Lawrence Berkeley Laboratory
UNIVERSITY OF CALIFORNIA

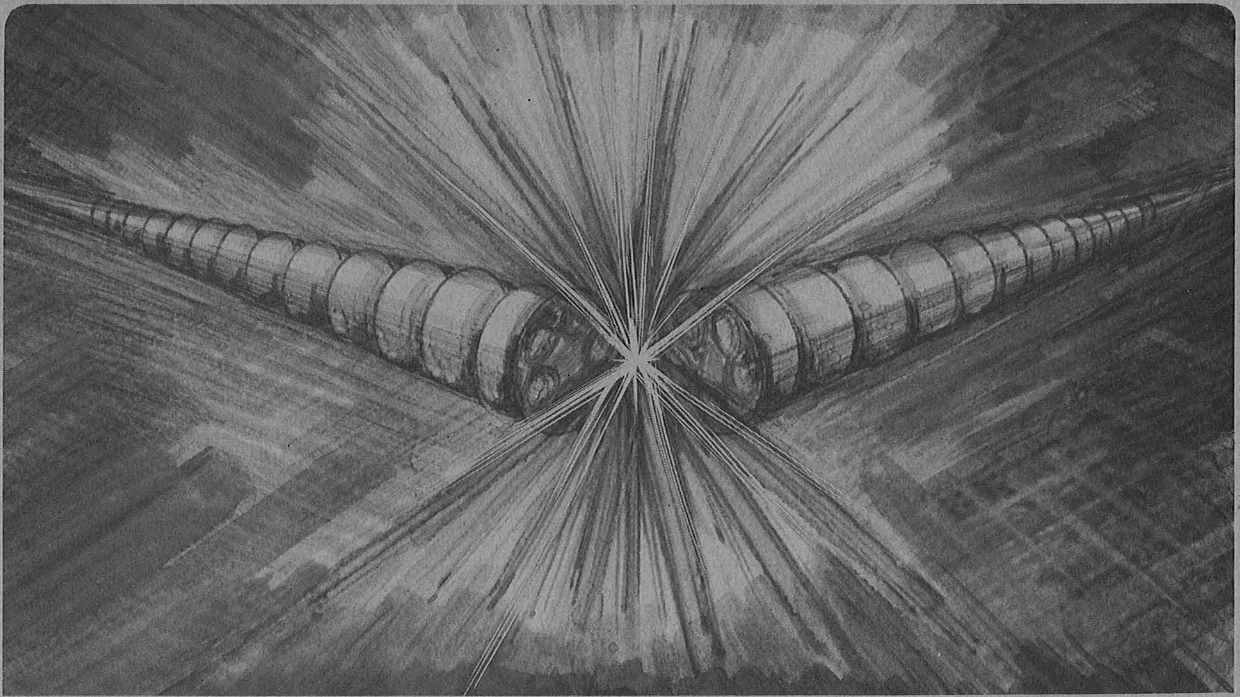
Accelerator & Fusion
Research Division

Presented at the 1979 Linear Accelerator Conference
Montauk, New York, September 10-14, 1979

A MICROCOMPUTER CONTROL SYSTEM FOR THE SUPERHILAC THIRD INJECTOR

H. D. Lancaster, S. B. Magyary, J. Glatz, F. B. Selph,
M. P. Fahmie, A. L. Ritchie, S. R. Keith, G. R. Stover,
and L. J. Besse

September 1979



Prepared for the U.S. Department of Energy under Contract W-7405-ENG-48

A MICROCOMPUTER CONTROL SYSTEM FOR THE SUPERHILAC THIRD INJECTOR*

H. D. Lancaster, S. B. Magyary, J. Glatz, F. B. Selph, M. P. Fahmie
A. L. Ritchie, S. R. Keith, G. R. Stover, and L. J. Besse**

Abstract

A new control system using the latest technology in microcomputers will be used on the third injector at the SuperHILAC.[1] We believe it incorporates some new and progressive ideas in both hardware and software design. These ideas were inspired by the revolution in microprocessors.

The third injector project consists of a high voltage pre-injector, a Wideroe type linear accelerator, and connecting beam lines, requiring control of 80 analog and 300 boolean devices. To solve this problem, emphasizing inexpensive, commercially available hardware, we designed a control system consisting of 20 microcomputer boards with a total of 700 kilobytes of memory. Each computer board using a 16-bit microprocessor has the computing power of a typical minicomputer.

With these microcomputers operating in parallel, we can greatly simplify the programming, literally replacing software with hardware. This improves system response speed and cuts costs dramatically.

An easy to use interpretive language, similar to BASIC, will allow operations personnel to write special purpose programs in addition to the compiled procedures.

Design Concepts

In most accelerator control systems software costs have been several times purchased hardware costs because the software had to be developed for each particular job. This has usually involved substantial changes in the purchased computer's operating system, or the invention of new networks, message services, etc., trying to solve the control problem particular to that accelerator. Besides being expensive, the systems software is generally so complicated that only a few highly specialized programmers can understand its function. This makes it almost impossible for the accelerator engineers and scientists to interact with the control system when new instrumentation or control applications are needed. Whereas hardware has increased in performance and decreased in cost, software has not kept pace. Therefore, it follows that one should buy more hardware if it reduces or simplifies the software task, at least until the two costs are equal.

In the last two years a number of very powerful, inexpensive 16-bit microprocessors have become available. The commercial board-level implementation of these products rival minicomputers in speed and memory, but are much more oriented toward control rather then general calculations.

The key to our implementation is the use of the Multibus[2] which makes it easy to link these new computers when needed. These computer boards must have local busses so that the Multibus is used only for intercommunication, therefore allowing true parallel processing. (In standard single bus systems the bus becomes the bottleneck.) We have found that this approach opens up an entirely new way to implement control systems which is cheaper, more versatile and faster. Because the only software is for applications programs, they can be implemented by accelerator engineers and scientists.

A new avenue for software implementation is thus made available and one can now attack the dominant cost in a computer control system, the software. In our system we tailored the computer hardware and software to the existing control problem, rather than trying to modify the control problem so it will fit some existing computer architecture.

Each computer board performs only one permanently assigned task, in a continuous loop or on a hardware interrupt basis. The programs are stored onboard in read-only-memory (ROM). This eliminates the need for program loading, scheduling, and synchronization. Thus, we have no need for an operating system! All programs are applications programs. Because the program on each board is self-contained, programming interference is virtually eliminated. Individual styles of programming (even languages) are acceptable.

The logical link between the various computers and processes is the database which is available to all computers. This allows:

- fast, direct access to accelerator data without message exchanges,
- programs that are entirely unaffected by additions to the accelerator hardware, because these are only reflected in the database information, which is interpreted by the programs,
- adding processing power or new procedures by simply plugging a programmed board into the Multibus.

* This work was supported by the High Energy & Nuclear Physics Division of the U.S. Department of Energy under contract No. W-7405-ENG-48.

** Lawrence Berkeley Laboratory,
Berkeley, CA 94720

[1] J. Staples, et. al, A WIDEROE-BASED HEAVY ION PREACCELERATOR FOR THE SUPERHILAC, IEEE Transactions on Nuclear Science, Vol. NS-26, No. 3, June 1979.

[2] Multibus is a trademark of Intel Corporation.

Through parallel processing of sections of a particular task we can achieve very fast performance. This allows us to write all programs in a high level language, PL/M-86 (in spite of its slower execution), resulting in drastically shortened programming and testing times as compared to assembly language. Also, most accidental programming errors are eliminated by this compiler's elaborate syntactical checks.

We believe this accelerator control system design, by using the latest microcomputer technology, offers a way to both lower costs and implement a system which can be understood by accelerator engineers and scientists. From our experience a person with both hardware and software experience is best suited to apply this new technology.

Timing Considerations

The SuperHILAC (Fig. 1) is a relatively fast pulsed machine, running at 36 Hz. It accelerates ions from three injectors on a pulse to pulse basis. This means at the hardware level we must complete our tasks in at most 27.7ms, with a new machine state or mode, e.g. linac tank gradients and magnet currents, being transmitted each pulse if needed.

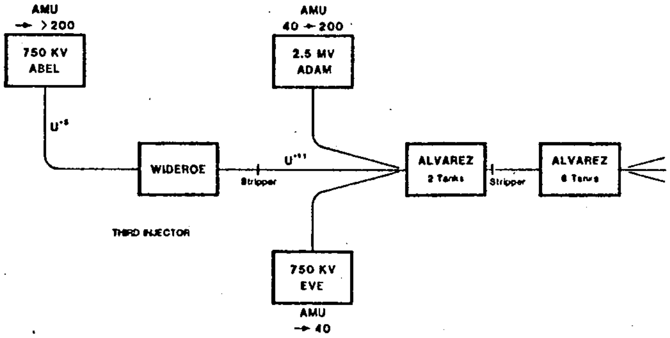


Fig. 1: Schematic layout of SuperHILAC. The control system described in this paper is needed for the third injector, comprising the 750 kv Cockcroft-Walton, the Wideroe, and associated transport systems.

Acceptable response to operator requests should be on the order of 100 ms. Interactive graphics at this rate impose a severe computational burden.

The least severe requirement is for storage and retrieval of accelerator parameters which occurs at the beginning and end of an experiment and can take several seconds.

With parallel processing we can cover this range of timing and execution requirements.

Architecture

The system architecture (Fig. 2) is of the star type which lends itself nicely to our machine because we have clustered equipment. Each "module" in our system is an Intel SBC 660 card cage with a Multibus backplane, containing up to eight boards.

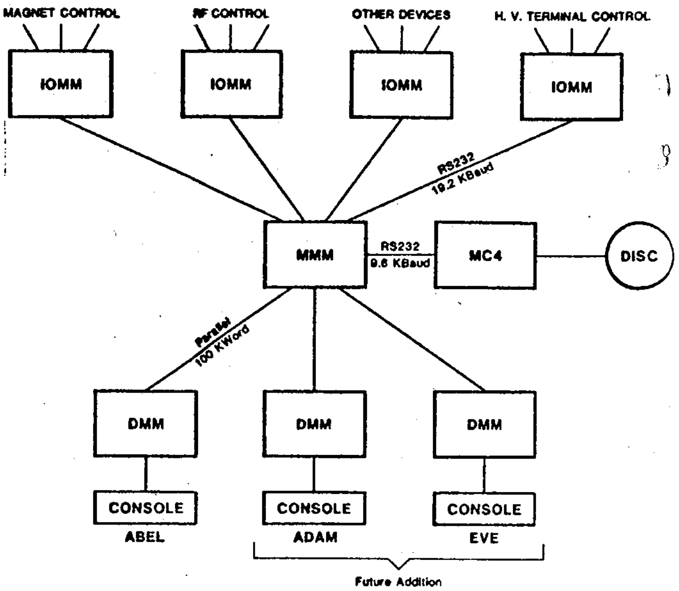


Fig. 2: The third injector control system utilizes the star concept, in which a central processor, the MMM, acts to link several hardware interfaces, the IOMMs, with several operator stations, the DMMs. The great advantage of this system over previous systems is that it employs distributed intelligence so that the MMM is not overworked, which could cause it to be a bottleneck.

We have several distributed input-output modules (IOMM) which interface to the accelerator equipment. The modules control the accelerator parameters in synchronism with accelerator timing. Data from the IOMMs is collected by a multiplexer (MMM) which then disseminates it to the display module (DMM) for data display and operator interaction. The MMM and DMMs are not synchronized with the accelerator but must respond in 100ms for adequate interaction between operator and machine. These modules are described in detail later.

To bridge the long distances between the IOMMs and the MMM we use serial transmission (RS 232) over fiber-optics at 19.2 kbaud. This limited rate is acceptable since data is exchanged primarily on demand.

The full data flow from the MMM to the DMM is handled by a 16-bit parallel bus over a few meters. This communications system is set up to provide access to continuously updated copies of the complete database (total of all the primary databases residing in the IOMMs) for every computer in the MMM and the DMM.

Database

Each IOMM has its own database describing the connected devices completely. After initialization the MMM and DMM contain and have control of the sum of all IOMM databases. This gives immediate access to all injector parameters and their descriptors for operator analysis.

The IOMM database is composed of three major parts: common passive, particular passive, and active. The passive parts are stored in ROM.

The common passive part contains descriptors required for every element, for example, element name, element type, and pointers to the other two parts.

The particular passive part contains the data that is unique to that element and does not change, for example, conversion constants and hardware channel numbers.

The active part located in random-access-memory (RAM) contains those element parameters which change with time, for example, setpoints and monitored values.

The database has two element types, the hardware and the pseudo. The hardware type element refers to parameters associated with actual hardware signals. These are: analog control, analog monitor, boolean (single bit) control, boolean monitor, digital (byte) control and digital monitor. These elements can be coupled to allow closed loops, interlocks, and complex devices.

The pseudo type elements are system or special function elements that are required to fulfill special software constructs. For example, examining pseudo element "Link-Error" allows us to monitor transmission errors between the IOMM and MMM.

Input-Output Microcomputer Module (IOMM)

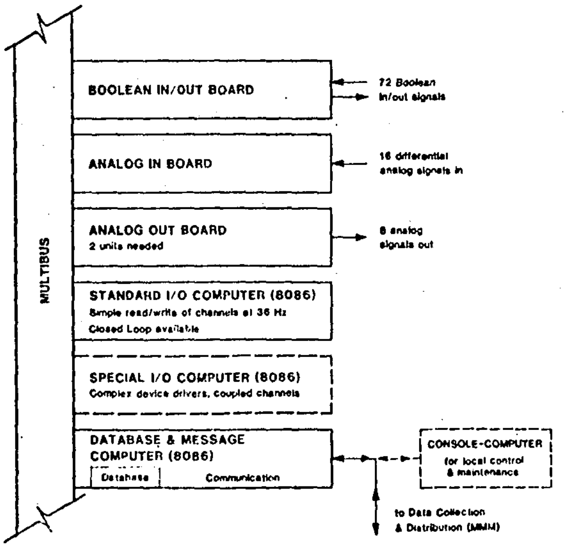

MULTIBUS
BOOLEAN IN/OUT BOARD
72 Boolean in/out signals
ANALOG IN BOARD
16 differential analog signals in
ANALOG OUT BOARD
2 units needed
8 analog signals out
STANDARD I/O COMPUTER (8086)
Simple read/write of channels at 36 Hz
Closed Loop available
SPECIAL I/O COMPUTER (8086)
Complex device drivers, coupled channels
DATABASE & MESSAGE COMPUTER (8086)
Database
Communication
CONSOLE-COMPUTER
for local control & maintenance
to Data Collection & Distribution (MMM)

Fig. 3: Input/Output Microcomputer Module. It provides the hardware interface to the accelerator and holds the primary database. The three computer boards (shown in the lower boxes) work in parallel and have access to every board on the Multibus.

Each IOMM has a common set of external interrupts and accelerator mode data supplied by a high speed serial fiber-optic link. This synchronizes the IOMM's with the SuperHILAC pulsing sequence. To satisfy our timing requirements and to simplify the software, we require that all I/O programs be completed in one pulse.

The IOMM (Fig. 3) is a card cage with two SBC 86/12 computers and additional boards with 16 analog channels, 72 I/O lines, and provision for 8-bit data exchange to external devices. The 16 DAC's and 16 ADC's are 12-bit converters. The 72 I/O lines are interfaced with the real world using optically-coupled, Opto 22 modules. The data base and message computer (CPU1) transmits the active data to the MMM in order of priority, which is determined by operator demand and changing parameters. Five records can be sent between machine pulses. One of these five is non-prioritized and steps through the database so that a minimum trickle rate of two seconds is assured for low priority devices. CPU1 constantly evaluates and tailors the priority based on information received from the standard I/O computer (CPU2). The MMM sends new setpoints to CPU1 which stores them in the active database, then echoes them back to the operator for confirmation of data transfer. CPU1 contains the IOMM's database. CPU2 handles the interface to the accelerator parameters; it observes monitored data for fast and slow drift, and ability to follow the setpoint. Data failing these tests are flagged with an appropriate priority and sent to CPU1. Closed or open loop operation is provided. I/O channels can be treated independently or in groups for special tasks.

A portable console-computer can be connected to the IOMM in place of the MMM for local control and debugging of the IOMM and all devices connected to it. After check-out the IOMM can be reconnected to the MMM without disturbing the rest of the system.

Multiplexer Microcomputer Module (MMM)

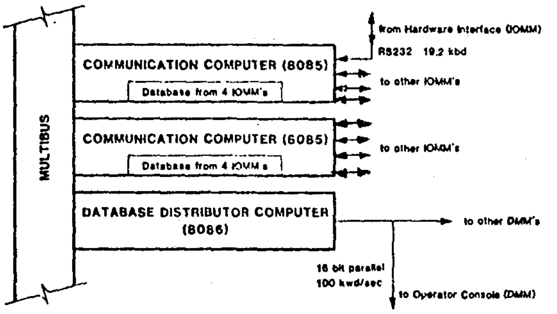

MULTIBUS
COMMUNICATION COMPUTER (8085)
Database from 4 IOMM's
from Hardware Interface (IOMM)
RS232 19.2 kbd
to other IOMM's
COMMUNICATION COMPUTER (6085)
Database from 4 IOMM's
to other IOMM's
DATABASE DISTRIBUTOR COMPUTER (8086)
to other DMM's
16 bit parallel
100 kwd/sec
to Operator Console (DMM)

Fig. 4: Multiplexer Microcomputer Module. Database information is exchanged between eight IOMMs and one of the display microcomputer modules through this unit, using both serial and parallel transmission. Shared access to any of the IOMM databases is essential.

The MMM (Fig. 4) simultaneously collects database information from all the IOMMs (via the RS 232 links) and forwards it asynchronously to the DMM. Likewise, tuning data generated in the DMM is sent through the MMM to the IOMMs. Since the MMM concentrates all the IOMM data it is a potential bottleneck. This problem was solved by parallel processing.

The data from the IOMMs comes into Intel SBC 544 communication boards via four onboard USARTs. It is mapped into the dual-ported memory, with each IOMM allotted a 4K byte block of RAM. Each memory block contains the full database of the respective IOMM. The computer on these boards, an 8085, is fully occupied servicing the USARTs and managing the data. Each board handles four IOMMs.

A 16-bit computer, Intel SBC 86/12, then reads the databases from the dual-ported SBC 544 memory via the Multibus and transmits it to the DMM over the fast parallel link.

The MMM may also house an additional computer which operates the link to a disc used for storage and retrieval of injector parameter sets.

Display Microcomputer Module (DMM)

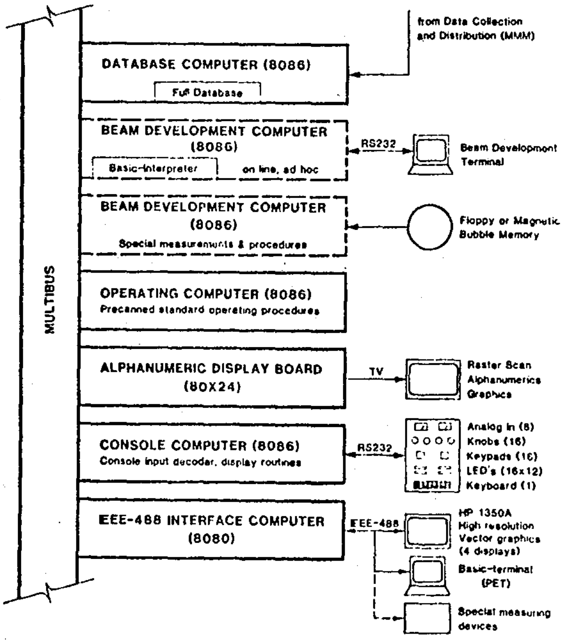

MULTIBUS
from Data Collection and Distribution (MMM)
DATABASE COMPUTER (8086)
Full Database
BEAM DEVELOPMENT COMPUTER (8086)
Basic-Interpreter
on line, ad hoc
RS232
Beam Development Terminal
BEAM DEVELOPMENT COMPUTER (8086)
Special measurements & procedures
Floppy or Magnetic Bubble Memory
OPERATING COMPUTER (8086)
Precanned standard operating procedures
ALPHANUMERIC DISPLAY BOARD (80X24)
TV
Raster Scan Alphanumerics Graphics
CONSOLE COMPUTER (8086)
Console input decoder, display routines
RS232
Analog In (8)
Knobs (16)
Keypads (16)
LED's (16x12)
Keyboard (1)
EEE-488 INTERFACE COMPUTER (8080)
EEE-488
HP 1350A High resolution Vector graphics (4 displays)
Basic-terminal (PET)
Special measuring devices

Fig. 5: Display Microcomputer Module. Up to three independent computers (boxes 2 to 4 from top) are planned to perform the operating tasks. They have access to the accelerator by writing into or reading from the database. They control the console devices by exchanging messages with the console computer over the Multibus.

The DMM (Fig. 5) displays accelerator data and generates control data from operator inputs and built-in procedures. To the DMM the accelerator is fully represented by a database copy allowing the reading of parameter values independent of machine timing and actual access procedures. The database copy in the DMM database computer is updated through a fast link from the MMM at a rate of at least 10/sec. The operating computer is the actual master of the whole control system. It processes requests, sends these or computed setpoints to the database, monitors all error conditions and generates various formats for displaying the accelerator data to the operator.

The operating computer does not see the actual hardware of the console. Console devices and displays are handled by three additional boards which share the Multibus: the console computer, an alphanumeric display board, and an IEEE-488 interface computer. The console computer generates alphanumeric and graphic displays and evaluates the messages from such control units as knobs, keypads and touch buttons. For alphanumerics, the memory-mapped display board converts an 80 x 24 character pattern into a video signal.

Generation of fast, high resolution live graphics is more elaborate. We use an HP-1350A graphics translator that is addressed through the IEEE-488 interface board. We plan to mix graphics with real-time analog signals. For both types of displays, the hardware choices assure a display frame update of at least 10/second. This IEEE-488 board is also used for other compatible devices as they are needed (such as a commercially available BASIC-terminal).

The operator console holds up to forty input devices and sixteen 12-character LED-displays. They are controlled by a local microprocessor (8085) in order to reduce cabling. Information is exchanged with the console computer in the DMM through an RS 232 link.

The beam development computers perform special beam measurements, such as emittance or bunch structure, and allow on-line generation and execution of one-time or prototype programs, using a BASIC interpreter. They may utilize any of the console input devices and display units by writing appropriate messages into the console computer. The interpretive language approach has proved its usefulness at a number of accelerators, particularly at the CERN SPS where the NODAL language was implemented.

Schedule

At present, we have completed and tested the software and hardware for the IOMMs, the serial communication links, and a preliminary version of the DMM. We plan to have the software for an operational control system and the operator console hardware finished by January 1980. The other hardware, which is mostly purchased, will be installed in October 1980. The third injector project is scheduled to be finished in June 1981.

TECHNICAL INFORMATION DEPARTMENT
LAWRENCE BERKELEY LABORATORY
UNIVERSITY OF CALIFORNIA
BERKELEY, CALIFORNIA 94720