NI-SWITCH LabVIEW Reference Help

June 2007, 372293A-01

Use the VIs on the NI-SWITCH palette to build the block diagram.

To comment on National Instruments documentation, refer to the <u>National Instruments Web site</u>.

© 2007 National Instruments Corporation. All rights reserved.

niSwitch Initialize With Topology

Returns a session handle used to identify the switch module in all subsequent NI-SWITCH calls and sets the topology of the switch module.

niSwitch Initialize with Topology creates a new IVI instrument driver session for the switch module specified in the **resource name** parameter. NI-SWITCH uses the topology specified in the **topology name** parameter and overrides the topology specified in MAX.

By default, the switch module is reset to a known state.

Enable <u>simulation</u> by specifying the topology and setting the **simulate** parameter to True.





resource name specifies the <u>resource name</u> of the switch module to initialize.

Syntax:

Optional fields are shown in square brackets ([]).

MAX Configuration Pane Category	Valid Syntax
NI-DAQmx Devices	DAQmxDeviceName
Traditional NI-DAQ (Legacy) Devices	SCXI[chassis ID]::slot number
PXI System	PXI[bus number]::device number



Tip IVI logical names are also valid for **resource name**.

Default values for optional fields:

- chassis ID = 1
- bus number = 0

The following table lists example resource names:

Resource Name	Description	
SC1Mod3	NI-DAQmx module in chassis "SC1" Slot 3	
MySwitch	NI-DAQmx module renamed to "MySwitch"	

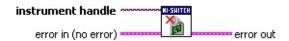
niSwitch Close

Terminates the NI-SWITCH session and all of its attributes and deallocates any memory resources NI-SWITCH uses.

If **error in** describes an error that occurred before the call to this VI, niSwitch Close still attempts to close the session. However, if the attempt fails, this VI returns the error information that was passed in from **error in**.



Note After calling niSwitch Close, you cannot use the NI-SWITCH again until you call <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u>.



- instrument handle identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u>.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurs before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - status is TRUE (X) if an error occurred before this VI ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI ran. The default is FALSE.
 - code is the error or warning code. The default is 0. If status is TRUE, code is a non-zero error code. If status is FALSE, code is 0 or a warning code.
 - source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error

niSwitch Connect Channels

Creates a path between channel 1 and channel 2. NI-SWITCH calculates and uses the shortest path between the two channels. Refer to <u>Setting Source and Configuration Channels</u> for information about channel usage types. <u>Details</u>



instrument handle identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u>.

channel 1 specifies one of the channel names of the desired path. Pass the other channel name as the channel 2. Refer to Devices for valid channel names for the switch module.

Examples of valid channel names:

ch0, com0, ab0, r1, c2, cjtemp

channel 2 specifies one of the channel names of the desired path. Pass the other channel name as the channel 1. Refer to Devices for valid channel names for the switch module.

Examples of valid channel names:

ch0, com0, ab0, r1, c2, citemp

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurs before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

status is TRUE (X) if an error occurred before this VI ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI ran. The default is FALSE.

code is the error or warning code. The default is 0. If status is TRUE, code is a non-zero error code. If status is

niSwitch Connect Channels Details



Note Paths are bidirectional. For example, if a path exists between channels CH1 and CH2, then the path also exists between channels CH2 and CH1.

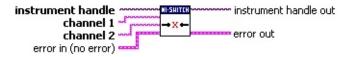
If a path is unavailable between the two channels, NI-SWITCH returns the following errors:

Error	Description
Explicit Connection Exists	The channels have already been explicitly connected by calling either niSwitch Connect Channels or niSwitch Set Path.
Is Configuration Channel	One of the channels is a configuration channel. Error elaboration contains information about which of the two channels is a configuration channel.
Attempt To Connect Sources	Both channels are connected to a source. Error elaboration contains information about the sources connected to channel 1 and 2.
Cannot Connect To Itself	Channel 1 and channel 2 are identical.
Path Not Found	No paths between channel 1 and channel 2 are available.
Resource In Use	Channel 1 and/or channel 2 is in use. This error often occurs when one of the channels is set as a configuration channel and is in use or if one of the channels is a common multiplexer channel in use.

niSwitch Disconnect Channels

Breaks the path between two channels created with <u>niSwitch Connect</u> Channels or niSwitch Set Path.

If no connection exists between the channels, NI-SWITCH returns an error.



instrument handle identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u>.

channel 1 specifies one of the channel names of the path to break. Pass the other channel name as the channel 2. Refer to Devices for valid channel names for the switch module.

Examples of valid channel names: ch0, com0, ab0, r1, c2, cjtemp

channel 2 specifies one of the channel names of the path to break. Pass the other channel name as the channel 1. Refer to Devices for valid channel names for the switch module.

Examples of valid channel names: ch0, com0, ab0, r1, c2, citemp

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurs before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

status is TRUE (X) if an error occurred before this VI ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI ran. The default is FALSE.

code is the error or warning code. The default is 0. If status is TRUE, code is a non-zero error code. If status is

niSwitch Wait For Debounce

Pauses until all created paths have settled. Call this VI before <u>niSwitch</u> <u>Get Relay Count</u> to ensure an accurate relay count.



- instrument handle identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u>.
- maximum time specifies the maximum length of time to wait for all relays in the switch module to activate or deactivate. If the specified time elapses before all relays activate or deactivate, a timeout error is returned.

Default value: 5000 (milliseconds)

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurs before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - status is TRUE (X) if an error occurred before this VI ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI ran. The default is FALSE.
 - code is the error or warning code. The default is 0. If status is TRUE, code is a non-zero error code. If status is FALSE, code is 0 or a warning code.
 - source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- instrument handle out identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u> and used for all subsequent NI-SWITCH calls.

arrar aut contains arrar information. If arrar in indicates that an

niSwitch Set Path

Connects two channels by specifying an explicit path in **path list**. This VI is particularly useful where path repeatability is important, such as in calibrated signal paths. If this is not necessary, use <u>niSwitch Connect Channels</u>.

To obtain the exact path for a given connection, use <u>niSwitch Get Path</u>.



instrument handle identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u>.

path list—A string composed of comma-separated paths between channel 1 and channel 2. The first and last names in the path are the path endpoints. All other channels in the path are configuration channels.

Example of a valid path list string:

ch0->com0, com0->ab0.

In this example, com0 is a configuration channel.

Obtain the path list for a previously created path with <u>niSwitch</u> <u>Get Path</u>.

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurs before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

status is TRUE (X) if an error occurred before this VI ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI ran. The default is FALSE.

code is the error or warning code. The default is 0. If status is TRUE, code is a non-zero error code. If status is FALSE, code is 0 or a warning code.

niSwitch Can Connect Channels?

Verifies that a path between **channel 1** and **channel 2** can be created.

If a path is possible in the switch module, the availability of that path is returned given the existing connections. If the path is possible but in use, an Implicit Connection Exists warning is returned.



instrument handle identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u>.

channel 1 specifies one of the channel names of the desired path. Pass the other channel name as channel 2. Refer to Devices for valid channel names for the switch module.

Examples of valid channel names: ch0, com0, ab0, r1, c2, citemp

channel 2 specifies one of the channel names of the desired path. Pass the other channel name as channel 1. Refer to Devices for valid channel names for the switch module.

Examples of valid channel names:

ch0, com0, ab0, r1, c2, cjtemp

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurs before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

status is TRUE (X) if an error occurred before this VI ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI ran. The default is FALSE.

code is the error or warning code. The default is 0. If status is TRUE. code is a non-zero error code. If status is

niSwitch Disconnect All Channels

Breaks all existing paths.

If the switch module cannot break all paths, a warning is returned.

If an **error in** parameter describes an error that occurred before the call to this VI, NI-SWITCH still attempts to break all connections and returns the error information that was passed in from **error in**.

Relays closed with <u>niSwitch Relay Control</u> are also disconnected.



- instrument handle identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u>.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurs before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - status is TRUE (X) if an error occurred before this VI ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI ran. The default is FALSE.
 - code is the error or warning code. The default is 0. If status is TRUE, code is a non-zero error code. If status is FALSE, code is 0 or a warning code.
 - source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- instrument handle out identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u> and used for all subsequent NI-SWITCH calls.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains

niSwitch Switch Is Debounced?

Indicates if all created paths have settled by returning the value of the **is debounced** Property.



- instrument handle identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u>.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurs before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - status is TRUE (X) if an error occurred before this VI ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI ran. The default is FALSE.
 - code is the error or warning code. The default is 0. If status is TRUE, code is a non-zero error code. If status is FALSE, code is 0 or a warning code.
 - source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- instrument handle out identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u> and used for all subsequent NI-SWITCH calls.
- is debounced indicates whether all created paths have settled.

Status	Description	
FALSE	All created paths have not settled	
TRUE	All created paths have settled	

error occurred before this VI or function ran error out contains

niSwitch Get Path

Returns a string that identifies the explicit path created with <u>niSwitch</u> <u>Connect Channels</u>. Pass this string to <u>niSwitch Set Path</u> to establish the exact same path in future connections.

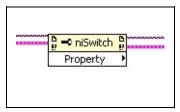
In some cases, multiple paths are available between two channels. When you call <u>niSwitch Connect Channels</u>, NI-SWITCH selects an available path; however, the driver may not always select the same path through the switch module.

This VI only returns those paths explicitly created by <u>niSwitch Connect Channels</u> or <u>niSwitch Set Path</u>. For example, if you connect channels CH1 and CH3, and then channels CH2 and CH3, an explicit path between channels CH1 and CH2 does not exist, and an error is returned.



- instrument handle identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u>.
- channel 1 specifies one of the channel names of the desired path. Pass the other channel name as the channel 2 parameter. Refer to Devices for valid channel names for the switch module. Examples of valid channel names: ch0, com0, ab0, r1, c2, cjtemp
- channel 2 specifies one of the channel names of the desired path. Pass the other channel name as the channel 1 parameter. Refer to Devices for valid channel names for the switch module. Examples of valid channel names: ch0, com0, ab0, r1, c2, cjtemp
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurs before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error

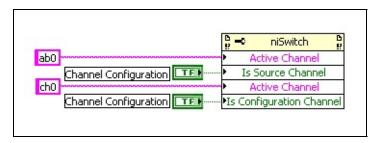
niSwitch Property Node



Gets (reads) and/or sets (writes) properties of the NI-SWITCH driver. For a complete list of NI-SWITCH properties, refer to NI-SWITCH Properties.

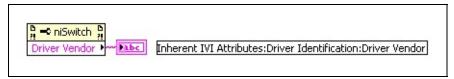
Channel Based Properties

Some NI-SWITCH properties are channel based or apply to a specific channel. When a property is channel based, you must specify an active channel before setting or getting properties. In the following example, ab0 is set as a source channel and ch0 is set as a configuration channel. Both properties are channel based.



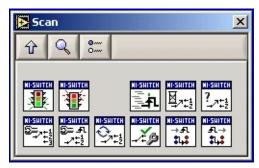
Non-Channel Based Properties

To set or get NI-SWITCH properties that are not channel based, you do not need to specify an active channel. The following example shows a property node used to get the value of the <u>Driver Vendor</u> property.



Scan

Use the VIs located on the **NI-SWITCH»Scan** palette to configure a scan. **Click the icons for VI and function descriptions.**



niSwitch Abort Scan
niSwitch Commit
niSwitch Configure Scan List
niSwitch Configure Scan Trigger
niSwitch Initiate Scan

niSwitch Route Scan Advanced Output niSwitch Route Trigger Input niSwitch Send Software Trigger niSwitch Set Continuous Scan niSwitch Switch Is Scanning? niSwitch Wait For Scan To Complete

niSwitch Initiate Scan

Commits the configured <u>scan list</u> and trigger settings to hardware and initiates the scan. If <u>niSwitch Commit</u> was called earlier, this VI only initiates the scan and returns immediately.

Once the scanning operation begins, you cannot perform any other operations other than <u>niSwitch Abort Scan</u> or <u>niSwitch Send Software Trigger</u>, as well as retrieval of attributes. All other VIs return a Scan In Progress error.

To stop the scanning operation, call <u>niSwitch Abort Scan</u>.



- instrument handle identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u>.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurs before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - status is TRUE (X) if an error occurred before this VI ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI ran. The default is FALSE.
 - code is the error or warning code. The default is 0. If status is TRUE, code is a non-zero error code. If status is FALSE, code is 0 or a warning code.
 - source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- instrument handle out identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u> and used for all subsequent NI-SWITCH calls.

niSwitch Abort Scan

Aborts the scan in progress.

Initiate a scan with <u>niSwitch Initiate Scan</u>.

If the switch module is not scanning, a No Scan In Progress error is returned.

If **error in** describes an error that had occurred before calling this VI, the VI still attempts to abort the scan. However, if the attempt fails, the VI returns the error information that was passed in from the **error in**.



Note If you are using a legacy resource descriptor (SCXI:: or PXI::) and you abort a scan, the switch module returns to a disconnect all state (equivalent to calling <u>niSwitch Disconnect All Channels</u>). If you are using an NI-DAQmx resource descriptor (DAQmx device name), the switch module returns to the state it was in before the scan was initiated.



- instrument handle identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u>.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurs before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - status is TRUE (X) if an error occurred before this VI ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI ran. The default is FALSE.
 - code is the error or warning code. The default is 0. If status is TRUE, code is a non-zero error code. If status is FALSE, code is 0 or a warning code.
 - source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced

niSwitch Send Software Trigger

Sends a software trigger to the switch module specified in the NI-SWITCH session. When the **trigger input** parameter is set to **Software Trigger Function** using <u>niSwitch Configure Scan Trigger</u> or the Trigger Input property, the scan does not proceed from a semicolon (wait for trigger) until this VI is called.



- instrument handle identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u>.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurs before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - status is TRUE (X) if an error occurred before this VI ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI ran. The default is FALSE.
 - code is the error or warning code. The default is 0. If status is TRUE, code is a non-zero error code. If status is FALSE, code is 0 or a warning code.
 - source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- instrument handle out identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u> and used for all subsequent NI-SWITCH calls.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out

niSwitch Wait For Scan To Complete

Pauses until the switch module stops scanning or the **maximum time** has elapsed and returns a timeout error.



- instrument handle identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u>.
- maximum time specifies the maximum length of time to wait for the switch module to stop scanning. If the specified time elapses before the scan ends, NI-SWITCH returns a timeout error.
 - Default value: 5000 (milliseconds)
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurs before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - status is TRUE (X) if an error occurred before this VI ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI ran. The default is FALSE.
 - code is the error or warning code. The default is 0. If status is TRUE, code is a non-zero error code. If status is FALSE, code is 0 or a warning code.
 - source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- instrument handle out identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u> and used for all subsequent NI-SWITCH calls.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains

niSwitch Switch Is Scanning?

Indicates the status of the scan.

instrument handle	MI-SHITCH	instrument handle out
error in (no error)	? 2+1	is scanning
error in (no error)	<u>-√ +2</u>	error out

- instrument handle identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u>.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurs before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - status is TRUE (X) if an error occurred before this VI ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI ran. The default is FALSE.
 - code is the error or warning code. The default is 0. If status is TRUE, code is a non-zero error code. If status is FALSE, code is 0 or a warning code.
 - source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- instrument handle out identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u> and used for all subsequent NI-SWITCH calls.
- is scanning indicates whether the switch module is scanning. The driver returns the value of the is scanning property.

TRUE	The switch module is scanning.
FALSE (default)	The switch module is idle.

error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains

niSwitch Configure Scan List

Configures the scan list and scan mode used for scanning.

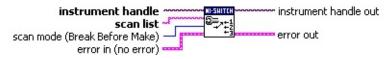
Refer to <u>Devices</u> to determine if the switch module supports scanning.

The scan list is comprised of a list of channel connections separated by semicolons. For example, the following scan list will scan the first three channels of a multiplexer:

com0->ch0; com0->ch1; com0->ch2;

Refer to Scan Lists for more information on scan list syntax.

To see the status of the scan, call either <u>niSwitch Switch Is Scanning?</u> or <u>niSwitch Wait For Scan Complete</u>. Use <u>niSwitch Configure Scan Trigger</u> and <u>niSwitch Initiate Scan</u> to configure the scan trigger and start scan respectively.



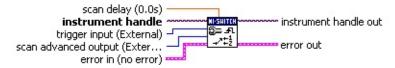
- instrument handle identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u>.
- scan list specifies the scan list to use. NI-SWITCH uses this value to set the Scan List property.
- scan mode specifies how the switch module breaks existing connections when scanning. NI-SWITCH uses this value to set the Scan Mode property. Refer to scan modes for more information. The default value is Break Before Make.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurs before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - status is TRUE (X) if an error occurred before this VI ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI ran. The default is FALSE.

niSwitch Configure Scan Trigger

Configures the scan triggers for the <u>scan list</u> established with <u>niSwitch</u> <u>Configure Scan List</u>.

Refer to **Devices** to determine if the switch module supports scanning.

This VI sets the location where the switch expects to receive an input trigger to advance through the scan list. This VI also sets the location where it generates a scan advanced signal after it completes an entry in the scan list. <u>Details</u>



instrument handle identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize</u>.

scan delay specifies the minimum length of time you want the switch device to wait after it creates a path until it asserts a trigger on the Scan Advanced Output channel. NI-SWITCH uses this value to set the Scan Delay property. The scan delay is in addition to the settling time.

Default value: 0.00 (seconds)

(1)

trigger input specifies the trigger source you want the switch module to use during scanning. NI-SWITCH uses this value to set the Trigger Input property. The default value is External.

The switch module waits for a trigger at the specified location when it encounters a semicolon in the scan list. When the trigger occurs, the switch device advances to the next entry in the scan list and waits for a trigger from the location specified in **trigger input**.

Value	Switch Behavior
Immediate	Immediately processes the next entry in the scan list.
External	Waits for a trigger on the front connector before processing the next entry in the scan list. Same as Front Connector.
Software Trigger Function	Waits until <u>niSwitch Send Software Trigger</u> is called.

niSwitch Configure Scan Trigger Details

For synchronous scanning, set **scan advanced output** to **None** and define **trigger input**. For handshaking, **trigger input** and **scan advanced output** must be set.

Triggers generally originate from the SCXI or PXI trigger bus, the terminals on the front connector of the switch module, or the AUX connector on the rear of the switch module (referred to as the rear connector and supported by SCXI only).

When scanning a module that is not directly cabled to the trigger from its own front or rear connector, use <u>niSwitch Route Trigger Input</u> and/or <u>niSwitch Route Scan Advanced Output</u> to route the triggers on the cabled module to one of the backplane lines. When using these Vis to route triggers to the backplane, specify the backplane trigger location when calling <u>niSwitch Configure Scan Trigger</u> (even if it is the module cabled to the trigger source).

niSwitch Set Continuous Scan

Sets the switch to loop continuously through the <u>scan list</u> or to stop scanning after one pass through the scan list.

Call <u>niSwitch Abort Scan</u> to halt a continuous scan.



- instrument handle identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u>.
- **continuous scan** specifies whether the <u>scan list</u> is run once or continuously during scanning.

TRUE	Loops continuously through the scan list.
FALSE (default)	Stops after one pass through the scan list

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurs before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - status is TRUE (X) if an error occurred before this VI ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI ran. The default is FALSE.
 - code is the error or warning code. The default is 0. If status is TRUE, code is a non-zero error code. If status is FALSE, code is 0 or a warning code.
 - source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- instrument handle out identifies a particular NI-SWITCH session established with niSwitch Initialize With Topology, niSwitch Initialize With Options, or niSwitch Initialize and used for all subsequent NI-SWITCH calls.

niSwitch Commit

Downloads the configured scan list and trigger settings to hardware.

Calling this VI is optional as it is implicitly called during <u>niSwitch Initiate</u> <u>Scan</u>. Use this VI to arm triggers in a given order or to control when hardware operations are performed.



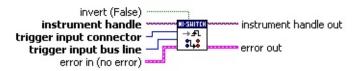
- instrument handle identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u>.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurs before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - status is TRUE (X) if an error occurred before this VI ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI ran. The default is FALSE.
 - code is the error or warning code. The default is 0. If status is TRUE, code is a non-zero error code. If status is FALSE, code is 0 or a warning code.
 - source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- instrument handle out identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u> and used for all subsequent NI-SWITCH calls.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out indicator on the front panel and select Explain Error from the

niSwitch Route Trigger Input

Routes the input trigger from the front or rear connector to a trigger bus line (TTLx). To disconnect the route, call this VI again and select **None** for **trigger bus line** parameter.

This VI should be used when scanning a module in a different NI-SWITCH session that is not directly cabled to the trigger source (<u>multimodule scanning</u> operations).

If this VI is used to route a trigger to a TTL line, specify the TTL line as the **trigger input** in <u>niSwitch Configure Scan Trigger</u>.



- instrument handle identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u>.
- trigger input connector specifies the location of the input trigger source on the switch module. Valid locations are the front and rear connectors. The default Value is Front Connector.
- trigger input bus line specifies the trigger line to route the input trigger. Select **None** to break an existing route.
- **invert** inverts the input trigger signal from falling to rising or vice versa.

Inverts the input trigger signal from falling to rising.
Inverts the input trigger signal from rising to falling.

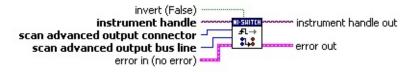
error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurs before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the past node.

niSwitch Route Scan Advanced Output

Routes the Scan Advanced Output trigger from a trigger bus line (TTLx) to the front or rear connector.

Use this VI when handshaking with a switch module that is not directly cabled to the controller. In this module's session, set the **scan** advanced output bus line to a trigger bus line. In the cabled module's session, use this VI to route from the trigger bus line to the front or rear connector of the cabled module.

If this VI is used to route scan advanced from a TTL line, specify the TTL line as the **scan advanced output** destination in <u>niSwitch</u> <u>Configure Scan Trigger</u>.



- instrument handle identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u>.
- scan advanced output connector& is the Scan Advanced Output destination. Valid locations are the front and rear connectors.
- scan advanced output bus line is the trigger line to route the scan advanced output trigger from the front or rear connector. Select None (default) to break an existing route.
- invert (False) inverts the input trigger signal from rising to falling. If TRUE, inverts the input trigger signal from falling to rising.

 Default value: FALSE
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurs before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

Relay

Use the VIs located on the **NI-SWITCH»Relay** palette to control and query individual <u>relays</u>.

Click the icons for VI and function descriptions.

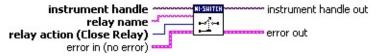


niSwitch Get Relay Count niSwitch Get Relay Position niSwitch Relay Control

niSwitch Relay Control

Controls individual relays of the switch module. When controlling individual relays, the protection offered by setting the usage of <u>source</u> <u>channels</u> and <u>configurations channels</u> is void.

Refer to <u>Devices</u> to determine if the switch module supports <u>individual</u> <u>relay control</u>.



instrument handle identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u>.

relay name specifies the name of the relay.

Examples of valid relay names:

ch0, ab0, 1wire, hlselect

Refer to <u>Devices</u> for a list of valid relay names for the switch module.



abc

Note Use the <u>niSwitch Wait For Debounce</u>, <u>niSwitch Switch Is Debounced?</u>, and the Is Debounced property to determine if the relay has settled.

relay action specifies whether to open or close a given relay.

Relay Open	Opens a given relay.
Relay Closed (default)	Closes a given relay.

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurs before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

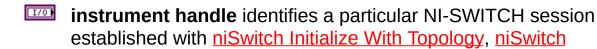
status is TRUE (X) if an error occurred before this VI ran or FALSE (checkmark) to indicate a warning or that no

niSwitch Get Relay Position

Returns the relay position for the relay specified in **relay name**.

Refer to <u>Devices</u> to determine if the switch module supports <u>individual</u> <u>relay control</u>.





relay name specifies the name of the relay.

<u>Initialize With Options</u>, or <u>niSwitch Initialize</u>.

Examples of valid relay names:

ch0, ab0, 1wire, hlselect

abc

Refer to <u>Devices</u> for a list of valid relay names for the switch module.

Note Use the <u>niSwitch Wait For Debounce</u>, <u>niSwitch Switch Is</u> <u>Debounced?</u>, and the Is Debounced property to determine if the relay has settled.

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurs before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

- status is TRUE (X) if an error occurred before this VI ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI ran. The default is FALSE.
- code is the error or warning code. The default is 0. If status is TRUE, code is a non-zero error code. If status is FALSE, code is 0 or a warning code.
- source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

niSwitch Get Relay Count

Returns the number of times the relay has changed from Closed to Open. Relay count is useful for tracking relay lifetime and usage. Call niSwitch Wait for Debounce before this VI to ensure an accurate count.

Refer to <u>Devices</u> to determine if the switch module supports <u>individual</u> <u>relay control</u>.



instrument handle identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u>.

relay name specifies the name of the relay.

Examples of valid relay names:

ch0, ab0, 1wire, hlselect

Refer to <u>Devices</u> for a list of valid relay names for the switch module.



abc

Note Use the <u>niSwitch Wait For Debounce</u>, <u>niSwitch</u>
<u>Switch Is Debounced?</u>, and the **is debounced** property to determine if the relay has settled.

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurs before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

- status is TRUE (X) if an error occurred before this VI ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI ran. The default is FALSE.
- code is the error or warning code. The default is 0. If status is TRUE, code is a non-zero error code. If status is FALSE, code is 0 or a warning code.
- source describes the origin of the error or warning and is,

Utility

Use the VIs located on the **NI-SWITCH»Utility** palette to use additional features of NI-SWITCH.

Click the icons for VI and function descriptions.



niSwitch Get Channel Name niSwitch Read Calibration Data

[OBSOLETE]

<u>niSwitch Get Relay Name</u> <u>niSwitch Reset</u>

<u>niSwitch Initialize</u> <u>niSwitch Write Calibration Data</u>

[OBSOLETE]

niSwitch Initialize With Options

Subpalettes

Other IVI Palette

niSwitch Initialize With Options

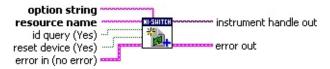
Returns a session handle used to identify the switch module in all subsequent NI-SWITCH calls and optionally sets the initial state of the session.

This VI creates a new IVI instrument driver session for the switch module specified in the **resource name**. If multiple <u>topologies</u> are valid for that device, NI-SWITCH uses the default topology specified in MAX. The topology is also configurable in **option string**.

By default, the switch module is reset to a known state.

Enable <u>simulation</u> in **option string**.

An error is returned if a session to the specified resource exists in another process. The same session is returned if this VI is called twice in the same process for the same resource with the same topology.





resource name specifies the <u>resource name</u> of the switch module to initialize.

Syntax:

Optional fields are shown in square brackets ([]). The default values for optional fields are as follows:

chassis ID = 1

bus number = 0

Configured in MAX Under	Valid Syntax
NI-DAQmx Devices	DAQmxDeviceName
Traditional NI-DAQ (Legacy) Devices	SCXI[chassis ID]::slot number
PXI System	PXI[bus number]::device number



Tip IVI logical names are also valid for the resource name.

The following table provides example resource names.

Resource Name	Description	
SC1Mod3	NI-DAQmx module in chassis "SC1" Slot 3	

niSwitch Initialize

Returns a session handle used to identify the switch module in all subsequent NI-SWITCH calls.

This VI creates a new IVI instrument driver session for the switch module specified in the **resource name** parameter. If multiple topologies are valid for that switch module, NI-SWITCH uses the default topology specified in MAX.

By default, the switch module is reset to a known state.

An error is returned if a session to the specified resource exists in another process. The same session is returned if niSwitch Initialize is called twice in the same process for the same resource with the same topology.





resource name specifies the <u>resource name</u> of the switch module to initialize.

Syntax:

Optional fields are shown in square brackets ([]).

Configured in MAX Under	Valid Syntax
NI-DAQmx Devices	DAQmxDeviceName
Traditional NI-DAQ (Legacy) Devices	SCXI[chassis ID]::slot number
PXI System	PXI[bus number]::device number



Tip IVI logical names are also valid for the resource name.

Default values for optional fields:

chassis ID = 1

bus number = 0

Example resource names:

Resource Name	Description
SC1Mod3	NI-DAQmx module in chassis "SC1" Slot 3
MySwitch	NI-DAQmx module renamed to "MySwitch"
CCVI1::2	Traditional NL DAO (Laggay) modula in abassis

niSwitch Reset

Disconnects all created paths and returns the switch module to the state at initialization. Configuration channel and source channel settings remain unchanged.

If **error in** describes an error that had occurred before calling this VI, this VI still attempts to reset the switch. However, if the attempt fails, this VI returns the error information that was passed in from the **error in** parameter.



- instrument handle identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u>.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurs before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - status is TRUE (X) if an error occurred before this VI ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI ran. The default is FALSE.
 - code is the error or warning code. The default is 0. If status is TRUE, code is a non-zero error code. If status is FALSE, code is 0 or a warning code.
 - source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- instrument handle out identifies a particular NI-SWITCH session established with niSwitch Initialize With Topology, niSwitch Initialize With Options, or niSwitch Initialize and used for all subsequent NI-SWITCH calls.
- error out contains error information. If error in indicates that an

niSwitch Get Channel Name

Returns the **channel string** that is in the channel table at the specified index.

Use this VI in a For Loop to get a complete list of valid channel names for the switch module. Use the Channel Count property to determine the number of channels.



- instrument handle identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u>.
- index is a 1-based index into the channel table. The default value is 1. The maximum value is equal to the value of the Channel Count property.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurs before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - status is TRUE (X) if an error occurred before this VI ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI ran. The default is FALSE.
 - code is the error or warning code. The default is 0. If status is TRUE, code is a non-zero error code. If status is FALSE, code is 0 or a warning code.
 - source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- instrument handle out identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u> and used for all subsequent NI-SWITCH calls.

niSwitch Get Relay Name

Returns the **relay name** that is in the relay list at the specified index.

Use this VI in a For Loop to get a complete list of valid relay names for the switch module. Use the Number of Relays property to determine the number of relays.



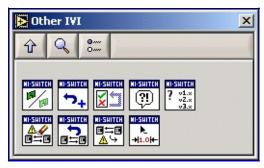
- instrument handle identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u>.
- index is a 1-based index into the channel table. The default value is 1. The maximum value is equal to the value of the Channel Count property.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurs before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - status is TRUE (X) if an error occurred before this VI ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI ran. The default is FALSE.
 - code is the error or warning code. The default is 0. If status is TRUE, code is a non-zero error code. If status is FALSE, code is 0 or a warning code.
 - source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- instrument handle out identifies a particular NI-SWITCH session established with niSwitch Initialize With Topology, niSwitch Initialize With Options, or niSwitch Initialize and used for all subsequent NI-SWITCH calls.
 - **relay name** specifies the name of the relay.

abc

Other IVI

Use the VIs located on the **NI-SWITCH»Utility»Other IVI** palette to perform additional IVI functions.

Click the icons for VI and function descriptions.



niSwitch Disable

niSwitch Reset With Defaults

niSwitch Self-Test

niSwitch Error Message

niSwitch Revision Query

niSwitch Clear Interchange Warnings

niSwitch Reset Interchange Check

niSwitch Get Next Interchange Warning

niSwitch Get Next Coercion Record

niSwitch Disable

Places the switch module in a quiescent state where it has minimal or no impact on the system to which it is connected. This VI disconnects all channels, and any scan in progress is aborted.

If **error in** describes an error that had occurred before calling niSwitch Disable, this VI still attempts to disable the switch. However, if the attempt fails, this VI returns the error information that was passed in from **error in**.



- instrument handle identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u>.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurs before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - status is TRUE (X) if an error occurred before this VI ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI ran. The default is FALSE.
 - code is the error or warning code. The default is 0. If status is TRUE, code is a non-zero error code. If status is FALSE, code is 0 or a warning code.
 - source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- instrument handle out identifies a particular NI-SWITCH session established with niSwitch Initialize With Topology, niSwitch Initialize With Options, or niSwitch Initialize and used for all subsequent NI-SWITCH calls.
- error out contains error information. If error in indicates that an

niSwitch Reset With Defaults

Resets the switch module and applies initial user-specified settings from the logical name used to initialize the session. If the session was created without a logical name, this VI is equivalent to <u>niSwitch Reset</u>.

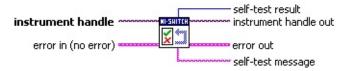
If **error in** describes an error that had occurred before calling niSwitch Reset With Defaults, this VI still attempts to reset the switch. However, if the attempt fails, this VI returns the error information that was passed in from **error in**.



- instrument handle identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u>.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurs before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - status is TRUE (X) if an error occurred before this VI ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI ran. The default is FALSE.
 - code is the error or warning code. The default is 0. If status is TRUE, code is a non-zero error code. If status is FALSE, code is 0 or a warning code.
 - source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- instrument handle out identifies a particular NI-SWITCH session established with niSwitch Initialize With Topology, niSwitch Initialize With Options, or niSwitch Initialize and used for all subsequent NI-SWITCH calls.
- error out contains error information. If error in indicates that an

niSwitch Self-Test

Verifies that NI-SWITCH can communicate with the switch module.



- instrument handle identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u>.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurs before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - status is TRUE (X) if an error occurred before this VI ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI ran. The default is FALSE.
 - code is the error or warning code. The default is 0. If status is TRUE, code is a non-zero error code. If status is FALSE, code is 0 or a warning code.
 - source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- instrument handle out identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u> and used for all subsequent NI-SWITCH calls.
- self-test result returns a value from the switch self-test. 0 equals success and 1 equals failure.
- **self-test message** returns a response string from the switch self-test.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains

niSwitch Error Message

Converts an error code returned by NI-SWITCH into a user-readable string. Generally this information is supplied in **error out** of any NI-SWITCH VI. Use this VI for a static lookup of an error code description.

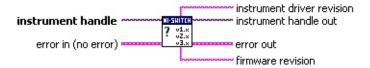


- instrument handle identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u>.
- error code is the status code returned by any NI-SWITCH VI.

 Default value: 0
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurs before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - status is TRUE (X) if an error occurred before this VI ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI ran. The default is FALSE.
 - code is the error or warning code. The default is 0. If status is TRUE, code is a non-zero error code. If status is FALSE, code is 0 or a warning code.
 - source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- instrument handle out identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u> and used for all subsequent NI-SWITCH calls.
- **error message**& is the error information formatted into a string.
- error out contains error information. If error in indicates that an

niSwitch Revision Query

Returns the revision of NI-SWITCH.



- instrument handle identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u>.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurs before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - status is TRUE (X) if an error occurred before this VI ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI ran. The default is FALSE.
 - code is the error or warning code. The default is 0. If status is TRUE, code is a non-zero error code. If status is FALSE, code is 0 or a warning code.
 - source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- instrument handle out identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u> and used for all subsequent NI-SWITCH calls.
- **instrument driver revision** specifies the NI-SWITCH software revision numbers in the form of a string.
- **firmware revision** is currently unsupported.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error

niSwitch Clear Interchange Warnings

Clears the list of current interchange warnings. **Details**

instrument handle	ALCONOMICAL MISSINGS	instrument handle out
error in (no error)		error out

- instrument handle identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u>.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurs before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - status is TRUE (X) if an error occurred before this VI ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI ran. The default is FALSE.
 - code is the error or warning code. The default is 0. If status is TRUE, code is a non-zero error code. If status is FALSE, code is 0 or a warning code.
 - source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- instrument handle out identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u> and used for all subsequent NI-SWITCH calls.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out indicator on the front panel and select Explain Error from the shortcut menu for more information about the error.
 - status is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error

niSwitch Clear Interchange Warnings Details

When developing a complex test system that consists of multiple test modules, it is generally a good idea to design the test modules so that they can run in any order. To do so, ensure that each test module completely configures the state of each instrument it uses. If a particular test module does not completely configure the state of an instrument, the instrument state depends on the configuration from a previously executed test module. Therefore, if you execute the test modules in a different order, the behavior of the instrument and therefore the entire test module is likely to change. This behavior change is generally instrument specific and represents an interchangeability problem.

You can use <u>niSwitch Reset Interchange Check</u> to test for such cases. After you call <u>niSwitch Reset Interchange Check</u>, the interchangeability checking algorithms in the specific driver ignore all previous configuration operations. By calling <u>niSwitch Reset Interchange Check</u> at the beginning of a test module, you can determine whether the test module has dependencies on the operation of previously executed test modules.

To guarantee that <u>niSwitch Get Next Interchange Warning</u> only returns those interchangeability warnings generated after calling <u>niSwitch Reset Interchange Check</u>, clear the list of interchangeability warnings by repeatedly <u>calling niSwitch Get Next Interchange Warning</u> until no interchangeability warnings are returned. If you are not interested in the content of those warnings, call <u>niSwitch Clear Interchange Warnings</u>.

niSwitch Reset Interchange Check

After calling this VI, the interchangeability checking algorithms in the specific driver ignore all previous configuration operations. <u>Details</u>



- instrument handle identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u>.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurs before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - status is TRUE (X) if an error occurred before this VI ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI ran. The default is FALSE.
 - code is the error or warning code. The default is 0. If status is TRUE, code is a non-zero error code. If status is FALSE, code is 0 or a warning code.
 - source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- instrument handle out identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u> and used for all subsequent NI-SWITCH calls.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out indicator on the front panel and select Explain Error from the shortcut menu for more information about the error.
 - status is TRUE (X) if an error occurred or FALSE

niSwitch Reset Interchange Check Details

When developing a complex test system that consists of multiple test modules, it is generally a good idea to design the test modules so that they can run in any order. To do, ensure that each test module completely configures the state of each instrument it uses. If a particular test module does not completely configure the state of an instrument, the instrument state depends on the configuration from a previously executed test module. Therefore, if you execute the test modules in a different order, the behavior of the instrument and therefore the entire test module is likely to change. This behavior change is generally instrument specific and represents an interchangeability problem.

You can use this VI to test for such cases. After you call this VI, the interchangeability checking algorithms in the specific driver ignore all previous configuration operations. By calling this VI at the beginning of a test module, you can determine whether the test module has dependencies on the operation of previously executed test modules.

This VI does not clear the interchangeability warnings from the list of previously recorded interchangeability warnings. To guarantee that niSwitch Get Next Interchange Warning only returns those interchangeability warnings generated after calling this VI, clear the list of interchangeability warnings by repeatedly calling niSwitch Get Next Interchange Warning until no interchangeability warnings are returned. If you are not interested in the content of those warnings, call niSwitch Clear Interchange Warnings.

niSwitch Get Next Interchange Warning

Returns the interchangeability warnings associated with the IVI session. This VI retrieves and clears the oldest instance in which the class driver recorded an interchangeability warning. Interchangeability warnings indicate that using your application with a different instrument might cause different behavior.

The driver performs interchangeability checking when the Interchange Check property is set to TRUE.

The VI returns an empty string for interchange warning if no interchangeability warnings remain for the session.

In general, the instrument driver generates interchangeability warnings when an property that affects the instrument behavior is in a state that you did not specify.



- instrument handle identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u>.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurs before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - status is TRUE (X) if an error occurred before this VI ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI ran. The default is FALSE.
 - code is the error or warning code. The default is 0. If status is TRUE, code is a non-zero error code. If status is FALSE, code is 0 or a warning code.
 - source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

niSwitch Get Next Coercion Record

Returns the coercion information associated with the IVI session. This VI retrieves and clears the oldest instance in which the instrument driver coerced a value you specified to another value.

If you set the Record Coercions property to TRUE, NI-SWITCH keeps a list of all coercions it makes on ViInt32 or ViReal64 values you pass to instrument driver VIs. You use this VI to retrieve information from that list.

The VI returns an empty string if no coercion records remain for the session.

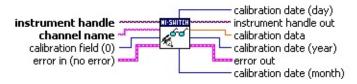


- instrument handle identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u>.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurs before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - status is TRUE (X) if an error occurred before this VI ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI ran. The default is FALSE.
 - code is the error or warning code. The default is 0. If status is TRUE, code is a non-zero error code. If status is FALSE, code is 0 or a warning code.
 - source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- instrument handle out identifies a particular NI-SWITCH session established with niSwitch Initialize With Topology, niSwitch Initialize With Options, or niSwitch Initialize and used for all subsequent NI SWITCH calls

niSwitch Read Calibration Data [OBSOLETE]

Retrieves the calibration data, typically in terms of the amplifier offset, stored in the EEPROM.

The NI PXI-2501 has an amplifier that may require periodic calibrations. You can perform the necessary calibration and store the data locally on the switch module EEPROM. The calibration date is also stored in the EEPROM.



1/0

instrument handle identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u>.

abc

channel name specifies the name of the channel calibrated.

Examples of valid channel names:

ch0, com0, ab0, r1, c2, citemp

Refer to <u>Devices</u> for a complete list of valid channel names.

While this VI and <u>niSwitch Write Calibration Data</u> take a channel name, some switch devices only support a single calibration for all input channels. For example, the NI PXI-2501 uses an optional single amplifier for channels ch0 through ch47 to decrease settling time. In these cases, writing a different value to a different channel causes the previous value to be overwritten. Therefore, reading different channels returns the same calibration data.

For more information on calibration, refer to **Devices**.

I32

calibration field tells NI-SWITCH which particular calibration parameter associated with this channel to read. Valid values depend on the switch hardware.

Examples of valid fields:

CJTEMP Amp Calibration (0), Channel Amp Calibration (1)

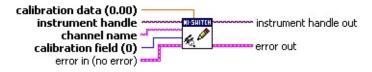
Default value: 0

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the **error in** value to **error out**. This VI or function runs normally only if no

niSwitch Write Calibration Data [OBSOLETE]

Writes the calibration data, typically in terms of the amplifier offset, in the EEPROM.

The NI PXI-2501 has an amplifier that may require periodic calibrations. You can perform the necessary calibration and store the data locally on the switch module EEPROM. The calibration date is also stored in the EEPROM.



instrument handle identifies a particular NI-SWITCH session established with <u>niSwitch Initialize With Topology</u>, <u>niSwitch Initialize With Options</u>, or <u>niSwitch Initialize</u>.

channel name specifies the name of the channel calibrated.

Examples of valid channel names:

ch0, com0, ab0, r1, c2, cjtemp

abc

Refer to <u>Devices</u> for a complete list of valid channel names.

While niSwitch Read Calibration Data and this VI take a channel name, some switch modules only support a single calibration for all input channels. For example, the NI PXI-2501 uses an optional single amplifier for channels ch0 through ch47 to decrease settling time. In these cases, writing a different value to a different channel causes the previous value to be overwritten. Therefore, reading different channels returns the same calibration data.

For more information on calibration, refer to **Devices**.

calibration field tells NI-SWITCH which particular calibration parameter associated with this channel to write. Valid values depend on the switch hardware.

Examples of valid fields:

CJTEMP Amp Calibration (0), Channel Amp Calibration (1)

Default value: 0

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no

NI-SWITCH Properties

Use the NI-SWITCH properties to access advanced configuration options and information.

- Channel Configuration
- Inherent IVI Attributes
- Matrix Configuration
- Module Characteristics
- Scanning Configuration

Is Configuration Channel

Short name: Is Configuration Channel

This property specifies whether to reserve the channel for internal path creation. A channel that is available for internal path creation is called a configuration channel. NI-SWITCH may use configuration channels to create paths between two channels you specify in the niSwitch Connect Channels VI. Configuration channels are not available for external connections. Set this property to TRUE to mark the channel as a configuration channel. Set this property to FALSE to mark the channel as available for external connections. After you identify a channel as a configuration channel, you cannot use that channel for external connections. The niSwitch Connect Channels VI returns the Is Configuration Channel error when you attempt to establish a connection between a configuration channel and any other channel.

VI_	TRUE	True
VI_	FALSE	False

Data Type	ViBoolean
Channel Based	Yes

Is Source Channel

Short name: Is Source Channel

This property specifies whether you want to identify the channel as a source channel. Typically, you set this property to TRUE when you attach the channel to a power supply, a VI generator, or an active measurement point on the unit under test, and you do not want to connect the channel to another source. NI-SWITCH prevents source channels from connecting to each other. The niSwitch Connect Channels VI returns the Attempt To Connect Sources error when you attempt to connect two channels that you identify as source channels.

VI_	TRUE	True
VI_	FALSE	False

Data Type	ViBoolean
Channel Based	Yes

Driver Setup

Short name: Driver Setup

This property indicates the DriverSetup string that the user specified when initializing the driver. Some cases exist where the end-user must specify instrument driver options at initialization time. An example of this is specifying a particular instrument model from among a family of instruments that the driver supports. This is useful when using simulation. The end-user can specify driver-specific options through the DriverSetup keyword in the **option string** parameter to the <u>niSwitch Initialize With Options</u> VI, or through the IVI Configuration Utility. If the user does not specify a DriverSetup string, this property returns an empty string.

Data Type	ViString
Channel Based	No

IO Resource Descriptor

Short name: IO Resource Descriptor

This property indicates the resource descriptor NI-SWITCH uses to identify the physical device. If you initialize NI-SWITCH with a logical name, this property contains the resource descriptor that corresponds to the entry in the IVI Configuration utility. If you initialize NI-SWITCH with the resource descriptor, this property contains that value.

Data Type	ViString
Channel Based	No

Logical Name

Short name: Logical Name

This property contains a string with the logical name specified when opening the current IVI session. You may pass a logical name to the niSwitch initialize or niSwitch Initialize With Options VIs. The IVI Configuration utility must contain an entry for the logical name. The logical name entry refers to a virtual instrument section in the IVI Configuration file. The virtual instrument section specifies a physical device and initial user options.

Data Type	ViString
Channel Based	No

Channel Count

Short name: Channel Count

This property indicates the number of channels that the specific

instrument driver supports.

Data Type	Vilnt32
Channel Based	No

Class Group Capabilities

Short name: Class Group Capabilities

This property contains a comma-delimited list of class-extension groups that this driver implements.

Data Type	ViString
Channel Based	No

Supported Instrument Models

Short name: Supported Instrument Models

This property contains a comma-delimited list of supported instrument models.

Data Type	ViString
Channel Based	No

Class Specification Major Version

Short name: Class Specification Major Version

This property contains the major version number of the IviSwtch class specification.

Data Type	Vilnt32
Channel Based	No

Class Specification Minor Version

Short name: Class Specification Minor Version

This property contains the minor version number of the driver compliant class specification.

Data Type	Vilnt32
Channel Based	No

Specific Driver Description

Short name: Specific Driver Description

This property contains a brief description of the instrument driver.

Data Type	ViString
Channel Based	No

Driver Prefix

Short name: Driver Prefix

This property contains the prefix for the instrument driver. The name of each available VI in this driver starts with the prefix specified in this property.

Data Type	ViString
Channel Based	No

Driver Vendor

Short name: Driver Vendor

This property contains the name of the vendor that supplies the

instrument driver.

Data Type	ViString
Channel Based	No

Revision

Short name: Revision

This property contains additional version information about the instrument driver.

Data Type	ViString
Channel Based	No

Firmware Revision

Short name: Firmware Revision

This property contains the firmware revision information for the instrument you are currently using.

Data Type	ViString
Channel Based	No

Manufacturer

Short name: Manufacturer

This property contains the name of the instrument manufacturer you are

currently using.

Data Type	ViString
Channel Based	No

Model

Short name: Model

This property contains the model number or name of the instrument that you are currently using.

Data Type	ViString
Channel Based	No

Cache

Short name: Cache

This property specifies whether to cache property values. When caching is enabled, the instrument driver keeps track of the current instrument settings and avoids sending redundant commands to the instrument. The instrument driver can choose always to cache or never to cache particular properties regardless of this property setting. The default value is TRUE. Use the <u>niSwitch Initialize With Options</u> VI to override this value.

VI	TRUE	True
VI_	FALSE	False

Data Type	ViBoolean
Channel Based	No

Interchange Check

Short name: Interchange Check

This property specifies whether to perform interchangeability checking and retrieve interchangeability warnings when you call niSwitch Connect Channels, niSwitch Set Path and niSwitch InitiateScan VIs. The default value is FALSE. Interchangeability warnings indicate that using your application with a different instrument might cause different behavior. Use niSwitch Get Next Interchange Warning to extract interchange warnings. Call the niSwitch Clear Interchange Warnings VI to clear the list of interchangeability warnings without reading them. Interchangeability checking examines the properties in a capability group only if you specify a value for at least one property within that group. Interchangeability warnings can occur when a property affects the behavior of the instrument and you have not set that property, or the property has been invalidated since you set it.

VI_	TRUE	True
VI_	FALSE	False

Data Type	ViBoolean
Channel Based	No

Query Instrument Status

Short name: Query Instrument Status

This property specifies whether the instrument driver queries the instrument status after each operation. Querying the instrument status is very useful for debugging. After you validate your program, you can set this property to FALSE to disable status checking and maximize performance. The instrument driver can choose to ignore status checking for particular properties regardless of the setting of this property. The default value is TRUE. Use the <u>niSwitch Initialize With Options</u> VI to override this value.

VI_	TRUE	True
VI_	FALSE	False

Data Type	ViBoolean
Channel Based	No

Range Check

Short name: Range Check

Specifies whether to validate property values and VI parameters. If enabled, the instrument driver validates the parameter values that you pass to driver VIs. Range checking parameters is useful for debugging. After you validate your program, set this property to FALSE to disable range checking and maximize performance. The default value is TRUE. Use the <u>niSwitch Initialize With Options</u> VI to override this value.

VI_	TRUE	True
VI_	FALSE	False

Data Type	ViBoolean
Channel Based	No

Record Value Coercions

Short name: Record Value Coercions

This property Specifies whether the IVI engine keeps a list of the value coercions it makes for ViInt32 and ViReal64 properties. Call niSwitch Get Next Coercion Record to extract and delete the oldest coercion record from the list. The default value is FALSE. Use the niSwitch Initialize With Options VI to override this value.

VI_	TRUE	True
VI_	FALSE	False

Data Type	ViBoolean
Channel Based	No

Simulate

Short name: Simulate

This property specifies whether or not to simulate instrument driver I/O operations. If simulation is enabled, instrument driver VIs perform range checking and call Ivi_GetAttribute and Ivi_SetAttribute VIs, but they do not perform instrument I/O. For output parameters that represent instrument data, the instrument driver VIs return calculated values. The default value is FALSE. Use the niswitch Initialize With Options VI to override this value.

VI_	TRUE	True
VI_	FALSE	False

Data Type	ViBoolean
Channel Based	No

Number of Columns

Short name: Number of Columns

This property returns the number of channels on the column of a matrix or scanner. If the switch is a scanner, this value is the number of input channels.

The <u>Wire Mode</u> property affects the number of available columns. For example, if your device has 8 input lines and you use the 4-wire mode, then the number of columns you have available is 2.

Data Type	Vilnt32
Channel Based	No

Number of Rows

Short name: Number of Rows

This property returns the number of channels on the row of a matrix or scanner. If the switch is a scanner, this value is the number of output channels.

The <u>Wire Mode</u> property affects the number of available rows. For example, if your device has 8 input lines and you use the 2-wire mode, then the number of columns you have available is 4.

Data Type	Vilnt32
Channel Based	No

Bandwidth

Short name: Bandwidth

This property returns the channel bandwidth. The units are Hz.

Data Type	ViReal64
Channel Based	Yes

Characteristic Impedance

Short name: Characteristic Impedance

This property returns the characteristic impedance for the channel. The units are $\[\Omega \]$ s.

Data Type	ViReal64
Channel Based	Yes

Is Debounced

Short name: Is Debounced

This property indicates whether the entire switch has settled since the last switching command. A value of TRUE indicates that all signals going through the switch are valid.

VI_	TRUE	True
VI_	FALSE	False

Data Type	ViBoolean
Channel Based	No

Maximum AC Voltage

Short name: Maximum AC Voltage

This property returns the maximum AC voltage the channel can switch.

The units are volts RMS.

Data Type	ViReal64
Channel Based	Yes

Maximum Switching AC Current

Short name: Maximum Switching AC Current

This property returns the maximum AC current the channel can switch.

The units are amperes RMS.

Data Type	ViReal64
Channel Based	Yes

Maximum Switching AC Power

Short name: Maximum Switching AC Power

This property returns the maximum AC power the channel can switch.

The units are volt-amperes.

Data Type	ViReal64
Channel Based	Yes

Maximum DC Voltage

Short name: Maximum DC Voltage

This property returns the maximum DC voltage the channel can switch.

The units are volts.

Data Type	ViReal64
Channel Based	Yes

Maximum Switching DC Current

Short name: Maximum Switching DC Current

This property returns the maximum DC current the channel can switch.

The units are amperes.

Data Type	ViReal64
Channel Based	Yes

Maximum Switching DC Power

Short name: Maximum Switching DC Power

This property returns the maximum DC power the channel can switch.

The units are watts.

Data Type	ViReal64
Channel Based	Yes

Maximum Carry AC Current

Short name: Maximum Carry AC Current

This property returns the maximum AC current the channel can carry.

The units are amperes RMS.

Data Type	ViReal64
Channel Based	Yes

Maximum Carry AC Power

Short name: Maximum Carry AC Power

This property returns the maximum AC power the channel can carry. The units are volt-amperes.

Data Type	ViReal64
Channel Based	Yes

Maximum Carry DC Current

Short name: Maximum Carry DC Current

This property returns the maximum DC current the channel can carry.

The units are amperes.

Data Type	ViReal64
Channel Based	Yes

Maximum Carry DC Power

Short name: Maximum Carry DC Power

This property returns the maximum DC power the channel can carry. The units are watts.

Data Type	ViReal64
Channel Based	Yes

Number of Relays

Short name: Number of Relays

This property returns the number of relays.

Data Type	Vilnt32
Channel Based	No

Power Down Latching Relays After Debounce

Short name: Pwr Down Latch Relays After Settling

This property indicates whether to power down latching relays after calling Wait For Debounce. When Power Down Latching Relays After Debounce is enabled (VI_TRUE), a call to Wait For Debounce ensures that the relays are settled and the latching relays are powered down.

VI_	TRUE	True
VI_	FALSE	False

The following table lists the characteristics of this property.

Data Type	ViBoolean
Channel Based	No



Tip You can also use the DAQmx Switch Connect VI to power down latching relays.

Serial Number

Short name: Serial Number

This read-only property returns the serial number for the switch controlled by NI-SWITCH. If the switch does not return a serial number, NI-SWITCH returns the Invalid Attribute error.

Data Type	Vilnt32
Channel Based	No

Settling Time

Short name: Settling Time

This property returns the maximum length of time from after you make a connection until the signal flowing through the channel settles. The units are seconds.

Data Type	ViReal64
Channel Based	Yes

Wire Mode

Short name: Wire Mode

This property returns the wire mode of the switch. This property affects the values of the <u>Number of Rows</u> and <u>Number of Columns</u> properties. The actual number of input and output channels on the switch is fixed, but the number of channels depends on how many lines constitute each channel.

Data Type	ViInt32
Channel Based	Yes

Cabled Module Scan Advanced Bus

Short name: Cabled Module Scan Advanced Bus

Data Type	Vilnt32
Channel Based	No

Cabled Module Trigger Bus

Short name: Cabled Module Trigger Bus

Data Type	Vilnt32
Channel Based	No

Continuous Scan

Short name: Continuous Scan

When a switch is scanning, the switch can either stop scanning when the end of the scan (FALSE) or continue scanning from the top of the scan list again (TRUE).

Notice that if you set the scan to continuous (TRUE), <u>niSwitch Wait For Scan To Complete</u> will always time out, and you must call <u>niSwitch Abort Scan</u> to stop the scan.

VI_	TRUE	True
VI_	FALSE	False

Data Type	ViBoolean
Channel Based	No

Handshaking Initiation

Short name: Handshaking Initiation

This property allows you to start <u>handshaking</u> with a measurement device.

Measurement_Device_Initiated	niSwitch Initiate Scan does not return until the switch hardware is waiting for a trigger input. This ensures that if you initiate the measurement device after calling niSwitch Initiate Scan, the switch is sure to receive the first measurement complete (MC) signal sent by the measurement device. The measurement device should be configured to first take a measurement, send MC, then wait for scanner advanced output signal. Thus, the first MC of the measurement device initiates handshaking.
Switch_Initiated	niSwitch Initiate Scan returns immediately after beginning scan list execution. It is assumed that the measurement device has already been configured and is waiting for the scanner advanced signal. The measurement should be configured to first wait for a trigger, then take a measurement. Thus, the first scanner advanced output signal of the switch module initiates handshaking.

Data Type	Vilnt32
Channel Based	No

Is Scanning

Short name: Is Scanning

This property indicates whether the switch has completed the scan operation. The value TRUE indicates that the scan is complete.

VI_	TRUE	True
VI_	_FALSE	False

Data Type	ViBoolean
Channel Based	No

Is Waiting for Trigger?

Short name: Is Waiting for Trigger?

In a scan list, a semicolon is used to indicate that at that point in the scan list, the scan engine should pause until a trigger is received from the trigger input. If that trigger is generated through either a hardware pulse or <u>niSwitch Send Software Trigger</u>, you need to know when the scan engine has reached such a state.

VI_	TRUE	True
VI_	_FALSE	False

Data Type	ViBoolean
Channel Based	No

Parsed Scan List

Short name: Parsed Scan List

Data Type	ViString
Channel Based	No

Scan Advanced Bus

Short name: Scan Advanced Bus

When multiple switches need to scan using a common trigger output (Scan Advanced Output), you must route the output trigger from the slave switches to the master.

This property should be set to the same for all switches involved in the scan.

NISWITCH_VAL_NONE	The switch does not produce a Scan Advanced Output trigger.
NISWITCH_VAL_EXTERNAL	External Trigger. The switch produces the Scan Advanced Output trigger on the external trigger output.
NISWITCH_VAL_TTL0	The switch produces the Scan Advanced Output on the PXI TRIGO line.
NISWITCH_VAL_TTL1	The switch produces the Scan Advanced Output on the PXI TRIG1 line.
NISWITCH_VAL_TTL2	The switch produces the Scan Advanced Output on the PXI TRIG2 line.
NISWITCH_VAL_TTL3	The switch produces the

	Scan Advanced Output on the PXI TRIG3 line.
NISWITCH_VAL_TTL4	The switch produces the Scan Advanced Output on the PXI TRIG4 line.
NISWITCH_VAL_TTL5	The switch produces the Scan Advanced Output on the PXI TRIG5 line.
NISWITCH_VAL_TTL6	The switch produces the Scan Advanced Output on the PXI TRIG6 line.
NISWITCH_VAL_TTL7	The switch produces the Scan Advanced Output on the PXI TRIG7 line.
NISWITCH_VAL_PXI_STAR	The switch produces the Scan Advanced Output on the PXI STAR trigger bus.
NISWITCH_VAL_FRONTCONNECTOR	The switch produces the Scan Advanced Output trigger on the front connector.
NISWITCH_VAL_FRONTCONNECTOR_MODULE1	The switch produces the Scan Advanced

	Output trigger on the front connector of module 1.
NISWITCH_VAL_FRONTCONNECTOR_MODULE2	The switch produces the Scan Advanced Output trigger on the front connector of module 2.
NISWITCH_VAL_FRONTCONNECTOR_MODULE3	The switch produces the Scan Advanced Output trigger on the front connector of module 3.
NISWITCH_VAL_FRONTCONNECTOR_MODULE4	The switch produces the Scan Advanced Output trigger on the front connector of module 4.
NISWITCH_VAL_FRONTCONNECTOR_MODULE5	The switch produces the Scan Advanced Output trigger on the front connector of module 5.
NISWITCH_VAL_FRONTCONNECTOR_MODULE6	The switch produces the Scan Advanced Output trigger on the front

	connector of module 6.
NISWITCH_VAL_FRONTCONNECTOR_MODULE7	The switch produces the Scan Advanced Output trigger on the front connector of module 7.
NISWITCH_VAL_FRONTCONNECTOR_MODULE8	The switch produces the Scan Advanced Output trigger on the front connector of module 8.
NISWITCH_VAL_FRONTCONNECTOR_MODULE9	The switch produces the Scan Advanced Output trigger on the front connector of module 9.
NISWITCH_VAL_FRONTCONNECTOR_MODULE10	The switch produces the Scan Advanced Output trigger on the front connector of module 10.
NISWITCH_VAL_FRONTCONNECTOR_MODULE11	The switch produces the Scan Advanced Output trigger on the front connector of module 11.

NISWITCH_VAL_FRONTCONNECTOR_MODULE12	The switch
	produces the
	Scan Advanced
	Output trigger on
	the front
	connector of
	module 12.

Data Type	Vilnt32
Channel Based	No

Scan Advanced Output

Short name: Scan Advanced Output

This property specifies the method you want to use to notify another instrument that all signals going through the switch have settled following the processing of one entry in the scan list.

NISWITCH_VAL_NONE	The switch does not produce a Scan Advanced Output trigger.
NISWITCH_VAL_EXTERNAL	External Trigger. The switch produces the Scan Advanced Output trigger on the external trigger output.
NISWITCH_VAL_TTL0	The switch produces the Scan Advanced Output trigger on the PXI TRIGO line.
NISWITCH_VAL_TTL1	The switch produces the Scan Advanced Output trigger on the PXI TRIG1 line.
NISWITCH_VAL_TTL2	The switch produces the Scan Advanced Output trigger on the PXI TRIG2 line.
NISWITCH_VAL_TTL3	The switch

	produces the Scan Advanced Output trigger on the PXI TRIG3 line.
NISWITCH_VAL_TTL4	The switch produces the Scan Advanced Output trigger on the PXI TRIG4 line.
NISWITCH_VAL_TTL5	The switch produces the Scan Advanced Output trigger on the PXI TRIG5 line.
NISWITCH_VAL_TTL6	The switch produces the Scan Advanced Output trigger on the PXI TRIG6 line.
NISWITCH_VAL_TTL7	The switch produces the Scan Advanced Output trigger on the PXI TRIG7 line.
NISWITCH_VAL_PXI_STAR	The switch produces the Scan Advanced Output trigger on trigger bus.
NISWITCH_VAL_FRONTCONNECTOR	The switch produces the

	Scan Advanced Output trigger on the front connector.
NISWITCH_VAL_FRONTCONNECTOR_MODULE1	The switch produces the Scan Advanced Output trigger on the front connector module 1.
NISWITCH_VAL_FRONTCONNECTOR_MODULE2	The switch produces the Scan Advanced Output trigger on the front connector module 2.
NISWITCH_VAL_FRONTCONNECTOR_MODULE3	The switch produces the Scan Advanced Output trigger on the front connector module 3.
NISWITCH_VAL_FRONTCONNECTOR_MODULE4	The switch produces the Scan Advanced Output trigger on the front connector module 4.
NISWITCH_VAL_FRONTCONNECTOR_MODULE5	The switch produces the Scan Advanced Output trigger on the front

	connector module 5.
NISWITCH_VAL_FRONTCONNECTOR_MODULE6	The switch produces the Scan Advanced Output trigger on the front connector module 6.
NISWITCH_VAL_FRONTCONNECTOR_MODULE7	The switch produces the Scan Advanced Output trigger on the front connector module 7.
NISWITCH_VAL_FRONTCONNECTOR_MODULE8	The switch produces the Scan Advanced Output trigger on the front connector module 8.
NISWITCH_VAL_FRONTCONNECTOR_MODULE9	The switch produces the Scan Advanced Output trigger on the front connector module 9.
NISWITCH_VAL_FRONTCONNECTOR_MODULE10	The switch produces the Scan Advanced Output trigger on the front connector module 10.

NISWITCH_VAL_FRONTCONNECTOR_MODULE11	The switch produces the Scan Advanced Output trigger on the front connector module 11.
NISWITCH_VAL_FRONTCONNECTOR_MODULE12	The switch produces the Scan Advanced Output trigger on the front connector module 12.

Data Type	Vilnt32
Channel Based	No

Scan Advanced Polarity

Short name: Scan Advanced Polarity

This attribute sets the driving level for the Scan Advanced Output signal sent from the switch module through either the external (PXI) or front connector (SCXI) lines.

When the Scan Advanced Output signal is sent to one of the TTL lines, the driven level is always low and this property is ignored. Between each Scan Advanced Output signal, the line is not driven and is in a high-impedance state.

<u> </u>	The line is driven up when the Scan Advanced Output signal is sent.
	The line is driven down when the Scan Advanced Output signal is sent.

Data Type	Vilnt32
Channel Based	No

Scan Delay

Short name: Scan Delay

This property specifies the minimum amount of time the switch waits before it asserts the Scan Advanced Output trigger after opening or closing the switch. The switch always waits for debounce before asserting the trigger. Thus, the actual delay is always the greater value of the settling time and the value you specify as the switch delay. The units are seconds. Different switch designs may cause the actual time to be longer.

Data Type	ViReal64
Channel Based	No

Scan List

Short name: Scan List

This property contains a <u>scan list</u>. The <u>niSwitch Initiate Scan</u> VI makes or breaks connections and waits for triggers according to the instructions in the scan list. The scan list is comprised of channel names that you separate with special characters. These special characters determine the operations the scanner performs on the channels when it executes this scan list.

- To create a path between two channels, use the following character between the two channel names: -> (a dash followed by a '>' sign) Example: \CH1->CH2\ tells the switch to make a path from channel CH1 to channel CH2.
- To break or clear a path, use the following character as a prefix before the path: ~ (tilde) Example: \~CH1->CH2\ tells the switch to break the path from channel CH1 to channel CH2.
- To tell the switch to wait for a trigger event, use the following character as a separator between paths: ; (semicolon) Example: \CH1->CH2;CH3->CH4\ tells the switch to make the path from channel CH1 to channel CH2, wait for a trigger, and then make the path from CH3 to CH4.
- To tell the switch to create multiple paths simultaneously, use the following character as a separator between the paths: , (comma) Example: \A->B;CH1->CH2,CH3->CH4\ instructs the scanner to make the path between channels A and B, wait for a trigger, and then simultaneously make the paths between channels CH1 and CH2 and between channels CH3 and CH4.

Data Type	ViString
Channel Based	No

Scan Mode

Short name: Scan Mode

This property specifies what happens to existing connections that conflict with the connections you make in a scan list. For example, if CH1 is already connected to CH2 and the scan list instructs the switch to connect CH1 to CH3, this property specifies what happens to the connection between CH1 and CH2. If the value of this property is NISWITCH_VAL_NONE, the switch takes no action on existing paths. Most switches support only one of the possible values. In such cases, this property serves as an indicator of the device behavior.

NISWITCH_VAL_NONE	No implicit action on connections when scanning.
NISWITCH_VAL_BREAK_BEFORE_MAKE	When scanning, the switch breaks existing connections before making new connections.
NISWITCH_VAL_BREAK_AFTER_MAKE	When scanning, the switch breaks existing connections after making new connections.

Data Type	Vilnt32
Channel Based	No

Trigger Bus

Short name: Trigger Bus

When multiple switches need to scan using a common trigger input, it is necessary to route the input trigger from the master switch to the slaves.

This property should be set to the same setting for all switches involved in the scan.

NISWITCH_VAL_IMMEDIATE	Immediate Trigger. The switch does not wait for a trigger before processing the next entry in the scan list.
NISWITCH_VAL_SW_TRIG_FUNC	The switch waits until you call the niSwitch Send Software Trigger VI before processing the next entry in the scan list.
NISWITCH_VAL_EXTERNAL	External Trigger. The switch waits until it receives a trigger from an external source through the external trigger input before processing the next entry in the scan list.
NISWITCH_VAL_TTL0	The switch waits until it receives a trigger on the PXI TRIGO line before

	processing the next entry in the scan list.
NISWITCH_VAL_TTL1	The switch waits until it receives a trigger on the PXI TRIG1 line before processing the next entry in the scan list.
NISWITCH_VAL_TTL2	The switch waits until it receives a trigger on the PXI TRIG2 line before processing the next entry in the scan list.
NISWITCH_VAL_TTL3	The switch waits until it receives a trigger on the PXI TRIG3 line before processing the next entry in the scan list.
NISWITCH_VAL_TTL4	The switch waits until it receives a trigger on the PXI TRIG4 line before processing the next entry in the scan list.
NISWITCH_VAL_TTL5	The switch waits until it receives a trigger on the PXI TRIG5 line before processing the next entry in the

	scan list.
NISWITCH_VAL_TTL6	The switch waits until it receives a trigger on the PXI TRIG6 line before processing the next entry in the scan list.
NISWITCH_VAL_TTL7	The switch waits until it receives a trigger on the PXI TRIG7 line before processing the next entry in the scan list.
NISWITCH_VAL_PXI_STAR	The switch waits until it receives a trigger on the PXI STAR trigger bus before processing the next entry in the scan list.
NISWITCH_VAL_REARCONNECTOR	The switch waits until it receives a trigger on the rear connector.
NISWITCH_VAL_FRONTCONNECTOR	The switch waits until it receives a trigger on the front connector.
NISWITCH_VAL_REARCONNECTOR_MODULE1	The switch waits until it receives a trigger on the rear connector module 1.
NISWITCH_VAL_REARCONNECTOR_MODULE2	The switch waits

	until it receives a trigger on the rear connector module 2.
NISWITCH_VAL_REARCONNECTOR_MODULE3	The switch waits until it receives a trigger on the rear connector module 3.
NISWITCH_VAL_REARCONNECTOR_MODULE4	The switch waits until it receives a trigger on the rear connector module 4.
NISWITCH_VAL_REARCONNECTOR_MODULE5	The switch waits until it receives a trigger on the rear connector module 5.
NISWITCH_VAL_REARCONNECTOR_MODULE6	The switch waits until it receives a trigger on the rear connector module 6.
NISWITCH_VAL_REARCONNECTOR_MODULE7	The switch waits until it receives a trigger on the rear connector module 7.
NISWITCH_VAL_REARCONNECTOR_MODULE8	The switch waits until it receives a trigger on the rear connector module 8.
NISWITCH_VAL_REARCONNECTOR_MODULE9	The switch waits until it receives a

	trigger on the rear connector module 9.
NISWITCH_VAL_REARCONNECTOR_MODULE10	The switch waits until it receives a trigger on the rear connector module 10.
NISWITCH_VAL_REARCONNECTOR_MODULE11	The switch waits until it receives a trigger on the rear connector module 11.
NISWITCH_VAL_REARCONNECTOR_MODULE12	The switch waits until it receives a trigger on the rear connector module 12.
NISWITCH_VAL_FRONTCONNECTOR_MODULE1	The switch waits until it receives a trigger on the front connector module 1.
NISWITCH_VAL_FRONTCONNECTOR_MODULE2	The switch waits until it receives a trigger on the front connector module 2.
NISWITCH_VAL_FRONTCONNECTOR_MODULE3	The switch waits until it receives a trigger on the front connector module 3.
NISWITCH_VAL_FRONTCONNECTOR_MODULE4	The switch waits until it receives a trigger on the

	front connector module 4.
NISWITCH_VAL_FRONTCONNECTOR_MODULE5	The switch waits until it receives a trigger on the front connector module 5.
NISWITCH_VAL_FRONTCONNECTOR_MODULE6	The switch waits until it receives a trigger on the front connector module 6.
NISWITCH_VAL_FRONTCONNECTOR_MODULE7	The switch waits until it receives a trigger on the front connector module 7.
NISWITCH_VAL_FRONTCONNECTOR_MODULE8	The switch waits until it receives a trigger on the front connector module 8.
NISWITCH_VAL_FRONTCONNECTOR_MODULE9	The switch waits until it receives a trigger on the front connector module 9.
NISWITCH_VAL_FRONTCONNECTOR_MODULE10	The switch waits until it receives a trigger on the front connector module 10.
NISWITCH_VAL_FRONTCONNECTOR_MODULE11	The switch waits until it receives a trigger on the front connector

module 11.
The switch waits until it receives a trigger on the front connector module 12.

Data Type	Vilnt32
Channel Based	No

Trigger Input

Short name: Trigger Input

This property specifies the trigger source for which the switch can wait when processing a scan list. The switch waits for a trigger when it encounters a semicolon in a scan list. When the trigger occurs, the switch advances to the next entry in the scan list.

NISWITCH_VAL_IMMEDIATE	Immediate Trigger. The switch does not wait for a trigger before processing the next entry in the scan list.
NISWITCH_VAL_SW_TRIG_FUNC	The switch waits until you call the niSwitch Send Software Trigger VI before processing the next entry in the scan list.
NISWITCH_VAL_EXTERNAL	External Trigger. The switch waits until it receives a trigger from an external source through the external trigger input before processing the next entry in the scan list.
NISWITCH_VAL_TTL0	The switch waits until it receives a trigger on the PXI TRIGO line before

	processing the next entry in the scan list.
NISWITCH_VAL_TTL1	The switch waits until it receives a trigger on the PXI TRIG1 line before processing the next entry in the scan list.
NISWITCH_VAL_TTL2	The switch waits until it receives a trigger on the PXI TRIG2 line before processing the next entry in the scan list.
NISWITCH_VAL_TTL3	The switch waits until it receives a trigger on the PXI TRIG3 line before processing the next entry in the scan list.
NISWITCH_VAL_TTL4	The switch waits until it receives a trigger on the PXI TRIG4 line before processing the next entry in the scan list.
NISWITCH_VAL_TTL5	The switch waits until it receives a trigger on the PXI TRIG5 line before processing the next entry in the

	scan list.
NISWITCH_VAL_TTL6	The switch waits until it receives a trigger on the PXI TRIG6 line before processing the next entry in the scan list.
NISWITCH_VAL_TTL7	The switch waits until it receives a trigger on the PXI TRIG7 line before processing the next entry in the scan list.
NISWITCH_VAL_PXI_STAR	The switch waits until it receives a trigger on the PXI STAR trigger bus before processing the next entry in the scan list.
NISWITCH_VAL_REARCONNECTOR	The switch waits until it receives a trigger on the rear connector.
NISWITCH_VAL_FRONTCONNECTOR	The switch waits until it receives a trigger on the front connector.
NISWITCH_VAL_REARCONNECTOR_MODULE1	The switch waits until it receives a trigger on the rear connector module 1.
NISWITCH_VAL_REARCONNECTOR_MODULE2	The switch waits

	until it receives a trigger on the rear connector module 2.
NISWITCH_VAL_REARCONNECTOR_MODULE3	The switch waits until it receives a trigger on the rear connector module 3.
NISWITCH_VAL_REARCONNECTOR_MODULE4	The switch waits until it receives a trigger on the rear connector module 4.
NISWITCH_VAL_REARCONNECTOR_MODULE5	The switch waits until it receives a trigger on the rear connector module 5.
NISWITCH_VAL_REARCONNECTOR_MODULE6	The switch waits until it receives a trigger on the rear connector module 6.
NISWITCH_VAL_REARCONNECTOR_MODULE7	The switch waits until it receives a trigger on the rear connector module 7.
NISWITCH_VAL_REARCONNECTOR_MODULE8	The switch waits until it receives a trigger on the rear connector module 8.
NISWITCH_VAL_REARCONNECTOR_MODULE9	The switch waits until it receives a

	trigger on the rear connector module 9.
NISWITCH_VAL_REARCONNECTOR_MODULE10	The switch waits until it receives a trigger on the rear connector module 10.
NISWITCH_VAL_REARCONNECTOR_MODULE11	The switch waits until it receives a trigger on the rear connector module 11.
NISWITCH_VAL_REARCONNECTOR_MODULE12	The switch waits until it receives a trigger on the rear connector module 12.
NISWITCH_VAL_FRONTCONNECTOR_MODULE1	The switch waits until it receives a trigger on the front connector module 1.
NISWITCH_VAL_FRONTCONNECTOR_MODULE2	The switch waits until it receives a trigger on the front connector module 2.
NISWITCH_VAL_FRONTCONNECTOR_MODULE3	The switch waits until it receives a trigger on the front connector module 3.
NISWITCH_VAL_FRONTCONNECTOR_MODULE4	The switch waits until it receives a trigger on the

	front connector module 4.
NISWITCH_VAL_FRONTCONNECTOR_MODULE5	The switch waits until it receives a trigger on the front connector module 5.
NISWITCH_VAL_FRONTCONNECTOR_MODULE6	The switch waits until it receives a trigger on the front connector module 6.
NISWITCH_VAL_FRONTCONNECTOR_MODULE7	The switch waits until it receives a trigger on the front connector module 7.
NISWITCH_VAL_FRONTCONNECTOR_MODULE8	The switch waits until it receives a trigger on the front connector module 8.
NISWITCH_VAL_FRONTCONNECTOR_MODULE9	The switch waits until it receives a trigger on the front connector module 9.
NISWITCH_VAL_FRONTCONNECTOR_MODULE10	The switch waits until it receives a trigger on the front connector module 10.
NISWITCH_VAL_FRONTCONNECTOR_MODULE11	The switch waits until it receives a trigger on the front connector

module 11.
The switch waits until it receives a trigger on the front connector module 12.

Data Type	Vilnt32
Channel Based	No

Trigger Input Polarity

Short name: Trigger Input Polarity

	The trigger occurs on the rising edge of the signal.
NISWITCH_VAL_FALLING_EDGE	The trigger occurs on the falling edge of the signal.

Data Type	Vilnt32
Channel Based	No

Trigger Mode

Short name: Trigger Mode

If there is only a single switch involved in a scan, then set this property to SINGLE.

If there is more than one switch involved in the scan, then the triggers should be routed between the devices. One switch acts as the master, and all others will act as slaves.

NISWITCH_VAL_SINGLE	When scanning, the switch does not share tri other switches. You must set NISWITCH_ATTR_SCAN_ADVANCED_OUT NISWITCH_ATTR_TRIGGER_INPUT for this
NISWITCH_VAL_MASTER	Multiple switches are sharing bused trigger lir this device is the trigger master. Set NISWITCH_ATTR_MASTER_SLAVE_TRIGG NISWITCH_ATTR_MASTER_SLAVE_SCAN_ NISWITCH_ATTR_SCAN_ADVANCED_OUT NISWITCH_ATTR_TRIGGER_INPUT for this
NISWITCH_VAL_SLAVE	Multiple switches are sharing trigger lines for device is one of the trigger slaves. Set NISWITCH_ATTR_MASTER_SLAVE_TRIGGNISWITCH_ATTR_MASTER_SLAVE_SCAN for this device.

Data Type	Vilnt32
Channel Based	No