveform.
daqIvISwitchAllBanksAux	Set input port of the Channel 1,2,3,4,5,6,7 and Channel 8 to auxiliary output port of the DAQ Module.
daqIvISwitchBank1Aux	Set input port of the Channel 1 and Channel 2 to auxiliary output port of the DAQ Module.
daqIvISwitchBank2Aux	Set input port of the Channel 3 and Channel 4 to auxiliary output port of the DAQ Module.
daqIvISwitchBank3Aux	Set input port of the Channel 5 and Channel 6 to auxiliary output port of the DAQ Module.
daqIvISwitchBank4Aux	Set input port of the Channel 5 and Channel 6 to auxiliary output port of the DAQ Module.
daqIvISwitchInputPortToAux	Set input port to auxiliary output port of the DAQ Module.

Table 4-5. dgn (Diagnostic) Actions

Actions	Descriptions
abeSetDisplay	This action enable user to set 34980 abe display to either on or off.
Ag3499SlotCheck	Verifies that the 3499 contains the N2261, N2262, and N2264 cards in one of the 5 slots
ArbAdjust	**AutoAdjust Action:** Measures Voltage/Current (V/I) gains using the MCM and DMM on external Arb connections to the system. Writes the gain values to the system registry. The referenced arb must be connected to the standard ("Inst5") port on the E6171 (MCM) board. **Usage:** Set the "line_frequency" parameter to either 50 or 60 (50 Hz or 60 Hz). Set the "verbose" parameter to 1 to display intermediate results. A non-zero result indicates how many tests failed. **Notes:** This action is part of the "DGN5450.TPA" and "DGN5430.TPA" testplans and are best used in that environment. Example: DGN5450.TPA or DGN5430.TPA. **See Also:** AutoAdjust **Instrument:** MCM
EFTMuxRelayTest1	Tests bus 1, 2 and UUTCOM matrix relays for opens and shorts in the Eft PinCard.
EFTMuxRelayTest2	Tests bus 3 and 4 matrix relays for opens and shorts in the Eft PinCard.
dgnConnectString	Connects the switch path referenced by the parameter 'pathstring'. This action is helpful when the path you wish to connect may be a variable or a symbol table entry.
dgnDisConnect	Disconnects the switch path referenced by the parameter 'path'. This action is helpful when the path you wish to disconnect may be a variable or a symbol table entry.
EFTMuxTestPath12	This action test the Series 2 instrument mux relays for ABus1 and ABus2.
EFTMuxTestPath34	This action test the Series 2 instrument mux relays for ABus3 and ABus4.
EFTMuxTestDebugPorts	This action tests the Series 2 debug ports by prompting the user to put a 50 ohm resistor on each port one-at-a-time.
fetchDeviceByName	This action returns the module handle of the module referenced by string name.
getICA	This action returns: • 0 - no ICA defined • 1 - Mass Interconnect (MAC panel) ICA • 2 - Test System Interface (Express connect) ICA • –1 - error condition **See Also:** getModByName, getModuleName, isThereA33120, isThereA34401, isThereA53131, whichDmm

Table 4-5. dgn (Diagnostic) Actions (continued)

Actions	Descriptions
GetICASlotFromAlias	To obtain ICA slot from the module's Alias name.
GetICASlotFromWire	To obtain ICA slot from the module's wire name which is attached to other/its module's Alias.
getLoadBoxModuleName	This action gets Load Box's handle from System Topolgy file.
getLoadBoxSlotModuleName	This action gets Load Card/Pin Matrix's handle from System Topology file.
getModByModuleName	This action gets module handle with the input of the module name.
getModByName	This action returns the module handle of the module specified by the device ID in modName.
getModuleName	This action returns the string name of a module.
getMultipleModByName	This action returns the multiple module name of the module specified by the device ID.
GetMux_port_position	Verifies that the 3499 contains the N2261, N2262, and N2264 cards in one of the 5 slots.
GetSUTopologyInfo	This action returns the overall SLU information from the System Topology file.
isThereA33120	This action returns a value of 1 if there is a 33120 Arbitrary Waveform Generator in the system.ust file and is enabled.
isThereA33220	This action returns a value of 1 if there is a 33220 Arbitrary Waveform Generator in the system.ust file and is enabled.
isThereA34401	This action returns a value of 1 if a 34401 DMM is in the system.ust file and is enabled.
isThereA53131	This action returns a value of 1 if a 53131 Counter is in the system.ust file and is enabled.
isThereDSO6054L	This action returns a value of 1 if there is a 6054L Digital Oscilloscope in the system.ust file and is enabled.
MCMSafetyInterlock	Diagnostics action to enable or disable the MCM safety interlock.
PinProtBypass	Closes relays to short the 200 ohm column protection resistors on the Pin Matrix Module.
PinAuxRelayTests	Tests the "Aux" relays for opens and shorts on the Pin Matrix Module. Call this action to see if the "Aux" relays on the Pin Matrix Module open and close correctly.
PinAuxRelayTestsAll	Tests the "Aux" relays for opens and shorts on all of the Pin Matrix Modules in the system.
PinRelayOpenTest	Tests the Pin Matrix Module for shorted relays.
PinRelayOpenTestAll	Tests all of the Pin Matrix Modules for stuck open relays.
PinRelayPathTest	Tests the Pin Matrix Module for stuck open relays.
PinRelayPathTestAll	Tests all the Pin Matrix Modules for stuck open relays.
PinShortsTest	Test for shorts between pins on the Pin Matrix Module.
PinShortsTestAll	Test for shorts between pins on all of the Pin Matrix Modules in the system.
PinReset	Reset Pin Matrix Module relays.
SYS_GetMemory	Retrieve information about computer memory.

Table 4-5. dgn (Diagnostic) Actions (continued)

Actions	Descriptions
AutoAdjust	Measures all Voltage/Current (V/I) gains and offsets using the MCM and DMM. Writes the gain and offset values to the system registry.
VI Debug Port Test	Tests debug ports 1-4 of the MCM using the DMM.
VI HV DC Accy	Tests HV DC source accuracy of the MCM using the DMM. Action is typically used to test the High Voltage DC accuracy of the MCM, after using the AutoAdjust Action.
VI LV DC Accy	Tests LV DC source accuracy of the MCM using the DMM. Action is typically used to test the High Voltage DC accuracy of the MCM, after using the AutoAdjust Action.
VIRelayTest1	Tests bus 1, 2 and UUTCOM matrix relays for opens and shorts in the MCM. Use this action to test the ABus1, ABus2, and UUT Common matrix relays in the MCM.
VIRelayTest2	Tests bus 3 and 4 matrix relays for opens and shorts in the MCM. Use this action to test the ABus3 and ABus4 matrix relays in the MCM.
whichDmm	This action returns the system DMM as follows: • 0 - No system DMM found • 1 - E1411 with MCM • 2 - 34401 with E8792 / E8782 mux • 3 - 34980A DMM • 4 - 34411A

Table 4-6. digitizer Actions

Actions	Descriptions
adcConfArm	**Low-Level Action:** Configures the arm subsystem of the Agilent E1563, E1564, or E1429 Digitizers. **Usage:** 1. Add the appropriate values into the parameters and then call this action. 2. Call any other configuration action(s). 3. Call "adcIsSet" to wait until the Digitizer is ready to make measurements. 4. Call "adcInitiate" to start the measurement cycle after a trigger or call "adcStart" to trigger the Digitizer and start the measurement cycle. 5. Read the Digitizer. **Requires:** "adcStart" or "adcInitiate" after this action. **Notes:** This setup is common to both channels of the digitizers; sending to either channel sets both channels.
adcClear	**Low-Level Action:** Clears output buffer on digitizer (ADC) and aborts reading, if in progress.
ADC_DCV-Avg	**High-Level Action:** Returns average DC voltage from the Agilent E1429 Digitizer in a window defined by the "start" parameter and "stop" parameter time values from the trigger point. **Usage:** 1. Call the "adcConfInControls" Action to set up the Digitizer's input (only needed to change filter and input impedance, if Digitizer is not set to the differential mode). 2. Call "ADC_Config_1_Chan" to configure the Digitizer. 3. Call this action to read the voltage. **Notes:** 1. If pre-sampling is in effect and start and stop straddle the trigger point, the average may be skewed because of the different rates of pre and post sampling. 2. To store the data into a file, set the "store" parameter to "1". A file name and path must be stored in the "filename" parameter to store the readings. 3. Sample rate ("samplerate" parameter) must match the Digitizer's configured rate (see "adcConfInControls" Action). 4. To use channel 2, change the "adc" parameter to "adc2".
adcFastOff	**Low-Level Action:** This action shuts off the Agilent1429 Digitizer's fast fetch mode until it is re-opened. This is provided for those cases where processor timing interferes with the digitizer. **Notes:** For more information see the hpe1429_disableHighSpeed function in the hp1429 VXI-PNP driver help file.

Table 4-6. digitizer Actions (continued)

Actions	Descriptions
adcConfFreq	**Low-Level Action:** Configures the frequency subsystem of the Agilent E1563, E1564, or E1429 Digitizer (ADC). **Usage:** 1. Add the appropriate values into the parameters and then call this action. Call any other configuration action(s). 2. Call "adcIsSet" to wait until the Digitizer is ready to make measurements. 3. Call "adcInitiate" to start the measurement cycle after a trigger or call "adcStart" to trigger the Digitizer and start the measurement cycle. 4. Read the Digitizer. **Requires:** "adcStart" or "adcInitiate" after this action. **Notes:** This setup is common to all channels of the digitizers; sending to either channel sets both channels.
ADC_Config_1_Chan	**High-Level Action:** Configures Agilent E1563, E1564, or E1429 for capturing a waveform for later analysis on channel **Usage:** 1. Call the "adcConfInControls" Action to set up the Digitizer's input (only needed to change filter and input impedance, if Digitizer is not set to the differential mode). 2. Call this action to configure the Digitizer. 3. Call ADC_Transform, ADC_Min_Max, ADC_Min_Max_(Win), ADC_DCV-Avg, ADC_Transform, or ADC_DCV to make the measurement. **Notes:** 1. Triggering is assumed to be based on an input level. 2. Clock rate affects all channels.
adcGetSampleFreq	**Low-Level Action:** Returns the sample rate values from the most recent setup of the Agilent E1429 Digitizer. **Usage:** Use to adjust sample analysis for the actual frequency used for sampling. **Notes:** This action can only set valid frequency values.
adcGetSampleInfo	**Low-Level Action:** Returns sample rate and data point count values from the most recent setup of the Agilent E1429 Digitizer (ADC). **Usage:** Use to adjust digitized sample analysis. **Notes:** Sample rate information is useful to determine exactly when a sample was taken. Sample count information is useful to make sure that the analysis is restricted to real sample data returned in the "results" array.

Table 4-6. digitizer Actions (continued)

Actions	Descriptions
ADC_DCV	**High-Level Action:** Returns instantaneous DC voltage from the Agilent Technologies E1563, E1564, or E1429 Digitizer using the "offset" parameter time from the trigger point. **Usage:** 1. Call the "adcConfInControls" Action to set up the Digitizer's input (only needed to change filter and input impedance, if Digitizer is not set to the differential mode). 2. Call "ADC_Config_1_Chan" to configure the Digitizer. 3. Call this action to measure the voltage. **Notes:** 1. To store the data into a file, set the "store" parameter to "1". A file name and path must be stored in the "filename" parameter to store the readings. 2. Sample rate ("samplerate" parameter) must match the Digitizer's configured rate (see "adcConfInControls" Action). 3. To use channel 2, change the "adc" parameter to "adc2".
adcConfInControls	**Low-Level Action:** Configures the input circuit of the Agilent E1563, E1564, or E1429 Digitizer (ADC). **Usage:** 1. Add the appropriate values into the parameters and then call this action. 2. Call any other configuration action(s). 3. Call "adcIsSet" to wait until the Digitizer is ready to make measurements. 4. Call "adcInitiate" to start the measurement cycle after a trigger or call "adcStart" to trigger the Digitizer and start the measurement cycle. 5. Read the Digitizer. **Requires:** "adcStart" or "adcInitiate" after this action. **Notes:** To use channel 2, change the "adc" parameter to "@hwconfig.adc2".
adcInitiate	**Low-Level Action:** Arms the Agilent E1563, E1564, or E1429 Digitizer (ADC) and retrieves the data after triggering occurs. Data can be used by any routine which allows use of previously obtained data, such as: ADC_Analyze_Wave, ADC_DCV, ADC_DCV-Avg, ADC_Min_Max, ADC_Min_Max_(Win), ADC_Transform. **Usage:** 1. Call the "adcConfInControls", "adcConfFreq", and/or "adcConfArm", and "adcIsSet" Actions to configure the Digitizer. 2. Call this action to execute a measurement after a trigger. 3. Call an action to analyze the data. **Notes:** To use channel 2, change the "adc" parameter to "adc2".

Table 4-6. digitizer Actions (continued)

Actions	Descriptions
adcIsSet	**Low-Level Action:** Waits until the Agilent E1563, E1564, or E1429 Digitizer (ADC) is ready for measurement. (Action returns a true when ready.) **Usage:** Call this action after setting up the Digitizer using the "adcConfInControls", "adcConfFreq", and/or "adcConfArm" Actions (in a loop if called from a C program). This allows the Digitizer sufficient time for the configuration to complete. **Notes:** To use channel 2, change the "adc" parameter to "@hwconfig.adc2".
adcLastReading	**Low-Level Action:** Transfers the current digitizer data from the Agilent E1563, E1564, or E1429 from memory into a waveform data type to allow easy viewing. **Usage:** Call this action after the digitizer has taken a reading. It can be placed after each reading and the data viewed for each measurement.
ADC_Min_Max_(Win)	**High-Level Action:** Returns Vmin or Vmax DC voltage values from the Agilent E1563, E1564 or E1429 Digitizer in a window defined by the "start" parameter and "stop" parameter times from the trigger point. **Usage:** 1. Call the "adcConfInControls" Action to set up the Digitizer's input (only needed to change filter and input impedance, if Digitizer is not set to the differential mode). 2. Call "ADC_Config_1_Chan" to configure the Digitizer. 3. Call this action to measure the voltages. **Notes:** 1. To store the data into a file, set the "store" parameter to "1". A file name and path must be stored in the "filename" parameter to store the readings. 2. Sample rate ("samplerate" parameter) must match the Digitizer's configured rate (see "adcConfInControls" Action). 3. To use channel 2, change the "adc" parameter to "adc2".

Table 4-6. digitizer Actions (continued)

Actions	Descriptions
ADC_Min_Max	**High-Level Action:** Measures a waveform by the Agilent E1563, E1564, or E1429 Digitizer and directly analyzes it for Vmin & Vmax values. **Usage:** 1. Call the "adcConfInControls" Action to set up the Digitizer's input (only needed to change filter and input impedance, if Digitizer is not set to the differential mode). 2. Call "ADC_Config_1_Chan" to configure the Digitizer. 3. Call this action to measure the waveform. **Notes:** 1. When presampling is used, it may be necessary to use offset to avoid the presampled area. 2. To store the data into a file, set the "store" parameter to "1". A file name and path must be stored in the "filename" parameter to store the readings. 3. Sample rate ("samplerate" parameter) must match the Digitizer's configured rate (or be set to 0). 4. To use channel 2, change the "adc" parameter to "adc2". 5. The "level" parameter is NOT USED.
adcReset	**Low-Level Action:** Resets the digitizer (ADC) to its power-on state for all channels. **Usage:** Call this action when the Digitizer configuration is unknown and the Digitizer is to be configured into a known state.
adcSetReturnMode	**Low-Level Action:** This action allows time-outs on the specified Agilent E1563, E1564 or E1429 Digitizer (ADC) to raise an exception when encountered. **Notes:** The default is not to raise an exception with time-outs.
adcSetTimeout	**Low-Level Action:** Sets the Agilent E1563, E1564 or E1429 Digitizer (ADC) operation time out value (in milliseconds). **Notes:** All channels are set to the same time-out value. If a measurement requires a duration longer than the current time-out setting, it will bump up the value to be sample time plus 1 ms.

Table 4-6. digitizer Actions (continued)

Actions	Descriptions
adcStart	**Low-Level Action:** Arms the Agilent E1429 Digitizer (ADC) for the configured trigger event and returns. THIS ROUTINE DOES NOT WAIT FOR THE TRIGGER TO OCCUR AND DOES NOT FETCH DATA FROM THE INSTRUMENT! **Usage:** Call the "adcConfInControls", "adcConfFreq", and/or "adcConfArm", and "adcIsSet" Actions to configure the Digitizer. Then call this action to trigger the Digitizer and start the measurement cycle. **Requires:** Should follow an "adcIsSet" action or ADC_Config_1_Chan. In order to retrieve the data from the device one of the following need to be executed: • adcInitiate • ADC_Analyze_Wave • ADC_DCV • ADC_DCV-Avg • ADC_Min_Max • ADC_Min_Max_(Win) • ADC_Transform **Notes:** Use this action to arm the Digitizer without waiting for the triggered event and subsequent data action to occur.
adcSelfTest	**Low-Level Action:** Runs a selftest on the Agilent E1429 Digitizer (ADC) that is provided by its Plug and Play driver. **Notes:** ADC is left in its reset state after execution of this command.
adcSetTimeoutException	**Low-Level Action:** This action allows enabling/disabling exception response to a time-out on the specified Agilent E1563, E1564 or E1429 digitizer (ADC). **Notes:** The default is not to raise an exception with time-outs. **Instrument:** ADC
adcType	**Low-Level Action:** Returns the type of digitizer (ADC) installed: either E1429, E1563, or E156

Table 4-6. digitizer Actions (continued)

Actions	Descriptions
ADC_Analyze_Wave	**High-Level Action:** Measures a waveform using the Agilent E1563, E1564, or 1429 Digitizer to analyze it for Vmin, Vmax, High pulse width, Low pulse width and period values. Uses transitions from the "level" parameter to find edges and measures times from edge to edge. Measures Vmax and Vmin at the "offset" parameter's sample count after edges. **Usage:** 1. Call the "adcConfInControls" Action to set up the Digitizer's input (only needed to change filter and input impedance, if Digitizer is not set to the differential mode). 2. Call "ADC_Config_1_Chan" to configure the Digitizer. 3. Call this action to measure the waveform. **Notes:** 1. E1563 or E1564 uses presample count = 1 2. E1429 uses presample count = 0 3. To store the data into a file, set the "store" parameter to "1". A file name and path must be stored in the "filename" parameter to store the readings. 4. Sample rate ("samplerate" parameter) must match the Digitizer's configured rate (see "adcConfInControls" Action). 5. To use channel 2, change the "adc" parameter to "adc2".
ADC_Transform	**High-Level Action:** Transform Agilent E1563, E1564, or E1429 Digitizer returned data with Measurement Control Module attenuator gain and offset terms. This action converts Digitizer readings through the attenuator to the values at the input of the attenuator. **Usage:** 1. Call the "adcConfInControls" Action to set up the Digitizer's input (only needed to change filter and input impedance, if Digitizer is not set to the differential mode). 2. Call "ADC_Config_1_Chan" to configure the Digitizer. 3. Call this action to read the Digitizer. **Notes:** 1. To store the data into a file, set the "store" parameter to "1". A file name and path must be stored in the "filename" parameter to store the readings. 2. Sample rate ("samplerate" parameter) must match the Digitizer's configured rate (see "adcConfInControls" Action). 3. To use channel 2, change the "adc" parameter to "adc2".
L453xA_Abort	This command aborts lengthy operations, such as a data acquisition in progress.
L453xA_AcqWave	Acquire Digitizer Waveform Trace.
L453xA_AcqWave2	Acquire Digitizer Waveform Trace (Using Waveform Parameter for the output).
L453xA_AnalyseGetVoltage	This action is used to calculates the average value of a waveform after the detection of a definable trigger point and delay.
L453xA_AnalysePeakPulse	This action is used to search for a peak voltage of a given pulse waveform after a definable delay time. It also calculates and returns the time of the definable edge transitions.

Table 4-6. digitizer Actions (continued)

Actions	Descriptions
L453xA_AnalyseWave	This action is used to analyze a given waveform (obtained with instrument) for Vmin, Vmax, high pulse width, low pulse width and period values. Measure high pulse width and low pulse width after definable start-time from threshold parameter to find edges and measures period time from edge to edge. Measures Vmin and Vmax at delay parameter over average time parameter.
L453xA_ConfAcq	Config Acquisition (Low Level).
L453xA_ConfArmTimer	This command specifies the time interval used where the ARM source must be set to TIMER before using this action. The timer causes the arm to delay for <time> seconds each time the digitizer returns to the state where it is waiting for an arm.
L453xA_ConfArmTrig	This command configures the source for arming & triggering the digitizer.
L453xA_ConfChannel	Config L453xA Channel Number and Vertical Range.
L453xA_ConfExtTrigInOut	Configures the slope of the EXTernal trigger BNC signal (Trig In/Out) for either ARM or TRIGger. You can configure the slope to be either a POSitive (rising) or NEGative (falling) edge.
L453xA_ConfTrigChanEdge	This command configures EDGE trigger attributes for the channels in the <chan_list>. To enable a configured channel trigger to generate an actual digitizer trigger, you must use L453xA_ConfArmTrig to configure CHANnel (or OR) as the trigger source.
L453xA_ConfTrigChanOff	Channel triggering based on channel-level events. To turn channel triggering off, use this command.
L453xA_ConfTrigChanWin	This command configures window trigger attributes for the specified channel list. A window trigger specifies a voltage region within the channel's voltage range where a trigger is generated when the signal enters or leaves the window boundaries.
L453xA_ConfTrigTimestamp	This command specifies whether the digitizer's timestamp counter runs continuously or is cleared at each INITiate command.
L453xA_Init	This action is use to calculates the time offset between the trigger event of waveform 1 and the second trigger event of waveform 2.
L453xA_MeasDutyCycle	This query returns the duty cycle, starting at the midpoint of a falling or rising edge.
L453xA_MeasFreq	This query returns the frequency (in Hz) of the given cycle, starting at the midpoint of a falling or rising edge. The frequency of a signal is defined as the reciprocal of the period (1 / period). For the MEASure:FALL:FREQuency[VERBose]? query, the period is defined as the time from the midpoint of the specified falling edge to the midpoint of the next falling edge.
L453xA_MeasOvershoot	This query returns the falling or rising overshoot as a percentage.
L453xA_MeasPeriod	This query returns the period of the given cycle (in seconds).
L453xA_MeasPreShoot	This query returns the falling or rising preshoot overshoot as a percentage.
L453xA_MeasPulseWidth	This query returns the pulse width (in seconds) of the given cycle, starting at the midpoint of a rising/falling edge.

Table 4-6. digitizer Actions (continued)

Actions	Descriptions
L453xA_MeasTime	This query measures and returns the fall time value of the falling or rising edge closest to the trigger reference. Fall time is defined as the time in seconds from the upper threshold to the lower threshold of a falling edge. Rise time is defined as the time in seconds from the lower threshold to the upper threshold of a rising edge.
L453xA_MeasVamplitude	This query returns the amplitude for the specified <channel>, as assessed by the histogram method or the absolute method.
L453xA_MeasVbase	This query returns the base voltage used for cycle based measurements. This value forms the lower voltage of an amplitude measurement. Base may be evaluated using the STANdard (histogram) method or the ABSolute method.
L453xA_MeasVmax	This query returns the maximum voltage for the specified record and channel.
L453xA_MeasVmin	This query returns the minimum voltage on the specified channel and record number.
L453xA_MeasVpp	This query returns the peak-to-peak voltage for the specified record on the specified channel.
L453xA_MeasVrms	This query returns the root mean square (RMS) average voltage for either a cycle or a measurement window.
L453xA_MeasVthreshold	This query returns the base voltage used for cycle based measurements. This value forms the lower voltage of an amplitude measurement. Base may be evaluated using the STANdard (histogram) method or the ABSolute method.
L453xA_MeasVtop	This query returns the top voltage used for cycle based measurements. This value forms the higher voltage of an amplitude measurement. Top may be evaluated using the STANdard (histogram) method or the ABSolute method.
L453xA_QueryCmd	Send a query command to the ADC.
L453xA_Reset	This command resets the instrument to the Factory configuration.
L453xA_ScaleWaveform	This action will scales a given waveform base on the gain and offset. The whole waveform sample data in the memory will be modified by Gain and Offset factors.
L453xA_SelfTest	This command performs a complete self-test of the instrument and returns a pass/fail indication. The self-test runs a series of tests and will take more than 45 seconds to complete. If all tests pass, you can have a high confidence that the instrument is operational.
L453xA_SendCmd	Send command to L453xA.
L453xA_SetMeasThresholdAbs	Configures the absolute threshold level for the given channels.
L453xA_SetMeasThresholdPercent	Configures the percentage threshold level for the given channels.
L453xA_SetMeasThresholdTopBase	Configures the Top-Base method for the given channels. If <Method> ABSOLUTE is selected, vTop and Base value will be used to configure the absolute Top and Base threshold levels for the specified channels.
L453xA_SetMeasWin	Sets the measurement window in which on-board measurements are performed.
L453xA_SetTimeOut	Sets the operation time out value (in milliseconds).

Table 4-6. digitizer Actions (continued)

Actions	Descriptions
PXD731x_ErrorQuery	This action queries the last error queue in the device and returns extended error information.
PXD731x_ForceTrigger	This action generates a software trigger on the specified channel. This is a force trigger independent from the trigger configuration.
PXD731x_GetConfiguration	This action returns the configuration setting of the device.
PXD731x_GetData	This action returns the sampled data of a specified channel upon completion of data acquisition. The sampled data can be stored in a waveform buffer or a text file (.txt).
PXD731x_GetDCVoltage	This action measures DC voltage and returns the measured value.
PXD731x_GetDeviceInfo	This action retrieves the information of PXD731x Device: 1. Driver revision 2. Instrument revision (Firmware revision) 3. Serial Number 4. Calibration Date (Channel dependent) 5. Calibration Status (Channel dependent)
PXD731x_GetFreqMeasConfig	This action returns the frequency measurement configuration of the device.
PXD731x_GetFreqMeasResult	This action returns the result of the frequency measurement. Use frequencybyper-result for frequencies less than 10 kHz and frequencybycnt-result for frequencies greater than 10 kHz. This may result in a more precise measurement for frequencies in each range.
PXD731x_GetTimeMeasConfig	This action returns the rise/falltime measurement configuration.
PXD731x_GetTimeMeasResult	This action returns the result of the rise/falltime measurement.
PXD731x_Reset	This action returns the device to its default state.
PXD731x_SelfTest	This action commands the device to perform self test.
PXD731x_SetChannelStartStop	This action will start/stop the waveform sampling.
PXD731x_SetChannelTriggerState	This action enables/disables the trigger engine of the specified channel.
PXD731x_SetFreqMeasConfig	This action sets the frequency measurement configuration of the device.
PXD731x_SetFreqMeasStartStop	This action will start/stop the frequency measurement
PXD731x_SetInputConfig	This action configures the digitizer input based on the specified range, impedance, and filter.
PXD731x_SetSamplingConfig	This action configures the digitizer sampling based on the specified interval, pre-samples and post samples.
PXD731x_SetTimeMeasConfig	This action sets the rise/falltime measurement configuration.
PXD731x_SetTimeMeasStartStop	This action will start/stop the time measurement
PXD731x_SetTriggerConfig	This action specifies the trigger configuration of the digitizer.

Table 4-6. digitizer Actions (continued)

Actions	Descriptions
PXD731x_GetData	This functions returns the sampled data of a channel when the channel is ready. The sampled data can be stored in a waveform buffer or in a text (.txt) file.
AnalyseAddInstrument	This action adds a new device to the list of device/instruments supported by AnalyseWave.dll actions. The supported device list is located at: [TS5000 Directory]\Config\AnalyseWaveform.csv **Note:** To use actions from AnalyseWave.dll to process data acquired by a device, the device hardware handler must comply with specific requirements. Please refer example in [TS5000 Directory]\samples\AnalyseWaveSample
AnalyseCheckInstrument	This action checks the compatibility of AnalyseWaveform actions with the instrument's hardware handler by checking the supported device list. The supported device list is located at: [TS5000 Directory]\Config\AnalyseWaveform.csv **Note:** 1. To use this action, the device must already been added to the list of device/instrument supported by AnalyseWave.dll. Refer to action AnalyseAddInstrument. 2. To use actions from AnalyseWave.dll to process data acquired by a device, the device hardware handler must comply with specific requirements. Please refer example in [TS5000 Directory]\samples\AnalyseWaveSample
AnalyseGetAverageVoltage	This action calculates the average voltage of a given waveform upon detection of a specified trigger level point and delay. **Note:** This action works with waveform(s) captured by device(s) supported by AnalyseWave.dll and stored in this dll only. See action AnalyseAddInstrument for more information.
AnalyseGetLevelTime	This action returns the level time (or the time when trigger event happened) for a given waveform. **Note:** This action works with waveform(s) captured by device(s) supported by AnalyseWave.dll and stored in this dll only. See action AnalyseAddInstrument for more information.
AnalyseMeasureDelay	This action calculates the time offset (or delay time) between the trigger event on first channel and the trigger event on second channel **Note:** This action works with waveform(s) captured by device(s) supported by AnalyseWave.dll and stored in this dll only. See action AnalyseAddInstrument for more information.

Table 4-6. digitizer Actions (continued)

Actions	Descriptions
AnalysePeakPulse	This action searches peak voltage of a given pulse waveform upon a specified start time. It also calculates the edge transition time (rising or falling edge). **Note:** This action works with waveform(s) captured by device(s) supported by AnalyseWave.dll and stored in this dll only. See action AnalyseAddInstrument for more information.
AnalyseScaleWaveform	This action scales a given waveform based on the specified gain and offset. The scaled waveform data will overwrite the existing waveform data. Scaled sample = (given sample * ‘Gain’) + 'Offset' **Note:** This action works with waveform(s) captured by device(s) supported by AnalyseWave.dll and stored in this dll only. See action AnalyseAddInstrument for more information.
AnalyseWaveformMinMax	This action analyzes a given waveform and returns the value for Vmin, Vmax, high pulse width, low pulse width and period. • Vmin is the lowest voltage measured on the waveform. • Vmax is the highest voltage measured on the waveform • High Pulse Width is the time between first rising edge and the following falling edge • Low Pulse Width is the first falling edge and the following rising edge. • Period = High Pulse width + Low Pulse Width **Note:** This action works with waveform(s) captured by device(s) supported by AnalyseWave.dll and stored in this dll only. See action AnalyseAddInstrument for more information.

Table 4-7. dio (Digital I/O) Actions

Actions	Descriptions
ADDigitalRead	This actions reads data from the input pins. The PCI-1750 has 16 input pins. This action reads the input pins in 2 groups of eight pins. The first group with pins IDI0 to IDI7 is grouped under Port0. Port1 consists of pins IDI8 to IDI15. Refer to the PCI-1750 User Manual from Advantech for the pin assignment diagram. When there is no input voltage, the pin will be at 'High'. When there is an input the pin will be pulled 'Low'. Each port status is read as a byte. Bit0 -> 1 Bit1 -> 2 Bit2 -> 4 Bit3 -> 8 Bit4 -> 16 Bit5 -> 32 Bit6 -> 64 Bit7 -> 128 **Example:** If Bit0 and Bit7 has input, without considering the mask applied, the input value returned would be 129. The mask is used to mask of unwanted bits, leaving only useful bits for comparison in the result. The final result will be (data AND mask). A mask of 0xFF would mean all bits are considered and a mask of 0x00 would return 0 regardless of input port status.

Table 4-7. dio (Digital I/O) Actions (continued)

Actions	Descriptions
ADDigitalWrite	This actions sets the output pins. The PCI-1750 has 16 output pins. This action sets the output pins in 2 groups of eight pins. The first group with pins IDO0 to IDO7 is grouped under Port0. Port1 consists of pins IDO8 to IDO15. Refer to the PCI-1750 User Manual from Advantech for the pin assignment diagram. When an output pin is set, it will be 'High'. When it is not set, it will be 'Low'. The pins are 'Low' by default and will return to 'Low' when reset. Each port status is set as a byte. Bit0 -> 1 Bit1 -> 2 Bit2 -> 4 Bit3 -> 8 Bit4 -> 16 Bit5 -> 32 Bit6 -> 64 Bit7 -> 128 **Example:** To turn Bit0 and Bit7 on, without considering the mask applied, the output value set should be 129 in decimal, Ox81 and 10000001 in binary. The mask is used to mask on bits which never need to be set. A mask of (255/0xFF/11111111) would allow all bits to be set, a mask of (96/0x60/01100000) would allow Bit5 and Bit6 to be set, and a mask of (0/0x00/00000000) would not turn any bits on regardless of output value set.
Agt34950ConfigureInput	This action configures a channel to input channel on 34950A module. Threshold voltage can be set from 0 to 5 V in 0.02 V step.
Agt34950ConfigureOutput	This action configures a channel to output channel on 34950A module. Voltage level can be set from 1.66 V to 5 V in 0.02 V step.
Agt34950ReadBit	This action reads 1-bit digital data from a channel in 34950A module.
Agt34950ReadByte	This action reads 8-bit digital data from a channel in 34950A module.
Agt34950ReadLongWord	This action reads 32-bit digital data from a channel in 34950A module.
Agt34950ReadWord	This action reads 16-bit digital data from a channel in 34950A module.
Agt34950WriteBit	This action writes 1-bit digital data to a channel in 34950A module. **Note:** If an input channel is written, that channel is reconfigured to be an output channel.
Agt34950WriteByte	This action writes 8-bit digital data to a channel in 34950A module. **Note:** If an input channel is written, that channel is reconfigured to be an output channel.

Table 4-7. dio (Digital I/O) Actions (continued)

Actions	Descriptions
Agt34950WriteLongWord	This action writes 32-bit digital data to a channel in 34950A module. **Note:** If an input channel is written, that channel is reconfigured to be an output channel.
Agt34950WriteWord	This action writes 16-bit digital data to a channel in 34950A module. **Note:** If an input channel is written, that channel is reconfigured to be an output channel.
DigitalRead	**Low-Level Action:** Reads a value from the TS-5430 digital input ports.
digitalReadCC	**Low-Level Action:** Reads a value from the custom card digital input used in the Agilent E6198 switch/load unit.
digitalReadRegCC	**Low-Level Action:** Reads a value from the specified E8794A custom card digital input register.
digitalReadSU	**Low-Level Action:** Reads a value from the Agilent E6198/E6218A switch/load unit digital input ports or fixture ID. Note: Fixture ID only applicable for E6198.
DigitalWrite	**Low-Level Action:** Write a value to the TS-5430 digital output ports.
digitalWriteCC	**Low-Level Action:** Write a value to the Custom Card open collector digital output port used in the Agilent E6198 switch/load unit.
DigitalWriteQuery	**Low-Level Action:** Returns the current output state of the TS-5430 digital output ports.
digitalWriteQueryCC	**Low-Level Action:** Returns the current output state of the custom card open collector digital output port used in the Agilent E6198 switch/load unit.
digitalWriteQuerySU	**Low-Level Action:** Returns the current output state of the Agilent E6198/E6218A switch/load unit digital output ports. **Note:** Spare digital output only applicable for E6198.
digitalWriteRegCC	**Low-Level Action:** Write a value to the specified Agilent E8794A Custom Card digital output register.
digitalWriteRegQueryCC	**Low-Level Action:** Returns the last value written to the specified register on the Agilent Technologies E8794A custom card.
digitalWriteSU	**Low-Level Action:** Write a value to the Agilent E6198/E6218A switch/load unit digital output ports. **Note:** Spare digital output only applicable for E6198.
Spare Dig Read	**Low-Level Action:** Reads a value from the digital input port of the Agilent E1330 Digital I/O Module. **See Also:** Spare Dig Write
Spare Dig Write	**Low-Level Action:** Sends a value to the digital output port of the Agilent E1330 Digital I/O Module. **See Also:** Spare Dig Read
m9187GetCardInfo	A brief description of DIO M9187A module information.

Table 4-7. dio (Digital I/O) Actions (continued)

Actions	Descriptions
m9187InputConfigThresHold	Configures the level of both thresholds in one operation. Note that the two levels may be set to the same value if desired.
m9187InputQueryAll	Reads all 32 channels of input. Return values are as follows: 0 = State Low = below both thresholds; 1 = State High = above both thresholds; 2 = State Middle = in between the two threshold levels.
m9187InputQuerySingle	Reads the specified input channel. Return values are as follows: 0 = State Low = below both thresholds; 1 = State High = above both thresholds; 2 = State Middle = in between the two threshold levels.
m9187InputReadThresHoldLevel1	To read back Input Threshold level 1. Specifies threshold level 1. Units are volts.
m9187InputReadThresHoldLevel2	To read back Input Threshold level 2. Specifies threshold level 2. Units are volts.
m9187OutputConfigAll	Configures all 32 output channels.
m9187OutputConfigSingle	Configures the output state on a single output channel.
m9187OutputQueryAll	Returns all 32 output settings, with channel 1 in the lowest order array element. Each element contains a single channel's setting. The values returned are: 0 for a Sink (driven low), 1 for a Source (driven high), or 2 if the channel is Off (inactive).
m9187OutputQuerySingle	Returns the current setting of the specified output channel. The value returned will be either 0, 1, or 2 where: 0 = Sink (driven low), 1 = Source (driven high), and 2 = Off (inactive).
m9187Reset	This function is built to reset configuration in DIO.
digitalResetCC	**Low-Level Action:** Perform a soft reset on the Agilent E8794 custom card. This will cause the Digital I/O ports to go to their reset state.

Table 4-8. DMM Actions

Actions	Descriptions
dmmMeasureACI	**High-Level Action:** Measures A current using the Agilent 34401/11 DMM & Agilent M9182A/M9183A, only. **Usage:** This action is self contained; there is no need to call other actions for this measurement. **Notes:** 1. If the "MeasurementMode" parameter is set to "1", then call "dmmInitiate" (or trigger the DMM) and "dmmGetResults"/"dmmGetResultEX" to get the measurement. 2. If the "MeasurementMode" parameter is set to "2", then call "dmmGetResults"/"dmmGetResultEX" to get measurement. 3. If the "MeasurementMode" parameter is set to "3", the action makes the measurements without any other actions.
dmmMeasureACV	**High-Level Action:** Measures AC voltage using the Agilent E1411, Agilent 34401/11 DMM or Agilent M9182A/M9183A. **Usage:** This action is self contained; there is no need to call other actions for this measurement. **Requires:** Connect ISrcLo to DVMLo using Switching. **Notes:** 1. If the "MeasurementMode" parameter is set to "1", then call "dmmInitiate" (or trigger the DMM) and "dmmGetResults" /"dmmGetResultEX" to get the measurement. 2. If the "MeasurementMode" parameter is set to "2", then call "dmmGetResults"/"dmmGetResultEX" to get measurement. 3. If the "MeasurementMode" parameter is set to "3", the action makes the measurements without any other actions.
dmmMeasureCAP	**High-Level Action:** Measures Capacitance using the Agilent 34411A DMM, only. **Usage:** This action is self contained; there is no need to call other actions for this measurement. **Notes:** 1. If the "MeasurementMode" parameter is set to "1", then call "dmmInitiate" (or trigger the DMM) and "dmmGetResults" to get the measurement. 2. If the "MeasurementMode" parameter is set to "2", then call "dmmGetResults" to get measurement. 3. If the "MeasurementMode" parameter is set to "3", the action makes the measurements without any other actions.

Table 4-8. DMM Actions (continued)

Actions	Descriptions
dmmMeasureCurrent	**High-Level Action:** Measures voltage across a user supplied shunt or sense resistor to calculate current using the Agilent E1411 or Agilent 34401/11 DMM. Result is calculated value of Measured Voltage divided by Shunt/Sense Resistor Value. **Usage:** This action is self contained; there is no need to call other actions for this measurement. **Requires:** Connect ISrcLo to DVMLo using Switching. **Notes:** 1. If the "MeasurementMode" parameter is set to "1", then call "dmmInitiate" (or trigger the DMM) and "dmmGetResults" to get the measurement. 2. If the "MeasurementMode" parameter is set to "2", then call "dmmGetResults" to get measurement. 3. If the "MeasurementMode" parameter is set to "3", the action makes the measurements without any other actions.
dmmMeasureDCI	**High-Level Action:** Measures DC Current using the Agilent 34401/11 or Agilent M9182A/M9183A DMM only. **Usage:** This action is self contained; there is no need to call other actions for this measurement. **Notes:** 1. If the "MeasurementMode" parameter is set to "1", then call "dmmInitiate" (or trigger the DMM) and "dmmGetResults"/"dmmGetResultEX" to get the measurement. 2. If the "MeasurementMode" parameter is set to "2", then call "dmmGetResults"/"dmmGetResultEX" to get measurement. 3. If the "MeasurementMode" parameter is set to "3", the action makes the measurements without any other actions.
dmmMeasureDCV	**High-Level Action:** Measures DC voltage using the Agilent E1411, Agilent 34401/11 DMM or Agilent M9182A/M9183A. **Usage:** This action is self contained; there is no need to call other actions for this measurement. **Requires:** Connect ISrcLo to DVMLo using Switching. **Notes:** 1. If the "MeasurementMode" parameter is set to "1", then call "dmmInitiate" (or trigger the DMM) and "dmmGetResults"/"dmmGetResultEX" to get the measurement. 2. If the "MeasurementMode" parameter is set to "2", then call "dmmGetResults"/"dmmGetResultEX" to get measurement. 3. If the "MeasurementMode" parameter is set to "3", the action makes the measurements without any other actions.

Table 4-8. DMM Actions (continued)

Actions	Descriptions
dmmMeasureFrequency	**High-Level Action:** Measures frequency using the Agilent 34401/11 DMM, only. **Usage:** This action is self contained; there is no need to call other actions for this measurement. **Notes:** 1. If the "MeasurementMode" parameter is set to "1", then call "dmmInitiate" (or trigger the DMM) and "dmmGetResults" to get the measurement. 2. If the "MeasurementMode" parameter is set to "2", then call "dmmGetResults" to get measurement. 3. If the "MeasurementMode" parameter is set to "3", the action makes the measurements without any other actions.
dmmMeasureOhms	**High-Level Action:** Measures resistance using the Agilent E1411 or Agilent 34401/11 DMM. **Usage:** This action is self contained in MeasurementMode 3 such that there is no need to call other actions to complete this measurement. **Requires:** Connect ISrcLo to DVMLo and ISrcHi to DVMHi somewhere in the measurement path. **Notes:** 1. If the "MeasurementMode" parameter is set to "1", then call "dmmInitiate" (or trigger the DMM) and "dmmGetResults" to get the measurementt. 2. If the "MeasurementMode" parameter is set to "2", then call "dmmGetResults" to get measurement. 3. If the "MeasurementMode" parameter is set to "3", the action makes the measurements without any other actions. **See Also:** dmmMeasureACV, dmmMeasureCurrent, dmmMeasureDCV, dmmMeasureACI, dmmMeasureDCI, dmmMeasureFrequency, dmmMeasurePeriod, dmmMeasureTrigVoltage **Instrument:** DMM
dmmMeasurePeriod	**High-Level Action:** Measures period using the Agilent 34401/11 DMM, only. **Usage:** This action is self contained; there is no need to call other actions for this measurement. **Notes:** 1. If the "MeasurementMode" parameter is set to "1", then call "dmmInitiate" (or trigger the DMM) and "dmmGetResults" to get the measurement. 2. If the "MeasurementMode" parameter is set to "2", then call "dmmGetResults" to get measurement. 3. If the "MeasurementMode" parameter is set to "3", the action makes the measurements without any other actions.

Table 4-8. DMM Actions (continued)

Actions	Descriptions
dmmMeasureTrigVoltage	**High-Level Action:** Configures the Agilent E1411 or Agilent 34401 DMM to measure a DC voltage, triggered by a VXI backplane TTL signal (Agilent E1411 only) or External signal. **Usage:** See "Notes" below. **Requires:** Setup the trigger source. Connect ISrcLo to DVMLo using Switching. **Notes:** 1. If the "MeasurementMode" parameter is set to "1", then call "dmmInitiate" (to arm the DMM) and "dmmGetResults" to get the measurement. 2. If the "MeasurementMode" parameter is set to "2", then call "dmmGetResults" to get measurement.
dmmAutoZeroEX	Valid for the 34980A and 34411A dmm. Sets Auto Zero to ON, OFF, or ONCE for the function specified. This does not effect the Auto Zero state of other functions.
dmmConfCal	**Low-Level Action:** Configures the 50/60Hz line frequency and enables/disables autozero of the Agilent E1411 DMM. For the Agilent 34401A/11A & 34980A, the line frequency parameter is ignored. **Usage:** 1. Add the appropriate values into the parameters and then call this action. 2. Call "dmmSet" to send the setup information to the DMM. 3. Call "dmmIsSet" to wait until the DMM is ready for configuration. 4. Call "dmmInitiate" to configure the DMM (skip if trigger mode is set to immediate). **Requires:** "dmmIsSet" or "dmmInitiate" after this action. **Notes:** Incorrect power line frequency causes noisy readings. The 34980A automatically reads and sets line frequency at power up. Autozero ON decreases measurement speed 2X.
dmmConfTrigInEX	Valid for the 34980A/34411A dmm. Allows the user to set up Bus (software) trigger mode as well as Immediate, External and Internal. **Notes:** Bus (software) and Internal trigger only applicable for 34411A DMM.

Table 4-8. DMM Actions (continued)

Actions	Descriptions
dmmConfSamp	**Low-Level Action:** Selects the sample source, sample count, and sample period of the Agilent E1411 and 34411A DMM. Only the sample count is relevant for the Agilent 34401A and 34980A DMMs. Presample count only applicable for 34411A DMM. **Usage:** 1. Add the appropriate values into the parameters and then call this action. 2. Call any other configuration action(s). 3. Call "dmmSet" to send the setup information to the DMM. 4. Call "dmmIsSet" to wait until the DMM is ready for the measurement. 5. Call "dmmInitiate" to start the measurement cycle. 6. Trigger the DMM and then call "dmmGetResults" to read the measurement. If the "trigfirst" parameter in "dmmGetResults" is set to "1", the action triggers the DMM and makes the measurement; no other trigger is needed. **Requires:** "dmmSet", "dmmInitiate", or "dmmGetResults" after this action. **Notes:** The "count" parameter must be set to "1" if using the "dmmGetResults" Action to read the DMM.
dmmConfTriggerIn	**Low-Level Action:** Selects the trigger input parameters of the Agilent E1411, 34401A/11A, 34980A DMM, Agilent M9182A or M9183A. **Usage:** 1. Add the appropriate values into the parameters and then call this action. 2. Call any other configuration action(s). 3. Call "dmmSet" to send the setup information to the DMM. 4. Call "dmmIsSet" to wait until the DMM is ready for the measurement. 5. Call "dmmInitiate" to start the measurement cycle. 6. Trigger the DMM and then call "dmmGetResults" to read the measurement. If the "trigfirst" parameter in "dmmGetResults" is set to "1", the action triggers the DMM and makes the measurement; no other trigger is needed. (M9182A/M9183A use dmmGetResultEX.) **Requires:** "dmmSet", "dmmInitiate", or "dmmGetResults (M9182A/M9183A use dmmGetResultEX)" after this action.

Table 4-8. DMM Actions (continued)

Actions	Descriptions
dmmConfFunction	**Low-Level Action:** Sets the function, range and resolution of the Agilent E1411, Agilent 34401A/11A,34980A, M9182A or M9183A DMM. **Usage:** 1. Add the appropriate values into the parameters and then call this action. 2. Call any other configuration action(s). 3. Call "dmmSet" to send the setup information to the DMM. 4. Call "dmmIsSet" to wait until the DMM is ready for the measurement. 5. Call "dmmInitiate" to start the measurement cycle. 6. Trigger the DMM and then call "dmmGetResults" to read the measurement. If the "trigfirst" parameter in "dmmGetResults" is set to "1", the action triggers the DMM and makes the measurement; no other trigger is needed. (Agilent M9182A/M9183A use dmmGetResultEX.) **Requires:** "dmmSet", "dmmInitiate", or "dmmGetResults (Agilent M9182A/M9183A use dmmGetResultEX) " after this action.
dmmGetMultipleResultsEX	Valid for the 34980A, 34411A and Agilent M9182A/M9183A dmm. Returns the number of DMM measurements requested. If there are fewer measurements available than the number requested an error is generated. For instance, if the number of samples is configured for 40 samples and the dmm is configured to measure DC volts in the 10 volt range with 10ms aperture, then the Initiate action will clear the DMM measurement memory and store 40 DC voltage measurements. This action can then be called to return up to 40 data points (if fewer than 40 points are requested, then the oldest points will be returned first). If 41 points or more are requested an error will be generated. **Notes:** Agilent M9182A/M9183A will return error code when there is no reading available (-2147204573), Setting conflict (-2147204585), Time-out exception (-2147204587) and Maximum time exceeded (-2147213309).
dmmGetResultEX	Valid for the 34980A & Agilent M9182A/M9183A dmm. Returns the most recent DMM measurement. Returns only one measurement therefore the "numresults" parameter is not required. The measurement should be Configured, Initiated and Triggered prior to calling this action. **Notes:** Agilent M9182A/M9183A will return error code when there is no reading available (-2147204573), Setting conflict (-2147204585), Time-out exception (-2147204587) and Maximum time exceeded (-2147213309).

Table 4-8. DMM Actions (continued)

Actions	Descriptions
dmmInitiate	**Low-Level Action:** Initiates a reading cycle on the Agilent E1411, Agilent 34401A/11A, 34980A, Agilent M9182A/M9183A DMM. **Usage:** 1. Call any other configuration action(s). 2. Call "dmmSet" to send the setup information to the DMM. 3. Call "dmmIsSet" to wait until the DMM is ready for the measurement. 4. Call this action to start the measurement cycle. 5. Call "dmmGetResults"/"dmmGetResults EX" to read the measurement.
dmmIsSet	**Low-Level Action:** Waits until the Agilent E1411, 34401A/11A, 34980A, Agilent m9182A or M9183A DMM is ready for measurement. (Action returns a true when ready.) **Usage:** 1. Call any other configuration action(s). 2. Call "dmmSet" to send the setup information to the DMM. 3. Call this action to wait until the DMM is ready for the measurement. 4. Call "dmmInitiate" to start the measurement cycle. 5. Trigger the DMM and then call "dmmGetResults"/"dmmgetResultsEX" to read the measurement. If the "trigfirst" parameter in "dmmGetResults" is set to "1", the action triggers the DMM and makes the measurement; no other trigger is needed. **Requires:** "dmmInitiate" or "dmmGetResults"/"dmmGetResultEX" after this action.
dmmMeas2WResEx	Valid for the 34980A/34411A & Agilent M9182A/M9183A dmm. Does a quick (meaning less complicated- no configuration required) resistance (2 wire) measurement. The measurement is taken immediately.
dmmMeas4WResEx	Valid for the 34980A/34411A & Agilent M9182A/M9183A dmm. Does a quick (meaning less complicated- no configuration required) resistance (4 wire) measurement. The measurement is taken immediately.
dmmMeasACCurrentEX	Valid for the 34980A/34411A dmm. Does a quick (meaning less complicated- no configuration required) AC current measurement. The measurement is taken immediately.
dmmMeasACVoltsEX	Valid for the 34980A/34411A dmm. Does a quick (meaning less complicated- no configuration required) AC voltage measurement. The measurement is taken immediately.
dmmMeasCAPPXI	**High-Level Action:** Measures Capacitance using the Agilent M9182A/M9183A DMM, only. **Usage:** This action is self contained; there is no need to call other actions for this measurement. **Notes:** Ignore MeasurementMode parameter.

Table 4-8. DMM Actions (continued)

Actions	Descriptions
dmmMeasDCCurrentEX	Valid for the 34980A/34411A dmm. Does a quick (meaning less complicated- no configuration required) DC current measurement. The measurement is taken immediately.
dmmMeasDCVoltsEX	Valid for the 34980A/34411A dmm. Does a quick (meaning less complicated- no configuration required) DC voltage measurement. The measurement is taken immediately.
dmmMeasFrequencyEX	Valid for the 34980A/34411A dmm. Does a quick (meaning less complicated- no configuration required) frequency measurement. The measurement is taken immediately.
dmmMeasFrequencyPXI	Valid for the Agilent M9182A/M9183A dmm. Does a quick (meaning less complicated- no configuration required) frequency measurement. The measurement is taken immediately.
dmmMeasPeriodEX	Valid for the 34980A/34411A dmm. Does a quick (meaning less complicated- no configuration required) period measurement. The measurement is taken immediately.
dmmMeasPeriodPXI	Valid for the Agilent M9182A/M9183A dmm. Does a quick (meaning less complicated- no configuration required) frequency measurement. The measurement is taken immediately.
dmmGetMultipleResults	**Low-Level Action:** Returns multiple readings from the DMM. It waits, if necessary, for optionally triggering the DMM first. **Usage:** 1. Call any other configuration action(s). 2. Call "dmmSet" to send the setup information to the DMM. 3. Call "dmmIsSet" to wait until the DMM is ready for the measurement. 4. Call "dmmInitiate" to start the measurement cycle. 5. Trigger the DMM and then call this action to read the measurement. If the "trigfirst" parameter is set to "1", the action triggers the DMM and makes the measurement; no other trigger is needed.
dmmPbQueryCmd	**Low-Level Action:** Sends a SCPI query string to the 34401A/11A and returns the result. VALID FOR 34401A/11A ONLY. THE RESULT BUFFER IS LIMITED TO 32767 CHARACTERS.
dmmPbSendCmd	**Low-Level Action:** Sends a SCPI command string to the 34401A/11A. VALID FOR 34401A/11A ONLY.
dmmQueryCmd	**Low-Level Action:** Sends a SCPI query string to the 34401A/11A and returns the result. VALID FOR 34401A/11A ONLY. THE RESULT BUFFER IS LIMITED TO 100 CHARACTERS.

Table 4-8. DMM Actions (continued)

Actions	Descriptions
dmmGetResults	**Low-Level Action:** Returns one reading from the Agilent E1411, 34401A/11A or 34980A DMM. It waits, if necessary, for optionally triggering the DMM first. **Usage:** 1. Call any other configuration action(s). 2. Call "dmmSet" to send the setup information to the DMM. 3. Call "dmmIsSet" to wait until the DMM is ready for the measurement. 4. Call "dmmInitiate" to start the measurement cycle. 5. Trigger the DMM and then call this action to read the measurement. If the "trigfirst" parameter is set to "1", the action triggers the DMM and makes the measurement; no other trigger is needed.
dmmReset	**Low-Level Action:** Resets the Agilent E1411, 34401A/11A, 34980A DMM or Agilent M9182A/M9183A to its power-on state. **Usage:** Call this action when the DMM configuration is unknown and the DMM is to be configured into a known state.
dmmConfSample	**Low-Level Action:** Selects the sample source, sample count, and sample period of the Agilent E1411 and 34411A DMM. Only the sample count is relevant for the Agilent 34401A and 34980A DMMs. **Usage:** 1. Add the appropriate values into the parameters and then call this action. 2. Call any other configuration action(s). 3. Call "dmmSet" to send the setup information to the DMM. 4. Call "dmmIsSet" to wait until the DMM is ready for the measurement. 5. Call "dmmInitiate" to start the measurement cycle. 6. Trigger the DMM and then call "dmmGetResults" to read the measurement. If the "trigfirst" parameter in "dmmGetResults" is set to "1", the action triggers the DMM and makes the measurement; no other trigger is needed. **Requires:** "dmmSet", "dmmInitiate", or "dmmGetResults" after this action. **Notes:** The "count" parameter must be set to "1" if using the "dmmGetResults" Action to read the DMM.
dmmSelfTestEX	Valid for the 34980A, 34411A & Agilent M9182A/M9183A dmm. THIS ACTION WILL RESET THE 34980A AND ALL OF THE CARDS IN THE CHASSIS. Executes the DMM Selftest and returns both the result code and message. NOTE: This might take a while- this action sets the time-out to 20 seconds and then returns it to the previous value when complete (or even if it fails). For Agilent M9182A/M9183A, input terminals must be disconnected before running Self Test.
dmmSendSWTrigEX	Valid for the 34980A/34411A dmm. Sends a Bus (Software) trigger to the DMM.

Table 4-8. DMM Actions (continued)

Actions	Descriptions
dmmSetTimeout	**Low-Level Action:** Sets the Agilent E1411, 34401A/11A, 34980A DMM or Agilent M9182/M9813A operation time out value (in milliseconds).
dmmConfSimple	**Low-Level Action:** Selects one of three measurement types on the Agilent E1411, 34401A/11A, 34980A, Agilent M9182A or M9183A DMM for easy configuration. These are: fast measurement speed with low resolution, medium measurement speed with medium resolution, and slow measurement speed with high resolution. **Usage:** 1. Add the appropriate values into the parameters and then call this action. 2. Call any other configuration action(s). 3. Call "dmmSet" to send the setup information to the DMM. 4. Call "dmmIsSet" to wait until the DMM is ready for the measurement. 5. Call "dmmInitiate" to start the measurement cycle. 6. Trigger the DMM and then call "dmmGetResults" to read the measurement. If the "trigfirst" parameter in "dmmGetResults" is set to "1", the action triggers the DMM and makes the measurement; no other trigger is needed. (M9182A/M9183A use dmmGetResultEX). **Requires:** "dmmSet", "dmmInitiate", or "dmmGetResults (M9182A/M9183A use dmmGetResultEX)" after this action.
dmmSendCmd	**Low-Level Action:** Sends a SCPI command string to the 34401A/11A. VALID FOR 34401/11A ONLY.
DmmSelfTest	**Low-Level Action:** Runs a selftest on the Agilent E1411, 34401A/11A or 34980A Digital Multimeter (DMM).
dmmSwitchFuncEX	Valid for the 34980A/34411A dmm. This action allows the user to switch from the current DMM function to the function specified. Settings for the function that is switched to will be those that were last programmed for that function. So the user could configure the DMM for DC Volts 10 V range, 10ms aperture and measure DC Volts. Then they could configure the DMM for 4-wire resistance, 1000 ohms range, 100ms aperture and measure resistance. Then this action could be used to switch back to the DC voltage configuration without re-configuring the DMM.

Table 4-8. DMM Actions (continued)

Actions	Descriptions
dmmConfTrigIn	**Low-Level Action:** Selects the trigger input parameters of the Agilent E1411, 34401A/11A or 34980A DMM. **Usage:** 1. Add the appropriate values into the parameters and then call this action. 2. Call any other configuration action(s). 3. Call "dmmSet" to send the setup information to the DMM. 4. Call "dmmIsSet" to wait until the DMM is ready for the measurement. 5. Call "dmmInitiate" to start the measurement cycle. 6. Trigger the DMM and then call "dmmGetResults" to read the measurement. If the "trigfirst" parameter in "dmmGetResults" is set to "1", the action triggers the DMM and makes the measurement; no other trigger is needed. **Requires:** "dmmSet", "dmmInitiate", or "dmmGetResults" after this action.
dmmConfTrigOut	**Low-Level Action:** Enables you to route the Agilent E1411 DMM's "voltmeter complete" signal to the VXIbus TTL trigger lines. This action is not supported on the Agilent 34401A/11A or 34980A DMMs. **Usage:** 1. Add the appropriate values into the parameters and then call this action. 2. Call any other configuration action(s). 3. Call "dmmSet" to sent the setup information to the DMM. 4. Call "dmmIsSet" to wait until the DMM is ready for the measurement. 5. Call "dmmInitiate" to start the measurement cycle. 6. Trigger the DMM and then call "dmmGetResults" to read the measurement. If the "trigfirst" parameter in "dmmGetResults" is set to "1", the action triggers the DMM and makes the measurement; no other trigger is needed. **Requires:** "dmmIsSet", "dmmInitiate", or "dmmGetResults" after this action.
dmmVerifyTerminalSwitch	**Low-Level Action:** Checks the state of the front/rear terminal switch on an Agilent 34401A/11A DMM and compares the current switch state with the value in the parameter block. Returns 1 if the switch is in the correct state, 0 if it is not. **Notes:** This action is not supported on the Agilent 1411 and 34980A DMMs.
SM2040ArmAnalogTrigger	Setup the SM2040 for analog level trigger operation. Following reception of this command the DMM makes measurements continuously, waiting for a value which exceeds the threshold. Use SM2040ReadBuffer to get results.
SM2040ArmTrigger	Setup the SM2040 for external hardware trigger operation. Following reception of this command the DMM enters a wait state. Use SM2040ReadBuffer to get results.

Table 4-8. DMM Actions (continued)

Actions	Descriptions
SM2040BurstRead	This function is designed to read bursting measurements form the DMM, resulting from SM2040SetTrigRead and SM2040SetBurstRead operations. Set 'numReadings' to be number of settle samples + 1 + number of samples to take.
SM2040Cal	This function re-calibrates the DMM, and returns it to the current operating mode. This is an internal calibration.
SM2040ConfigureMeasurement	Sets the function, range, and rate to the selected values.
SM2040GetInfo	Returns the hardware and software versions as well as the type of card.
SM2040Read	SM2040Read reads the next result from the DMM. This routine can read all the Primary functions (those that can be selected using SM2040ConfigureMeasurement).
SM2040ReadBuffer	This function wait for a trigger, and return an array of measurements from the internal DMM buffer. Use the Abort button or a time-out value to prevent the measurement from waiting forever on a trigger. A return code of 99 indicates the abort condition and a return code of 88 indicates a time-out condition.
SM2040SetAutoRange	This function enables or disables autorange operation of the DMM.
SM2040SetBuffTrigRead	Setup the DMM for multiple triggered readings operation.
SM2040SetBurstRead	This function will set up the DMM to read bursting measurements.
SM2040SetFunction	Sets the DMM function to the selected value.
SM2040SetRange	Sets the DMM range to the selected value.
SM2040SetRate	Sets the DMM rate to the selected value.
SM2040SetRelative	Sets or clears the relative reading mode.
SM2040SetSynchronized	This function enables or disables the Synchronized operation of the DMM. Default operation is non-synchronized. Select the Synchronized mode when it is necessary to settle full scale input transitions from one reading to the next, and maintain the accuracy of the DMM. This is appropriate for VDC, Ohms, Leakage, DCI, Diode and Guarded Ohms. The result of the synchronized mode is a reduced measurement rate. To run synchronized, reading rate must be set to 10 rps or higher.
SM2040SetTrigRead	Setup the DMM for multiple triggered readings operation.
SM2040Test	
SM2040TrigRead	This function will arm the DMM, wait for a trigger, and return an array of measurements for the internal DMM buffer. Use the Abort button or a time-out value to prevent the measurement waiting forever for a trigger.

Table 4-9. DSO Actions

Actions	Descriptions
dsoConfigureTrigger	This action configure the trigger setting of the oscilloscope.
dsoGetDutyCycle	This action measures and outputs the duty cycle of the signal.
dsoGetFallTime	This action measures and outputs the fall time of the displayed falling (negative-going) edge closest to the trigger reference.

Table 4-9. DSO Actions (continued)

Actions	Descriptions
dsoGetFrequency	This action measures and outputs the frequency of the cycle on the screen closest to the trigger reference.
dsoGetPeriod	This action measures and outputs the period of the cycle closest to the trigger reference on the screen.
dsoGetPulseWidth	This action measures and outputs the width of the displayed positive pulse closest to the trigger reference.
dsoGetRiseTime	This action measures and outputs the rise time of the displayed rising (positive-going) edge closest to the trigger reference.
dsoGetVmaxVmin	This action measures and outputs the maximum and minimum vertical value present on the selected waveform.
dsoGetVpp	This action measures the maximum and minimum vertical value for the selected source, then calculates the vertical peak-to-peak value and returns that value.
dsoGetWaveform	This action acquires the signal and save the data in the "Waveform" parameter array.
dsoMsgQuery	This action sends a SCPI query string to the oscilloscope and returns the result. THE RESULT BUFFER IS LIMITED TO 32767 CHARACTERS
dsoMsgSend	This action sends a SCPI command string to the oscilloscope.
dsoSetTimeOut	This action set time-out (ms).

Table 4-10. Electronic Load Actions

Actions	Descriptions
eloadCalcInputPower	This action returns input power calculated from the latest voltage and current measurements (Average, Min/Max).
eloadConfInput	This action configures selected channel: 1. Mode (CC, CV, CR) 2. Range
eloadCreateList	This action reads the file that contain the List Commands then generates and saves complex sequences of input changes into mainframe non-volatile memory.
eloadGetCurrRange	This action gets the current range of the selected channel.
eloadGetCurrTransient	This action gets the current transient mode and level of the selected channel.
eloadGetCurrTrigger	This action gets the current trigger mode and level of the selected channel.
eloadGetMode	This action gets the mode of the selected channel.
eloadGetOhmRange	This action gets the resistance range of the selected channel.
eloadGetOhmTransient	This action gets the ohm transient mode and level of the selected channel.
eloadGetOhmTrigger	This action gets the resistance trigger mode and level of the selected channel.
eloadGetVoltRange	This action gets the voltage range of the selected channel.
eloadGetVoltTransient	This action gets the voltage transient mode and level of the selected channel.

Table 4-10. Electronic Load Actions (continued)

Actions	Descriptions
eloadGetVoltTrigger	This action gets the voltage trigger mode and level of the selected channel.
eloadInputOnOff	This action switches the input on/off.
eloadInputShort	This action simulate a short circuit at its input by turning the load on with full-scale current.
eloadMeasInputCurr	This action measures input dc current (Average, RMS, Min/Max).
eloadMeasInputVolt	This action measures input dc voltage (Average, RMS, Min/Max).
eloadRecallSettings	This action recalls present settings for all channels in the mainframe's memory. States saved in locations 1-6 are volatile, the data will be lost when power is turned off. States in locations 0, 7, 8, and 9 are nonvolatile, the data will be saved when power is removed. If a particular state is desired at power-on, it should be stored in location 0. It then will be recalled at power-on if the power-on state is set to RCL0. Any lists associated with a device state are also saved if they are stored in locations 0, 7, 8, or 9.
eloadReset	This action resets ALL channels of the electronic load to its factory-defined states.
eloadRstLatchedProt	This action removes protection latch when input tripped (protection feature).
eloadSaveSettings	This action saves present settings for all channels in the mainframe's memory. States in saved in locations 1-6 are volatile, the data will be lost when power is turned off. States in locations 0, 7, 8, and 9 are nonvolatile, the data will be saved when power is removed. If a particular state is desired at power-on, it should be stored in location 0. It then will be recalled at power-on if the power-on state is set to RCL0. Any lists associated with a device state are also saved if they are stored in locations 0, 7, 8, or 9.
eloadSelfTest	This action causes the electronic load to do a self-test and report any errors.
eloadSetCurrRange	This action sets the current range of the selected channel.
eloadSetCurrTransient	This action sets the current transient mode and level of the selected channel.
eloadSetCurrTrigger	This action sets the current trigger mode and level of the selected channel.
eloadSetMode	This action sets the mode of the selected channel.
eloadSetOhmRange	This action sets the resistance range of the selected channel.
eloadSetOhmTransient	This action sets the ohm transient mode and level of the selected channel.
eloadSetOhmTrigger	This action sets the resistance trigger mode and level of the selected channel.
eloadSetVoltRange	This action sets the voltage range of the selected channel.
eloadSetVoltTransient	This action sets the voltage transient mode and level of the selected channel.
eloadSetVoltTrigger	This action sets the voltage trigger mode and level of the selected channel.

Table 4-10. Electronic Load Actions (continued)

Actions	Descriptions
eloadTrigger	This action generates a trigger.
eloadQueryCommand	This action will perform an SCPI Query based on the input SCPI command.
eloadSendCommand.umd	This action will execute Send SCPI command based on the command input.

Table 4-11. Spectrum Analyzer Actions

Actions	Descriptions
esaCalGetPathLoss	This action is use to get the cal loss value of the path set. **Note:** Use after esaCalReadData. Pass in 4 frequency value from esaCalReadData to the freq value in the parameter. Insert RFTC # and Test Freq to retrieve the path loss value.
esaCalReadData	This action is use to read back the cal data store from action esaCalStoreData. 1. Freq1: Cal Data store in Freq1. 2. Freq1: Cal Data store in Freq2. 3. Freq1: Cal Data store in Freq3. 4. Freq1: Cal Data store in Freq4.
esaCalStoreData	This action is use to store the cal path loss data. 1. NoOfFreqs: The number of freq points cal. Range 1 to 4. 2. calData: The cal value obtained from measurement. 3. Freq1, Freq2, Freq3, Freq4: Frequency points to cal.
esaDemod	**High-Level Action:** To set the demodulation parameter in ESA.
esaGetTraceData	This method transfers data from the instrument to the controller. There are normally 401 points in a trace.
esaID	Gets the *IDN? string from the ESA
esaMarkerPeakSearch	**Low-Level Action:** The method uses the spectrum analyzer's marker peak search functions.
esaMarkerSetMode	**Low-Level Action:** This method selects the type of markers to activate.
esaMeasFreqDev	**High-Level Action:** Measures Frequency deviation E4402B ESA. **Usage:** This action is self contained; there is no need to call other actions for this measurement. **Requires:** Provide signal to ESA. **See Also:** esaSetFreqCenter, esaSetFreqSpan, esaSetBandwidth, esaSetAmpliRefLev, esaSweepSetTime. **Instrument:** ESA

Table 4-11. Spectrum Analyzer Actions (continued)

Actions	Descriptions
esaMeasureSetMode	**Low-Level Action:** This method stops the current measurement and sets up the instrument for the specified measurement using the factory default instrument settings. This will always set the instrument in single sweep mode and place the measurement in the idle state.
esaOBWConfig	**Low-Level Action:** This method sets up the occupied bandwidth measurement operation.
esaOBWMeas	**Low-Level Action:** This method retrieves the scalar results of occupied bandwidth and transmit frequency error from the instrument, based on the method of retrieval (measure, read, or fetch).
esaReset	**Low-Level Action:** Resets the Agilent ESA to its power-on state. **Usage:** Call this action when the ESA configuration is unknown and the ESA is to be configured into a known state.
esaSelfTest	Call this action to perform self test for the ESA.
esaSetAmpliAttenuation	This method sets the spectrum analyzer's RF Input Attenuator value. Setting the auto to 0 (Auto off), the amplitude attenuation uncouples the attenuator from the amplitude reference level. To couple the attenuator to the reference level, set auto to 1 (Auto on).
esaSetAmpliRefLev	**Low-Level Action:** To set the amplitude reference level.
esaSetAmpliScale	**Low-Level Action:** To set the display vertical scale format as either logarithmic or linear.
esaSetAverage	**Low-Level Action:** This method sets the spectrum analyzer's average type and count. Then, turns the spectrum analyzer's averaging on or off.
esaSetBandwidth	**Low-Level Action:** This action allow user to set the Bandwidth of the spectrum analyzer.
esaSetFreqCenter	This action sets the spectrum analyzer's center frequency. For E4402B ESA: Range 9 kHz ~ 3 GHz
esaSetFreqSpan	Sets the freq span for ESA.
esaSetFreqStartStop	This action sets the spectrum analyzer's start and stop frequencies. For E4402B ESA: Range 9 kHz ~ 3 GHz
esaSweepSetMode	**Low-Level Action:** This method selects whether the trigger system is continuously initiated or not. This corresponds to continuous measurement or single measurement operation.
esaSweepSetTime	**Low-Level Action:** This method specifies the time in which the instrument sweeps the displayed frequency range.

Table 4-11. Spectrum Analyzer Actions (continued)

Actions	Descriptions
esaTraceSetMode	**Low-Level Action:** This method selects the spectrum analyzer's trace mode for the selected trace.
esaTriggerSetSource	**Low-Level Action:** This method selects the spectrum analyzer's source (or type) for triggering used to start a measurement. Use esaTriggerSetVideoLevel to set the video trigger level.
esaTriggerSetVideo	**Low-Level Action:** This method sets the spectrum analyzer's video trigger level for when the trigger source is set to video.

Table 4-12. Signal Generator Actions

Actions	Descriptions
esgCustomConfigFilter	This action configures the Custom Digital Modulation Format pre-modulation filter parameters, including selecting user defined FIR filter stored in the instrument's memory.
esgCustomConfigPattern	**High-Level Action:** This method is used to configure Custom Digital Modulation Format pattern data format
esgCustomMode	**Low-Level Action:** To set the custom mode to on/off.
esgCustomModifyModulation	**High-Level Action:** This method customizes the Custom Digital Modulation Format current modulation type.
esgCustomModifyStandard	**High-Level Action:** This method is used to configure Custom Digital Modulation Format standard parameters, modifying the following aspects of the standard transmission: phase polarity, user-defined Differential Encoder state, symbol rate, and bursted RF signal shape.
esgFSKConfig	**High-Level Action:** This action set the FSK configuration for the ESG.
esgID	Gets the *IDN? string from the ESG.
esgReset	**Low-Level Action:** Resets the Agilent ESG to its power-on state. **Usage:** Call this action when the ESG configuration is unknown and the ESG is to be configured into a known state.
esgSelfTest	Call this action to perform self test for the ESG.
esgSetFrequency	**Low-Level Action:** This function sets the signal generator's CW (Continuous Wave) output frequency.
esgSetPower	**Low-Level Action:** This function sets the RF output power.
esgSetPowerStatus	**Low-Level Action:** This function sets the operation state of the signal generator's RF output.

Table 4-13. Event Detector Actions

Actions	Descriptions
eventConf	**Low-Level Action:** Configures the Event Detector for a measurement. **Usage:** 1. Add the appropriate values into the parameters and then call this action. 2. Call "eventSet" to sent the setup information to the Event Detector. 3. Call "eventIsSet" to wait until the Event Detector is ready for the measurement. 4. Call "eventInitiate" to start the measurement. 5. Call "eventGetResults" Action to make the measurement. **Requires:** "eventSet" or "eventInitiate" after this action. **Notes:** Resolution and range depend on the clock frequency setting (10 Mhz = 0.1us/tic, etc.; see Agilent Z2902 User's manual).
eventInitiate	**Low-Level Action:** Starts the previously configured Event Detector measurement. **Usage:** 1. Call the "eventConf" Action to set up the Event Detector. 2. Call "eventSet" to sent the setup information to the Event Detector. 3. Call "eventIsSet" to wait until the Event Detector is ready for the measurement. 4. Call this action to start the measurement. 5. Call "eventGetResults" Action to make the measurement.
eventIsSet	**Low-Level Action:** Waits until the Event Detector is ready for operation. (Action returns a true when ready.) **Usage:** 1. Call the "eventConf" Action to set up the Event Detector. 2. Call "eventSet" to sent the setup information to the Event Detector. 3. Call this action to wait until the Event Detector is ready for the measurement. 4. Call "eventInitiate" to start the measurement. 5. Call "eventGetResults" Action to make the measurement. **Requires:** "eventInitiate" after this action.
eventMeasure	**High-Level Action:** Makes a measurement using the Event Detector and stores data into memory. **Usage:** Use this action to make a measurement to read now, or later from memory using the "eventGetResults" Action. **Notes:** This action is self contained; there is no need to call other actions for this measurement.

Table 4-13. Event Detector Actions (continued)

Actions	Descriptions
eventGetResults	**High-Level Action:** Returns stored data from the Event Detector. **Usage:** This action reads back all stored data generated by previous events.
eventReset	**Low-Level Action:** Resets the Event Detector to its power-on state. **Usage:** Call this action when the Event Detector configuration is unknown and it is to be configured into a known state. **Notes:** Connect all unused Event Detector pins to ground.
eventSet	**Low-Level Action:** Sends current setup information to the Event Detector. **Usage:** 1. Call the "eventConf" Action to set up the Event Detector. 2. Call this action to sent the setup information to the Event Detector. 3. Call "eventIsSet" to wait until the Event Detector is ready for the measurement. 4. Call "eventInitiate" to start the measurement. **Requires:** "eventIsSet" or "eventInitiate" after this action.

Table 4-14. Miscellaneous Actions

Actions	Descriptions
CalcuateRMS	Calculates the RMS value for a waveform. Usage: The action takes a waveform and returns the RMS value as an output value. **Notes:** The calculation should be done on a integer number of cycles of the waveform.
ConcatenateStrings	**High-Level Action:** The value of String2 is concatenated onto the end of String1. **Instrument:** N/A
ConvertArrayToWaveform	Converts a real array into a waveform datatype. Usage: Provide a real array, a start time and stop time and the action will convert the data to a waveform. **Notes:** The size of the array is currently limited to 8K.

Table 4-14. Miscellaneous Actions (continued)

Actions	Descriptions
DialogOkay	**Low-Level Action:** Display a message box with a prompt message and "Ok" button. Use it to temporarily halt test execution or notify an operator to do some manual test setup (e.g., make temporary connection). **Notes:** This action is similar to the DialogOkayModal action, except the display box does not keep on top (i.e., does not remain in front of other windows).
DialogOkayModal	**Low-Level Action:** Display a message box with a prompt message and "Ok" button. Use it to temporarily halt test execution or notify an operator to do some manual test setup (e.g., make temporary connection). Use this to prevent input to other windows in the application. **Notes:** This action is similar to the DialogOkay action, except the display box keeps on top (i.e., remains in front of other windows).
DialogYesNo	**Low-Level Action:** Display a message in a dialog box; gives a Yes/No choice to continue in two different ways. Can be used to direct a test flow. For example, it allows you to stop a test if no more data is needed. Usage: Use the "result" parameter to determine the test flow. **Notes:** This action is similar to the DialogYesNoModal action, except the display box does not keep on top (i.e., does not remain in front of other windows).
DialogYesNoModal	**Low-Level Action:** Display a message in a dialog box; gives a Yes/No choice to continue in two different ways. Can be used to direct a test flow. For example, it allows you to stop a test if no more data is needed. Use this to prevent input to other windows in the application. **Usage:** Use the "result" parameter to determine the test flow. **Notes:** This action is similar to the DialogYesNo action, except the display box keeps on top (i.e., remains in front of other windows).
IVIDriverDisable	Quickly places the instrument in a state where it has no, or minimal, effect on the external system to which it is connected. This state is not necessarily a known state.
IVIDriverErrorQuery	Queries the instrument and returns instrument specific error information. This function can be used when QueryInstrumentStatus is True to retrieve error details when the driver detects an instrument error.
IVIDriverLockObject	Obtains a multithread lock on the driver after waiting until all other execution threads have released their locks on the instrument session.
IVIDriverReset	Places the instrument in a known state and configures instrument options on which the IVI specific driver depends (for example, enabling/disabling headers). For an IEEE 488.2 instrument, Reset sends the command string *RST to the instrument.

Table 4-14. Miscellaneous Actions (continued)

Actions	Descriptions
IVIDriverSelfTest	Performs an instrument self test, waits for the instrument to complete the test, and queries the instrument for the results. If the instrument passes the test, TestResult is zero and TestMessage is 'Selftest passed'.
IVIDriverUnlockObject	Releases a previously obtained multithread lock.
FormatString	**High-Level Action:** The value of Argument is processed through the FormatString and will show up in ResultString. **Notes:** 1. In this version the Argument is of type Int32, but the FormatStringItem routine will also accept Argument of type Real64 or String. 2. FormatString will accept any valid 'C' output format. 3. You are only allowed a single argument per Action call. Use with ConcatenateStrings to form more complex strings.
GetFixtureID	**Low-Level Action:** Reads back the fixture I.D. A 0…255 value where 255 = no ID jumpers are present.
globalReset	**Low-Level Action:** Call all reset routines of instruments listed in the hardware configuration table. TestExec SL automatically calls this action at start up to initialize all instruments in the system. It is also automatically called when an exception is detected by TestExec SL when there is nothing in the testplan's error sequence. **Notes:** This action resets all instruments and then resets all switching.
DelayMillisecond	High-Level Action: Delays the test by the specified number of milliseconds. **Usage:** Use to delay the requested milliseconds time for instrument or UUT settling times. **Notes:** Overhead caused by executing this action increases the delay time longer than set in the action.
msgStringIO	**Low-Level Action:** This action sends and reads strings from an instrument. **Notes:** Instrument must be a message based device opened with the hwhmsg handler. No state tracking or checking of this device is provided.
NameSwitch	**High-Level Action:** Switch string name. Cycle through names in StringArray returning current one. Wraps back to the first element. **Usage:** Use to extract a string at a time from the StringArray. Set CurrentIndex to a number greater than MaxStrings so the first increment starts the names at the first string name.

Table 4-14. Miscellaneous Actions (continued)

Actions	Descriptions
LogReport	**High-Level Action:** This action allows the user to place any string into the log data file by producing a report record which will use the string argument. **Usage:** Used to add custom information to the data log file. **Notes:** 1. Sensitive to data log options. 2. Must have String1 data item and have it be a member of the REPORT record type.
LogDoubleToReport	**High-Level Action:** This action allows the user to place any double into the log data file by producing a report record which will use the string argument. **Usage:** Used to add custom information to the data log file. **Notes:** 1. Sensitive to data log options. 2. Must have String1 data item and have it be a member of the REPORT record type.
LogIntToReport	**High-Level Action:** This action allows the user to place any integer into the log data file by producing a report record which will use the string argument. **Usage:** Used to add custom information to the data log file. **Notes:** 1. Sensitive to data log options. 2. Must have String1 data item and have it be a member of the REPORT record type.
LogReportImmediate.	**High-Level Action:** This action allows the user to place any string into the log data file by producing a report record which will use the string argument. **Usage:** Used to add custom information to the data log file. **Notes:** 1. Sensitive to data log options. 2. Must have String1 data item and have it be a member of the REPORT record type.
DelaySecond	**High-Level Action:** Delays the test by the specified number of seconds. **Usage:** Use to delay the requested seconds before executing the next action.

Table 4-14. Miscellaneous Actions (continued)

Actions	Descriptions
StoreLogString	**Low-Level Action:** This action allows the user to place any string into the log data storage area which will be used by any log record which accesses that field. **Usage:** Used to modify information in the data log storage area. **Notes:** 1. Sensitive to data log files used (tstexcsl.ini [Data Log] section.). 2. Must have a data item named (first column of *dsdef.ini) the same as it appears in the DSFieldName parameter value (case sensitive) otherwise an exception is raised.
string_to_real64	Convert string number to a Real 64 number
UsecsElapsed	**High-Level Action:** This action returns the time elapsed since the last "UsecsStart" Action. **Usage:** This action stores the result into variable "Initiate_usec".
UsecsStart	**High-Level Action:** Starts clock for use with the "UsecsElapsed" Action.
DelayMicroseconds	**High-Level Action:** Delays the test by the specified number of microseconds. **Usage:** Use to delay the requested microseconds before executing the next action. **Notes:** Overhead caused by executing this action increases the delay time longer than set in the action.

Table 4-14. Miscellaneous Actions (continued)

Actions	Descriptions
UUTConnect	**High-Level Action:** Connects to UUT pins by constructing Path names by adding a prefix (NamePrefix) to the base path name (BaseNames). The base path name has the form 'PinId...NodeId' where adding the 'NamePrefix' to the PinID makes a valid UUT node and 'NodeId' is the system resource which the UUTPin should connect to (ex: DmmHi). Auto disconnect paths at end of test. Example: BaseNames[0] = "Pin1...DmmHi" NamePrefix = "UUT1-" Action looks for a path named "UUT1-Pin1...DmmHi" in the public symbol tables and, if found, Connects that path at the setup of the action and disconnects it at the end of the action. If no such path is found, this action attempts to construct one, adding it to the TestplanGlobals symbol table of the testplan. Usage: 1. Create UUT node names like Uut1-Pin1, Uut2-Pin1, … in UUT Layer. 2. Create Paths using these node names in the TestplanGlobals symbol table. Ex: Path "Uut1-Pin1...DmmHi" with a path value of [Uut1-Pin1 Somenode DmmHi] 3. Set the NamePrefix to the desired UUT (ex: Uut1-) using 'NameSwitch' or 'StringFromArray' action. 4. Use this action in place of the switch action to connect to the UUT. **Notes:** Connection wait times are overlapped or all the paths found by this routine from BaseNames. To avoid this, use extra UUTTConnect actions.
WaveformDataRead	**High-Level Action:** Reads waveform sample data into waveform data type. Start time is set to 0, stop time is set to the number of samples times the time per sample (SampleTime). Usage: Use to read a waveform definition from a text file. **Notes:** 1. This action is self contained; no other action needed. 2. Previous data in waveform is lost. 3. A maximum of 8000 samples can be read.
WaveformDataWrite	**High-Level Action:** Write samples from 'hWaveform', one floating point number sample per line, to 'Filename.' (Most likely to be used with ADC actions.) **Usage:** Use to write a waveform definition to a text file. **Notes:** This action is self contained; no other action is needed.

Table 4-15. Measurement Control Module Actions

Actions	Descriptions
ForceIMeasV	Uses the MCM as a current source and measures the resultant voltage on the UUT using the DMM.
ForceIMeasValt	Uses the MCM as a current source to measures the resultant voltage on the UUT using the DMM.
ForceVMeasI	Uses the MCM as a voltage source and measures the resultant UUT current flow using the DMM.
viConfAttenuator	Configures the knee voltage of the MCM's programmable attenuator.
viConfCompare	Configures the MCM's comparator threshold voltage for triggering other instruments.
viIsSet	Waits until the MCM is ready for operation. (Action returns a true when ready.)
viConfSourceACV	Configures the MCM to source AC voltages.
viConfSourceDCI	Configure the MCM to source DC current.
viConfSourceDCV	Configure the MCM to source DC voltage.
viReset	Resets the MCM to its power-on state.
viSet	Sends current setup information to the MCM.
viOutputACV	Configures the MCM to source AC voltages.
viSetCompare	Selects the comparator's trigger threshold voltage of the MCM.
viOutputDCI	Configure the MCM to source DC current.
viOutputDCV	Configure the MCM to source DC voltage.
viSetSourceACV	Configures the MCM to source AC voltage.
viSetSourceDCI	Configures the MCM to source DC current.
viSetSourceDCV	Configures the MCM to source a DC voltage.

Table 4-16. msginst Actions

Actions	Descriptions
34941SwClose	This action send a "CLOSE" command to the 34941A RF Switch card to close the specified switch.
HP53131SetTimeOut	Sets the Agilent HP53131 Counter operation time out value (in milliseconds).
PMCal	This action causes the power meter to perform calibration sequence on the specified channel.
PMCalRef	This action causes the power meter to perform a calibration on the specified channel.
PMCalZero	This action causes the power meter to perform its zeroing routine on the specified channel. Zeroing takes approximately 10 seconds.
PMGetResult	Low level action that sends a query command to a Power Meter to get the result. Execute command: INIT and FETCH.

Table 4-17. Power Supply Actions

Actions	Descriptions
n57xxConfigureCurrentLimit	This action configures the power supply current limit and its behavior when the output current is equal or greater than the specified limit.
n57xxGetPsInfo	Get instrument information.
n57xxMeasureCurrent	This action measures the power supply output current upon completion of all pending operations. Some delay may be required before calling this action to allow current settling time. Delay time may vary depending on output loading.
n57xxMeasureVoltage	This action measures the power supply output voltage upon completion of all pending operations. Some delay may be required before calling this action to allow current settling time. Delay time may vary depending on output loading.
n57xxMsgQuery	Sends a SCPI query string to the power supply and returns the result.
n57xxMsgSend	Sends a SCPI command string to the power supply.
n57xxOutputEnable	This action disables or enables the power supply output.
n57xxQueryCurrentLimitMax	This action returns the maximum programmable current limit.
n57xxQueryOVPLimitMax	This action returns the maximum overvoltage protection level.
n57xxQueryState	This action returns the power supply output state.
n57xxQueryVoltageLevelMax	This action returns the maximum programmable voltage level.
N57xxReset	This action sets the power supply to the *RST state.
n57xxResetOutputProtection	This action clears the latch that disables the output when an over- voltage/current/temperature is detected. All conditions that generate fault must be removed before latch can be cleared. The output is then restored to state it was in before the fault.
n57xxSelfTest	This action sets the power supply to run self test, waits for test completion, and queries the results.
n57xxSetMinVoltageLimit	This action sets the limit on how low the voltage can be programmed. The min voltage limit cannot be programmed above the voltage limit of the power supply.
n57xxSetOutputVoltage	This action sets the power supply output voltage level.
n57xxSetOVPLimit	This action sets the overvoltage protection level.
n57xxSetPONState	This action sets the Power ON state of the power supply.
n57xxWaitForOperationComplete	This action waits and returns when all pending operations have completed or when the MaxTimeMilliseconds has exceeded.
psConnect	Enables output relays of the selected Power Supply.
psDisconnect	Disconnects output relays from the selected Power Supply.
psSetVI	Selects the voltage and current parameters of the power supplies. This action also calls the psSet and psIsSet actions.
psGetCurrent	Reads and returns the output current of the Power Supply.
psGetStatus	Reads and returns Power Supply error status from last "psSet" or "psIsSet" Action.
psGetVoltage	Reads and returns the output voltage of the Power Supply.
psIsSet	Waits until the Power Supply is ready for output. (Action returns a true when ready.)

Table 4-17. Power Supply Actions (continued)

Actions	Descriptions
psPbMsgQuery	Sends a SCPI query string to the power supply and returns the result.
psPbMsgSend	Sends a SCPI command string to the power supply.
psProgVI	Programs and outputs the voltage/current of the Power Supply. This action combines the psConfVI, psConnect, psSet, and psISet actions.
psProgVI_Ex	Programs and outputs the voltage/current of the Power Supply. This action combines the psConfVI, psConnect, psSet, and psISet actions.
psReset	Resets the Power Supply to its power-on state.
psSet	Sends configuration information to the Power Supply.
psSetReturnMode	Sets sequencer behavior on Power Supply time-out errors. The routine will also determine if it should wait for command completion.
psSetTimeout	Sets Power Supply operation time out value (in milliseconds).
psSelfTest	Runs a selftest on the Agilent Power Supply specified.
psConfTrigIn	Selects the trigger input of the Power Supplies.
psTrigger	Send a software trigger to the Power Supply. This action is **NOT IMPLEMENTED** yet; current power supplies use only internal triggers.
psConfVI	Selects output voltage and current of the Power Supplies.

Table 4-18. Serial Interface Actions

Actions	Descriptions
scommBreak	Sends or clears a break signal.
scommFlush	Flushes the specified port.
scommGetConfig	Returns the current configuration of the given serial port.
scommEscapeFunc	Performs extended functions on the serial port communications device.
scommGetComPort	This action returns 1 if com port is detected.
scommSpawnSendBytes	Sends user specified bytes of data to the Serial Port.
scommWaitForSendBytes	Sends user specified bytes of data to the Serial Port.
scommReceive	Reads the specified number of characters from the Serial Port and checks those numbers against the "compareString" parameter.
scommReceiveBytes	Reads the Serial Port.
scommReset	Resets the Serial Port.
scommSend	Sends data to the Serial Port.
scommSet	Sends current setup information to the Serial Port.
scommSendBytes	Sends user specified bytes of data to the Serial Port.

Table 4-18. Serial Interface Actions

Actions	Descriptions
scommStop	Flushes the transmit buffer used for the Serial Port and resets the number of bytes sent and received to 0.
scommTransmit	Writes to and reads from the Serial Port the number of bytes requested.
scommTransmitBytes	Writes to and reads from the Serial Port the number of bytes requested.

Table 4-19. Automotive Serial Protocol Actions

Actions	Descriptions
CANGetInfo	Gets information on current state of J1939 or CAN interface including Bit Rate, Mode and Header Style (11 or 29 bit version).
CANRead	Reads CAN bus. Either 11 or 29 bit ID field, as well as up to 8 byte data field is returned in 0xdd... format, where d is a valid hexadecimal digit.
CANSetBitRate	Sets J1939 or CAN Interface bit rate in Bits/Sec.
CANSetIdMode	Sets Header Style (11 or 29 bit version).
CANSetReadFilter	Setup the CAN read filter. Either 11 or 29 bit ID field may be used for set filter with this action. Enter filter value and mask value in 0xdd... format where d is a valid hexadecimal digit. Mask can be used to select bits to be used for comparison and thus message accept filter will pass range of messages.
CANSetTermination	Sets 124 ohm termination resistor across CAN bus (from CAN+ to CAN–). Used to properly terminate CAN bus.
CANSetUp	Performs general purpose set-up of CAN/J1939 protocol.
CANWrite	Writes to CAN bus. Either 11 or 29 bit ID field, as well as up to 8 byte data field is entered in 0xdd... format where d is a valid hexadecimal digit.
CANWriteAlternate	Writes to CAN bus. Either 11 or 29 bit ID field, as well as up to 8 byte data field is entered in 0xdd... format, where d is a valid hexadecimal digit. This action varies from CANWrite only in data format. The parameter HexStringID has been removed and is now expected to be inserted into HexStringData as the first 2 or 4 bytes, for the 11 or 29 bit identifier, respectively.
ISO9141SetByteTime	Sets the time delay between bytes in an ISO9141 message. Specified time is in mS.
ISO9141EchoKeyword	Provides detection for 2 keywords and echoes the complement keyword #2 as specified in ISO9141-2.
ISO9141FRead	Reads ISO9141 formatted data.
ISO9141FReadConfig	Constructs ISO9141 frame for the "ISO9141FRead" and "ISO9141Read" actions. Header data is entered in 0xdd… format, where d is a valid hexadecimal digit.
ISO9141FWrite	Writes formatted ISO9141 data. Data is entered in 0xdd… format, where d is a valid hexadecimal digit.
ISO9141FWriteConfig	Configuration parameters for ISO9141 formatted write action.
ISO9141GetInfo	Gets information on the configuration and mode of operation of the ISO9141 serial interface device.
ISO9141InitSeq	Initializes the ISO9141 sequence. This action writes one 8 bit word at 5 baud to initiate communications with ECU. It waits until the last bit of the byte is transmitted before it continues.

Table 4-19. Automotive Serial Protocol Actions (continued)

Actions	Descriptions
ISO9141Read	Reads unformatted ISO9141 data. Data returned in 0xdd... format where 'd' is a valid hexadecimal digit. All bytes contained in message are returned by this action including header, length and checksum (if present).
ISO9141SetBitRate	Sets the bit rate of the ISO9141 serial protocol interface. Enter data in Bit/Sec.
ISO9141SetUp	Provides general purpose set-up of ISO9141 protocol.
ISO9141Write	Writes unformatted data to the ISO9141 protocol. Data is entered in 0xdd… format where d is a valid hexadecimal digit.
J1850BlockTransfer	Reads data from a file and sends the data along with header data to the J1850 device.
J1850GetConfig	Gets information on the configuration and mode of operation of the J1850 serial interface device.
J1850Read	Reads unformatted J1850 data. Data returned in 0xdd… format, where d is a valid hexadecimal digit. All bytes contained in message are returned by this action including header, length and checksum (if present).
J1850SetConfig	Sets information on the configuration and mode of operation of the J1850 serial interface device.
J1850SetSourceAddr	Sets the physical ID of the sender of the message. This determines which messages will be received.
J1850Write	Writes unformatted J1850 data. Data is entered in 0xdd… format where d is a valid hexadecimal digit. CRC is automatically appended to end of message.
J1939Read	Reads J1939 bus. Either 11 or 29 bit ID field, as well as up to 8 byte data field is returned in 0xdd... format where d is a valid hexadecimal digit.
J1939Write	Writes unformatted J1939 data. Either 11 or 29 bit ID field, as well as up to 8 byte data field is entered in 0xdd… format where d is a valid hexadecimal digit.
mComClrRxBuff	Clears the receive buffer of the associated serial protocol interface.
mComConfigGroup	Sets information on the configuration and mode of operation of the group message feature. This feature allows for a set of messages to be sent out at regular intervals.
mComGetBoardInfo	Returns results of Multicom serial interface board self test, board serial number, board version number and firmware/hardware id numbers for on-board functions & Physical Interface Module (PIM) slots.
mComMask	Converts string to real number. Designate numbers to convert by setting mask character to 'n'. Designate hex digits to ignore by placing a '*'.
mComReset	Resets the Multicom box.
mComSetBitRate	Sets the bit rate of the associated serial protocol interface.
mComStartGroup	Starts group messages that were previously defined by mComConfigGroup.
mComStopGroup	Stops any group messages that were previously started by mComStartGroup.

Table 4-19. Automotive Serial Protocol Actions (continued)

Actions	Descriptions
softCANDefineID	Defines a Standard or Extended identifier. Before a message may be transmitted or received on a channel, its ID must be defined. Once reading and writing has begun, no more ID's may be defined and an error will be raised. Used in Dynamic Object Buffer Mode. Calling SoftCANSetup or performing a global reset will clear the list of all defined ID's. If this is done, ID's must again be defined before they may be read or written after the setup or reset.
softCANGetFifoLevels	This function will return the number of transmit jobs in the transmit FIFO waiting to be transmitted by the interface as well as the number of events in the receive FIFO waiting to be read. The ReturnCode parameter will indicate if any errors occurred.
softCANGetSerialNumber	This action returns the serial number of softing CAN card.
softCANRead	Read data on the specified CAN channel. Used in Dynamic Object Buffer Mode. The expected ID to be read must be specified. The action will return a code indicating whether data was read, data was not read, or there was data in the buffer that was not read and overwritten by new data (DATA_OVERRUN).
softCANReadFifo	This function will read a data frame from the FIFO. If more than one channel is being used to receive messages, the channel return value should be checked to verify which channel this messages came from. The Softing card shares the singular FIFO with both channels. Either channel can be passed in as the 'CANChannel' parameter. The ReturnCode parameter will indicate if any errors occurred.
softCANReadFifoAlt	This function will read a data frame from the FIFO. This is similar to softCANReadFifo except that it returns integers instead of strings. If more than one channel is being used to receive messages, the channel return value should be checked to verify which channel this messages came from. The Softing card shares the singular FIFO with both channels. Either channel can be passed in as the 'CANChannel' parameter. The ReturnCode parameter will indicate if any errors occurred.
softCANResetFifos	This function will reset the specified fifo. Since the singular FIFO is shared between the 2 channels, either channel can be passed in. The ReturnCode parameter will indicate if any errors occurred.
softCANSetBaudrate	Set baudrate on CAN channels.
softCANSetup	Setup the softing CAN-AC2-PCI card.
softCANStartGroupMsg	Starts group messaging (periodic or cyclic messaging) on one channel of the Softing CAN-AC2-PCI card.
softCANStopGroupMsg	Stops group messaging for one ID. If you wish to stop group messaging for all ID's, enter "ALL" (no quotes) in the HexStringID parameter. Used in Dynamic Object Buffer Mode.
softCANSwitchSpeed	Switch between highspeed or lowspeed on CAN channels.
softCANWrite	Writes one ID and data to the CAN channel specified.
softCANWriteFifo	This function transmits a data frame onto the specified channel (CanChannel).
softCANWriteFifoAlt	This function transmits a data frame onto the specified channel.

Table 4-20. SLU and Switching Actions

Actions	Descriptions
hardResetSU	Reset all Agilent E6198/E6218A switch units and all cards contained therein.
GetLoadCardID	Reads the configuration registers of the load box. (0…255)
loadCardGetInfo	Returns information about the loadcard referenced by the instrument handle parameter.
loadCardReset	Routine to reset a given load card. This action will reset all relays to the open position and wait for the relays to settle.
loadCardSwitchDiffAmp	Only for the E6210A load card. Controls the switching of the differential amplifier into the ISense path. Use a switching action to switch in the Current Sense node prior to using this action. Subsequent switching of any ISense paths on this card will remove the differential amp from the path. This includes the switching cleanup of the ISense path. Default state does not have the differential amp in the ISense path.
loadCardTest	Validates that a relay is set to a given position.
SUreset	Perform a soft reset on the Agilent Technologies E6198/E6218A switch unit. This will reset the digital inputs, and cause the digital outputs and DAC channels to go to their reset state as specified by the user in the module parameter block.
switching	This action connects / disconnects up to 10 switching paths.

Table 4-21. Voltage/Current Source Actions

Actions	Descriptions
m9186GetCardInfo	This action retrieves the card information. This action should be used after initializing the card.
m9186GetInterLockStatus	This function gets the interlock status of the V/I card.
m9186Reset	This function resets the V/I card. The function returns zero if successful.
m9186SetHighCurrent	This function sets the high range current.
m9186SetHighVoltage	This function sets the high range voltage.
m9186SetLowCurrent	This function sets the low range current.
m9186SetLowVoltage	This function sets the low range voltage.

Table 4-22. Spectrum Analyzer Actions

Actions	Descriptions
esaCalGetPathLoss	This action is use to get the cal loss value of the path set. **Notes:** Use after esaCalReadData. Pass in 4 frequency value from esaCalReadData to the freq value in the parameter. Insert RFTC # and Test Freq to retrieve the path loss value.
esaCalReadData	This action is use to read back the cal data store from action esaCalStoreData. 1. Freq1: Cal Data store in Freq1. 2. Freq1: Cal Data store in Freq2. 3. Freq1: Cal Data store in Freq3. 4. Freq1: Cal Data store in Freq4.
esaCalStoreData	This action is use to store the cal path loss data. 1. NoOfFreqs: The number of freq points cal. Range 1 to 4. 2. calData: The cal value obtained from measurement. 3. Freq1, Freq2, Freq3, Freq4: Frequency points to cal.
esaDemod	**High-Level Action:** To set the demodulation parameter in ESA.
esaGetTraceData	This method transfers data from the instrument to the controller. There are normally 401 points in a trace.
esaID	Gets the *IDN? string from the ESA
esaMarkerPeakSearch	**Low-Level Action:** The method uses the spectrum analyzer's marker peak search functions.
esaMarkerSetMode	**Low-Level Action:** This method selects the type of markers to activate.
esaMeasFreqDev	**High-Level Action:** Measures Frequency deviation E4402B ESA. **Usage:** This action is self contained; there is no need to call other actions for this measurement. **Requires:** Provide signal to ESA. **See Also:** esaSetFreqCenter, esaSetFreqSpan, esaSetBandwidth, esaSetAmpliRefLev, esaSweepSetTime. **Instrument:** ESA
esaMeasureSetMode	**Low-Level Action:** This method stops the current measurement and sets up the instrument for the specified measurement using the factory default instrument settings. This will always set the instrument in single sweep mode and place the measurement in the idle state.
esaOBWConfig	**Low-Level Action:** This method sets up the occupied bandwidth measurement operation.
esaOBWMeas	**Low-Level Action:** This method retrieves the scalar results of occupied bandwidth and transmit frequency error from the instrument, based on the method of retrieval (measure, read, or fetch).

Table 4-22. Spectrum Analyzer Actions (continued)

Actions	Descriptions
esaReset	**Low-Level Action:** Resets the Agilent ESA to its power-on state. **Usage:** Call this action when the ESA configuration is unknown and the ESA is to be configured into a known state.
esaSelfTest	Call this action to perform self test for the ESA.
esaSetAmpliAttenuation	This method sets the spectrum analyzer's RF Input Attenuator value. Setting the auto to 0 (Auto off), the amplitude attenuation uncouples the attenuator from the amplitude reference level. To couple the attenuator to the reference level, set auto to 1 (Auto on).
esaSetAmpliRefLev	**Low-Level Action:** To set the amplitude reference level.
esaSetAmpliScale	**Low-Level Action:** To set the display vertical scale format as either logarithmic or linear.
esaSetAverage	**Low-Level Action:** This method sets the spectrum analyzer's average type and count. Then, turns the spectrum analyzer's averaging on or off.
esaSetBandwidth	**Low-Level Action:** This action allow user to set the Bandwidth of the spectrum analyzer.
esaSetFreqCenter	This action sets the spectrum analyzer's center frequency. **For E4402B ESA:** Range 9 kHz ~ 3 GHz
esaSetFreqSpan	Sets the freq span for ESA.
esaSetFreqStartStop	This action sets the spectrum analyzer's start and stop frequencies. **For E4402B ESA:** Range 9 kHz ~ 3 GHz
esaSweepSetMode	**Low-Level Action:** This method selects whether the trigger system is continuously initiated or not. This corresponds to continuous measurement or single measurement operation.
esaSweepSetTime	**Low-Level Action:** This method specifies the time in which the instrument sweeps the displayed frequency range.
esaTraceSetMode	**Low-Level Action:** This method selects the spectrum analyzer's trace mode for the selected trace.
esaTriggerSetSource	**Low-Level Action:** This method selects the spectrum analyzer's source (or type) for triggering used to start a measurement. Use esaTriggerSetVideoLevel to set the video trigger level.
esaTriggerSetVideo	**Low-Level Action:** This method sets the spectrum analyzer's video trigger level for when the trigger source is set to video.

Table 4-23. Signal Generator Actions

Actions	Descriptions
esgCustomConfigFilter	This action configures the Custom Digital Modulation Format pre--modulation filter parameters, including selecting user defined FIR filter stored in the instrument's memory.
esgCustomConfigPattern	**High-Level Action:** This method is used to configure Custom Digital Modulation Format pattern data format
esgCustomMode	**Low-Level Action:** To set the custom mode to on/off.
esgCustomModifyModulation	**High-Level Action:** This method customizes the Custom Digital Modulation Format current modulation type.
esgCustomModifyStandard	**High-Level Action:** This method is used to configure Custom Digital Modulation Format standard parameters, modifying the following aspects of the standard transmission: phase polarity, user-defined Differential Encoder state, symbol rate, and bursted RF signal shape.
esgFSKConfig	**High-Level Action:** This action set the FSK configuration for the ESG.
esgID	Gets the *IDN? string from the ESG.
esgReset	**Low-Level Action:** Resets the Agilent ESG to its power-on state. Usage: Call this action when the ESG configuration is unknown and the ESG is to be configured into a known state.
esgSelfTest	Call this action to perform self test for the ESG.
esgSetFrequency	**Low-Level Action:** This function sets the signal generator's CW (Continuous Wave) output frequency.
esgSetPower	**Low-Level Action:** This function sets the RF output power.
esgSetPowerStatus	**Low-Level Action:** This function sets the operation state of the signal generator's RF output.

Index