OPTIONS FOR INTERFACING EPICS TO COTS HARDWARE THROUGH LABVIEW

A. Veeramani, T. Debelle, National Instruments, Austin, USA
W. Blokland, R. Dickson, A. Zhukov, ORNL RAD, Oak Ridge, USA

Abstract

Over the years, many have developed custom drivers to interface hardware to EPICS Input Output Controller (IOC). With LabVIEW having native drivers for supporting commercial hardware, development time can be reduced if an interface with LabVIEW and EPICS IOC was developed. This paper examines the different ways of interfacing LabVIEW and EPICS IOC both on VxWorks and Windows operating systems. Implementation of the Channel Access (CA) server on LabVIEW will also be covered along with the advantages and limitations of such an approach. The paper will also list the status of the different implementations at Oak Ridge and Los Alamos National Laboratory.

LABVIEW BACKGROUND

LabVIEW is a general purpose graphical programming environment used to develop measurement, test, and control systems using intuitive graphical icons and wires that resemble a flowchart. LabVIEW is supported on a variety of platforms including Microsoft Windows (2K, XP, Vista), Linux, and Macintosh. LabVIEW Real-Time, the embedded systems solution for LabVIEW, is supported on VxWorks targets on hardware platforms such as CompactRIO and PXI. LabVIEW offers tight integration with thousands of hardware devices and provides hundreds of built-in libraries for advanced analysis and data visualization.

LabVIEW includes a shared variable engine (SVE) that runs as a separate process along-side LabVIEW and LabVIEW Real-Time applications. On Windows, LabVIEW configures the SVE as a service and launches the SVE at system start-up. On a real-time target, the SVE is an installable start-up component that loads when the system boots. The SVE is a software framework that enables LabVIEW applications to interact in a distributed manner with other LabVIEW and 3rd party applications, see Figure 1. LabVIEW application developers use a simple read/write API to access variables on a LabVIEW block diagram.

Figure 1: LabVIEW shared variable nodes.

The SVE uses a plug-in architecture, which makes it possible to interface with a wide range of data protocols such as OPC, Modbus Master and Slave.

INTERFACING EPICS AND LABVIEW

In order to take advantage of commercial off the shelf (COTS) hardware that is available through LabVIEW, the physics community has come up with different ways to bridge LabVIEW and EPICS such as the Shared Memory Interface and Channel Access (CA) Server from Oak Ridge National Lab. National Instruments has also developed EPICS client and CA Server in addition to the EPICS IOC to LabVIEW Real-Time interface that was developed in partnership with Cosylab.

SHARED MEMORY INTERFACE

The Spallation Neutron Source (SNS) is a particle accelerator driven pulsed neutron source. The Experimental Physics and Industrial Control System (EPICS) is used as the control system for the accelerator complex. There are more than 300 devices being controlled by Windows based PCs. These include: Beam Current Monitors, Beam Position Monitors, Wire Scanners and more. LabVIEW is used as the software development environment for these PCs. To integrate with EPICS clients, a communication interface between LabVIEW and EPICS is required.

Currently the main method at SNS of connecting LabVIEW with EPICS is the EPICS Shared Memory Interface [2]. The idea behind the Shared Memory Interface is to use shared memory from the operating system to allow different processes to exchange information. In principle more than two processes can communicate through the memory but in practice, the LabVIEW and EPICS IOC tasks are the only ones. In the EPICS context the shared variables are called Process Variables (PV) and the values of these are reflected in the shared memory. The shared memory interface implements additional features besides the sharing of values such as:

1. The setting and receiving of interrupts to notify other processes of events such as new value or to set the same timestamp for a group of variables
2. The buffering of values to avoid overwriting of old data that has not been read yet by one of the processes
3. Information fields about the variable such as the data type and size

To avoid that the LabVIEW programmer must also be an EPICS expert, the LabVIEW EPICS libraries generate all the files needed for the EPICS based on a VI the user has to customize to declare the variables that will be shared, see Figure 2. This also means that all declarations about the shared variables come from one place only and that is the LabVIEW code. Each time the LabVIEW program is started, all the EPICS files are regenerated to

avoid dealing with the EPICS IOC being out of sync with the LabVIEW program.

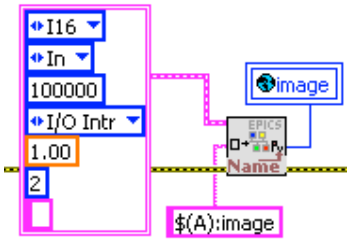

Figure 2: The declaration of a PV through LabVIEW code.

PURE LABVIEW CA SERVER

The Channel Access protocol is defined not by a description but by its libraries. This makes it difficult to implement a new version in a different coding environment. Only recently, documentation was made available describing the protocol [3]. This allowed us to consider a native LabVIEW implementation.

For our purposes, only the CA Server part of the EPICS IOC is required not the full IOC as all processing is done in LabVIEW. The main advantage of a LabVIEW native implementation is that the code will now run on any LabVIEW version that implements TCP/IP and UDP/IP.

This implementation takes advantage of LabVIEW dataflow paradigm and delegates all thread management to LabVIEW. This effectively allows running the same code on completely different hardware. We implemented limited set of CA types for testing purposes. The CA server supports all main CA operations like PUT, GET and MONITOR and works seamlessly with standard CA clients.

We have tested this on Windows, Linux, Mac OS X, CompactRIO [4]. We found that the maximum data throughput was about 160Mbit/s, which exceeds our needs. It was used for Beam Diagnostics Class at US Particle Accelerator School to show the integration of LabVIEW into an EPICS environment, see Figure 3. We chose the LabVIEW CA Server because of the simplicity of the installation; just copy the CA library. There is no installation or setup required.

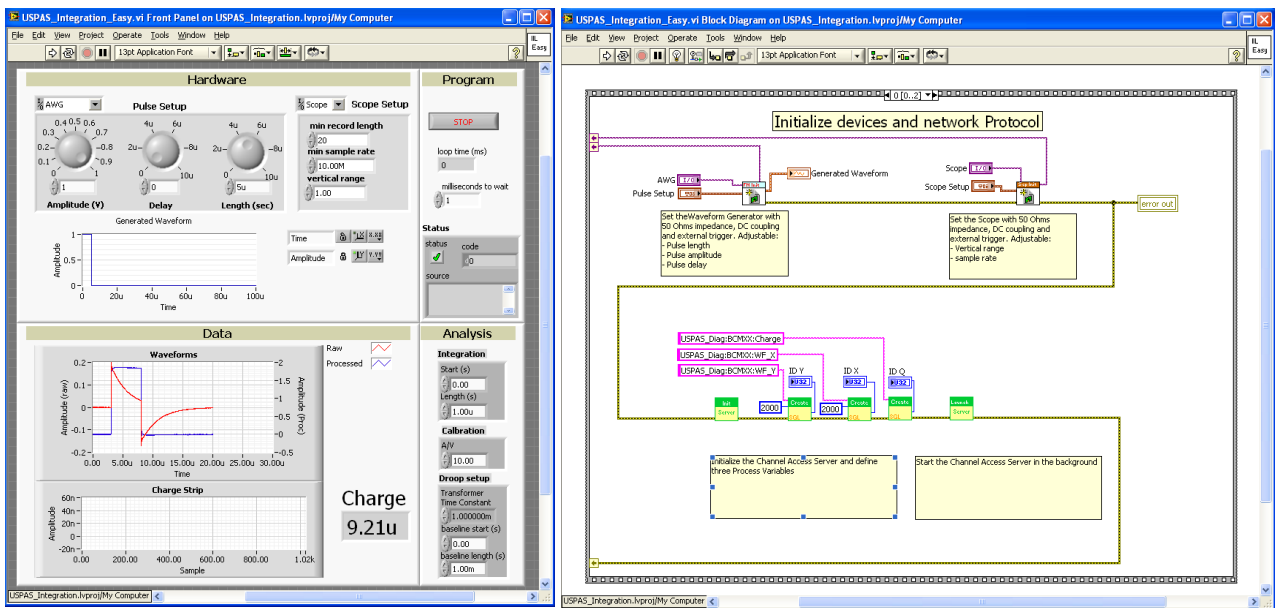

Figure 3: Pure LabVIEW CA server used for US Particle Accelerator School Class.

CONFIGURATION BASED CA SERVER IN LABVIEW 2009

In addition to the EPICS Client available in LabVIEW since version 8.5, LabVIEW 2009 introduced an EPICS Server which is really a configuration based CA Server. Similar to the CA Server implemented by SNS, the CA Server in LabVIEW allows LabVIEW enabled hardware to appear as an EPICS node. The processing is all done using the function blocks in LabVIEW.

Using the configuration based CA Server, which plugs-in to the SVE architecture, you choose which shared variables in LabVIEW need to be published as PVs over

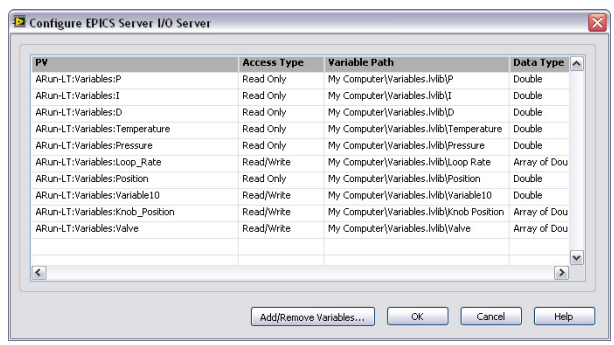

Figure 4: Configuration-based CA Server in LabVIEW 2009.

the CA network. The CA Server publishes value (VAL), description (DESC), timestamp (TIME) and the number of elements (NELM) in an array data type. This implementation of the configuration based CA Server can run on Windows and also in VxWorks and Pharlap RTOS. With the ability to run on Pharlap RTOS, PXI based instruments can now be integrated into an EPICS based control and data acquisition system.

EPICS IOC AND LABVIEW REAL TIME ON COMPACTRIO

The approach discussed here involves running a full EPICS IOC on CompactRIO's VxWorks operating system alongside with LabVIEW Real-Time [5]. A prototype of this solution was developed and is in use in the Los Alamos National Laboratory (LANL). The CompactRIO RTC is a Power PC processor running VxWorks – an architecture already supported by EPICS. The main requirements for the system were:

- System must run full EPICS IOC on the VxWorks platform.
- The FPGA must be configurable using LV FPGA.
- The two main processes (LV RT and EPICS) must be able to efficiently exchange data using different data types and arrays.
- System must be configurable without a need for recompilation of the EPICS source code (use of text configuration files).

It was decided to base the work on the Windows-based approach implemented by SNS discussed in this paper. The priorities of the EPICS-enabled VxWorks kernel are adjusted so that EPICS runs at lower priorities than the LabVIEW code. This prevents EPICS from interfering with LabVIEW's time-critical functions.

This system was installed in one accelerating module at LANL in 2009[6]. Once installed, the new system ran for the duration of the run cycle without any issues. Major part of the application was implemented in FPGA backplane of the CompactRIO. The LabVIEW code running on the real-time controllers was used only to serve the data between EPICS and the FPGA. The EPICS code's only function was to serve the data between the operator interface and the CompactRIO. As a result, the RTC itself was very lightly loaded, and the relative priorities between EPICS and LabVIEW were not a problem.

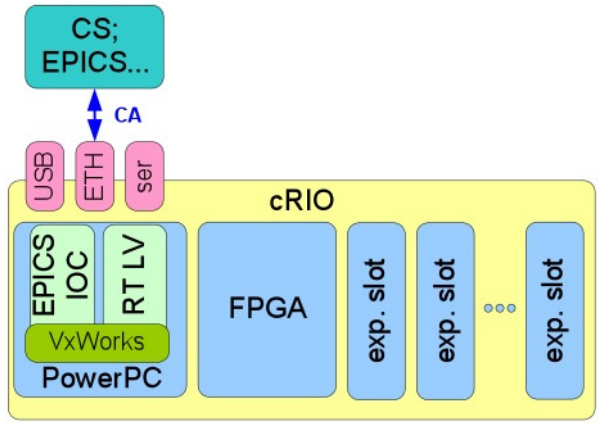

Figure 5: Basic co-habitation architecture.

ACKNOWLEDGEMENTS

ORNL/SNS is managed by UT-Battelle, LLC, for the U.S. Department of Energy under contract DE-AC05-00OR22725. The authors would like to thank Eric Björklund of LANL for sharing experience on using EPICS and LabVIEW on CompactRIO. We also would like to thank Rok Šabjan of Cosylab for his work on the VxWorks and EPICS portions, as well as for the initial shared library implementation that integrated all the parts together.

REFERENCES

[1] Experimental Physics and Industrial Control System http://www.aps.anl.gov/epics/.

[2] D. Thompson and W. Blokland, "A Shared Memory Interface between LabVIEW and EPICS", ICALEPCS 2003, pp275-277. Gyeongju, Korea, Oct 13-17 2003.

[3] Channel Access for Java; http://cosylab.com/solutions/particle_accelerators/Channel_Access_for_Java.

[4] A. Zhukov, W. Blokland, R. Dickson, "EPICS Channel Access Server Implementation in LabVIEW", THP018, this proceedings.

[5] A. Veeramani, K.E. Tetmeyer,R. Sabjan, A. Zagar "Interfacing EPICS IOC and LabVIEW for FPGA Enabled COTS Hardware", PCaPAC'08, Ljubljana, Slovenia. October 2008, TUX01; http://www.JACoW.org.

[6] E. Björklund, A. Veeramani, T. Debelle "Using EPICS Enabled Industrial Hardware for Upgrading Control Systems", WEP078, this proceedings.