A DISTRIBUTED LINAC CONTROL SYSTEM FEATURING SDLC LOOP COMMUNICATION

R.W. Goodwin and M.F. Shea
Fermi National Accelerator Laboratory†
Batavia, Illinois

Summary

A distributed control system has been designed as a possible replacement for the existing analog and digital interface equipment in use at the Fermilab 200-MeV linear accelerator. In addition to replacing the present interface equipment that is no longer supported by the manufacturer, the goals for a new system include improved reliability, easier maintenance and faster monitoring of accelerator parameters. About 15 local microcomputers, one for each major linac system, would be fabricated from commercially available Multibus-compatible hardware. These local stations can operate as stand-alone systems, providing control and readout of parameters to facilitate maintenance of the accelerator components. Values of all critical devices can be monitored at the accelerator repetition rate (15 Hz), so that beam may be inhibited by any out-of-tolerance reading on a pulse-to-pulse basis, to prevent unnecessary and damaging beam loss.

One of our design criteria is to maintain the fast response and interactive nature of the existing system. The operation of a group of individual processors as a single integrated control system depends on the type of communication link used to interconnect the local stations with the operator's console computer. For this link, the Synchronous Data Link control (SDLC) in the loop configuration was chosen. Using SDLC interface circuits with direct memory access controllers, arbitrary length blocks of binary data can be transmitted and received at bit rates of 1 MHz. Details of the hardware and software organization of this control system design will be discussed.

Introduction

The original Fermilab linac control system has been in operation for about 10 years.[1] Although the system still operates well, the interface equipment and the X530 control computer are no longer supported by the manufacturer. A change of computer would necessitate a change in the interface equipment because this equipment is structured as an extension of X530 I/O bus. However, because the linac I/O is integrated into the console computer itself, it is a very fast system. The four linac digitizers are run simultaneously by the 530's I/O processor, placing data in memory under DMA control. Because of this capability, all the linac analog data ($\sim$ 800 channels) are collected for each accelerator cycle (15 Hz) so that current correlated data are always available to the application and monitor programs. Although binary data are not collected for each cycle, the status can be read as a single 6μs instruction by the program that needs the status information.

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

It is this type of high performance system that we are attempting to replace.

System Overview

A new control system that is being designed today should almost certainly be built with small, modular, intelligent local stations, each controlling a part of the accelerator. Experience with such systems in the Fermilab Cancer Therapy Facility and Preaccelerator areas, has shown them to be extremely flexible and reliable additions to the control system.[2,3] The new design has a microprocessor controlled station for each system of the linac. The local stations are relatively straightforward. They include the processor, a small console, and the analog and digital interface equipment. Each system is assembled from a collection of commercially available cards that are compatible with an industry standard bus structure, such as Multibus§. The more difficult part of the design, and the part that will determine the performance of the entire system, is the communication link that connects the local processors to the console computer. The response of the console (or consoles) will depend in detail upon the transmission speed, the message handling overhead and the turnaround time for receiving data requested from local systems.

It was decided to use a serial bit-oriented protocol, such as HDLC (High Level Data Link Control), to provide the communication to and from the local stations because it is fast; it is an industry standard[4]; it is supported by large-scale integrated transmitter-receiver circuits supplied by semiconductor manufacturers. The link overhead when using a DMA controller is low to allow the high performance required of the communication channel.

HDLC is a superset of the original IBM bit-oriented specification SDLC (Synchronous Data Link Control) protocol.[5] Both use the same message format, but SDLC contains some simplifications and includes a loop mode that is particularly useful for control systems.

The SDLC Protocol

Data is transmitted on a SDLC link in a format called a frame. All frames begin and end with a flag. Between the opening and closing flag, a frame contains an address field, control field, information field, and a frame check sequence as shown in Fig. 1.

OPENING FLAG	ADDRESS (1 BYTE)	CONTROL (1 BYTE)	INFORMATION FIELD (ANY NUMBER OF BYTES)	16 BIT CRC FRAME CHECK SEQUENCE	CLOSING FLAG

Fig. 1 SDLC Message Format

§Multibus is a trademark of the Intel Corporation.

The flag character is the binary pattern 01111110. It provides the frame boundary and reference for the position of other fields within a frame. The address and control fields are each one byte long. The address byte is used by the primary station to direct a message to a particular secondary. When a secondary transmits a response, it includes its own address in the address field. The control byte identifies the type of message. The information field may be any length (including zero)--it contains only data. The frame check sequence field is a two byte cyclic redundancy check (CRC) to test the integrity of the transmission. In this protocol, the link overhead is the same 6 bytes for any length message so that long blocks of data may be sent very efficiently. Only three special bit patterns are used: flag, abort (8 "ones") and inactive idle (15"ones"). All three are handled by the hardware.

In the loop mode of operation, a link controller or primary manages the message traffic on the loop. Data flow is always in the same direction around the loop and each station re-transmits the data it receives, so that messages sent from the controller are returned to the controller.

Two characteristics of the loop mode are extremely useful for the present system: 1) a controller can send data to all stations on the loop with a single transmission; and 2) with one poll sequence the controller can poll all the secondaries and receive a message from each secondary in turn. This drastically reduces the link controller overhead for most of the communication with secondaries.

In the quiescent state, the loop contains a continuous stream of flag bytes. If a secondary receives no flag bytes for a period of time, it transmits a special message, called a beacon, to the primary. This flag timeout will occur if a problem developed in the link, or in a station on the link. When the beacon is received by the primary, the address contained in the beacon message is the location of the last station that is operating normally. This feature is useful for locating malfunctions on the loop.

The SDLC protocol would be difficult to implement if it were not for the LSI circuits that have been made available by several semiconductor suppliers. For this application the Motorola MC6854 data link controller was selected, operating in conjunction with the MC6844 DMA controller. Together they allow a processor to transmit and receive messages at a 1-MHz baud rate under direct memory access. The processor need only set up pointers and byte counts and the rest of the work is done by the controllers. An interrupt is received when the message transfer is complete. The link controller also performs the functions of flag detection, CRC generation and detection, and zero insertion-deletion--a technique that causes flags to be unique while allowing binary data to be transmitted. The transmitter inserts a zero after any succession of five 1's within a frame and the receiver deletes any zero that follows 5 successive 1's. Only when a flag, abort or inactive idle is being sent will the controller permit more than 5 consecutive 1's. This operation is handled entirely by the controller--the processor is unaware that the zero insertion and deletion is taking place.

Hardware

The individual microcomputers will be fabricated from commercially available Multibus modules and card cages. Necessary cards that are not available will be designed and fabricated to comply with the Multibus specifications.

One card that will be fabricated is the CPU-Link card. This will be an MC6809-based single board computer that will include the MC6854 link controller and the MC6844 DMA controller. The card will also contain RAM, ROM, a programmable timer (MC6840) an asynchronous interface (MC6850) and a parallel I/O (2-MC6821). This computer card is expected to become a general purpose building block. Because of the bus arbitration logic, an additional CPU can be inserted in a system where more capability is needed. For example, a second CPU card in the preaccelerator ground station will provide an SDLC link to the 750-kV dome to operate the ion source electronics.

The processor's programs and fixed data are programmed into UV erasable PROM memories. In this way the local stations can restart in an operating state following a power outage. Descriptor information and nominal and tolerance values for accelerator parameters need to be updated only occasionally. These data will be placed in non-volatile magnetic core memory.

Local Systems

Each major linac system will be controlled by a small local microcomputer as shown in Fig. 2.

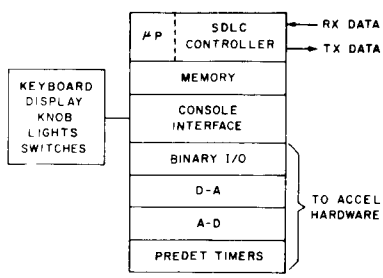

Fig. 2 Local Control Station

These will include a small console, loop interface, A-D, D-A and binary I/O.

A single Multibus card provides the operator's console interface. It contains a 16-line x 32-character video display generator, a keyboard input, an incremental shaft encoder input with up-down counter, and three bytes of binary I/O

for miscellaneous sense switches and indicators.

Loop Primary

The primary station will control no hardware except for a small console that is used to monitor the operation of the communication system. The link controller can display the number and length of messages being transmitted and received each cycle, the time spent polling secondaries, link error rates, and link malfunction locations.

Connection to Consoles

The primary also connects to other computers that service consoles in the control room. To make the system separate from the console computers, that communication occurs through the use of shared memory as shown in Fig. 3A. In a Multibus system, multiple processors automatically share memory because of the bus arbitration logic included on the processor boards. A second method of sharing memory is possible through the use of a pair of commercially available Multibus-compatible two-port memory modules that connect two independent Multibus systems (Fig. 3B).

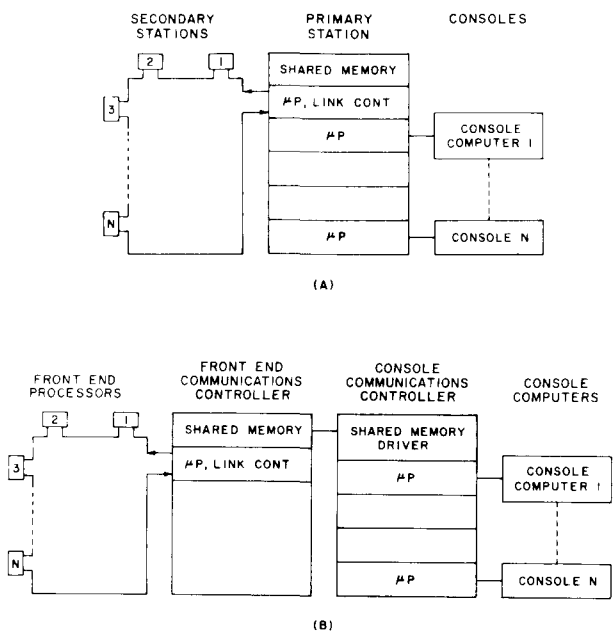

Fig. 3 Interconnection of Primary, Secondaries and Consoles

With this method, the system can be expanded to include additional Front End Communication Controllers each connected to the Console Communications Controller via separate shared memory. All consoles can then request data from any front-end communications controller in the network.

In Fig. 3, the block labeled "Console" is usually some computer that executes application programs, operates console hardware and requests and receives data from the linac through the shared memory. The connection to the console computer may be SDLC, byte serial DMA or some other communication link. A small console could be driven directly by the microprocessor--that is, no processor in the box labeled console. These various options are indistinguishable to the link primary. Each is simply a console, requesting data.

The Physical Link

Because the link is synchronous, a clock must accompany the data stream. This clock will be generated in the primary, transmitted around the loop and returned to the primary. The clock and data will be encoded and transmitted as a single, self-clocking signal using either biphase or Manchester format. Connection between chasses can be RS422, transformer coupled or fiber optic. The link driver-receiver electronics may be built as a separate chassis to allow loop bypass for an off-line station that is powered down.

System Operation

Functions of the Primary

The primary station is, in a very real sense, the loop controller. No data are placed on the link unless the primary initiates or solicits the transmission. Secondaries transmit only when polled, transmit only the data requested in the poll, and only one message per poll. In this way the primary can maintain a synchronous character to the link traffic. When a response is received, it is the answer to a specific question, not an arbitrary message initiated asynchronously from an autonomous secondary. This tight control exercised by the primary implies that a poll sequence must be sent to solicit all of the possible responses from the secondaries. This is feasible because programs for the secondaries are static; that is, it is not intended that code will be sent to a secondary for execution. This possibility is not excluded by the design and could be incorporated for some other application if needed.

Functions of the Secondaries

Secondary stations perform the task of data acquisition and monitoring for the local hardware. Each cycle, the secondary digitizes all the analog data and reads the digital status from the hardware so that a current data pool is available in the secondary. Then the secondary prepares groups of data to return to the primary. When this work is completed, the processor executes the application program to service the local console and update the video display.

Link Messages

This section describes the organization of the messages transmitted on the link to show how the individual stations and the link controller operate to collect the data required by the consoles. Three separate types of message traffic occur on the link.

Single-Terminal-Address Messages

When the primary station has an individual message to be sent to a controller, that message will contain the address of the target secondary station. This type of message is used to transmit settings to selected local hardware or to ask for data that is located only in that station, such as a memory dump. As in all other transmissions, the response from the secondary station must await the poll sequence from the primary.

All-Parties-Address Transmission

The all-parties address is used when the primary intends to send data to all secondaries simultaneously. Through the use of this transmission, the primary can assemble a group of readings collected from the present beam pulse and, with a single transmission, place these data in the memory of all stations on the loop. A nominal list of data will be sent each cycle so that each secondary will contain a "remote data pool" of selected readings from other stations. Then, for example, the Tank 4 system can display a summary of the linac operation without requesting data from the primary.

Poll Sequences

The poll command instructs stations to prepare to send data to the primary. The poll command will include an information field to allow the primary to select the type data to be sent. This is normally just the list ID number that had been sent at some earlier time. Many of the poll commands will be directed to the all-parties address (addr FF) and each station then prepares to send its contribution, if any, to the requested data list. Single-terminal-address poll commands can also be used to solicit response from a single secondary.

Several poll sequences will be done each cycle to request a variety of information: A-D data, out-of-tolerance parameters, secondary station status, etc.

List Organization

The primary communicates with the console and with the secondaries in terms of "lists". Data needed by the console, called a "request" list, is presented to the primary as a random list of channel numbers. When the data are acquired, a parallel "answer" list is assembled to return the data to the console in the order it was requested. Each channel in the request list has an index number that is simply the location of that channel within the list. In the course of retrieving the data, secondary lists are generated, containing the index numbers, requested channel numbers and the answers grouped according to the linac station where the data originate.

A console request is serviced as follows: the primary receives the request list, gives it a list ID number and sends it to the secondaries using the all-parties address. All secondaries search the list for channel numbers they recognize and construct their own secondary request list and a secondary index list. Each machine cycle the secondary station goes through its request list and builds an answer list from readings in its data pool. The primary will poll once for the secondary index lists and poll each cycle for the secondary answer lists. When the primary performs the index poll, it will receive all of the secondary index lists grouped by secondary stations in the order the secondaries are connected to the loop. When the primary polls for the answer list (which it does at 15 Hz), the secondary answer list will be in exactly the same order as the index list. The primary can then use the entries in the index list as pointers to the console answer list for returning the data to the console.

Using the technique described here, neither the console nor the primary needs to know which secondary is responsible for measuring each data channel; the secondary itself can recognize its own channels and respond accordingly. Also the primary does not know (or need to know) how many stations will respond or how long their response will be. The primary simply receives messages till the poll is finished. Secondary stations may be added to or taken from the loop without changing the primary's software. New data channels would become active automatically; data channels that are not recognized by any secondary would simply not be updated. This sequence is repeated for other consoles or requestors. Note that a requestor may be a secondary station on the loop. By this mechanism a local station can request data from any other station on the loop, so that the operator at any secondary could select a display that would show, for example, the gradient of all nine linac tanks along with the preaccelerator high voltage and source pressure. By keeping the lists separate for each requestor, the primary can cancel a given list with a single short transmission to the secondaries. The addition or cancellation of one list has no effect on other lists.

In normal operation the lists are relatively static, changing on the order of seconds or longer. When the operator calls up a new display, that act will usually result in the cancellation of an old list and the generation of a new one. The new list will remain active as long as the console display is selected. It makes little difference to the secondary if a given list is polled on a given cycle or not. The primary can choose to poll certain lists less frequently, such as only on beam cycles. Answer lists in the secondaries would be overwritten with new data each cycle in any case. Servicing the poll sequences is done at an interrupt level so that monitoring of the data can proceed at the background level.

Timing Considerations

To achieve the interactive feel of the control room consoles, careful attention must be given to servicing the console requests immediately. With the system described here, it is felt that

a console request list presented to the primary during one cycle will be transmitted to the secondaries during the same cycle and data will be polled from the secondaries on the next and all succeeding cycles. Settings sent to the primary will be passed on to the secondary on the same cycle, and changes in the hardware will reflect the changed settings on the next cycle.

Detailed timing is calculated assuming the instruction set of the MC6809 CPU. This processor is now available and is particularly adept at handling lists of data and addresses. The primary station can service one message and be ready for the next message in about 100 μs. To this one must add 50 μs for the transmission time of the six bytes of SDLC protocol, so the message overhead is about 150 μs per responding secondary. For a poll that results in 10 secondaries returning 50 bytes of data, the total time of the poll is less than 2 ms. Even if 10 such polls were conducted each cycle, only 20 ms of the 67 ms cycle would be needed for collection. The time to sort the data for return to the console in the order requested is 20 μs per channel.

An application program that plots all the linac quadrupole currents at 15 Hz, results in a request for 171 words (342 bytes) of data. This request takes 4.3 ms to collect and 3.4 ms to sort, for a total of only 7.7 ms. These times indicate that the system is fast enough to service several consoles at 15 Hz.

Present Status

Although no decision has been made regarding a replacement for the linac control system, we are implementing this system to the point that a link controller will communicate with several secondaries and with a separate processor that supports an operator's console. At this stage we can confirm the timing calculations and study the integrity of the SDLC loop communications.

A local console described here has been fabricated using Multibus card cages and commercially available A-D, D-A, memory and binary I/O cards. The console controller card, including the Video RAM display, keyboard, shaft encoder interface and two bytes of sense switches and status indicators, has been prototyped. This console is now being used to control the negative ion source in the terminal of the second Fermilab preaccelerator.

The SDLC-DMA link has been operated at 1 MHz and has shown that messages can be sent error free from one processor to another. A Multibus single board computer card that includes the MC6809-SDLC-DMA combination is being fabricated.

The distributed control system described here is a reasonable replacement for the existing linac control system. The 1-MHz SDLC loop is fast enough to collect all the data in a single cycle, although we expect to transmit only the data requested by the consoles. Limit checking and software closed loops will be done in the local stations. Because of the modularity and local console facilities, the system should be much easier to maintain than the existing system.

References

1. E. W. Anderson, H. C. Lau and F. L. Mehring, "The Computer Monitoring and Control System for the NAL 200-MeV Linac", Proc. of the 1970 Proton Linear Accelerator Conf., Natl. Accel. Lab., pp. 451-469.

2. R. W. Goodwin, R. F. Kocanda and M. F. Shea, "A Microprocessor-Based Preaccelerator Control System", Proc. of the 1976 Proton Linear Accelerator Conference, Chalk River, Ontario, AECL-5677, pp. 264-268.

3. R. W. Goodwin and M. F. Shea, "The Microprocessor-Based Control System for the Fermilab Cancer Therapy Facility", Trans. Nucl. Sci., NS-25, 496(1978).

4. American National Standards Institute, ANSI Pub. No. BSR X3.66.

5. International Business Machines Publication Nos. GA27-3093-1 and GA27-3098-0.