THE FAST AMSTERDAM MULTIPROCESSOR (FAMP) SYSTEM HARDWARE

L.O. HERTZBERGER, G. KIEFT, B. KISIELEWSKI, L.W. WIGGERS,
NIKHEF-H Amsterdam, The Netherlands*

C. ENGSTER, L. VAN KONINGSVELD,
CERN, Geneve, Switzerland

Abstract:

The architecture of a multiprocessor system is described that will be used for on-line filter and second stage trigger applications. The system is based on the MC68000 microprocessor from Motorola. Emphasis is paid to hardware aspects, in particular the modularity, processor communication and interfacing, whereas the system software and the applications will be described in separate articles.

1.0 Introduction

Present day electronic experiments in High Energy Physics (HEP) produce event rates as high as 100 K/sec whereas the number of electronic channels to be recorded amounts to 10^3 - 10^4 / events.
The complexity of the experiments that have to be performed has increased and consequently the trigger criteria need to be more selective. As a consequence fast trigger systems cannot fully cope with this increasing complexity and the interesting events become contaminated with an overwhelming background.
For the experiment this situation results in an unacceptable increase in dead-time, which can only be improved by adding more trigger stages.

The Fast Amsterdam Multi Processor (FAMP) system [1] that will be described in this paper has been designed to be used as an on-line filter and a second stage trigger system. Using a subset of the experimental data the system can be programmed to perform computations like:

- track reconstruction,
- momentum determination,
- invariant mass calculation.

The system works in real time which implies that the results of the computations can be used to filter the dataflow to the on-line computer.

In this paper special emphasis is paid to the FAMP hardware. The software [2] and the applications, i.e. a momentum trigger [3] in the experiment NA11 at the SPS and a topology trigger [4] in experiment UA1 at the $p\bar{p}$ collider will be described in separate papers.

2.0 FAMP design philosophy

Until now the introduction of commercially available microprocessors (μP's) in second stage triggers has been hampered by the following factors:

- low execution speed of available μP's,
- limited, only eight bit, data bus,
- restricted addressing capabilities.

In the last two years this situation has been drastically changed. Fast processors with a sixteen bit wide data bus and with elaborate instruction sets, like the Zilog Z8000 and the Motorola MC68000 have become available. However, although the speed has been increased considerably, it is still a limiting factor for second stage trigger applications, which can only be relaxed by making parallel systems.

Multiprocessor systems have the potential advantage of offering a large redundancy in computing power, This can, however, easily be spoiled by deficiencies in the system design. Limitation in the speed and the number of available buses or a too complicated arbitration scheme for the allocation of buses to memories and processors can reduce the overall system performance. In the design of the FAMP system we have tried to implement a pipe-lined structure taking into account the type of calculation that has to be performed.

In most of the event selection criteria some form of track reconstruction plays an important role. Track reconstruction can easily be split up into a number of sub-tasks that can run concurrently.
This situation has led to a system philosophy where complete tasks are performed independently on separate processing elements which either run in parallel or at different hierarchical levels.
The data can be divided into parts which are offered to those cells where the information is required.
The organisation of such a network of processors will depend on the particular experimental application. The following ideas have defined the design criteria for our multiprocessor system:

- each processing element should be self-contained having its own data- and program memory as well as its own I/O channels. It will be called a processing cell,
- the exchange of data between one processing cell and the other should be fast and must never interact with the computational tasks in progress in the other cell. This is even more important for the dataflow coming from the experiment entering a processing cell,
- to take full advantage of the potential redundancy in processing power of the multiprocessing system it should be modular making the addition of processing cells and memory simple.

3.0 Detailed description of the FAMP system

3.1 System overview

The FAMP hardware is designed such that different levels of processing can be distinguished. The extension with more processing cells within one level is called horizontal extension.

In the FAMP system there is no degradation in system performance due to bus bandwidth limitations, as is often the case for multiprocessor systems with all processors connected to a common bus. This is because the coupling of two processing cells, called vertical system expansion, is realized by applying two fully decoupled bus systems. This vertical system expansion is realized by the application of dual port memories (see fig. 1). The processing cells connected to such a memory, exchange information by writing in or reading from the memory. In this way it is guaranteed that interprocessor communication does not interfere with computational tasks in progress in the various cells. Examples of such networks are illustrated in fig. 2a and fig. 2b. The last figure shows that the hardware of the system allows multiple trees with a processing cell at the root of each tree. The processing cell at the root of a tree will be called the supervisor. With the two basic elements i.e. the processing cell and the dual port memory, a flexible system is realized that can be adapted and optimized for the particular requirements of any experimental application.

The interconnection of processing cells and dual port memories has certain implications at bus level. Therefore two completely separated busses are provided in the system. Both are derived from the MC68000 bus protocol [5]. The internal system bus can be used to extend a processing cell with resources like memories and interfaces (see fig. 3). The external system bus is used to connect the processing cell with one of the input ports of the dual port memory. The other memory port is connected to the internal system bus of the next processing cell (see fig. 4). Up to 8 dual port memories and associated processors can be hooked up to the external bus.

An example of a typical FAMP configuration is given in fig. 5. In this example data enters via the dual port data memories and is processed by a number of processing cells concurrently operating at one horizontal level. These cells independently exchange results with a processor at the next hierarchical level. This processor collects the data from the various cells and transmits them to the data taking computer.

A processing cell always consists of a CPU module and depending on the application can be extended with the following modules:

- 16K Byte Dual Port Memory,
- 128K Byte dynamic RAM,
- 64K Byte EPROM,
- FAMP REMUS Output buffer,
- Dual Port Data Memory,
- Control Bus Driver.

These units will be discussed in more detail in the next section.

The system is built using CAMAC mechanics while not applying the CAMAC bus protocol. Only the CAMAC power supply lines are used, the other lines are replaced by a special backplane carrying the signals of the internal system bus.

3.2 CPU-module

The CPU-module (fig. 6) reflects in its structure the same modularity as present in the system. It consists of two boards, that communicate via the internal system bus. One board contains the CPU and necessary support chips, whereas the other mainly contains memory, the function of which will be discussed later on.

For the CPU we have chosen the Motorola MC68000 processor in view of its advanced architecture and speed. An extensive description of this micro-processor is given elsewhere [5], however, an overview is given of those properties that are relevant for the application discussed here.

The MC68000 has a sixteen bit wide data bus and in this respect is comparable with most minicomputers that have been used in HEP experiments in the past 5-10 years. The cycle time of 125 ns provides an average buscycle time of 500 ns equal to the minimum instruction time. This makes the MC68000 an order of magnitude faster than the previous generation of microprocessors, like the MC6800.

The instruction set supports operations on bytes (8 bits), words (16 bits) and long words (32 bits). The 24 address lines give the possibility to address 16 MByte of memory, a feature which is very useful when handling large look-up tables.

The processor can run in the user state or in the supervisor state, which is indicated by processor function lines. Depending on the state of the processor two different stack-pointers can be accessed. This feature allows for protection of the stacks used by the operating system and those handled by the user programs.

The MC68000 has three interrupt lines that make a distinction between eight levels of priority possible. Except for the highest interrupt level (level 7), all levels can be masked.

The MC68000 has an elaborate exception processing facility, which is being used not only during interrupt handling but also comes into action when an illegal situation is detected such as:

- bus error,
- illegal or non implemented instructions.

The instruction set supports 14 addressing modes and, moreover, has instructions like "read modify write" specially designed to run in a multiprocessor environment.

All features discussed show that the MC68000 is powerful especially if supported by a modern operating system.

The local memory present in the CPU module has a capacity of up to 16K Byte. It is built out of static (150 ns access time) RAM. This local memory can be used to store the trigger programs and the data acquisition routines.

Up to 16K Byte EPROM can be located on the CPU module which can be used to store a part of the operating system. Moreover, jumpers on the board offer the possibility to exchange EPROM for RAM so that the RAM capacity can be enlarged to a maximum of 22K Byte. In this particular situation a minimum of 2K Byte EPROM is left.

The location of EPROM as well as RAM in the MC68000 memory map is defined by programming the address selecting PROMS. The memory map as currently used in the system is illustrated in fig. 7.

The MC68000 uses a handshake bus protocol for communication. To prevent bus hang-ups, a time-out signal is generated in if no bus activity is observed during 1 ms after the bus active signal has become valid.

A Programmable Interface Adapter (PIA) [6] with signal conversion to NIM levels allows the user to communicate decisions to the experiment and vice versa.
We have chosen the following configuration:

- 3 input lines to be polled by the software,
- 2 input lines providing auto vectored interrupts, or to be polled by software,
- 3 output lines under software control.

The priority of the autovector interrupts can be determined using on-board switches.

Additional interrupt facilities can be accessed via the Control Bus Driver (see section 3.5). This unit offers the possibility to set three more interrupt lines and eight associated vectors. The hardware providing these functions is present on the CPU board using the so-called system control bus. Another feature of this bus is that it allows control of memory protection on board as well as in the Dual Port Memory units.

Two Asynchronous Communication Interface Adapters (ACIA) [7] provide RS 232 compatible interfacing for slow peripherals. These interfaces can be used for connecting terminals for system monitoring. Another possibility is to use one of these interfaces for down line loading of programs.

3.3 Dual Port Memory

The features making the dual port memory attractive for use as an interface between processing cells are:

- two fully separated MC68000 compatible ports,
- independent address decoding logic for both ports,
- arbitration logic preventing clashes at simultaneous access requests,
- memory protection,
- a "broadcast write" mode of operation,
- a "wired-or read" mode of operation.

Of the two independent I/O ports providing the access to the memory, one is realized as a back-panel connector that can be interfaced with the internal system bus, the other is realized as a front panel connector that can be connected to the 64 lines flat cable of the external system bus.

The independent address decoding PROMs available for both ports allow for locating the memory at different addresses in the memory maps of the associated processing cells. Such an addressing scheme offers the possibility to chain a number of memory banks connected to various processing cells in one level into one block of memory in the memory map of the processing cell at the next level (see fig. 8).

The arbitration logic handles multiple access requests from both ports. The design is such that a certain access priority can be granted to one of the ports.

The unit can comprise of up to a maximum of 16K Byte static (150 ns access time) RAM which is segmented into 2K Byte. Also memory protection can be adjusted in 2K Byte steps.

In a configuration as illustrated in fig. 4 it may be necessary for the supervisor to send the same data to all the processing cells. Therefore it is useful to have the possibility of writing data into all dual port memories in parallel, often called broadcasting. The supervisor can access this facility by using specific addresses. With the "wired- or read" option the supervisor can simultaneously test the status of processing cells connected via the various dual port memories. This facility saves time-consuming polling of successive status words.

3.4 Memory extension

Two different memory extension units will be available in the system:

- 64K Byte EPROM module,
- 128K Byte RAM module.

The application of both modules will be different. The EPROM module will be used when programs and data have to remain valid after a power failure, e.g. a bootstrap loader or a monitor program. The RAM module is meant as extension of the program memory space. This module is built out of 16K Byte 150 ns access time dynamic RAM chips. The unit has on-board refreshment logic. Parity control is used for error detection.

3.5 System interfacing

All the input of data into the system as well as the output of data from the system is buffered. For this purpose and for interrupt control a number of units are developed that will be described in this section, i.e.

- Dual Port Data Memory (DPDM),
- FAMP REMUS [8] Output buffer (FROB),
- Control Bus Driver (CBD).

3.5.1 Dual Port Data Memory

The data from various detectors like MWPC's and driftchambers are stored in memories that are interfaced with the internal system bus. These modules resemble the dual port memories in many aspects. However, the external system bus is different and the control of this bus is not performed by a CPU module, but by external - experiment dependent - controller. The dual port data memory will also support extra functions such as reordering of data and the generation of data-pointer tables.

3.5.2 FAMP REMUS Output buffer

The FAMP REMUS Output buffer (FROB) is used for the transfer of data and results to a host computer system. In most applications the unit will be connected to the internal system bus of the supervisor. The CPU module writes the data into the onboard 8K Byte memory of the FROB. Upon request of the data taking computer the data are then transferred in a block transfer mode.

Because the supervisor can generate end of crate markers, using a special function, it has full control over the data format written in the FROB. In addition a special option allows the functional behaviour of the output buffer to be tested.

3.5.3 Control Bus Driver

Each CPU Module has a number of NIM lines for communication with the central trigger of the experiment. Often a centralized interrupt facility is also wanted which here is provided by the Control Bus Driver (CBD). With this unit connected to its internal system bus the supervisor CPU module can drive three interrupt lines and set eight interrupt vectors. A number of NIM inputs also allows these lines to be driven by signals generated by the experiment. The associated vectors must then be generated by the supervisor upon system initialization.

4.0 Concluding remarks

With the increasing complexity of experimentation in High Energy Physics selective trigger systems become more and more important. The application of commercially available microprocessors in these systems offer the possibility to build economic systems that can benefit fully from new developments taking place. In this respect a high degree of modularity is very important because it offers the possibility of replacing only those units that are ageing.

The technique of buffering the data transport everywhere in the system not only makes interfacing simple, but also allows data flow processing with a minimum of synchronisation tasks. In this manner, the time consuming software synchronisation overhead can be kept to a minimum, leaving the maximum time available for data processing.

Acknowledgements

We thank the NIKHEF electronic workshop led by G. De Vries and in particular A. Rijllart and W. Schüller, for their help in the hardware development. We appreciate the

support of the Aachen group participating in the UA1 experiment and the Amsterdam NIKHEF-H group participating in the NA11 experiment. In particular we would like to mention K. Eggert and W. Hoogland, for their support.

* This work is part of a joint program of FOM and ZWO.

References

1) L.O. Hertzberger et al., Proc. EPS Conf. on Computing in High Energy and Nuclear Physics, Bologna 1980, Comp. Phys. Comm. 22 (1981) nrs. 2, 3, p.253-260.

2) D. Gosman et al., The FAMP operating system, these proceedings.

3) L.W. Wiggers, et al., Applications of the FAMP system in the NA11 experiment at the SPS, these proceedings.

4) K. Eggert et al., Fast On-line Track finding for the Muon Trigger in the UA1 Experiment, proceedings thereof.

5) Motorola, MC68000 user's manual, MC68000 UM (AD2), 1980.

6) Motorola, MC6800 application manual (1975) 3-8.

7) Motorola, MC68000 application manual (1975) 3-21.

8) P.J. Pointing, CERN-E8/80-01.

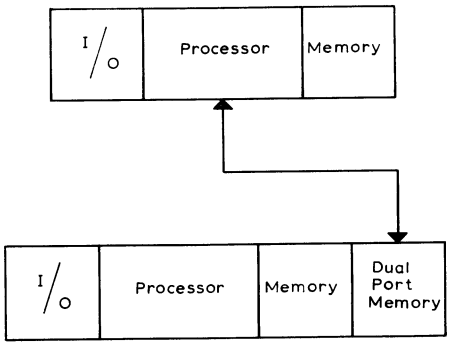


Fig. 1 Processing cell coupling

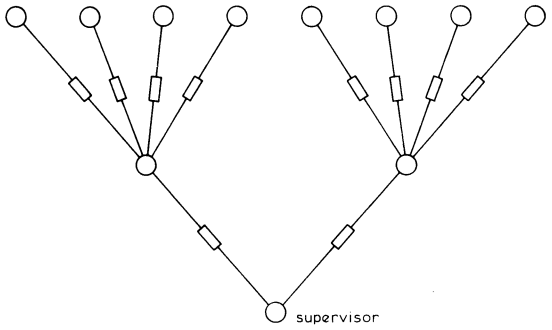
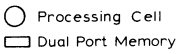
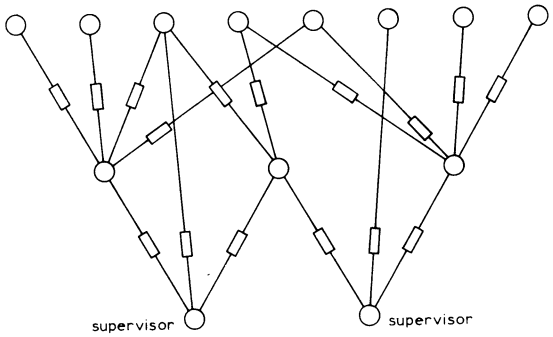


Fig. 2a Vertical system expansion





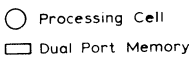


Fig. 2b Vertical system expansion with two supervisors

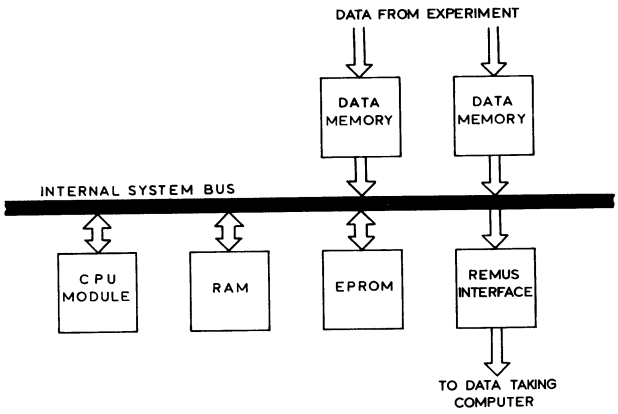


Fig. 3 Processing cell

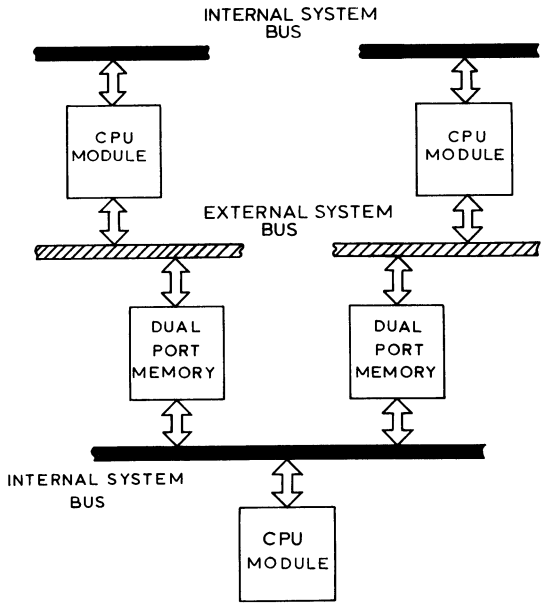


Fig. 4 Bus connections

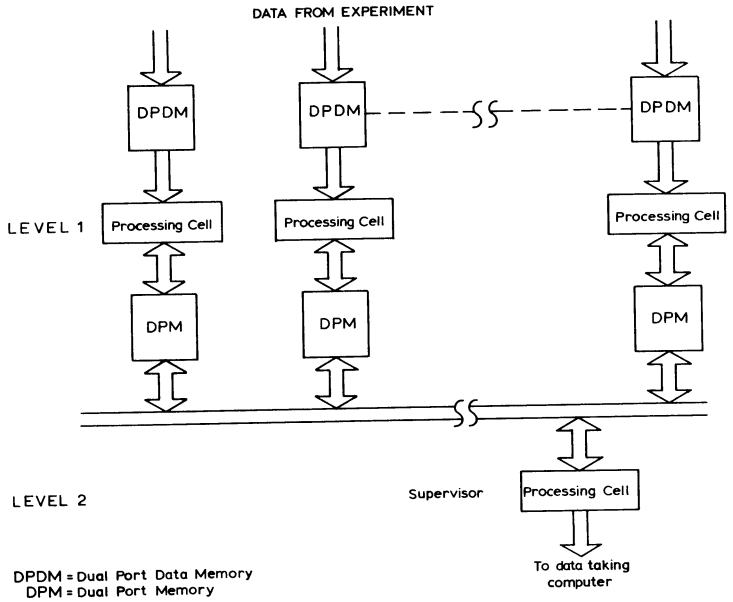


Fig. 5 Two level FAMP system

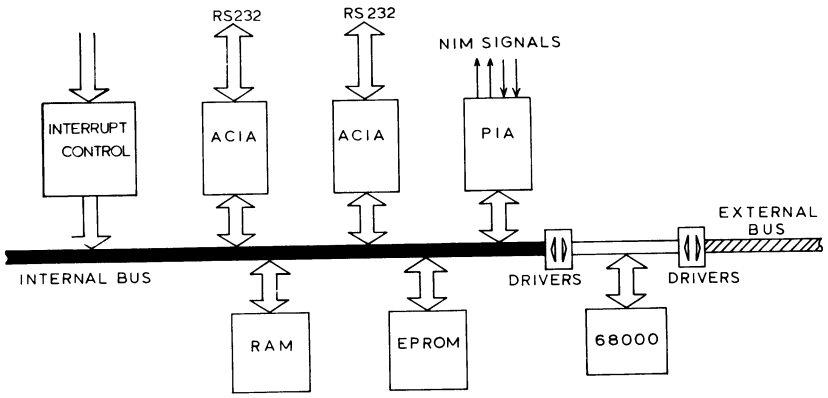


Fig. 6 CPU module

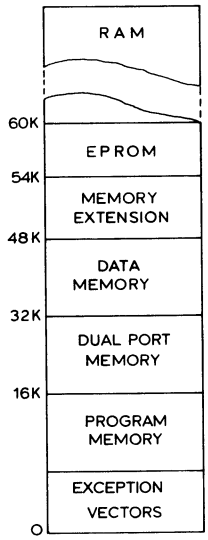


Fig. 7 System memory map

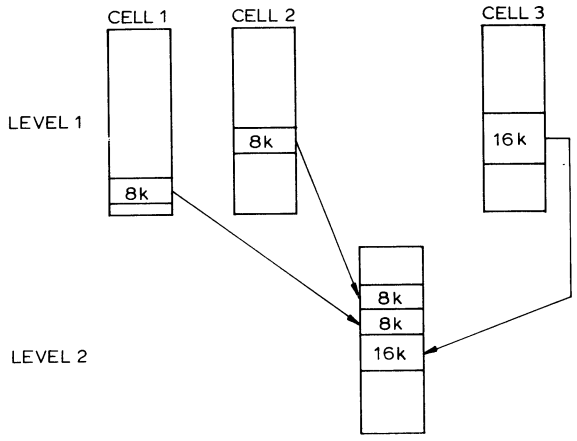


Fig. 8 Memory map chaining