Access of Epics Channel Using Library of LabVIEW

Prof. A. B. Chavan, Prof. A.G. Bajaj, Prof. S. N. Suryatale and Prof. S. S. Jadhao

Abstract: **The SNS diagnostics platform is PC-based and will run Windows for its OS and LabVIEW as its programming language. There will be about few hundred rack-mounted computers. The Channel Access (CA) protocol of the Experimental Physics and Industrial Control System (EPICS) is the SNS control system communication standard. The accelerator complex consists of a 1 GeV linear accelerator, an accumulator ring and associated transport lines. DA (Data acquisition) hardware will be based on PCI cards. The library implements the asynchronous CA monitor routine using LabVIEW's occurrence mechanism instead of a call back function (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. This paper describes the approaches, implementation, and features of a LabVIEW library interface to CA for Windows, etc.**

Keywords: LabVIEW, SNS, EPICS, IOC, EPIC CA, Data Acquisition (DA).

I. INTRODUCTION

LabVIEW works in Windows, Linux and Mac Oss. The Experimental Physics and Industrial Control System (EPICS) is a Supervisory Control & Data Acquisition system (DAS) that has its own communications protocol between clients and servers. At client side EPICS Channel Access (CA) protocol is a collection of methods and is implemented as C library.

LabView can also use a CA. library developed to export ActiveX interfaces, 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.

In an effort to help standardize a network communication protocol for large scale scientific experiments, Los Alamos National Laboratory and Argonne National Laboratory collaborated to develop EPICS. Many scientific and industrial organizations worldwide use EPICS and a group of those organizations are charged with maintaining the EPICS standards, documentation and tools. This group of user organizations are also referred to as EPICS.

Prof. A. B. Chavan, Prof. S. N. Suryatale and Prof. S. S. Jadhao are working as *Asst. Professor, Dept. Of Electronics & Telecommunication Engg, Aurangabad, India*[1] *and* Prof. A.G. Bajaj is working as *Asst. Professor, Dept. Of Electronics & Telecommunication Engg, Jalna, India*[2]

Large scale scientific applications often require hundreds of devices to communicate over a single network to form large distributed control systems. EPICS provides the standards and tools necessary to make this kind of communication possible. To view more information about EPICS or download the latest documentation and software tools, visit the Argonne National Labs website.

Starting with version 3.14.0 EPICS is ported into many different 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. Some interface libraries for other languages that wrap the CA C library have been developed.

Architecture

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.

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.

LabVIEW has several methods to communicate such as network protocols 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.

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

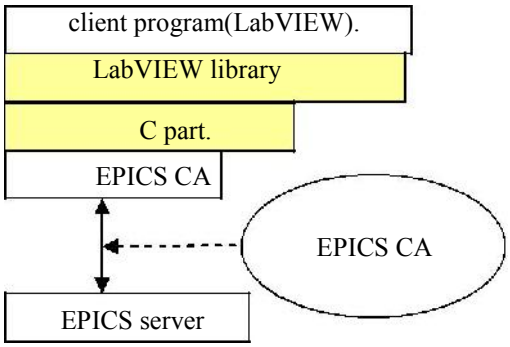

Figure 1: LabVIEW library Architecture.

checks to make sure that all data has been read out, otherwise it gives an additional occurrence to LabVIEW to 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 because the LabVIEW user has to know just 7 VIs and needs only a minimum CA knowledge.

The LabVIEW library supports individual CA synchronous put/get and monitor. Figure 2 shows LabVIEW Code. The data types supported are scalar and array versions of character, short, long, float, double, and EPICS string.

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 was written repeatedly.

An interface to Linux was also needed as the Spallation Neutron Source (SNS) project uses the Linux OS for the standard console.

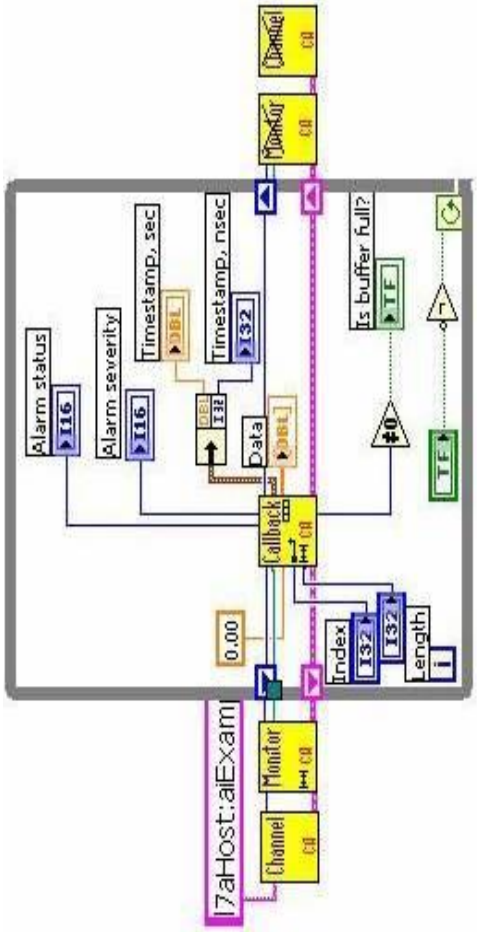

Figure 2: LabVIEW Code.

The crash behaviour depends on the computer type, quantity of functions being used and program "velocity". That problem will be fixed from EPICS version 3.8.

APPLICATIONS

The library has already been used for many Applications. There is a LabVIEW based console application for the emittance scanner to test instrument performance, and an application to create a visual on-line summary report.

FUTURE SCOPE

The system meets to a problem, in client programs. The library provides a simple interface but the interface has less power as compared to EPICS CA library. It created difficulties in complicated programs. Creating library that supports all functions of the EPICS CA library can be a possible solution.

REFERENCES

[1] K.U. Kasemir, "ActiveX: CA client and server for Active-X programs"
[2] M. Sundaram ET. Al." SNS Diagnostics Tools for Data Acquisition and Display". PAC05 conference.
[3] W. Blokland ET. al. "Dynamic Visualization of SNS*

Diagnostics Summary Report and System Status" PAC05 conference.

BIOGRAPHY

Prof. Amar Chavan was born at Udgir, Maharashtra, India on 1st Nov.' 1986. Currently, He is working as Assistant Professor. He did his M.tech in Electronic Design & Technology from DOEACC, Aurangabad. Also he did his B.tech in Electronics and Communication Engineering from Dr. Babasaheb Ambedkar Marathwada University, Aurangabad. His area of interest includes Digital Electronics, Power Electronics, Instrumentation and system designing

Prof. Sachin Jadhao was born at Buldhana, Maharashtra, India on 29th June.' 1985. Currently, He is working as Assistant Professor. He did his M.tech in Electronic Design & Technology from DOEACC, Aurangabad. Also he did his B.E. in Electronics Engineering from Dr. Babasaheb Ambedkar Marathwada University, Aurangabad. His area of interest includes Embedded Systems, Electronics System Design, DSP Processors.

Prof. Ashish G Bajaj has graduated in Electronics & Telecommunication Engineering from Dr.BAMU, Aurangabad in 2006 and had lifetime membership for Indian Society for Technical Education.

Prof. Satish N. Suryatale has graduated in Electronics & Telecommunication Engineering from Dr.BAMU, Aurangabad in 2008 and had lifetime membership for Indian Society for Technical Education. He is perusing ME in Electronics from Govt .College of .Engg. Aurangabad. His area of interest includes Digital system and Communication.