

## LA-UR-15-28925

Approved for public release; distribution is unlimited.

Title: Multiple Flow Loop SCADA System Implemented on the Production Prototype Loop

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

Intended for: Report

Issued: 2015-11-16

---

**Disclaimer:**

Los Alamos National Laboratory, an affirmative action/equal opportunity employer, is operated by the Los Alamos National Security, LLC for the National Nuclear Security Administration of the U.S. Department of Energy under contract DE-AC52-06NA25396. By approving this article, the publisher recognizes that the U.S. Government retains nonexclusive, royalty-free license to publish or reproduce the published form of this contribution, or to allow others to do so, for U.S. Government purposes. Los Alamos National Laboratory requests that the publisher identify this article as work performed under the auspices of the U.S. Department of Energy. Los Alamos National Laboratory strongly supports academic freedom and a researcher's right to publish; as an institution, however, the Laboratory does not endorse the viewpoint of a publication or guarantee its technical correctness.

# **Multiple Flow Loop SCADA System Implemented on the Production Prototype Loop**

Scott Baily, Dale Dalmas, Robert Wheat, Keith Woloshun, Gregory Dale

## **Overview**

The following report covers FY 15 activities to develop supervisory control and data acquisition (SCADA) system for the Northstar Moly99 production prototype gas flow loop. The goal of this effort is to expand the existing system to include a second flow loop with a larger production-sized blower. Besides testing the larger blower, this system will demonstrate the scalability of our solution to multiple flow loops.

## **EPICS Based Control System**

The system currently fielded in MPF-14 for control of the Target cooling system 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 original control system used on the flow visualization loop at LANL. For the new production prototype flow loop, we added an additional IOC (compact RIO) and an additional desktop control computer. It is convenient to have an additional control computer in the same room as the test equipment, but it would certainly be possible to monitor and control both flow loops from any of the computers.

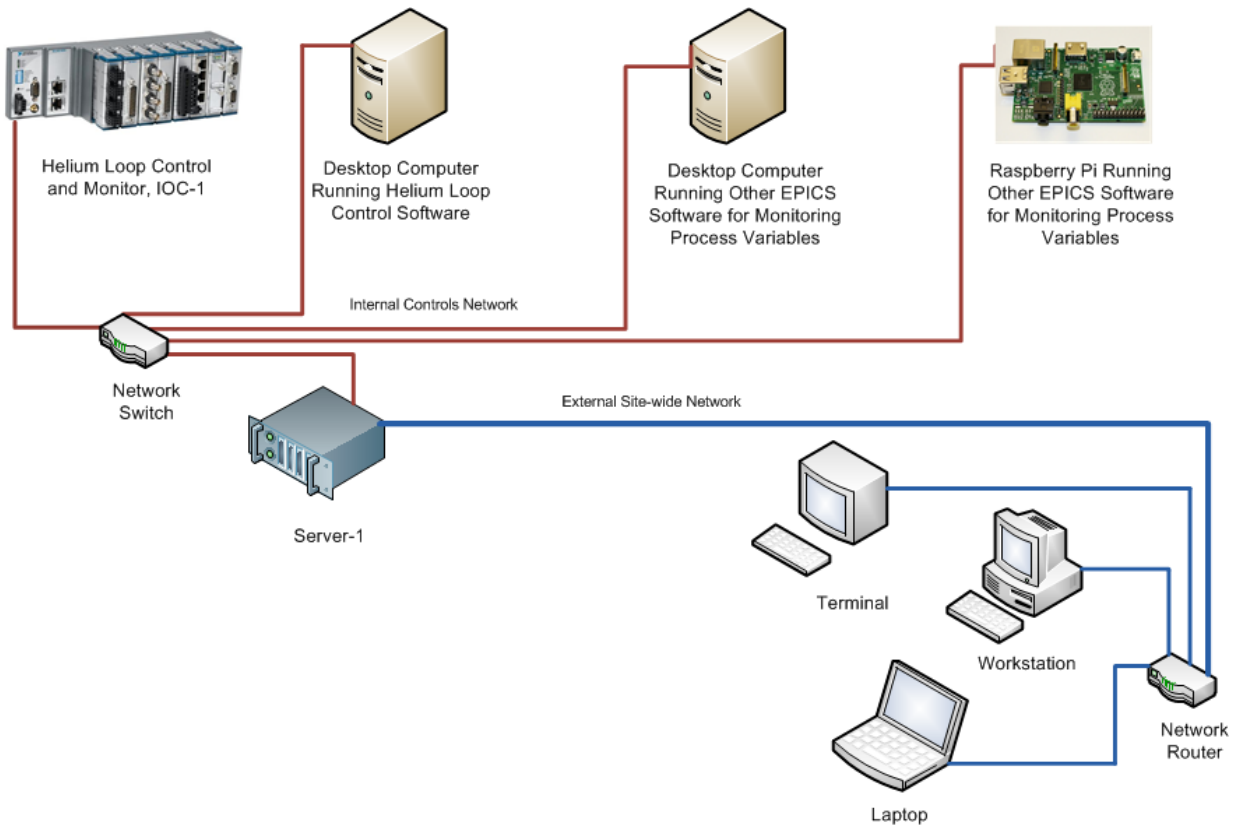


Figure 1 - A schematic of the current prototype production facility control system implementation.

## Hardware and Components

The currently implemented prototype facility control system consists of the server computer, two cRio systems (one for each flow loop) interfaced to the cooling loops controls and sensors, 3 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 system.

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.

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. Some of the most used command line programs are ‘caget’, ‘caput’, and ‘camonitor’.

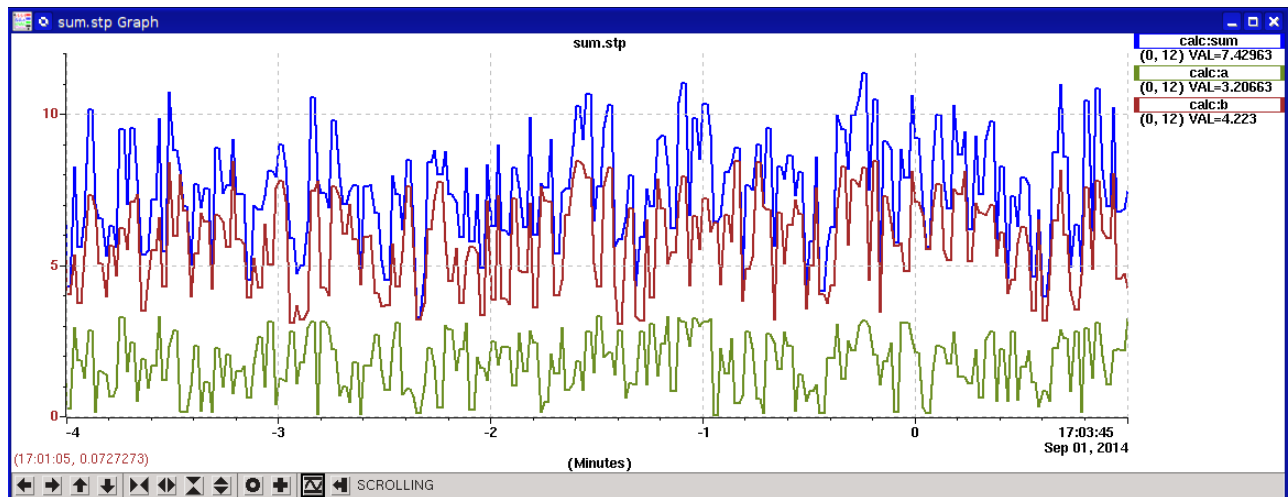


Figure 3 - A sample StripTool operator screen.

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.



Figure 4 - The Nagios service details web page display.

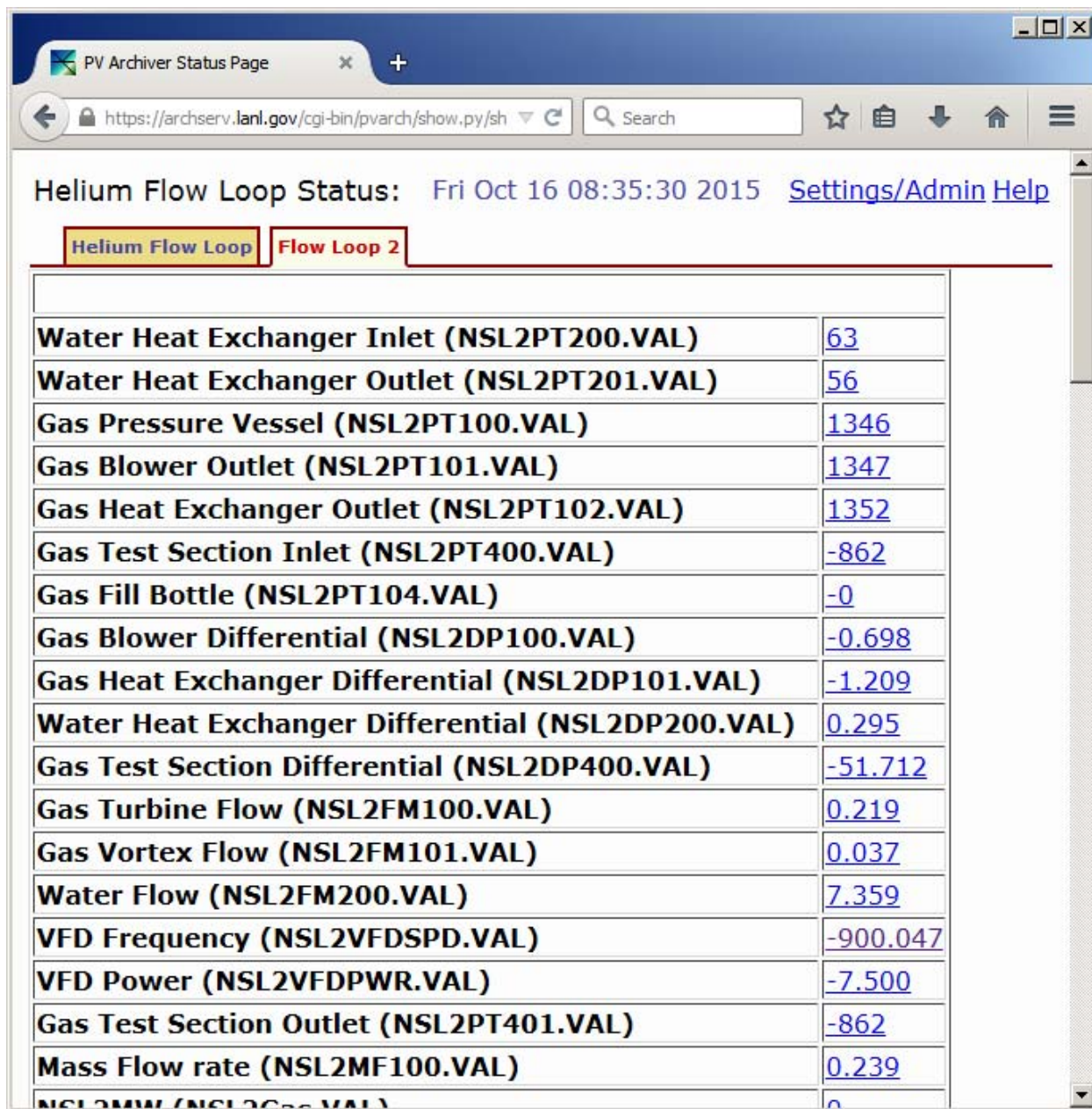
## Data Archiving

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. Figure 5 is a screen capture picture of the Archiver web interface, and Figure 6 is a screen capture picture of web interfacing for plotting the PV data over time. 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. Figure 7 is a screen capture picture of the Archiver Admin configuration interface.



The process variables for the new IOC were added to the archiver, but this time we included nearly every variable including binary I/O values. The archiver interface permits grouping of process variable by flow loop.



Helium Flow Loop Status: Fri Oct 16 08:35:30 2015 [Settings/Admin Help](#)

**Helium Flow Loop** **Flow Loop 2**

Water Heat Exchanger Inlet (NSL2PT200.VAL)	<a href="#">63</a>
Water Heat Exchanger Outlet (NSL2PT201.VAL)	<a href="#">56</a>
Gas Pressure Vessel (NSL2PT100.VAL)	<a href="#">1346</a>
Gas Blower Outlet (NSL2PT101.VAL)	<a href="#">1347</a>
Gas Heat Exchanger Outlet (NSL2PT102.VAL)	<a href="#">1352</a>
Gas Test Section Inlet (NSL2PT400.VAL)	<a href="#">-862</a>
Gas Fill Bottle (NSL2PT104.VAL)	<a href="#">-0</a>
Gas Blower Differential (NSL2DP100.VAL)	<a href="#">-0.698</a>
Gas Heat Exchanger Differential (NSL2DP101.VAL)	<a href="#">-1.209</a>
Water Heat Exchanger Differential (NSL2DP200.VAL)	<a href="#">0.295</a>
Gas Test Section Differential (NSL2DP400.VAL)	<a href="#">-51.712</a>
Gas Turbine Flow (NSL2FM100.VAL)	<a href="#">0.219</a>
Gas Vortex Flow (NSL2FM101.VAL)	<a href="#">0.037</a>
Water Flow (NSL2FM200.VAL)	<a href="#">7.359</a>
VFD Frequency (NSL2VF DSPD.VAL)	<a href="#">-900.047</a>
VFD Power (NSL2VF DPWR.VAL)	<a href="#">-7.500</a>
Gas Test Section Outlet (NSL2PT401.VAL)	<a href="#">-862</a>
Mass Flow rate (NSL2MF100.VAL)	<a href="#">0.239</a>
NSL2MM (NSL2Cg VAL)	<a href="#">0</a>

Figure 5 - The archiver interface.



## Helium Flow Loop Status:

Mon Oct 26 13:36:28 2015

[PV Status](#) [Instruments](#) [Alerts](#) [Settings / Admin](#) [Help](#)

PV 1

PV 2

**Related pvs:**

PV (Y) range:  :

:

[Swap PV1 and 2](#)

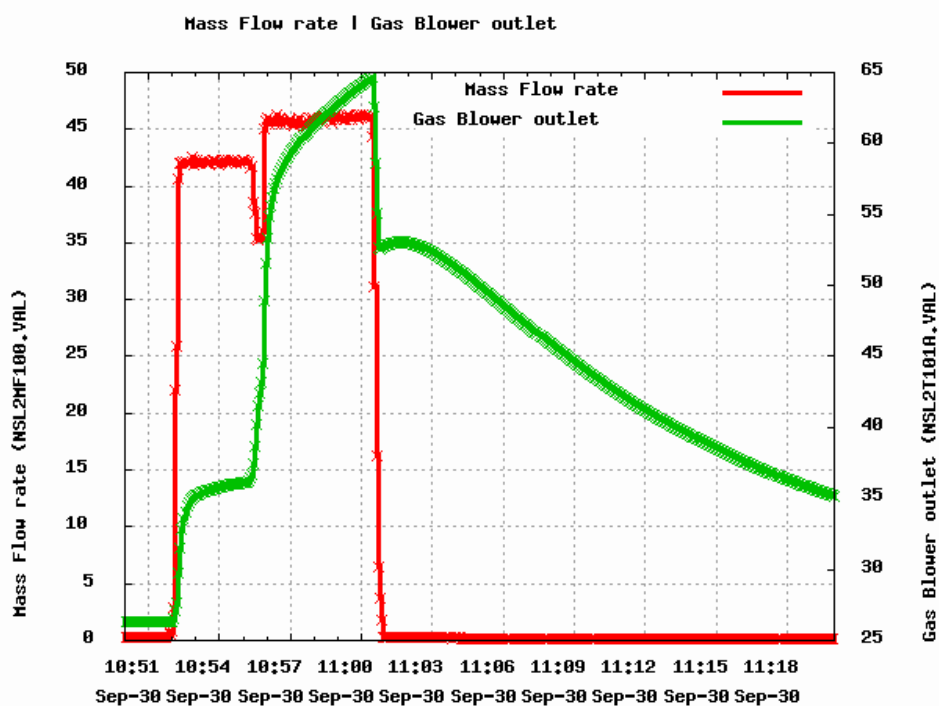
Log Scale? ☐ Yes ☐ No ☒ Auto

☐ Yes ☐ No ☒ Auto

[NSL2T101A.VAL](#)

Time From Present

Date Range From:  To:



[data for Mass Flow rate \(NSL2MF100.VAL\)](#)

[data for Gas Blower outlet \(NSL2T101A.VAL\)](#)

[gnuplot script](#)

Figure 6 - Plot of the gas mass flow rate and blower outlet temperature from the data archiver.

Helium Flow Loop Status:

Fri Oct 23 10:



**NSL2PT201.VAL**

[Show Plot](#)

---

Data Type	double
Actively Archived:	<input checked="" type="radio"/> Yes <input type="radio"/> No
Description	<input type="text" value="Water Heat Exchanger Outlet"/>
Deadtime (seconds)	<input type="text" value="5"/>
Deadband (fraction)	<input type="text" value="0.5"/>
Graph Upper Limit	<input type="text"/>
Graph Lower Limit	<input type="text"/>
Graph Type	<input checked="" type="radio"/> normal <input type="radio"/> log <input type="radio"/> discrete
<input type="button" value="Update PV Settings"/>	

---

Figure 7 - Archiver Admin interface

## 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 boxes, 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.

## 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. For the production prototype gas flow loop system, the IOC is a CRIO that monitors temperatures, pressures, blower speed, and some binary interlocks. It controls the blower through the variable frequency drive, and it controls

inputs to the interlock chassis based on calculations. Other IOCs in the production system would be those that control other flow loops, Beam Current Monitor IOCs, Beam Position Monitor IOCs, those that control the IR and OTR cameras, etc.

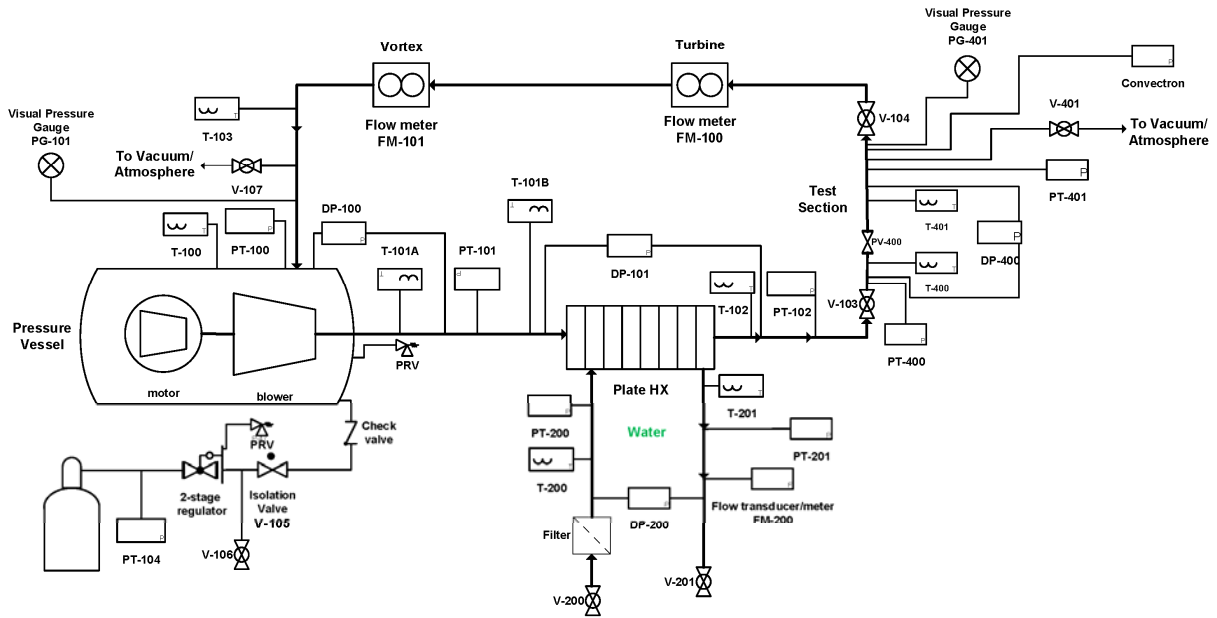
Adding additional IOC's can be as simple as plugging them into the controls network. If the new system is running the proper EPICS software, then the EPICS software on all other computers in the system will automatically see this new IOC and all of the associated Process Variables. If it is desired, 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 (via DNS).

## **Production Prototype Gas Flow Loop IOC**

### **Overview**

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 circulating system, the water cooling system, the test section, controls and sensors, the cooling system Input Output Controller (IOC), the system server, and a "user" computer used to run EPICS control and monitoring software as user interfaces to the system. This second system currently fielded in MPF-14 at LANL for control of the production prototype gas flow loop incorporates a larger production-sized blower. Figure 8 shows a schematic of the flow loop. While the first flow loop system used RTD temperature sensors, we have chosen to test the use of thermocouples for the second system, this makes it less expensive to incorporate a larger number of sensors, as long as long cable runs of thermocouple wire are avoided.

**Bldg 14 Gas Flow Loop II**  
7/8/15



**Figure 8 - Production Prototype Gas Flow Loop Schematic**

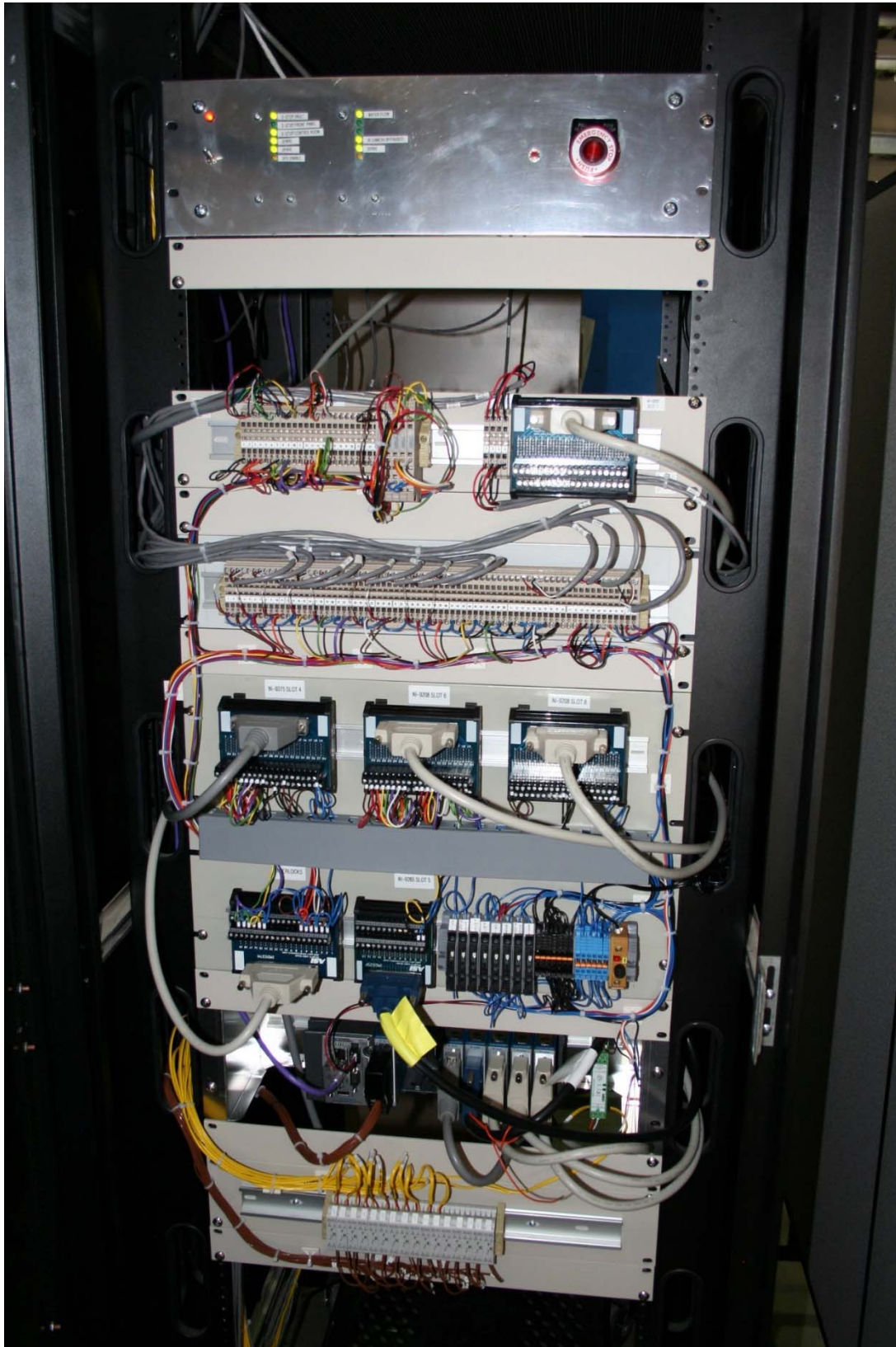


Figure 9 - Control system and wiring for the second system



## Hardware

The hardware for the 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 4 c-series modules installed: one 9213 thermocouple module, one 9375 binary I/O module, one 9265 module, and one 9208 current input module. The software is configured to accommodate the addition of two 9217 RTD sensor modules, and an additional current input module. We also plan to add a 9205 (32 channel multiplexed 16-bit ADC) Voltage input module to measure accelerometer sensors. This system, as currently configured, is capable of measuring 16 thermocouple sensors ( $\pm 78$  mV with 24-bit resolution), reading 16 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 4 current outputs (0 to 20 mA, 16-bit resolution). Figure 9 shows the rack with the wiring and IOC. Figure 10 is a close-up image of the IOC.



Figure 10 - Production Prototype Gas Flow Loop IOC

## Sensors

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 as in Figure 11. 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. For now we're using

only type-K thermocouple sensors (the thermocouple module has an internal thermistor for junction temperature compensation) as the cable lengths are short for the test setup.

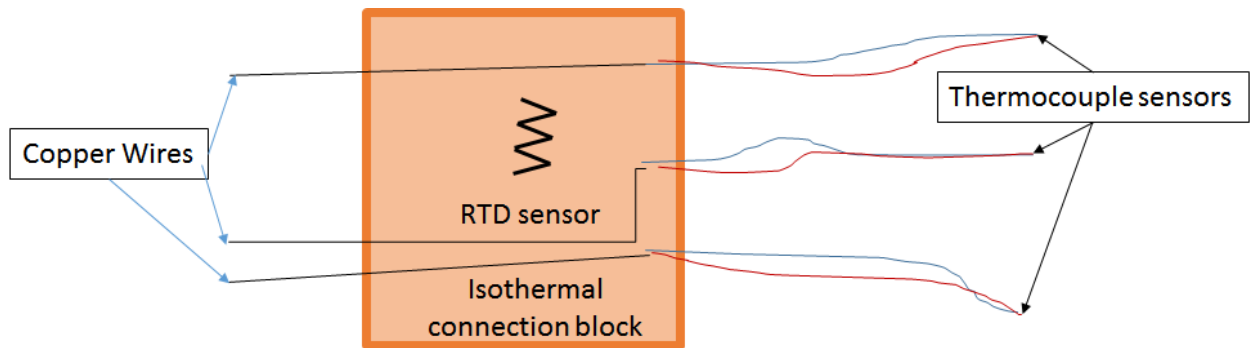


Figure 11 - Method to avoid long runs of thermocouple wire

## 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, we chose caQTDM. This is a recently developed operator interface for EPICS, with an active development community. 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.

The previously existing control screens developed for the Target Loop Cooling System were used for the new user interface. While some minor improvements were made, the only significant change was the addition of macros to select which flow loop is controlled/viewed. This permits us to maintain only a single set of control screens. Figure 12 shows the main control screen. In the few cases where sensor locations differ, aliases (T101 and T101A are in fact the same process variable) permit a compatible view.



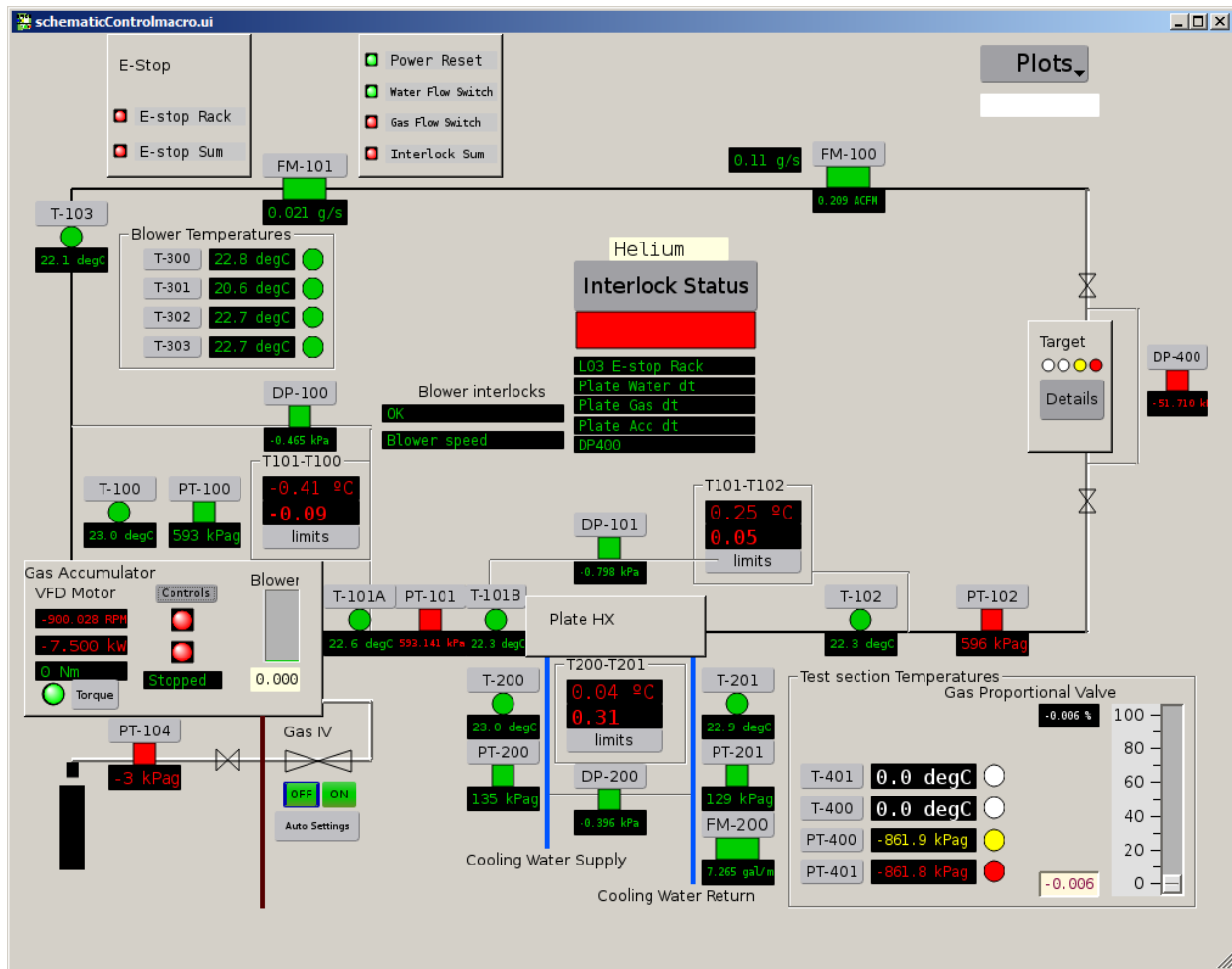


Figure 12 - Main Flow loop interface.

Since the test system blower is run in ambient air, 100 psi nitrogen, and full pressure helium, we have implemented three modes depending on the gas selected. Manually saved alarm limits are stored separately for each mode, and the conversion from volume to mass flow is based on the type of gas selected.

## 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. The cRIO IOCs check the contents of the server via NFS at boot time, and will update themselves to incorporate database changes automatically. The database for the new IOC has been updated to use metric units for pressure.

## Channels

EPICS has four alarm limits for analog data LOLO, LOW, HIGH, and HIHI. In the typical configuration values above HIHI or below LOLO result in major alarm status, values between LOW and HIGH are no alarm status, and intermediate values result in minor alarm (i.e., warning) status. Invalid status is used by device support to indicate hardware errors or thermocouple voltages that are outside the valid range for that sensor type. Table 1 shows all hardware and interlock channels with the current alarm limits.

Table 1 - Channel information

Channel	Description	LOLO	LOW	HIGH	HIHI	Units	Blower	"Beam"
L01	Power Reset							x
L03	E-stop Rack							x
L06	E-stop Sum							x
L07	Water Flow Switch							x
L08	Gas Flow Switch							x
L11	Interlock Sum							x
VFDMR	VFD Motor Ready							
VFDMR	VFD Motor Running							
Accdt	Gas Accumulator delta temperature (calc)	0	0	0	0	°C		x
Accpw	Gas Accumulator power (calc)	0	0	0	0			x
DP100	Gas Blower Differential pressure	-34.5	-28	172	207	kPa	x	x
DP101	Gas Heat Exchanger Differential pressure	-13.8	-6.9	172	207	kPa		x
DP200	Gas Heat Exchanger Differential pressure	-14	-6.9	172	207	kPa	x	x
DP400	Gas Test Section Differential pressure	0	0	0	0	kPa		x
FM100	Gas Turbine Flow	5	10	0	0	ACFM		x
FM101	Gas Vortex Flow	0	0	0	0	g/s		x
FM200	Water Flow	4	6	20	20	gal/m	x	x
INTRLKTRIP	Manual Interlock							x
PlateHedt	Plate HX Gas delta temperature (calc)	0	0	0	0	°C		x
PlateHepw	Plate HX Gas power (calc)	0	0	0	0			x
PlateWdt	Plate HX Water delta temperature (calc)	0	0	0	0	°C		x
PlateWpw	Plate HX Water power (calc)	0	0	0	0			x
PT100	Gas Pressure Vessel pressure	-335	-310	1971	2034	kPag		x
PT101	Gas Blower Outlet pressure	1379	1724	3454	3461	kPag		x
PT102	Gas Heat Exchanger Outlet pressure	1379	1724	3454	3461	kPag		x
PT104	Gas Fill Bottle pressure	2034	3447	20677	20684	kPag		x
PT200	Water Heat Exchanger Inlet pressure	69	89.6	620	689	kPag		x
PT201	Water Heat Exchanger Outlet pressure	69	89.6	620	689	kPag		x
PT400	Gas Test Section Inlet pressure	-965	0	0	0	kPag	x	
PT401	Gas Test Section Outlet pressure	0	0	0	0	kPag		
T100	Gas Pressure Vessel temperature	-2	-1	35	75	°C	x	x
T101A (T101)	Gas Blower Outlet temperature	-2	-1	100	150	°C	x	x
T101B	Gas Heat Exchanger inlet temperature	-2	-1	100	150	°C		
T102	Gas Heat Exchanger outlet temperature	-2	-1	30	50	°C		x
T103	Gas Blower Return temperature	-2	-1	30	50	°C	x	
T200	Water Inlet temperature	0	5	30	30	°C	x	x
T201	Water Outlet temperature	0	5	30	30	°C	x	x
T300	Blower far side temperature	-2	-1	60	80	°C	x	x
T301	Blower bearing side temperature	-2	-1	60	80	°C	x	x
T302	Motor bearing side temperature	-2	-1	60	80	°C	x	x
T303	Motor far side temperature	-2	-1	60	80	°C	x	x
Torque	Blower Torque (calculated)	-40	-40	45	50	Nm	x	x
VFDPWR	Blower power	-2	-1	28	30	kW	x	x
VFDSPD	Blower frequency	-15	-12	3500	3600		x	
BlowerSPD	Blower Speed					RPM		
PV400	Gas Proportional Valve					%		
GasIV	Gas IV							
CCIntrlk	Computer Control Interlock							x

## Interlocks

Any safety interlocks must be implemented in hardware, however it is useful to have additional interlocks implemented in software. If we decide on fixed limits, some of these software interlocks may later be implemented in the FPGA firmware. Software interlock channels are noted in Table 1.

## Hardware

Critical interlocks are performed in hardware with electromechanical relays. These electromechanical relays are contained in the interlock chassis, shown in Figures 13-15. This relay chassis follows the design of the interlock chassis designed for the LANL flow loop at ANL. The major difference is that the new system uses relays mounted on a custom designed PC board, whereas the system at ANL uses wired relays and terminal blocks. Now that the design of the interlock chassis is generally fixed, the use of a custom PC board greatly reduces the time necessary to wire and troubleshoot the interlock chassis.

Schematics and PC board layouts are contained in Appendix A. In general, the interlock chain consists of several relays in series, each of which must be actuated for the end function to be enabled. On a second set of contacts, a signal is sent to the cRio for monitoring the interlock status. A third set of contacts is used to send status signals to LEDs on the front panel of the interlock chassis.

For the flow loop, the end function is the blower enable signal that goes to the blower motor's VFD and enables the motor to be turned on. Each of the interlocks must be made up for the blower to turn on.

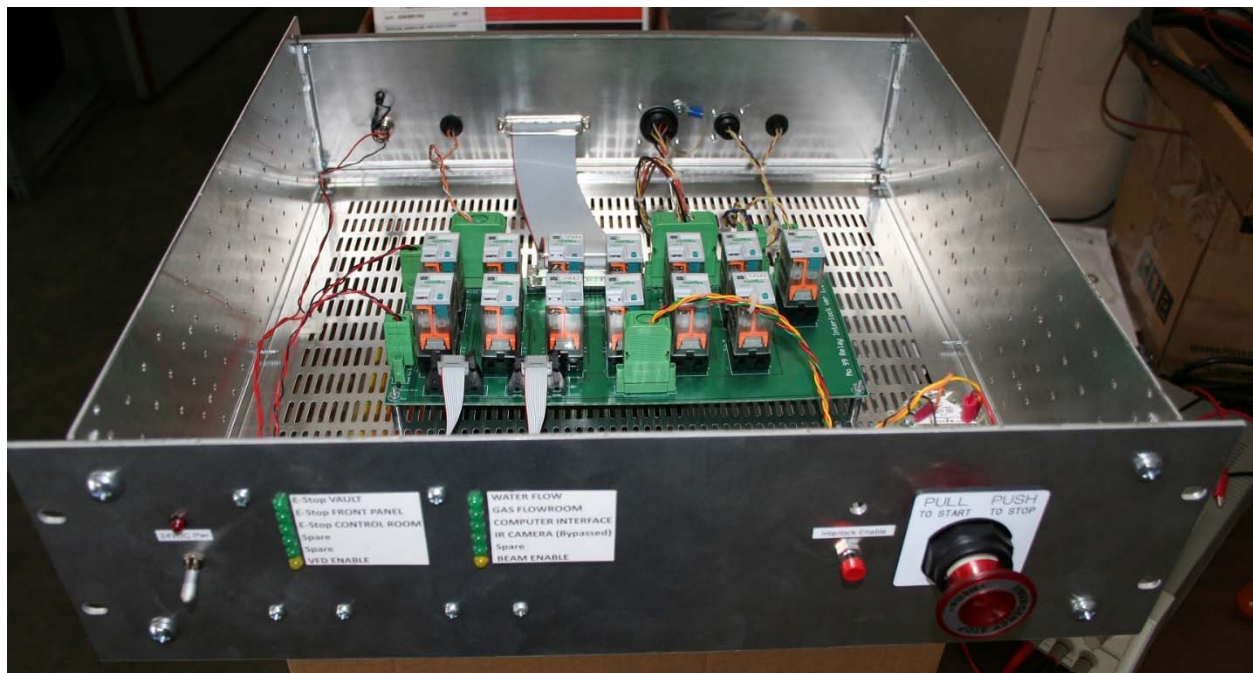


Figure 13 - Interlock Chassis front view



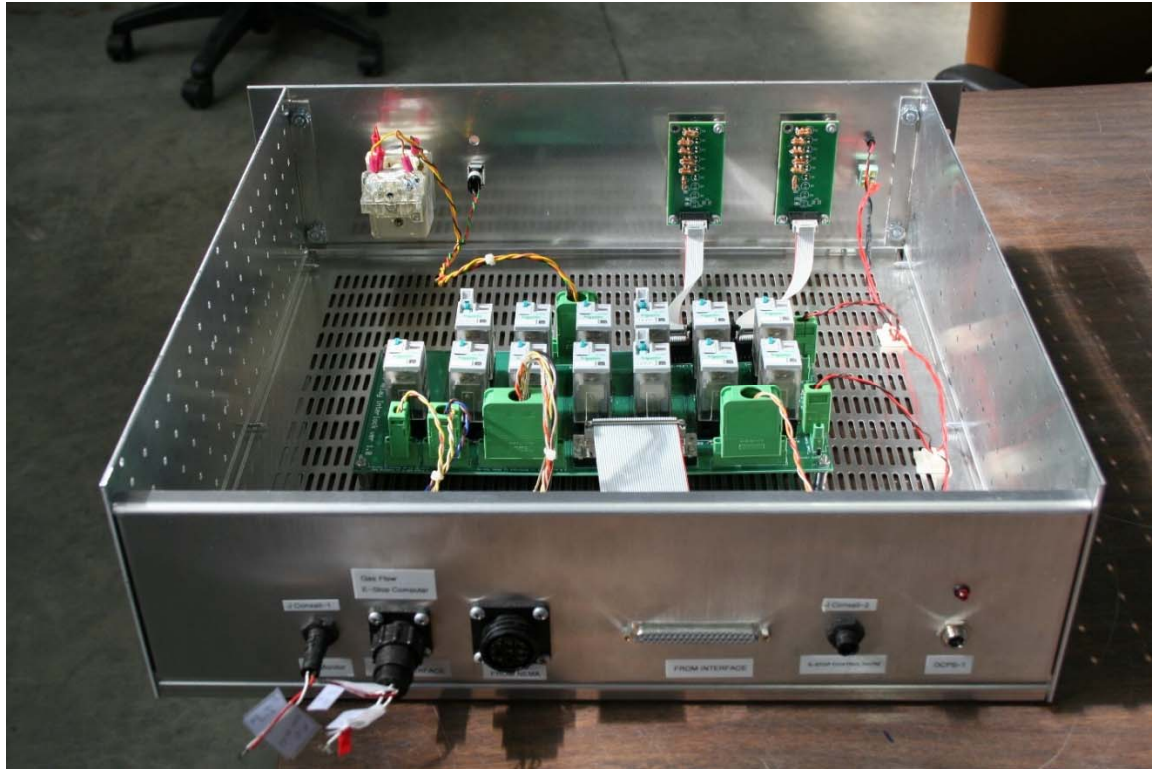


Figure 14. Interlock chassis back view.



Figure 15. Interlock chassis top view showing the PC board with relays.

For the production prototype flow system, the hardware interlocks consist of two emergency stop buttons. One is located on the relay chassis, and another is located next to the control computer. Each of these e-stop buttons must be pulled out for the blower to operate. If one of these buttons is depressed, it will turn off the blower. Since the interlock chassis is a duplicate of the one used at ANL, there is provision for a third e-stop that would be wired to the accelerator control room. This is currently jumpered such that the relay is always enabled. There are also two spare relays that are also jumpered enabled.

A second feature duplicated from the interlock chassis used at ANL is a second string of interlocks that are used to interface with the accelerator. For this second set of interlocks to be made up, several additional conditions must be satisfied. For the accelerator beam interlock to be enabled, all blower interlocks must be enabled, the heat exchanger water flow switch must be actuated, the gas flow limit switch from the mass flow meter must be actuated, and the computer interlock must be enabled. In addition, there is provision for an interlock signal from the IR camera system and one spare. This interlock chain is currently not functional in the production prototype system since there is no accelerator to interface with, but this capability has been maintained in this prototype system since these interlocks will be necessary for the actual production system.

## Software

Analog interlocks are based on the user-configurable alarm limits (Table 1). If a channel goes into major alarm status or invalid status, it will trip the corresponding interlock, minor alarm status is only used to warn the operators that something is outside of the expected range, or may be approaching a trip point. Since many equipment protection interlocks rely on software, it is important for the interlock chassis to know that the software is continuing to monitor the equipment. We have configured one input to the interlock chassis from the IOC to indicate the state of health of the IOC. Everytime the complete set of software interlock conditions is checked, a cRIO's FPGA is told to send a high signal on the state of health output. If this command is not received after a certain amount of time (currently set to 15 seconds), then the FPGA firmware will set the signal low. Thus the interlock will drop out if the software is stopped, frozen, or running much slower than expected.

## Appendix A – Interlock Chassis

Additional Interlock Chassis Details:





