Developing an EPICS IOC in LabVIEW

Kenneth Butler  
*Virginia Commonwealth University*  

David Shires  
*Virginia Commonwealth University*  

Michael Talbott  
*Virginia Commonwealth University*

Follow this and additional works at: [https://scholarscompass.vcu.edu/capstone](https://scholarscompass.vcu.edu/capstone)  

Part of the [Computer Engineering Commons](https://scholarscompass.vcu.edu/capstone)

© The Author(s)
Developing an EPICS IOC in LabVIEW

Team Members:
Kenneth Butler, David Shires, Michael Talbott
Faculty Advisor:
Preetam Ghosh
Sponsor:
Jefferson Labs
Sponsor Advisor:
Christiana Wilson

The goal of this project was to prove (or disprove) the ability of caLabs soft IOC, which is an interface between LabVIEW and EPICS, to serve data over a network. The larger project this will be implemented in is the upgrade of the superconducting test facility, called the Vertical Test Area. Jefferson Labs builds superconducting cavities for particle accelerators around the world. Once these superconducting cavities are built they need to be cooled with liquid helium and injected with high power radio frequency (RF). The output power of the radio frequency is then measured to determine proper quality. There are 8 dewars that hold the liquid helium that is used to cool the cavities to the proper temperature.

Original System
- Apple II computers read dewar data and relayed information to a computer running labVIEW
- This data is sent to a soft IOC running on Linux

Proposed System
- Dewar information is read by Programmable Logic Controllers (PLCs) then relayed to a Cisco switch
- Data is then read using soft IOCs through labVIEW and caLabs

Our Task
- Prove data can be passed from server to client over a TCP/IP connection using Windows LabVIEW and caLabs
- Test reliability of data transfer
- Develop UI for testing soft IOCs

Initial Proof of Concept
- This LabVIEW program acts as a soft IOC server
- Process Variables (PVs) are given a name and a value (via string input) and broadcast over the network on port 5065

Initial Proof of Concept
- This LabVIEW program acts as the soft IOC client
- It searches the network for PV names given in list format
- It displays the PV values for the names listed (Test input displayed above)

User Interface
- Interface was designed in python using pyepics and wxpython libraries
- Can dynamically add or remove PV fields for monitoring PV values
- Takes PV name as string input (Enter PV name)
- Outputs PV value associated with that PV name

Demo
- This is an example of the UI reading two PVs and displaying their values
- When a PV value is changed the UI updates the value field accordingly
- (In the background) using the EPICS command line tools to set the values for the PVs

In Conclusion
Our concept has proven that LabVIEW, caLabs, and EPICS can be served over a Windows platform and over a network. In the future, these findings will serve as a basis for Jefferson Labs to move forward in their implementation of this system on their network.

In the Original System, data was read by Apple II computers and relayed to a computer running LabVIEW. The data was then sent to a soft IOC running on Linux. In the Proposed System, dewar information is read by Programmable Logic Controllers (PLCs) and relayed to a Cisco switch. Data is then read using soft IOCs through LabVIEW and caLabs.

Our Task involves proving that data can be passed from server to client over a TCP/IP connection using Windows LabVIEW and caLabs. We will test the reliability of data transfer and develop a UI for testing soft IOCs.

The Initial Proof of Concept involves a LabVIEW program acting as a soft IOC server. It allows users to give Names and Values (via string input) and broadcast them over the network on port 5065. For the Initial Proof of Concept, the LabVIEW program acts as the soft IOC client. It searches the network for PV names given in list format and displays the PV values for the names listed (Test input displayed above).

The User Interface was designed in Python using pyepics and wxpython libraries. It allows users to add or remove PV fields for monitoring, and takes PV name as string input (Enter PV name). The Outputs PV value associated with that PV name.

The Demo involves an example of the UI reading two PVs and displaying their values. When a PV value is changed, the UI updates the value field accordingly. (In the background) using the EPICS command line tools to set the values for the PVs.

In Conclusion, our concept has proven that LabVIEW, caLabs, and EPICS can be served over a Windows platform and over a network. In the future, these findings will serve as a basis for Jefferson Labs to move forward in their implementation of this system on their network.