The SSRL Injector Control System*

C. Wermelskirchen†, S. Brennan, T. Götz‡, W. Lavender, R. Ortiz, M. Picard‡, J. Yang
Stanford Synchrotron Radiation Laboratory, P.O. Box 4349, Bin 69, Stanford, CA 94309-0210

Abstract

The control system for the new SSRL injector for SPEAR is based on one central VAX/VMS workstation with additional workstations as consoles and standard CAMAC equipment for data transmission and hardware interfacing. The CAMAC crates are connected to the central workstation by using a serial highway.

The central computer has the data base that describes all the accelerator hardware characteristics and the actual accelerator status information. The application software is built on top of the central data base and is independent of the specific hardware characteristics.

The operator interface uses DECWindows (X-Windows) and provides menu driven access to all machine parameters and control applications. The application programs can react on parameter changes on an event driven basis.

I. Introduction

The new SSRL injector is a 10 Hz machine especially designed to accelerate electrons to energies up to 3.0 GeV to fill one or several buckets in SPEAR up to intensities of 100 mA for the synchrotron light experiments. The new accelerator includes a 150 MeV linac and a 3.0 GeV synchrotron [1] [2]. The control system for the injector was designed to make the operation of the accelerator user friendly, keeping in mind that the machine has to be operated from two different places (as stand–alone machine as well as from the SPEAR control room). The decision was made to interface all the digital and analog signals to a central computer by using standard CAMAC equipment without local intelligence. A VAX Workstation 3500 was chosen as the central control system computer, handling CAMAC as well as being the local operator console.

II. Control System Kernel

The control system kernel is made out of several modules (see Figure 1) that access the data base, the communication with the front end system (CAMAC), the synchronization between different applications and the notification mechanism for parameter changes.

A. Data Base

The data base itself is kept in a shared memory section that is mapped from a disk file and written back to that file every minute to prevent loss of data in case of a system failure. A separate process, which is activated during

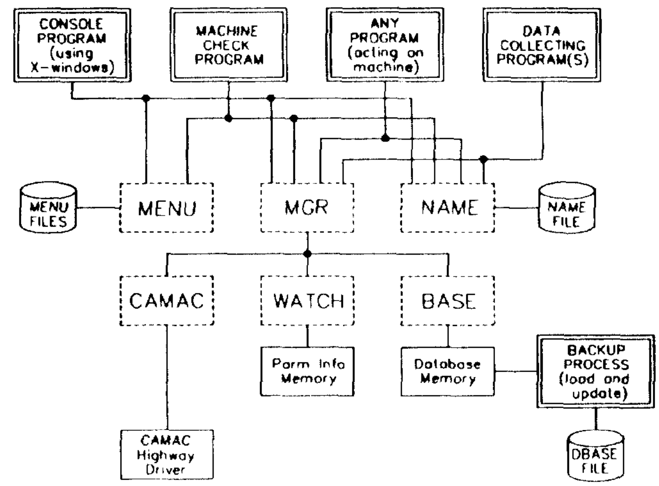

Figure 1
Control System Kernel Modules

system startup, is responsible for this task. Whenever it is activated it also reorganizes the data base to remove space from deleted devices. All other processes access the shared memory data base by using subroutines from the BASE package.

The data base keeps all information about hardware characteristics and actual parameter settings. It contains records that describe *devices* and each device contains all the *parameters* for that device. Each parameter has several fields, for example minimum/maximum allowed value, actual value, dimension, conversion data, addressing information, etc.

Parameters are handled in the following way: The direct physical parameter, for a power supply for example, is the output current. It is stored in the data base in exactly the form that is used by the hardware module (DAC, ADC). Data base routines transform the parameter to its floating point value, which represents the real current, for application programs and menu representation.

But parameters are not limited to direct physical values, they can also be linked to other parameters. In this case these parameters do not have a value by themselves, instead the value is taken from another parameter by using additional transformations. The data base access routines handle *simple* conversion of parameter values (factors, offsets) or they can call additional routines for special conversions or relations between different parameters. This way, for example, a parameter that defines the energy of a beam line can actually set the current of several magnets at the same time.

Currently there are some 300 devices defined in the data base with more than 1400 parameters total.

*Work supported by the Department of Energy, Office of Basic Energy Sciences, Division of Material Sciences.

†Now at Gesellschaft für Mathematik und Datenverarbeitung (GMD), Bonn, Germany.

‡Now at Physikalisches Institut der Universität Bonn, Germany.

B. Maintenance Shell

All entries in the data base can be created, modified or deleted by using the maintenance shell, a program that can be run on any terminal. Devices and their parameters are described in normal text files and this information can be imported into the active data base. Devices can also be deleted at any time. The maintenance shell allows to interactively modify all characteristics of the devices and parameters.

There are tools included to generate different kinds of listings and reports to view the information in the data base.

C. Application Interface

The data base routines and the CAMAC interface software make the lower level of the central control system software. Between this layer and application programs there is the application interface layer. It completely separates the hardware dependencies of the machine parameters from the applications. For the programs on top of the application interface there exists a *virtual* accelerator, represented only by parameters and their values. These values may be *simple* values, like magnet currents, or they may represent more complex parameters, like beam line energies. For the application programs there is no difference between these parameters, they can be read and set, independent of the number of actually involved parameters that may be affected by setting a complex parameter.

The application interface also provides *real* names for all devices and parameters. Internally these are handled by using binary IDs. Externally there are names according to a defined scheme and there are modules that translate between these two representations. Each name is composed from four parts: The name of the accelerator or section to which the device belongs, the device name itself, the name of the parameter and the characteristic of the parameter. GUN_RF.PHASE_SHIFTER_AS, for example, is the analog setpoint (AS=characteristic) of the phase shifter (PHASE_SHIFTER=parameter name) for the RF (RF=device) of the gun (GUN=section).

Another important function that the application interface provides is a notification mechanism. Application programs can register themselves to be notified whenever a particular parameter is set by any program. This function provides the fastest way to react to changes of the value of any parameter and is used by all realtime applications, like the operator interface, the beam position readout program, or the waveform generating program.

III. CAMAC

The CAMAC crates are connected through a serial highway to a KineticSystems 2060 CAMAC driver module. The KineticSystems CAMAC library and a modified version of their VMS driver is used to execute the CAMAC commands. There is one CAMAC readout program that knows how to collect the data from the (standard) CAMAC modules (ADCs, digital inputs). It gets it's addressing information from the central data base and reads all the CAMAC modules every 200 msec. It detects changes in the parameter values and feeds the changed values into the data base. This also means that they are propagated to the applications by using the notification mechanism.

The readout program also monitors all the CAMAC crates and modules and invalidates values of parameters that are no longer available because of a CAMAC failure (powered down crates, removed modules). So the operator or application programs will get an identification about some CAMAC malfunction. The program tries to address lost modules/crates in certain time intervals and when it becomes available again, it will automatically be recovered by the program. The modules will be initialized to their previous settings and data will be collected again. This makes replacement of modules very easy, because after turning on a CAMAC crate again, normal operation is continued after only a short recovery time.

So far there are more than 40 CAMAC modules handled by the readout program. From these CAMAC modules about 600 words are read every 200 msec and more than 150 parameters are being set during a complete CAMAC initialization.

There are also some special programs that handle (non standard) modules like beam position monitors or waveform generators. These applications will be described in separate articles [3][4].

IV. Operator Interface

From the beginning the operator interface for the new SSRL injector was designed to be easy to use, be usable on workstations at different locations at the same time, be easy to adapt to the changing operator needs. A lot of final work could only be done during commissioning, when the entire control system had to be operational.

The hardware basis for the operator interface are VAX workstations that use X-Windows (DECWindows) as graphics standard. The operator interface was therefore made on top of DECWindows. Because X-Windows already has the complete networking functionality in it, all operator controls are accessible on all workstations in the network and no modifications have to be done to connect additional consoles.

The operator is controlling the injector by using *menus* which present the values of the machine parameters and allow the operator to manipulate these values. The menus are displayed in windows on the workstation screen and can be placed and arranged in any way in which the operator needs them. From one menu the operator can pop up additional menus, so it's possible to have additional parameters available to control areas or devices in greater detail.

In general a menu is made of graphic elements. These can either be static, like texts, lines or colored blocks, or be dynamic. Dynamic elements are connected to the parameters in the data base and can be used for input and output of parameter values. They are updated whenever the value of the parameter changes or is modified by any other program. The main dynamic elements are horizontal and vertical sliders and bars. The resolution of the horizontal input slider is switchable so that the operator can change the precision of the adjustments between a zero to hundred percent input and high resolution, where a pixel on the screen affects the least significant bit of the parameter.

The operator modifies parameter values and thus controls the machine by using the workstation's mouse as input device. He can either set parameters to fixed values by clicking at fields on the menu or he can drag sliders by using the mouse and so set parameters continuously to new values.

Several options to start/stop additional programs are available through pull down menus from the main menu. These are used to activate the standardization of beam lines, start the linac feedback program, etc. Another option allows to save or restore a complete machine configuration into separate disk files.

To create new menus or to modify existing menus there is a menu design program that also runs on the workstation. It is an interactive program that allows to manipulate all the graphic elements. They can be added, moved, changed, deleted, etc. The foreground and background colors of these elements can be set to 6*6*6=216 different colors, which is enough for the control system application and will allow to work with 8 color planes without problems.

V. Applications

Several application programs to handle complex settings or sequences of settings of parameters have been implemented by different people. Subroutines have been added to the data base routines that allow the setting of complete beam lines to different energies, affecting all magnets in these beam lines, using measured calibration data of the magnets. This is done whenever the operator uses the *energy knob* for such a beam line.

There are other programs running in a background mode, to perform tasks that take some time and can not be executed for each change of a certain parameter. One of these programs handles the standardization of magnets to reproduce the magnetic fields very precisely, once a beam line has been set to a new energy. This program is started by operator request from the main menu.

Another program stabilizes the linac for drifts in the electron energy. It gets the position of the electron beam after the first bend magnet downstream the linac from the data base and changes the linac parameters so that the position of the beam is kept, which means the electron energy is stabilized[5]. Once started, this feedback program performs its task without further interaction.

The program for the generation of the RF wave form is more interactive. It waits for notification of modification of some special control parameters, like phase, offset, amplitude by the operator and then recalculates a complete new table of values for the RF wave form table parameter. This table parameter is then loaded into the CAMAC module as part of the normal application interface function and is also displayed in a menu, so the operator will see the result of this calculation.

These programs are all coupled to the menus by either using the application notification function for watching parameters to perform their task or they can be started/stopped from the pull down menus of the main menu.

VI. Conclusion

The control system was operational when the first tests during the commissioning of the new injector had to be made. Not all devices were connected to it at that time and some devices were controlled locally to do special tests with machine components. During commissioning all devices have been connected to the control system and are monitored and controlled from the menus. Operators can use the local console as well as the remote consoles in the SPEAR control room.

The new SSRL injector is successfully being operated from the workstations, the menus are easily usable and have been adopted to the different operator needs. The injector reached its goal to produce an electron beam of designed intensity to fill SPEAR within several minutes.

VII. References

[1] H. Wiedemann, "3 GeV Booster Synchrotron for SPEAR," ACD-Note 45, SSRL, March 1987.

[2] H. Wiedemann, "3 GeV Injector Synchrotron for SPEAR." In these proceedings, 1991.

[3] S. Brennan, S. Baird, W. Lavender, H.-D. Nuhn, C. Wermelskirchen, and J. Yang, "The Control and Operation for the Programmable Waveform Generator of the SSRL Injector." In these proceedings, 1991.

[4] W. Lavender, S. Baird, M. Borland, S. Brennan, R. Hettel, H.-D. Nuhn, R. Ortiz, J. Safranek, J. Sebek, C. Wermelskirchen, and J. Yang, "The SSRL Injector Beam Position Monitoring Systems." In these proceedings, 1991.

[5] L. Emery, "Energy Feedback System for the SSRL Injector Linac." In these proceedings, 1991.