MUMIE

A MULTIPLE MICROPROCESSOR ENGINE

PROPOSED FOR

TREATMENT OF DATA FROM HIGH-ENERGY PHYSICS EXPERIMENTS

SUMMARY

A Multiple Microprocessor Engine, MUMIE, has been designed to increase the on-line processing power of the data acquisition system in high energy physics experiments. The system is intended for real-time filtering of raw data from an experiment under the control of a mini-computer.

The engine consists of a set of fast microprocessors operating in parallel each processing the new data from one event. In this way very high effective speed of the system is obtained. E.g. with 10 processors the speed equivalent to a CDC 7600 can be reached. The processors simulate the PDP-11 instruction set, hence the system is fully programmable and very flexible to changing demands.

The modularity of the system makes it possible for the user to choose a configuration that is suitable for his own needs, and it is easy to develop interfaces for different purposes. The testsystem that is being built incorporates an interface to CAMAC.
1. FEATURES OF THE MUMIE

- Modular in computing power and cost. Modules re-usable.

- Short delay specification - realization.
  Fully programmable.
  Flexible to changing requirements.
  Easy and fast system integration.

- Processors emulating PDP-11 instruction set.
  Programmable in Assembler.
  Use PDP-11 development software.
  16 bits fixed point arithmetic.
  4K program memory, 1K data memory.
  Maximum 30 PUs.

- High processing capability through parallelism.
  Overlapped input and output as DMA.
  MUMIE assumes sequential event streams.
  Possibility for event order change.

- Speed with 10 PUs expected equivalent to CDC 7600.
  Speed with 30 PUs expected equivalent to hardware.
  (for point finding in four planes).

- Easily expandable with new modules.
  E.g. fast interface to mini-computer,
  direct interface to mass-storage system,
  PUs emulating a different mini-computer.

- Testsystem in 19" crate with interface to CAMAC

- Estimated price 11.000 + N x 5.300 SFr.
  N = Number of processors; N < 10.
  Material and workshop labour counted.
2. BACKGROUND

In recent years the development of new detectors and sources of high energy particles has led to a dramatic increase in event rates in particle physics experiments. Due to the great difficulties in constructing ideally selective experimental trigger systems, large numbers of events have to be passed through a filter mechanism in order to select only the interesting events for further in-depth analysis.

The use of mini-computers for data acquisition and control in high energy physics experiments has now become standard practice. A mini-computer is generally not sufficiently powerful to perform the filtering mechanism required for event reduction in real time. Thus, the acquisition computer is mainly used to write on a magnetic tape the, now, unfiltered, data from all the triggered events. The filtering is left for off-line analysis which very often requires a large, fast computer. However, the increase in data-taking capabilities has lately been so great that these large computers are becoming more and more saturated.

One solution to this problem is to increase on-line processing power, so that the filtering, or parts of it, can be performed in real-time, thus providing an enriched sample with a higher percentage of interesting events for the off-line analysis. Since many filtering algorithms are rather simple, it has been possible to develop special purpose processors which are entirely hardwired and designed for high speed execution of a specific algorithm. By integrating these processors into the data acquisition system, a first level of event filtering, in real-time, can be achieved.

This solution has certain disadvantages that are due to the fact that the algorithm is hardwired and not easily reprogrammed. There is inevitably a rather long delay between the specification and the realization of the algorithm, since hardware has to be designed, produced and tested. The algorithm must be specified in detail during an early stage of preparation of an experiment, and changes in the requirements can have a very long response time.

A special purpose processor generally is of the type "one of its kind" and designed to fit one algorithm and one experiment. It cannot easily be re-used or adapted to another experiment, since hardware is rather inflexible in this respect. Only parts can be recuperated, and used in new designs. In addition, the integration into the data acquisition system is not trivial, because each processor is a new design. The above suggests that using special purpose processors in experiments is a rather costly method compared to that of re-usable plug-in modules which are generally used in data acquisition systems.
In this paper another method to increase the on-line processing power of the data acquisition system will be described. The recent development of high speed, large scale integrated circuits has made it possible to build fast and inexpensive general purpose micro-processors. However, a fast micro-processor is not fast enough to satisfy the requirements of increased computing power alone, but a set of these micro-processors can be made to operate in parallel and thus obtain a higher overall speed by parallelism. The usual communication and control complications encountered in a system of cooperating processors can be avoided by making the processors completely independent. The processors will execute their own copy of a common filtering program, but they would treat different events. Thus, there is no communication between the processors, and the problems with parallel operation of processors are radically reduced. The system only has to keep track of what each processor is doing, and it must be able to feed and retrieve data from the processors.

Evaluations of filter algorithms already realized in special purpose processors suggest that a set of about 30 fast micro-processors operate as fast as a hardwired processor.

Some advantages of this solution compared to that with special purpose processors are evident. This system is easily programmable and flexible to changes in requirements, hence there is a short delay between specification and realization when the basic system is available. Hardware can be tested with the data acquisition system once and for all. The system contains re-usable parts and is modular in price and processing power. More processors give more power and also cost more.

One disadvantage of this solution is that the hardware in a full-blown system with about 30 processors would probably cost more than the equivalent special purpose processor. However, since the modules are re-usable, the long-term cost of parallel processor systems diminishes all the time. Another disadvantage is that the order of events input to the system is not necessarily maintained at the output, since the processing time can be different for different events.

An important restriction for a parallel processor system is that the result of the filtering operation is delayed with respect to the event input time. Although results come out of the system with a speed that depends on the number of processors used, the delay for each event is always equal to the processing time in one processor. In experiments where all the information from an event is not needed in the filtering process, this supplementary data has to be stored until the result of the filtering is available. Thus, supplementary data from several events need to be stored. This storage can be either in the micro-processors themselves or external to them. However, in experiments with a need for decisions faster than this delay a hardwired processor is the preferred solution.
3. **SYSTEM CONCEPT**

In the figure 3.1 is shown a block diagram of the MUMIE in an experiment configuration. The Data Source, DS, should be seen as the read-out system of an experiment that needs to have its raw data filtered. The Controlling Minicomputer, CM, should be seen as the mini-computer system normally used in the data acquisition system of an experiment.

The MUMIE can, from the external world, be looked upon as a black box with one data input channel and one data output channel. Data flow with different speeds in the two channels. The input flow is a high rate flow of raw data from the experiment, whereas the output flow is a low rate flow of filtered data to the data acquisition system.

Within the MUMIE is a set of micro-processors that all have an identical copy of the filtering program used for the data reduction. Each processor works on the data from a complete event and needs no information from other processors. Thus, the processors can all operate in parallel and are completely independent of each other. Thus, the overall speed at which an event can be filtered depends only on the number of processors in the system, since there is no overhead for interprocessor communications. A maximum of 30 processors is foreseen.

The operation of the MUMIE requires the following types of modules:

- **PU** Processors, maximum 30
- **DI** Data Interface
- **CI** Computer Interface
- **CS** Communication System
- **AC** Allocation Controller

The Data Interface, DI, is, as is seen in the block diagram, the interface with the experiment. Once an event has been triggered, the read-out system will request the DI to establish a data path to a processor in order to send the raw data block from that event for filtering. Upon receiving this request the DI requests the Allocation Controller, AC, for a free processor, i.e. a processor that is not occupied for the moment with processing or transferring results. If there is a free processor, the DI will establish a data path to the selected processor, then it will inform the read-out system and the transfer starts. The transfers are of DMA type, thus the memory in the processor is directly accessed by the CS without intervention from the processor. When the transfer of the block is finished, the processor starts treating the data.
As can be understood from the above description, the AC is the administrative centre of the MUMIE. The AC keeps track of what the processors do, if they process data, if they are free waiting for data, or if they are waiting to send results. From the DI it receives requests for free processors, from the PUs information that processing is finished, and from the Communication Interface, CI, requests for results.

When the mini-computer is able to read results from the MUMIE, it sends a request for a data path to a PU with results. The CI receives this requests and transmits it to the AC. If the AC answers that a processor is ready, the CI establishes a data path with the selected processor memory, then informs the mini-computer, and the transfer starts. Also this transfer is of DMA type between the two memories involved.

The sequence described above shows how data from one event traverse the system. The high throughput of the system is however obtained through parallelism which means that the activities mentioned above can be concurrent. Thus, several events can be processed at the same time as data is transferred to and from the system.

In a system where the data transport times are neglectable compared to the processing time the event input rate to a MUMIE with N processors is increased N times with respect to a one processor system.

In a well balanced system, where all the resources of the MUMIE are fully utilized, one processor would receive raw data, another send result data and the other N-2 process data. The event input rate would in this case be increased N-2 times. This assumes of course that the result reading capacity is not limiting.

From the discussion above it can be seen that the event input rate for a MUMIE with N processors can be increased by a factor varying between N and N-2 compared to a 1 processor system as long as the data transport times are not limiting.
Figure 3.1 Blockdiagram of a MUMIE system

AC = Allocation Controller
CI = Computer Interface
CM = Controlling Minicomputer
CS = Communication System
DI = Data Interface
DS = Data Source
PU = Processor Unit
XI = Extra Interface
4. TESTSYSTEM DESCRIPTION

In order to try out the system concept a test system has been designed. A blockdiagram of this testsystem is shown in Figure 4.1. Comparison with the general blockdiagram in Figure 3.1 shows that the two interfaces to the external would have been replaced with one common interface. This Communication Controller, CC, connects to the CAMAC input-output system normally used in the data acquisition and control system for an experiment.

The Communication System is implemented using a Bus Controller, BC, controlling an Intermodular Communication Structure, ICS, which contains a set of buses and control signals. A two BC system is provided for.

In the first phase of implementation the Allocation Controller, AC, will not be included in the testsystem. The AC functions will be performed by the controlling mini-computer which will thus have to keep track of what the processors do.

The processors in the testsystem are fast micro-processors emulating the PDP-11 series instruction set. Thus, program development and debugging can be done on any standard PDP-11. Some instructions that are not useful for this application, e.g. interrupt handling, are not implemented. These processors are expected to execute data reduction oriented programs 3-4 times faster than a PDP-11/40 with core memory.

Seen from the users side the testsystem looks like four CAMAC stations, each implementing one channel to the CC. The four channels are:

- Status channel
- Control channel
- Data Input channel
- Data Output channel

The CAMAC side of these channels consists of four standard CAMAC modules, NP-170, general purpose register.

The following type of functions can be performed from the mini-computer.

- Loading PU memory
- Reading PU memory
- Loading status into PU
- Reading status from PU

There is a general procedure that always has to be followed whenever any of these functions shall be executed. First of all the
AC = Allocation Controller  
BC = Bus Controller  
CC = Communication Controller  
CM = Controlling Mini  
ICS= Intermodular Communication Structure  
PU = Processor Unit 

Figure 4.1  Blockdiagram of the MUMIE testsystem
control over the interface CC must be gained by sending a special control word via the control channel. A bit in the status channel shows when the mini-computer has control. Next the control information necessary for the function is sent, and then the data words are either sent or read via the data output or input channel.

In the case of a load or a read PU memory operation the data channels would operate in DMA mode. The CC can operate in either Repeat Mode or Stop Mode according to the CAMAC specifications. Normally the Stop Mode would be used since the testsystem is fast enough to communicate at the maximum CAMAC speed.

Changes in important system status bits will generate an interrupt in the CAMAC system, e.g. when a processor has finished processing.

The 4 NP-170 modules each occupy one station in the CAMAC system. The MUMIE testsystem modules, however, are not built in CAMAC standard since CAMAC boards are too small to carry all the components in one module. Especially a high speed processor must be built on one board. Thus, the testsystem modules, out of which some are also modules in the final system, are built on bigger boards and are housed in a self-contained 19" crate that connects to CAMAC via cables.