6 CONTROL SYSTEM

6.1 Introduction and Scope

The control system for NSLS-II is designed to convey all monitor, control, model-based, and computed data from all accelerator, facility, experimental, safety, and operations subsystems to accomplish supervisory control, automation, and operational analysis. The scope of the control system extends from the interface of the equipment being controlled through to the designers and operators of the accelerator facility, as well as synchrotron beamline experimenters and staff. The control system includes all hardware and software for global systems such as timing, deterministic data communication, network communication, control room operations, automation and optimization. The control system includes the computers and software required to implement and integrate all subsystems including: diagnostics, power supply control, low level RF, vacuum, personnel protection, equipment protection, undulator, experimental beamlines, and conventional facilities

To provide this comprehensive monitoring, control, and automation, the NSLS-II control system must scale to support 100,000 physical I/O connections and 350,000 computed variables that can be correlated to analyze events and provide data for all control aspects. It must support 1 Hz model-based control, 110 kHz power supply digitization, 500 MHz RF control, 5 KHz orbit feedback, and 20 millisecond equipment protection mitigation. It also must provide 5 Hz updates to operators of up to 1,000 chosen parameters, provide coherent turn-by-turn orbit data for up to 2^{10} = 1,024 consecutive turns (for FFT), archive up to 6,000 parameters at a rate of 0.5 Hz continually, latch the last 10 seconds of data from all parameters in the storage ring when a fault is detected in the Machine Protection System (MPS), archive up to 1,024 consecutive turn by turn data for 1,000 parameters at a rate of 10 Hz, and provide pulse-to-pulse beam steering in the linac at 1 Hz.

Our proposed client-server architecture is depicted in Figure 6.1.1. Different levels of access and control reside at distinct layers. At the highest layer (layer 3), access is provided for activities that do not involve moment-by-moment control or monitoring of the accelerator. Layer 3 includes high level physics modeling, making use of live data and data stored in the site Relational Database (RDB in the figure). Experimental activities that do not require synchronization with the ring also reside at layer 3. Layer 2 contains accelerator operation and monitoring activities. Layer 1 contains dedicated equipment controllers, which in turn interface to specific equipment through point-to-point protocols (event system, deterministic data communication hardware for Fast Orbit Feedback. Layer 0 contains remote, slow, independent, reliable control that requires synchronization with the accelerator no faster than 2 Hz such as facility control, vacuum control, or Personnel Protection.

Communication between subsystems takes place via four distinct buses as indicated. Fast Feedback, MPS, and Global Synchronization buses supply information as implied for the needs in a deterministic and reliable fashion of these control operations. Asynchronous information flow which does not require specific transfer rates is achieved by Channel Access Protocol. This is the most global communication standard in the system and, accordingly, most devices in every layer are identified as channel access clients, servers, or both.

The standard two-layer client server architecture ensures scalability and avoids performance limitations. NSLS-II controls must be built upon a proven tool kit with well-defined interfaces at both the server and client to enable integration and development. It should enable the use of hardware and software already developed for specific light source requirements. The core of the Experimental Physics and Industrial Control System (EPICS) has been chosen as the basis for the control system. The advantages in three key areas drove this decision: large user base in the accelerator community, functionality for accelerator-related systems, and support for the required hardware interfaces.

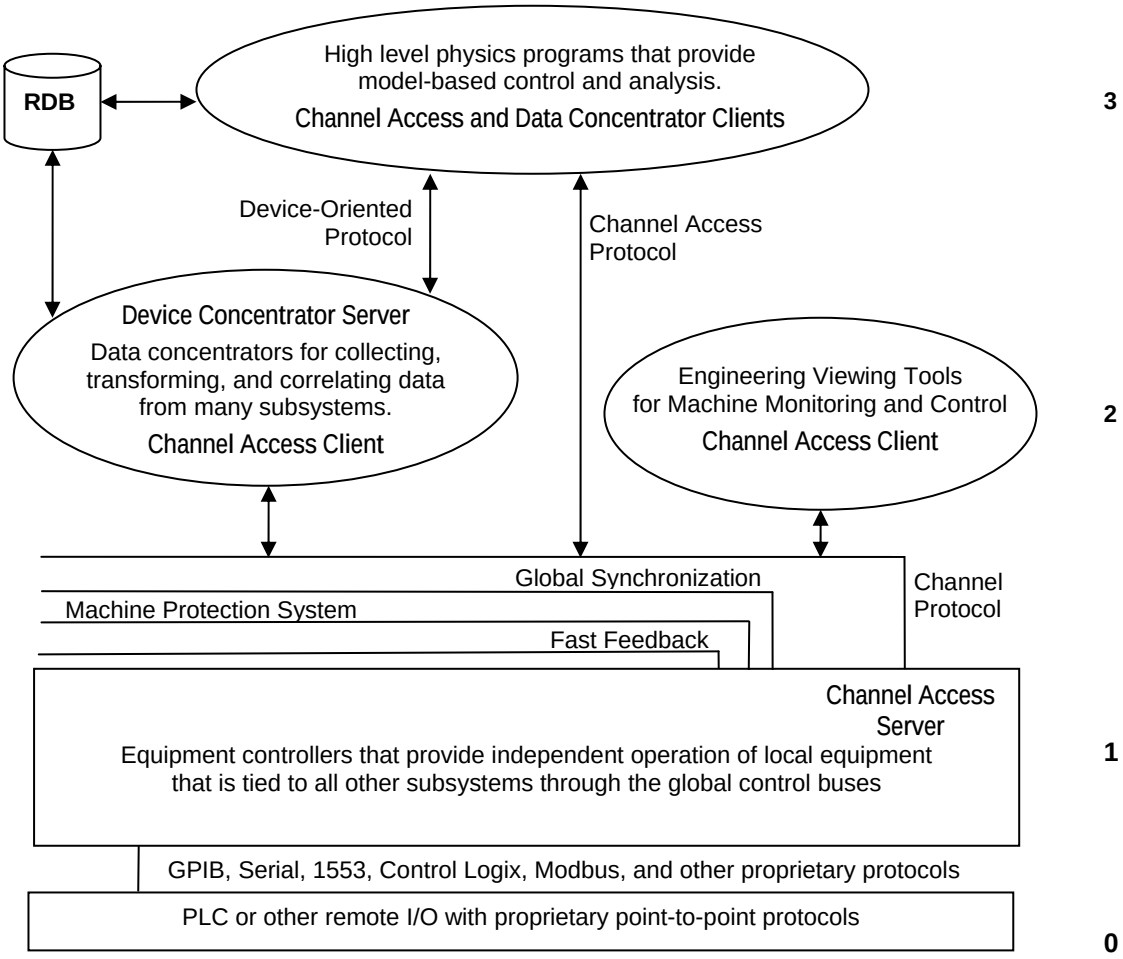

Figure 6.1.1 NSLS-II software architecture.

6.2 Control System Requirements

6.2.1 Technical Requirements

The control system must be modular, incrementally upgradeable, scaleable, and extendable. Expansion of the control system to accommodate the build-up of the accelerator and beamlines from early testing, through installation and commissioning, and during the life of the facility, should not impact the performance. The control system must be available to support all aspects of the project schedule, from component tests during prototyping to beam characterization and optimization at commissioning. To achieve this, the NSLS-II control system is based on open standards and commercial off-the-shelf equipment, whenever possible. Developments needed by NSLS-II are to be accomplished in a manner that meets the project schedule, budget, and performance needs, with consideration for knowledge transfer to the wider accelerator community.

Machine Control

Machine control supports linac control that is synchronized to the master timing system to fill the booster ring with 80 nanosecond pulse trains made up of 40 micro bunches of 2 ns length each. Pulse-to-pulse timing jitter will be less than 80 ps at the 1 Hz rate. The pulse is ramped up to the correct energy in the booster ring over 400 ms. It is then injected into the storage ring. The revolution rate for the booster and storage ring is 2.6 μs. Manual control of orbit trims, quadrupoles, sextupoles, and insertion devices is asynchronous. These are controlled by the operators, accelerator physicists, high level applications, or users. In particular, ~10 Hz write/read is suitable for "turning knobs" for a power supply. The only "fast" process is the fast orbit feedback system with 5 KHz bandwidth (and feedback systems for coherent bunch instabilities of order MHz). To summarize, the beam bunches in the ring and the injection process are synchronous to the RF, but everything else has its own time scales. Model-based control is used to correct steering, the orbit, tune, linear chromaticity, optics, etc. in the storage ring at 1 Hz.

System Reliability

The control system must have 99.99% availability, 24 hours per day, 365 days per year. Control system modifications that add new functionality will be performed during scheduled down times. New functionality for the operational facility will be tested on equipment test stands before installation. The cryogenic control must achieve even higher standards. Failures or modifications to the cryogenic control system must not result in the loss of temperature control for greater than 15 minutes. Subsystems will be designed to meet system reliability goals using high reliability technology, where required. This includes the use of an uninterruptible power supply, programmable logic controllers, battery backup, and redundant power supplies for VME crates. All subsystems will be designed to achieve operational reliability goals.

Security and Integration across Operator Base

The system must manage access requirements for the different classes of user. It must also incorporate appropriate tools to guarantee security of its computers and network systems. The source of the data should be irrelevant from the point of view of any software designer or user. For example, it should be possible to display data from the control system, from the associated relational database and from an accelerator model on one full-screen synoptic.

6.2.2 Software Requirements

Control system applications must be designed to enable future upgrades to be incorporated economically. Well defined interfaces to support the modular upgrade/replacement of code is a key component for this requirement. The EPICS architecture provides these interfaces at all levels implemented in EPICS. The software used to implement Fast Orbit Feedback (FPGA code) and the software used to implement High Level Applications will attempt to accomplish this same level of modularity.

Code Maintenance

All code, control system tools, and applications will be placed under source/release control. A standard tool will be used to control the version of software running and to keep previous versions and an audit trail for changes to released and commissioned software. Accelerator components and signal lists such as: magnetic lengths, min/max currents, calibration coefficients for currents vs. gradients, diagnostics channels, and configuration parameters also will be kept and their versions managed. The data that are also needed by accelerator models are to be kept in a similarly controlled relational database.

Open Standards, Integration, and Ease of Use

The control system will use open standards and an open architecture. The long life expectancy of an accelerator complex implies that the control system will need to evolve to incorporate upgrades and new technology. The control system must enable seamless integration of systems at both the server and client side through well-defined APIs. It is beneficial for the applications that comprise the NSLS-II control system to have a consistent look and feel. Related functions should be linked, to reduce the number of mouse clicks a user has to perform. For example, trends could be accessed by clicking on a process value hotspot displayed on a plant synoptic value. All control system facilities that need to be accessed directly will be accessible via menus, where the menu titles give a clear indication of the facility being called up. It should never be necessary for a user to remember the name of a program or of data files in order to use the system.

Context-sensitive online help facilities should be designed into the system, where feasible. Conscientious attention to common-sense ergonomics during application development will clearly pay dividends for long-term ease of use and will minimize familiarization and training costs for new operators. Production of a concise Style Guide document at an early stage in the development cycle of the project will provide a good ethos for addressing these issues.

6.2.3 Architecture Requirements

The four-layer EPICS-based client-server architecture illustrated in Figure 6.1.1 implies further design considerations for its implementation:

Network

The connection of the control system layers will use standard network components. These should be in a redundant configuration and include provision for network evolution, i.e., the development to the latest network standards. Control system network security is to include physical security that limits access to the control network from outside, using gateways and firewalls. It requires appropriate types of password and key protection within the control network with access control to manage who is controlling which operation.

Operator Interface

The operator interface will be either workstations or PCs running Linux. The control system should seamlessly integrate with office systems through a gateway process to maintain security.

Equipment Interface

The equipment interface will provide the physical connection to the equipment being controlled through a variety of interfaces. The preferred standards will include VME because of physical and electrical performance, Compact PCI where higher performance backplanes or lower point count make this more cost effective, and PLC I/O for applications where equipment safety is required and speed is not. The control system includes all VME crates and processors, any network hardware required for integrating instrumentation, the timing/event system, all hardware used for fast feedback, and all the crates and processors used to integrate the I/O. Except where noted, the intelligent device controllers, I/O, and PLCs are provided by the subsystem. The notable exceptions are the controllers for the undulator equipment and the non-BPM diagnostics in the Storage Ring. The network cables and cables to implement the global buses are the responsibility fo the control system.

Relational Database

The control system must include a relational database as a central repository for all configuration information. This should include all static information about accelerator components such as coefficients to calculate field magnetic strength from current. Consideration should be given to extending the database to include all technical information to enable subsequent support and maintenance. At the application level, there should be a unified and seamless interface to both the static and dynamic data.

6.3 Identification of Control System User Groups

The control system must support several user groups, each with varying requirements.

Accelerator Operators

Accelerator operators are the principal managers and users of the control system. It must be a complete and consistent interface for them to perform any function in the accelerator complex. The data and interfaces must be consistent in how data is presented and how equipment is seen to behave. The operation of the accelerators requires real-time control and monitoring of the equipment, archiving, alarm handling, sequencing, backup and restore for routine operation. For these users, alarm and error messages should be supported by information regarding recommended courses of action. The control system should allow the automation of plant operating tasks. It should provide applications that encourage and facilitate the keeping and passing of operation logs, particularly from shift to shift.

Accelerator Physicists

The accelerator physicists' requirements for the control system include all the routine operations of the control system together with the ability to integrate programs developed to support different accelerator models. Functionality is required to allow easy acquisition of data produced as part of an experimental run, and to provide the ability to switch between different accelerator models. Data retrieved from the control system must be acquired with sufficient time accuracy to enable accurate correlation.

Technical Groups

The technical groups require diagnostics to enable maintenance such as calibration and fault finding. Access to the control system is required in the main Control Room, local to the equipment, and potentially in the offices, laboratories and off-site. Applications must provide all diagnostic information necessary to assist in commissioning and debugging of equipment. They must provide useful fault diagnosis facilities to assist with plant equipment maintenance and maintenance of the control system itself (both hardware and software). An easy interface to databases of equipment properties, manufacturers, documentation, cabling data and fault histories is required, as well as access to information clearly identifying the geographical location of equipment and a system of fault prediction facilities to allow for scheduled maintenance of components likely to fail.

Beamline Staff and Experimenters

The end users of the experimental station require a straightforward graphical interface to the control system. They also require good integration of control system parameters with the experimental control and data acquisition systems. This is particularly necessary in the case of synchronizing scanning of a sample with changing a parameter on an insertion device in the storage ring, e.g., the gap of an undulator. Experimenters require clear information on light source status, performance, and timing signals, and may require remote access (i.e., from off site) to experiments and beam-lines.

Control System Engineers

Control system engineers require current and archived data on the status and behavior of the entire control system. Information required includes CPU loading, network loading, application monitoring (for frozen/crashed applications), connectivity status, and reports of any control system faults.

Facility Managers

The control system should be capable of producing operating reports and statistics in a form that can then be imported into software applications (i.e., spreadsheets, web-based tools, etc.) used by management. Information required could include the number of hours of beam time supplied to users and unplanned beam dump statistics – how often these events occur, time taken to restore beam, reason for beam dump, and signs of common modes of failure.

Public Users and Staff

A wide range of other groups will require information from the control system. These include technical and scientific groups on and off site. These groups should be served through a web service as the user interface.

6.4 EPICS Toolkit

EPICS is the result of a collaboration of control groups, across a number of research organizations, to produce a tool kit to build distributed control systems. The resultant tool kit reduces software development and maintenance cost by providing: configuration tools in place of programming, a large user base of proven software, a modular design that is expandable, and well defined interfaces for extension at all levels.

Worldwide, EPICS has a very large user base for a variety of accelerators, detector systems, astronomical projects, and industrial processes. Most recently, EPICS has been successfully deployed at the Diamond Light Source, the Spallation Neutron Source at ORNL, and the Australian Synchrotron Project. It is being used for the LINAC Coherent Light Source at SLAC, the Shanghai Light Source, and the multi-faceted accelerator facility JPARC at Jaeri.

The use of EPICS on a diverse range of projects means that there is a large base of drivers and hardware support already available. The existence of these makes interfacing of the underlying systems less dependent on software development.

The EPICS tool kit is supported through the collaboration with software distribution and documented through the web. There are EPICS training courses run each year by many groups in the collaboration, and there are two EPICS workshops rotating through the U.S., Europe, and Asia each year; a number of individuals and companies are also available to provide support and training.

6.4.1 Structure of an EPICS Control System

EPICS embodies the standard client server model for a distributed control system, and shown in Figure 6.4.1. The user consoles are one class of client that receives and processes information. The servers are the source of information and in the general case, the interface to the equipment being controlled. The clients and servers are physically connected using network technology and they communicate with the EPICS protocol Channel Access.

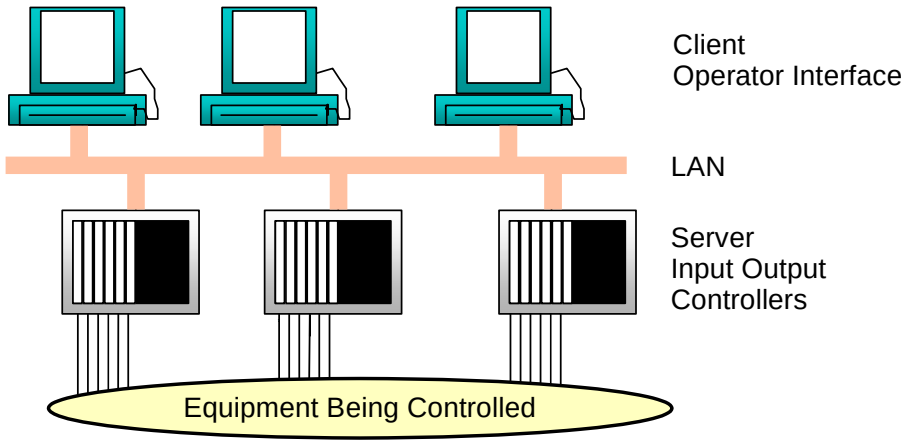

Figure 6.4.1 EPICS model.

6.4.2 EPICS Servers

The physical realization of EPICS servers is typically as multiple embedded VME systems, which are called IOCs, Figure 6.4.2. IOCs interface to the equipment being controlled, for which EPICS supports a large range of physical interface standards, protocols, and devices. IOCs also support the use of an event timing signal, to time-stamp transactions and enable synchronous acquisition or control across multiple IOCs.

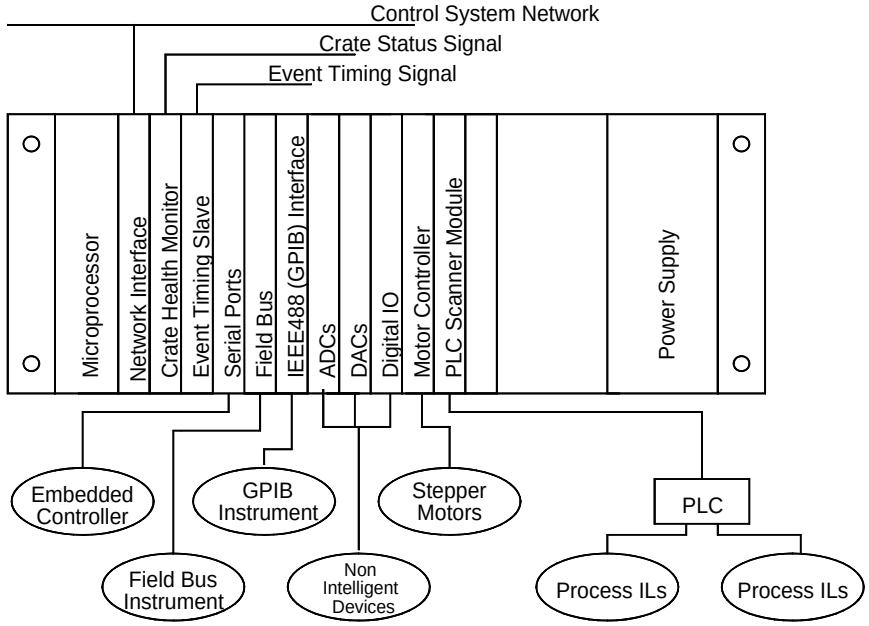

Figure 6.4.2 Example EPICS IOC.

6.4.3 Server Side Processing

Within the IOC, the CA server communicates with the Process Database, which uses the Record Support, Device Support, and Driver Support layers to interface to the plant, Figure 6.4.3. The communication from the EPICS client, over CA to the database, can be by synchronous call and reply or by the client establishing a monitor whereby the server asynchronously serves the data. The update of monitors can be on a periodic basis, on change of data, or on external event.

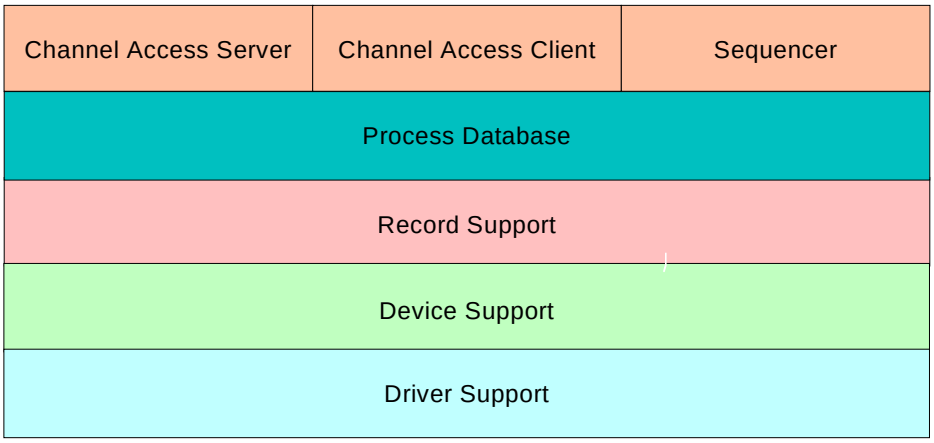

Figure 6.4.3 EPICS IOC data model.

The process database is a memory resident database that defines the functionality of the IOC. The database uses the Record Support layer to perform the processing necessary to access IO, perform data conversion, check alarms, and update monitors. The IO operations are carried out through the Device Support layer, which handles equipment specific protocols, and through the Driver Support layer, for the hardware interfaces. The structure provides support for interfacing to embedded controllers, field buses, IEEE488 (GPIB), DACs, ADCs, Digital IO, stepper motors, PLCs, power supplies, and a range of instrumentation.

Within the Input/Output Controller there is also a CA client to facilitate IOC-to-IOC communication. This is realized by linking process information from one process database to a process database on another IOC.

An IOC also contains a Sequencer to perform Finite State Machine control on the process database. The sequencer logic is defined as SNL, which is compiled to C code, then to an executable to run on the IOC. This allows for easy production on complex sequences, such as switching through the steps in bringing on a piece of equipment.

A standalone version of the CA server is available, which can be integrated into other systems without the process database and support layers. This facilitates integration of self-contained systems into EPICS, one example being the integration of LabView systems.

6.4.4 EPICS Clients

The client side of EPICS is realized on either Unix workstations or PCs running Windows and is called the OPerator Interface (OPI).

In the standard EPICS model, the OPI application programs interfaced directly to the CA client. This has limitations in that it only provides access to the dynamic control data through the CA API and so limits seamless integration of data from other sources, e.g., a RDB. The EPICS toolkit provides a suite of applications for the OPI. Among the choices for the core tools are: a synoptic user interface for control and monitoring (EDM), an Alarm Handler, an Archiver for recording and retrieving the historical state of the control system, a backup and restore facility to take snapshots of parameter settings, a knob manager to provide attachment of physical knobs to parameters and a parameter plotting tool. There is support within EPICS for the scripting languages Jython, Matlab, Tcl/Tk, Perl, and LabView. Data can be further served up to web pages through a CGI server.

6.4.5 EPICS Development Environment

The functionality of a control system built with EPICS is defined in three places: the Operator Interface, the Process Database, and the Sequencer logic. EPICS provides a number of tools to develop each of these which do not require software development. The OPI applications can be produced by a number of interface-generating tools, one of which is EDM. These tools allow for control objects to be placed on pages and animated with a connection to the control parameters. There are both text and graphical tools to produce the Process Database, which involves selecting records and drivers, and linking them to process data and alarms. The Sequencer logic is produced from SNL, which can be defined as text or, more recently, in a graphical form.

6.4.5.1 Distributed Development

Each developer will work on either a Linux PC or a Windows PC. These will be networked to a development file server providing read-access to all the necessary development tools and applications (VxWorks, RTEMS, Matlab, XAL EPICS base and extensions, configuration and database files, etc.), see Figure 6.4.4.

Modifications will only be made to local copies of applications, which will then be checked in to a branch of the CVS repository to enable any changes to be backtracked. When the application has been fully tested and is stable, this branch will become the main development branch for the control system.

A central boot server will be used to provide all the necessary files required by the local IOCs for booting. These files will be generated from the CVS repository. This will ensure that all IOCs are running a consistent and stable version of the control system. The contents of the central boot server will be mirrored on boot servers located in the control room, which will provide local booting of the IOCs.

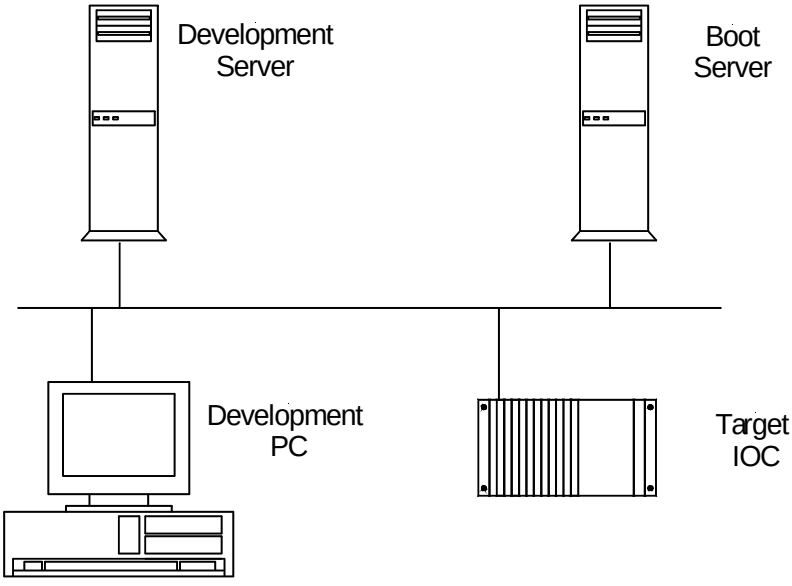

Figure 6.4.4 Distributed development structure.

File Management with CVS

CVS [6.5] is a version control system for keeping track of all modifications to project source code files. CVS is widely used in both open source and proprietary software development projects, and is generally

considered to be the best freely available, full-featured version control tool. Two features make CVS particularly suited to collaborative development across any network, including the Internet:

- Multiple developers can edit their own working copies of files simultaneously. CVS then deals with combining all the changes and notifying developers when there are conflicts.
- Developers have remote access to source code file repositories. Project members can obtain and modify project files from virtually anywhere.

CVS is a client-server system. The CVS repository is maintained on a server; clients run on users' machines and connect to the server via the network (or Internet). Clients are available for nearly all platforms, including Unix, Windows, Macintosh, and any Java-based platform. CVS allows project members to:

- Check out source files and directories
- View differences between versions of files
- View change log history and comments
- Commit changes made in their local copies of the source files to the main source code repository
- Update their local project files when they want to remain in sync with changes committed by other project members

CVS has proven very beneficial to many other accelerator projects in the world, and there is a very large CVS knowledge base within the EPICS community.

6.4.5.2 Application Development

Most application requirements can be met through the standard tools discussed in this section. Where more intelligence is required at the application level, there are EPICS interfaces to all the popular programming languages. The preferred solution will be to use C/C++, Java, or scripting languages to minimize the number of supported languages.

C/C++. C, and C++ are high-level programming languages that have become the de facto standard for portable open systems solutions on Unix/Linux platforms, with C++ usage increasing due to the popularity of Object-Oriented design and programming. Both languages have been widely used both for the EPICS baseline product and for driver software and other applications built on top of the baseline. For the NSLS-II Control System, the emphasis will be on re-use of existing software. Improvements are expected to meet NSLS-II requirements.

Tcl/Tk, Python/Jython. Tcl and Python are widely-used open-source scripting languages. They have simple and programmable syntax and can be either used as a standalone application or embedded in application programs. Tk and Python are Graphical User Interface toolkits that can be used for rapid development of powerful GUIs. Tcl/Tk and Python are highly portable, running on essentially all flavors of Unix (Linux Solaris, IRIX, AIX, *BSD*, etc.), Windows, Macintosh, and more. Tcl/Tk and Python are well supported and extensively used on many EPICS-based projects, particularly for GUI development.

Java. Java is an object-oriented interpretative programming language with a built-in Application Programming Interface that can handle graphics and user interfaces. Java can be used to create standalone applications. However, a more important use is in the development of applets, programs that can be embedded in a Web page. The growth of the Internet, together with Java's hardware independence, has made the language essential for web-based developments.

Currently, Java performance issues mean its usage will only be considered for applications where response time is unimportant. Generally, though, Java solutions providers are seeking to improve performance with developments such as just-in-time compilers and Java processors. If these developments yield effective

performance improvements during the development phase of the NSLS-II project, then Java's importance to the project will increase.

6.4.5.3 Server Development

Development of EPICS at the server level is required potentially in three places, namely record and device support, database, and state notation language.

6.4.5.4 Record and Device Support

While there is extensive record and device support available for EPICS, addition of unsupported hardware will necessitate the development of Device and possibly Record Support layers. The EPICS toolkit provides well-defined interfaces to each of these layers and examples to aid development. Device development is carried out in C within the standard EPICS development environment.

Building and developing EPICS requires either the VxWorks [6.6] or RTEMS development environment. VxWorks is currently only available for Windows or Solaris. However, given that the development environment is based on the GNU tool chain, it should be possible to run the RTEMS tools on Linux. The preference will be to standardize on one operating system for development, preferably Linux.

Database Configuration Tools

There are several Database Configuration Tools available. These DCTs allow designers to create EPICS databases by implementing them visually with a "block diagram and link wire" approach, similar to that used in electronic schematic design packages.

NSLS-II will use VisualDCT [6.7] as its database configuration tool. VisualDCT is an EPICS database configuration tool written in Java. It can therefore run under any operating system that supports a Java Runtime Environment. It was developed to provide features missing in existing configuration tools and to make databases easier to understand and implement.

The database development cycle will involve importing the EPICS runtime database into the central relational database to have a single repository of all control system information. VisualDCT has a powerful database parser, which allows existing DB and DBD files to be imported with ease. The parser detects syntax errors in databases, as well as defective visual composition data or its absence. Faults in DB files are safely handled and do not raise any critical errors. VisualDCT automatically lays out all objects that have no visual composition data and saves all visual data as comments to maintain backward compatibility. The output from VisualDCT is also a DB file, with all comments and record order preserved.

Visual DCT has been written within the EPICS community specifically to support EPICS, and is available free to EPICS database developers. However, some development of VisualDCT required to add some missing functionality will need to be undertaken.

State Notation Language / Sequencer Tools

The sequencer is a tool within EPICS that allows the implementation and control of one or more state machines on the IOC. The state machines are created using EPICS SNL. SNL has a C-like syntax, with constructs for building state machines. Once the SNL source code has been written, a SNC pre-processes it into "C" code and then compiles it to create an object file which the sequencer runs in the IOC.

6.4.5.5 Client Tools and Middleware Data Servers

Client tools are available at level 2 of the control system architecture. Clients at this level can directly access all channels in the control system through the Channel Access protocol. These data are time stamped by the Event System for reconstruction of accurate time sequences or correlation of events. At this level, client tools can use data from the IOCs directly, use control system data along with accelerator equipment information for model-based physics applications, or provide computed or correlated data to other clients.

6.4.5.6 Console Applications

The EPICS software package offers comprehensive operator display applications, which include:

- Extensible Display Manager
- Channel Archiver and Archive Viewing Tools
- Strip Chart Tool (StripTool)
- Array Display Tool (ADT)
- Parameter Display Page (DP)
- Alarm Handler
- Knob Manager (KM)
- Operator Electronic Log (CMLOG)

These applications will be used to supply operator display facilities, which will include the following functions.

Operator Menu Bar

This will provide rapid single-click access to all key operator facilities.

Plant Synoptics

These full-screen plant schematic diagrams will provide operators with an at-a-glance indication of plant conditions. Each process value displayed on a synoptic will constitute a “hotspot”; clicking on a hotspot will produce a pull-down menu providing access to further information and control actions relevant to that process value. Typically, a text description of the signal, the units of measurement, alarm limits, maximum and minimum, trend, alarm history, wiring information, operator comment and web access to online help might be provided. By this means, plant synoptics will act as the launch platforms which allow operators to access a wide variety of data in a seamless manner.

Ease of navigation will be considered during the detailed design stage for plant synoptics. An overall Synoptic Menu will be provided, which lists all synoptics grouped by functional area, presenting a clear hierarchy. In addition, where appropriate, plant synoptics will contain links to other associated synoptics. The design aim will be that operators should be able to navigate around the hierarchy without the constant need to return to the Synoptic Menu. Plant synoptics will be designed to have a simple, uncluttered appearance so as not to present more information to the operator than can reasonably be taken in.

Control Panels

Usually sized smaller than full-screen, control panels will be available with a wide variety of control widgets (radio buttons, slider bars, data entry fields with data validity checking, etc.) to allow users to apply control actions to the plant.

Control panels can be configured such that a single slider bar is used to control simultaneously a number of control outputs. Mathematical functions are available to define how these combined control outputs operate in relation to one other.

User-Configurable Tabular Displays

Operators will be able to configure their own sets of tabular displays showing closely-related accelerator parameters. Facilities will be provided to save these user-configured displays with a user-chosen name, and to recall the display from a list presented in a pull-down menu.

System Status Indicators

These schematics will show the status of IOCs, operator monitors, printers, etc. They will also display the health of key applications—so that, for example, operators are made aware quickly if alarm processing stops due to an alarm server program crash.

Operator Comments Facility

This will allow operators to enter lines of text comment for any plant input—to record, for example, when an input is not reading correctly due to a known fault. The presence of an operator comment for a process variable will be clearly indicated on any synoptic which displays that process variable. Individual comments will be easily readable via a suitable control panel, and it will also be possible to collate lists of comments (e.g., all operator comments entered during a shift).

Signal Information Panel

Only a subset of the process variables will be displayed on plant synoptics. However, operators require rapid access to information about any process variable and the Signal Information Panel satisfies this requirement. The panel will provide a Search section and a Display section. The Search section will enable the user to carry out a name search on the relational database, using a name mask to search for either an EPICS database record name or an EPICS record descriptor. Clicking on one of the returned search results will enable the user to request further information (e.g., trend, alarm history, operator comment, etc.).

Message Logging

The CMLOG package available with EPICS will be used to provide a distributed message logging system. This package can be used by any application or system that needs to log messages to centralized log files and display distributed messages to users. The CMLOG package supports C++, C, and CDEV application interfaces for logging messages and has C++ application interfaces for searching/retrieving messages from a dedicated logging server. Applications may send a selection rule to the server to select a subset of log messages for viewing; these rules can be in a form similar to C logic syntax or in a form similar to SQL.

A sample Message Log Browser (an X-Windows Motif application) is included with the CMLOG package. An additional browser will be developed using the supplied application interfaces once detailed requirements are established during the detailed design phase of the project.

6.4.5.7 Alarm Handling

The EPICS Alarm Handler package will be used to provide the following facilities:

An alarm list allows the users to view and manipulate current plant alarms. The alarm list will incorporate the following facilities:

- Indication of alarm acknowledgement state.
- Alarm message which includes EPICS record name, descriptive text, alarm value and date/time of alarm generation.
- Removal of acknowledged alarms from the Alarm List when they are no longer in the alarm state.
- Access to a menu-based set of facilities from each alarm in the Alarm list. The menu would give access to further information about the alarmed signal, including:
  - Trend
  - Alarm history
  - Access to a synoptic which includes the alarmed signal.
    - Web access (e.g., a link to a text page with more details about the alarm condition and possible corrective action)
    - Operator-selectable alarm inhibition to prevent use of the Alarm List from being disrupted by non-genuine alarms (e.g. “flickering” alarms being generated by a faulty switch). The names and descriptions of inhibited signals will be viewable on a separate list, from where it will be possible to de-inhibit each signal.
    - Association of each alarm with a plant area, along with the ability to display only alarms for a particular plant area.
    - Color indication of alarm severity.

All alarm messages will be logged to a text file for interrogation and archiving purposes. An alarm log viewer will be available, with various filtering options such as date/time, alarm severity, input name, etc. Provision will be made for audible alarm tones, driven from software using wav files. A separate alarm tone will be available for each alarm severity. An alarm banner window will be available to display a configurable number of recent alarms in a dedicated window at the top or bottom of the screen. Alarms can be acknowledged via the banner without having to call up the main Alarm List.

6.4.5.8 Archiving

The EPICS software toolkit offers comprehensive short, medium, and long-term data collection, archiving and retrieval through the EPICS Channel Archiver package. This package will be used to provide the following facilities. For long-term archiving, the archiver provides support for:

- Data retrievable in tabular and trend form
- A data quality indicator associated with each item of data
- Data compression to minimize the size of archive files
- Dumping of data to removable storage media, for long-term storage
- Loading of archive data from removable storage media for data analysis
- Timely warning to operators when archive data collection is compromised by a “disk full” condition on the archive server
- Variable data collection intervals for archiving
- A mechanism for easily configuring large numbers of process variables for archiving (e.g., by use of name masks)
- Facilities for collecting data in user-definable data sets, where data sets can include files as well as process variable data

The Historical Data Collection provides for short- to medium-term data collection offering the following features:

- Data retrievable in tabular form and trend form
- Data quality indicator associated with all data
- Variable data collection intervals
- Mathematical functions (e.g., averaging, MIN-MAX, etc.) applicable to historical data

A wide variety of data retrieval and data management tools are available with the standard Channel Archiver package, including:

- Retrieval via scripting tools, provided by the Channel Archiver Scripting Interface. Tcl, Python or Perl can be used to develop automation of archive handling.
- Retrieval via native tools, with Xarr/Striptool for UNIX-based systems and WinBrowser for Win32 systems. WinBrowser also provides data export in spreadsheet format or in a format suitable for the Matlab data analysis and modeling package.
- Retrieval via a web server plug-in, offered by the CGIExport client, which allows users to browse the archive via any web browser. File download in spreadsheet or Matlab format is supported by this plug-in.
- Command-line tools provided by the ArchiveExport/ArchiveManager component, providing commands to manage archives and to export data to a spreadsheet, to Matlab or to the GnuPlot plotting utility program.
- The Archive Engine component of the Channel Archiver package includes a built-in Web server. By using this feature, current operational parameters can be viewed and interactive configuration can be carried out via any Web browser.

6.4.5.9 Plotting

The StripTool program will be used for displaying trends of current and archived data. The key features of the StripTool program are:

- A StripTool chart displaying recent live data can be scrolled back to view archive data.
- Data acquisition via both Channel Access and CDEV, thereby allowing trending of both EPICS and non-EPICS data on the same axes.
- Ability to drag signals from synoptic diagram (drawn using MEDM) into a graph window.
- Flexible configuration options, including logarithmic and linear transformations, sampling rate, graph refresh rate, plot colors, grid lines, graph legend coloring, and plot line width. The trend can also be easily reconfigured to make one or more plot curves invisible without removing the plot configuration information for that curve.
- Trends can be printed and trend data saved to file.
- Trends are customizable via X resources, giving access to a wider set of configuration options than those offered by the standard StripTool configuration facilities.

6.4.5.10 Automatic Sequencing

For increased operational efficiency, and in support of a demanding accelerator availability requirement, the control system will include the capability of automatic sequencing, including decision making. These sequences could include automatic run-up procedures, automatic fault-recovery sequences, and automatic data-taking routines. The system will provide simple tools for defining sequences as experience is gained and will be capable of monitoring the status of automatic sequences, annunciating problems encountered in sequences, and intervening or overriding sequences if necessary.

6.4.5.11 Data Server

Computed data and aggregate data are to be done with consideration to overall performance metrics. Where it is reasonable, these data are to be created once in a server and provided to other clients in the control system. Examples of this are first turn data, ring current, and emittance measurements.

6.5 Physics Applications Rapid Prototyping

Rapid prototyping of physics applications is supported through a number of programming language interfaces to the Channel Access protocol. These include: analysis packages such as Matlab, Labview andMathmatica; scripting languages such as Jython, Pearl, Tcl/TK, and SWIG; and programming language interfaces such as: Java, C, and C++. Applications that are prototyped in this environment can be migrated into the standard EPICS front end controllers and the XAL environment for operations. (Figure 6.5.1).

6.5.1 Model-Based Physics Applications

Model-Based Physics applications must be available for all phases of commissioning. The system should be capable of operating in online and predictive modes. A RDB must contain the model parameters needed for the model based control (Figure 6.5.1). A physics modeling system will be needed to provide an interface between the control system and standard codes such as Tracy2, Elegant, or RING. These codes can mathematically model the behavior of various accelerator systems. They can be used to aid understanding of the machine, as well as being a vital tool for optimizing and operating a complex accelerator.

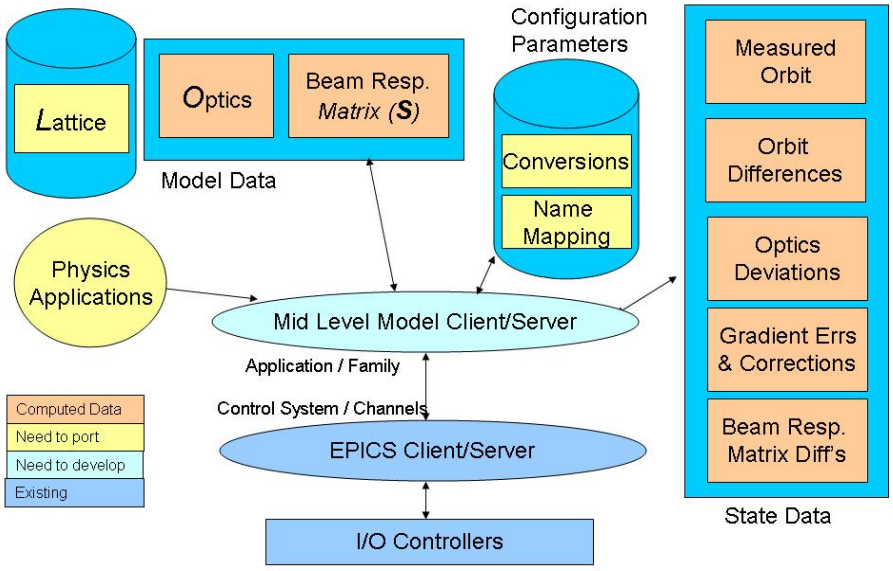

Figure 6.5.1 High-level application architecture.

Matlab Midfdle Layer Toolkit

Initial model based control is to be implemented using the Matlab Middle Layer Toolkit in conjunction with an accelerator simulation running through the EPICS process database. This provides an environment to develop commissioning tools that can be tested well in advance of the presence of any equipment.

Online Mode

Real-time magnet and RF data will be passed to the modeling system. Computed values of Twiss parameters, Q values, chromaticity, momentum compaction factor, etc. will be made available to the rest of the control system or to Accelerator Physics applications.

Predictive Mode

In this mode, offline magnet and RF values are supplied to the modeling system rather than real-time data. This will allow predictive and "look-and-see" experiments to be performed using the machine model to predict the behavior of the accelerator.

Middleware Data Servers

Middleware data servers will be provided to separate the data collection from the visualization layers. All client applications that collect data from the control system that result in the production of new data will be served to multiple, distributed clients. The distribution of this data is done over a standard protocol such as DDS or Croba. Alarm information, orbit, archive data, and emittance are some examples of data that may be produced by clients and served to other clients. The goal is to define a narrow interface that supports the reuse of clients of the API.

XAL – the Model-Based Control Framework

NSLS-II will use the new Middleware Data Servers to create applications for commissioning and operation. Operational tools are to be derived from commissioning tools. They will be modified for ease of use and clarity, maintainability, and revision control. Many of these tools are expected to be from the XAL toolkit in use at the SNS.

6.5.2 High Level Application Tools

High level tools include: system queries by physics devices or computed values, correlation plots, beam trip displays, machine protection first fault displays, and others as identified by the commissioning and operations teams.

6.6 Relational Database

MySQL will be used as the main data store for all beamline component information that is needed for model-based control. The IRMIS Entity Relationship Diagram will be used to describe all component and wiring data. It will be extended to support the Lattice information early in the project. Tools will be provided to enter this data, produce reports, and create files needed for all aspects of the project. These tools are to be developed early in cooperation with the users.

6.7 I/O Controllers / Equipment Interfaces

The I/O controllers and equipment interfaces must support the wide range of applications that are encountered in an accelerator control system. To minimize integration, training, and maintenance costs, this hardware should be limited to a solution that can meet each class of hardware integration needed. The front-end computers will run a Real-Time Operating System. The RTOS candidates are vxWorks and RTEMS. Although vxWorks runs on a wide variety of CPUs and provides much functionality, it does not provide

source code without a prohibitive investment. RTEMS is an open-source RTOS that requires more man-power to wield. Several EPICS sites now support RTEMS. Control at this level of the architecture can be done at the rate of 1 kHz with latencies of 33 μsec. Dedicated data and timing buses are required to achieve this level of performance over distributed controllers.

6.6.1 High-Speed Signals

High-speed signals such as RF, image diagnostics, and beam position signals may be processed with an FPGA to produce results that are used by the control system at a slower rate. These devices may operate on data into the MHz range and be used to analyze high-speed signals from LLRF, Beam Position Monitors, and Power Supply Controllers. These may be implanted as single device controllers that are equipped with dedicated processors to run EPICS and provide the interface to the control system, an FPGA to process the signal, a high-speed interface between the FPGA and the control system, and an interface to the timing system. These device controllers may control a single device or a set of devices. A standard architecture that includes a Power PC with a PCI or PCI interface in the Coldfire format is a candidate for this application.

6.6.2 Low Latency Response I/O

I/O that requires the control system to respond in the minimum time (known as high-density I/O) requires an instrumentation bus that provides interrupts on an external trigger and reasonable data transfer times between I/O boards. This can be implemented using either VME or PCI.

6.6.3 High-Reliability IO

Applications such as vacuum interlocks, flow switch interlocks, and cryogenic control require high reliability control of simple processes. A Programmable Logic Controller will be provided for these applications. All data from the PLC shall be available through the control system. The Control Logix PLC in conjunction with the Flex-I/O line could provide this function at a reasonable price. In any case, one PLC family will be chosen as the NSLS-II standard. These PLCs will be integrated into the control system through an IOC.

6.7 Global Control System

The control system must provide some global communication that requires higher performance than is available in a commercial network. NSLS-II requires: an Event System for synchronizing data acquisition and control; a high-speed data network for providing beam-steering data to all ring power supplies for orbit correction; and a Machine Protection System that is a fast-response bus provided for mitigation against failures that greatly impact the operation of the facility by either producing excessive radiation or causing equipment damage. We will evaluate the systems available from other laboratories. We are also considering the development of an open-source set of functionality that provides the timing, event, and data communication needed for high speed, distributed applications such as Fast Orbit Feedback and Machine Protection.

6.7.1 Event System

The Event System, also referred to as a timing system, provides all beam and RF synchronization for all control and data acquisition. The event system provides a master pattern of events that reflect the operation mode of the machine. It provides the synchronization needed to control the beam injection into the ring for initial fill and top-off. The event system may also communicate data that are required for operation and data

correlation, as well as data communicated to the subsystems that change with the mode of the machine. Examples include time stamp/pulse ID, machine mode, and global machine status.

The timing system is required to provide control of the beam transfer from the electron source to the storage ring and provide diagnostic equipment and beamline equipment with synchronization signals. The most recent light sources [8] have made use of commercial equipment and built on equipment designed by other light sources, often in collaboration with industry; it is envisaged that the same approach will be adopted for NSLS-II.

6.7.1.1 Fast Timing

The task of a timing system is to synchronize all the relevant components in an accelerator complex. One part of this task is to control the injection by triggering the particle source and firing the transfer line components, such as injection- and extraction-pulsed magnets, at the correct times. Also, beam diagnostic components such as beam position monitors and current transformers must be synchronized to the passage of the beam. This has to happen with fine time resolution, to RF frequency, clock precision, and low jitter, and is termed Fast Timing.

6.7.1.2 Event System Signals

Other tasks for the timing system are related to synchronizing components where the resolution is more relaxed. Examples include triggering the magnets for an acceleration ramp, triggering operational sequences such as the filling of the storage ring, BPM acquisition, feedback timing, insertion device control, and supplying the distributed control system with time synchronization for control and correlation of data. The time resolution for these tasks is less demanding; these tasks are often termed Events. Event Signals will be produced with a precision set by the storage ring revolution period and with predictable jitter.

6.7.1.3 Timing System Components

In designing the accelerator timing system, it is important to consider what has been used at other recently constructed sources and the integration into the EPICS control system. The time-stamp system already exists within the EPICS infrastructure and can be used in conjunction with the Event System, which was developed at APS [6.9] and enhanced by SLS and, more recently, DIAMOND (Table 6.7.1). The APS/SLS Event System can be used to meet all slow timing requirements. The Event System is fully EPICS compatible and the required VME modules are available.

Table 6.7.1 Diamond Version of the SLS Version of the APS Event System Specification.

Events	8-bit code – 255 events
Resolution	8 ns
Event TX trigger	Hardware input, software, Event Ram Clock.
Event RX output	Hardware output, software (EPICS record process)
Transmission medium	Gigabit Ethernet

The requirements for fast timing are more specific to a particular accelerator dimensions and operation. Two options (Table 6.7.2) are available for the hardware for fast timing, the KEK TD4V as a VME module delay generator and the Stanford Research DG535 as a self-contained instrument. Each is available with EPICS drivers to provide the controlled delays.

Table 6.7.2 Fast Timing Hardware Options.

	KEK TD4V	Stanford Research DG535
Form	VME 6U	Bench / Rack mounting
Delay	16 Bit / RF clock	0 to 1000 s – 5 ps steps
EPICS Support	Yes	Yes, via GPIB
Channels	1	4
Jitter	4.5 ps at 508 MHz	<60 ps

6.7.1.4 System Structure

Figure 6.7.1 gives an overview of the Event System and Fast Timing control. The Event Generator receives a start signal from the RF clock gated with a line frequency component. Events are then sent to all IOC Event Receivers for timestamp synchronization and to output relevant event signals or process EPICS records. The fast-timing IOCs will require a fast clock and trigger derived from the RF source, but fast sequences can also be initiated upon receipt of an event.

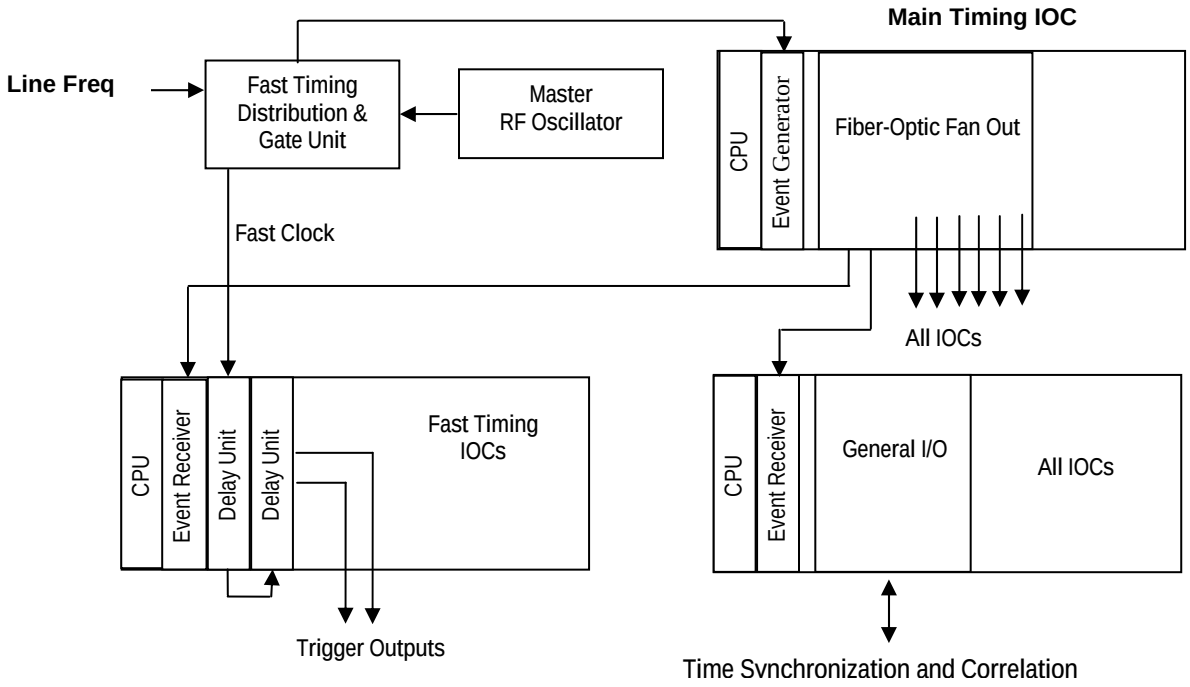

Figure 6.7.1 Block diagram of the Event system and Fast Timing system.

6.7.1.4 Signal Distribution

The Fast Timing and Event signals will be distributed over fiber-optic cable for reasons of noise immunity and distance capabilities. Further investigation is needed into the delay drifts that could be introduced by the fiber installation from temperature differentials and the standardization of length-induced delays in a facility the size of NSLS-II.

6.7.2 Fast Feedback

A beam position stabilizing system is required to maintain orbit stability to within 10% of beam dimension and to provide dynamic correction of low-frequency orbit disturbances. The proposals presented here are very much based on the work on Diamond, APS [6.10], ALS [6.11], and SLS [6.12].

6.7.2.1 Global Feedback

The feedback system will use measurements of the position of the electron beam in the storage ring and the photon beams in the beamline front-ends. This information will be compared against a reference orbit and the error used to calculate desired corrections to be applied to corrector magnets in the storage ring.

The response matrix relates the effect of small changes in corrector magnet fields to the resulting changes in the particle beam orbit as measured at chosen BPMs. By inverting the response matrix the relationship that maps orbit perturbations to changes in corrector magnet fields is obtained. For small orbit errors, this relationship is assumed to be linear and time-invariant. Different objectives, such as correcting the orbit in an rms sense or correcting specific locations, can be achieved by choice of BPM and corrector locations and by applying different weights to correctors or BPMs when computing the inverse response matrix.

6.7.2.2 Performance

Two global feedback systems, operating in parallel, are proposed to correct orbit errors on NSLS-II, namely a Slow system, correcting DC drift, and a Fast system, correcting beam disturbances to 5 KHz (Table 6.7.3). These systems will use data from both the electron and photon BPMs and operate on either or both of the steering magnet or fast correctors. For both systems, the BPMs need to be sampled synchronously, which will be achieved using Events distributed to the IOCs. In addition, this architecture must support the capture of beam dump failures.

Table 6.7.3 Feedback System Comparisons.

	Correcting Feedback	Update Rate Feedback
Slow	DC drift	0.1 Hz
Fast	0.2 mHz –100 Hz	5 KHz

6.7.2.2.1 Slow Correction

The Slow correction will correct the orbit at 10 second intervals, using the desired correctors and BPMs to compensate for slow changes in the orbit. This will maintain the user-steered orbit applied at the beginning of each fill. Communication to the BPM and Steering IOCs will use the EPICS CA communication mechanism. The slow correction will be realized as a dedicated application running on either a console or a computer server.

6.7.2.2.2 Fast Correction

Fast correction is not possible through EPICS CA mechanisms because of insufficient bandwidth. It will be realized at the IOC level on separate feedback processor boards dedicated to this function. This involves partitioning the correction calculation across the 30 Steering IOCs to calculate the correction values for the steering elements local to that IOC. Each steering IOC requires access to all the BPM values, to give flexibility in the correction algorithm. This requires a high speed connection to share data between the all BPM devices and 90 Steering IOCs. Two potential solutions for this are to use either reflective memory or network broadcasts.

EPICS process variables will be used to control the feedback process, by downloading algorithms to the feedback processors and setting coefficients and update rates.

6.7.2.3 Reflective Memory

Reflective memory is an off-the-shelf solution to distribute information across multiple computer systems without requiring processor time. It enables BPM data to be written to the reflective memory module in each of the BPM IOCs and appear in memory in all the Steering IOCs. In the system shown in Figure 6.7.2, an event received by all BPM devices would cause the photon and electron BPM values to be read by the feedback processor and written to the reflective memory board for each of the processors. The data would propagate to all the steering IOCs and when all values are received, the feedback calculation would be carried out on the Steering IOC to produce the new steering settings. These values would then be written to the steering elements in conjunction with the slow system values received through EPICS process variables.

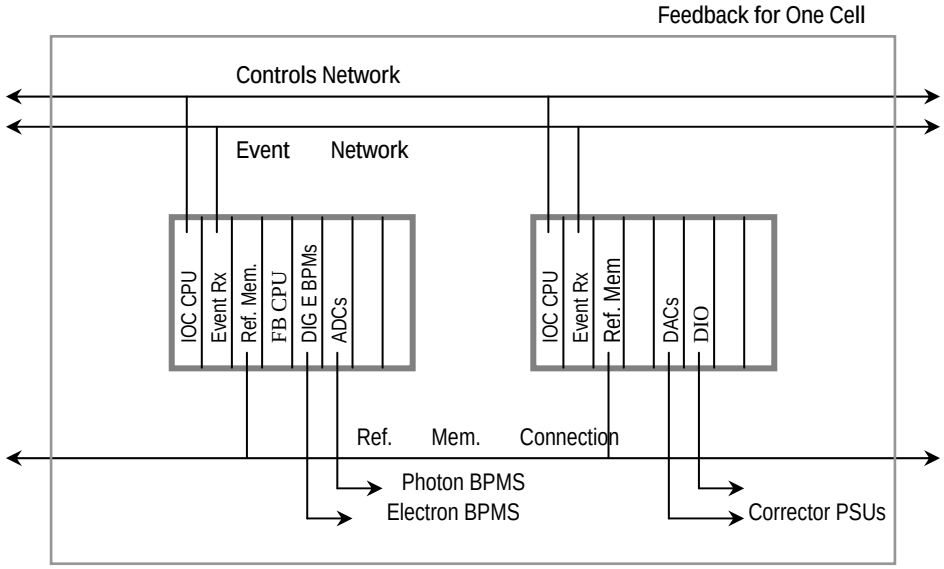

Figure 6.7.2 Reflective memory structure for one cell.

Commercially available reflective memory sizes and update rates provide for moving multi-megabytes per second across tens of boards, and so should easily meet the requirements of this application.

As the BPMs and Motor Controllers may be implemented using special FPGA boards, a second level of communication may be required to meet specifications. The architecture under consideration is show below (Figure 6.7.3).

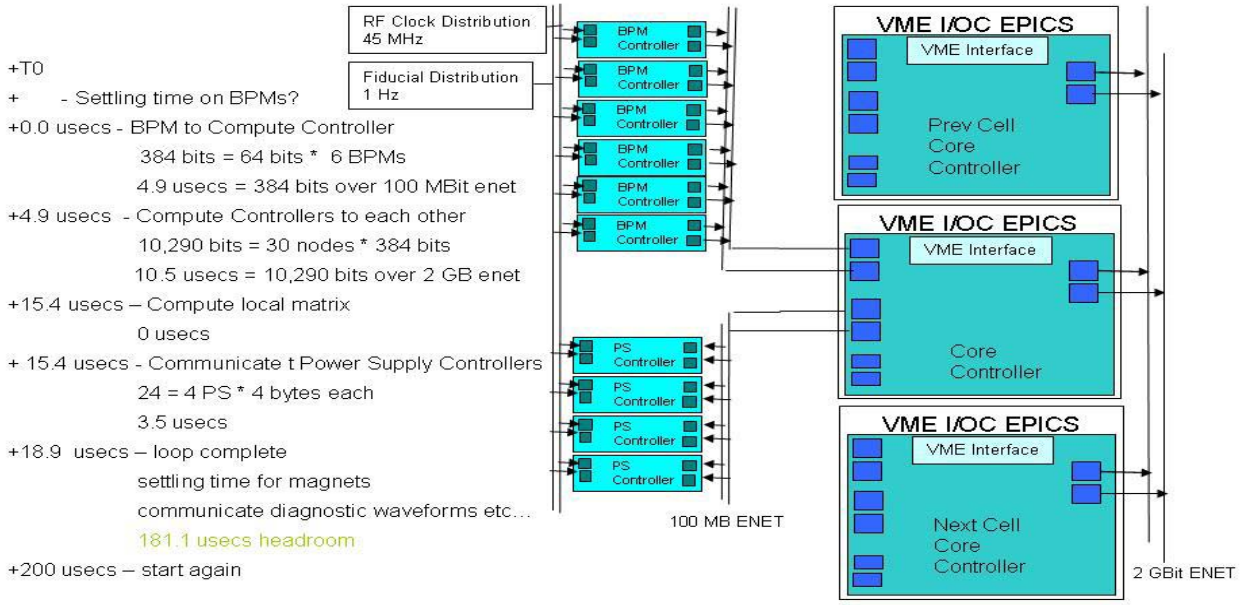

Figure 6.7.3 Architecture for the BMPs and motor controllers.

6.7.2.4 Network Broadcast

In the Network Broadcast system each of 60 feedback processors in the BPM and Steering IOCs is connected to a private network with a central switch in a star configuration. The feedback processor in each of the BPM IOCs reads the BPM values and broadcasts them over the network to be received by each steering IOC. The broadcasts take place simultaneously but do not collide, because the switch buffers each packet as it receives it. The switch then forwards the packets to all the Steering IOCs.

In the system shown in Figure 6.7.4, an event received by all BPM IOCs would cause the photon and electron BPM values to be read by the feedback processor and broadcast over the private network. When each of the 30 broadcasts has been received by all of the Steering IOCs, the calculation would be carried out on each Steering IOC to produce the new steering settings. These values are then written to the steering elements in conjunction with the slow system values received through EPICS process variables.

This option is cheaper in terms of hardware because it alleviates the need for the reflective memory boards, but incurs a development overhead to produce software for the broadcasting. The performance achievable using broadcasts also needs to be determined, to establish whether it would meet the requirements of this application.

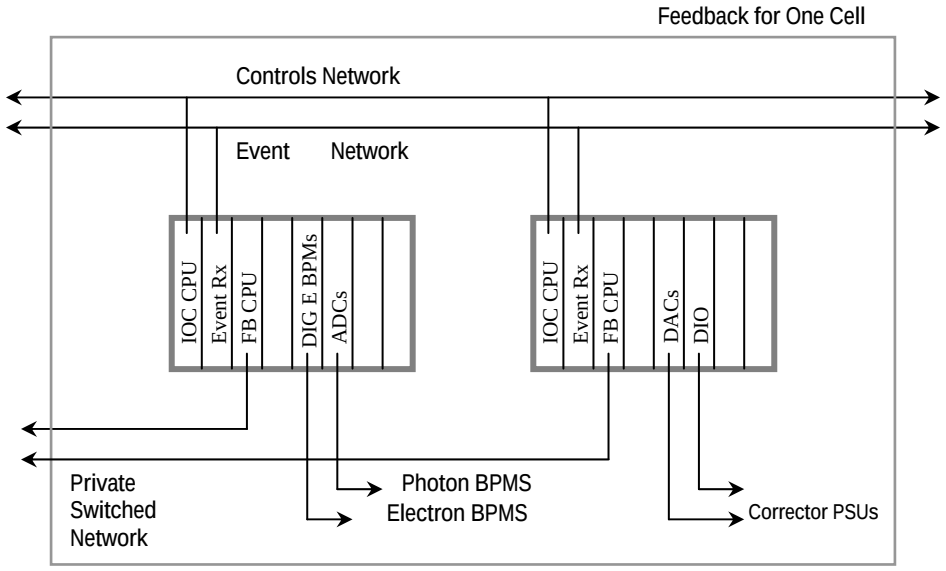

Figure 6.7.4 Network broadcast structure for one cell.

6.7.2.5 Feedback Processor

On the Steering IOC, the calculation to produce the new steering values from BPM values and the inverse response matrix needs to be carried out. The time available to carry out this calculation is dependent on the desired update rate, the time to acquire and distribute the data, and the time for the correction to propagate through the steering power supply, magnet and vessel to the beam. While the current generation of PPC processors offers similar performance as DSP processors, in terms of operations per second, they do not have the architectural feature of DSP processors for signal processing intensive calculations. However, the performance available from current PPC makes them suitable to carry out the feedback calculations at a price advantage over DSP processors.

6.7.3 Machine Protection

The control system must monitor beam envelope and other operational parameters that determine if the beam is threatening to damage equipment. Detection and reaction to dangerous conditions must be completed within 20 msec. A node on the fast feedback bus is to be used to determine if the beam is in a safe orbit.

6.8 Subsystem Control

The major subsystems being controlled are: diagnostics, power supply control, low level RF, vacuum, personnel protection, equipment protection, undulator, experimental beamlines, and conventional facilities. They fall into two distinct categories: those requiring high speed, low latency performance such as the LLRF, Diagnostics, Undulator Control, Beam Envelope Equipment Protection, and Fast Correctors, and those that do not require this such as: Slow power supply control, vacuum, personnel protection, slow equipment protection, and conventional facilities. Beamline control is a special case, in that the beam lines are a special cast in that they are run by groups outside of the NSLS II machine group.

6.8.1 High-Speed, Low-Latency Applications

High-speed, low-latency applications are integrated into the global control system through the EPICS Channel Access protocol, over Ethernet (Figure 6.8.1). This provides integration with modeling, archiving, and other operations tools. It is not adequate for response times required for FOFB at 5 KHz or machine turn-off in under 20 msecs. To meet these needs requires the event system, in conjunction with a fast, reliable, low-latency communication network. Intelligent device controllers in the front end process the signals at the 500 MHz rate and deliver data to the other device controllers over dedicated networks at the 20 KHz rate to support global feedback at 5 KHz. The intelligent device controllers also support turn-by-turn data collection, where a trip is detected and distributed over the event system.

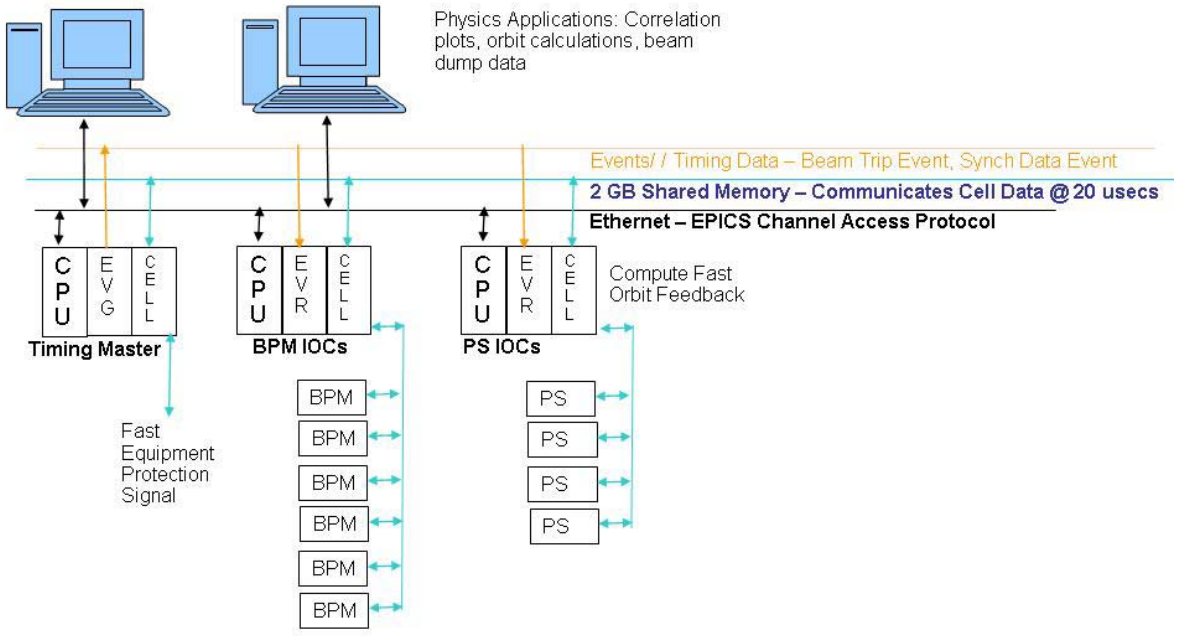

Figure 6.8.1 Control system for high-speed, low-latency applications.

6.8.2 Diagnostics Control

The slow applications, which require control and synchronization to the machine in the 60 Hz to .5 Hz range, have no need of specialized data communication buses. The event system may be used to provide triggers for synchronous data collection and time stamps for correlating events. All data communication is accomplished through the EPICS Channel Access Protocol (Figure 6.8.2). The I/O may be distributed over Ethernet or serial communication to slow, remote, controllers such as PLCs or off-the-shelve gauge controllers. This distribution of I/O reduces cable and maintenance costs. These distributed I/O are used when the data being correlated only needs to be within 500 msec. I/O requiring tighter correlation, such as slow magnets or motion controllers, are placed directly in the EPICS I/O Controller typically in a VME crate.

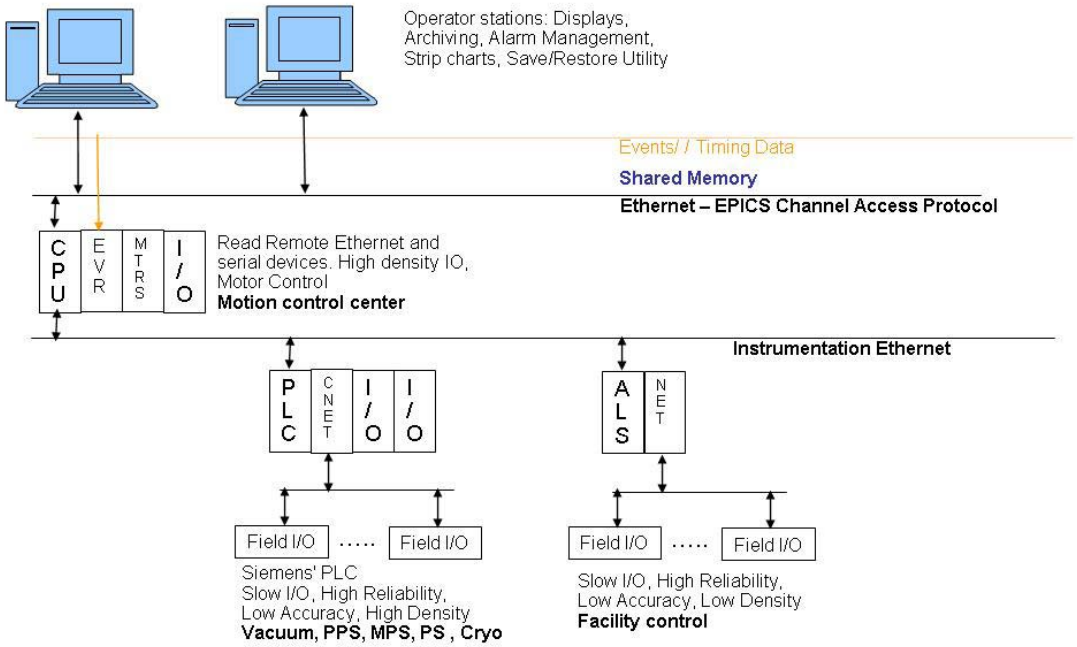

Figure 6.8.2 Control system for slow applications (within 500 msec).

6.8.3 Beamline Control

The NSLS-II accelerator control system will be in an isolated network by itself and each beamline is expected to have its own network to connect its computers and its hardware. Connections between these systems will be designed to provide the integrated flow of information between the accelerator and end stations that is needed to meet the requirements for beam monitoring and stability.

All insertion device beamlines require control of the insertion devices [6.13] which are located in the controls network. In addition beam position information, as well as numerous other signals, is needed from the accelerator control system. Similarly, beamline information, along with intensity and beam position from the beamlines, will be needed in the accelerator control system to provide continuous control of the beam.

The infrastructure needed for the exchange of information between the beamlines and the accelerator will be built into the facility. It is anticipated that single-mode fiber will be employed to connect the beamline EPICS hardware and accelerator EPICS hardware. Every beamline will be provided with a dedicated single-mode fiber bundle from the beamline to the accelerator control system racks located on top of the storage ring. In addition, there will dedicated single-mode fibers to the main control room, where some of the timing hardware is expected to be located. These single-mode fibers will be used to provide high-precision timing signals to synchronize the beamline experiments with the arrival of the x-ray beam.

Data exchange between the beamline and the accelerator EPICS systems, which do not require a very fast data rate, will be provided through an EPICS channel access process variable gateway system. This system will reside in the accelerator control system and will have an additional network connection to each of the beamline networks. This way the control, as well as data readbacks, can be accomplished with a high level of security without jeopardizing the integrity of the accelerator or beamline control systems. Such schemes have been used successfully at other facilities, such as APS and BESSY.

The development of the EPICS software for the beamline will be conducted in parallel with the accelerator to ensure that they are consistent and the exchanges of data between the two are seamless.

The beamlines will have their own computers and servers for data acquisition and control. There will be large storage capacity at the beamlines, and a central storage and backup service, with large disk farms, will be available as well. There will be 10 gigabit redundant network structure built into the facility. Each beamline network will be redundantly connected to a central computing facility, which will have a firewall to provide secure access to the outside world.

The offices in the LOBs will also have computing capability. Each LOB will be on a different network and will be serviced by a central computing facility. The LOBs will also be serviced by 10-gigabit network infrastructures. The centralization of data collected from the beamline will allow a user to retrieve data from multiple beamlines for analysis. Data reduction and analysis software will be developed by, and available from, the central computing services.

The variety of information exchange between the various control systems related to the needs of the experimental facilities is schematically indicated in Figure 6.8.3. Patterned after the system at APS [6.14], this system will be based on EPICS process variables, as discussed above.

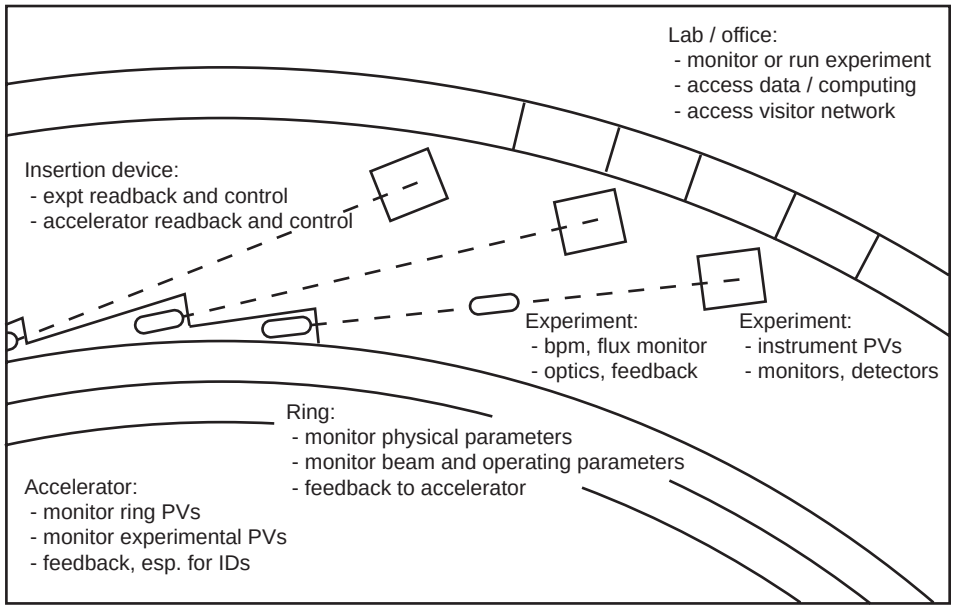

Figure 6.8.3 Activities requiring information distribution in experiment hall.

6.8.3.1 Design Concepts

From the point of view of the beamline and end station, the requirement is that every beamline have at least one dedicated EPICS IOC controlling its insertion devices and intercepting other information from the facility. For many of the hard and soft x-ray facility beamlines, EPICS will be used as a standard to control optics and beamline motors also. Specialized equipment may have different control systems, which may exist as additional EPICS clients or may in some cases be independent. It will be beneficial for the EPICS IOCs to be of a standardized hardware type. VME is a current favorite, due to its reliability, but the availability of drivers in the future and the improvements in PC-based hardware may cause the latter to be more favorable some years from now. The standardization of beamline server hardware will be assessed during the years preceding beamline commissioning and a standard will be chosen. The same requirements we have today must be satisfied. Hardware must be robust and reliable. A large basis set of scalers, motor indexers, analog and digital I/O, multi-channel analyzers, and so on must be in use. Pre-existing EPICS support for the hardware will be a third criterion.

The network as seen from the beamline is illustrated in Figure 6.8.4. As an example, a Linux PC running Medm GUIs may serve as the operator interface, while a VME CPU serves as the IOC for the beamline control system. Beam transport components with PVs that need to be available to the facility are connected to this network. (The insertion devices, on the other side of the shield wall, are networked separately.) A server acting as router, firewall, and PV gateway connects this sector or beamline network to the experimental network, to other beamlines, and to the other facility networks. The PV gateway controls all permissions necessary to establish who has control permission vs. read-only permission of the PVs.

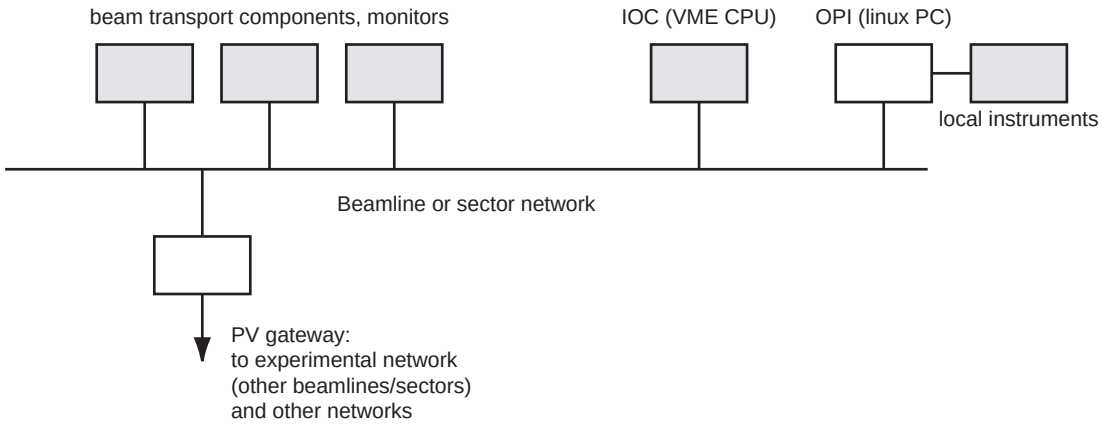

Figure 6.8.4 Sector networks make important process variables (PVs) available to the entire facility through a PV gateway. IOC=input/output controller, OPI=operator interface (as Medm GUI).

We anticipate that the system will be designed so by default read-access is available for everything, everywhere. This way, there will be no need to make assumptions during the initial configuration as to which machines' PVs and monitor values need to be communicated between the accelerator and the experiment hall. Finally, local instrumentation can be assembled on private networks as desired by the users of the individual beamlines. Specialized detectors and other instrumentation controllers may be interfaced to the beamline controls as applicable, particularly as EPICS clients.

6.8.3.2 Considerations for Accelerator–End Station Information Exchange

The accelerator monitoring complex will have far too much information in it to be useful to users as raw PVs. The accelerator group will design simple ways of mapping and colorizing the status of the accelerator with respect to beam excursions, pressure, temperature, and other sensor readouts, and make this digested information available as a monitoring system that the experimenters will be able to use. This will allow the beamline staff and users to gauge where problems may be coming from. It will also provide a unified way for the experimental division to log local beam stability in connection with pertinent information about the ring stability. This information will be provided as part of the general control system information.

The following elements will be expected to provide useful data from the beamlines back to the operations staff:

- position monitors (beams and pipes)
- ambient conditions (pressure, temperature, cooling water flow)
- status of insertion device and recent history (i.e., feedbacks triggered)
- beam figure of merit (quality after optics as defined by the end stations' needs: divergence, angular motions, spatial motions, energy drift).

It will be beneficial to create more generalized "status" readouts for beamlines as well as for the accelerator.

6.8.3.3. Design of the Main EPICS Node at Each Beamline

Resident on the experimental network and associated with each beamline, there will be a CPU running an EPICS IOC and a Linux PC configured with control GUIs and administrative scripts for that beamline. These computers will be quite similar across the NSLS-II complex, since many upstream beam components and interfaces to the gateways and other networks will be common. These computers, and the configuration to interface with facility networks and information buses, will be centrally administered. Giving each beamline a similar, centrally managed system benefits operations by allowing a standard set of tools and debugging know-how to be applied everywhere. One important concern is cybersecurity requirements. Even if the experiment network is within a firewall, the laboratory will require certain security scans. Uniformity of the beamline machines on this network will make it easier for staff to provide patches and fixes consistent with both security and smooth operations.

EPICS tools alone are sometimes insufficient for scientific data acquisition. Synchrotron beamlines have diverse needs, such as dedicated detector drivers, reciprocal space calculation, spectroscopic tools, and visualization tools for imaging. The end station equipment will be so varied that a top-down attempt at standardization would be very harmful. Thus, each beamline is expected to instrument itself in an independent way. Still, NSLS-II users and staff will benefit from having as much common support as is reasonable to interface different experimental systems and connect them to the EPICS platforms. Many different data acquisition platforms can be clients of the EPICS system. For example, LabView and Spec are widely used control and data analysis systems.

References

[6.1] M.T. Heron, and B.G. Martlew, “A Review of Options for the DIAMOND Control System,” Proc. PAC96.
[6.2] EPICS, http://www.aps.anl.gov/EPICS.
[6.3] TANGO, http://www.esrf.fr/computing/cs/tango/tango.html
[6.4] V System, http://www.vista-control.com/
[6.5] CVS, www.cvshome.org
[6.6] http://www.windriver.com/
[6.7] VisualDCT, http://kgb.ijs.si/VisualDCT
[6.8] T. Korhonen, and M. Heiniger, “Timing System of the Swiss Light Source,” Proc. ICALEPCS2001.
[6.9] F. Lenkszus, and R. Laird, “The APS Event System,” Proc. ICALEPCS95.
[6.10] J.A. Carwardine, and F.R. Lenkszus, “Real-Time Orbit Feedback at the APS,” Proc. BIW98.
[6.11] C. Steier, A. Biocca, E. Domning, S. Jacobson, G. Portmann, and Y. Wu, “Design of a Fast Global Orbit Feedback System for the Advanced Light Source,” Proc. PAC 2001.
[6.12] M. Böge, M. Dehler, T. Schilcher, V. Schlott, and R. Ursic, “Global Orbit Feedback System for the SLS Storage Ring,” Proc. ICALEPCS 99.
[6.13] M. Ramanathan, M. Smith, J. Grimmer, and M. Merritt, “Overview of insertion device controls at the Advanced Photon Source,” *Rev. Sci. Instrum.,* **73** (2002) 1448.
[6.14] M. Ramanathan, M. Smith, N. Arnold, F. Lenkszus, R. Laird, K. Evans Jr., J. Anderson, and K. Sidorowicz, “An overview of the information distribution system at the Advanced Photon Source,” *Rev. Sci. Instrum.,* **73** (2002) 1445.
[6.15] http://hpdrc.cs.fiu.edu/Sem-DDS/documentation.html