# A CONTROL SYSTEM UPGRADE OF THE SPEAR SYNCHROTRON AND INJECTOR*

R. Garrett, S. Howry, C. Wermelskirchen, J. Yang

*Stanford Synchrotron Radiation Laboratory*
*Stanford Linear Accelerator Center*
*2575 Sand Hill Road*
*Menlo Park, CA 94025*

## ABSTRACT

The SPEAR electron synchrotron is an old and venerable facility with a history of great physics. When this storage ring was converted to serve as a full-time synchrotron light source, it was evident that the time was due for an overhaul of the control system. Outdated hardware interfaces, custom operator interfaces and the control computer itself were replaced with off-the-shelf distributed intelligent controllers and networked X-workstations. However, almost all applications and control functions were retained by simply rewriting the layer of software closest to each new device. The success of this upgrade prompted us to do a similar upgrade of our Injector system. Although the Injector was already running an X-windows-based control system, it was not networked and Q-bus-based. By using the same Ethernet-based controllers that were used at SPEAR, we were able to integrate the two systems into one that resembles the 'standard model' for control systems, and at the same time preserve the applications software that has been developed over the years on both systems.

## INTRODUCTION

SPEAR was originally commissioned in the early 1970's as an electron-positron collider, and, when the PEP ring was built at SLAC, the control system at SPEAR was upgraded by installing a version of the PEP system [1]. The PEP system used custom-built CAMAC hardware which had a parallel crate architecture. This system used a Grinnell graphics processor to provide the operations displays and the background display for the touch-panel controls. This processor provided primitive graphics and text with color capability. The PEP system also used Modcomp computers to enable distributed control to the CAMAC crates due to the physical separation of the control room from the various parts of the PEP ring. The Modcomp system was not required for SPEAR because of its more compact physical layout. The port of the PEP system to SPEAR was done in the early 1980s and ran the SPEAR ring for more than ten years using VAX 11/750 and 11/780 computers. In 1990, the SPEAR ring became a dedicated synchrotron radiation source operated by the Stanford Synchrotron Radiation Laboratory.

The SPEAR Injector was built in 1989 and used a control system that was transferred from a system developed at the ELSA ring in Bonn, Germany [2].

## REQUIREMENTS AND OPTIONS

Our motivation for doing the upgrade of SPEAR (and then using that experience to upgrade the Injector as well), was to replace a couple of very old systems which were either already obsolete and unsupported, or soon would be. The two systems that were of most concern were the VAX 11/780 that ran the control system software and provided the UNIBUS interface to the CAMAC hardware and the Grinnell graphics display processor.

The 11/780 was already obsolete and very expensive ($15k/year) to maintain and was an obvious candidate for replacement. The problem was that the current interface to the CAMAC hardware was via a proprietary UNIBUS-based controller called the VAX CAMAC Channel (VCC) [3], a SLAC built device. Since none of the modern computer systems currently available supported the UNIBUS, it was going to be necessary to replace the VCC as part of the upgrade.

* Supported in part by the Department of Energy, Office of Basic Energy Sciences, High Energy and Nuclear Physics under Department of Energy contract DE-AC03-76SF0015

The Grinnell graphics processor drove the operator displays and the touch-panels. It provided only primitive graphics capability and the manufacturer of the hardware had gone out of business many years previously, making maintenance difficult as spare parts became scarce.

Our basic requirements for replacing the existing system were to:

- Provide CAMAC performance no worse than the current system which operated at about 3 Hz.
- Provide an X-windows-based graphical user interface (GUI).
- Continue to provide a capability to knob operator-selected parameters.
- Eliminate the VAX 11/780 and Grinnell processor.

Several proposals were submitted on how to meet these requirements and a design review was held in November 1992. The proposals submitted were to:

1) Obtain a VAXBI based computer and use a specially built fiber optic interface (which was similar to units being built for the SLC project at SLAC) to connect to the CAMAC crates.

2) Expand and install the control system used at our Injector to SPEAR.

3) Replace the current VCC and crate controllers with VCC emulation software and MicroVAX crate controllers from Kinetic Systems.

All proposals involved replacing the Grinnell system by converting the Grinnell graphics calls to X-windows equivalents and replacing the old touch-panels and color monitors with X-terminals.

The result of the design review was to choose the MicroVAX controller option.

## THE UPGRADES

*Hardware -- Crate controllers*

We replaced the outdated SPEAR VCC interface and aging parallel crate controller with an Ethernet-based distributed intelligent controller. The advantage was that the Ethernet solution leaves the SPEAR control system open to the option of non-CAMAC hardware such as VME.

The Kinetic Systems model 3968 was adopted for both SPEAR and the Injector. The controller features a 2.7 VUP rtVAX 300 processor including both floating-point and advanced Ethernet co-processors, as well as 8 megabytes of RAM memory, two terminal ports, a CAMAC interface, and 8 kilobytes of dual-ported memory.

*Hardware -- Knob boxes*

The existing knob boxes at SPEAR were home-built and old, and the person responsible for maintaining them was retiring. They connected to the existing VAX system via an RS-232 link. After some searching, only one commercial unit was found that could provide similar capability and this was a knob box manufactured by Hytec.

The Hytec unit has only two knobs, as opposed to the four knobs on the old boxes, but is programmable in such a manner as to actually provide the capability of controlling twice as many parameters from one box as compared to the old knob boxes. These new units also use an RS-232 link, but are connected to an Ethernet based terminal server instead of directly to a VAX.

*Hardware -- Computers and X-terminals*

Once the decision to go over to an Ethernet-based control system, it allowed us to choose virtually any computer system that could run VMS to replace the VAX 11/780. Given our budget constraints, it was decided to get a VAX station 4000-90 system with two 1 GB hard disks and 128 MB of memory. This system provides more than 30 times the raw CPU power of the 11/780 and gives us the ability to do more compute-intensive activities locally, such as machine modeling, that would have been extremely slow or impossible to do previously.

It was also decided to provide several operator stations in the control room area using Tektronix X-terminals. The main operator station has an X-terminal with dual display capability, allowing the operator to display much more information in one place at one time. These units are all Ethernet-based.

*Software -- SPEAR*

As mentioned previously, the original SPEAR control system was built on top of VCC hardware that was interfaced to the CAMAC crates via a parallel data link and was connected to the VAX UNIBUS via the VCC (see figure 1).

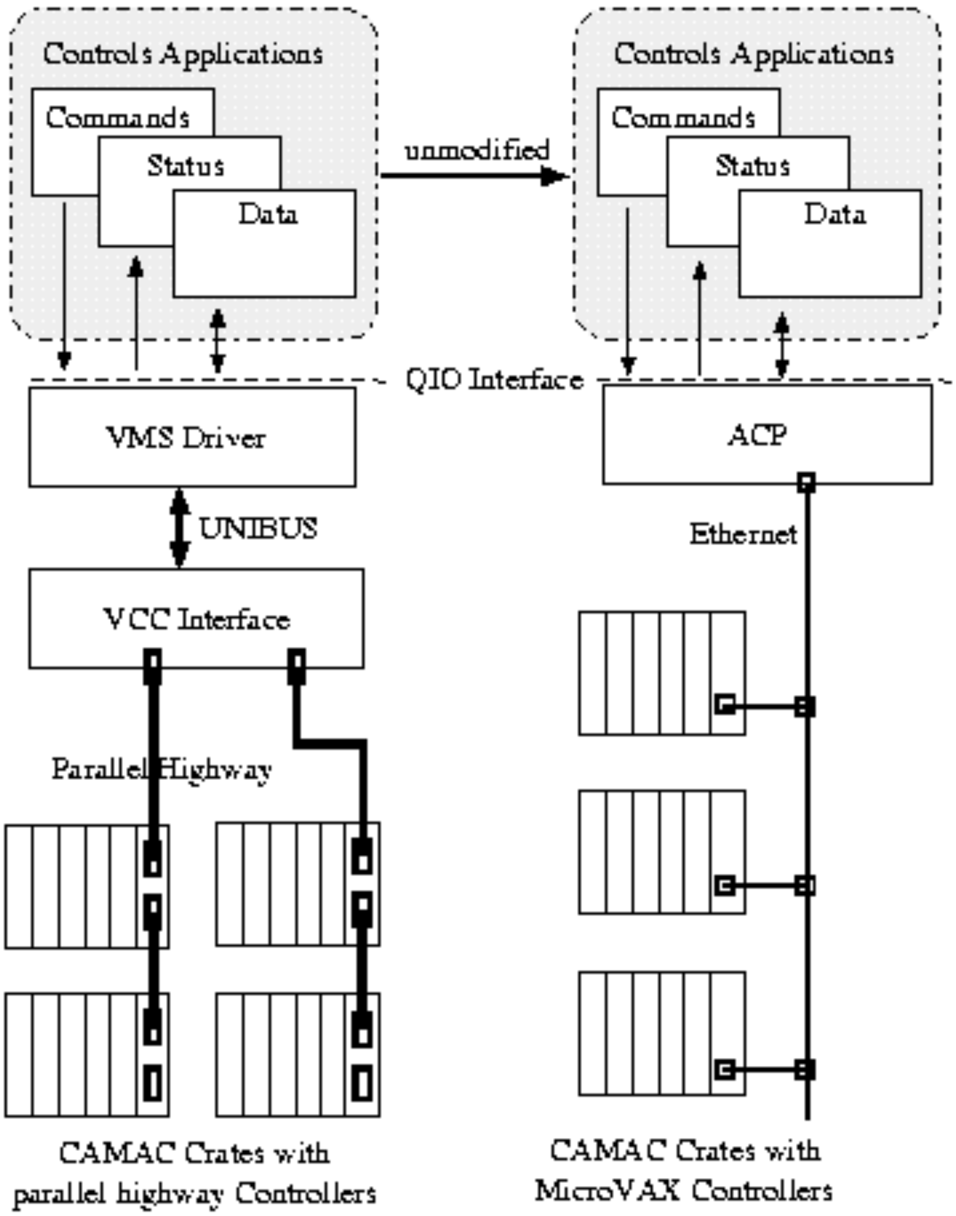


**Figure 1**
**Schematic of Old and New CAMAC Systems**

A VMS driver provided the operating system interface for the application software. All programs used this QIO device interface to execute CAMAC operations. The parallel data highway was organized in several branches (0 to 7), each branch connecting up to 8 crates (0 to 7). Each CAMAC crate was addressed by using the branch and crate number. An important decision for the upgrade project from the beginning was that the new CAMAC interconnect should keep the existing control application programs as they were, without any modification to the existing code. This was possible with the use of the well-defined QIO interface and provision of an identical device interface. For many years, SSRL ran application programs for the beam-line controls that used the capability of VMS to have an Ancillary Control Process (ACP). Such a process provides a QIO interface for application programs and handles input / output requests for these programs within a regular program [4]. This interface was chosen for the upgrade project because it yielded the flexibility for the new software to emulate the old CAMAC functions on the main host. The additional software overhead was tolerable as the new SPEAR control host (VAX station 4000-90) is about a factor of 30 faster than the old VAX 11/780.

The ACP on the central host now establishes network connections to all crate controllers and thus executes the CAMAC instructions. In more detail the CAMAC system now works as follows:

- An (untouched) application program generates a list of CAMAC commands, provides a data buffer for write and read data as well as a status buffer. This is done in exactly the same format as for the old VCC system.
- The application issues a QIO and passed these buffers to the operating system.
- The ACP gets these buffers.
- The ACP generates independent lists of CAMAC commands and write data for each crate addressed.
- If not done already, the ACP establishes a DECnet connection to each crate used.
- The ACP sends a packet containing commands and write data to each crate.
- The MicroVAX crate controllers receive the command and data packets.
- The crate controllers (in parallel) execute the given CAMAC commands, using the given write data.
- The crate controllers each build a list of status information and read data.
- Each crate controller sends a status and data packet back to the ACP.
- The ACP receives the status and data packets from the crate controllers.
- The ACP returns the status information and data to the application program's buffers.
- The QIO operation is complete.

As described, this new system fulfills the requirement of replacing the old hardware system without any modification to the existing application software. The critical aspect of this project was the real-time performance of the new system. This was difficult to estimate as the number of CAMAC commands per CAMAC list varies from a single command to up to 100 commands. At the same time, the commands in a CAMAC list can address an individual CAMAC crate or all crates at once. When first implemented, the ACP sent the command packet to the first involved CAMAC crate, waited for the response and only then sent the command packet to the next involved crate. As expected, the performance was dramatically improved after a revision, so that now all packets are sent and then the status and data packets are received. A measurement of the worst-case scenario, i.e., a single CAMAC command in a list, shows a total round-trip execution time of about 7 milliseconds. This is the minimum response time, but additional commands in the same list increase this time by very little. In the running system the ACP now collects statistical information about the data throughput to each CAMAC crate. This indicates that the average execution time for a SPEAR CAMAC operation is about 9 milliseconds. The performance of the Ethernet was also taken into account. To be independent of the global Ethernet traffic, the control system segment was isolated from the rest of the network by a LAN bridge. The current SPEAR machine operationrequires that the injected and stored electron beam be accelerated from 2.3 to 3.0 GeV. This is done over a time of about 3 minutes and it requires that all the main and corrector power supplies are ramped in a very synchronized procedure. In the old VCC hardware system this was guaranteed by the VCC hardware. In the new system, the commands for the different CAMAC crates are handled more independently by the ACP and there exists a potential for variations in the millisecond range. However, this problem has not been experienced.

Displays on the old control system were driven by a 1970's era video controller from Grinnell. This device accepted 16-bit commands to set position, draw pixels, lines, rectangles, and to manage a color lookup table that depended on the division of available planes among the different screens. Ours was configured to two black and white touch-panel screens and three 4-plane color display screens. A 'switch channel' command allowed programmers to send commands to selected screens. The device was shared, allowing (in principle) any application to write to any or all screens.

The replacement of this device was greatly facilitated by the modular, object-oriented flavor of the original control system software. In this design, all applications were event driven, quickly-executed 'methods' organized into concurrently executing VAX/VMS processes (APs). Each received event always contained the screen number(s) in its auxiliary data, so applications never had to make this decision. Events are scheduled at selected frequencies, or by operator touch-panel button pushes, or by something happening in the storage ring.

Further, each application communicates to the displays through a 'graphics layer'. This layer generates 16-bit commands from requests to draw standard (but limited) graphics items. The protocol is:

- initialize ( screen#, buffer );
- draw_item#1( data, buffer );
- . . .
- draw_item#2( data, buffer );
  . . .
- flush( buffer );
- return.

At 'flush' time, the buffer of commands was sent to the device as a single I/O operation (the layer automatically generates extra I/O operations if the buffer becomes full). The graphics layer is linked to each AP containing an application that requires graphics.

To replace the video controller device:

- Each screen of the device was replaced by an X-window client running on an X-server. The client process, which can also reside at any network node, receives buffers of 16-bit commands and converts them into Motif requests. It also maintains backing structures to redraw the window when requested to do so by the server. The structures are necessarily 'AP semantics-independent', built up only from the packets of 16-bit commands received since the last 'clear_screen' command.
- A 'dispatch' layer of software was inserted at the device end (where the I/O operation used to be invoked) of the graphics layer described above. This layer arranges for each buffer of 16-bit commands to be forwarded to the client process of the buffer's screen. The layer is linked to each AP that needs the graphics layer.
- No modifications were made to any of the AP's (but of course they had to be re-linked).
- A possible problem was the proliferation of network sockets if each AP could send to each client. We decided to reduce this by requiring the APs dispatch software to forward buffers to the 'master scheduler' process (the one that also manages the touch-panels; there can only be one of these at a node). This process, is the only one that talks over the network to the X-clients.
- Since things are executing asynchronously on many nodes, ring buffers had to be installed at several places. The most important spot is where buffers entered the clients, particularly if there are slow 'dumb X-terminals' installed as X-servers on the system. Another spot is where buffers from applications are received by the master-scheduler process.
- On the input side, physical touch-panels are not easily available, so button pushes are replaced with mouse (release) clicks. These mouse clicks from the X-server’s screen are received by the client and sent (over the network if necessary) to the Touch-panel Manager, wrapped to look like button pushes from the old touchpanel screens.

The new system automatically provides the following enhancements:

All screens can have color. At present 16 colors are available and deemed sufficient. The screen windows can be re-sized by the Motif window manager. Fonts look much better since the old device didn't even support lower case(!).

The number of X-servers executing at one time is limited only by the number of network sockets allowed. A single server can show windows of different screens. Two servers at different locations can monitor the action of the same screen.

The number of logical screens can be increased. Applications can be dedicated to screens, or they can be dynamically allocated to screens by the operator (an almost redundant feature, but it comes free!).

*Software -- Injector*

The successful upgrade of the SPEAR control system by using MicroVAX CAMAC crate controllers defined the upgrade for the Injector control system. The Injector also got rid of the CAMAC interface that was connected to the proprietary bus (Q-bus) of the MicroVAX 3600 on which the control system ran. All CAMAC crates -- for SPEAR and the Injector -- are now controlled by the same type of crate controller, thereby minimizing maintenance activities. These crates are connected through Ethernet, which eliminates dependence on a single manufacturer.

The Injector control system is much younger than the SPEAR system and therefore does not have an extensive history in the application programs. The design of the control system incorporated the feature of putting machine parameter values into the control system database only when the parameter values had changed. This was implemented by a data acquisition process that collected all input data from the CAMAC modules and applied the parameter values after filtering, i.e., comparing with previous values. The upgraded Injector control system now generates a configuration for each CAMAC crate from the control system database. This configuration is loaded into the crate controller CPU. The processor now runs the data acquisition within the crate; it filters the data and only sends a data packet through the network to the central host for the parameters that have changed. This significantly reduces the network utilization.

As the CAMAC interconnection was already isolated from the application programs before the upgrade, the change in data management within the crate controllers did not affect the existing programs. By replacing the shareable library that addresses the CAMAC hardware, the control applications now "talk" to the crates through a network instead of the serial highway. The only other change made to the Injector was forced upon us when we ported the control system from the MicroVAX 3600 to a (somewhat) newer VAXstation 3138. The VAXstation was running a more recent version of VMS, necessitating a conversion of all the DECwindows routines to their equivalent Motif calls. The MicroVAX was showing signs of age and was becoming under-powered for the load we were putting on it.

## POST-UPGRADE IMPROVEMENTS AND FUTURE

*Alpha Port*

As always, whenever you make new capabilities available in a system, people will always find a way to exploit those capabilities beyond the capacity of the system to support them. In an effort to stay ahead of the game and to remain in synchronism with computer technological trends, we plan to port the entire SPEAR control system to a DECstation 3600 AXP. Although the current SPEAR computer is more than adequate for our foreseeable needs, we also realize that it will take a significant amount of time to port all the various SPEAR applications to the Alpha architecture and wish to make that transition before we are forced to make it.

For most of the applications it will be mainly a matter of compiling and relinking the code. However, the CAMAC and database interface codes will most likely need more extensive rework in order to meet the requirements imposed by the 64-bit architecture and the larger page size.

*Feedback / BPM Upgrade*

In the previous SPEAR control system, the central host had to execute the very complex CAMAC command lists to perform the beam-position measurement (BPM). The BPM data acquisition was all done in a single CAMAC crate. This crate controlled RF multiplexers, which selected a single monitor button out of the *24x4* signals, started a BPM signal processor and digitized the signal. From the raw data values the central host then calculated calibrated beam positions. After the replacement of the old VCC CAMAC hardware, the process of the BPM data acquisition and data calibration was moved into the intelligent crate controller. The MicroVAX controller now handles the modules in the CAMAC crate by itself. The dedicated software performs essentially the same steps as the old central host, but it does this in a much faster manner. Beside generating the raw button data, it also generates calibrated beam positions. The calibration functions were actually linked into the MicroVAX crate controller code as they were before. This was made possible when existing VMS FORTRAN subroutines were linked with the new C code that runs under the VAXELN operating system in the crate controller. The migration of the BPM processing into the new MicroVAX crate controller resulted in the following improvements:

- The data acquisition time to measure a complete orbit was improved from one non-averaged measurement per minute to about 1 measurement per second, each averaged over more than 1000 orbit readings.
- The central host receives calibrated data much faster without any overhead.

*SPEAR / Injector Merging*

Both accelerators now have identical CAMAC crate controllers. The functionality, that is, the type of commands from the central hosts, is still different. For the SPEAR system, lists of commands are exchanged and executed about 3 times per second. For the Injector system, the acquisition configuration is loaded once and the crates send data when needed. Of course, changes of parameter settings are possible at any time.

The two different software packages for SPEAR and Injector were merged together. Each CAMAC crate controller can now be addressed by the SPEAR control system as well as by the Injector system. To date this feature is not fully utilized. Care has to be taken that control settings have a single-master system.

*VME Integration*

Recently, a VME crate was added to the SPEAR control system. This crate will eventually be used for a feedback system [5]. As a migration path for the future, this crate is now housing DAC modules that control some SPEAR power supplies. The VME crate also has a MicroVAX controller that is connected to the SPEAR network. The software that runs in this controller emulates a CAMAC crate for the SPEAR control system and thus completely hides the existence of a VME crate from the SPEAR system.

In the future, more applications will be moved into the front-end crates. This will allow the central control systems to concentrate on more global management tasks as well as the operator interface. In the long-term, the different control systems will merge, as different individual systems can be seen as parts of a global system that are all interconnected and exchange data with each other.

*CDEV Compatibility*

The CDEV application programming interface (API) being developed at CEBAF and APS appears to have the potential to make the sharing of accelerator programs between various facilities much simpler. We would like to take advantage of this capability by modifying our control system so that it can transmit and receive data at the CDEV layer. We believe this can be accomplished with only about a one to two man-months effort.

## CONCLUSION

The new SPEAR control system has been running now for about 2 years and has been very reliable. We have seen no evidence of any network bottlenecks or other kinds of problems associated with Ethernet communications. We continue to make minor adjustments to the crate software to improve performance and to support new functions.

The Injector control upgrade is in the checkout stage at the time of writing, but is essentially fully operational and appears to be working quite well. We are using this system as we are starting up for our next user run, with the intention of operating with the new system for that run.


## ACKNOWLEDGMENTS

Thanks to Max Cornacchia, Heinz-Dieter Nuhn, Jeff Corbett, Ed Guerra and all the other operations and support personnel at SSRL who helped to make this project a success.


References:


1. S. Howry et al., “A portable database driven control system for SPEAR,” SLAC Publication 3618, April 1985

2. C. Wermelskirchen et al., “The SSRL injector control system,” *Proceedings of the 1991 IEEE Particle Accelerator Conference*, 1991

3. D. Nelson, M. Briedenbach, et al., “The VAX CAMAC channel” *IEEE Transactions on Nuclear Science,* NS-28, 1981

4. A.D.Cox, "The SSRL beamline data acquisition systems", Rev. Sci. Instrum. 63(1), 854-855, 1992 (Special issue: *Proceedings of the 4th International Conference on Synchrotron Radiation Instrumentation*, 15-19 July 1991, Chester, UK)

5. R. Hettel et al., “Digital orbit feedback control for SPEAR,” *Proceedings of the 1995 IEEE Particle Accelerator Conference*, 1995