Using the NI SoftMotion™ Axis Interface

This document is designed to get you started using the LabVIEW NI SoftMotion Module to create a customized hardware interface between the NI SoftMotion function block API and hardware devices. NI SoftMotion provides VIs and functions that allow you to use the NI SoftMotion function block API to communicate with hardware devices as well as to perform path planning, trajectory generation, output control, PID control, and synchronization.

Contents

Conventions

The following conventions are used in this guide:

» — The » symbol leads you through nested menu items and dialog box options to a final action. The sequence **File»Page Setup»Options** directs you to pull down the **File** menu, select the **Page Setup** item, and select **Options** from the last dialog box.

This icon denotes a tip, which alerts you to advisory information.

This icon denotes a note, which alerts you to important information.

bold — Bold text denotes items that you must select or click in the software, such as menu items and dialog box options. Bold text also denotes parameter names.

italic — Italic text denotes variables, emphasis, cross-references, or an introduction to a key concept. Italic text also denotes text that is a placeholder for a word or value that you must supply.

`monospace` — Text in this font denotes text or characters that you should enter from the keyboard, sections of code, programming examples, and syntax examples. This font is also used for the proper names of disk drives, paths, directories, programs, subprograms, subroutines, device names, functions, operations, variables, filenames, and extensions.

Required Components

This section lists the software and hardware used in the tutorial. This section also lists documents you may find helpful while completing the tutorial.

Required Software

The following software is required for this tutorial.

- ❑ NI LabVIEW 2009 or later
- ❑ LabVIEW Real-Time Module 2009 or later
- ❑ LabVIEW FPGA Module 2009 or later
- ❑ NI-RIO 3.2.0 or later
- ❑ LabVIEW NI SoftMotion Module Premium 2009 or later

Note Refer to `ni.com/motion/softmotion` for information about the different available NI SoftMotion packages.

- ❑ NI-Motion 8.0 or later driver software

Required Hardware

The following hardware is required for this tutorial.

- ❑ CompactRIO controller and chassis
- ❑ Power supply for the controller
- ❑ Ethernet connection and cable
- ❑ NI 9263 AO module
- ❑ NI 9401 DIO module
- ❑ Servo motor, drive, and encoder for each axis

Related Documentation

The following documents contain information that you may find helpful as you read this tutorial:

- *NI SoftMotion Module Help*—Use this help file to learn about using NI SoftMotion in LabVIEW including information about function blocks and using NI SoftMotion with the LabVIEW Project. To access this help file from LabVIEW, select **Help»Search the LabVIEW Help**, then expand the *NI SoftMotion Module* book on the **Contents** tab.
- *NI-Motion Help*—Use this help file to learn about the NI-Motion driver software, including background, configuration, and programming information. The purpose of this help file is to provide a basic understanding of the NI-Motion driver software, and provide programming steps and examples to help you develop NI-Motion applications. To access this help file select **Start»All Programs» National Instruments»NI-Motion»Documentation» NI-Motion Help**.
- Operating instructions for the controller and modules (shipped with the hardware and available at `ni.com/manuals`).
- *LabVIEW Help*—Use the *LabVIEW Help* to access information about LabVIEW programming concepts, step-by-step instructions for using LabVIEW, and reference information about LabVIEW VIs, functions, palettes, menus, tools, properties, methods, events, dialog boxes, and so on. The *LabVIEW Help* also lists the LabVIEW documentation resources available from National Instruments. Access the *LabVIEW Help* by selecting **Help»Search the LabVIEW Help**.
- *Getting Started with LabVIEW*—Use this document as a tutorial to familiarize yourself with the LabVIEW graphical programming environment and the basic LabVIEW features you use to build data acquisition and instrument control applications. Access the *Getting Started with LabVIEW* PDF by selecting **Start»All Programs» National Instruments»LabVIEW»LabVIEW Manuals» LV_Getting_Started.pdf**.
- *Getting Started with the LabVIEW Real-Time Module*—Use this document to learn how to develop a real-time project and VIs, from setting up RT targets to building, debugging, and deploying real-time applications. Access the *Getting Started with the LabVIEW Real-Time Module* PDF by selecting **Start»All Programs»National Instruments»LabVIEW» LabVIEW Manuals»RT_Getting_Started.pdf**.
- *CompactRIO Reference and Procedures (FPGA Interface)*—Use this help file to learn about using the CompactRIO system in FPGA Interface programming mode. To access this help file from LabVIEW, select **Help»Search the LabVIEW Help**, then expand **Real-Time Module** on the **Contents** tab and select **CompactRIO Reference and Procedures (FPGA Interface)**.

Motion Controller Overview

A motion controller is the center of a typical motion system. A motion controller typically contains supervisory control, trajectory generation, a spline engine, a control loop, and I/O.

The controller converts high-level commands from the user into command signals used by drives to move actuators. The motion controller also monitors the system for error conditions, faults, and asynchronous events that can cause the system to change speed, direction, or start/stop the actuators. Figure 1 illustrates the parts and processes of a typical motion controller.

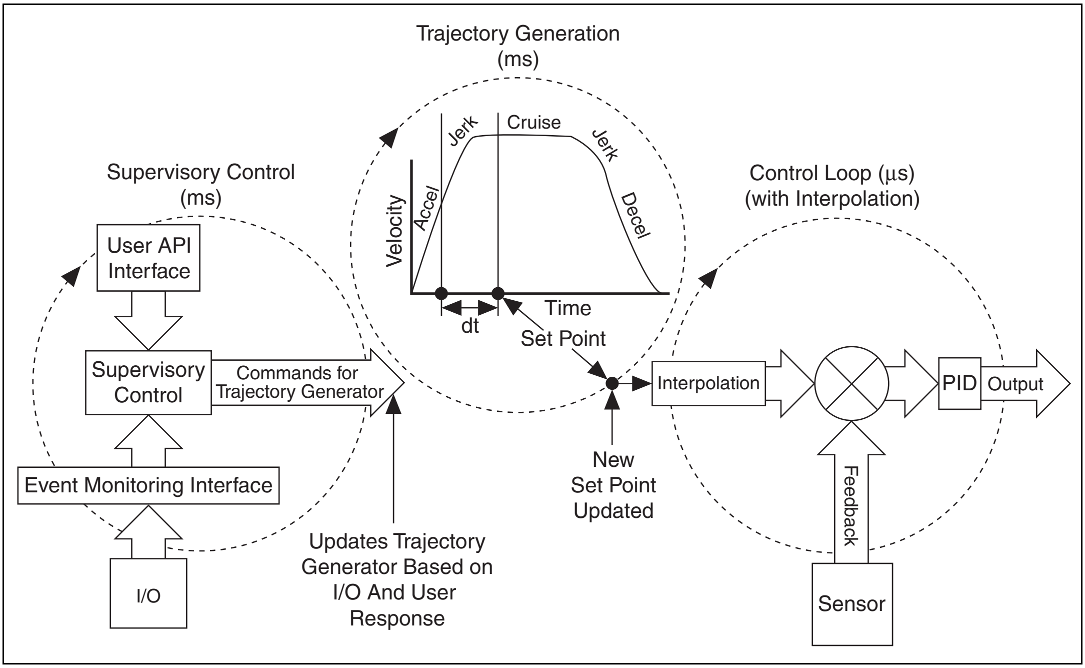


Figure 1. Three Loops of a Typical Motion Controller

Supervisory Control

The supervisory control is the main loop of the motion controller. This loop processes commands from the user and signals the trajectory generator to start or stop moves. The supervisory control loop also monitors all I/O needed to perform initialization tasks, such as finding the reference position or origin. This loop also monitors the system for faults, and aids in synchronizing moves relative to changes in external conditions.

Trajectory Generator

The trajectory generator is a path planner that creates setpoints for the control loop. The trajectory generator creates new setpoints every loop period, based on move constraints provided by the user. These move constraints include the maximum velocity, maximum acceleration/deceleration, and maximum jerk that the mechanical system can tolerate. The setpoints created by the trajectory generator do not violate the specified move constraints.

Spline Engine

The Spline Engine function uses a cubic spline algorithm and four setpoints to calculate interpolated positions between two positions from the trajectory generator. You can configure the Spline Engine function to return either setpoint information or return step data in addition to setpoints.

Using the Spline Engine function results in smoother motion and allows you to run the trajectory generator loop slower than the control loop.

Stepper Generator

The Stepper Generator function takes step data from the Spline Engine function and returns step and direction signals to use with stepper motors. Using the Stepper Generator function provides smoother step pulses.

Control Loop

The control loop creates the command signal based on the setpoint provided by the trajectory generator. In most cases, the control loop includes both a position and a velocity loop, but in some cases the control loop may include only a position loop.

The position is typically read from encoders, but also may be read from an analog device using an analog input. The velocity is calculated from the position values that are read. The velocity also may be read directly from a velocity sensor, such as a tachometer.

When controlling stepper motors, the control loop converts the setpoint generated by the trajectory generator into stepper signals (step/direction).

For control loop implementation, NI SoftMotion provides LabVIEW source code for an enhanced PID implementation of the control loop. The source code is installed to `<LabVIEW>\examples\motion\NI SoftMotion\ControlLoop`.

I/O

Analog and digital I/O are required to send the command signals to the drives and receive feedback from the motors. Some servo motor drives use analog I/O to receive feedback. However, most I/O requirements for motion controllers are digital. For example, feedback from incremental encoders requires digital I/O.

NI SoftMotion Components

NI SoftMotion provides properties and methods that provide low-level access to NI SoftMotion axis data, including configuration information and reading and writing data and status information.

NI SoftMotion also provides RT and FPGA synchronization VIs to compensate for drift and jitter between the RT host loop and FPGA target loop. The following figure shows how the different components of NI SoftMotion interact.

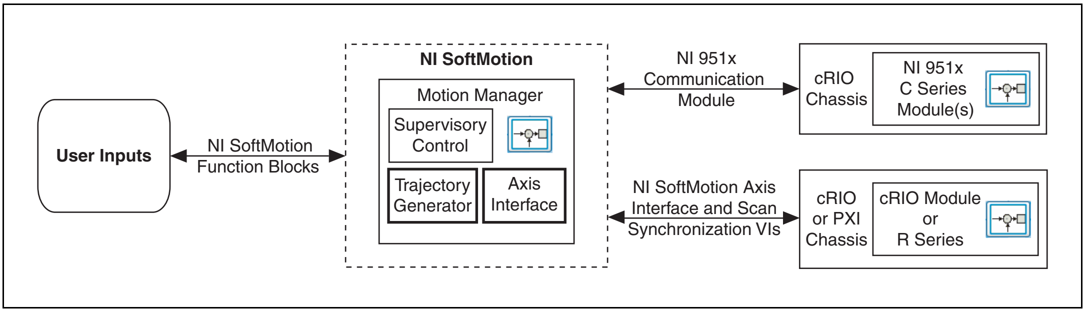


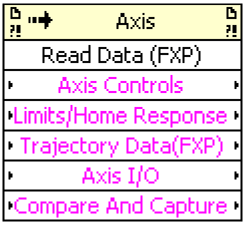
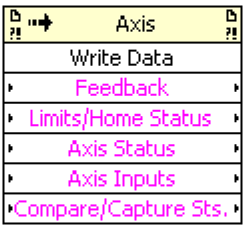
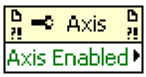
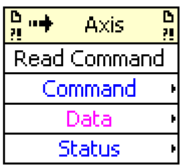
Figure 2. NI SoftMotion Components

User inputs from the NI SoftMotion function blocks and axis configuration information from the deployed LabVIEW Project are accessed using the Axis Interface nodes. The Axis Interface nodes are typically implemented in a separate VI that runs under LabVIEW RT. Execute this VI before running the VI containing motion profile code created with NI SoftMotion function blocks, and keep it running in the background while your motion application executes.

Axis Interface Property and Method Nodes

The Axis Interface Property and Method Nodes are used to communicate directly with NI SoftMotion and to obtain configuration information from the deployed LabVIEW Project. They are used in the RT host VI. Refer to `labview\examples\motion\SoftMotion\AxisInterface` for additional examples using the Axis Interface property and method nodes.

Note The description for each node references figures later in the document that show how they are used in the RT VI.

	The Read Execution Data method, used in the Read Data subVI in the Axis Interface Loop in Figure 8, reads execution information for axes from NI SoftMotion, including I/O and trajectory information.
	The Write Execution Data Method, used in the Write Data subVI in the Axis Interface Loop in Figure 8, writes execution information returned by the hardware to NI SoftMotion for processing and status monitoring.
	The Axis Interface Configure Property Node, used in the Get PID Gains subVI in the Initialization section of Figure 8, obtains the settings configured using the **Axis Configuration** dialog box for use in the Axis Interface RT VI.
	The Axis Interface Read Command Method Node, used in the Command loop in Figure 8, obtains command information from NI SoftMotion including information for fault handling and control loop tuning.
	The Axis Interface Acknowledge Command Method Node, used in the Command loop in Figure 8, sends execution and status information back to NI SoftMotion.

Scan Synchronization VIs

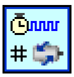
The NI SoftMotion Scan Synchronization functions provide precise, jitter-free synchronization between the RT host and FPGA target. Refer to `labview\examples\motion\SoftMotion\AxisInterface` for additional examples using the Scan Synchronization VIs in both the RT VI and FPGA VI.

Note The description for each VI references figures later in the document that show how the VIs are used in the RT and FPGA VIs.

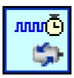
The NI SoftMotion Scan - Initialize Timer VI, shown in Figure 9, initializes the loop timer used to synchronize the RT host loop and the FPGA target loop. This information is then used with the NI SoftMotion Scan - Loop Timer VI. Because each axis can have a different control loop rate, you use a different NI SoftMotion Scan - Initialize Timer VI for each axis in your application. This calculates axis-specific inputs to use in the NI SoftMotion Scan - Loop Timer VI in the FPGA VI.

The NI SoftMotion Scan - Begin Sync Transaction VI, shown in Figure 11, starts the synchronization between the RT host loop and the FPGA target loop each loop period.

The NI SoftMotion Scan - End Sync Transaction VI, shown in Figure 11, stops the synchronization between the RT host loop and the FPGA target loop each loop period.



The NI SoftMotion Scan - Update Scan Control VI, shown in Figure 11, updates the scan control data that is used by the NI SoftMotion Scan - Generate Sync Scan VI.



The NI SoftMotion Scan - Generate Sync Scan VI, shown in Figure 4, generates the FPGA clock side of the synchronization with the RT clock.

The NI SoftMotion Scan - Loop Timer VI, shown in Figure 5, regulates the loop that synchronizes the FPGA clock with the RT clock. Because each axis can have a different control loop rate, use a different NI SoftMotion Scan - Loop Timer VI for each axis in your application using the output from NI SoftMotion Scan - Initialize Timer VI for that axis on the RT target.

Note For proper synchronization of the RT host and FPGA target, National Instruments strongly recommends that you use the Scan Synchronization VIs exactly as implemented in these examples.

Axis Interface Example Program

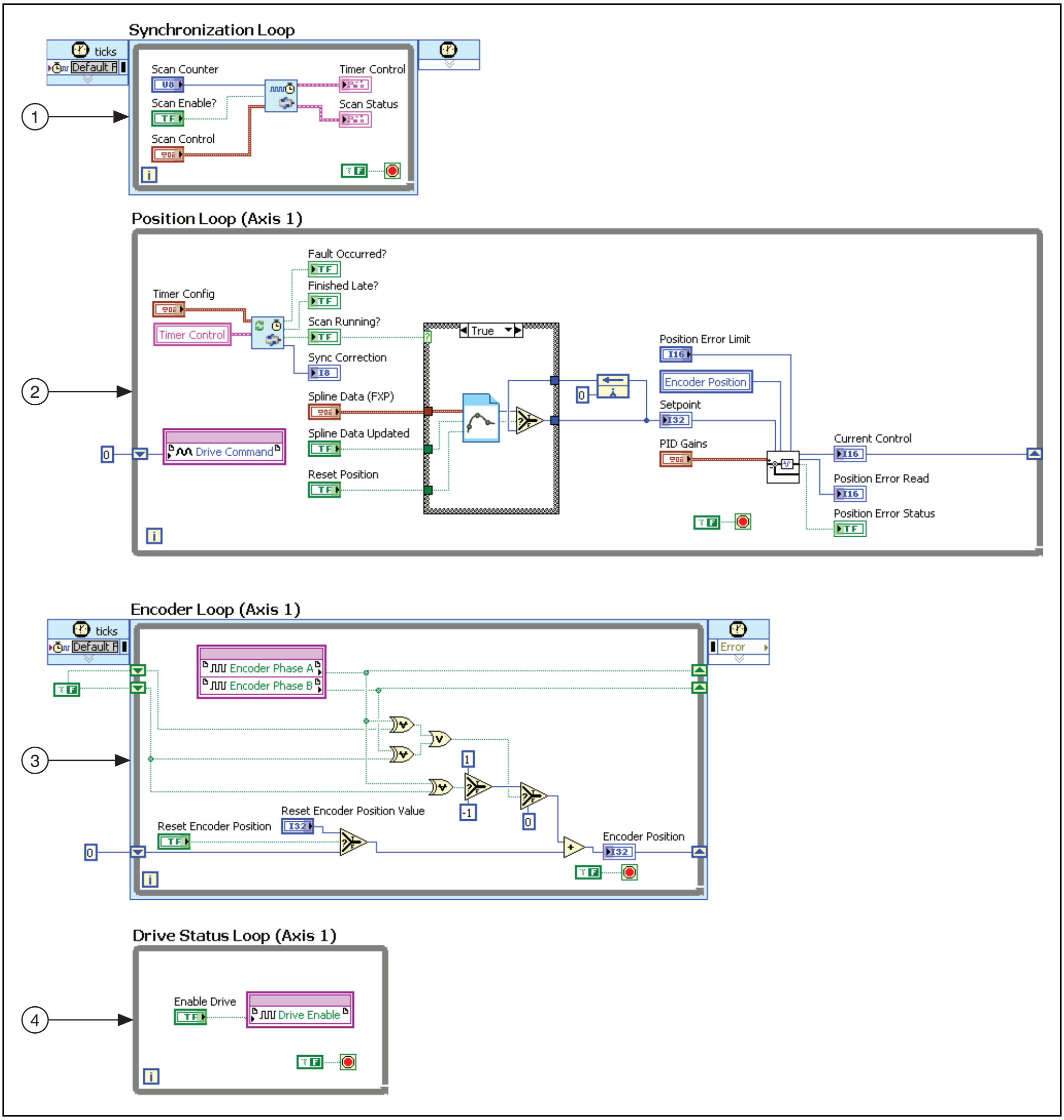
Overview of the Application in this Tutorial

This tutorial describes the Getting Started 2 Axis Servo example installed at `labview\examples\motion\SoftMotion\AxisInterface\GS 2 Axis Servo (9263 & 9401)` to explain how to use the NI SoftMotion Axis Interface nodes and NI SoftMotion Scan Synchronization VIs to create an interface between NI SoftMotion and your hardware.

You will use two C Series modules to control two servo motors. The NI 9263 AO module is used to provide the Drive Command signals to two servo drives, and the NI 9401 DIO module is used for encoder feedback and to enable the drive. This tutorial will also discuss the NI SoftMotion function block examples referenced in this example.

LabVIEW FPGA Target VI

Figure 3 shows the block diagram of the LabVIEW FPGA target VI. For demonstration purposes, only the Axis 1 loops are shown in the figure. You need a duplicate of each loop labeled **(Axis 1)** for any additional axes in your application.



1	**Synchronization Loop**—Synchronizes the LabVIEW FPGA clock to the LabVIEW RT clock to correct for drift and jitter in the system.
2	**Position Loop**—Contains the Spline Engine and control loop for setpoint generation.
3	**Encoder Loop**—Contains logic required to implement the encoder for position control.
4	**Drive Status Loop**—Controls the Drive Enable signal for the servo drive.

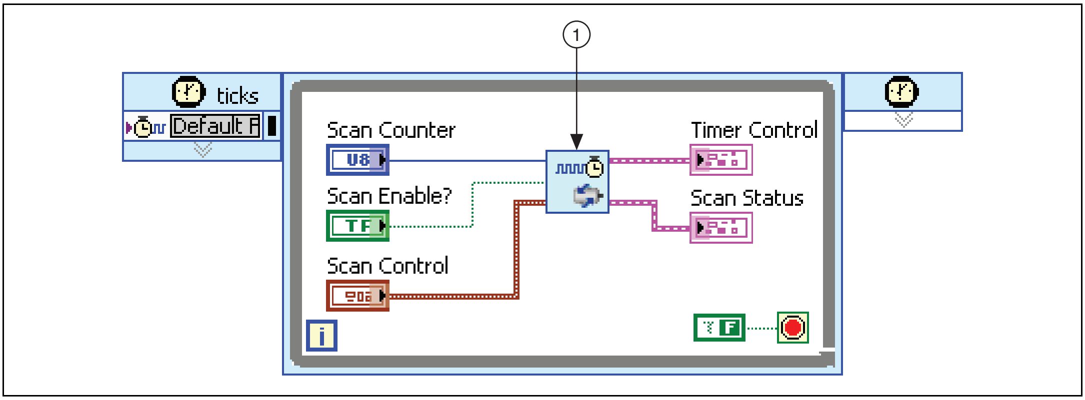
Figure 3. Getting Started 2 Axis Servo LabVIEW FPGA VI Block Diagram

Synchronization Loop

The Synchronization loop, shown in Figure 4, synchronizes the FPGA clock (slave) to the RT clock (master) to correct for drift and jitter between the two clocks. This method is the preferred method of synchronization for NI SoftMotion because the master clock is on the host side, so the application can synchronize to the NI Scan Engine. This allows you to synchronize motion over a distributed network such as EtherCAT.

Tip Refer to *Using the NI Scan Engine* in the *LabVIEW Help* for more information about the NI Scan Engine.

This loop is required to be implemented as shown in this example. Configure the Synchronization loop so that it is synchronized to the NI Scan Engine and so that it is the highest priority loop in the system.



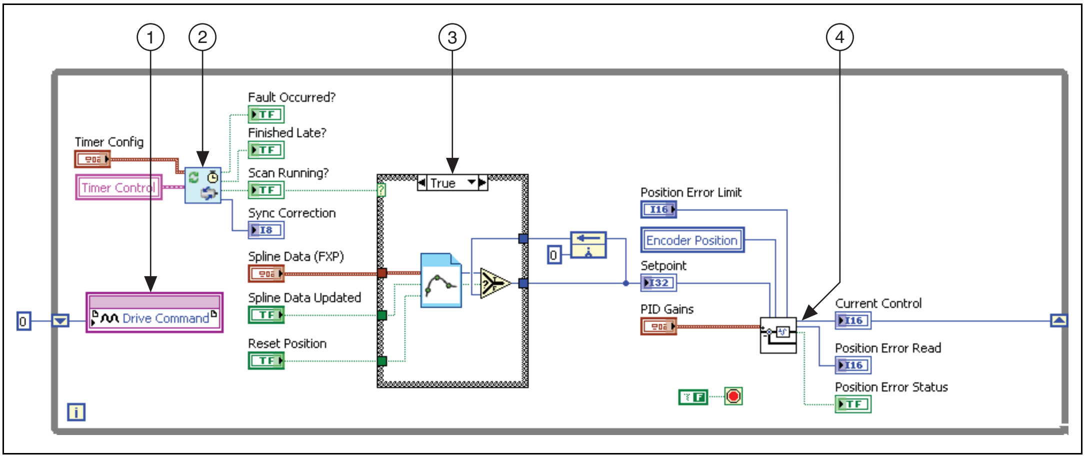
1 NI SoftMotion Scan - Generate Sync Scan

Figure 4. Getting Started 2 Axis Servo LabVIEW FPGA VI Synchronization Loop

Position Loop

The Position loop, shown in Figure 5, executes at the FPGA target loop rate and generates intermediate spline points based on inputs from the trajectory generator. When you are controlling a servo motor the Position loop can be used to perform the control loop because both the Spline Engine function and the control loop execute at the same loop rate. When you are controlling a stepper motor the Position loop also calculates the Step Data used by the Stepper Generator function.

The Position loop for each axis also contains an NI SoftMotion - Scan Loop Timer VI used to synchronize this loop to the RT clock, which is running at the NI Scan Engine scan rate.

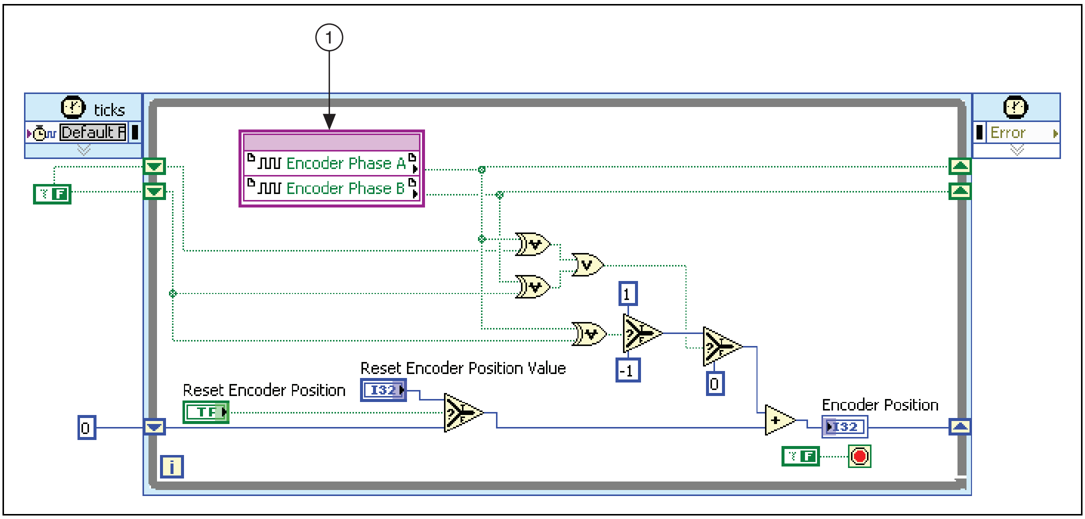


1	Axis 1 Drive Command Output	3	NI SoftMotion Spline Engine Function
2	NI SoftMotion Scan - Loop Timer VI	4	NI SoftMotion Control Loop (Fixed Point PID) Function

Figure 5. Getting Started 2 Axis Servo FPGA VI Position Loop

Encoder Loop

The Encoder loop, shown in Figure 6, enables the digital inputs of the DIO module to read incremental encoder inputs. The loop measures position by incrementing or decrementing a counter depending on the configured encoder line state and current direction. This information is used by the RT VI Axis Interface loop to calculate position error.

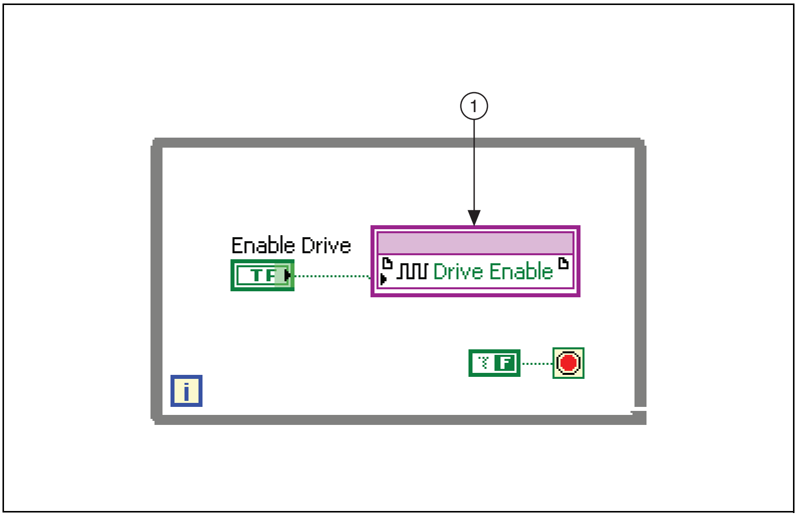


1	Axis 1 Encoder Phase A and Phase B Input State

Figure 6. Getting Started 2 Axis Servo FPGA VI Encoder Loop

Drive Status Loop

The Drive Status loop, shown in Figure 7, controls the Drive Enable output for each drive. This output must be active for the drive to acknowledge commands from the hardware.

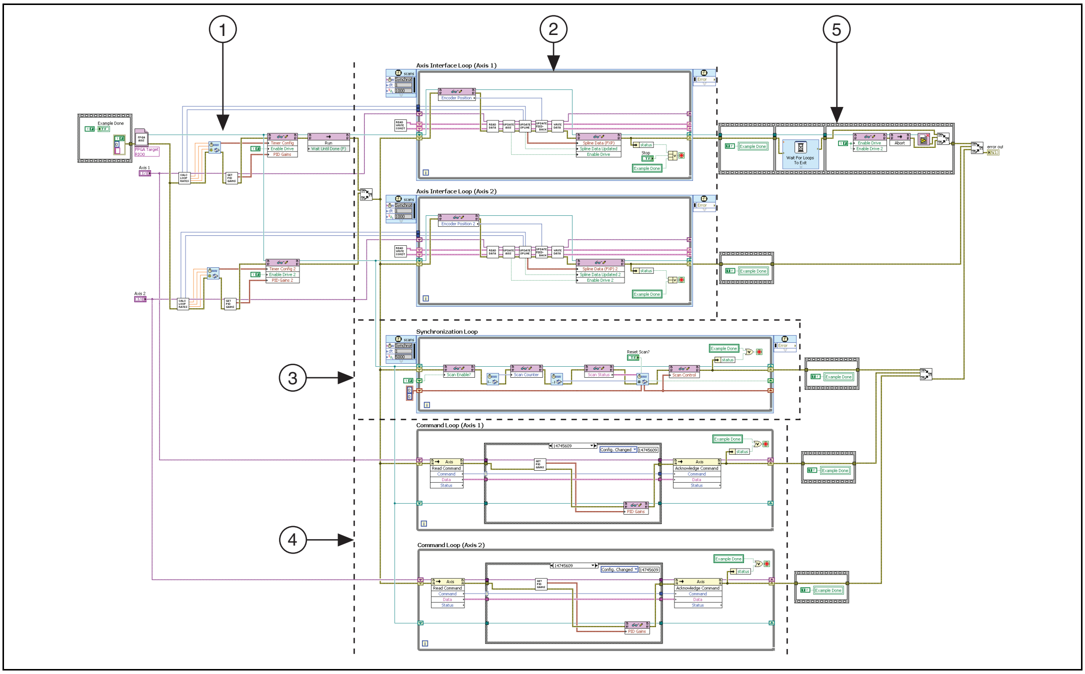


1	Axis 1 Drive Enable Output

Figure 7. Getting Started 2 Axis Servo FPGA VI Drive Status Loop

LabVIEW RT Host VI

Figure 8 shows the block diagram of the LabVIEW RT host VI. The Axis Interface nodes run under LabVIEW RT and communicate execution data to and from NI SoftMotion. The RT host VI exchanges data with the LabVIEW FPGA target VI using the FPGA I/O nodes. The different components of the VI are described in more detail in the following sections.



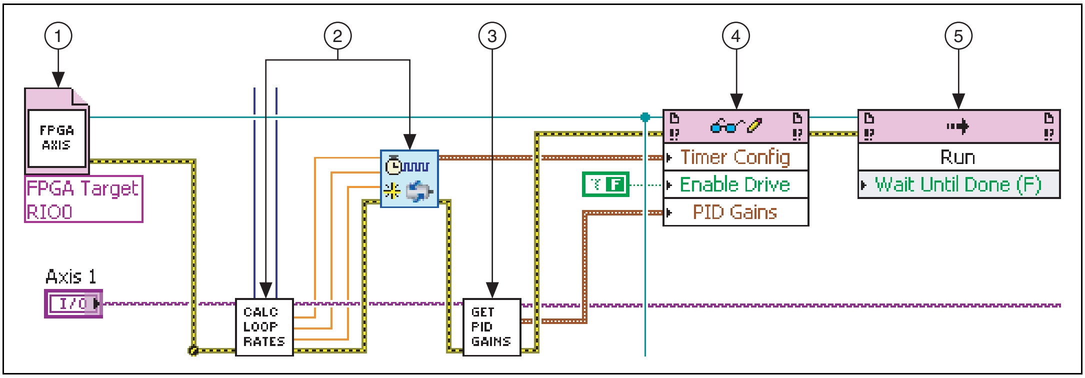
1 **Initialization**—Opens the LabVIEW FPGA reference, configures the NI SoftMotion synchronization loop, and runs the LabVIEW FPGA VI.
2 **Axis Interface Loops**—Implements the axis communication required each host loop period for the axis to work properly.
3 **Synchronization Loop**—Synchronizes the LabVIEW FPGA clock to the LabVIEW RT clock to correct for drift and jitter in the system.
4 **Command Loop**—Processes user commands from NI SoftMotion.
5 **Cleanup**—Stops loops, disables drives, and stops the LabVIEW FPGA VI.

Figure 8. Getting Started 2 Axis Servo LabVIEW RT VI Block Diagram

Initialization Section

The Initialization section, shown in Figure 9, performs the following functions required to start communication with the hardware through LabVIEW FPGA:

1. Opens a reference to the LabVIEW FPGA VI.
2. Gets information required to calculate scan synchronization information for the FPGA VI.
3. Gets the PID parameters configured on the **Control Loop** page of the **Axis Configuration** dialog box using the Axis Interface Configure property node.
4. Sends the output from each NI SoftMotion Scan - Initialize Timer VI to the NI SoftMotion Scan - Loop Timer VI in the FPGA VI Position loop for each axis, sends PID gains to the FPGA VI position loop for each axis, and enables the drive associated with each axis.
5. Starts the FPGA VI after the previous initialization steps are complete.



1 Open FPGA VI Reference
2 NI SoftMotion Scan Configuration
3 Get PID Gains From LabVIEW Project
4 Scan Info, Drive Enable, and PID Gains from RT to FPGA
5 Run FPGA VI

Figure 9. Getting Started 2 Axis Servo RT VI Initialization Section

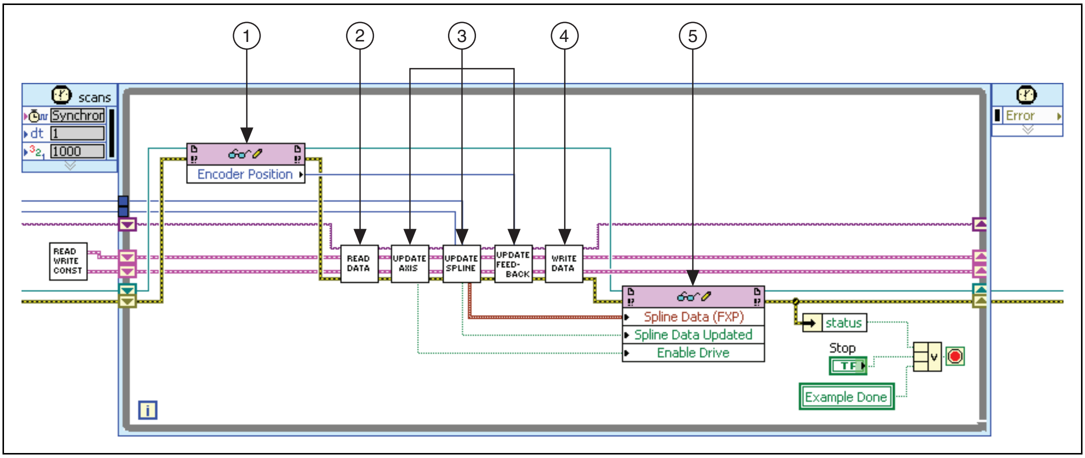
Axis Interface Loop

The Axis Interface loop, shown in Figure 10, performs the following operations that implement the axis communication required each host loop period for the axis to work properly:

1. Reads all necessary axis I/O information from the FPGA VI using the FPGA I/O node. This includes information such as feedback position, limit and home input status, position compare and position capture information, drive status, and the state of any general-purpose I/O on the hardware.

   For simplicity, in the provided Axis Interface examples each I/O type is processed in a separate subVI.
2. Gets updated data from NI SoftMotion using the Read Execution Data node contained in the Read Data subVI.
3. Processes the data received from the FPGA I/O node for sending to NI SoftMotion or processes information received from the Read Execution Data node for sending to LabVIEW FPGA. These operations are performed in the Update Axis, Update Spline, and Update Feedback subVIs.
4. Sends updated information to NI SoftMotion using the Write Execution Data node contained in the Write Data subVI.
5. Sends updated information to LabVIEW FPGA using the FPGA I/O node.

The Axis Interface loop must be synchronized to the NI Scan Engine.



1	Encoder Position from FPGA VI
2	Read NI SoftMotion Axis Data from NI SoftMotion
3	Axis Interface node subVIs for Axis Data
4	Write NI SoftMotion Axis Data from Hardware to NI SoftMotion
5	Spline Engine inputs and Drive Enable status from RT to FPGA

Figure 10. Getting Started 2 Axis Servo RT VI Axis Interface Loop

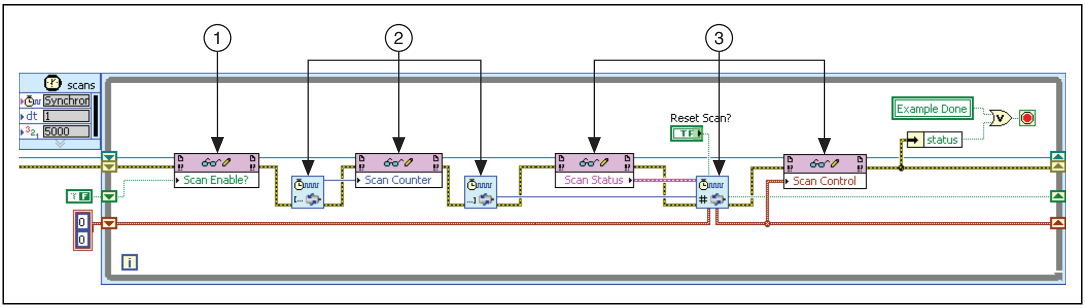
Synchronization Loop

The Synchronization loop, shown in Figure 11, synchronizes the FPGA clock (slave) to the RT clock (master) to correct for drift and jitter between the two clocks by updating the **Scan Control** input used by the NI SoftMotion Scan - Generate Sync Scan VI in the LabVIEW FPGA VI Synchronization loop. This method is the preferred method of synchronization for NI SoftMotion because the master clock is on the host side, so the application can synchronize to the NI Scan Engine. This allows you to synchronize motion over a distributed network such as EtherCAT.

The Synchronization loop performs the following operations:

1. Enables the NI SoftMotion Scan - Generate Sync Scan FPGA VI.
2. Calculates new counter values each loop period and sends the updated scan counter information to the FPGA VI Synchronization loop.
3. Gets scan synchronization information from the FPGA VI and calculates updated scan control information that is sent to the FPGA VI each loop period.

The Synchronization loop is required to be implemented as shown in this example. Configure this loop so that it is synchronized to the NI Scan Engine and so that it is the highest priority loop in the system.



1	Input to NI SoftMotion Scan - Generate Sync Scan FPGA VI	3	Calculate new Scan Control for FPGA
2	Calculate new Counter for FPGA		

Figure 11. Getting Started 2 Axis Servo RT VI Synchronization Loop

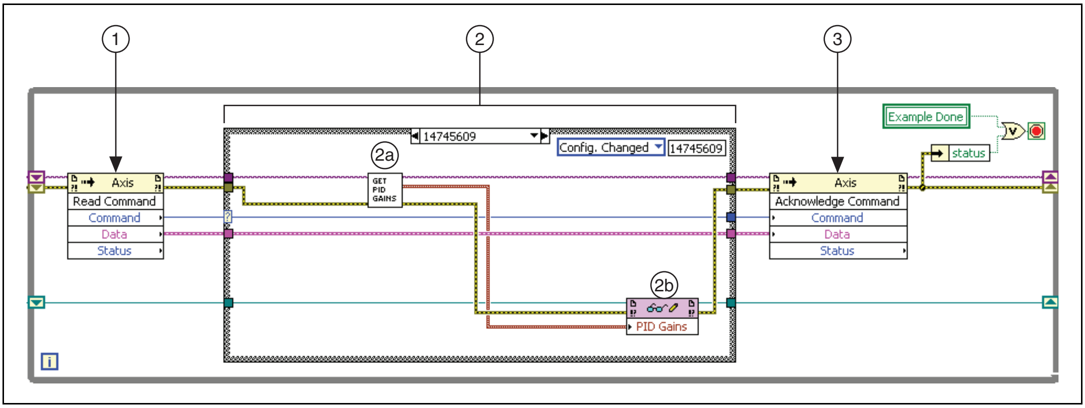
Command Loop

The Command loop, shown in Figure 12, processes user commands from NI SoftMotion, such as fault handling or getting updated PID Gains when tuning the motor using the **Gain Tuning Panel**. Figure 12 only shows the **Config. Changed** command. This command is used to indicate that a control loop configuration parameter has changed. This notification allows the user to perform an action based on the configuration change.

The Command loop performs the following operations:

1. Receives commands from NI SoftMotion through the Read Command node.
2. Performs the appropriate user-defined action depending on the command sent.
3. Returns execution and status information back to NI SoftMotion through the Acknowledge Command node.

The Command loop cannot take longer than 100 ms to complete execution or a timeout error occurs. Configure the loop rate so that the Command methods and any additional code takes less than the maximum time to run.



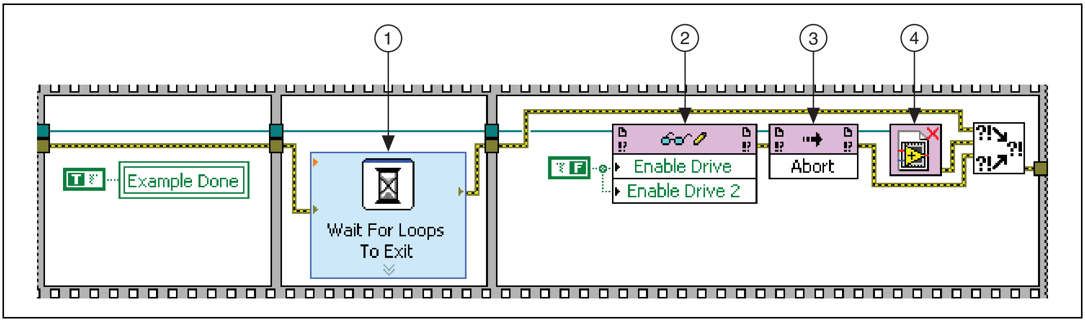
1 Axis Interface Read Command Node
2a Get PID Gains subVI (contains the Axis Interface Configuration Properties)
2b Send updated PID Gains to the FPGA Control Loop VI
3 Axis Interface Acknowledge Command Node

Figure 12. Basic Servo RT VI Command Loop

Cleanup

The cleanup section of the VI performs the following actions to ensure that all processes are removed from memory before exiting:

1. Stops all loops.
2. Disables each drive.
3. Stops the FPGA VI
4. Closes the FPGA VI.



1	Time Delay to Allow Loops to Finish Execution	3	Abort the FPGA VI
2	Disable Both Drives	4	Close the FPGA VI Reference

Figure 13. Getting Started 2 Axis Servo RT VI Cleanup Section

Setting up the Hardware

Setting up the Controller and Modules

Complete the following steps to set up the hardware for the application in this tutorial.

1. Install the controller on the chassis if you are not using an integrated controller and chassis. Refer to the controller operating instructions for information about installing the controller.
2. Install the NI 9263 module in slot 1 of the chassis.
3. Install the NI 9401 module in slot 2 of the chassis.
4. Connect the controller to a power supply and an Ethernet network on the same subnet as the development computer. Refer to the controller operating instructions for information about wiring the controller to the power supply and Ethernet network.
5. Connect the NI 9263 module to the servo drive command input and the NI 9401 module to an encoder and the Drive Enable input on the servo drive.

Installing Software to and Configuring the Controller

Complete the following steps to configure the controller and install software on it.

1. Launch Measurement & Automation Explorer (MAX) on the development computer.
2. Select the controller under **Remote Systems** in the **Configuration** pane. If you do not see the controller, you may need to disable the firewall on the development computer.
3. Verify that the **Serial Number** in the **Identification** section matches the serial number on the device.
4. If you do not want to format the disk on the controller, eliminating all installed software and files, power on the controller and skip to step 9.
5. Set the **Safe Mode** switch on the controller to the **On** position.
6. Power on the controller. If it is already powered on, press the **Reset** button on the controller to reboot it.
7. Right-click the controller under **Remote Systems** in the **Configuration** pane and select **Format Disk**. Click **Yes** on the dialog box that appears.
8. When MAX finishes formatting the disk, set the **Safe Mode** switch to the **Off** position and press the **Reset** button on the controller to reboot it.
9. Select the **Obtain an IP address automatically** radio button to assign an IP address or select the **Use the following IP address** radio button to specify a static IP address in the **IP Settings** section.
10. Type a descriptive name for the system in the **Name** field.
11. Click **Apply** above the **Network Settings** tab and let MAX reboot the system.
12. When the new system name appears under **Remote Systems**, expand the controller item in the tree, right-click **Software**, and select **Add/Remove Software**.
13. Select a recommended software set that includes NI-RIO 3.2.0 or later with NI Scan Engine Support and the following add-ons enabled:
    - LabVIEW NI SoftMotion Module
    - NI Scan Engine Support for LabVIEW NI SoftMotion Module
    - NI-Motion driver software
14. Click **Next** to install the selected software on the controller. Click **Help** if you need information about installing recommended software sets.
15. After MAX finishes installing software on the controller, close MAX.

Working with the Application

The Getting Started 2 Axis Servo example provides the RT and FPGA VIs required to implement the Axis Interface and Scan Synchronization VIs with the specified modules. Complete the following steps to use the examples with your hardware:

1. Launch LabVIEW.
2. Click the **Empty Project** link in the **Getting Started** window to display the **Project Explorer** window. You can also select **File»New Project** to display the **Project Explorer** window.
3. Select **Help** and make sure that **Show Context Help** is checked. You can refer to the context help for information about items on the block diagram.
4. Right-click the top-level project item in the **Project Explorer** window and select **New»Targets and Devices** from the shortcut menu to display the **Add Targets and Devices** dialog box.
5. Make sure that the **Existing target or device** radio button is selected.
6. Expand **Real-Time CompactRIO**.
7. Select the CompactRIO controller to add to the project and click **OK**.
8. If you have LabVIEW FPGA installed, the **Select Programming Mode** dialog box appears. Select **FPGA Interface** to put the system into FPGA Interface mode.
9. Click **Continue**. LabVIEW adds the controller, the chassis, and all the modules to the project.
10. When LabVIEW finishes discovering hardware, select **File»Save Project** and save the project.
11. Select the FPGA VI and all other items under the original target in the example and drag the items to the new target.

Note The new target must support all items. Refer to *Reusing FPGA VIs and FPGA Items among Multiple FPGA Targets* in the *LabVIEW Help* for more information about reusing items.

12. Recompile the FPGA VI.
13. Update the FPGA VI references, if necessary.
    a. Right-click the Open FPGA VI Reference function.
    b. Select **Configure Open FPGA VI Reference** from the shortcut menu.
    c. Navigate to the FPGA VI in the new target.

Using the VI with NI SoftMotion Function Blocks

The Getting Started 2 Axis Servo example contains links to several NI SoftMotion function block examples you can use with your CompactRIO hardware and the Axis Interface VIs.

To use these examples first run the RT VI then execute the function block examples to move the axis. You can also use the **Interactive Test Panel** to test and debug your axis settings or the **Gain Tuning Panel** to interactively tune your control loop settings and view step response plots for each axis. You can access the Interactive Test Panel or the Gain Tuning Panel by right-clicking the axis in the **Project Explorer** window and selecting the appropriate option from the shortcut menu.

The following examples are available:

- **Status and Data Monitor**—This example provides a simple status monitor for a single axis using different methods of the NI SoftMotion Read function block, showing a position plot, and various axis status information.
- **Blending Coordinate Arcs and Straight Line Moves**—This example shows how to use blending to sequence lines and arcs without stopping motion using the NI SoftMotion Line and Arc function blocks.
- **Circular Arc Move**—This example shows how to do a single circular arc move using the NI SoftMotion Arc function block.
- **Coordinate Straight Line Move**—This example shows how to move axes in a coordinate space in a straight line using the NI SoftMotion Line function block. The axes all start and end their movement at the same time.
- **Multi-Axis Simultaneous Start**—This example shows how to start multiple axes at the same time. Unlike a coordinate move, each axis move profile is independent and the axes all stop at a different time using NI SoftMotion function blocks.

Note Refer to the *NI SoftMotion Module* book of the *LabVIEW Help* for more information about NI SoftMotion function blocks. To access this help file from LabVIEW, select **Help»Search the LabVIEW Help**, then expand the *LabVIEW NI SoftMotion Module* book on the **Contents** tab.

What You Have Learned

This tutorial taught the following concepts about using the NI SoftMotion Axis Interface nodes to create a custom hardware interface between NI SoftMotion and your hardware.

- The NI SoftMotion Axis Interface nodes allow you to directly communicate with NI SoftMotion and access deployed LabVIEW Project configuration data.
- The NI SoftMotion Scan Synchronization VIs synchronize the LabVIEW FPGA clock to the LabVIEW RT clock and allow your motion system to synchronize over a distributed deterministic network.
- You must execute the Axis Interface RT VI before performing any operations using NI SoftMotion function blocks.

Figure 14 shows how the different components of NI SoftMotion interact in detail.

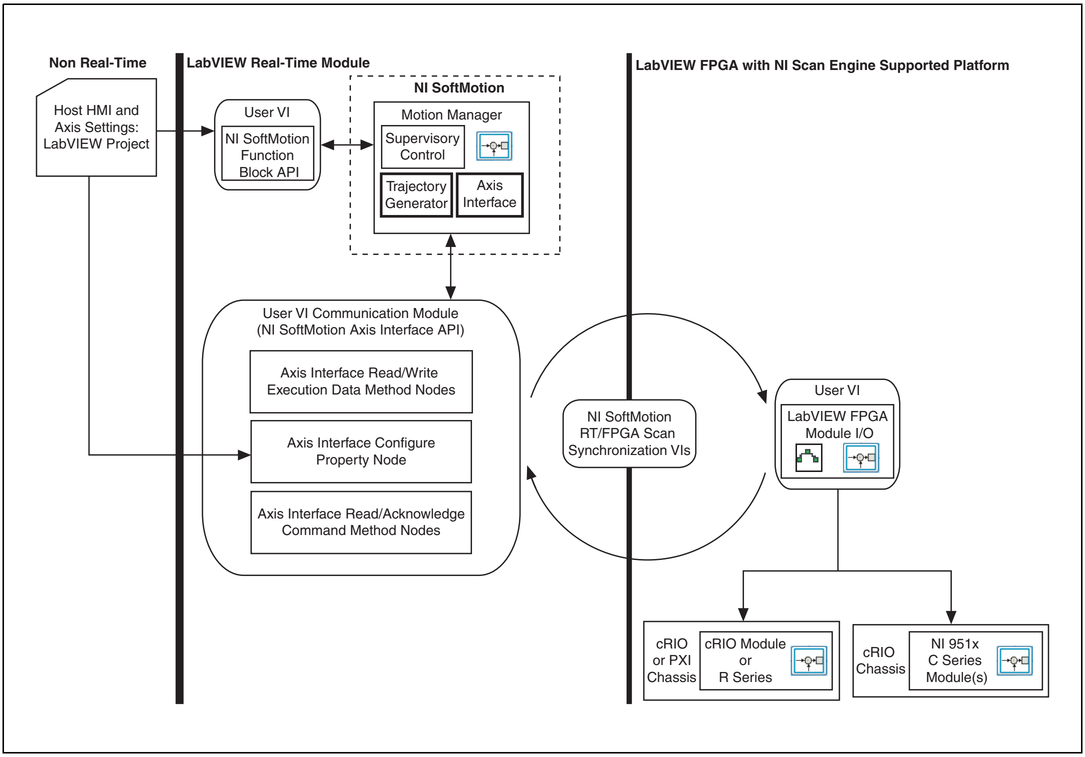


Figure 14. NI SoftMotion Axis Interface

Where to Go for Support

The National Instruments Web site is your complete resource for technical support. At ni.com/support you have access to everything from troubleshooting and application development self-help resources to email and phone assistance from NI Application Engineers.

National Instruments corporate headquarters is located at 11500 North Mopac Expressway, Austin, Texas, 78759-3504. National Instruments also has offices located around the world to help address your support needs. For telephone support in the United States, create your service request at ni.com/support and follow the calling instructions or dial 512 795 8248. For telephone support outside the United States, contact your local branch office:

Australia 1800 300 800, Austria 43 662 457990-0, Belgium 32 (0) 2 757 0020, Brazil 55 11 3262 3599, Canada 800 433 3488, China 86 21 5050 9800, Czech Republic 420 224 235 774, Denmark 45 45 76 26 00, Finland 358 (0) 9 725 72511, France 01 57 66 24 24, Germany 49 89 7413130, India 91 80 41190000, Israel 972 3 6393737, Italy 39 02 41309277, Japan 0120-527196, Korea 82 02 3451 3400, Lebanon 961 (0) 1 33 28 28, Malaysia 1800 887710, Mexico 01 800 010 0793, Netherlands 31 (0) 348 433 466, New Zealand 0800 553 322, Norway 47 (0) 66 90 76 60, Poland 48 22 328 90 10, Portugal 351 210 311 210, Russia 7 495 783 6851, Singapore 1800 226 5886, Slovenia 386 3 425 42 00, South Africa 27 0 11 805 8197, Spain 34 91 640 0085, Sweden 46 (0) 8 587 895 00, Switzerland 41 56 2005151, Taiwan 886 02 2377 2222, Thailand 662 278 6777, Turkey 90 212 279 3031, United Kingdom 44 (0) 1635 523545

372597A-01 Jul09