A Programmable Beam Intensity Display System for the Fermilab Accelerators

S. Johnson, D. Capista
Fermi National Accelerator Laboratory *
P.O. Box 500, Batavia, Illinois 60510

Abstract

A programmable system has been constructed to provide an easy means to display beam intensities and efficiencies from the various Fermiblab accelerators. The purpose of these displays is to provide an alternative display method for this information. The system consists of a central microprocessor, a data acquisition subsystem, an interface to the Fermilab control system, and a link driver to drive multiple display boxes. Each display box has eight displays, with a display consisting of the name of the parameter being displayed, the units for the parameter displayed, the time the data was taken, and the actual data. The final system will have a display box at each control console in the Main Control Room. The system is also a backup way to view accelerator performance when the accelerator control system is down for any reason.

Introduction

At Fermilab there are approximately eighty intensity parameters from the accelerators and transport lines. The programmable beam intensity display system gives the user a hardware display to view the intensity parameters of interest, usually to aid in making adjustments to machine parameters. Each display box will be programmable through any control console.

The system consists of a VME crate which contains the following cards: CPU, Token ring interface, MADC, scalar , display link driver, and an accelerator clock monitor. The CPU uses the link driver to communicate with the displays. The token ring interface is for communication with the accelerator control system. The CPU uses the other three cards for data acquisition.

Available Signals

There are several types of signals available which this data acquisition system uses. One type of signal is a voltage which is proportional to the beam intensity. The Linac and circular accelerators at Ferimilb use a voltage type signal to provide real time beam intensity data. Some of the beam transport lines also use a voltage signal to represent the integrated intensity which passes through the line during a machine cycle. Another type of signal is the TTL level pulse train or scalar. Pulse train signals are found in some of the beam transport lines, particularly in the Switchyard experimental areas. These signals represent the total integrated intensity an area receives by the number of pulses the area returns.

System Hardware

The system hardware consists of a VME crate and several display boxes. The VME crate contains several cards which are commercially available and several cards which have been built in house. The display boxes and the VME crates' scalar, link driver, and accelerator clock monitor cards are in house design and construction. The CPU, MADC, and token ring interface cards in the VME crate are commercial cards.

The Fermilab Accelerator Clock contains information about the status of the various accelerators such as: injection, start of acceleration, end of acceleration, and extraction [1]. The accelerator clock card in this system monitors these eight bit clock events so that the CPU will know when to sample the various signals. The CPU tells the clock card which events to monitor. When one of these events occur, the clock card stores the event data in a register and interrupts the CPU (fig. 1).

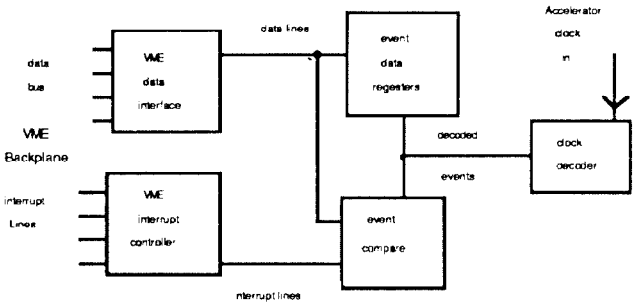

Figure 1. Accelerator Clock Card

The CPU then processes the interrupt and reads the event data. The clock card is capable of four priority levels of interrupts with a storage register for each interrupt. These four levels of interrupts are necessary since several clock events may occur close to each other in time.

The CPU communicates with the display boxes through a parallel link. Each transmission consists of sixteen bits of address and sixteen bits of data. When the CPU communicates with the link driver, the address and data information are moved into latches. The link driver then clears communication with

*Operated by the University Research Association under contract with the U.S. Department of Energy.

0-7803-0135-8/91$01.00 ©IEEE

the CPU to allow it to perform other tasks while the link driver transmits the information on the link.

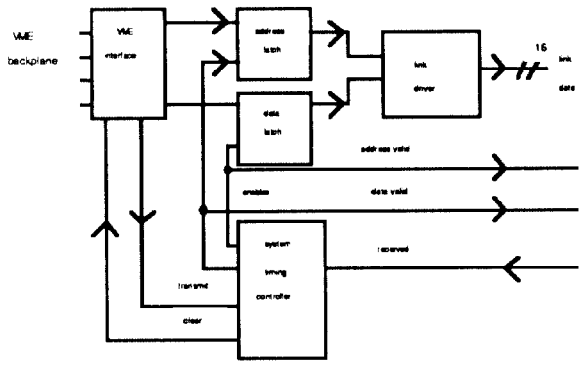

Figure 2. Link Driver Card

After the VME interface moves the address and data information into the latches, it signals the system controller to begin transmission. The system controller then moves the address information onto the sixteen bit link and drives the address valid line true. Next the system controller waits for the display box to respond to the address by driving the received line true. When the system controller views the received line true, it drives the address valid line false and waits for the received line to go false. The system controller then moves the data to the link and drives the data valid line true. The display box which responded to the address processes the data and responds to the link driver by again driving the received line true. To complete the transmission, the system controller drives the data valid line false and issues a clear to the VME interface to allow further transmissions. Each transmission requires about 1us.

Each display box on the parallel link needs to repeat the link information to the next box, decode addresses, pass data to its displays, and respond to the link driver (figure 3).

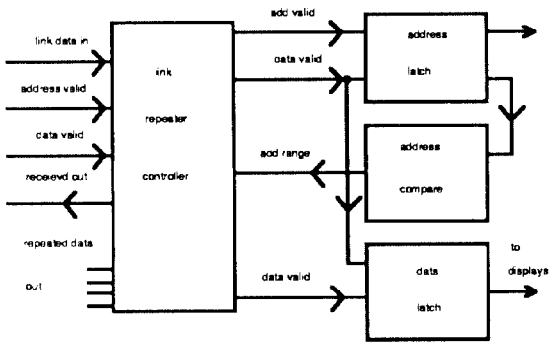

Figure 3. Link Repeater/controller

Once the link driver places the address on the link and drives address valid true, each display box will receive the address through its link repeater/controller and move the information into the address latch. The address compare unit then looks at the most significant byte to see if the address is for this box and if the compare is true, the address range line will be driven true to notify the link repeater/controller. Upon receiving address range true, the link repeater/controller will drive the received line true to notify the link driver it has the address. The link driver then puts the data on the link and drives the data valid line true. Next the link repeater/controller receives the data and with the address range line still true, transfers the data to the data latch and routes the address and data information to the displays. To complete the link communication, the received line is driven true by the link repeater/controller.

Each of the eight displays in a display box consists of four numeric and sixteen alphanumeric displays (figure 4).

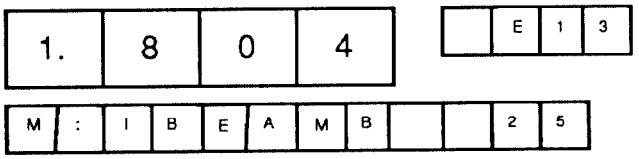

Figure 4. Display layout.

The four alphanumeric displays to the right of the numeric displays represent the units of the data. The twelve alphanumeric displays under the numeric displays represent the the name of the parameter and the accelerator clock event the display updates on.

Software Considerations

The software for the system consists of four major components. These components consist of an operating system, networking software, user interface, and the actual data acquisition and display software. The system software services are provided by the MTOS operating system. This package is a commercial product supplied by Industrial Programming Inc.. The other software components are layered on top of MTOS.

MTOS is designed as a real time operating system for use in embedded applications. MTOS provides multitasking abilities, along with intertask communication facilities. Examples of the services provided by MTOS are semaphores, controlled shared variables, event flags, signals, message buffers, and mailboxes. A prioritized scheduling algorithm is used by MTOS to make sure that critical tasks are run on time. Since the operating system design is for embedded applications, MTOS itself does not support program development. All development work is done on a separate machine, and the finished code is downloaded to the system.

The second part of the software is the network interface. This allows the system to communicate with the Fermilab ACcelerator NETwork or ACNET. The physical connection to the network is through a Token Ring interface. The network interface is provide for convenience in changing the data acquisition setup. This part of the software was written in house at Fermilab.

The user interface task allows the user to change the setup of the system from an attached terminal. This same task is

also called by the network software when it changes the data acquisition setup. This task runs at a very low priority and will only make changes when all other tasks are idle. This task also provides debugging routines through the attached terminal for both the hardware and software.

The last part of the software is the actual control and display software that makes up the system. This part of the software can be viewed as four different tasks. Each of these tasks provide some function that can be preformed independently of the other tasks. The data is then passed between the different task via software mailboxes.

The data acquisition is controlled through a central data structure. This structure is an array of linked lists, with each list corresponding to one reset on the Accelerator Clock system. A node on this list will contain the following information: index of the disired parameter, raw data, scaled data, and one entry for each box on the link. The lower eight bits of each box entry correspond to the eight displays on each box. This information is used to determined which displays are updated for this node. When adding or deleting nodes from the linked list, the user interface task will first lock all other tasks from the list to prevent corrupting their pointers to the linked list. Figure 5 shows an example of a node on the linked list.

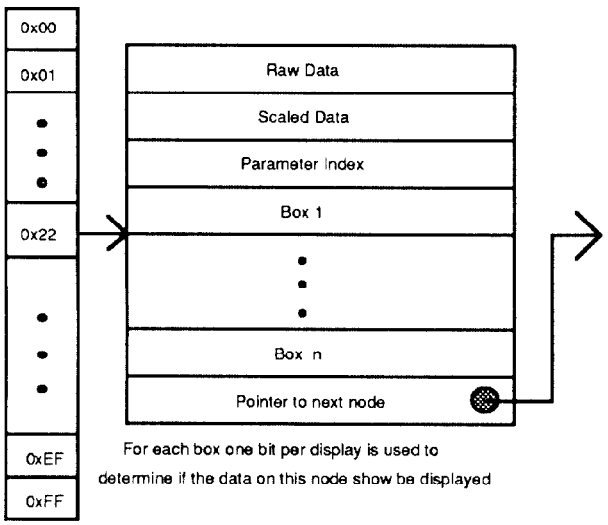

Figure 5. Node on Clock Event Linked List

A very simple database is also maintained in the system. There is one entry for each data parameter, and the database is accessed using the parameter index stored in the node on the linked list. The database provides information on what card and channel to read to get the raw data, how to scale the raw data, and in what units the scaled data is displayed.

The first task reads the clock event to be serviced off the Accelerator Clock card. This task interfaces with the interrupt handler in the system software to provide this information. Once it has read the clock event it passes the information on to the next task.

The second task takes the clock event provided by the first task and traverses a linked list associated with that clock event. Each node on this linked list is a channel on the scalar or MADC card that should be read. The task reads each channel, stores the raw data in the linked list, and the passes onto the next task the clock event to be processed.

The third task again takes the clock event being passed to it and traverses that link list. This time the task will take the raw data that was collected by the last task, and scale that data and put it into a displayable form.

The last task takes the clock event data and again traverses the linked list. Encoded on each node is the box and display number for where the information should be sent. Figure 6 shows a view of each task's input and output.

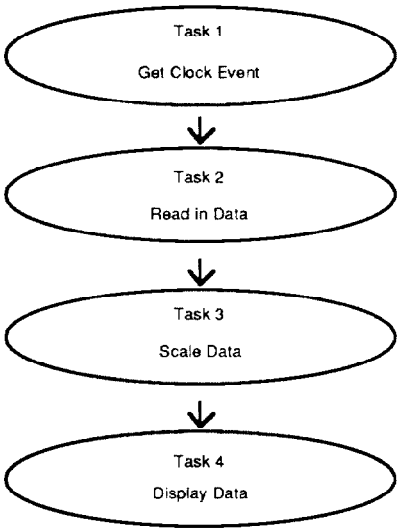

Figure 6 Task Input and Output

Summary

The display system is currently up and running in the Main Control Room at Fermilab. The interface between the network and the data acquisition software still needs to be improved. This should be completed in the next couple of mouths. Additional display boxes are being built so that a box will exist at each console.

Acknowledgements

We would like to thank Linda Klamp who came up with the original ideas for the system, and who was involved with the early programming efforts on the system. We would also like to acknowledge the efforts of James Morgan who is doing the actual display box construction, and Charles Robertson who designed the scalar card.

References

[1] R. J. Ducar, "Tevatron Clock Event Assignments", Fermilab controls hardware release No. 17.35, February 1991.