

Fermilab

TM-1261
2126.000
2563.000

MODERN CONTROL TECHNIQUES FOR ACCELERATORS*

R. W. Goodwin and M. F. Shea

May 1984

*Submitted to the Tenth International Conference on Cyclotrons and Their Applications, East Lansing, Michigan, 30 April - 3 May, 1984.

MODERN CONTROL TECHNIQUES FOR ACCELERATORS

R.W. Goodwin and M.F. Shea

Fermi National Accelerator Laboratory*
 P.O. Box 500
 Batavia, Illinois 60510

Abstract

Beginning in the mid to late sixties, most new accelerators were designed to include computer based control systems. Although each installation differed in detail, the technology of the sixties and early to mid seventies dictated an architecture that was essentially the same for the control systems of that era. A mini-computer was connected to the hardware and to a console.

Two developments have changed the architecture of modern systems: a) the microprocessor and b) local area networks. This paper discusses these two developments and demonstrates their impact on control system design and implementation by way of describing a possible architecture for any size of accelerator. Both hardware and software aspects are included.

Introduction

Until the late 1960's, particle accelerators were operated using hardwired controls. Early computer control systems were designed around a single minicomputer that was connected by a multiplexing scheme to the accelerator equipment. The interface electronics and the multiplexing schemes were usually custom designed for the specific application.

Techniques available today make it possible to distribute literally hundreds of inexpensive computers throughout a control system. However, the architecture used to distribute multiple processors, the partitioning of the tasks among processors, and the communication protocol that interconnects these processors, will all affect the capability, performance and ease of use of the final system.

This paper will attempt to point out some of the modern techniques that are available to control system designers and describe how these tools may be applied to a control system. It should be pointed out that there is no single "right" way to implement a control system, but there are some obviously wrong choices that can be made that unnecessarily compromise the performance and complicate both the hardware and the software of the system. Choices outlined in this paper represent preferences of the authors - others may choose differently!

*Operated by Universities Research Association Inc., under contract with the U.S. Department of Energy.

The Single Computer Architecture

Although single processor architecture was initially chosen to minimize cost and complexity, it is interesting to consider the strengths and limitations of such systems. Figure 1 shows the classical single processor/single console control system. It is a very straight forward manageable concept - the single computer executes all tasks, acquires data from the hardware either periodically or only as needed, runs application programs that reside on the CPU's disk, and handles operator input and application program output that goes directly to either the attached console or to the accelerator hardware. In many cases the data is memory mapped so input/output (I/O) operations are simply "Move" instructions. This makes a very simple, high performance system. None of the complications of multi-user systems and communication between multiple processors applies to the single CPU/single console case. If such a system is adequate, it should certainly be considered.

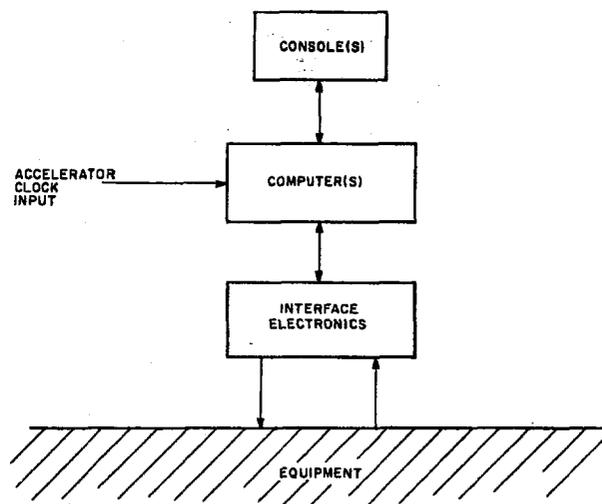


Figure 1. Diagram of a Single CPU Control System

The limitations of a single CPU begin to show up when more than one console is needed. Processor time must then be split between multiple consoles, so the response time is increased, but for a small system with parallel-connected I/O the performance is probably still adequate. For larger accelerator installations, I/O becomes too spread out to be connected to the processor in parallel, so a serially connected multiplexing scheme is used. Registers in the equipment can then be read and set using this serial link. A distributed serial CAMAC system is an

example of such a configuration. Although this makes a cost effective way to distribute the I/O hardware, the fast access to the interface hardware is gone. Instead of reading memory locations, which requires a few microseconds, the processor must now drive the serial link hardware and communicate with the equipment by its address on the serial link - a process that takes more than 100 microseconds for a random operation. Because communication with hardware is now much slower, the amount of time the computer spends reading data and the amount of traffic on the link become important. The time the console computer spends reading data can be reduced by adding a separate computer to drive the link, and if more consoles are needed, each can have its own computer. The traffic on the serial link can be reduced by putting additional processors closer to the hardware.

Distributed Multiprocessor Systems

To support the demands of a larger control system, more processors are required. There are basically two choices for distributing more processors in the system: use a large number of small dedicated processors, or a smaller number of multifunction processors.

Large Numbers of Dedicated Processors

When dedicated, or embedded, processors are used, the modules containing the processors can be tailored to fit the application and only necessary features are included in the design. Often these smart controllers are configured to act like hardware so they can be read and set in the same way as hardware devices. The system now contains both non-intelligent and smart devices that are all driven from the serial link. Even though individual devices are smart, communication between them is difficult and coordination of multiple devices must still be done by the central computers. Smart controllers usually complicate the software of the central computer, and these controllers are seldom commercially available. In a large accelerator, several hundred of these smart modules may be used, so an effort is made to minimize the cost of each. As a result, little or no local control is provided. The overall architecture has not really been changed.

Smaller Numbers of Multifunction Processors

The second way to add distributed processing is to implement multifunction computers that provide the interface to a local area of equipment or an entire accelerator subsystem. These computers provide all the processing required for their area such as reading analog and binary data, setting analog and binary data, and monitoring the analog reading or binary states of local equipment. With the addition of a small console, as shown in Figure 2, and the data base for its local devices, these systems are, in fact, the simple single-computer control system discussed above. All control services are provided locally. The local computer has access to all the local data so functions like closed loop control, that require action based upon processed readings, can be executed

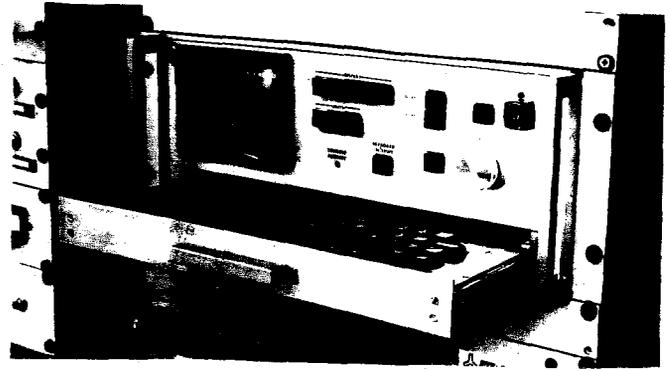


Figure 2. Photograph of a Local Console

without involving the central computers. Of course dedicated processors can be accommodated as needed. If the local data base is stored in non-volatile memory, these stations do not depend upon central computers to operate. Characteristics of the hardware, scale factors, closed loop parameters and even the last settings will be known when the station is powered up.

Networks

The interconnection of various processors in a system is called the network. Because of the extended distance involved, the network will be a serial connection. For systems that combine hardware I/O interface equipment and small dedicated processors the network is usually a hardware link; multifunction processors use a Local Area Network (LAN).

Hardware Link

Hardware links are typically word-oriented. That is, a single read or write command is addressed to a specific hardware register in the I/O equipment. The apparent fast access to remote registers is somewhat misleading because the high software overhead associated with each transmission makes it impossible to sustain a high throughput. Access times to a random register, typically greater than 150us, is determined almost entirely by software and, for link transmission speeds of a few MHz, little benefit is gained by higher speed links. Data are checked to insure that each word is transmitted error free. When sending a message to a remote processor, multiple words are sent and although individual words are checked by the hardware, integrity of the entire message must be done by software. A serial CAMAC link is an example of a hardware link.

A hardware link is clearly a master-slave arrangement where all communication originates with the single controlling computer and link communication between remote slave processors is impossible.

As the system grows, there will be a definite limit on the throughput of the master computer.

Local Area Networks (LAN)

Local Area Network denotes a class of communication facilities that provide high speed data exchange among computers located within a moderate geographical area. A LAN falls in the middle between long distance, low speed networks that carry data for hundreds or thousands of kilometers, and very high speed connections that are limited to a few tens of meters. The communication needs of an accelerator can be satisfied by a Local Area Network. Modern LANs typically provide communication between computers at bit rates of one to a few tens of megahertz.

In a LAN, a full message is sent from one computer to another. Because of the high sustained bit rate, data is sent from and stored into the computer memory under DMA control. A typical LAN message is sent as a frame that includes the address of the source and destination computers, a few bytes to describe the message type, a block of data and a frame check sequence used to test the integrity of the entire message. An error anywhere in the message renders the entire message invalid. The length of the data field of the message can range from a few bytes to thousands of bytes. Each specific network protocol defines the allowed message size.

In contrast with the hardware link, a Local Area Network communicates between computers, and complete messages are sent as a single transmission. The receiving computer cannot be expected to respond instantaneously like a read operation of a hardware register. However, a list of data can be requested and the destination computer can prepare the list of answers to be returned in a single message. If necessary, answers can be sent periodically following a single request for repetitive data. As a result of using a LAN, messages can be longer, the requests for repetitive data are sent only once and remembered by the local computer, and broadcast messages are supported. Using local stations, alarm checking may be done locally, and the number of computers is relatively small - of the order of tens rather than hundreds. All these effects tend to reduce the link traffic.

The complexity of a Local Area Network would limit our ability to implement them if it were not for the large scale integrated circuits that are now becoming available. In some cases a single chip contains the entire support for the communication protocol and also the buffer management for the memory used to hold the messages. The LSI controller, memory, driver, and receiver electronics for the transmission media can be incorporated on a small circuit board. Complete network interface boards are now available commercially for several network types and several common microprocessor buses

Transmission Medium The transmission medium for Local Area Networks can be twisted pair wire, coaxial cable or fiber optic cable depending on the network requirements. Signalling may be baseband which means the medium carries only the signal for a single network, or broadband, a technique that makes use of cable TV technology to carry multiple communication, voice and video channels on a

single cable. For purposes of this paper, assume the transmission medium works.

Network Protocols Local Area Networks are characterized by their access method, signalling technique, bit rate and message protocol. The access method is the means by which a station is allowed to use the shared transmission medium to send a message.

Master-slave networks depend upon one station, the master, to orchestrate all activity on the network. Peer protocol networks consist of stations that are all equal. That is, any station may send a message to any other station thereby eliminating the bottleneck that can be caused by funneling all message traffic through any single station.

A variety of peer protocol access mechanisms have been developed, and some of these are currently being standardized by the IEEE-802 committee. Carrier Sense Multiple Access with Collision Detection (CSMA/CD) is used by (Xerox Corporation's) Ethernet. Token access network protocols are already available as LSI and board level products. ARCnet (by Datapoint Corporation) is an example of an easily implemented token passing bus protocol. An IBM-designed token passing ring architecture should become available soon, and a slotted ring network is being developed at Cambridge University in England. These networks operate in the range from one to fifty megahertz and have characteristics that make each of them possible choices for use in various parts of a large control system.

The Local Station

In the control system design that utilizes multifunction local stations interconnected by a Local Area Network, the architecture of the stations is the same as the original single CPU control system, but now the minicomputer is replaced with a microcomputer. Many of the modules needed to implement the local station are commercially available. The design of the local station is the process of defining the console and selecting the processor type, the microcomputer bus, and the I/O connection to the accelerator hardware. Considerations for making these choices are given below. Note that although the choices may change as new devices become available, the selection criteria remain valid. In order to determine specific characteristics of a local station the overall size of the station must be known.

The Size of a Local Station

By sizing of a local station, we mean deciding how much accelerator equipment or how large a physical area is to be controlled by a single microcomputer. As the local station grows, the cable runs become longer and more expensive, analog signals can become noisy, the local console gets farther from hardware and eventually the microcomputer becomes overloaded and its high performance is lost. If the area and amount of controlled equipment is too small the number of local stations becomes too large and individual stations have so little to do that it is difficult to justify the existence of local control

consoles at all. If the local station does not include a complete accelerator subsystem, it may be impossible to implement stand-alone closed loop control because the reading of a controlled parameter and the setting of the controlling device may not be in the same processor.

We have found that a convenient size is a local station that can accommodate about one hundred analog channels. Using this size of local station, the Fermilab 200 Mev Linear Accelerator is controlled by fifteen stations. Preliminary designs for the National Superconducting Cyclotron Laboratory at Michigan State University indicate that about ten local stations will be needed. In most cases, the organization or geography of an accelerator will suggest a reasonable grouping of signals to be monitored by a single station.

The Microprocessor

A local station that controls a major subsystem of an accelerator will cost a few thousand dollars not including the I/O interface equipment. At this level, the cost of the microprocessor chip itself is negligible, so the selection is made on the basis of performance and ease of programming. The Motorola MC68000 is an example of a good, high performance 16/32-bit processor that is supported with all the necessary assemblers and high level language compilers. This ease of programming and the additional performance far outweigh the small price differential between the 68000 and an 8-bit processor. The performance of a 68000 exceeds that of a PDP-11/34 and, with 24 bits of addressing, there is no memory size limitation. The 68000 family is soon to expand with the introduction of the 68020, a 32-bit data, 32-bit address processor that is software compatible with the 68000. This new processor will bring VAX level processing power to small microcomputer systems.

The Microcomputer Bus

The microcomputer bus is the backplane that interconnects various modules of the local station. Currently several 16/32 bit backplanes have been announced: VMEbus (IEEE-P1014) from Motorola, Multibus II from Intel, and Futurebus (IEEE-P896). Of these only VMEbus is currently available; 50 to 100 VMEbus manufacturers now offer CPU, memory and I/O interface modules. Figure 3 shows a Local Station resident in a VMEbus chassis, with significant room for I/O modules and controllers. Although many existing systems use the 8/16-bit Multibus I (IEEE-796), VMEbus is a better choice for a new design.

All of the 16/32-bit buses use the Eurocard format with the more reliable two-piece pin and socket connectors, rather than card edge connectors. The rack enclosures, backplanes, connectors and module kits are available from several vendors. In contrast with earlier backplane designs, the VMEbus specification defines a high performance, multilayer backplane with ground traces between signal traces and resistive terminations on both ends of the bus.

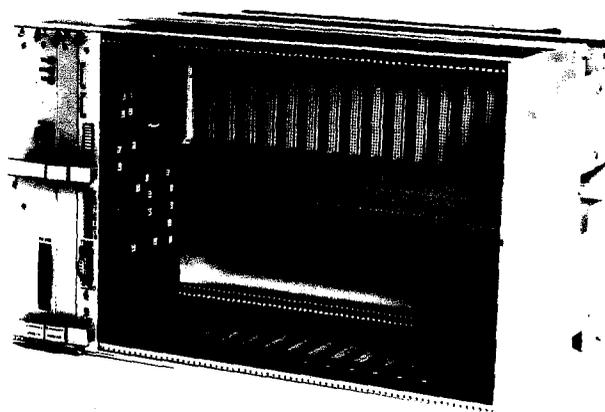


Figure 3. Example of a VMEbus-based Local Control Station

I/O Interfacing

The local station must include analog input and output, and digital input and output. Using commercially available VMEbus modules, nearly all the data acquisition electronics could be located in the VMEbus chassis, thereby eliminating special in-house design, fabrication, and documentation of interface modules. In spite of those advantages, it is often necessary to use interface equipment external to the VMEbus enclosure. For example, if special needs dictate that digitizers or control electronics be closer to the accelerator hardware, then external interface equipment must be used. In this case, one or more industry standard I/O formats may be used. CAMAC, IEEE-488, MIL-1553B and Motorola's I/O channel are examples of available I/O hardware systems.

CAMAC (IEEE-583) The CAMAC standard has been used for many accelerator control systems because, historically, it was one of the first standards to arrive. CAMAC was originally designed to be a 24-bit I/O system for use in particle physics experiments. Since its introduction in the late sixties, CAMAC has been used to house not only dumb I/O modules, but also intelligent controllers containing processors. The CAMAC dataway is not intended as a computer bus - the standard is not defined to have a large address space. Memory arrays larger than 256 words can only be accessed sequentially.

Within the VMEbus format it is possible to design a simple memory-mapped CAMAC branch highway driver that works with the standard Type-A CAMAC controller (IEEE-596). If the CAMAC Crate (C), slot Number (N), subAddress (A) and Function code (F) are mapped into VMEbus memory as shown in Figure 4, then all the possible CNAF commands for all slots of the seven crates allowed by the Branch Highway standard are available to the local processor by way of simple memory MOVE instructions. The memory-mapped Branch Driver occupies one quarter megabyte of memory space - less than 2% of the 68000 address range. Note that this technique is not available to processors with limited addressing capability, such as 8-bit processors and PDP-11 computers. The use of this Branch Driver makes all previously

designed CAMAC modules available for use in the VMEbus based local stations.

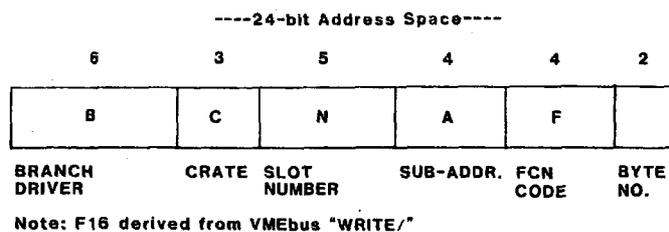


Figure 4. Mapping CAMAC CNAF Commands into VMEbus Memory Address Space

GPIB-(IEEE-488) The GPIB (general purpose interface bus) is used to control many laboratory instruments such as spectrum analyzers, digital voltmeters and frequency counters. Commercially available VMEbus-GPIB controllers can be used to interface directly to these instruments.

MIL-1553B This is a serial data bus developed for use in military avionics and aerospace applications. The standard defines a one megahertz, time division multiplexed data bus that provides half duplex communication between a controller and any of thirty-one possible Remote Terminals (RT) using a single twisted pair cable. MIL-1553B is interesting for accelerator control because all stations are transformer coupled to the bus to provide the isolation needed for data acquisition in noisy environments. MIL-1553B is feasible because LSI interface chips are now available to support the communication protocol. The protocol allows the transfer of up to 32 sixteen bit words between the controller and any of the thirty subaddresses contained within one of the thirty-one remote terminals. Distances of several hundred feet may separate the controller from the remote terminal. This standard is a word oriented hardware link and, therefore, it is not well suited to message passing between computers.

Motorola I/O Channel Motorola has defined a byte wide memory mapped connection to external I/O hardware, called the I/O Channel. This specification allows the 50 conductor I/O Channel ribbon cable to be about four meters long, enough to reach equipment contained in two or three relay racks. The simplicity of interfacing equipment with this bus makes the I/O Channel an attractive way to connect local hardware.

Smart Subsystems In some cases, the complexity of operating some group of hardware may suggest the need for an intelligent controller. A tightly coupled controller includes an additional computer board that may be plugged into the VMEbus enclosure and share the bus with the local station processor. The necessary I/O boards may also be located in the VMEbus bin. The two processors communicate using shared memory. In a loosely coupled system, the smart subsystem may be connected to the local station using a Local Area Network such as ARCnet. Messages are then passed between processors using the network.

VMEbus is designed to support multiprocessor configurations. The serial VMSbus is intended to meet the needs of such systems by supporting event communication, fault tolerance, and intelligent semaphores.

The Local Console

The distributed multifunction processor organization can support an inexpensive full function console at each local station. With very little hardware, a small console may be built that includes a video display, a full keyboard, several push buttons and a shaft encoder knob. Figure 2 is a photograph of such a console shown installed at the Fermilab 200MeV Linac.

From this console, an operator can monitor and control devices at the local station, at any other local station on the local area network or even devices in other buildings or accelerators within the complex.

Application Programs

An application program in a local station is a program which displays information on the display screen and responds to a local operator interactively. An application program may be invoked as it resides in PROM, be downloaded from the network into non-volatile RAM, or loaded from local disk into memory. If the network dies, the local station can still invoke programs which it keeps in PROM or those maintained in non-volatile memory.

One example of an application program commonly used in Fermilab control systems is called a Parameter Page program. It displays arbitrary lists of devices (configurable by the operator) including their readings, settings, alarm reference values, status. It also allows knob control of analog settings or on/off digital control of devices. Readings may be plotted as a function of time or of the parameter being adjusted with the knob.

Data base access programs facilitate entry of new devices into the local database for a station, or the modification of existing device specification. Scale factors for engineering units scaling, descriptive text and allowed analog control specification are examples of data base information which may be entered. When a completely new station is to be added to the network, the local database may be configured completely and checked out locally before attaching to the network. That entire set of devices then becomes instantaneously accessible to any of the other nodes in the network.

Diagnostic programs allow easy checkout of new hardware or software.

Local Station Summary

The preferred architecture for a local station is one that centers around a capable microprocessor such as the 68000 (or the 68020). The station provides all the control services of an accelerator subsystem, and all settings and readings for the subsystem are done by the local station. Communication with the local station is by way of a peer-protocol

Local Area Network to allow any local station on the network to ask for data from elsewhere, and to return data to requestors at the requested rate.

The I/O interface can take on many different forms and the design of the local station should allow the use of several interfacing techniques and protocols. Direct parallel, CAMAC, IEEE-488, MIL-1553B and I/O Channel are some of the currently available interface methods. Complicated smart interface controllers may be tightly coupled using multiprocessors on the common backplane or loosely coupled using their own (very) local area network such as ARCnet. Here, embedded processors on smart modules, such as 1553 RT's or CAMAC, should be structured to appear as hardware registers rather than communicating using a message protocol. Messages are the preferred way to communicate only when there is Local Area Network support so that an LSI LAN Controller can handle the memory-to-memory transfer, protocol and message integrity checking.

The local stations are the real control system. Everything else is not control system, but communication system. The field of Local Area Networks is developing rapidly. The proper choice of networks may change as new devices become available; however, the organization described here is not essentially changed by the LAN choice.

In the selection of LANs and I/O interfaces, the use of standards should be encouraged. Although in house designs may work well, the design effort and lack of commercial availability make in-house hardware less attractive. It is difficult to compete with the semiconductor industry when an LSI controller becomes available to do some portion of the control hardware. This is particularly true of communication controllers where the entire job of transmitting and receiving messages is handled by the LSI devices with no processor intervention required. The IEEE-802 committee is currently defining LAN standards. Both chip and board level products that support these standards will become available in the near future.

It is possible to implement the core of a local station as a single chassis that contains the computer, memory, and a few I/O and communication controllers. Figure 5 is a photograph of the rear panel of a local console. A small VMEbus crate has been added and the entire local station including CAMAC, MIL-1553B, I/O Channel and ARCnet controllers are all contained in a seven inch chassis.

For new accelerator projects, the control system organization described here makes the final control system available for use during the construction, installation and testing phases of the project. Older designs that do not function without the support of the central computer facilities cannot be used until the project is nearly complete. When the inexpensive local consoles are an integral part of the local station, there are always enough consoles to use and testing does not always require two people--a common occurrence in systems that support only central control consoles.

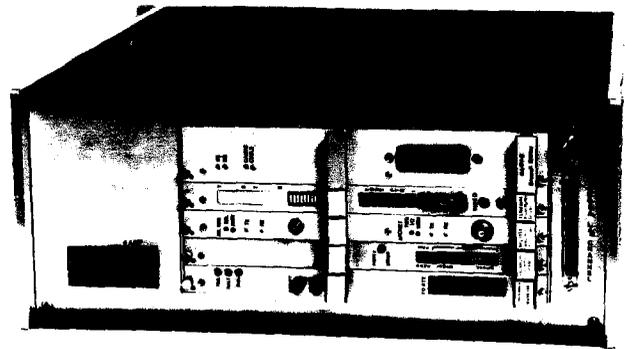


Figure 5. Rear View of a Local Console that Houses the Entire Local Station

Because of the modularity, an entire subsystem can be calibrated and tested using the local station, perhaps at a vendor's facility. It is then installed and becomes immediately part of the entire system by simply plugging into the local area network. Future expansion is handled the same way.

An Example System

Given the basic tools described in the preceding sections, it is interesting to look at the design of a control system that uses the concepts described above. The local station provides the modularity, expandability, and local console capability. The remaining major design decision is the architecture of the local area networks. If our example system is to work with any size of accelerator, we should first consider a very large machine. A smaller accelerator will be just a subset of the large control system.

For the sake of definiteness, we consider an installation like the Fermilab accelerator complex that includes several large and small accelerators. Note that this is not the existing control system but rather an example of how such a facility might be controlled if it were built today.

Network Example

Figure 6 shows an overall networking organization that we will consider. This figure shows three groups of communication rings, R1, R2, and R3, connected by bridges. A bridge is simply a computer that joins two Local Area Networks. It receives messages from one network and retransmits them on the other, after making any required protocol changes. Ring 3 (R3) stations are the local stations described in earlier sections. A Ring 3 may be, for example, a service building containing refrigerators, power supplies, RF systems etc. In smaller accelerators, R3 may not exist and local stations connect to R2. R1 is the link that interconnects all the individual accelerators. It may also include consoles and file servers that hold archived data and programs. It seems that three levels of rings should satisfy most requirements. This architecture is chosen to allow local communication to be contained on a single network, and also to limit the number of stations on any single network. The failure

of any single node or network does not destroy the entire system.

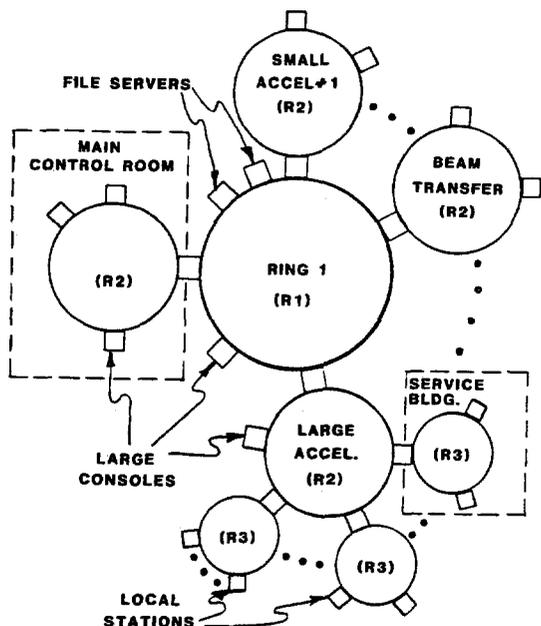


Figure 6. Example LAN Organization for a Large Accelerator

Although Figure 6 shows the communication paths as rings, the networks may be any type that fit the requirements. For example, at the present time, R3 may be ARCnet, R2 a few megahertz token passing ring and R1 an Ethernet. With these choices, the bandwidth of R1 will be higher to match the higher offered load and the bandwidth of R3 will be lower to match the more modest requirements. R2 tends to be geographically the longest link, and a LAN is needed that does not have a fixed length limitation. A slotted ring may be the best choice for R2 because the access time does not depend on the ring size. Of course the preferred LAN type to use may change as more networking products become available. The important point is that appropriate Local Area Networks do exist now, and it is not necessary to have all networks the same. In fact, not all R3 links need to be the same.

What's in a Name?

An operator tends to remember devices by their location, such as a particular valve in a refrigerator system in a specific service building of the Tevatron accelerator. In a big accelerator complex, it is virtually impossible to think up different names for devices without including location information in the name. The repetitive nature of accelerators is a useful mnemonic aid to reduce the number of different names an operator needs to know in order to interact with the accelerator. We would therefore suggest the use of a naming convention that includes the location explicitly and to let the name define both the geographical and the control system's network routing information. To do this in a familiar way, suppose the name of individual devices were patterned after

disk file names, so a general name looks like "R1:R2.R3.NAMES.EXT". Let the first three fields be one or two character fields; the device name itself is a six character field, and the optional extension could reference various properties of the same device. Then a Tevatron vertical flying wire detector located in the second diagnostic station in service building E0 would be named "T:E0.D2.FWVERT", and the D/A reference for Linac tank 3 quadrupole number 6 would have the name "L:T3..Q6.DA". The wildcard characters "?" and "*" could be used to interrogate the system about various areas of the accelerator. For example, if one asked for L:T3..Q* the system would return all the names of Linac tank three devices that begin with the letter "Q". The names of all the Linac local stations could be found by asking for "L:*". These requests use the broadcast feature of the LAN.

A local station would only know the name and extension fields for its own devices (along with the detailed database information). The contents of the R1 and R2 fields would be supplied to the local stations by the network bridge. In this way, identical local stations could be installed in some repetitive subsystem, such as a refrigerator or an RF system, and when that local station is plugged into its R3 network, its data is immediately available to the entire accelerator complex. Furthermore, the local station database could have been installed in the equipment at the fabricator's site or at a different laboratory as the subsystem was being assembled and tested. The control system is then a tool that is useful at early stages of development and fabrication, rather than something that is added on only after the subsystem is installed in the permanent equipment enclosures.

Data Base Considerations

A distributed system should have a distributed data base to facilitate system expansion and minimize communication bottlenecks. Each node of the entire complex houses its own portion of the database. Thus, both device data such as A/D readings as well as device characteristics such as scaling data or descriptive text are supported by similar requests. Consider an example of requesting information from a node on ring 3 about a set of devices. Using the naming convention described above, the requesting node separates out the set of device nodes by node names and forms separate messages according to each different node name. These messages are then transmitted to the bridge system for this ring 3. Each message is sent by the bridge either to other ring 3 nodes on the same ring or to other ring 2 bridges. Each message remains intact until it arrives at the named destination.

The destination node then examines the device name component and property designation to look up the requested data. Since the data may be requested to be returned at a specified frequency, data structures are created to facilitate efficient repetitive collection of the data requested. Later, that station uses the data structures to assemble a set of answer data and transmit it to the original

requesting node at the requested frequency. If the data request were for data base information, the requested frequency would presumably be specified as once-only, but the means of making the request would be the same as for any other type of data.

By dividing the data base support into the individual nodes, one eliminates the bottlenecks of a centralized data base, where all data base information for tens of thousands of devices may be stored on one mass storage system with the attendant loss of throughput. A search of an entire central database for a selected set of information may take 20 minutes or more. Using the distributed approach, such a search may be expected to take but a fraction of a second.

Programs as Devices

Each local station has a set of selectable application programs which can be invoked by an operator at the local console. There may be one or more library nodes in the network which house the most recent versions of application programs usable at local stations. When a program is requested by the operator at a console the library version of the program is interrogated to determine if the local station has the most recent version. If it does, then the local version is used. If it does not, the more recent version may be downloaded from the library node to the local station. Thereafter, the local station will have the most recent version for subsequent invocations of that program.

Programs may be referenced by name just as devices are. They may be devices which have a program header property and a program code property. The program header describes the type of program including language and execution CPU type, the program size and the date of creation of the version. The program code may be requested by use of the program header information.

Note that programs may be requested in a way similar to data base requests described earlier. Wildcarding may be used to search for programs in the network or to find the names of available programs. Accessing programs, database information and device data readings are handled in similar ways because they all involve merely the flow of information. This serves to underscore the fact that building a control system for a large accelerator means first building a viable local console facility and second constructing a viable communications network to make the distributed systems play together.

Discussion

In the foregoing sections we have described some of the modern techniques that are available to control system designers. Several major changes have occurred over the past 10-15 years. The availability of standard components and interconnecting schemes allow systems to be assembled from hardware supplied by many different vendors. No longer must a control system be locked into a single computer vendor's product line. Inexpensive, fast Local Area Networks are becoming more available. New technology can

be included in control systems as additions are made. The new 32-bit microprocessors have capabilities in the same league with a VAX-11/780. Portable consoles may be implemented by simply installing a network interface, such as ARCnet, in a portable IBM-PC compatible computer (and writing a little software!). A mixture of VMEbus-based 68000 computers, VAX's, and IBM-PCs can all be connected to the same network as long as each of them has a network interface card and understands the format of the messages. This allows the control system to include the excellent program development environment of a VAX, the economy of mass produced PC's and the configurability and high performance of microcomputers built around an industry standard bus. Some of the problems associated with developing software for different CPU types can be alleviated through the use of higher level languages such as C, Pascal, and Fortran. In general, the job has become easier because more components are available commercially.

However, it is still possible to make wrong choices and, in some cases, a single bad selection can cripple the performance and complicate the implementation of the entire system. For example, if a hardware link is used in place of a communication network, then stations cannot communicate with each other, local control of devices in other stations is nearly impossible and all traffic on the link must be funneled through a single computer. If a central database is used, then either local stations must depend upon the links and other computers to function, or there are immediately two copies of the database--the local one that the hardware uses and the central one, which may be incomplete or wrong, that everyone else uses. A decision to use one vendor's computer or one type of I/O interface throughout will cause the system to become obsolete quickly.

Many of the techniques described here have been applied to the Fermilab 200 MeV Linac control system; others are being considered for use in the National Superconducting Cyclotron Laboratory at Michigan State University. New developments exist that have not yet been incorporated into a system. Software techniques that are being used in Computer Aided Design (CAD) systems are directly applicable to accelerator control systems. The use of pull-down menus, icons, and multiple windows displayed on a very high resolution color graphics terminal could eliminate several lower resolution displays that are now used. Some of the CAD "work stations" contain 68000 processors, megabytes of memory, hundred megabyte Winchester disks, 1000 by 800 color graphics terminals and Ethernet communication interface. They could be considered full pre-packaged consoles that can replace the minicomputer based consoles that are now used. Control system designers should be aware of systems and integrated circuits designed for communication, process control, and military markets. Accelerator control needs are very similar to process control and CAD requirements, and solutions developed for those areas often apply directly to accelerator control.