SLAC – PUB – 3668
May 1985
(A)

REPORT ON THE SLC CONTROL SYSTEM*

N. PHINNEY

Stanford Linear Accelerator Center
Stanford University, Stanford, California, 94305

ABSTRACT

The SLC control system is based on a VAX 11/780 Host computer with approximately 50 microprocessor clusters which provide distributed intelligence and control of all CAMAC interface modules. This paper will present an overview of the system including current status and a description of the software architecture and communication protocols.

1. INTRODUCTION

The SLAC Linear Collider (SLC) project will collide high intensity bunches of electrons and positrons at a center of mass energy of approximately 100 GeV. The design of the SLC has been presented in previous publications.[1] Major components of the machine include the CID high intensity injector,[2] the upgraded SLAC 2 mile Linac, electron and positron damping rings, the positron target and return line, the collider arcs and the final focussing section. The SLC control system must provide the monitoring and control functions required to reliably direct the complex SLC operation. At the same time it must improve the ability of the Linac to deliver beams to the storage rings and to other experimental areas.

2. BASIC ARCHITECTURE

The SLC control system uses a large central processor and a distributed network of 50-70 microprocessor nodes to control and monitor the machine. The logical topology is a star network with the host machine coordinating the μprocessor clusters. The task organization for the μclusters is geographical rather than functional; each cluster controls all functions for a given area rather than one function for a larger area. The host computer is a VAX 11/780 running VMS with a second VAX 11/780 used for software development and as a backup machine in the event of a hardware failure. The μcomputers are Multibus Intel 86/30 SBCs with 0.75 Megabytes of RAM, an SDLC network communications controller with modem, and a 5 Megabit/second serial Camac link (MBCD). The actual device interface is through Camac with up to 16 Camac crates per μcluster. The final system will include about 250 Camac crates and more than 3000 modules.

Control signals for the SLC flow along two major paths. The Main Drive Line (MDL) transmits all phase and 360 Hz fiducial timing information. It is an existing temperature stabilized coaxial line which carries the 476 MHz phase reference signal with a superimposed fiducial approximately 1 millisecond before beam time. The SLC timing system uses programmable modules which are synchronized by the fiducial and count down the fourth subharmonic of the 476 MHz waveform to produce adequately timed output signals. The Communications (Comm) Line is a CATV coaxial line with a mid-split system of cable amplifiers feeding an up-converter to provide approximately 140 MHz of useful bi-directional channel space. These subchannels have been allocated to computer-computer networks, terminal-computer networks, high speed command broadcast links, private high-bandwidth feedback control channels, general purpose video channels, audio communications channels and Comm Line diagnostics.

Operator interface to the SLC control system is provided by multiple independent control consoles which are (in principle) portable and receive information only from the Comm Line. Two versions of the console are supported: the Console on Wheels (COW) which has a full complement of interface hardware, and the CALF (small COW) which provides a subset of COW functionality for a much smaller cost. The COW[3] is a Multibus μcluster with an Intel 86/12 processor, monochrome touch panel, high resolution color graphics display, 4-8 general purpose knobs, standard terminal, video monitor, and audio intercom. The CALF uses a standard terminal, with optionally an audio intercom, to emulate the COW touch panel, knobs, terminal and alphanumeric displays. True graphics displays are not supported unless the CALF is also equipped with a graphics controller. Up to 15 consoles have been active simultaneously in current operation.

The primary communication link between the VAX and the control or console μcomputers is SLCNET,[4] a high speed (1 Megabaud) polled network developed at SLAC. The network is a logical star with the Host at the center which uses a half-duplex SDLC protocol on a subchannel of the SLC Comm Line. The VAX SLCNET Channel (VSC) is a high speed bit-slice μprocessor which controls the sequential polling (> 1200 polls/second) and provides a DMA channel to the VAX. The μclusters use a Computrol SDLC–Multibus interface with custom firmware supplementing the vendor code. The basic protocols support synchronized and unsynchronized reads and writes of buffers up to 8192 bytes, in addition to a variety of interrupt, broadcast and diagnostic functions. The network is used to download programs into the remote μclusters, to maintain the SLC database in the VAX and the μclusters, to transmit commands and responses using the SLC Message Service, to send graphics data to the COWs and receive operator input, to implement the cross debugging facilities, and to drive remote line printers.

3. SOFTWARE OVERVIEW

The SLC control system employs a hierarchical distribution of intelligence. Where possible, first level processing of signals is performed by the I/O modules themselves. The μclusters execute standard algorithms in response to commands from the Host computer, and convert measurements to physics units, e.g. kilogauss, millimeters, etc. The Host machine determines the operational configuration of the accelerator, maintains a centralized database, generates periodic checking and error reporting, and handles all operator interface functions such as display formatting and command transmission.

* Work supported by the Department of Energy, contract DE-AC03-76SF00515.

Poster paper presented at the 1985 Particle Accelerator Conference,
Vancouver, B.C., Canada, May 13-16, 1985

The bottom layer of the SLC software includes not only language and network support but also the software that defines the underlying architecture. It includes standardized routines for terminal I/O, touch panel communication, knob handling, database access, message services, error reporting, and display generation. These routines are carefully layered to isolate the other software levels from the details of network protocols. They provide structured and disciplined access to system resources.

The backbone of the control software has been implemented as Facilities which provide standard functions and displays for each of the major hardware systems, including Timing,[5] Klystrons,[6] Beam Position Monitors,[7] Magnets, and Analog and Digital Status. Each Facility has a dedicated job in the μclusters, a collection of control, display, and I/O routines in the Host, and a well defined set of functions and error codes for message communication. All of the Facility code is database-driven so that no software modification is needed when additional standard devices or new areas of the machine are added. The micro jobs perform high-level algorithms such as closed-loop trimming to set-points, standard diagnostic procedures, and routine monitoring and status reporting. The protocols permit actions to be taken on a single device or on a set of devices in parallel.

The highest layer of the software uses the functionality provided by the facilities to implement global algorithms or to provide special, tailored functions or displays not available from the standard software. Typical applications may use machine models to perform orbit correction, lattice matching, or klystron replacement. Utilities have also been implemented to digitize and fit data from profile monitors and to provide automated control and sampling of operator-selected parameters which may then be displayed or fit.

4. SOFTWARE ARCHITECTURE

Data Structures

Data for the SLC controls system is stored in three interrelated data structures. The first, called the database, stores static and quasi-static information about the machine. This information describes all control connections, device characteristics (e.g. magnetization polynomial fits), and quasi-static machine characteristics (e.g. values of magnetic fields). The database is hierarchical, identifying devices by symbolic names, locations, and sequence numbers. Actual data is retrieved by reference to secondary symbolic device attributes. No attempt is made to store information in the database relevant to only one pulse of the machine. The database is a structure designed explicitly for the distributed control architecture of the SLC control system. It consists of a master section of shared memory in the Host machine that is divided into sections for each μcluster.

Information organizing the quasi-static values of large sets of devices is stored as "configurations". A configuration is a formatted Ascii file identifying specific devices, secondary controllable attributes, and values for these attributes. The canonical example of a configuration is a set of explicitly identified magnets and their field values describing an operable lattice on the machine.

In addition to the quasi-static description of the machine which is primarily magnetic field values and "nominal" values of the RF amplitudes and phases, the behavior of the machine on a given pulse is determined by the triggerable devices, e.g. thyratrons for klystrons, or kickers for beam manipulations. As previously described, all timing is derived from the MDL by counting the fourth subharmonic of the RF following an AC line synchronized fiducial. A "Beam Definition" is a set of values for those counters. The control system can support up to 256 independently defined beams, whose sequence is arbitrary and determined by the Master Pattern Generator. The data structure containing all of the timing information is stored in the Host machine as the T-Matrix. The appropriate subsets of the T-Matrix are transferred to the corresponding μclusters.[5]

Communication Protocols

Communication among processes in the SLC system passes either transparently through the shared data structures or explicitly through the SLC message service. Messages are used to convey commands to the μclusters and command completion responses back to the VAX. They are also used for error messages or for data from a single pulse such as beam position monitor readings. All SLC messages have a simple structure consisting of ten words of header information followed by up to 502 words of data. A standard set of interface routines are available to construct messages, to synchronize commands and responses from multiple μclusters, and to signal error messages.[8]

5. HOST SOFTWARE

SLC Control Programs

Each user of a console (COW or CALF) has his own copy of the SLC Control Program (SCP). The SCP is the console interface to the database, configurations, T-Matrix and the message services. The SCP manages the console, translates user directives, and generates displays. The SCP handles only one user, and there is no requirement that all SCP's be identical. Because there is a separate copy of the SCP for each user, the SCP code is relatively easy to optimize and debug. The SCP executes interactively on a terminal which is used to receive messages from the SCP and other processes, and to conduct dialogs that are unsuited for touch panel communication.

The primary control interface for each console is a flexible touch panel system. Each panel is generated interpretively from a fixed format text file. A logical name service is used to provide communication between the application code and the panel description files. The COW also supports up to eight knobs for the input of quasi-continous variables to the program.

Paranoia

Paranoia is a Host process with two major responsibilities: error handling and machine monitoring. All errors detected by the SCPs, by other Host programs, or by the micro software are sent to Paranoia which receives the messages, logs them, formats them, and broadcasts them to the appropriate control consoles. Paranoia's monitoring activities are an evolving collection of diagnostic functions. It requests periodic checks of machine parameters such as magnet settings, klystron phases, temperatures, etc from the individual μclusters. Error conditions reported are metered and broadcast to the consoles. Future improvements will include global error analysis algorithms and more sophisticated diagnostics.

Other Host Programs

In addition to PARANOIA and the individual SCP programs running each COW or CALF console, several other programs are included in the Host software. The DataBase EXecutive process, DBEX, handles all network transfers of database information between the VAX and the μclusters. Another process called FEEDBACK will be responsible for the real time management of the SLC, primarily coordinating klystron replacement and the subsequent retuning of the machine, and the management of the feedback systems. Other programs provide network monitoring and diagnostics, error-log formatting and filtering, time logging of data, and Camac or database checking.

6. MICROCOMPUTER SOFTWARE

The μcluster software was designed to take advantage of the multitasking capabilities of the iRMX operating system. Each μcluster runs 10 to 15 jobs which timeshare according to their assigned priorities. The jobs are organized by function, with a separate job for each of the Facilities, e.g. Magnets or Klystrons, plus a number of additional server jobs. Each Facility job is responsible for performing all of the monitoring and local control algorithms for its class of devices. These jobs execute commands received through the message service and use a shared common database. Network input is handled by two separate network server jobs: one for database updates, the other for standard format messages. Since the database contains the list of devices to be controlled and their attributes, the same software executes in all of the standard control μclusters.

7. LANGUAGE AND SUPPORT

All software for the SLC is written, compiled, and linked on the VAX. Almost all of the code for either the VAX or the micros is written in Fortran 77, with some PLM 86 or ASM 86 routines where required in the micros. Considerable effort has been devoted to making the environment for μprocessor software development as efficient, user-friendly, and VAX-compatible as possible. A full array of iRMX cross-products including compilers, linkers, and a symbolic debugger have been provided to facilitate remote development and debugging.

ACKNOWLEDGMENTS

The original architects of the SLC Control System were M. Breidenbach and R. Melen who made major contributions to the design and implementation of all aspects of the system. The elegance of both the hardware and software is due to their insistence on standardization and system integration. Far too many people have contributed to the project to permit individual acknowledgments but particular mention should be given to A. Hunter for the iRMX operating system and cross-debugger and for SLCNET software; M. Breidenbach for the database; and D. Sherden for CALF and Camac software.

REFERENCES

1. R. E. Melen, "Design and Performance of the Stanford Linear Collider Control System," Proceedings of the Nuclear Science Symposium, Orlando, Florida, 1984.
2. M. C. Ross *et al.*, "Generation and Acceleration of High Intensity Beams in the SLC Injector," Proceedings of this conference.
3. J. E. Linstadt, "Dissecting the COW," Proceedings of this conference.
4. A. Hunter, "SLCNET Manual," Internal SLAC Document.
5. K. Thompson and N. Phinney, "Timing System Control Software in the SLC," Proceedings of this conference.
6. R. K. Jobe *et al.*, "Klystron Control Software in the SLC," Proceedings of this conference.
7. J. Bogart *et al.*, "Beam Position Monitor Readout and Control in the SLC Linac," Proceedings of this conference
8. N. Spencer *et al.*, "Error Message Recording and Reporting in the SLC Control System," Proceedings of this conference.