



**THE SIMULATION PLATFORM FOR
POWER ELECTRONIC SYSTEMS**

NI VeriStand Target Support User Manual Version 1.1

How to Contact Plexim:

☎	+41 44 533 51 00	Phone
	+41 44 533 51 01	Fax
✉	Plexim GmbH Technoparkstrasse 1 8005 Zurich Switzerland	Mail
@	info@plexim.com	Email
	http://www.plexim.com	Web

NI VeriStand Target Support User Manual

© 2022 by Plexim GmbH

The product described in this manual is furnished under a license agreement. The software may be used or copied only under the terms of the license agreement. No part of this manual may be photocopied or reproduced in any form without prior written consent from Plexim GmbH.

PLECS is a registered trademark of Plexim GmbH. VeriStand is a trademark of National Instruments. Neither Plexim GmbH, nor any software programs or other goods or services offered by Plexim GmbH, are affiliated with, endorsed by, or sponsored by National Instruments. Other product or brand names are trademarks or registered trademarks of their respective holders.

Contents

Contents	iii
1 Quick Start	3
Requirements	3
Run-time Engine Options	3
Install the Target Support Package	4
Install Required Software on Host PC	4
Install Cross Compiler Toolchain for NI Linux RT Target	4
NI Software Requirements	5
Install Required Software on NI Linux RT Target	7
Upload a Model	7
Connect to the Embedded Application	8
Connect via VeriStand	9
Connect via PLECS External Mode	9
Start, stop, and debug via linux shell or SSH	9
2 NI VeriStand Target Support Architecture	11
Overview	11
The Embedded Code Generation Workflow	12
NI VeriStand Model Framework	12
Engine Architectures	13
VeriStand Engine system architecture	14

Custom Engine system architecture	14
Timing and Synchronization	15
Hardware timed single point	16
Software timed single point	16
3 NI VeriStand Coder Options	17
4 Supported Hardware	19
PXI Controllers	19
PXI Modules	19
PXI Multifunction IO	19
PXI Analog Output	20
5 NI VeriStand Target Support Library Component Reference	21
DAQ Analog In	22
DAQ Analog Out	23
DAQ Counter Edge	24
DAQ Counter Out	26
DAQ Counter Position	27
DAQ Digital In	29
DAQ Digital Out	30
Signal Inport	31
Signal Outport	32

Quick Start

Requirements

The PLECS NI VeriStand™ Target Support Package (TSP) generates models compatible with NI VeriStand and Intel x64 NI Linux Real-Time (RT) hardware. The TSP is targeted toward real-time control applications and supports NI PXIe DAQ hardware modules with a focus on X-Series DAQ devices.

In order to use the PLECS NI VeriStand TSP you will need at the minimum:

- A host computer with Microsoft Windows 10
- PLECS Blockset or Standalone 4.5.4 or newer
- PLECS Coder
- A cross compiler to compile code for a NI Linux Real-Time target from Windows

Additional NI software packages are required on the host PC and NI Linux RT target device for additional functionality.

If you have not done so yet, please download and install the latest PLECS release on your host computer.

Run-time Engine Options

There are three primary methods to run a real-time application on the NI Linux RT hardware. The minimal software required on the host PC and NI Linux RT hardware will depend on the chosen run-time engine. The required software for the VeriStand Engine and the Custom Engine are compatible.

- 1 VeriStand Engine** The VeriStand Engine executes the model on the real-time target. The VeriStand Engine manages the software timing, hardware IO, and communication with a host PC running the VeriStand software. The user can interact with the model executing on the real-time target through both PLECS and NI VeriStand software.
- 2 Custom Engine** NI provides drivers to natively interface with the hardware IO using C/C++. The Custom Engine executes the model on the real-time target and uses the native drivers to manage the hardware IO. The user can interact with the model executing on the real-time target through PLECS.
- 3 Model Only** This build option compiles a model with a *.so extension. The model is compatible with the NI VeriStand Model Framework and can be manually imported into a VeriStand project. All mapping between the hardware and model must be performed manually.

A more comprehensive description of the different build options is available in Chapter 2.

Install the Target Support Package

Download the appropriate ZIP archive, extract it and move the NI_VeriStand folder to the PLECS Coder target support packages path e.g. to HOME/Documents/PLECS/CoderTargets.

In PLECS, choose **Preferences...** from the **File** drop-down menu to open the PLECS Preferences dialog. Navigate to the **Coder** tab and click on the **Change** button to select the HOME/Documents/PLECS/CoderTargets folder. The NI VeriStand target should be listed under **Installed targets**. You will also see these targets available in the **Coder + Coder options...** window in the drop-down menu on the **Target** tab.

A set of basic demos is also included with the NI VeriStand TSP.



Install Required Software on Host PC

Install Cross Compiler Toolchain for NI Linux RT Target

A cross compiler is required to build models suitable for execution on the NI Linux RT target from a Windows PC.

NI provides GNU C & C++ Compilers for x64 Linux (Windows host) for download on their website: <https://www.ni.com/en-us/support/downloads/software-products/download.gnu-c--c--compile-tools-x64.html>. An NI user account is required to complete the download.

Extract the files after downloading the cross compiler toolchain, for example to C:\toolchains\oecore-i686-core2-64-toolchain-6.0. The extracted folder should contain another directory titled sysroots.

To configure the PLECS Coder to use the cross compiler, select **Preferences...** from the **File** drop-down menu and open the PLECS Preferences dialog. Click the **Coder** tab to see the installed targets. There is a  icon next to the **NI VeriStand** entry in the **Family** column indicating the toolchain is not yet configured. After clicking the icon a dialog will appear where the user can enter the directory of the cross compiler toolchain, e.g. C:\toolchains\oecore-i686-core2-64-toolchain-6.0. Once the directory is entered, you will see a  icon in the **Family** column, as shown in Figure 1.1.

NI Software Requirements

Additional software is required on the host PC depending on the chosen runtime engine. The software is available for download from the NI website and the NI Package Manager tool. Refer to NI documentation for software compatibility and licensing requirements.

The following major software packages and drivers should be installed. The NI Package Manager will install other required software packages as part of these major software components. The default install options are recommended.

- VeriStand (*required for VeriStand Engine only*)
- NI Measurement & Automation Explorer (MAX)
- NI Linux RT PXI System Image
- NI-DAQmx 19.0+
- NI-VISA 19.0+
- NI-SLSC 19.0+

Once the tools are installed select **Preferences...** from the **File** drop-down menu and open the PLECS Preferences dialog. Click the **Coder** tab to see the installed targets. Click the icon next to the **NI VeriStand** entry in the **Family** column. A dialog will appear where the user can enter the installation directory of the VeriStand software and NI DAQmx libraries.

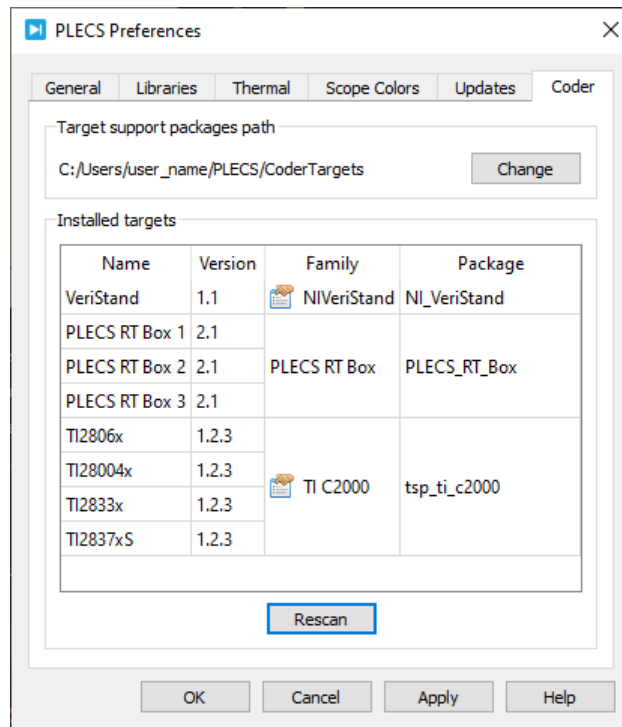


Figure 1.1: Configuring the target support package and toolchain directory

Paths to the x86 and x64 VeriStand installations are required for the VeriStand Engine workflow. The TSP assumes the default paths based on VeriStand version specified at build-time. If the software is installed in the default locations, then these fields can be left blank. If the VeriStand application is installed into a non-default location then the proper paths must be specified.

The default x86 path for VeriStand 2019 is:

```
C:\Program Files (x86)\National Instruments\VeriStand 2019
```

The default x64 path for VeriStand 2019 is:

```
C:\Program Files\National Instruments\VeriStand 2019
```

The NI DAQmx libraries are required for the Custom Engine workflow. The default path is below and should contain folders titled lib64 and include:

```
C:\Program Files (x86)\National Instruments\Shared\
ExternalCompilerSupport\C
```

Install Required Software on NI Linux RT Target

Install the required software on the remote target with the NI Measurement & Automation Explorer (MAX) tool. Use the PXI Linux RT System Image downloaded to the host PC to image NI Linux RT Target. Back up any previous system images if required.

The NI Linux RT Target software requirements for this TSP align with VeriStand software requirements. Refer to the “readme” section of the VeriStand version of the software currently installed on the host PC. It is recommended that the PXI Linux RT System Image suggested in the VeriStand documentation is used, for example when using VeriStand 2019 R2, the PXI System Image for Q1 2019. This aligns the driver versions included with the VeriStand installer for the host PC with that deployed to the real-time target.

The following minimal software package set must be installed on the NI Linux RT Target using NI MAX, if not part of the provided system image.

- NI VeriStand Engine
- PXI Platform Services
- NI-DAQmx 19.0+
- NI-VISA 19.0+
- NI-SLSC 19.0+

Upload a Model

Open the **Coder Options** window by selecting the **Coder + Coder options...** drop-down menu. In the **General** tab enter in a discretization step size for the model. Next, navigate to the **Target** tab and set the VeriStand target.

The **Build type** parameter in the **Coder options... + Target** menu is used to select between the different run-time engines. The **Deploy to target after build** checkbox in the **Coder options... + Target** selects between deploying the real-time application to the target automatically or simply generating the required files. If the box is unchecked, the user must open the generated files in VeriStand or manually load them onto the real-time target. The **Model Only** option cannot deploy automatically.

The **Target IP Address** and login information for the remote NI Linux RT target should be entered into the remaining fields in the **Target** tab.

A hardware configuration file exported from NI MAX is required to know the available hardware resources on the remote target. In NI Max, click the **File**

+ Export... drop-down option. Choose the appropriate target machine from the **Export from system** dropdown. Set the **File type** to NI Configuration Export File (*.nce) and click **Next**. In the next screen expand the **Devices and Interfaces** option in the left-hand menu, select the relevant chassis and modules, and export the file. Then, navigate back to the PLECS **Coder Options + Target** to refer to the exported *.nce file.

Once the target is fully configured, click **Build** in the **Coder Options + Target** menu to start the build process.

Note The NI VeriStand software must be installed on the host machine and open to use the VeriStand Engine workflow. The NI VeriStand application is used as a gateway to deploy the model and communicate via the External Mode. NI VeriStand is not required when using the Custom Engine or Model Only build types.

Connect to the Embedded Application

There are two means to connect to the embedded application running on a remote target. One method uses VeriStand and NI tools. The other method uses the PLECS External Mode feature. Once connected via VeriStand or the PLECS External Mode, the user can observe real-time waveforms and change certain simulation parameters. In both instances the **Enable External Mode** checkbox must be selected when building the project.

If using the VeriStand Engine then the remote application is fully compatible with the NI toolchain. One can use NI software such as VeriStand and TestStand for data logging, stimulus generation, configuring alarms, and automated testing.

The PLECS External Mode can also be used to connect to the remote target for modules deployed using either the VeriStand or Custom Engine. This workflow enables benchmarking of the real-time execution against simulation results in the PLECS Scope.

Connect via VeriStand

Open the generated VeriStand project. The project will have a *.vsproj extension and be located in model_name_codegen folder adjacent to the saved model or the specified codegen director. Once the project is open in VeriStand, click **Operate + Deploy** if the deploy to target option was not selected in PLECS, and then click **Operate + Connect**.

Connect via PLECS External Mode

To establish a communication link with your target, open the **Coder options... + External Mode** tab.

If the VeriStand engine build type is selected then the **Target device** field should refer to the local PC (e.g. localhost). The VeriStand application must also be open on the host PC.

When the Custom engine is used then the **Target device** address should be an IP address referring to the remote target (e.g. enter 192.168.0.101 for a remote machine with that IP address).

Click the **Connect** button and if the connection is successful you will see the trigger controls activate.

Set the **Number of samples** parameter to 200 and click on the **Start autotriggering** button. You will now see real-time data from the remote target in the PLECS Scopes. You can synchronize the data capture to a specific trigger event. To do so, change the **Trigger channel** selection from **Off** to the desired signal. The Scope will now show a small square indicating the trigger level and delay. If the level or delay are outside the current axes limits, a small triangle will be shown instead. Drag the trigger icon to change the trigger level; drag it with the left mouse-button pressed to change the trigger delay. Both parameters can also be set in the External Mode dialog.

While the PLECS model is connected via the External Mode, the model is locked against modifications. To disconnect from the remote target and other External Mode connections, click on the **Disconnect** button or close the Coder Options dialog.

Start, stop, and debug via linux shell or SSH

When the Custom engine build type is selected the user can manually start and stop the Custom Engine executing on the embedded target. By default the

project is deployed into a folder titled `veritarget` in the user's home directory. The `veritarget` directory contains the engine application, `*.out`, the compiled model with an `*.so` extension, and debug logs with `*.log` and `*.err` extensions.

Note The log files will contain useful debugging information for the Custom Engine In the event of an unanticipated engine or driver error.

NI VeriStand Target Support Architecture

Overview

As a separately licensed feature, the PLECS Coder can generate C code from a simulation model to facilitate code generation for real-time targets. Plexim provides and maintains target support packages that enables the PLECS Coder to generate code specific to particular processor, hardware, or modeling environments. Examples include the NI VeriStand modeling environment, the PLECS RT Box hardware, and the Texas Instruments C2000 family of microprocessors.

NI VeriStand is a software tool for the deployment and test of real-time applications on NI hardware. In addition to integrated data logging, test sequencing, and monitoring functionality, the software can incorporate custom simulation models that are executed in real-time. With the NI Veristand Target Support Package, PLECS can generate simulation models compatible with the VeriStand software.

Key advantages of using the PLECS Coder for NI VeriStand are:

- Streamlined workflow to quickly transition from initial model concepts to an application running in real-time on a hardware target.
- The embedded control logic can be tested extensively inside the PLECS simulation environment prior to real-time deployment.
- Hardware peripherals are configured using intuitive parameters and do not require detailed knowledge of the embedded programming environment.
- The full application is generated, compiled, and uploaded to the target device directly from the PLECS environment.

- One can connect to the real-time application via the PLECS External Mode or using the NI tools such as VeriStand.

The Embedded Code Generation Workflow

The embedded workflow is designed for you to easily transition from a PLECS model to an embedded code generation project without having to build and maintain separate models. A typical embedded code generation workflow consists of the following steps:

- 1** Design and simulate a model in PLECS. The model typically contains one subsystem representing the portion of the model that will ultimately be deployed to the real-time target .
- 2** Add components from the target support library into the subsystem representing the real-time target configure the model and hardware IO.
- 3** Run an offline simulation. All peripheral components in the target support library have behavioral offline models to facilitate the transition from simulation to real-time deployment.
- 4** Select a discretization step size and nominal control task execution frequency. When generating C code, the PLECS Coder will use the discretization step size to automatically transform all continuous states in the controller to the discrete state-space domain using the Forward Euler method. The control task execution frequency is based on the discretization step size and specifies the nominal execution rate of the model.
- 5** Build the project and deploy the real-time application to the remote target using PLECS or NI VeriStand.
- 6** Connect to the remote target using External Mode or NI tools such as VeriStand and TestStand to get real-time data and control execution of the real-time application.

NI VeriStand Model Framework

One the real-time target the VeriStand Engine is responsible for managing the execution of the simulation model, reading hardware IO, managing test procedures and alarms, and communicating results over a network to a host PC. The NI VeriStand Model Framework provides an interface between the

VeriStand Engine and compiled simulation models, such that the VeriStand Engine can execute the model. The NI VeriStand TSP generates code from PLECS that conforms to the NI VeriStand Model Framework.

The NI VeriStand Model Framework has provisions for:

- **Inports** to feed signals into the compiled model and are updated every model step.
- **Outports** to feed signals computed in the compiled model into the VeriStand environment and are updated every model step.
- **Parameters** serve as tunable variables within the model and can be changed to impact the behavior of a simulation.
- **Signals** to monitor internal variables and calculations within the model, similar to a probe or test point.

Inports and Outports can be mapped to hardware IO to have the simulation model interact with other hardware. Hardware components in the NI VeriStand TSP library automatically creates the required Inports or Outports into simulation model and maps the appropriate hardware signal to that port. There also is the provision to create generic Inports and Outports to a simulation model which the user can then manually configure in the VeriStand software.

Every Scope and Display block within a PLECS model are automatically converted to Signals in the VeriStand environment so the same values are available in both tools.

Lastly, parameters that are specified as tunable in the **Coder Options + Parameter Inlining** tab correspond to tunable parameters in VeriStand.

Engine Architectures

The NI VeriStand TSP supports two different architectures for the real-time engine. The overall architecture is divided into three key components:

- A Windows host PC with PLECS and optionally VeriStand
- One or more PXI remote targets running NI Linux RT
- One or more devices under test (DUT)

The host PC is used to generate and compile simulation models from PLECS and configure the VeriStand project, if applicable. The host PC communicates with the remote target via TCP/IP connection. The host PC deploys simulation

models to the remote target and interacts with the target, for example receiving real-time data and changing parameters.

On the remote target, the VeriStand Engine or Custom Engine controls the real-time execution of the simulation model, synchronizes model IO with the hardware IO, and provides an interface to the host PC. The hardware IO is comprised of PXI modules which connect to a device under test (DUT) through supported interfaces such as analog and digital signals.

Note A system reboot may be required when changing between the Custom Engine and VeriStand Engine, as certain hardware resources may not be released.

VeriStand Engine system architecture

Figure 2.1 shows the key components when the VeriStand Engine is used to run the model. PLECS communicates with the VeriStand application on the host PC through the ASAM XIL API. This interface is used to create VeriStand projects from PLECS, deploy projects to the remote hardware, and receive real-time data in the PLECS external mode.

Note that PLECS does not communicate directly to the remote target, and therefore the VeriStand application must be open at all times using this workflow. With the remote target running the VeriStand Engine, VeriStand's suite of workspace and third-party tools are accessible.

Alternatively, PLECS generates a compiled simulation model which is then manually imported into a VeriStand project. Hardware configuration is then performed in the VeriStand user interface prior to deploying to the real-time target.

Custom Engine system architecture

Figure 2.2 shows the key configuration for the Custom Engine workflow. PLECS communicates directly with the remote target. VeriStand does not need to be installed on the host PC and an active VeriStand license is not required. However, the advanced functionality that comes with the VeriStand toolchain is not available in this architecture.

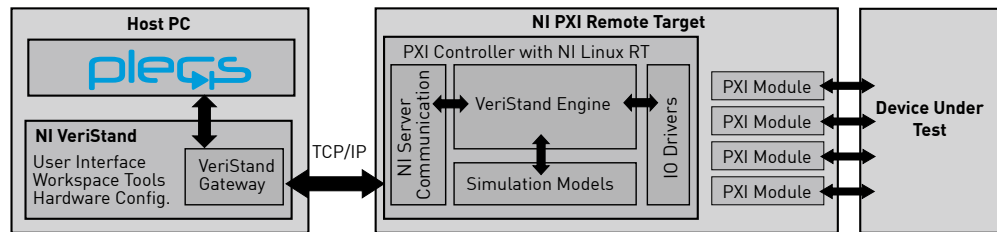


Figure 2.1: VeriStand Engine Architecture

The Custom Engine is responsible for executing the simulation model and interacting with the hardware drivers. The Custom Engine and server used for communication are automatically deployed to the target upon build.

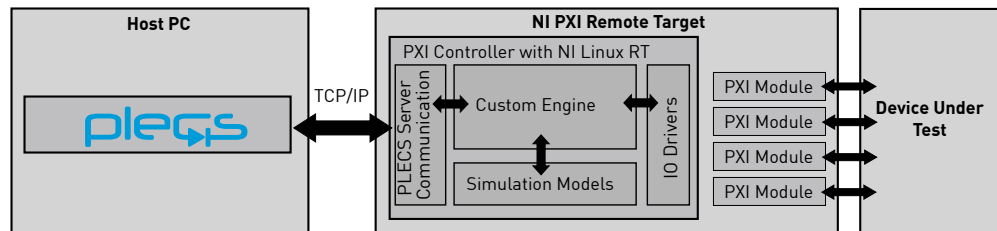


Figure 2.2: Custom Engine Architecture

Timing and Synchronization

NI PXI hardware IO can consist of several different types of IO across multiple modules. Synchronizing the reading and writing of the IO across different modules is critical to ensure data consistency and reliability of the real-time system. Both the VeriStand Engine and Custom Engine automatically select a synchronization method based on the hardware resources available.

The NI VeriStand TSP is configured for single-point execution, meaning that a single point of data is read or written every model step without a buffer. Single-point execution can be hardware timed or software timed. Hardware timed synchronization is also referred to as signal based synchronization.

Hardware timed single point

Where possible, hardware timed synchronization is used. Hardware timed synchronization uses hardware clocks and digital triggers that are shared between the PXI controller and modules across the PXI backplane to synchronize events. The hardware sampling latches the IO so all signals are updated simultaneously and then results communicated between the PXI controller and modules.

Hardware synchronization methods are further categorized into sample clock and reference clock synchronization. Both the VeriStand Engine and Custom Engine use reference clock synchronization when possible, as it has the lowest module-to-module timing skew, otherwise sample clock synchronization is used. One module is selected as the master device within a chassis and distributes timing signals to other modules.

Some functionality requires hardware based synchronization. Specifically DAQ Counter Inputs require a hardware sample clock.

The master device is also used as a timing source for the execution of the runtime engine. Each synchronization pulse triggers other iteration of the primary control loop and model. The NI VeriStand requires at least one DAQ analog input when using hardware based synchronization for the primary control loop

Software timed single point

Some PXI modules and channels do not support hardware based synchronization. Software timed sampling is used when a hardware synchronization method is not available. Software timed IO are accessed at the time the read or write function is called within the software. The operating system and software determine the sample time, not a common hardware signal. Signals are measured successively and are not latched during read/write events.

In the event that there is no hardware timing source in the entire system, then the model execution itself will be software timed.

NI VeriStand Coder Options

The **Target** page contains code generation options which are specific to the NI VeriStand Target Support Package.

General

Build type This setting specifies the action of the **Build** button. The build process creates an application that can run on the VeriStand Engine, a Custom Engine, or a compiled model that is suitable for manual import into VeriStand.

VeriStand version Selects the version of the VeriStand software installed on the PC, if applicable.

Deploy to target after build The application will automatically be deployed to the target when this option is selected.

Enable external mode This setting adds code that enables the External Mode on the target device.

Used key based authentication Use SSH keys for user authentication instead of plain-text passwords. Requires manual configuration of SSH keys using the default SSH client on the Host PC.

Hardware configuration file from NI Max Defines the NI Configuration Export file for the target device which information regarding the target hardware. NI Max is used to export the file, which has an `.nce` extension.

Target IP address Specifies the IP address of the remote target. The IP address must be specified as 4 bytes separated by dots, e.g. 192.168.0.101

Target username Username to log into the remote target.

Target password Password to log into the remote target.

Analog input voltage range [min,max] The allowed voltage range for analog input signals.

Analog output voltage range [min,max] The allowed voltage range for analog output signals.

Veristand project files

The user can specify optional files to include in the generated VeriStand project. These files must be created and saved in VeriStand prior to incorporating them into the code generation project. If a given file is not specified, a corresponding blank file is automatically generated.

Workspace screen file Include a Workspace screen in the generated VeriStand project. The file must have a *.nivsscreen extension.

Editor screen file Include an Editor screen in the generated VeriStand project. The file must have a *.nivsscr extension.

Calibration file Include a Calibration file in the generated VeriStand project. The file must have a *.nivscf extension.

Supported Hardware

The hardware listed in this chapter supports automatic configuration by the NI VeriStand TSP. Manual configuration of other hardware is possible in the VeriStand software. Contact the Plexim support team at support@plexim.com to inquire about other hardware modules.

PXI Controllers

The NI VeriStand TSP supports Intel x64 based PXI Controllers with NI Linux Real-Time. Refer to NI documentation for a full list of available real-time controllers and operating system compatibility.

The following PXI Controllers are supported:

- PXIe-8821
- PXIe-8840
- PXIe-8861
- PXIe-8880
- PXIe-8881

PXI Modules

PXI Multifunction IO

The following Multifunction IO devices are supported:

- PXIe-6349
- PXIe-6356
- PXIe-6358

- PXIe-6366
- PXIe-6368
- PXIe-6376
- PXIe-6378
- PXIe-6386
- PXIe-6396

PXI Analog Output

The following PXI Analog Output devices are supported:

- PXI-6723
- PXI-6733

NI VeriStand Target Support Library Component Reference

This chapter lists the contents of the NI VeriStand Target Support library in alphabetical order.

DAQ Analog In

Purpose Output the measured voltage of an analog input channel.

Library NI VeriStand

Description



This block configures an analog input measurement for a DAQ module. The block output signal represents the measured voltage at the analog input terminal. The output is scalable and can be used with an offset, where the output signal is calculated as $\text{input} * \text{Scale} + \text{Offset}$.

Parameters

Main

Slot

Specifies the location of the DAQ module in the chassis.

Analog input channel(s)

Index of the analog input channels. For vectorized input signals a vector of input channel indices must be specified.

Scale(s)

A scale factor for the input signal.

Offset(s)

An offset for the scaled input signal.

Minimum output voltage(s)

The lowest voltage value that the input will reach.

Maximum output voltage(s)

The highest voltage value that the input will reach.

Mode

Analog input configuration

Selects the input configuration for the analog input.

DAQ Analog Out

Purpose Set an analog output.

Library NI VeriStand

Description



This block generates an analog output voltage on the DAQ module output terminals. The output is scalable and can be used with an offset, where the output voltage signal is calculated as $\text{input} * \text{Scale} + \text{Offset}$. Output voltage limitations can also be set.

Parameters

Slot

Specifies the location of the DAQ module in the chassis.

Analog output channel(s)

Index of the analog output channels. For vectorized output signals a vector of output channel indices must be specified.

Scale(s)

A scale factor for the output signal.

Offset(s)

An offset for the scaled output signal.

Minimum output voltage(s)

The lowest voltage value that the output will reach.

Maximum output voltage(s)

The highest voltage value that the output will reach.

DAQ Counter Edge

Purpose Output the number of rising or falling edges on a counter input.

Library NI VeriStand

Description



This block configures a counter instance as an edge counter. The counter increments or decrements when it detects a rising or falling edge of a digital signal. The increment direction is configurable through a parameter or through an external digital signal. When the count direction is controlled by an external signal, a logical high direction input results in a counter increment and a logical low a counter decrement.

The default counter input channels can be used or the PFI inputs specified by the user.

Parameters

Main

Slot

Specifies the location of the DAQ module in the chassis.

Counter

Selects the device counter instance.

PFI channel selection

Use the default PFI input channels for the counter instance or explicitly specify the channels.

Count edge

Specifies if rising or falling edge detection is used to increment or decrement the count.

Count direction

Specifies to increment or decrement the count on each detected edge. The count direction can also be controlled by an external digital signal.

Initial count

Sets the initial value of the counter.

PFI channels**PFI channel**

Digital input channel for edge detection when PFI channels are specified.

Count direction PFI channel

Digital input channel for count direction when PFI channels are specified.

DAQ Counter Out

Purpose Generate a configurable pulse train using a counter output. The counter is output is used to generate pulse width modulated (PWM) signals.

Library NI VeriStand

Description



Creates a counter to generate digital pulses. The block input is a duty cycle that, along with the specified **Carrier frequency**, sets the interval between pulses. The duty cycle input accepts values the range of 0 to 1.

Note that the DAQ Counter modules have a hardware limitation resulting in a minimum on state or off state duration of two ticks of the Timebase clock. Duty cycles equal to exactly 0 or 1 are set to the closest achievable value.

The default counter output channel can be used or the PFI output channel specified by the user.

Parameters

Main

Slot

Specifies the location of the DAQ module in the chassis.

Counter

Selects the device counter instance.

PFI channel selection

Use the default PFI output channels for the counter instance or explicitly specify the channels.

Carrier frequency

The frequency of the carrier in hertz (Hz).

Carrier phase shift

The phase shift of the carrier signal, in p.u. of the carrier period.

Polarity

Defines the logical signal of the output when an active state is detected. The active state occurs when the modulation index exceeds the carrier.

PFI channel

PFI channel

Digital output channel for the counter when PFI channels are specified.

DAQ Counter Position

Purpose Counts edges generated from a quadrature encoder.

Library NI VeriStand

Description



The DAQ Counter Position counts edges which are generated from a quadrature encoder. The *A*, *B*, and *Z* outputs of the encoder are connected to the counter inputs of the DAQ Module. The block outputs the current counter value.

The counter counts up or down depending on the sequence of input pulses. The counter value will increase when the direction of rotation results in the rising edge of *A* leading the rising edge of *B* and will decrease in the opposite direction of rotation.

The **Decoding** parameter determines which edges result in an increment or decrement of the counter depending on the direction of rotation. The Count leading edge of *A* increments the counter on the rising edge of *A* when *A* leads *B* and decrements on the falling edge of *A* when *B* leads *A*. With Count both edges of *A* the counter increments or decrements on both the rising and falling edges of *A*. Similarly, the Count both edges of *A* and *B* option increments and decrements on both the rising and falling edges of the *A* and *B* signals.

If connected and configured by the **Counter reset method** parameter, the counter is reset when the rising edge of the index input is detected and the *A* and *B* signal conditions specified in the **Index mode** are met. The *Z* index signal alignment with *A* and *B* depends on the specific encoder hardware. The free-running counter will only reset once the maximum or minimum count supported by a 32 bit signed integer is reached.

Parameters

Main

Slot

Specifies the location of the DAQ module in the chassis.

Counter

Selects the device counter instance.

PFI channel selection

Use the default PFI output channels for the counter instance or explicitly specify the channels.

Index mode

Specifies the required states of the *A* and *B* inputs to latch the *Z* index counter reset signal.

Decoding

Specifies how to count and interpret the pulses that the encoder generates *A* and *B* inputs.

Counter reset method

Selects whether the counter should be reset by a positive pulse on the index input or on overflow only.

PFI channels

PFI channels [A,B,Z]

A vector of input channels for the *A*, *B*, and *Z* inputs when the PFI channels are specified.

DAQ Digital In

Purpose Read a digital input.

Library NI VeriStand

Description



The output signal is 1 if the input voltage is higher than the high level input voltage threshold, V_{IH} , and 0 if it is lower than the low-level input voltage, V_{IL} . For other input voltages the output signal is undefined. Refer to the module data sheet for the hardware electrical characteristics.

During an offline simulation the block behaves like a simple feedthrough.

Parameters

Slot

Specifies the location of the DAQ module in the chassis.

Port

Specifies the port number of the digital signal.

Digital in channel(s)

Index of the digital input channel for a given port. For vectorized input signals a vector of input channel indices must be specified.

DAQ Digital Out

Purpose Set a digital output.

Library NI VeriStand

Description The output is set low if the input signal is zero and is set high for all other values.



During an offline simulation the block behaves like a simple feedthrough.

Parameters

Slot

Specifies the location of the DAQ module in the chassis.

Port

Specifies the port number of the digital signal.

Digital out channel(s)

Index of the digital output channel for a given port. For vectorized output signals a vector of output channel indices must be specified.

Signal Inport

Purpose Signal inport connection to a compiled model.

Library NI VeriStand

Description



The component serves as a generic inport connection to a compiled model. When deployed to a hardware target the block output will remain zero unless the inport is explicitly connected to a data source in the VeriStand environment.

During an offline simulation the block behaves like a simple feedthrough.

Parameters

Data Type

Specifies inport data type.

Signal Output

Purpose Signal output connection to a compiled model.

Library NI VeriStand

Description The component serves as a generic output connection to a compiled model. When deployed to a hardware target the signal will not connect to any hardware unless explicitly connected in the VeriStand environment.



During an offline simulation the block behaves like a simple feedthrough.

Parameters **Data Type**
Specifies output data type.

plexim
electrical engineering software
