Document Type: Tutorial
NI Supported: Yes
Publish Date: Dec 6, 2011

Building a Real-Time System with NI Hardware and Software

Overview

National Instruments real-time hardware and software work together seamlessly to run your applications reliably and deterministically with precise timing. This paper explains the different components needed to create your NI real-time system, identifies the hardware and software options available, and outlines how to choose the best options for your project.

Table of Contents

Background

Before reading this paper, it is useful to have a basic understanding of what a real-time system is, and how building a real-time system can benefit your projects. For a review of this information, read the paper: Do I Need a Real-Time System?

This guide is meant to clearly explain the various NI hardware and software components that you need to build a real-time system, and provide some help in choosing the right equipment for your project. After reading this paper, it is recommended that you contact an NI representative by **calling 866-337-5042** to talk about your project requirements in more detail.

Introduction: Three Steps for Building an NI Real-Time System

To build an NI real-time system, you need to choose two major components: your development software and hardware platform. Detailed information on the options for each of these components is provided in the following sections.

Note that when building any real-time system in general, you typically must select a real-time capable operating system. National Instruments real-time development tools come packaged with all necessary real-time OS software, meaning that you do not need to choose this component separately. Depending on the hardware platform that you download and run your real-time programs on, National Instruments automatically uses real-time operating system software based on the Wind River VxWorks or Phar Lap ETS operating systems.

Step 1: Choose an NI Development Environment and Optional Debugging Tools

To develop real-time programs, you can use either LabVIEW graphical programming (recommended) or the LabWindows/CVI ANSI C environment.

Development Option: LabVIEW Real-Time Graphical Programming

To program NI supported real-time hardware using graphical programming, you must purchase LabVIEW along with the LabVIEW Real-Time Module. The LabVIEW Real-Time Module allows you to create reliable, hard real-time applications, download them to your hardware, and then debug and run them using just one tool.

You can use most of the built-in math and signal processing algorithms that come with LabVIEW in your real-time applications, and popular PID functions are also included. In addition, you can run textual math scripts on your real-time system with the optional LabVIEW MathScript RT Module, or integrate models that you create in The MathWorks, Inc. Simulink ® software using the optional LabVIEW Simulation Interface Toolkit.

LabVIEW Real-Time: Developing, Downloading, and Running Applications

When developing real-time applications in LabVIEW, you use the LabVIEW Project Explorer to organize your programs (VIs) and categorize them by the hardware platform that they will run on. Note that you develop your code on a general-purpose Windows computer, and then connect to your real-time hardware using Ethernet.

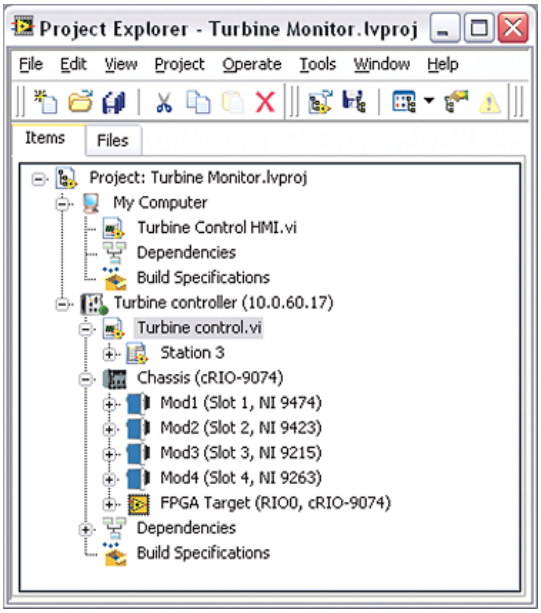

Figure 1. Manage LabVIEW programs (VIs) and assign them to real-time hardware using the LabVIEW Project Explorer.

Developing real-time programs in LabVIEW is nearly identical to developing standard LabVIEW applications for your PC. Several additional functions specific to real-time systems are packaged in the **Real-Time VIs** palette, including:

- Watchdog functions to automatically restart some hardware targets if your program stops running
- Functions to communicate data deterministically between parts of a real-time program
- Utilities to configure load balancing on systems with multiple CPU cores
- Timing functions to precisely control the execution of loops in your real-time programs

Figure 2. When developing real-time (RT) programs in LabVIEW, you develop on a host Windows computer, and then download and run them on a real-time hardware target.

To test your LabVIEW Real-Time program on your hardware platform, simply click on the run arrow and your application will automatically be transferred to your real-time hardware via Ethernet and begin running. You can use standard NI debugging tools such as highlight execution, single stepping, and breakpoints from your development system even though your real-time program is actually running on a separate system.

When you have finalized your real-time program, you can build an EXE in LabVIEW and download it to your real-time hardware as a startup application. After rebooting your real-time hardware, your program will automatically run in a reliable, stand-alone fashion.

LabVIEW Real-Time: Assigning Priorities to Parallel Code Sections

The dataflow programming model of LabVIEW frees you from the sequential architecture of text-based programming languages. Because the execution order is determined by the flow of data between nodes and not by sequential lines of text, you can easily create applications that execute multiple operations in parallel. Additionally, LabVIEW makes it easy to assign thread priorities with the Timed Loop structure. As shown below, each loop can contain a separate task with configurable timing source, period, priority and more.

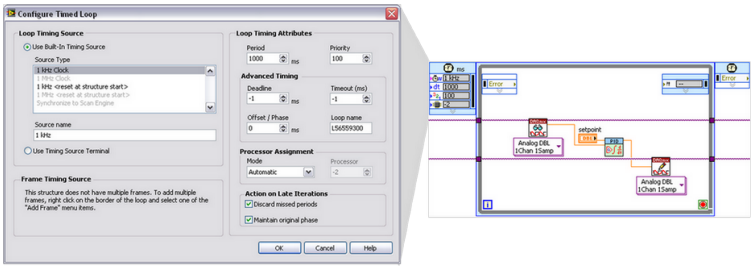

Figure 3. You can specify the priority of different code sections using the LabVIEW Timed Loop structure.

LabVIEW Real-Time: Multicore Processing

The LabVIEW Real-Time Module supports multicore processing and automatically maps parallel sections of your code to individual operating system threads - eliminating the need for you to manually create and manage them. By default, these threads are also automatically balanced across the CPUs available on your real-time hardware.

To further increase the performance and reliability of a real-time system, you can choose to manually assign Timed Loops to specific processor cores if desired. For example, you can dedicate one core of a processor to execute one time-critical loop and isolate it from less important tasks that run on different cores.

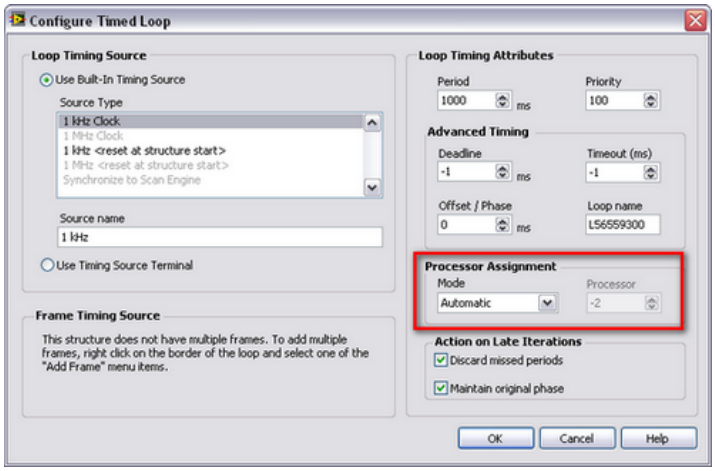

Figure 4. You can allow LabVIEW to automatically run your real-time program across multiple CPU cores, or manually dedicate certain cores for individual tasks.

Development Option: LabWindows/CVI Real-Time Module

If your organization standardizes on ANSI C, you can use the NI LabWindows/CVI development environment to create text-based measurement and control applications that take advantage of NI signal processing algorithms as well as a wide variety of I/O modules. By adding the LabWindows/CVI Real-Time Module, you can develop, download, and run these applications deterministically on certain NI supported real-time hardware targets.

Note that the LabWindows/CVI Real-Time Module supports only the real-time PXI, Industrial Controller, and PC hardware discussed below in this paper. For access to a wider variety of NI real-time hardware targets, consider using the LabVIEW Real-Time Module to develop your applications.

Additional Software Option: Debugging with the Real-Time Execution Trace Toolkit

For advanced multicore debugging, you can use the Real-Time Execution Trace Toolkit to verify the performance of your real-time programs without halting or pausing execution of the code. With minimal modifications to your real-time code, you can log application performance to a file and send it to a host computer for viewing and analysis. The trace tool viewer graphically displays multithreaded code execution while highlighting thread swaps, mutexes, and memory allocations. You can use the Real-Time Execution Trace Toolkit to optimize application performance by identifying undesirable execution characteristics and difficult-to-find race conditions.

Figure 5. The Real-Time Execution Trace Toolkit enables you to debug multithreaded real-time LabVIEW or LabWindows/CVI applications at a low level.

Step 2: Select an NI Hardware Platform and I/O Modules

All NI real-time hardware platforms are based on a common architecture, which means that programs that you write with the LabVIEW Real-Time Module will work across different pieces of NI supported hardware with only minor modifications. Specifically, each hardware platform features off-the-shelf computing components including a microprocessor, RAM, nonvolatile storage, and I/O bus interface. Some hardware platforms even feature an onboard Field Programmable Gate Array (FPGA) that you can program using the LabVIEW FPGA Module.

Each NI real-time platform has distinct characteristics geared toward a particular subset of use cases. Whether you are looking for cutting-edge multicore performance or a compact, rugged form factor, NI real-time hardware can meet your project needs. All NI real-time hardware platforms have the ability to function as autonomous systems.

Hardware Option: PXI (PCI Extensions for Instrumentation)

The industry standard PXI platform consists of a rugged chassis with integrated timing and triggering lines, embedded controller, and plug-in I/O modules. Serial, USB, Ethernet, and GPIB ports are also built in to the controller. PXI real-time hardware can be programmed using either the LabVIEW Real-Time Module or LabWindows/CVI Real-Time Module.

Do you have an existing PXI controller running Windows that you would like to change to a real-time controller? You can purchase a LabVIEW Real-Time Deployment License to convert your controller, or even set up a dual-boot system.

You can assemble your own PXI real-time system including controller, chassis, I/O modules, and software using the online PXI Advisor.

Figure 6. PXI hardware provides a rugged, high-performance option for your real-time projects.

NI PXI hardware is often used for high-performance real-time systems such as hardware-in-the-loop test of electronic control units and vibration analysis for machine-condition monitoring applications. When using a real-time PXI system, your applications have access to advanced timing and synchronization hardware features that simplify precise I/O triggering and multi-module synchronization.

Learn how Lockheed Martin used LabVIEW Real-Time and PXI hardware to reduce cost and time testing the F-35 Joint Strike Fighter aircraft.

Hardware Option: Industrial Controller

NI Industrial Controllers feature a rugged, fanless chassis, high-performance processor, and a single PCI or PCIe slot for I/O modules. Serial, USB, CompactFlash, and Ethernet connections are built into the controller. NI real-time Industrial Controllers can be programmed using either LabVIEW Real-Time or the LabWindows/CVI Real-Time Module.

You can assemble your own Industrial Controller system including controller, chassis, and software using the online advisor.

Figure 7. NI Industrial Controller hardware incorporates a high-performance processor in a rugged, fanless form factor.

NI Industrial Controllers are ideal for running reliable real-time applications in harsh environments that prohibit the use of active cooling fans. To connect with a wide variety of sensors and actuators deterministically, Industrial Controller hardware supports EtherCAT and MXIe connections to expansion chassis.

Hardware Option: CompactRIO (Compact Reconfigurable I/O)

NI CompactRIO combines a real-time processor, a Field-Programmable Gate Array (FPGA), and I/O modules. In addition, serial, USB, and Ethernet ports are built in to the controller. When using CompactRIO, your I/O modules are connected to the FPGA for fast processing in hardware, and then you exchange data between the FPGA and real-time processor as desired.

You can assemble your own CompactRIO system including controller, chassis, I/O modules, and software using the online CompactRIO Advisor.

Figure 8. CompactRIO provides a flexible, rugged, and portable option for real-time applications.

You can program the real-time processor on CompactRIO hardware targets using the LabVIEW Real-Time Module. You can also choose to write custom code for the onboard FPGA using the LabVIEW FPGA Module, or skip this step and use the NI Scan Engine built into the LabVIEW Real-Time Module to read from your I/O modules.

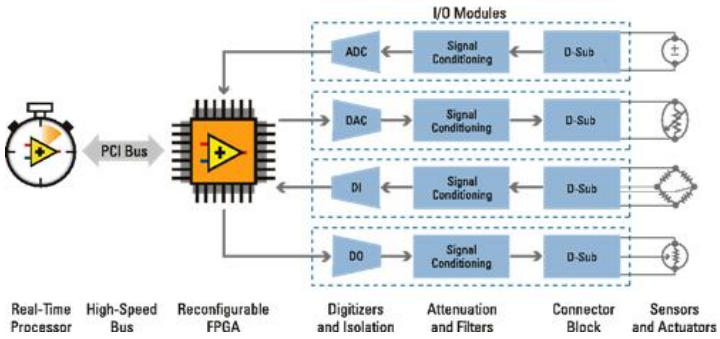

Figure 9. The CompactRIO hardware architecture features a real-time procesor, reconfigurable FPGA, and I/O modules.

CompactRIO hardware is often used for industrial applications including machine condition monitoring, motion control, and data logging. In addition, CompactRIO is ideal for quickly prototyping embedded systems for faster time to market.

Read how Animage LLC used LabVIEW Real-Time and CompactRIO to design a veterinary imaging system incorporating X-ray source control, X-ray detection, and motion control.

Hardware Option: Single-Board RIO (Single-Board Reconfigurable I/O)

NI Single-Board RIO systems are identical in architecture to CompactRIO systems, only in a single circuit board form factor. Single-Board RIO hardware features a real-time processor and programmable FPGA just as with CompactRIO, and several I/O modules are also available in a board-only form factor.

You can evaluate Single-Board RIO hardware and the LabVIEW Real-Time Module with the NI Embedded Software Evaluation Kit.

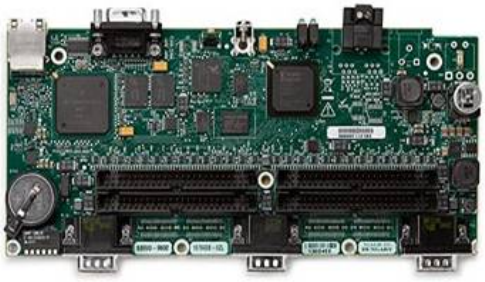

Figure 10. Single-Board RIO hardware features the same architecture as CompactRIO systems, only in a single circuit board form factor.

High-volume applications that require flexibility, reliability, and high performance can benefit from using Single-Board RIO hardware. Use cases include medical device control and monitoring, robotics and unmanned vehicle control, and more.

Explore how Ventura Aerospace used LabVIEW Real-Time and Single-Board RIO to create an FPGA-based fire monitoring and suppression control system.

Hardware Option: Real-Time Vision (Smart Camera, Compact Vision System, and Embedded Vision System)

NI offers a range of vision hardware for real-time applications, including the Smart Camera for all-in-one solutions, the Compact Vision System for processing data from multiple cameras, and the high-performance, fanless Embedded Vision System.

NI Smart Cameras combine industrial, high-quality image sensors and real-time power processors that are programmable with the LabVIEW Real-Time Module. You can also purchase the NI Vision Development Module for quick access to hundreds of image processing and machine vision functions. Smart Cameras also feature two optoisolated digital inputs, two optoisolated digital outputs, one RS 232 serial port, and two gigabit Ethernet ports.

Figure 11. NI Smart Cameras feature an onboard real-time processor that can be programmed using the LabVIEW Real-Time Module.

Compact Vision System (CVS) hardware incorporates an embedded real-time processor, Field Programmable Gate Array (FPGA) and connections for three IEEE 1394 DCAM cameras. Compact Vision System controllers also feature a local video display, an Ethernet port, 15 digital inputs, and 14 digital outputs. You can program the onboard real-time processor using the LabVIEW Real-Time Module, and you can choose to customize the FPGA (which communicates with the digital inputs and outputs) if desired using the LabVIEW FPGA Module.

Figure 12. The NI Compact Vision System (CVS) runs your real-time code and can communicate with up to three IEEE 1394 cameras.

Finally, NI Embedded Vision System hardware provides a rugged, fanless, high-performance option for vision applications. Embedded Vision System controllers feature support for both GigE and IEEE 1394 cameras, and also incorporate 29 DIO lines, serial, USB, and Ethernet connections. You can program the onboard real-time processor using the LabVIEW Real-Time Module, and you can choose to customize the FPGA (which communicates with the digital inputs and outputs) if desired using the LabVIEW FPGA Module.

NI Vision hardware is ideal for automated inspection of parts, reading 1D and 2D barcodes, reading or verifying text, and even taking heat measurements using an infrared camera. Vision systems can provide an accurate, non-destructive way to take measurements in your project.

Read about how Soliton Technologies used LabVIEW Real-Time and the Compact Vision System (CVS) to achieve six-sigma repeatability standards in inspection of automobile spark plugs.

Other Hardware Options

You can convert an industrial PC, single-board computer, or even a standard desktop PC to work with NI Real-Time software, as long as the hardware meets certain system requirements. To do this, you need to purchase a LabVIEW Real-Time Deployment License in addition to your NI real-time development software (LabVIEW Real-Time Module or LabWindows/CVI Real-Time Module).

Figure 13. You can convert other hardware to work with NI real-time software, such as an Industrial PC, a Desktop PC, a Single-Board Computer (SBC), and a PC-104 Processor Board.

Comparing NI Real-Time Hardware Platforms

Each NI real-time hardware platform is designed for a slightly different application. PXI systems provide the highest overall performance, while the Compact Vision System (CVS) delivers the most rugged hardware. The CompactRIO platform provides a combination of these features, and the flexibility of a programmable FPGA. The following sections compare each LabVIEW Real-Time hardware platform on the basis of I/O availability, performance, and physical attributes.

Keep in mind that you are not on your own: NI representatives are glad to help you determine which real-time hardware option is right for your application. To speak with a representative now, you can **call 866-337-5042**.

Hardware Platform Comparison: I/O Availability

NI real-time hardware platforms support a wide variety of I/O modules, including off-the-shelf modules available from NI, custom-built modules, and third-party modules. National Instruments I/O modules use standard driver APIs that you can quickly access in your LabVIEW Real-Time LabWindows/CVI Real-Time programs. Many NI platforms are also expandable to include multiple chassis, computers, or backplanes - each with their own I/O modules.

I/O Availability	PXI	CompactRIO	Vision Systems	Standard or Industrial PCs
○ Good ◖ Better ● Best				
Variety	●	◖	○	●
Standard Driver APIs	◖	◖	◖	◖
Customizability	◖	●	◖	◖
Expandability	●	◖	○	●

Table 1. Input/Output Device Comparison for NI Real-Time Hardware Platforms.

Specifically, PXI and PCI I/O modules available for NI real-time hardware platforms include: data acquisition (analog and digital), dynamic signal acquisition, image acquisition, motion control, FPGA-based Reconfigurable I/O (RIO), CAN, serial, GPIB, Ethernet, chassis expansion via MXI, IEEE 1394 hard drivers and cameras, external USB hard drives, and other third-party hardware.

Figure 14. You can incorporate a wide variety of PXI and PCI I/O modules into your NI real-time system.

In addition, CompactRIO real-time systems can take advantage of any of the following I/O modules: analog input, analog output (voltage and current), CAN, motion, digital input and output, relay output, counter, pulse generation, and other custom modules.

Figure 15. You can select from a variety of C-Series I/O modules when building a real-time CompactRIO system.

Smart Camera I/O includes two optoisolated digital inputs, two optoisolated digital outputs, one RS 232 serial port, and two gigabit Ethernet ports. The Compact Vision System includes connections for three IEEE 1394 DCAM cameras, a local video display, an Ethernet port, 15 digital inputs, and 14 digital outputs.

Hardware Platform Comparison: Performance

The performance of NI real-time hardware platforms can be categorized in terms of deterministic execution, I/O timing, triggering, synchronization, processor speed, and availability of multicore CPUs. Determinism is the most fundamental component of all real-time systems; it defines how consistently a system is able to perform a given operation within a specified amount of time.

Performance	PXI	CompactRIO	Vision Systems	Standard or Industrial PCs
○ Good ◖ Better ● Best				
Deterministic Execution	●	●	●	◖
Timing, Triggering, and Synchronization	●	●	◖	◖
Processor Speed	◖	◖	◖	●
Multicore Processing	●	○	○	●

Table 2. You should carefully consider performance when selecting an NI real-time hardware architecture for your application.

The above table presents a quick overview comparing the performance of NI real-time hardware architectures. For detailed benchmarks and performance data, please see the following references:

Benchmarking Single-Point Performance on National Instruments Real-Time Hardware

Benchmarking a Typical Control Application using LabVIEW Real-Time & NI-DAQmx

Hardware Platform Comparison: Physical Attributes

In addition to the deterministic performance and reliability that NI real-time software provides, many NI hardware platforms are designed to sustain harsh industrial environments as well. For example, the CompactRIO and Compact Vision System platforms are designed with no moving parts, eliminating the most common failures due to shock and vibration.

The table below compares the ruggedness and portability of various NI real-time hardware platforms.

Physical Attributes	PXI	CompactRIO	Vision Systems	Standard or Industrial PCs
○ Good ◖ Better ● Best				
Ruggedness	◖	●	●	Varies
Portability	◖	●	●	◖

Table 3. Many NI real-time hardware platforms are specially designed to sustain harsh industrial environments.

Step 3: Talk With an NI Representative Today

Still have questions? NI representatives are glad to help you choose the right software, hardware, and I/O for your real-time program - no matter what size your system is. You can also request a demonstration from an engineer in your area, and ask about the training opportunities available in order to get your first NI real-time system up and running in as little time as possible.

>> Call 866-337-5042 to talk to an NI representative about your application now