

LabVIEW LIBRARY TO EPICS CHANNEL ACCESS *

A. Liyu¹, W. Blokland, and D. Thompson, Spallation Neutron Source, Oak Ridge, TN, USA
¹Institute for Nuclear Research, Moscow, Russia.

DESIGN

Abstract

The Spallation Neutron Source (SNS) accelerator systems will deliver a 1.0 GeV, 1.4 MW proton beam to a liquid mercury target for neutron scattering research. The accelerator complex consists of a 1 GeV linear accelerator, an accumulator ring and associated transport lines. The SNS diagnostics platform is PC-based and will run Windows for its OS and LabVIEW as its programming language. Data acquisition hardware will be based on PCI cards. There will be about 300 rack-mounted computers. The Channel Access (CA) protocol of the Experimental Physics and Industrial Control System (EPICS) is the SNS control system communication standard. This paper describes the approaches, implementation, and features of a LabVIEW library interface to CA for Windows, Linux, and Mac OS X. We also discuss how the library implements the asynchronous CA monitor routine using LabVIEW's occurrence mechanism instead of a callback function (which is not available in LabVIEW). The library is used to acquire accelerator data and applications have been built on this library for console display and data-logging.

INTRODUCTION

The Experimental Physics and Industrial Control System (EPICS), see [1], is a Supervisory Control And Data Acquisition system that has its own communications protocol between clients and servers. On the client side EPICS Channel Access (CA) protocol is a collection of methods and is implemented as C library, see [2]. Some interface libraries for other languages (Perl, Matlab, Java) that wrap the CA C library have been developed.

LabView can also use a CA. library developed to export ActiveX interfaces see [3], but it only supports the Windows OS. An interface to Linux was also needed as the Spallation Neutron Source (SNS) project uses the Linux OS for the standard console.

Starting with version 3.14.0 EPICS is ported into many different OSs. LabVIEW works in Windows, Linux and Mac OSs. So it was decided to create a library that would work on all the three platforms where LabVIEW is available. The interface layer is to be OS independent and should also be able to buffer the incoming data for LabVIEW to provide easy synchronization to the LabVIEW user.

LabVIEW has several methods to communicate such as network protocols like TCP/IP, DDE, ActiveX, WEB and file I/O and can also interface to shared libraries. The shared library call is fast and can be easily integrated with the existing CA libraries. LabVIEW supports it with a Call Library Function Node that links to a dynamic library in Windows and shared library in Linux and Mac OS X.

The EPICS CA library is implemented on C language in form of a dynamic ca.dll library in Windows, a shared ca.so library in Linux and framework in Mac OS X. LabVIEW and C have similar scalar data types, like integers or floats, but LabVIEW requires a different array data format and thus also different structure formats. So it is not possible to call EPICS CA C functions directly from LabVIEW when an array argument is present.

The EPICS CA asynchronous mechanism of transferring data implements non-blocking library calls by passing a pointer to a function that CA can call later when data is available. LabVIEW does not support a callback mechanism. To solve the problem of asynchronous mechanism, we used inter-thread communication. To synchronize threads and transfer data between threads we used the LabVIEW's Occurrence mechanism that supports functions to wait for an occurrence and set an occurrence. Both functions can be called from C and LabVIEW.

IMPLEMENTATION

The LabVIEW interface to CA consists of two parts, a C layer and a library of LabVIEW VIs calling the C layer, see Figure 1. The first part, the C layer handles the difference in array and structure formats and implements a substitute callback function using the LabVIEW occurrence. In addition, to prevent data losses when data comes in temporarily faster than LabVIEW can retrieve the data, the C layer buffers the data. The C layer also

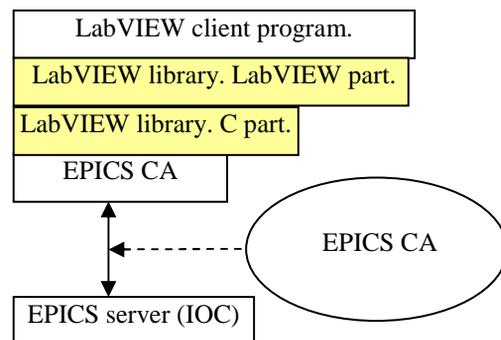


Figure 1: LabVIEW library structure.

checks to make sure that all data has been read out, and if not, gives an additional occurrence to LabVIEW to

* SNS is managed by UT-Battelle, LLC, under contract DE-AC05-00OR22725 for the U.S. Department of Energy. SNS is a partnership of six national laboratories: Argonne, Brookhaven, Jefferson, Lawrence Berkeley, Los Alamos and Oak Ridge.

retrieve the data. The C part was compiled by Microsoft Visual C in Windows, and GNU compiler in Linux and Xcode MacOS.

The second part is the LabVIEW library. The library is kept very simple so the LabVIEW user has to know just 7 VIs and needs only a minimum CA knowledge. LabVIEW of versions 6.1 and up were tested.

The LabVIEW library supports individual CA synchronous put/get and monitor. The library has VI examples to explain the basic EPICS CA usages. Figure 2 shows monitor example. The data types supported are scalar and array versions of character, short, long, float, double, and EPICS string.

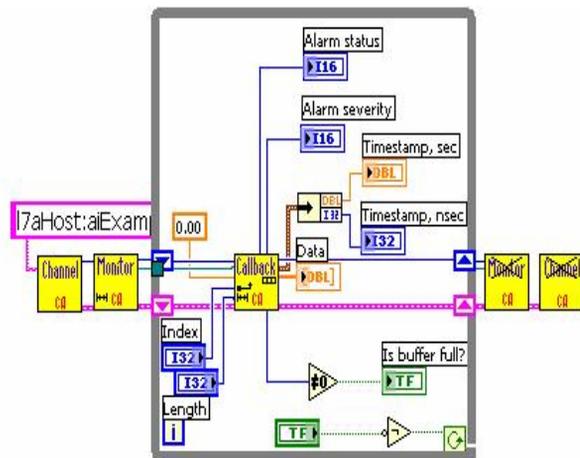


Figure 2: Monitor example.

We found one more problem when we started using the library. When the LabVIEW program finishes using CA and calls the last VI. That VI destroys CA and could sometimes cause LabVIEW to crash. The problem occurred only on Windows. Many tests were done to find the location of this problem (LabVIEW, the library itself or EPICS CA). Finally, it was noticed that LabVIEW calls a dll by explicit method. Usually, program practice is to call a dll by implicit method. The simplest C program with explicit method was written. The program still

repeatedly crashes. The crash behavior depends on the computer type, quantity of functions being used and program “velocity”. That problem will be fixed from EPICS version 3.14.8.

APPLICATIONS

The library has already been used in many ways. We have created LabVIEW based console applications for the emittance scanner for the Linux console and data logging tools for the Windows PCs, see [4], to test instrument performance, and an application to create a visual on-line summary report, see [5].

FUTURE

Another problem was met when programmers had written some client programs. The library provides a simple interface but the interface has less power than EPICS CA library. It created difficulties in complicated client programs. Creating library that supports all functions of the EPICS CA library is a possible solution.

ACKNOWLEDGEMENTS

We would like to address our thanks to J. Hill, LANL.

REFERENCES

- [1] EPICS home page, at <http://www.aps.anl.gov/epics/>
- [2] J. O. Hill, "EPICS R3.14 Channel Access Reference Manual"
- [3] K.U. Kasemir, "ActiveX: CA client and server for Active-X programs"
- [4] M. Sundaram et. al." SNS Diagnostics Tools For Data Acquisition and Display". PAC05 conference.
- [5] W. Blokland et. al. "Dynamic Visualization of SNS* Diagnostics Summary Report and System Status" PAC05 conference.