SOFTWARE DEVELOPMENT FOR A COMPACTRIO-BASED WIRE SCANNER CONTROL AND DATA ACQUISITION SYSTEM*

J. Sedillo, J.D. Gilpatrick, D. Martinez, S. Rodriguez, LANL, Los Alamos, NM 87545, USA
M. Gruchalla, URS (EG&G Division), Albuquerque, NM 87107, USA

Abstract

The Beam Diagnostics and Instrumentation Team at the Los Alamos Neutron Science Center is developing a wire scanner data acquisition and control system with a National Instrument's compactRIO at its core. For this application, the compactRIO controller not only requires programming the FPGA and RT computer internal to the compactRIO, but also requires programming a client computer and a touch panel display. This article will summarize the hardware interfaces and describe the software design approach utilized for programming and interfacing the four systems together in order to fulfill the design requirements and promote reliable interoperability.

INTRODUCTION

Software development for a National Instruments (NI) compactRIO-based wire scanner control and data acquisition system presents interesting challenges provided the hardware diversity offered by compactRIO (cRIO) platform. The successful deployment of a LANSCE cRIO-based wire scanner system requires programming several hardware devices with differing capabilities and interfaces. This report aims to describe one design approach with an emphasis on the software organization at the core of the control system.

OVERVIEW

Wire Scanner System Overview

Figure 1 depicts the high-level wire scanner system and its interfaces. At the core of the wire scanner controller is a NI cRIO enclosed in a 4U rack-mountable chassis. This chassis drives the wire scanner actuator with motor control signals and receives resolver positioning data, limit switch data, and wire secondary emission signals from the horizontal and vertically-aligned sense wires. Due to the pulsed nature of the particle beam, sense wire data is synchronized by an external beam gate signal [1]. Commands, wire scanner status, and scan data are passed back and forth over the network via Experimental Physics and Industrial Control System (EPICS) shared variables.

cRIO Overview

NI's cRIO is a small and rugged, industrial computing platform that possesses an FPGA (Field Programmable Gate Array), a PowerPC-based embedded computer with a RTOS (Real-Time Operating System, this will occasionally be referred to as RT for brevity), and hot-swappable I/O modules [2]. The FPGA acts as a programmable interface between the I/O modules and the RT allowing for FPGA-based pre-processing of data between the I/O modules and the RT.

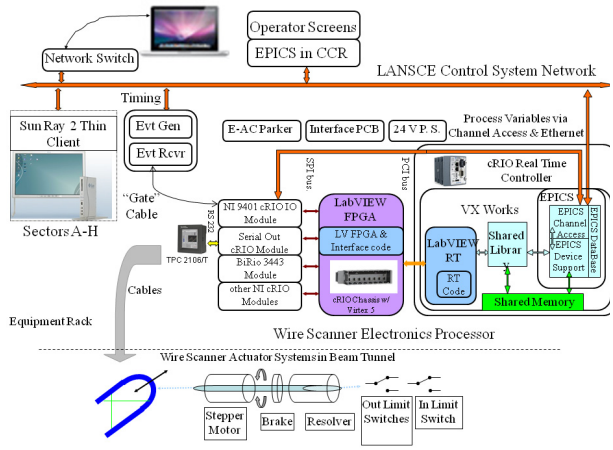

Figure 1: Wire scanner system overview [3].

PROGRAMMING CHALLENGES

Given the unique hardware properties of the FPGA, RT computer, touch-panel, and client; each has a unique set of features and limitations that can be leveraged for the wire scanner control system application.

Distribution of Process Complexity

Figure 2 Summarizes the programming complexity associated with the cRIO hardware and client. Complexity in this sense does not refer to the difficulty associated with programming the hardware, but the relative amount of functionality that has been programmed into each set of hardware.

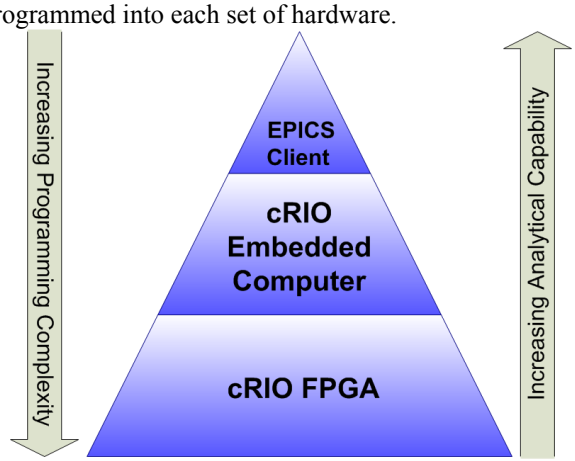

Figure 2: Relative system hardware programming complexity and analytical capability.

The pyramid of Figure 2 details the programming concept used for distributing the functionality among the

* Work supported by the U.S. Department of Energy.

ISBN 978-3-95450-115-1

wire scanner subsystems. The base of the control system consists of FPGA code which performs the core operations of the wire scanner such as motion control and data acquisition. An upward traversal of the pyramid leads to the next step, or embedded RT computer. This component of the wire scanner controller acts as a data accumulation, data post-processing, and client interface system. At the top of the pyramid is the client whose functionality need simply be to write scan parameters to the controller and read system status and data acquired from the controller.

cRIO FPGA Processes

FPGA's are configurable digital logic devices that can be programmed to fulfill a custom digital logic application, a feature which allows for consolidating digital logic circuitry from many discrete logic circuits into as few as one chip. Due to the FPGA's nature of being a digital "fabric," it has another noteworthy feature: the ability to operate many processes with true parallel execution. Moderate computation is possible with the help of fixed-point math and cordic algorithms, but FPGA's are not the best choice for floating-point calculation since, in this case, this task can be transferred to the RT computer. Figure 3 details the FPGA functions for the wire scanner.

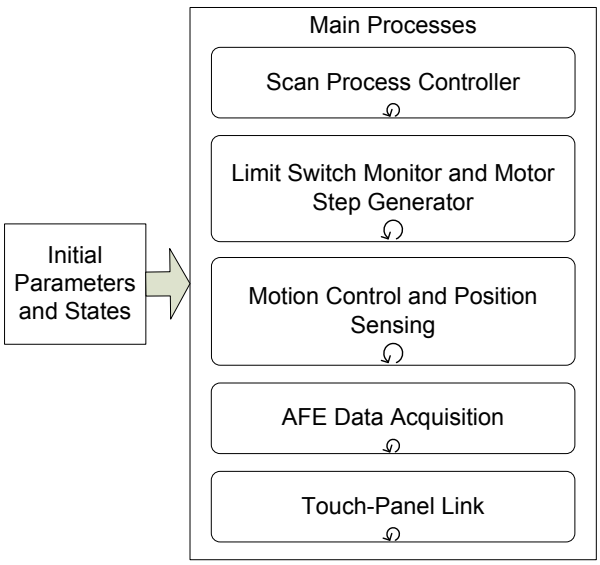

Figure 3: FPGA processes for the wire scanner.

The main FPGA process begins after the I/O modules have been initialized and the default parameters have loaded. The five processes within the main process box shown leverage the FPGA's ability to run them in true parallel fashion. Beginning at the top of the list is the "Scan Process Controller" which is the main software interface between the FPGA and the embedded computer. This process calls upon the processes below it to fulfill the operator's scan request. The "Limit Switch Monitor and Motor Step Generator" process generates step pulses for driving the wire scanner actuator's stepper motor at a variable rate as established by the "Motion Control and Position Sensing" process. Furthermore, this process monitors the wire scanner actuator's limit switches to prevent the actuator from attempting to drive the sensors beyond their operational envelope. Closed-loop motion control, resolver-based position measurement and scan motion recording occur in the "Motion Control and Position Sensor Driver." The "AFE Data Acquisition" process is responsible for simultaneously sampling the beam gate signal and the wire scanner's vertical and horizontal sense wire signals output from the AFE (Analog Front End) and ADC (Analog-to-Digital Converter) modules. Once a 1-ms sample set has been obtained, it is passed to the RT system for storage and additional processing. The last process is the "Touch Panel Link." The Touch panel Link transmits link status, actuator insertion position, limit status, and local control information between the FPGA and the chassis-mounted touch-screen.

cRIO Embedded Computer (RT) Processes

Following the guidance established by Figure 2, the cRIO's embedded computer has less programming complexity and thus has three main roles.

Its first role is to communicate data between the wire scanner system and the operator's client computer. Using the EPICS network variable interface, the operator's computer writes values to variables stored on the RT's memory. These values are continuously read by the RT's program and, if necessary, action is taken by the wire scanner control system to fulfill the request. The RT computer of the wire scanner control system updates its EPICS variables every 100ms, allowing the operator's client computer to obtain frequent updates relating to any wire scanner operation, if desired.

The second role of the embedded computer is to leverage its floating-point processor to translate numerical data flowing between the client computer and the FPGA. In one case, the client transmits operator-desired sensor positioning data in its various forms, depending on the type of scan. This data is converted by the controller's RT computer from millimeters to radians, the FPGA's unit convention for actuator positioning. In the second case, the embedded computer obtains the wire sensor data from the FPGA as a voltage from an ADC I/O module interfacing to the AFE module. In its raw form this data is in a fixed-point format which, upon receipt by the RT system, is immediately converted to floating-point. The RT system then converts the data from a voltage to a current, subtracts the signal's DC offset, converts the current to charge via integration, obtains a charge difference specified by the operator, and displays the array data as a macropulse and; in its accumulated form, a beam profile.

The third role is to obtain macropulse array data in a timely manner. This involves the use of an interrupt service routine (ISR). Since the FPGA is programmed to store one macropulse worth of data at a time, this data must be transferred to the RT system before the next macropulse arrives. Through the use of an ISR, the transference of the macropulse array takes immediate

precedence over all other processes within the RT system, thus allowing the FPGA to reposition the sensors before the arrival of the next macropulse.

Touch-Panel Interface to cRIO

The touch-panel computer system is located at the front of each wire scanner control system chassis and its purpose is to provide local status and control of the wire scanner actuator. Since the cRIO's RT and FPGA can function independently, a direct link to the FPGA for local control of the wire scanner actuator is desirable should either the RT system or the network fail. A direct RS-232 null modem link was the method employed since it was the only interface that the FPGA and touch panel hardware supported. Communication is accomplished by sending 8-bit data back and forth via the link. The FPGA and TP each scan the data strings they receive for certain tokens describing the data payload accompanying the token. Once the data is retrieved, it is either displayed or acted upon by the respective system. Robustness has been incorporated into the design whereby if the touch-panel fails, it simply will not transmit any data at all and thus the FPGA will have no touch-panel commands to react to.

CLIENT INTERFACE

The wire scanner control system's RT communication interface to the client has the organization shown in Figure 4.

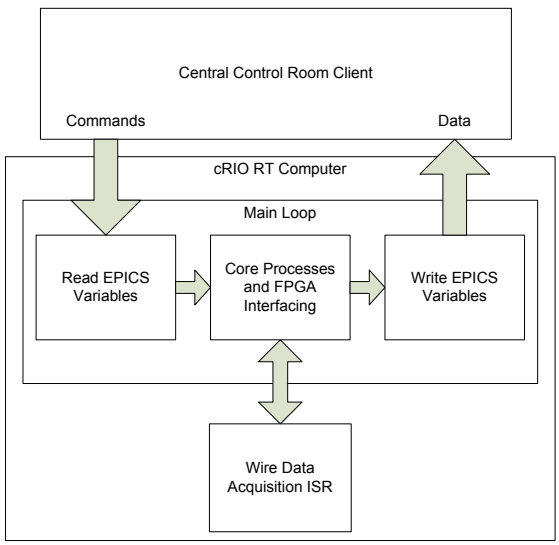

Figure 4: RT functional components and interface to the client.

This diagram shows how the programming has been divided into blocks. The first block is the main loop while the second block is the interrupt service routine described previously. The main loop has been further divided into three processes that occur in sequential order. The first process reads the input state from the EPICS variables and writes them directly to their associated LabVIEW variables. With this data loaded into LabVIEW variables, the second process performs the primary functionality of the RT system by transferring wire scanner controller data to and from the FPGA. The data received from the FPGA is then analyzed and written to LabVIEW local variables. Finally, the values of the LabVIEW variables are transferred to their associated EPICS variables for client access. The reasoning behind this approach is twofold. First, isolating the EPICS variables from the core code of the RT program allows for simpler interpretation of the EPICS interface between the wire scanner controller and client. The second reason is for simpler debugging of the wire scanner controller. By retaining native LabVIEW variables, the system can be tested without the EPICS interface.

SUMMARY

The utilization of a National Instruments compactRIO presents an interesting design challenge when targeting this platform for wire scanner control and data acquisition. The diversity of the programmable hardware allows the developer to trade functionality between the various hardware devices to leverage the relative strengths they possess.

REFERENCES

[1] J. Sedillo et al., "LANSCE Wire Scanner System Prototype: Switchyard Test," BIW'12 Newport News, April 2012, TUPG033; http://www.JACoW.org
[2] http://www.ni.com/compactrio/
[3] J.D. Gilpatrick et al., "Wire Scanner Beam profile Measurements: LANSCE Facility Beam Development," these proceedings, MOPPR080; http://www.JACoW.org

ISBN 978-3-95450-115-1