SLAC–PUB–6149
April 1993
(A)

MPS BEAM CONTROL SOFTWARE ARCHITECTURE*

K. Krauter and M. Crane
Stanford Linear Accelerator Center, Stanford University, Stanford, CA 94309 USA

Abstract

The new Machine Protection System (MPS) now being tested at SLAC has a beam control subsystem resident in processors located close to the beam monitoring devices within the machine. There are two types of beam control micros: Algorithm Processors (AP's) which collect and evaluate data from monitoring devices, and a Supervisor (SUPE) which collects and evaluates data from all the AP's. The SUPE also receives the global machine beamcode indicating beam presence, and passes it on to the AP's. The SUPE receives the beamcode pattern from the Master Pattern Generator (MPG) via a shared-memory communication link. MIL-1553 serial communication is used between the SUPE and the AP's, and between the AP's and the monitoring devices. Multitasking software is used to allow high priority handling of data evaluation and low priority handling of host/user interfacing and event reporting. Pipelining of data between acquisition and evaluation and reporting is used to accomodate the processing capacity, while still supporting full processing at the 360Hz broadcast rate of the beamcode pattern.

1. MOTIVATION

The beam control subsystem of MPS does the actual per-beam data acquisition, data processing, and resultant beam control. Data from beam monitoring devices is acquired on a per-beam basis and processed to determine whether a machine-endangering condition (fault) has been detected. If a fault is found, then the data processing produces a result which indicates the rate at which the beam may be operated safely given the presence of the fault. When the fault is corrected, through steering or some other correction, the processing produces a result which requests that the beam operate at full rate again. If a fatal error internal to MPS occurs, the beam is completely disabled until the fatal flaw is corrected. The full functionality is documented elsewhere [1].

The per-beam data acquisition must be done at the same rate as the beam frequency, and it must be done as soon as possible after the beam has passed a given beam monitoring device, in order to acquire timely per-beam data such as material temperature and radiation level. The acquired data must be processed at the same frequency as well, in order to respond immediately to changes in per-beam data. High speed processors and communications and a fast realtime operating system are therefore required in the beam control VME crates.

Processor and communication limitations necessitate the placement of several beam control micros along the length of the accelerator machine, each acquiring a subset of available per-beam data. These are called Algorithm Processors (APs) because they process acquired data by passing it through a logic machine called an algorithm. An algorithm is a list of tests dictating the level at which the beam may operate for each state detected by the beam monitoring devices [2]. Each AP produces a single result indicating the level at which the beam may safely operate in that AP's region of the machine. The results of all the APs are sent to the SUPE, a special consolidating AP. The SUPE processes the APs' results by passing them through a supervisor-level algorithm, producing a single result which is used to control the beam.

A secondary function of the beam control subsystem is to keep the operators in the main control center apprised of the machine status. Both APs and the SUPE report periodically which beam monitoring devices faulted and what beam level changes (events) have resulted. These reports are handled in the main control center VAX by the MPS code located there.

2. HARDWARE ARCHITECTURE

The beam monitoring devices have always been configured and read through CAMAC interfaces and SLAC's SLCNET network. MPS adds an additional serial MIL-1553B interface to these same devices which connects them to their respective APs. The APs are connected to the SUPE via MIL-1553B also, and the SUPE is connected to the beam controlling Master Pattern Generator (MPG) via a shared memory interface.

The AP and SUPE hardware consists of a 6-slot completely enclosed fan-cooled VME crate, each with a processor card and some number of 1553 cards. The SUPE also has a card for the MPG interface. The processor card is, currently, a Force CPU-30 68030 with ethernet. The VME-based 1553 card is adapted from work done at Fermilab. The SUPE-to-MPG shared memory interface is Bit-3 which is commercially available.

Both the APs and the SUPE are connected to the main control center VAX by LEBNET (Linac Ethernet Backbone NETwork). Figure 1 illustrates the hardware layout.

3. SOFTWARE ARCHITECTURE

The beam control micros use multitasking to support both high frequency per-beam processing together with low frequency user request interfacing and event reporting. Each of the Tasks on the micro generally has a corresponding Manager on the VAX [3]. The operating system chosen for the micros was pSOS+, in part because it had the best VMS cross development environment.

All communication between the micro MPS tasks and the VAX MPS managers is over TCP/IP through the new Area Message Service (AMS) which makes software access

* Work supported by Department of Energy contract DE–AC03–76SF00515.

Presented at the Particle Accelerator Conference (PAC '93), Washington, D.C., May 17–20, 1993.

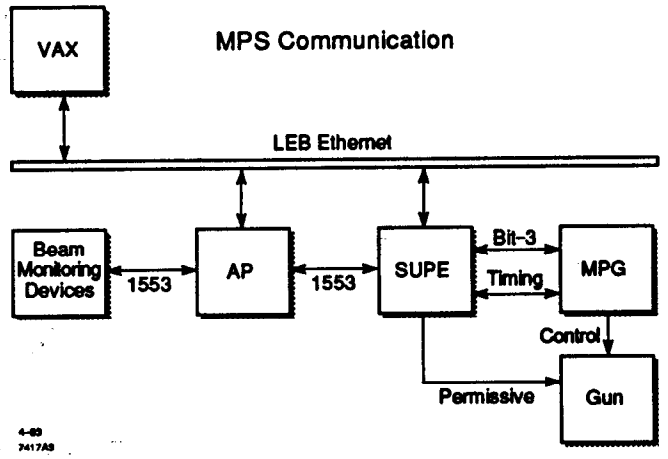

Figure 1. Hardware layout.

to the ethernet appear identical to the already existing software access to SLCNET [4]. The Slow Task also makes direct NFS calls to acquire algorithm description files from the VAX. There are two rules about I/O on the micro which dictate much of the architecture. Per-beam code may not do time-consuming ethernet I/O, and high volume small size message traffic must not be allowed to congest the ethernet. Per-beam data acquisition and processing is the highest priority task in the micro, doing the most important job—controlling the beam. Only when the micro is waiting between beams may it receive and answer ethernet requests, or ship events or error reports on the ethernet. Congestion is avoided by enforcing a minimum time interval between ethernet sends in each task. If a task is requested to send up a new event or error before this minimum time has elapsed since the last send, then the request is buffered up until the time interval has elapsed.

The micro Slow Task runs in the micro constantly and handles requests from the Slow Manager (also called the MPS controller) on the VAX, such as downloading MPS algorithms and reporting micro status. When the Slow Manager requests MPS be made operational, the micro Fast Task is started up to do per-beam processing.

The Fast Task spins off data acquisition I/O requests at up to 120 times per second, and does data processing in parallel. Always accompanying the Fast Task is the Event Task. Whenever the Fast Task detects a beam level change result (an event) from the data processing, the event is passed to the Event Task. The Event Task buffers the event as necessary and sends the event buffer to the Event Manager on the VAX, where the events are processed into displays and summarizations.

The Milcom Task waits for requests from the Fast Task to begin a data acquisition cycle. Since the configuration of the beam monitoring devices read by an AP is unchanging, the data acquisition commands can be prepared in advance and executed in a timely way on a per-beam basis.

Any errors or messages which the micro (Slow, Fast, Event, or Milcom Tasks) would like to report directly to the operators at the main control center are passed to the micro Error Task. The Error Task buffers the error as necessary and sends the error buffer to the Error Manager on the VAX. See Fig. 2 for a task data flow schematic.

4. FAST TASK

The Fast Task must complete a number of functions between each beam pulse that occurs on the machine. The original design assumed the worst case of the beam coming at 360Hz, allowing 2.7 ms between beam pulses for the Fast Task to complete a full processing of beam pulse data. It was quickly recognized that 2.7 ms was not enough time in which to sequentially acquire the data and process it and report on it. This was enough time only if the Fast Task were able to start the data acquisition and data processing at the start of the cycle and run them in parallel. The Fast Task accomplishes this with a two-stage data pipeline in which the data acquired in one cycle isn't processed until the next cycle.

The two-stage data pipeline is further complicated by the fact that the APs and the SUPE also represent a form of pipelining in that the APs must fully process their data and pass their results up to the SUPE before the SUPE may begin processing those results and pass its own result up to the MPG. The flow of the data must then be as follows. An AP starts a data acquisition cycle in cell ZERO of the pipeline. The AP starts the data processing on that same data in cell ONE, and at the end of cell ONE sends the result up to the SUPE. The SUPE starts a data acquisition cycle in cell TWO, acquiring the result data sent up from the APs in cell ONE. The SUPE starts the data processing on that same data in cell THREE, and at the end of cell THREE sends the result up to the MPG. Therefore a final beam controlling request from the SUPE is not generated until four beam pulse cycles after that beam has occured. Figure 3 shows the pipeline synchronization.

Only after the Fast Task has completed the result reporting at the end of its second stage may it finally send status information up to the main control center VAX. This is done at the end of the second stage of the pipeline.

An Interrupt Service Routine (ISR) has been set up on the SUPE to awaken the Fast Task whenever the machine timing system determines that a beam pulse is going to occur; at this time the SUPE expects new beam pattern data to be issued to it from the MPG. An ISR has also been set up on each AP to awaken their Fast Tasks whenever the SUPE is broadcasting new beam pattern data to the APs. There is an ISR which detects when an individual piece of beam monitoring device data has been received, moving the Milcom Task through its data acquisition cycle.

5. FUTURE

A couple learning experiences have occured during the testing of MPS. MPS has not been able to meet the design goal of operation at 360Hz since there is just too much processing to do: internal error handling, the algorithm processing itself, operating system overhead etcetera. MPS

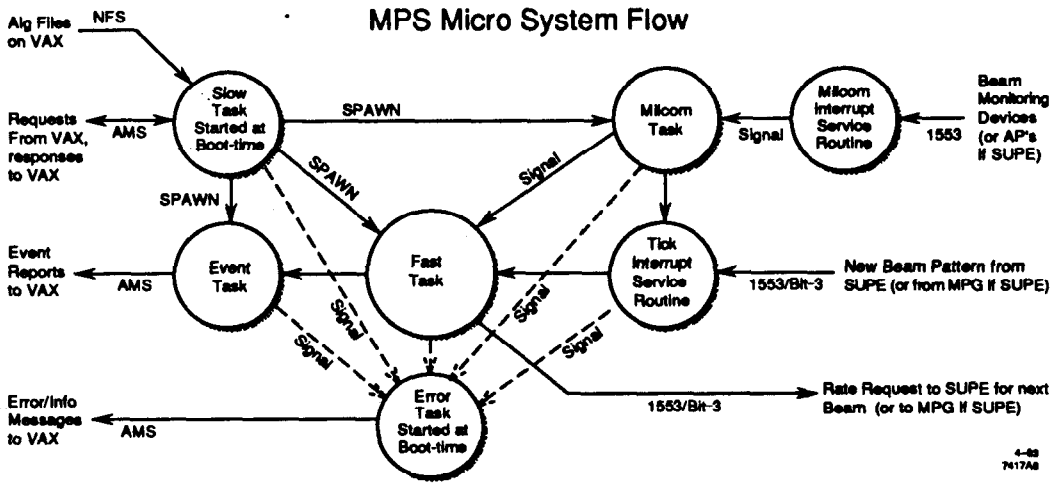

Figure 2. Task dataflow.

Table I. Fast task functional unit timing survey.

8252 μs	total time between beam ticks (120 Hz)
6548 μs	total time waiting for the next beam tick
1672 μs	total time processing one beam tick
228 μs	delay data acquisition until after beam passage
656 μs	SUPE broadcasting new pattern info to APs
72 μs	shifting the pipeline
292 μs	starting a data acquisition cycle
200 μs	milcom 1553 port reads (two ports in this case)
300 μs	milcom task overhead per data acquisition cycle
1500 μs	processing a small algorithm
96 μs	sending the result to the MPG
144 μs	reporting the results to the VAX

will be commissioned to run at 120Hz, although it has been tested at 180Hz. Future design enhancements include the Fast Task executing on its own CPU; however, MPS is presently not required to run at greater than 120Hz. Table I shows the big time consumers in the fast task.

The speed and priority at which the Fast Task must execute caused some multitask management challenges, wherein all other tasks must be prepared to be preempted by the Fast Task even if they aren't ready.

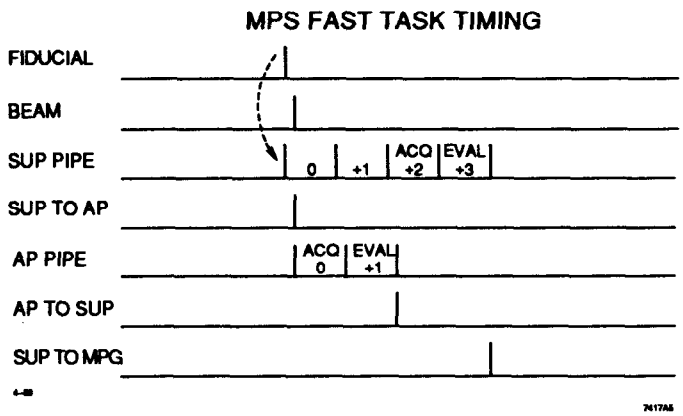

Figure 3. Pipeline.

References

[1] S. Clark, et. al., "Smart Machine Protection System," in Proceedings from the International Conference on Accelerators and Large Experimental Physics Control Systems, Tsukuba, Japan, SLAC–PUB–5688 (1991).

[2] G. White and G. Sherwin, "Machine Protection System Algorithm Compiler and Simulator," these proceedings, SLAC–PUB–6159 (1993).

[3] S. Allison, et. al., "MPS VAX Monitor and Control Software Architecture," these proceedings, SLAC–PUB–6155 (1993).

[4] M. Crane, R. McKenzie, D. Millsom, M. Zelazny "AMS: Area Message Service for SLC," these proceedings, SLAC–PUB–6166 (1993).