LA-UR-15-22131
Approved for public release; distribution is unlimited.

Title: Production Facility SCADA Design Report

Author(s): Dale, Gregory E.
Holloway, Michael Andrew
Baily, Scott A.
Woloshun, Keith Albert
Wheat, Robert Mitchell Jr.

Intended for: Report

Issued: 2015-03-23

Production Facility SCADA Design Report

FY14 NNSA Mo-99 Program

FY14 NorthStar Activity 3, Task 1 Report

Gregory Dale, Michael Holloway, Scott Baily, Keith Woloshun, Robert Wheat

Overview

The following report covers FY 14 activities to develop supervisory control and data acquisition (SCADA) system for the Northstar Moly99 production facility. The goal of this effort is to provide Northstar with a baseline system design The SCADA system will be designed to manage the following elements of plant operation:

- Control helium flow in the target cooling system
- Interrogate sensors that monitor helium temperature, pressure, and flow rate and to manage safety interlocks related to these values
- The following diagnostic systems and related safety interlocks:
    - Optical transition radiation (OTR) beam profile and position monitoring
    - Non-contact target window temperature monitoring (Infrared imaging)
    - Beam position monitors along the beam line
- Data archiving

The design and testing of these subsystems is taking place at both Los Alamos National Lab and Argonne National Lab. At Los Alamos we built prototype helium flow loop and at Argonne national Lab we fielded various systems test during production experiments at the accelerator facility. We drew from the experience of personnel responsible for similar control systems at the LANCE proton accelerator facility help guide many of our design choices. The following report discusses the current state of the design and testing effort and intentions for future improvement.

EPICS Based Control System

The system currently fielded in building MPF-14 at LANL for control of the flow visualization loop has been designed and implemented with future growth as well as disaster recovery in mind. Adding new subsystems and subsystem controllers has been simplified and can be performed without affecting existing subsystems. Examples of typical subsystems include computers for monitoring Process Variables (PC's) and computers or embedded systems controlling or monitoring hardware devices. Choices for Operating Systems, control hardware and components, and most other aspects of a subsystem design are left to the discretion of the engineer with two exceptions; the subsystems must be able to communicate on an Ethernet based network, and the subsystem must be able to provide EPICS compatible communications on the network. Disaster recovery is addressed by providing multiple "versioned" backups of the server disk drive per day, and shared disk drives as well as automated file synchronizing system services provide for all subsystems having access to the most current versions of control software and configuration files. The server computer archives data or process variables chosen by the system administrator. Figure 1 shows a schematic of the implementation of the current prototype facility control system.

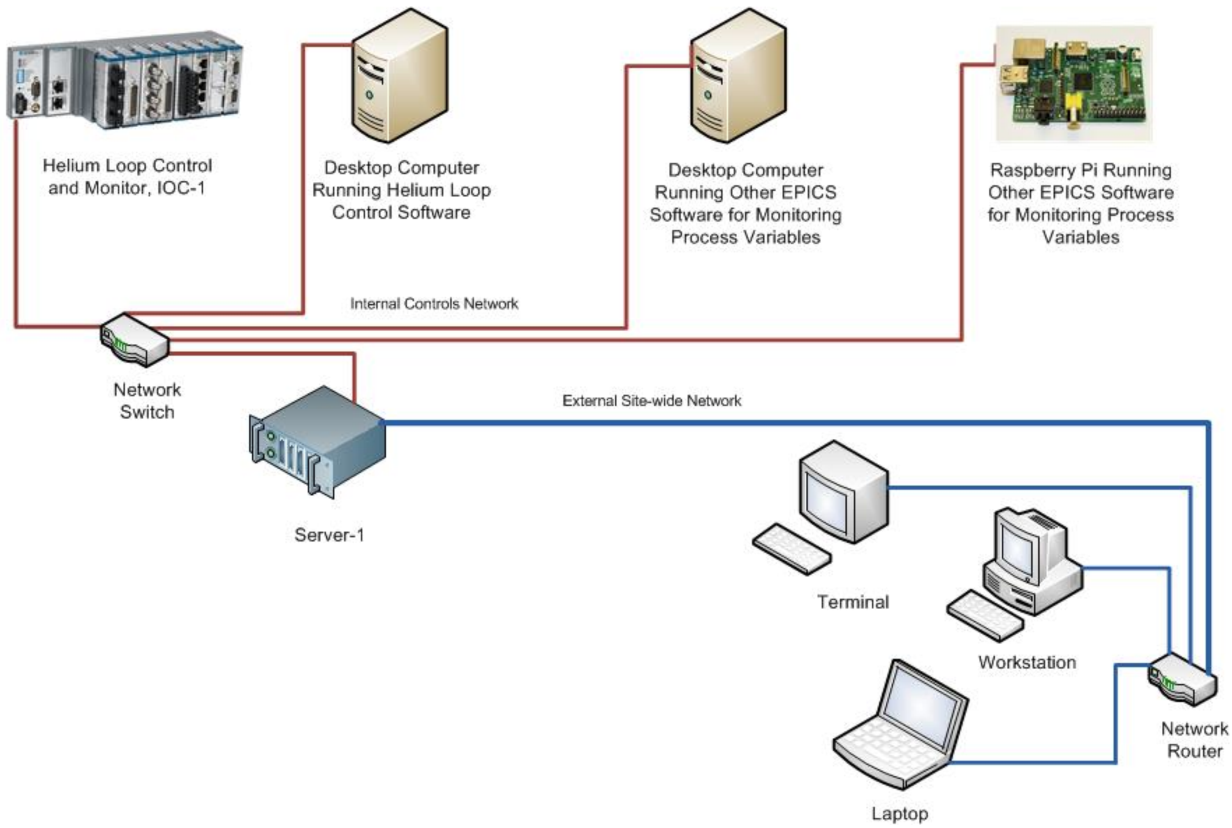

Figure 1 - A schematic of the current prototype production facility control system implementation.

Hardware and Components

The currently implemented prototype facility control system shown in Figure 1 consists of the server computer, the cRio system interfaced to the cooling loops controls and sensors, 2 desktop computers, and a Raspberry Pi embedded computer. Since the server contains two Ethernet adapters, one connected to the internal controls network and the other to the external "site-wide" network, authorized computers on the site-wide network can access the server to monitor or control any system connected to the internal controls network. Not shown in Figure 1 are the Uninterruptable Power Supplies, backup disk drive(s), server monitor and keyboard used for diagnostic purposes, and extra switches or hubs implemented for convenience.

Special Control and User Software

The server for the control system runs a version of Linux as its operating system. Many of the built-in functions of a Linux system are exploited for this control system and include, Cron to run jobs at specific times of the day, shell scripting to provide easy methods for executing multiple operations either automatically or manually from the command line, log "combing" functions to provide operational information to the administrator, system firewall services, remote desktop software, and secure shell and copying software. Experimental Physics and Industrial Control System (EPICS) software has been implemented both on the server as well many of the computers connected to the controls network. From the server side of the network, the EPICS software provides the capability to request the value of any Process Variable "published" by any IOC on the network. From the IOC side of the network, the EPICS software allows the controller to publish any Process Variable it controls or monitors. The EPICS software can be compiled and run on any Linux, Windows, MacOS, or VxWorks operating systems.

To provide for Graphical Representation of Process Variables as well as controls such as buttons and switches, software name caQtDM has been implemented. This allows for the creation of sophisticated control and monitoring graphical screens for user interfaces, communicates using the EPICS protocols (interfaces with the EPICS software), and can be compiled and executed on the operating systems mentioned earlier for EPICS itself (excepting VxWorks). Figure 2 shows an example of a caQtDM operator interface screen showing some controls and "read backs" or monitors.

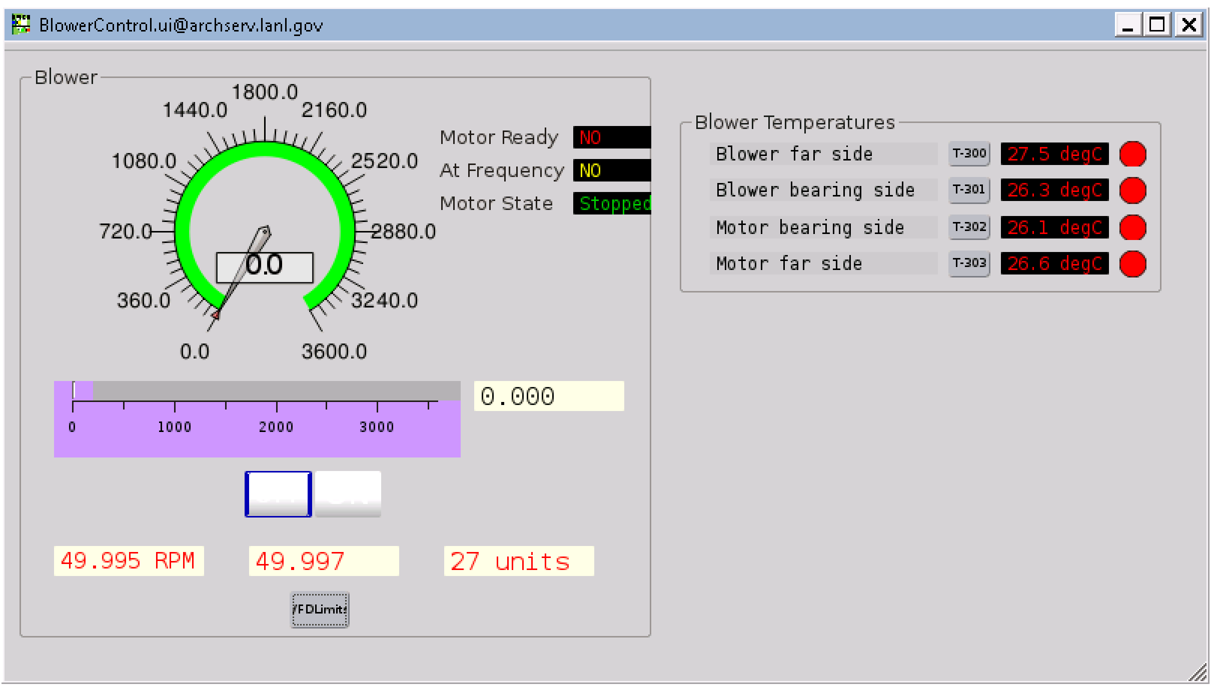

Figure 2 - An example caQtDM operator screen some controls and monitors.

The EPICS community provides software named StripTool which can be compiled and run on Linux, Windows and MacOS and is very useful. The software allows for very quickly, "on the fly", setting up a strip chart type monitor for observing one or more Process Variables. The configuration of this strip chart can be saved providing an even quicker method for monitoring data. Figure 3 shows an example of a StripTool screen providing plots for the data variables.

In addition to StripTool, the EPICS community also provides EDM and MEDM. MEDM can be compiled and executed on Linux, Windows, and MacOS, but EDM can only be compiled and executed on a Linux based machine. Both of these applications allow for quickly setting up and configuring graphical interfaces for both control and monitoring purposes. MEDM and EDM are provided on the server for the prototype system, and should always be included in a Linux based EPICS installation.

The EPICS standard installation provides several command line programs which are useful for diagnosing server or IOC issues, getting a quick "peek" at the value of a Process Variable without running a GUI, and many other purposes too numerous to mention here. Some of the most used command line programs are 'caget', 'caput', and 'camonitor'.

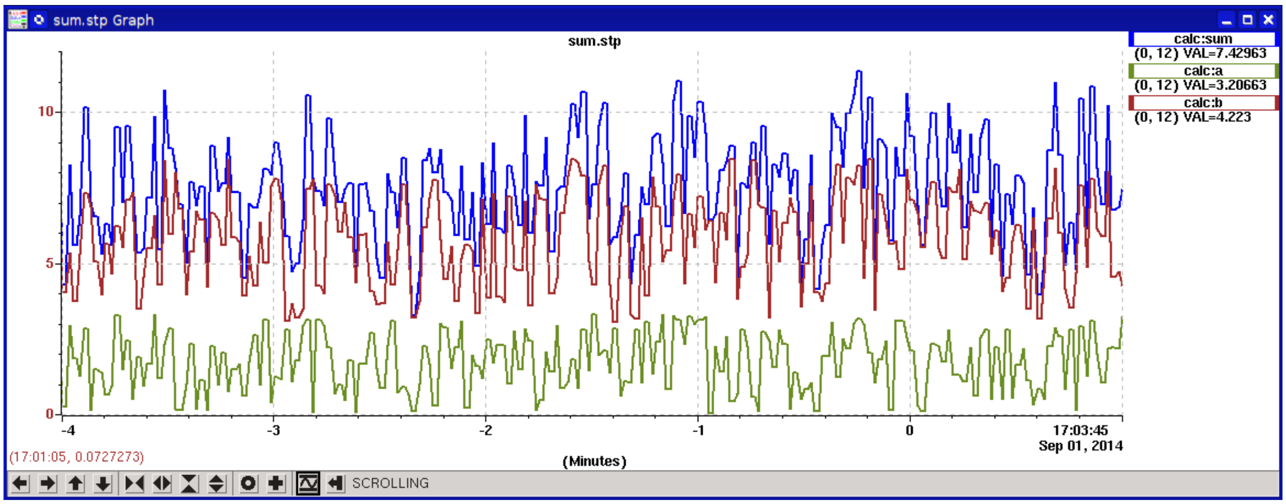

Figure 3 - A sample StripTool operator screen.

Finally, the system software named Nagios has been implemented on the control system server. Nagios is software that will monitor configured system functions, such as CPU usage, memory usage, disk usage, network usage, and many other aspects of a computer's operation. Nagios will also perform these monitoring functions on all computers it has been configured to monitor, including computers running Linux, Windows, or Mac operating systems. Nagios provides a web page interface such that authorized users can view the current status of any given computer in the configuration. Figure 4 shows a typical Nagios service details web page. Finally, when problems have been identified by the Nagios software, or limits have been reached, the software will send an email to the administrator(s) advising of the problem and the computer experiencing the problem.

Data Archiving

Archiving of data from any IOC connected to the same network as the server is performed by a Python based program named EPICS_PV_Archiver. This archiver saves data in a MySQL database and has the ability to archive transient events occurring on the time scale of tens of milliseconds. Conversely, if any given Process Variable remains unchanged, within a configured dead band range, no additional values are written to the database. The EPICS_Pv_Archiver also provides a web interface allowing for the listing of archived PV's and also for the plotting of up to two PV's at a time, over a user selectable time period. In addition to the web interface for the display of a time window of data, some C/C++ and Matlab programs and functions have been written to access the database, extract the data for the specified time period, and display the extracted data on graphs.

Some configurable options for the archiver include the specific Process Variables to archive, the dead band range for each variable, and the frequency at which to sample and save the data.

User Workstations

User workstations can be just about any kind of computing device that can access the network. Since the data associated with this prototype system is available from the archiver web page, data which is only a few seconds old can be displayed using the web page. Computing devices capable of running EPICS support software can exploit EPICS and the availability of data in real time. Otherwise, if the computing device is capable of running an X-Server, the device can login to the main server and access any or all of the user interface programs developed to date for the prototype system, based on authorization. As is shown in Figure 1, Windows systems easily co-exist with Linux systems. The Raspberry Pi is a Linux system running its own version of the EPICS tools to access the developed operator interfaces. The same can be said about the Windows computers, that is, each Windows computer runs a native version of EPICS, and a native version of caQtDM to access and execute the associated user interfaces. Both Windows computers and Linux computers can run the StripTool as well as MEDM, and the Linux boxes can also run EDM.

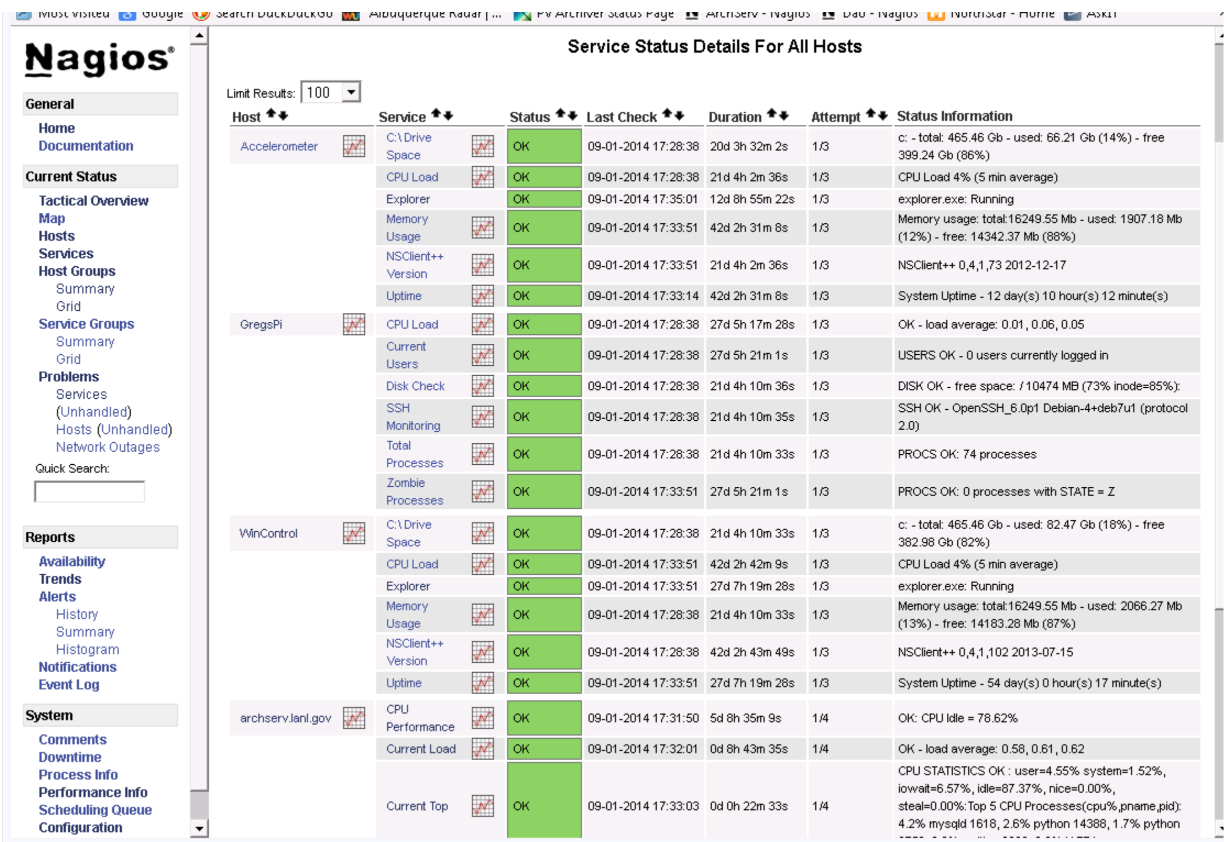

Figure 4 - The Nagios service details web page display.

Input Output Controllers

Input output controllers consist of a computing device interfaced directly to some hardware devices, either to control these devices, or monitor them, or both, and having the capability of "publishing" data to EPICS, and listening to EPICS for changes in control values. A simple example of an IOC would be a computer connected to a thermometer. The computer can "publish" the value read from the thermometer at configured time intervals or whenever queried, and can accept a command from EPICS to turn the thermometer on and off. The combination of computer and thermometer (and the interface used to connect the two), and the software allowing the two way EPICS communication, makes the system an IOC.

Adding additional IOC's can be as simple as plugging them into the controls network. If the new system is running the proper EPICS software, then the EPICS software on all other computers in the system will automatically see this new IOC and all of the associated Process Variables. If it is desired, these new Process Variables can be added to the archive list, to be archived as new data is generated. Also, if desired, this new IOC can be added to the list of systems to monitor by Nagios, to alert the administrator should something go wrong with the system. Finally, the name and IP address of the new IOC should be added to the "hosts" file of the server, at least, and possibly all computers on the controls network.

Future Improvements

The archiver software was written at a different institution and was very customized to its hardware configuration. Although the software has been modified enough to work for this prototype system, much improvement could be made to this software such that it would work better, and give the administrator much more flexibility in configuration. The archiver requires quite a bit of the server resources, and if the software were modified properly, much of this load could be lessened.

Additional IOC's exist for the accelerator facility, but still need to be added to the system. In most cases, EPICS support was not included in the original software, so that modifications and testing of this software is necessary to add these IOC's to the system.

The control system is a distributed system, and availability, performance, dependability, and predictability of each of the nodes in the distributed system is important. In working towards maximizing these goals, and understanding when and why they may not be optimum, many system performance and availability monitoring tools have been deployed including Nagios, top, htop, atop, nmon, sar, etc. There still remains much uncertainty as to the causes of temporary degradation in system performance and/or availability, such that, obviously, dependability and predictability suffer. Therefore, it is important to identify that which is currently lacking in available systems monitoring and analysis tools and providing tools to fill those gaps.

Redundancy of hardware such as power supplies and disk drives could be important as far as system availability. These aspects of system implementation need to be studied and determinations made as to cost, complexity, and payoff or return on investment.

Flow Loop IOC

Overview

The system currently fielded in MPF-14 at LANL for control of the Target Loop Cooling System is a subset or prototype for a production facility system. Aspects of a production facility target cooling system are tested and measured, along with the implemented hardware and software. It is also used as a test bed for prototype subsystems and new control software and methods. The system consists of the gas compression and circulating system, the water cooling system, the target holding apparatus, controls and sensors, the cooling system Input Output Controller (IOC), the system server, and several “user” computers used to run EPICS control and monitoring software as user interfaces to the system. Efforts to include other beam diagnostic subsystems such as the Beam Current Monitor and Beam Optical Diagnostics have begun and are goals for future improvements to the prototype facility system.

Hardware

The hardware for the helium flow loop input/output controller is a National Instruments 9024 compact RIO (800 MHz PowerPC, 512MB DRAM, 4GB single-level flash storage, -20 to 55 °C operating temperature) with a 9118 Chassis (8-slots with a Virtex-5 LX110) with 8 c-series modules installed: four 9217 RTD modules, two 9208 modules, one 9375 binary I/O module, and one 9265 module. This system is capable of measuring 16 RTD sensors (3 or 4 wire with 24-bit resolution), reading 32 current inputs (-21.5 to 21.5 mA range 24-bit resolution), reading 16 binary inputs (12 or 24 V sinking), commanding 16 binary outputs (6-30 V sourcing), and controlling 16 current outputs (0 to 20 mA, 16-bit resolution). This controller is fairly powerful, allowing for rapid development without having to spend time optimizing performance/size at every step, and the software runs on hardware that is identical to that used at LANSCE. These are fairly rugged controllers, and National Instruments has promised a fairly long support lifetime.

Sensors

Current loops and 24 V binary signals are relatively immune to the noisy environment in which these systems will need to be able to run reliably. We chose to use resistance temperature detectors (RTDs) instead of thermocouples because of the difficulties in installing wiring at ANL. RTSs also have much higher inherent accuracy than thermocouples. However, a single RTD input module (NI 9217) can only read 4 input channels, whereas a single Thermocouple input module (NI 9213) has 16 input channels. Thermocouples are much less expensive, but the wiring is more expensive because all wiring must be of the correct material. Feed-through connectors must also be of the same material, rather than a material chosen for robust connections. In the next system we might evaluate if it would be worthwhile to use thermocouple with an intermediate junction point. Such a system would have fairly short runs of thermocouple wire. These would have junctions to standard copper wire on an isothermal block with a measured temperature. A single sensor (thermocouple, RTD, or thermistor) would need to accurately measure the block temperature, and one would have to be careful that junctions take place at this temperature. Then only 1 set of thermocouple wires or 4-wire RTD wires would need to be run to the control system (to calculate provide thermal compensation), the rest could be a single copper pair for each thermocouple to the junction box. This more complicated setup would only be justified for a high channel count with long cable runs to the control system.

User Interface

For an operator interface we needed a cross platform user interface, and a native Microsoft Windows application was one of the requirements. We could have developed a custom control application, but this would not be easy to maintain and adapt as necessary. A generic operator interface application allows users with little experience to modify and develop new interface screens, as the process is similar to editing a document rather than compiling software. For this type of task, the most reasonable choices are caQTDM and BOY. These are the two most promising recently developed operator interfaces for EPICS. It's not clear which one (perhaps both) will be widely adopted by the accelerator community, but both have been around a few years and are under active development. Other widely adopted choices (nearly all are MOTIF and X11 based) are getting long in the tooth, active maintenance is limited, and are difficult to run on Microsoft Windows, except as a remotely hosted application. BOY (Best Operator interface Yet) is a Java Eclipse-based operator interface that's part of Control System Studio. The QT-based caQTDM can be built to run on the wide variety of systems that support QT and QTW (including Linux and Mac OS X), and even includes a binary distribution for Microsoft Windows. We have chosen to use caQTDM for the flow visualization loop control system.

cRIO

The cRIO IOC boots out of its local flash storage, and runs EPICS R3.14 with the same industrial I/O software and firmware design used by cRIOs on the LANSCE control system. This design allows for general purpose use of the I/O and permits many run-time modifications to the configuration. Generally, the FPGA software would not need to be reloaded unless different I/O modules are used than originally planned. Interactions between inputs/outputs are handled by the EPICS database loaded on the IOC. Local storage makes the system boot even in the event of network or server outages, as it is not dependent on the server. However it is desirable to keep the contents synced with that of a file server for ease of maintenance. We've started to add this feature for many of the systems, but have not fully implemented it for the compact RIO yet.

Additional Diagnostics and IOC's

Beam Current Monitor

Optical Transition Radiation Beam Diagnostics

Optical Transition Radiation (OTR) that is emitted when the electron beam strikes the target windows is used to monitor the position and size of the beam as it enters the target. This radiation is observed and recorded using a simple monochrome CCD camera. The cameras we currently field at Argonne National

Lab accelerator are Balser Aca640-120gm power over Ethernet CCD cameras. The hardware for the IOC consists of a local PC running Windows operating system, which is used to control the camera and publish data and control variables to the network.

The current version of the front panel software is written in LabVIEW and controls camera functions such as exposure time, trigger mode, intensity gain, etc. In addition the software performs post processing functions that finds the beam center and calculates its distance from the center of the target window. The software also measures the vertical and horizontal full width half max beam size either automatically using the beam center as or reference or from a user drawn region of interest. The current version of this software publishes raw image data to the network. The next iteration will read camera control variables from the network to allow for front panel control of the camera to be conducted from any location on the network.

Infrared Camera Temperature Monitoring

We use an IR camera to monitor the target window temperature during operation. The IR temperature monitor is intended to be used as a safety interlock to prevent the target window from failing due to overheating. The camera we currently use is a FLIR A650sc camera with a micro-bolometer sensor. The IOC is run on a local PC similar to the OTR IOC and the camera is connected to the computer via Ethernet.

The front end software that controls the camera was developed in LabVIEW similar to the OTR software. The software lets users set the important IR camera parameters namely: emissivity, reflected temperature, ambient temperature, and external optics transmissivity. The temperature is measured from user drawn region of interests on the image. Multiple ROI's can be drawn and the emissivity can be set separately for each region. The current version published the image data to the network and future will improvements to the software will publish the control variables to allow the IR camera to me managed from any workstation.

Beam current monitor

The beam current monitor (BCM) system is used to monitor and record the electron beam current in the beam pipe with fast current transformers. The fast current transformers used at ANL are Bergoz Model # FCT-CF4"1/2-34.9-40-UHV -20:1 -LD- H. Two current transformers are currently employed at ANL, although several more may be installed for the two sided irradiation. The signal from these transformers is currently acquired with a NI-5122 oscilloscope card, which was installed in a PXI chassis and recorded with a NI 8186 embedded controller. The data acquisition is currently controlled through a LabView virtual instrument (VI) running on the embedded controller. This system is not currently networked, and data from this system is stored directly to the disk drive in the embedded controller.

The BCM system worked well for the experiments at ANL because we already had the equipment and we could develop the system quickly with minimal effort. Expanding this system to acquire data from more than two transformers is the subject of ongoing development and will use a completely different architectural model.

The limits of the present BCM system and some future improvements are as follows. The embedded controller currently in use is over 10 years old. It uses a single core Pentium processor and has Windows 95 as an operating system. These PXI embedded processors cost on the order of $5k, and may not be warranted for this application where a small computer costing less than half as much would probably be sufficient. Even more cost effective would be replacing this controller with an embedded processor costing less than $100. This is the subject of an ongoing research effort at LANL. This embedded processor serves as the communications link between the computer network and the hardware for recording the fast-current transformer signals, and also recording the data locally, making the entire BCM system an IOC as described in the previous sections. The NI-5122 oscilloscope cards currently used to digitize the fast-current transformer signals, cost on the order of $10k for two channels, and are complete overkill for this application. We are currently exploring other options, such as using semi-custom analog to digital (ADC) converter circuit boards to digitize this data. These circuit boards will reduce the cost to several hundred dollars per channel. This is also the subject of ongoing research at LANL.

Beam Position Monitor

The Beam Position Monitor (BPM) system is a system which will be used to measure the position of the electron beam in the beam pipe using 4 antennas arranged on a flange in the beam pipe. This system will be used to aid the operator in steering the beam through the center of the beam pipe without using intercepting diagnostics, such as a phosphor screen. Since this diagnostic is nonintercepting, it can be used during full power operation to monitor and control the beam position during production. This will be very useful for steering the beam in areas far upstream of the target where the OTR diagnostic can only give limited information. This system is being implemented at ANL using antennas designed at LANL and fast signal amplifiers from Bergoz, and is the subject of ongoing research.

Conclusions and Lessons Learned

Server Hard Drive

After a scheduled power outage on June 15th 2014, during which ARCHSERV was left in service and, even though it was plugged into a UPS, the computer server went down for the duration of the outage and developed bad sectors on its hard drive after have been restarted. The hard drive was removed, cloned, and replaced. The original hard drive was a Seagate Barracuda, 500 GB, 7200 RPM, ST3500630AS SATA drive. This was replaced with a Western Digital Caviar Green, 1000 GB, 5400 RPM (this was actually sold as a 7200 RPM but subsequent research states that it is only 5400 RPM), WD10EARS-00Y5B1 SATA drive.

Immediately after installing the hard drive there were problems and issues. It was very hard to even log on to the machine because it was so busy. It turns out that it was spending most of the CPU time on IOWAIT caused by the journaling. it was thought at first that the journaling system had to finish “formatting the disk”, things did appear to get better, slightly, every day, until Saturday, June 21st, and especially Sunday, June 22nd. Nagios started sending many messages about the server having problems. It was also observed that the IOWAIT percentage was very high. Some of the boot parameters for the HOME partition were modified to “rw, suid, dev exec, auto nouser, async, noatime, nodiratime, relatime, nobarrier”. We then attempted to reboot the computer, and it never came back on its own.

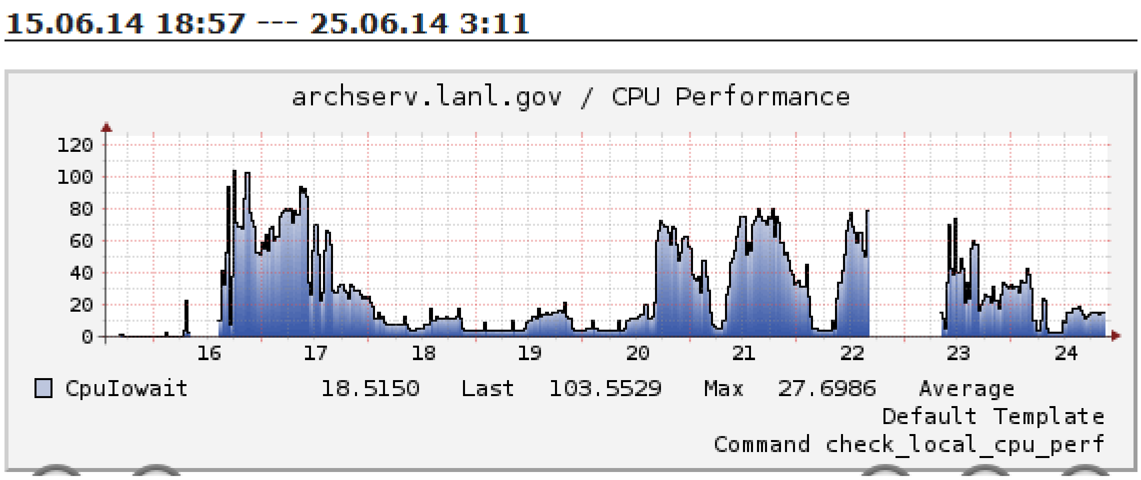

Figure 5 - Plot of the percentage of IOWAIT between June 15th and June 24th. Note that on the 15th was the power outage, and the server did go down (even though it was on a UPS). The attempted reboot on the 22nd should be obvious. Also shown is the way in which the IOWAIT storm seemed to subside and then come back again.

On June 23rd we restarted the server in the morning. The state of the CPU waiting on IO was pretty noisy and high usage at first, but seemed to slow down after a while, but by the end of the week the CPU was spending way too much time waiting on IO. Monday morning, June 30th, we replaced the SATA drive cable and restarted the server. Things quickly mellowed out. After watching the server for several days it appeared that the behavior was much improved, but certainly not the same as was before the hard drive replacement. On Wednesday, July 2nd, about 10:00 AM, we changed the boot parameters for the HOME partition back to “defaults” and rebooted the system. A plot of the IOWAIT data is slightly noisier now, but still not bad like it was before the drive cable replacement.

Our conclusion is that first, there was a bad cable, or at least a bad connection. Second, the reason the plotted data does not resemble the data prior to the drive replacement is that this new drive that was put in is about 3 to 10 times slower than the original drive. Using HDPARM to “benchmark” the hard drive, the new drive shows numbers like:

```
/dev/sda:
 Timing cached reads:   1584 MB in  2.00 seconds = 791.95 MB/sec
 Timing buffered disk reads: 118 MB in  3.02 seconds =  39.12 MB/sec
```

While DAO (for example) shows numbers like:

/dev/sda:

Timing cached reads: 14152 MB in 2.00 seconds = 7083.18 MB/sec

Timing buffered disk reads: 440 MB in 3.01 seconds = 146.11 MB/sec

22.06.14 23:10 --- 07.07.14 22:33

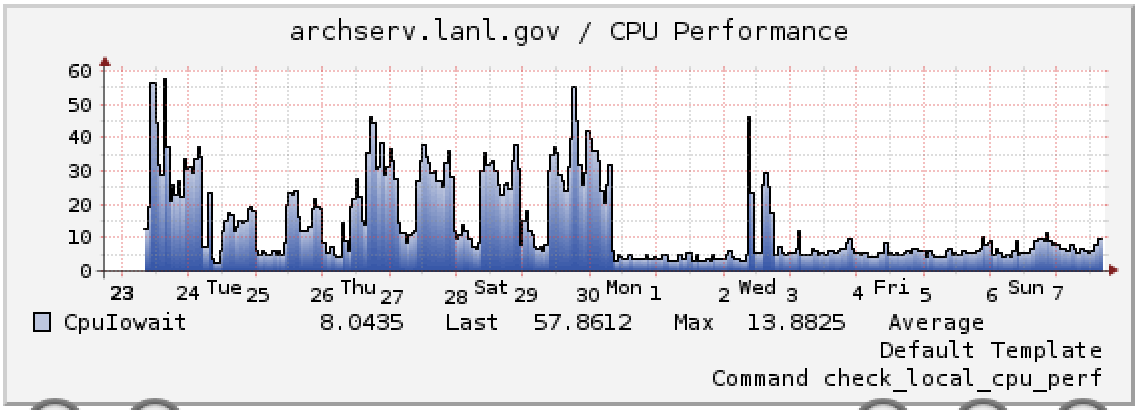

Figure 6 - Plot of the percentage of IOWAIT between June 22nd and July 7th. Note that on the 22nd the server was restarted after failing the remote reboot. Also shown is the IOWAIT behavior after replacing the drive cable and restarting on June 30th. On Wednesday, July 2nd, a YUM update was performed, and then some additional directories, '/usr' and '/boot', were added to the snapshot configuration such that the first snapshot took an inordinate amount of time.

On September 2, 2014 the current hard drive was removed, since the speed at which it operates was well below average, and replaced with a 500 GB Western Digital Black. After the drive replacement the behavior of the server returned to normal.

06.06.14 17:23 --- 08.07.14 17:23

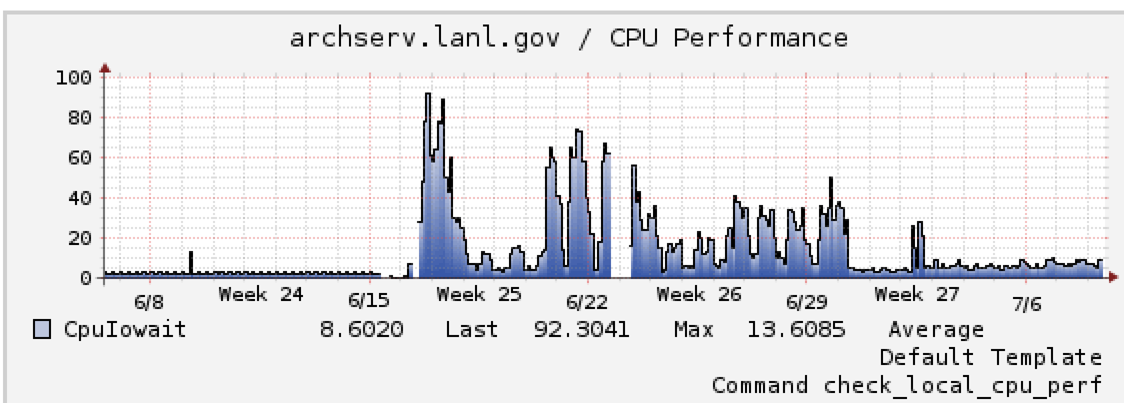

Figure 7 - The long term (1 month) behavioral profile for the IOWAIT portion of CPU resource.

The benchmark results for the new disk drive are as follows:

/dev/sda:

Timing cached reads: 1566 MB in 2.00 seconds = 783.09 MB/sec

Timing buffered disk reads: 316 MB in 3.01 seconds = 105.07 MB/sec
And the new Nagios CpuIowait plot is shown in Figure 8.

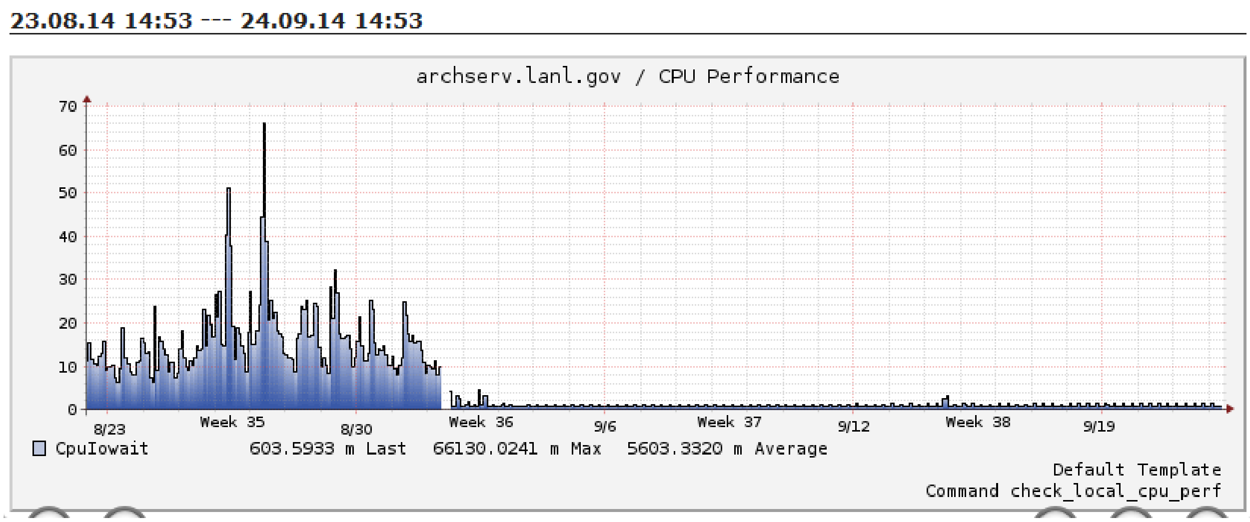

Figure 8 - A plot of the CpuIowait profile before and after the change from the slow, problematic disk drive to the new, "WD Black" drive.

This demonstrated that the archive server's performance is very dependent on the speed of the hard drive used. This exercise also demonstrated the helpfulness of the Nagios software for monitoring the behavior of the control system components.