Chapter 12

CONTROL SYSTEM

The control system of SESAME controls the operation of the synchrotron radiation facility and is the heart of the accelerator complex. The control system makes it possible to have access to the different devices of the infrastructure, the accelerator and the beam lines (see figure (12.1)) from the computers (workstations) located in the control room. The status of the accelerator and beam lines can be displayed world wide via the Internet by monitoring of some parameters (control variables) of the control system.

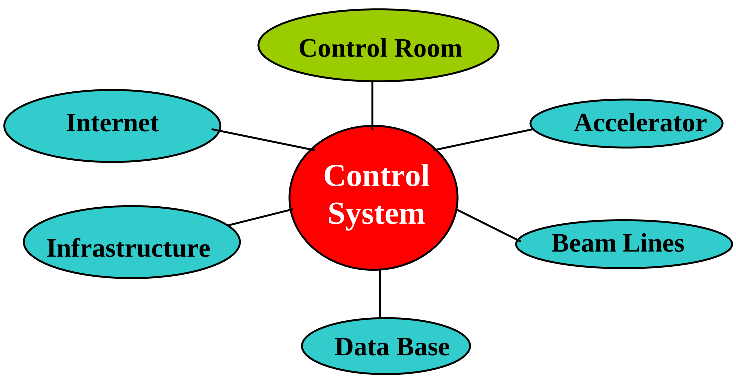


Figure 12.1: The connection of the control system to the different parts of the facility.

For the accelerator, the control system has to deal with the entire subsystems comprising an accelerator as given in figure (12.2). Each subsystem controls will be explained in more detail in section 12.5. Controls of the injector including all its subsystems is addressed in another section (see section 12.6).

Control System
Magnets
Vacuum System
RF-System
Power Supplies
Diagnostic
Cooling
Timing
Injector

Figure 12.2: Subsystems of the accelerators, which have to be controlled.

12.1 Executive Summary

SESAME control systems will utilize and use Experimental Physics and Industrial Control System (EPICS) toolkit for both Machine and Beam lines; it is a distributed hardware and software control system (standard model) that requires no central device or software at any layer. For the development of graphical user applications, the Java Abeans package will be used, which is an application framework optimised for accelerator control systems.

When developing the control system (CS) of SESAME we try to consistently follow our vision: **To produce a user friendly CS, where "user" stands for a physicist that operates the accelerator** and not a computer expert. Therefore, the main characteristics must be that the CS is easy to maintain and that it allows non-experts to easily build powerful applications. To make such user-friendly behaviour possible, a clean and consistent design is of paramount importance.

On the other hand, the budget for the whole machine must be kept tight and an optimum price/performance solution for the CS has to be found. For this reason one should incorporate as many standard and reliable commercial or open-source products as possible and not to unnecessary reinvent anything. We propose to use the same approach as for the SLS and ANKA control systems, which proved to be very successful both in technical performance, time-to market, simple installation and upgrade and management issues. It is possible, by adhering to the same design guidelines, to re-use the I/O boards of SLS, the core software EPICS and the applications of ANKA, which results in significant savings.

With the current state of technology, one should use PCs on the client side running either Windows or Linux operating systems. The software running on these PCs is best written in Java to allow platform independence. The input/output would be handled via VME boards, avoiding the need for an extra fieldbus system. However, these statements will probably change in time and therefore should be revised at the implementation phase of the project. Be it any system and programming language, it is very important that the installation of the whole CS is very similar to an installation of a commercial product – using an automatic installation procedure for the software and plug-and-play for the hardware.

12.2 Introduction

In comparison with other light sources, SESAME can be considered a small to medium sized installation. Its control system can therefore reflect a reasonably simple and efficient design, as has been applied to similar light sources such as ANKA, BESSY, SLS, etc. Even though all those light sources use somewhat different control system technologies, they all share the commonly accepted three-layer architecture (see figure (12.3)):

1. Device Input/Output Layer, which is the only one that connects to the actual hardware.
2. Process Control Layer, where processes distribute controlled device data to/from many clients simultaneously.
3. Visualization Layer, with graphical user interfaces and similar applications.

It is important to realize that the technology itself matters very little. The proof of this statement is the simple fact that all the above-mentioned light sources work very reliably and that their control system adds very little to the machine downtime. What is much more important is that detailed design and the implementation are made in a consistent way. This can only be achieved if all participants in the project, not just the control experts, adhere to well defined standard interfaces, both in terms of electrical connections and, probably even more importantly, in terms of software device interfaces.

The SESAME CS side must offer seamless crash recovery, flexibility in small variations of the design (e.g. new device types are added within minutes consistently across all layers) and above all it must enforce a consistent description of controlled items via standardized pattern interfaces. Such features have been implemented often by retrofitting existing set-ups and by highly non-portable solutions and spending a lot of resources, while a clean top-down design and the use of a powerful relational database could direct the implementation in the right direction with little added cost, just at the sake of increased discipline.

We assume the following tasks for the control system:

- Convert electric signals to digital values.
- Integrate other computerized systems.

- Display all values graphically on standard PC.
- Issue alarms to operator.
- Archive/timestamp values, alarms and logs.
- Synchronize instruments with timing signals.

12.3 Scopes of the Controls

Scope of the control system is from the supported electrical interfaces to the GUIs and all the hardware and software in between. It does not include personal safety (protection) systems (see 12.6.4).

These electrical Interfaces (signals) are supported:

1. Analogue I/O (-10 to +10 Volts).
2. Digital I/O (TTL, 0/24 Volts).
3. Motors (DC and Stepper).
4. Serial, RS232/422.
5. GPIB, IEEE 488.
6. Video (Analog/Digital).

12.4 General Requirements

The SESAME control system should be **reliable** enough to provide users (of the synchrotron light) with the required beam for the scheduled operations with the minimum number of failures and a short time for repair which means **availability**. High **performance** is required to ensure that the response to the operator action is within a certain time (in our case 100 ms). **Security** should be taken into account very carefully in such a way that lets every control parameter to be monitored or changed only within the SESAME accelerator complex where at the same time has a controlled access from outside. The control system should be **easy to use** for the machine operator; all the graphical user interfaces (GUI) should follow the same format and colors to monitor the signals and buttons.

Hardware should be replaced without interruption to the machine operation whenever possible. Regarding software development packages and tools it should be well defined at the beginning in order not to end up with using and supporting so many different packages.

12.4.1 User Requirements

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

a. Operators

The accelerator operators are the principal users of the control system. The control system must be a complete and consistent interface to perform any function on the accelerator complex. The GUIs should somehow let the user to start/stop the machine very quickly or change the parameters easily. Alarms should be monitored and acknowledged in a clear and well-defined manner. The Control System should allow the automation of onerous plant operating tasks.

b. Accelerator Physicists

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

c. 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. 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.

d. 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 synchronising 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 and performance, timing signals and may require remote (i.e. off-site) access to experiments and beam lines.

e. Control System Engineers

Control system engineers require current and (sometimes) archived data on the status and behaviour 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.

f. Facility Managers

The Control System should be capable of producing operating reports and statistics in a form, which can then be imported into software applications (i.e. spreadsheets, web based tools etc.) used by management.

Information required could include 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.

g. Staff and Public

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.

12.4.2 System Requirements

12.4.2.1 Architecture

Although some of the applied technologies are different, the architectures of ANKA and SLS are very similar. Therefore it is not a problem to take the best of both systems and connect them together for SESAME. Expectations of the final users determine the design of the CS. On one end of the CS they require the physical devices to be added easily and on the other one they want a powerful and easy-to-use user-interface. In addition it has to be taken into account that specifications are often modified during the course of development, usually by the addition of features. All this requirements demand a CS that is able to provide a great deal of flexibility.

Unlimited flexibility, however, results in unacceptably high cost. In order to achieve reasonable flexibility at low cost, the SESAME CS design needs a small number of fundamental building blocks across the whole control system, both in hardware and software, which are not allowed to be altered in any way. The design of the blocks is of course frozen only after careful investigation of the available requirements and has to foresee possible future needs.

In terms of hardware, on the client level, our goal is a solid framework of components that can easily be assembled into powerful applications. This structure of blocks or, components, respectively, allows the programmers to develop the layers of the CS independently of each other. The only constraints to the programmer are the interfaces that must be carefully designed in the first step of the design process in order to minimize interdependencies between layers, both in code and data. These interfaces must define all the possible interactions between layers in a consistent way. The result is cleaner code and better equipment, since every participant only has a limited number of things to worry about. If some components later have to be optimised, they can just be rebuilt from scratch without affecting the rest of the CS, because the interfaces stay the same. In addition, testing, debugging and error correction becomes much easier.

In table (12.1) we have shown some of the hardware being used at the SLS. They or their similar types can be used also for SESAME.

Table 12.1 List of the possible hardware can be used for SESAME.

VME Module - Stepper Motor Controller - 4ch with encoder
VME Module - Stepper Motor Controller - 8 ch/no encoder
VME Module - Synchronous Serial Interface - I/F to Encoder sub-system
Readout Module - Motor Encoder
Industry Pack - Digital in/out - 64ch
VME Module - IP Carrier Board - Analog/Digital
Transition Module - digital - 64in OR out - TTL
Industry Pack - Analog in - 8ch/16bit
Industry Pack - Analog out - 16ch/16bit
VME Module - IP Carrier Board - Analog/Digital
Industry Pack - Thermocouples
Transition Module - straight through - thermocouples
Accessories - Cold Junction Reference for Thermocouples
Industry Pack - Serial Interface
Transition Module - straight through
VME Module - I/O Controller
Powered Crate - VME64x - 21 slot
IOC /MVME2306 (Motorloa PowerPC IOC)
Trenew16878 (Crate 21 slot) / 64 bit

In the next few subsections we present the design from the technical aspect.

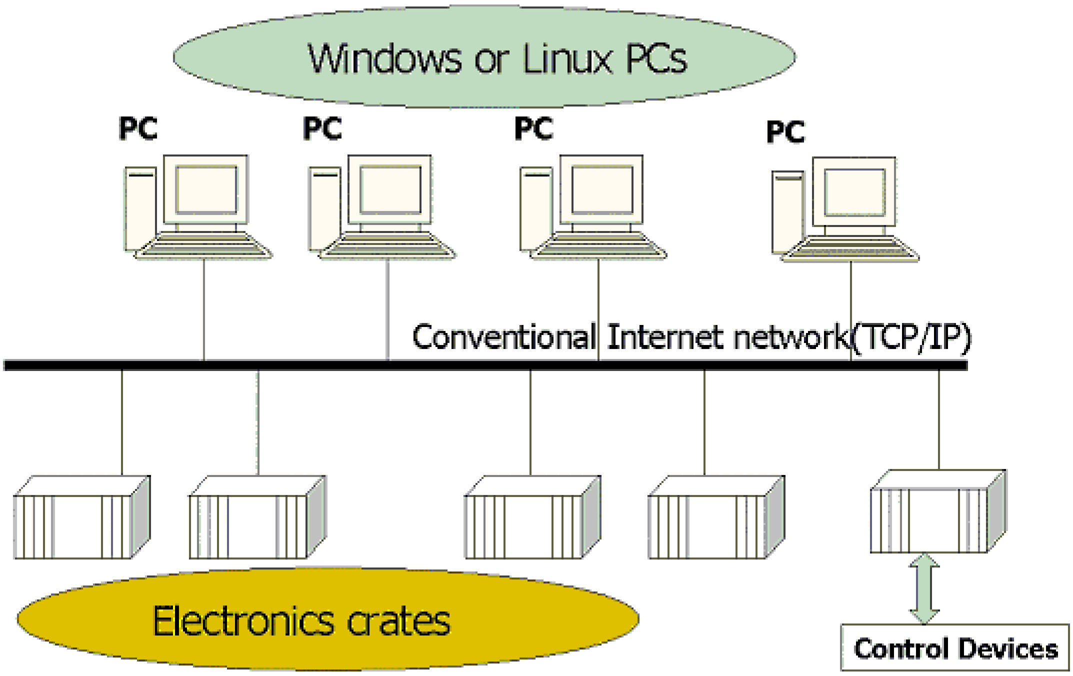


Figure 12.3: EPICS architecture.

12.4.2.2 Hardware Layer

The Front-End (FE) layer—referred to as the Input/Output Controller (IOC)—is built from VME crates, CPU boards (Motorola 680X0 and PowerPC families), and I/O boards and runs a real time OS. I/O boards support many standard field buses and interfaces like IEEE-488 (GPIB), Bitbus, CANbus, RS-232/485, and Ethernet.

The Back-End (BE) layer consists of PCs running Linux and Windows. These layers (FE and BE) are connected via a network layer, which is a media (Ethernet) and a set of repeaters and bridges supporting TCP/IP.

12.4.2.3 Software Layer

The software layer adopts 'client-server' model. The client layer usually runs on the Workstation/PC physical layer and represents the top software layer. The communication between server and client is done via channel access (CA), which is EPICS backbone that handles all communication via the network; it runs on both the client and the server. The CA server runs on the front end CPU and is a unique task on each network node. The IOC runs a real time OS, vxWorks is widely used but RTEMS is also highly considered.

Typical clients are:

- Operator interface panels, Display managers (MEDM, EDM), Strip Chart.
- Archiver.
- Alarm Handler.
- Applications.
- System Configuration Database.

12.4.2.4 Applications with Graphical User Interface

The applications of the visualization layer provide the interaction with the operator of the accelerator.They accept commands and display-controlled values, either one by one or in groups, through tables and charts. Another type of applications are applications that manage whole sets of devices at the same time, like generic device tables, ramping alarm table, snapshot, etc.

To represent the values of the accelerator variables (currents on magnets, status of the vacuum pumps, etc.), several graphical user interface (GUI) components will be used (see also figure (12.4)):

- Gauge to display the value of a property.
- Slider to manipulate the value of a property.
- Status LED display to display the status bits as on the physical panel of the device.
- Chart to display the same type of values of different devices along the ring, e.g. vacuum profile.
- Trend – a chart that displays one or several values as a function of time.

Another component is the “selector”, which enables the user to search for all available devices of a given type dynamically at run time and chose one or a group of them. When the choice is made, the system automatically takes care of the initialisation process and the gauge is immediately showing the correct value. An example of such an application is shown in figure (12.4).

In order make the automatic initialisation happen, a common framework for the development of all applications is needed. It is often neglected that this framework must also provide solutions for handling device errors and communication problems that occur. The framework should provide an application programmer’s interface (API) that completely hides the network. Ideally it should also be platform independent in order to be flexible for future hardware upgrades. Such platform independence can be achieved by using interpreted languages whose interpreter is implemented on several platforms, e.g. Java, tcl/tk, Python. However, one should not create a maintenance problem by allowing several of those language and GUI-building environments to be used. The libraries of the framework map the remote devices and their properties (encapsulate all remote calls from the client to a device server of the process control layer), following the same naming rules as defined in the process control layer. Besides this, they provide the following functionality:

- Open the connection and perform the function calls on remote objects.
- Report and manage all errors / exceptions / timeouts arising from network communication.
- Provide handles for asynchronous messages, queue / demultiplex responses.

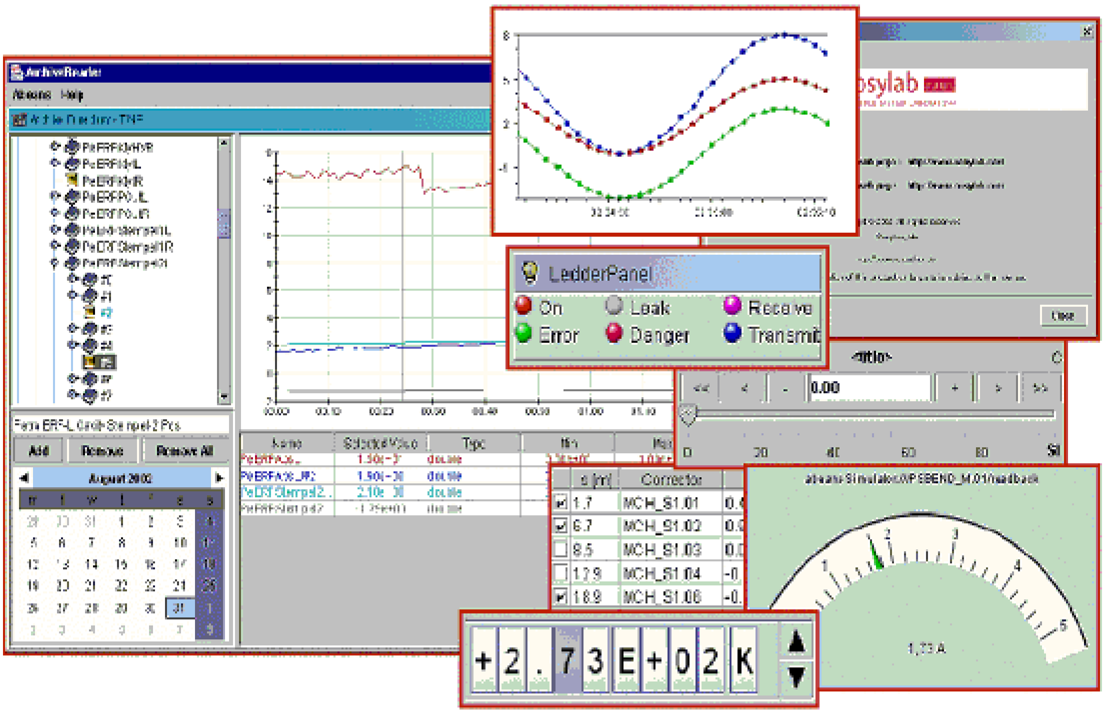


Figure 12.4: On the left half the picture is the archive reader panel. The tree on the far left, which resembles the familiar Windows Explorer, allows selecting any number of controlled signals. The datum selector in the lower left allows selecting the start and end date/time of the archived data. On the right half are several individual display components, which will be added together to form specific control applications: gauge (lower right), slider (above gauge), wheel switch (left of gauge), table (above wheel switch), status LED display (above table), trend/strip chart (above status LED display) and about panel (upper right).

The graphic user interface is very important, because this is what the operator uses in the daily operation of the accelerator. On this figure one can see a series of visual components from which a typical “control screen” is composed. They resemble mechanical displays and controls, because they should be as familiar as possible to a person that is otherwise not skilled in the use of computers.

Sometimes it is important that flow of the process is visualized. This is typical the case for the RF-system and the vacuum system. In such a case, symbols that represent the devices are used. They change color and shape according to their state. For example: a valve turns from green to red when it is closed; a pump is red when it is off.

Such types of displays are more common in control displays of conventional subsystems, such as the electric mains supply, the cooling system, water flow, personal safety system, etc. Usually, the company that provides the system already makes those displays. Such displays are then used for local control on a dedicated PC. See figure (12.5).

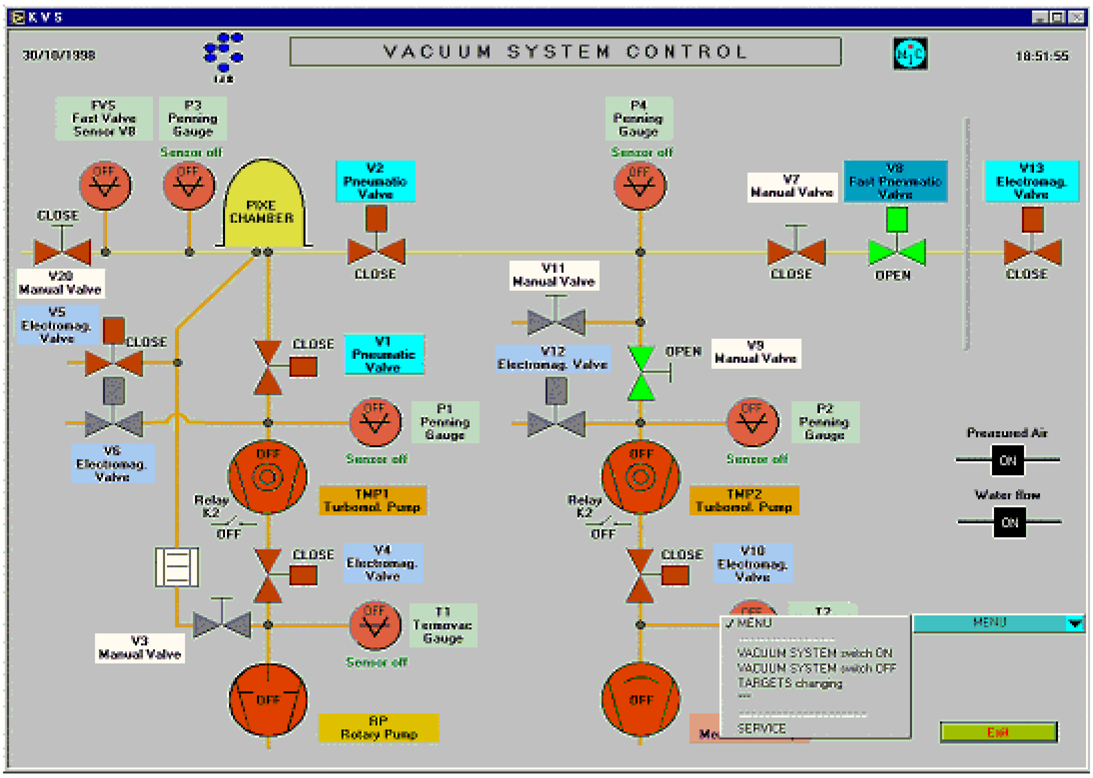


Figure 12.5: Vacuum Control System.

We will use the Abeans libraries, which is the only application development framework that combines all the above-mentioned features.

12.4.2.5 The Databases

The control system involves three main databases:

- Configuration database that stores configuration parameters like channel and device names, constants, calibration coefficients, attributes, alarm levels, I/O addresses, etc.
- Snapshot database that stores the state (i.e. all settings) of the machine.
- Historic database that logs data over long periods of time.

The main idea of the three-tier architecture is that clients don't access the database directly but through the EPICS and database server.

The static data used by the I/O boards are stored in the configuration database, too. They are downloaded to the EPICS server machines with a generic script procedure. Data that are used both by the client and EPICS server such as resolution, name, etc. are located only in one place of the database to ensure consistency.

To ensure a consistent configuration database, a version control of the database should be used. Ideally it should be the same versioning system as used by the software. The central version of the configuration database is stored on one computer and then the data are copied to all the others in the control system using a series of scripts.

The snapshot database is used and managed through a dedicated application. The data are stored using a relational database or just plain text files.

Long-term history data are stored into a relational database and retrieved off-line through dedicated applications. A compromise between the sampling rate, storage time and available disk space must be made when the system is set up.

12.4.2.6 Machine Physics Applications

Machine physicists need a set of applications that deal with the physics of the accelerator. Some applications must exist already at the start of commissioning like machine function display (figure (12.6)) and orbit correction; others are developed if the need occurs: tune optimisation, instability investigations, etc. As it cannot be predicted in advance, which applications are needed, most of them will be written by machine physicists and not the control group. Ideally, the machine physicists are given a software package for presenting the real-time state of accelerator transfer line in quantities relevant to machine physics. A unique design is the databush of Elettra, which has been revamped for ANKA and renamed DataBush. The DataBush connects to the accelerator devices, such as power supplies, beam position monitors, info server and then translates these objects to magnets and other to machine optics relevant devices and present them as software components. In this manner it acts as a bridge between machine physics and machine devices.

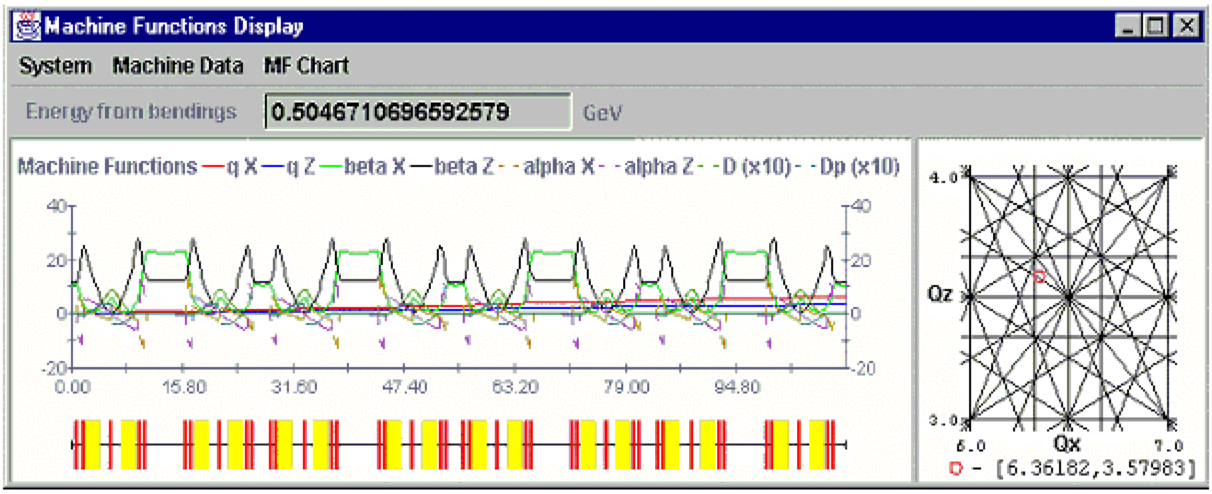


Figure 12.6:The machine function display made with DataBush. It displays properties calculated in "real-time" by the DataBush, which read all relevant data from the control system.

The DataBush is also able to simulate any device and work with it, as it would be a real machine device. This feature makes it easier to debug with DataBush and to use it similar to other accelerator optics calculating programs (MAD, etc.) but with the flexibility of a programming package.

Since the DataBush is Java package, it is platform independent. Plug-ins and models make it possible to fine tune DataBush to needs of particular implementation.

The DataBush is strongly integrated with CS. The accelerator specific implementation is hidden from the physics application into lower levels of the library. An application is communicating in this way with the accelerator through a general accelerator interface, not knowing/carrying about how this communication is realized. Two major benefits come out of this architecture

Portability: The architecture of the CS allows Java clients to be portable by wrapping the machine specifics in a well-defined interface. Machine physics application can be used on other accelerators without changing single line of code, with appropriate modification of CS plugs of course. This enables sharing accelerator software among different accelerator, i.e. ANKA and SESAME in our case.

More Efficient Programming: A physicist can concentrate on machine physics problems. No special knowledge of the CS is needed, since DataBush handles all communication with CS.

The following features make DataBush a flexible machine physics tool:

- Quasi real-time current or magnetic properties transformation.
- Quasi real-time linear optics calculation of machine function.
- Automation of mostly used machine physics routines.
- Representation of the machine with Java beans.
- Type and cast safe programming with DataBush beans.
- Customisation of DataBush operations with pluggable mechanisms.

12.5 Subsystems

Here we introduce a short description for the Timing and Diagnostics sub-systems, as they are critical and affect the overall performance of the machine and a brief to Beam lines controls.

12.5.1 Timing System

The task of the SESAME timing system is to synchronize the operation of the whole accelerator complex and its subsystems together with Beam lines and experiments. The triggering of the Microtron, the beam injection to the booster, the ramping of the energy, injection to the storage ring and ramping of the energy in the storage ring happen according to the synchronization pulses from the timing system. The timing system also provides the capability to inject to specific buckets in the storage ring and control the filling pattern.

12.5.1.1 Architecture

The SESAME timing system is based on the global distribution of timing signals using the so-called event system. This system is based on the design from the APS (Advanced Photon Source), which is redesigned for the SLS (Swiss Light Source).

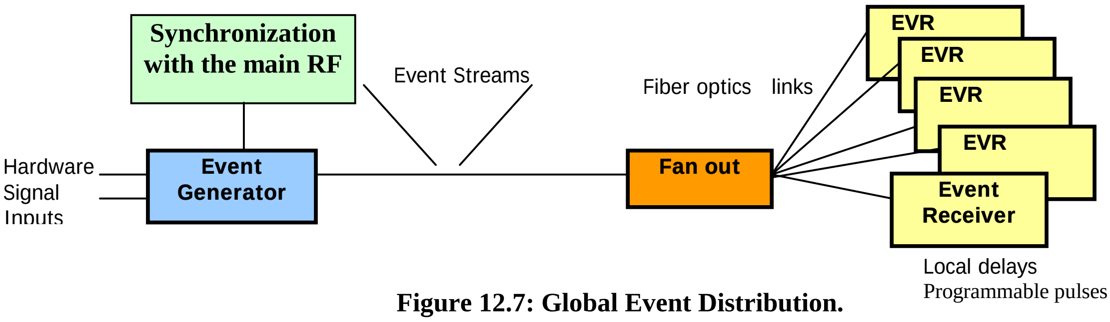


Figure 12.7: Global Event Distribution.

Events are requests for various actions like triggering of the injector or extraction from the booster or even start of a software action. Codes are assigned to the events and are stored in the RAM of the event generator then it is scanned through at a specified programmable clock rate. The events are time multiplexed in a frame and distributed to the all event receivers at a rate of 50 MHz (20 ns resolution, 15 ps RMS jitter). The timing system has been integrated fully into the EPICS control system software so that timing parameters are available as EPICS channels. All the event receivers have a timestamp counter (40 ns precision), which is latched when an event occurs.

12.5.2 Diagnostics

Fine integration of the Diagnostic devices into the control system has a great importance for operation and commissioning of the machine. Almost all these devices need to be synchronized with the beam operation, which is done by the proposed timing system.

Beam position monitors and current transformers (fast and slow) are the most important diagnostic tools and are monitored and controlled by number of ADC modules with 16 bit precision and also digital I/O modules.

Several screen and synchrotron light monitors are/is used to monitor the profile and presence of the beam in transfer line and ring, live on a dedicated monitor. All these cameras are synchronized with the beam injection and machine operation and connected to a video multiplexer, which will be placed in the control room and provides selection of each of the screen monitors at a time. Furthermore we will use a very cost effective PCI image grabber installed on a PC in control room, which provides us with digitised images and is available over the network for further analysis by physicists. One scraper is also used in the transfer line from booster to storage ring.

12.5.3 Beam Line Controls

An important aspect of SESMAE Beam lines is the fact that they benefit from the same EPICS based control system as the machine. This involves both hardware and software and allows smooth integration of several sub-systems (Monochromators, Insertion Device, Beam Diagnostics, etc.). The position and functionality of the VME crates follow the Beam line topology (typically front-end, Monochromators, Beam line optics, experimental station 1, 2...), see figure (12.8) for a typical Beam line structure

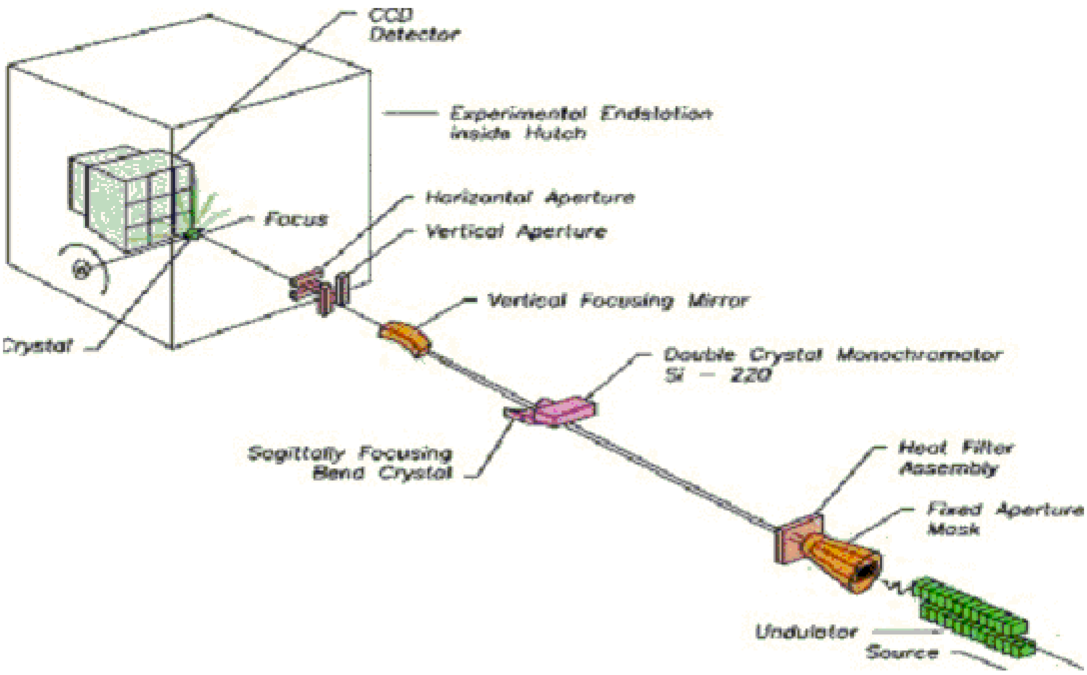


Figure 12.8: Typical Protein Crystallography Beam line Layout (courtesy SER-CAT, APS).

The bulk of custom Beam line control software is taken from synApps package (a bundle of compatible versions of selected EPICS software – wherever developed – in a single tar file that can run a Beam line)

. It includes software support for Motors, Scalers, Optical Tables, Slits, Multidimensional Scans, Multichannel Analyzers and miscellaneous devices.

The support for motors (both stepping and servo motors) provided by the EPICS motor record, fits well with the Oregon Micro Systems VME58 family intelligent motion controllers, proposed hardware for motors control and interface, figure (12.9) shows a typical GUI for the motor record showing all the different parameters that can be controlled.

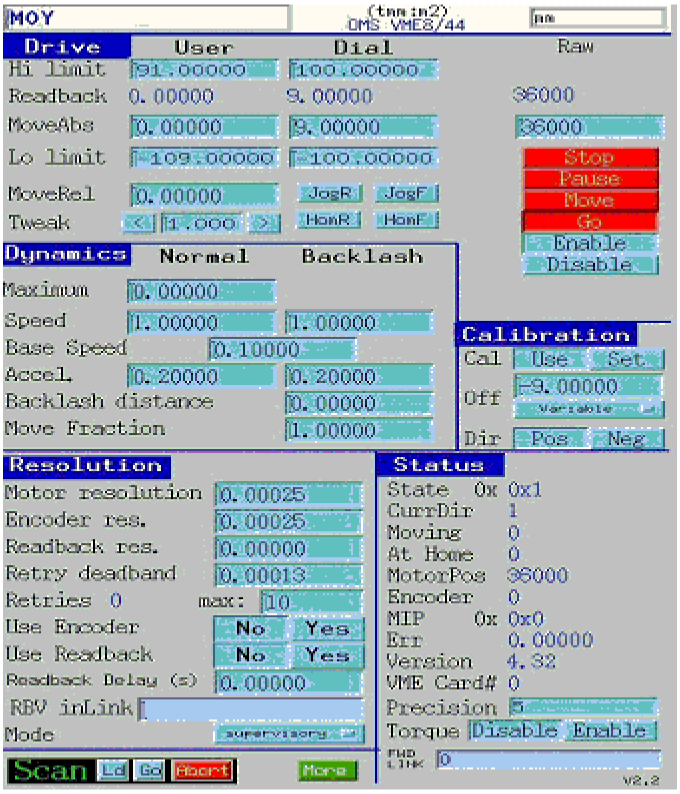


Figure 12.9: A Motor Panel.

A Monochromator, a Slit, an Optical Table or a Mirror Unit, can be considered as a collection of motors. They can be driven according to special tables or analytic expressions defined by EPICS *calc* records as shown in figure (12.10).

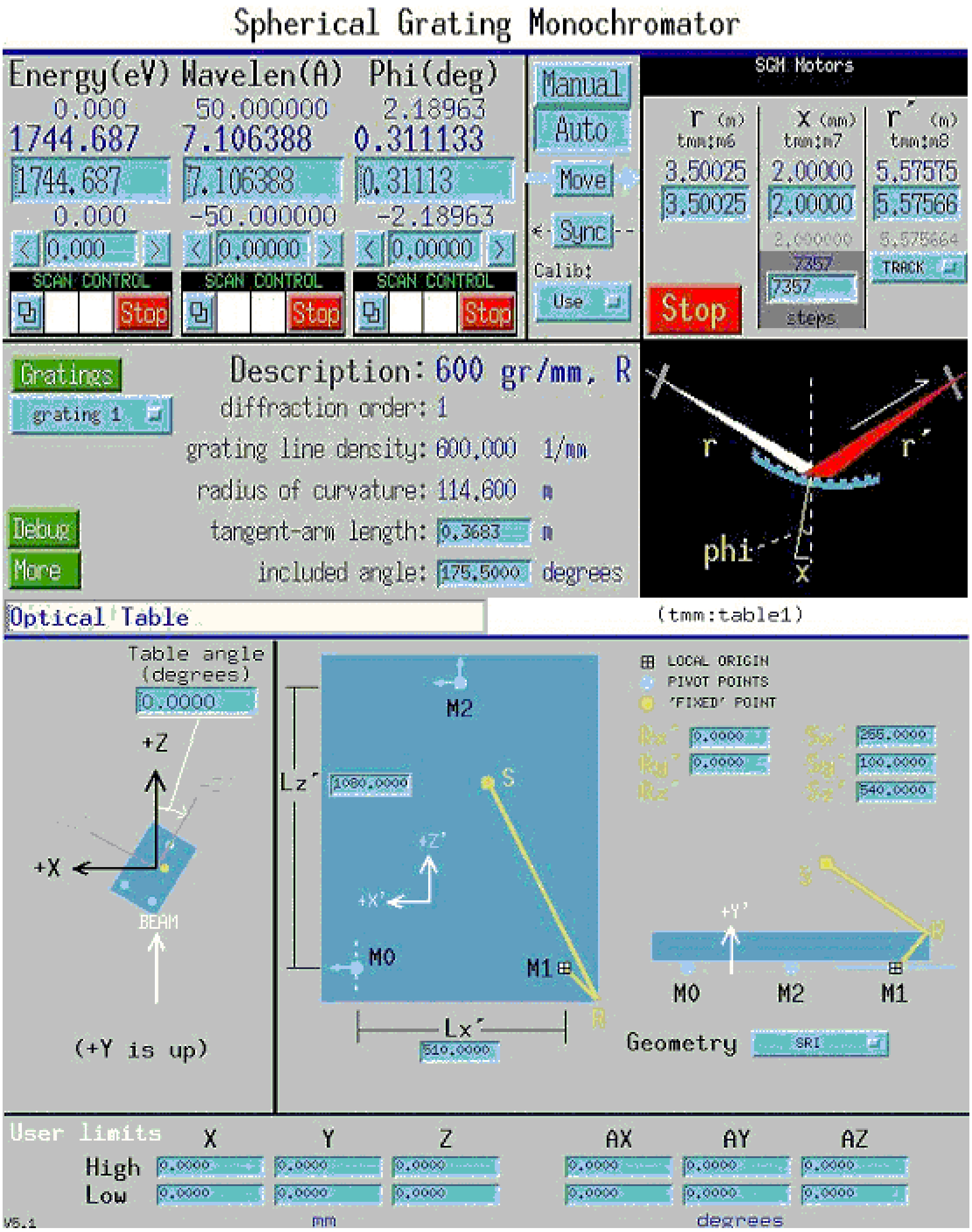


Figure 12.10: A Monochromator and an Optical Table GUI.

Another very useful support from synApps is the so-called *scan* EPICS record. It allows for a dynamic configuration of various types of scans based on EPICS process variables. This feature is of particular interest for Beam line commissioning where one can perform a run-time configuration of 1D/2D scans with several positioners detectors and detector triggers. The scan results can be automatically saved from the VME IOC to a file server or plotted on line while the scan is going on using custom IDL clients like scanSee and Catcher. *saveData* utility from the synApps package will save scans data to the file server. Simply put, one can perform simple or sophisticated scans without involving coding on some clients. See figure (12.11) that show an EPICS scan record display with all the parameters that can be controlled.

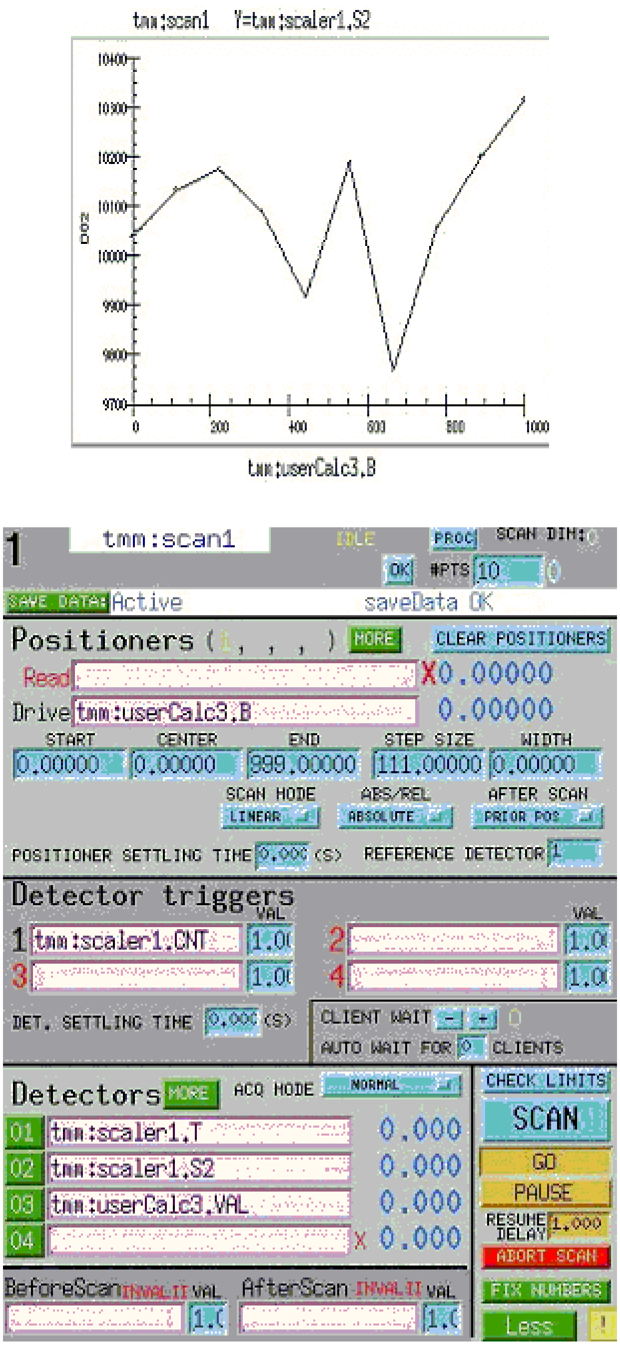


Figure 12.11: A Scan Panel.

The generic straight forward solutions provided by EPICS help the Beam line operators actively participate on the commissioning and operation of the Beam lines with no or minimal knowledge about EPICS. The goal is to keep this philosophy valid also for the experimental stations.

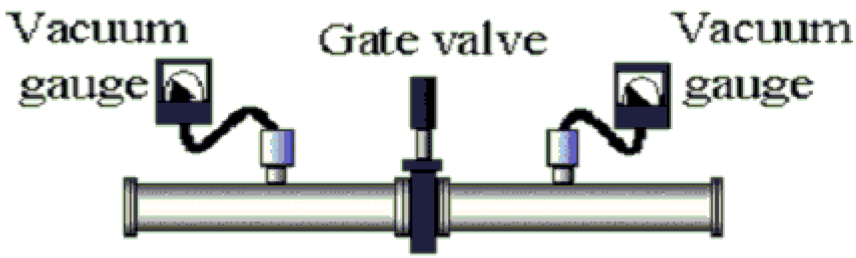
Date Storage and handling has high priority in the setup of the Beam lines in terms of resources. Security, availability of storage means, and ease of use are the major proposed features. This can be achieved by a dedicated file server for each Beam line, consoles equipped with high capacity storage hardware and interfaces (Optical, Tape, FireWire Drives), private user accounts and a secure network configuration.

12.5.4 Equipment Safety System

It should be noted here that the safety systems are not part of the control system. The equipment safety systems that are responsible to protect the equipment from physical damage are in the responsibility of the equipment groups. Such a system will typically be implemented with a programmable logic controller (PLC), which is a very secure and fail-safe type of computer. The control group could do the programming, but it is usually simpler to outsource it to a local company, because such type of programming is very common in industrial applications.

The personal safety system must be specified the person that is responsible for safety. Also this system is usually outsourced, because it must be accepted by the local health or work safety agencies.

Equipment Safety
Interlock



Personal Safety
Radiation Monitoring
Access Control and Interlocks

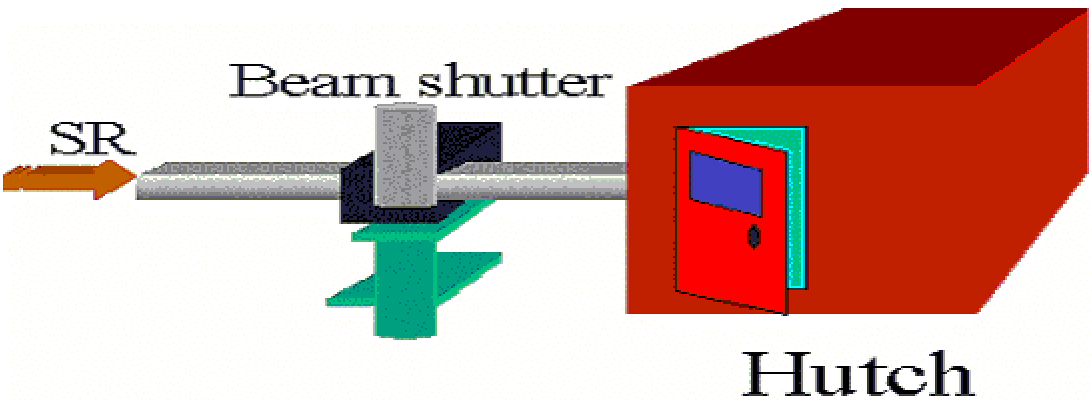


Figure 12.12: A very good example of personal safety system and interlocks. Whenever a person tries to enter the Hutch in the presence of the synchrotron light (SR) the beam shutter should act immediately and be closed.

This of course should be totally independent of the control system for safety reasons.

12.6 The Control System for the Pre-injector (Microtron)

The control system for the pre-injector of SESAME consists of controls for the different subsystems of the Microtron.

This Microtron is a classical one coming from BESSY I and it is going to be installed and used as it is but with a completely new control system.

Figure (12.13) shows the structure of the Microtron.

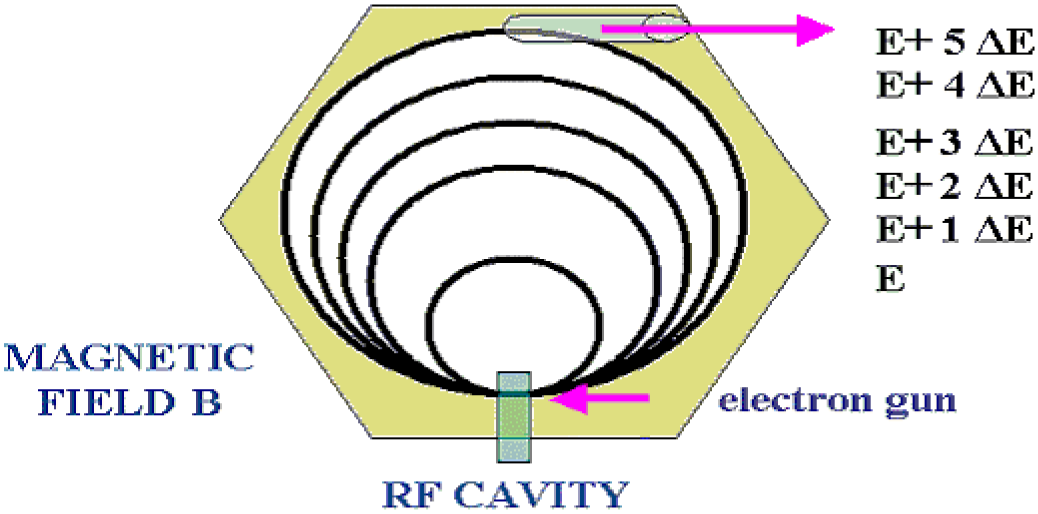


Figure 12.13: Schematic of the classical Microtron

This Microtron like most of the accelerating systems includes RF, timing, power supplies (PS), vacuum and some diagnostics in general.

12.6.1 Power Supplies (PS)

A constant and uniform magnetic field B guarantees that the electrons travel through a circular path and come back to the same point in the RF cavity and get accelerated again until they reach the final energy (22 MeV). Such a magnetic field is provided by one main bending magnet and four Trim-coils to adjust the imperfections of the main bending field. So there are total of 5 power supplies to be controlled. We have decided to control all the power supplies of SESAME in the same way and using the same type of controllers.

We have been considering three options for the interface of the power supplies to the control system.

12.6.1.1 Digital PS with Fibre-Optic Bus Approach

The first option is the SLS approach, which is to take the same PS controller being used at the SLS developed at PSI (Paul Scherrer Institute) and Industrial Pack modules (FPGA programmable) that sits on a VME carrier board and control system talks to the PS controller via these modules and through a pair of fibre optic links for each PS. The good thing about this approach is that everything is ready but it always needs PSI support, see figure (12.14).

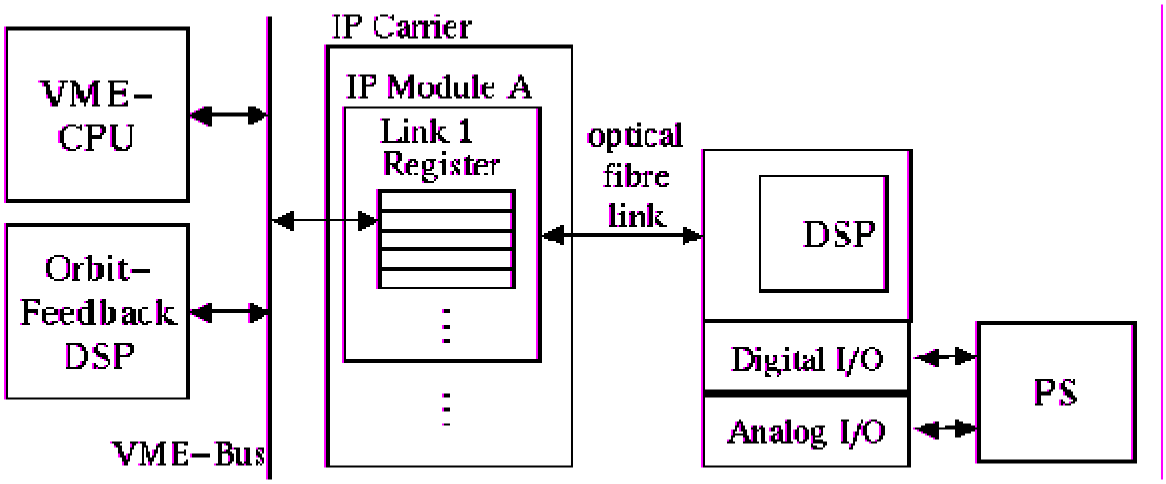


Figure 12.14: SLS Power Supply CS.

12.6.1.2 CAN Approach

The next approach is to connect PS controllers to the control system (VME crates) via a CAN-bus. In this approach one or more CAN-bus controller sits on a VME carrier board and communicate to several PS controller with CAN protocols. The driver and device support for EPICS already exist and can be found from other labs (e.g. DELTA in Dortmund, BESSY in Berlin) free of charge but the PS controller should be designed to have CAN interface and a local memory to save the ramping waveform, see figure (12.15).

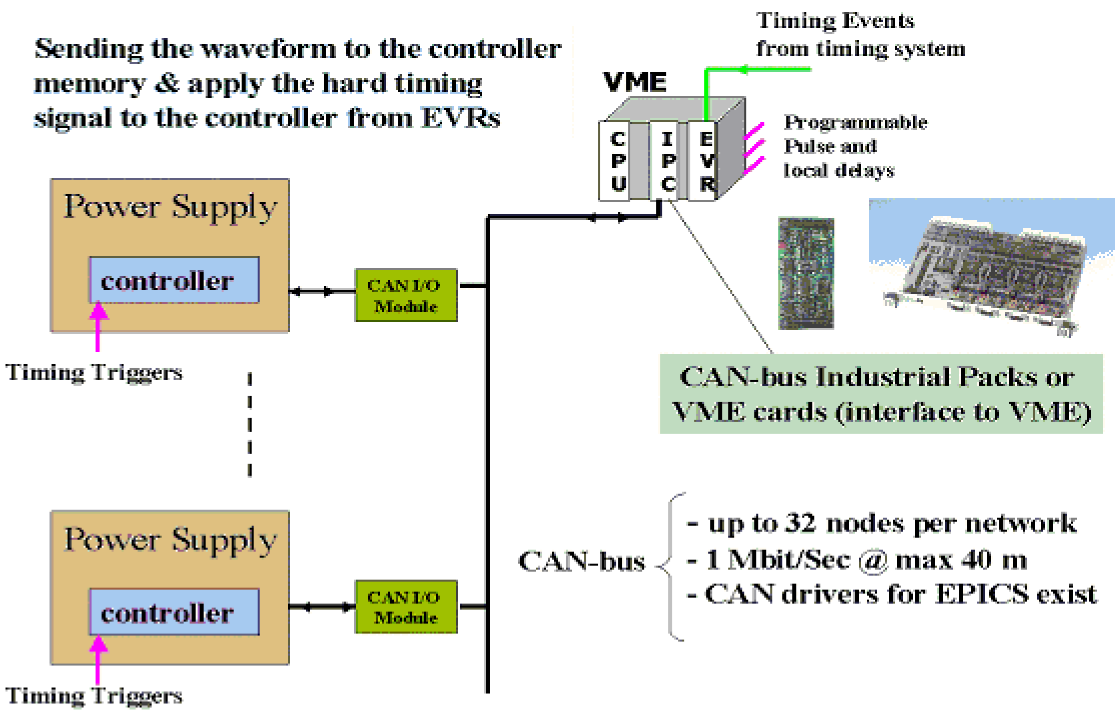


Figure 12.15: Can-bus PS controller

12.6.1.3 Ethernet Approach

The third option connects the PS controller directly with Ethernet and consequently with TCP/IP. The company Cosylab has developed a new card of the size Europacard (16cm x 10cm), i.e. 3U (half VME height), which contains a microprocessor running Linux and a CPLD (similar to an FPGA in functionality - it is programmed with VHDL language). The card is much cheaper than a VME equivalent, because it uses a mass-produced embedded micro controller. The digital interface to the PS controller is implemented in the CPLD. Alternatively, the CPLD drives a DAC and ADC and thus interfaces to the PS controller through conventional analog signals. In addition, ramping would also be done with the CPLD, which has also reserved inputs for the timing pulses.

The Linux microprocessor has 16-64 MByte RAM. It already has a 100 MBit Ethernet interface so it talks directly with TCP/IP. One can use the standard EPICS port to Linux. Then one has an EPICS IOC built into each power supply. There is no need to interface it to VME, because it is a completely independent IOC.

It is difficult to choose the best solution from these. The third option is definitely the most farsighted, as decentralizing the control system, brining intelligence as close as possible to the controlled devices is the trend in the controls world. With EPICS 3.14 on the market + (> 100 Mbit Ethernet), a small embedded controller running Linux (low memory/cpu overhead) loading iocCore is the trend/future in EPICS community (as PC104). The two main factors we have to consider are price and needed manpower. As the technology and price changes constantly, we will make the final decision at the last possible moment.

12.6.2 Radio Frequency System

The RF system of the Microtron is basically controlled by a dedicated circuitry called Pulse Forming Network (PFN), which produces and manages all the required signals and waveforms automatically, however, there are a small number of the signals that has to be controlled and monitored to get the RF system running.

There is a high voltage in PFN to be set and monitor, which is done by normal analogue input/output control. For the Magnetron we need to find at what frequency we have the maximum voltage. To find this frequency we monitor the signal by a spectrum analyser and then set the frequency to that of maximum amplitude. Later on this can also be done automatically by the control system through reading of this frequency by GPIB connection to the spectrum analyser and set the frequency of the magnetron accordingly.

We have to measure and monitor the temperature of the cooling water of the RF cavity too.

There is an auxiliary gun working at 75 kV and provides a regulation current of 20 to 30 mA for the gun. This regulation current has to be set by the control system (analogue output).

The PFN needs to be triggered externally. The triggering is done by the timing system of the accelerator. The requirement of this signal is a 1 microsecond TTL pulse with a frequency of 1 to 10 Hz depending on the repetition rate of the injector.

12.6.3 Vacuum Controls

Control of the vacuum system is a simple one in this case. It includes 2 pumps each with one gauge. One valve needs also to be opened or closed. For each pump and the valve we need digital output to set them to OFF or ON state and also corresponding digital inputs to monitor the status of the pumps and the valve. Each gauge needs also to be monitored, which is done using an analogue input control signal.

12.6.4 Diagnostics

There is a beam current monitor that shows the total current of the Microtron. Other diagnostic devices are an F-CUP (Farady cup) to measure the total charge coming out of the electron gun and also a screen monitor (SM) to look at the beam during the commissioning. A digital oscilloscope monitors the charges caught by F-CUP. The SM is driven by a pneumatic system, which needs digital inputs and outputs to be monitor and controlled.

12.6.5 The Timing System

The required timing signals for the Microtron are produced by the timing system of the whole accelerator using the event system.

To do so we need a pair of event generator (EVG) and receivers (EVR).

The EVG is the main one, which the master event generator that generates and sends event streams to all event receivers of the system (see figure (12.16)).

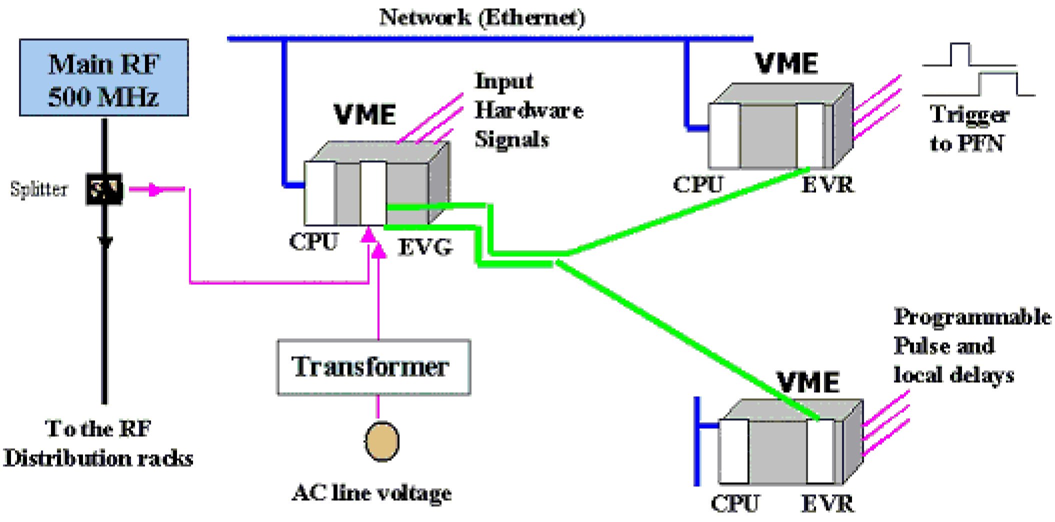


Figure 12.16: the timing system hardware.

12.6.6 Other Controls

There are total of 3 stepper motors to be controlled in the Microtron. Two of them drive a pair of axial and radial actuators to minimize the amount of the reflected RF power. The other motor moves the kicker rod and adjusts its position to get the beam out of the Microtron successfully. Dedicated motor control cards and software by EPICS take control over these stepper motors.

12.7 Estimated I/O

AI: Analogue Input, **AO**: Analogue Output, **D**: Digital (Binary).

	VA	DI	PS	RF
AI		1	5	2
AO			10	2
D	3	2	5	
Serial	10			
Motors				3
Timing		1	1	3

The Control System Hardware for the Microtron is coming from the Swiss Light Source as a donation to SESAME project:

12.8 Control Room

A central control room will operate the entire machine, operator stations with the same facilities provided, in addition to some special purpose instruments like oscilloscope and spectrum analyzers.

12.9 Network

Will support the control system, data acquisition and offices, it has to ensure a reliable and fast data transfer without bottlenecks or jams, based on fast Ethernet technology (100 Mbit/s), this will cover a wide range of supported VME interfaces, good performance and low cost.

A file server is needed where a central point of data collection and managing is provided, Development and Boot consoles in addition to port servers or serial cards (connect VMEs to hosts) are also required.

References

[1] Bob Dalesio, *"1999 Experimental Physics and Industrial Control System (EPICS)"*, Overview.
[2] Stephen A. Lewis, April 2000, "*Overview of the Experimental Physics and Industrial Control System: EPICS"*.
[3] Steven Hunt, Swiss Light Source, 1999, *"Requirements of the Control system"*.
[4] Canadian Light Source, Aug 2001, *"Control System Technical Specification"*.
[5] ANKA CS Web pages: http://kgb.ijs.si/KGB/Docs/Part%20I.%20User's%20Manual/1.%20Basic%20Concepts%20and%20Features%20of%20the%20Control%20System/Basic_Concepts_and_Features_of_the_Control_System_for_Anka.htm
[6] M. Plesko et al., "*The Object Oriented Approach to Machine Physics Calculations with Java Technology"*, in Proceedings of the ICALEPCS 2001 Conference, San Jose, 2001.
[7] D. Vermeulen and others, *"The SLS Beam lines Data Acquisition and Control System"*, Swiss Light Source.
[8] Tim Mooney, *"EPICS at APS Beam lines"*.