

NI-SWITCH Instrument Driver Quick Reference Guide

For National Instruments Switches

ICON	FUNCTION NAME †	TYPE	PARAMETER
------	-----------------	------	-----------

Initialize with Topology and Close



niSwitch Initialize With Topology (niSwitch_InitWithTopology)

Returns a session handle used to identify the switch module in all subsequent instrument driver calls. Also sets the topology of the switch. Refer to *Resource Name Syntax* at the end of this guide.

ViConstString	resource name
ViConstString	topology name
ViBoolean	simulate
ViBoolean	reset
ViSession (out)	vi



niSwitch Close (niSwitch_close)

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

ViSession	vi
-----------	----

Immediate Operations



niSwitch Connect Channels (niSwitch_Connect)

Creates the shortest available path between **channel 1** and **channel 2**. If a path is unavailable, an error is returned.

ViSession	vi
ViConstString	channel 1
ViConstString	channel 2



niSwitch Disconnect Channels (niSwitch_Disconnect)

Disconnects an existing path between **channel 1** and **channel 2**. Paths are created using [niSwitch Connect Channels](#) or [niSwitch Set Path](#).

ViSession	vi
ViConstString	channel 1
ViConstString	channel 2



niSwitch Disconnect All Channels (niSwitch_DisconnectAll)

Disconnects all existing paths.

ViSession	vi
-----------	----



niSwitch Get Path (niSwitch_GetPath)

Returns a string that uniquely identifies the path you created with [niSwitch Connect Channels](#). Pass this string to [niSwitch Set Path](#) to establish the exact same path in the future.

ViSession	vi
ViConstString	channel 1
ViConstString	channel 2
ViInt32	buffer size
ViChar (out)	path list []



niSwitch Set Path (niSwitch_SetPath)

Connects two channels by establishing an explicit path with the **path list** parameter. Use for applications where repeatability of the path is important, such as in calibrated signal paths.

If repeatability is not necessary, use [niSwitch Connect Channels](#). To obtain the exact path for a given connection, use [niSwitch Get Path](#).

ViSession	vi
ViConstString	path list

† Function names for C, C++, LabWindows™/CVI™, and Visual Basic are in parentheses.

ICON	FUNCTION NAME	TYPE	PARAMETER
------	---------------	------	-----------

Immediate Operations (continued)

	niSwitch Wait For Debounce (niSwitch_WaitForDebounce)	ViSession ViInt32	vi maximum time
Returns after all the paths that you created have settled.			
	niSwitch Switch Is Debounced? (niSwitch_IsDebounced)	ViSession ViBoolean (out)	vi is debounced
Returns the state of the switch module and indicates if all the created paths have settled.			
	niSwitch Can Connect Channels? (niSwitch_CanConnect)	ViSession ViConstString ViConstString ViInt32 (out)	vi channel 1 channel 2 path capability reference
Verifies that the switch module can create a path between the two channels specified in the channel 1 and channel 2 parameters. If the switch module can create a path, this function indicates whether the path is currently available given the existing connections.			

Scanning

	niSwitch Initiate Scan (niSwitch_InitiateScan)	ViSession	vi
Downloads the configured scan list and trigger settings to hardware, initiates the scan, and returns. Once the scan initiates, you cannot perform any other operation other than niSwitch Abort Scan or niSwitch Send Software Trigger , as well as the retrieval of attributes.			
	niSwitch Commit (niSwitch_Commit)	ViSession	vi
Downloads the configured scan list and trigger settings to hardware. niSwitch Commit is useful for arming triggers in a given order or to control when expensive hardware operations are performed.			
	niSwitch Abort Scan (niSwitch_AbortScan)	ViSession	vi
Aborts a scan in progress. Initiate a scan with niSwitch Initiate Scan .			
	niSwitch Configure Scan List (niSwitch_ConfigureScanList)	ViSession ViConstString ViInt32	vi scan list scan mode
Configures the scan list and scan mode used for 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: <code>com0->ch0; com0->ch1; com0->ch2;</code>			
To see the status of a scan, call niSwitch Is Scanning or niSwitch Wait For Scan Complete . Use niSwitch Configure Scan Trigger and niSwitch Initiate to configure the scan trigger and start the scan respectively.			
	niSwitch Set Continuous Scan (niSwitch_SetContinuousScan)	ViSession ViConstString ViInt32	vi scan list scan mode
Tells the driver whether to continuously loop the scan list (True) or to stop scanning after one pass through the scan list (False). Call niSwitch Abort Scan to halt a continuous scan.			

ICON	FUNCTION NAME	TYPE	PARAMETER
------	---------------	------	-----------

Scanning (continued)

	niSwitch Configure Scan Trigger (niSwitch_ConfigureScanTrigger)	ViSession	vi
	Configures the scan triggers for the scan last established with niSwitch Configure Scan List .	ViReal64	scan delay
		ViInt32	trigger input
		ViInt32	scan advanced output
	niSwitch Route Trigger Input (niSwitch_RouteTriggerInput)	ViSession	vi
	Routes the input trigger from the front or rear connector to a trigger bus line (TTLx).	ViInt32	input connector
		ViInt32	trigger bus line
		ViBoolean	invert
	niSwitch Route Scan Advanced Output (niSwitch_RouteScanAdvancedOutput)	ViSession	vi
	Routes the scan advanced output trigger from the front or rear connector to a trigger bus line (TTLx).	ViInt32	input connector
		ViInt32	trigger bus line
		ViBoolean	invert
	niSwitch Send Software Trigger (niSwitch_SendSoftwareTrigger)	ViSession	vi
	Sends a software trigger to the switch module specified by the NI-SWITCH session.		
	niSwitch Wait For Scan To Complete (niSwitch_WaitForScanComplete)	ViSession	vi
	Pauses until the switch module stops scanning or maximum time has elapsed.		
	niSwitch Switch Is Scanning? (niSwitch_IsScanning)	ViSession	vi
	Indicates the status of the scan.		

Relay Control

	niSwitch Relay Control (niSwitch_RelayControl)	ViSession	vi
	Controls individual relays of the switch module. When controlling individual relays, the protection offered by setting the usage of source channels and configuration channels is void.	ViConstString	switch name
		ViInt16	switch action
	niSwitch Get Relay Position (niSwitch_GetRelayPosition)	ViSession	vi
	Returns the relay position for the relay specified in the relay name parameter.	ViConstString	relay name
		ViInt16 (out)	position

Utility


niSwitch Initialize
(niSwitch_init)

Returns a session handle used to identify the switch module in all subsequent driver calls.

Refer to *Resource Name Syntax* at the end of this guide.

ViRsrc resource name
 ViBoolean id query
 ViBoolean reset device
 ViSession (out) vi


niSwitch Initialize With Options
(niSwitch_InitWithOptions)

Returns a session handle used to identify the switch module in all subsequent driver calls. Also can optionally set attributes of the switch module.

Refer to *Resource Name Syntax* at the end of this guide.

ViRsrc resource name
 ViBoolean id query
 ViBoolean reset device
 ViString option string
 ViSession (out) vi


niSwitch Reset
(niSwitch_Reset)

Resets the switch module to a known state.

ViSession vi


niSwitch Reset with Defaults
(niSwitch_ResetWithDefaults)

Resets the switch module and applies initial user-specified setting from the logical name used to initialize the session.

ViSession vi


niSwitch Get Channel Name
(niSwitch_GetChannelName)

Returns the channel string that is in the channel table at the specified index. Use this function/VI in a For Loop to get a complete list of valid channel names for the switch module. Use the Channel Count attribute to determine the number of channels.

ViSession vi
 ViInt32 index
 ViString (out) channel string


niSwitch Get Relay Name
(niSwitch_GetRelayName)

Returns the relay name string that is in the relay list at the specified index. Use this function/VI in a For Loop to get a complete list of valid relay names for the switch module. Use the Number of Relays attribute to determine the number of relays.

ViSession vi
 ViInt32 index
 ViString (out) relay string


niSwitch Self Test
(niSwitch_self_test)

Verifies the driver can communicate with the switch module.

ViSession vi
 ViInt16 (out) self test result
 ViChar (out) self test message []


niSwitch Revision Query
(niSwitch_revision_query)

Returns the revision numbers of the NI-SWITCH driver.

ViSession vi
 ViChar (out) instrument driver revision []
 ViChar (out) firmware revision []


niSwitch Disable
(niSwitch_Disable)

Places the switch in a quiescent state where it has minimal or no impact on the system to which it is connected.

ViSession vi


niSwitch Error Message
(niSwitch_ErrorMessage)

Converts a status code returned by an instrument driver function into a user-readable string.

ViSession vi
 ViStatus error code
 ViChar (out) error message [256]

ICON	FUNCTION NAME	TYPE	PARAMETER
------	---------------	------	-----------

Calibration



niSwitch Read Calibration Data (niSwitch_CalibrationDataRead)

Retrieves the measurement data, typically in terms of the amplifier offset, that was stored in the EEPROM. Also retrieves the stored calibration date.

Note Only supported by PXI-2501.

ViSession	vi
ViConstString	channel
ViInt32	calibration field
ViReal64 (out)	calibration data
ViInt32 (out)	year
ViInt32 (out)	month
ViInt32 (out)	day



niSwitch Write Calibration Data (niSwitch_CalibrationDataWrite)

Takes the measurement data, typically in terms of the amplifier offset, and stores the value in the EEPROM.

The operation automatically stores the current date along with the data in the EEPROM.

Note Only supported by PXI-2501.

ViSession	vi
ViConstString	channel
ViInt32	calibration field
ViReal64	calibration data

Interchangeability



niSwitch Clear Interchange Warnings (niSwitch_ClearInterchangeWarnings)

Clears the list of current interchange warnings.

ViSession	vi
-----------	----



niSwitch Reset Interchange Check (niSwitch_ResetInterchangeCheck)

Resets the interchangeability checking algorithms in the specific driver, thus ignoring all previous configuration operations.

ViSession	vi
-----------	----



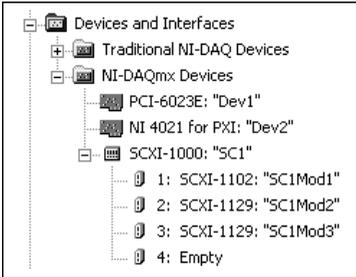
niSwitch Get Next Interchange Check (niSwitch_GetNextInterchangeCheck)

Returns the interchangeability warning associated with the IVI session.

ViSession	vi
ViConstString (out)	interchange warning

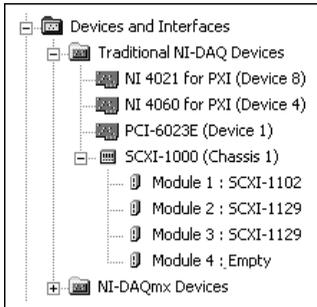
Resource Name Syntax

To establish a session with the correct switch module, you must pass a resource name to `niSwitch Initialize with Topology`. The syntax of the resource name depends on where in MAX you configured your switch module—under NI-DAQmx Devices, Traditional NI-DAQ Devices, or PXI System.



NI-DAQmx Devices

If you configured the switch module under NI-DAQmx Devices in MAX, the resource name is the string in quotes. For example, the resource name of the first SCXI-1129 in the following figure would be "SC1Mod2". Pass this string to `niSwitch Initialize With Topology`. You can rename the resource name for switch modules configured as DAQmx devices simply by clicking on the device in MAX and entering a new name.

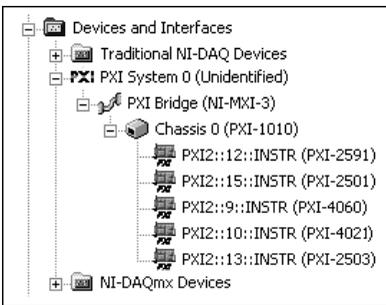


Traditional NI-DAQ Devices

If you configured the switch module in MAX under Traditional NI-DAQ Devices, the resource name syntax is:

`SCXI[chassis ID]::slot number`

For example, the resource name of the first SCXI-1129 for the following configuration would be "SCXI1::2". Pass this string to `niSwitch Initialize With Topology`.



PXI System

If you configured the switch module in MAX under PXI System, the resource name syntax is:

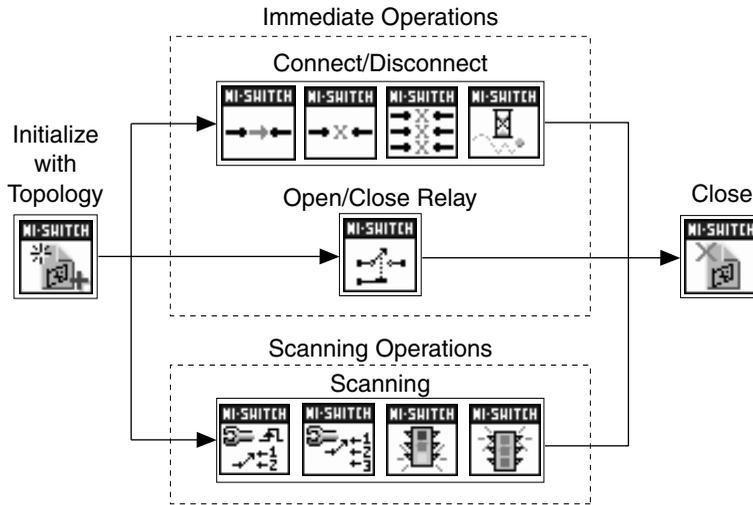
`PXI[bus number]::device number`

For example, the resource name of the PXI-2591 for the configuration to the left would be "PXI2::12". Pass this string to `niSwitch Initialize With Topology`.



Note If the module also appears under NI-DAQmx Devices, configure your PXI module under NI-DAQmx Devices instead of PXI System.

Programming Flow



CVI™, IVI™, National Instruments™, NI™, ni.com™, NI-DAQ™, and SCXI™ are trademarks of National Instruments Corporation. Product and company names mentioned herein are trademarks or trade names of their respective companies. For patents covering National Instruments products, refer to the appropriate location: **Help»Patents** in your software, the `patents.txt` file on your CD, or ni.com/patents.

© 2001–2003 National Instruments Corporation. All rights reserved.



323009B-01

Apr03