STATUS OF EVENT SELECTION PROCESSORS AT CERN

C. Verkerk
European Organisation for Nuclear Research
Geneva-Switzerland

Summary

A brief review is given of the uses in event selection of different processors, following the same classification scheme as has been used earlier. The developments which took place at CERN during the last year are described. The progress made with three processor systems, namely MICE, 168/E and FAMP will be emphasized.

Introduction

A year ago, a wealth of information was presented at the Topical Conference on the Use of Microprocessors in High-Energy Physics Experiments[1]. The coverage of this field was so complete, that it will be difficult for some time to come to discover an aspect of the use of (micro) processors in particle physics which was not mentioned. If we try to measure progress against what had been achieved already a year ago, we must admit that apparently - as far as CERN is concerned - no spectacular new things have seen the light. In the absence of real innovations however, more solid experience has been gained with a number of processors for event selection. A large part of the progress has been made in the new Underground experiments and the experience of one of them is the subject of a separate presentation at this conference[2]. The present paper will briefly review what has happened in the field, outside the UA1 experiment. Three processing systems will receive most of our attention MICE, 168/E and FAMP. A year ago these had just begun to find their way into experiments.

This brief review will be structured according to a classification scheme[3] adopted earlier and based on the principal use of the processor : pre-processing, event-selection, monitoring and control, tests. Inside each class a subdivision is made according to the implementation, which - in some complex way - is related to speed.

Pre-processing and data-acquisition

In the course of the year, it was decided to provide a limited support for CAB[4], the Camac Booster developed at Ecole Polytechnique, Paris. This is an Amd 2900 based microprogrammable processor, packaged as an auxiliary crate controller. In an earlier version it was very successfully used in the real-time analysis of a small angle scattering experiment at the PS[5]. Due to its short micro-word (24 bits), microprograms for this processor have a familiar flavour of normal assembly language programs. Cross-software for CAB is available on CDC Cyber machines. The software which has been developed at CERN aims at easy integration of CAB in the standard data-acquisition systems.

A typical example of the use of CAB is given by experiment NA3. Three Camac branches are controlled by one CAB each. Approximately 60 events per burst are treated; the CABs perform the read-out and the compaction of ADC data. Individual pedestals are subtracted and whenever a signal above threshold is detected, the adjacent cells are inspected and registered even if their signals are below threshold. The CABs are also used for calibration. One of the processors reads out 1500 words per event, the others approximately 300 each. The treatment takes 3 ms maximum. The events are dumped into an external memory from where they are read back and recorded during the time between bursts.

Event-Selection

Hardwired Processors

It had already been remarked that a large difference exists between Europe and the US in the application of hardwired processors. In Europe hardwired processors have been developed for a few experiments in an ad-hoc manner. No serious attempt was made to define a set of modules from which the physicist could build himself complex event-selection processors.

MBNIM[6] satisfies a number of the requirements and is well suited for fast decision making with some pre-processing, but it lacks so far the more complex modules needed to perform algorithmic processes for track finding or vertex reconstruction. A memory for look-up purposes and an arithmetic unit are part of the module set, but do-loop indexers and other control-of-flow modules do not exist yet. We are still far away from having an ECL-Camac or a Nevis modular system at CERN and there seems to be no great urge either to import those systems.

Microprogrammable Processors

8 ESOPs continue to be used in 4 different experiments, at CERN[7]. An additional processor is used in an experiment at Saclay. The preparation of the programs for these uses has shown up a few shortcomings of ESOP which make writing code rather difficult and introduce unnecessary overheads in execution. An improved design, called XOP[8], is being worked on at present. Helped by the experience, better ways of shuffling data - or better avoiding to move them around - have been investigated. The new design has register files and allows for nesting of loops and subroutines. The microcode memory will be distributed over the modules. The important advantage is that modularity can be easier achieved, but loading of programs becomes more complicated. For this and other reasons an M68000 control processor is foreseen. The programs will need to be written entirely in microcode, as was the case for ESOP. In spite of the high speed of XOP, this may limit its future use. The building of a prototype should start this year.

Emulators : MICE

Before we give some examples of the present use of MICE[9], we recall its more important characteristics.

MICE was designed for real-time use, for those cases where both speed and easy programming are important. The machine emulates the PDP11 instruction set and due to its implementation in ECL logic reaches three times the speed of the fastest processors in the PDP11 family : the 11/45 and 11/70. MICE is a true emulator : PDP11 machine code is executed by a microprogrammed interpreter. Programs can therefore be written in any language for which a compiler producing PDP11 code exists. Programs written in assembler, PL11 and Fortran run in fact on MICE without difficulty and always much faster than on a PDP11, except when many

byte-oriented instructions are used.

To preserve this high speed operation, the capabilities of MICE have been deliberately restricted in some other respects. The maximum memory size of MICE is therefore limited to 28 Kwords. As programs for event selection must be fast, thus short and simple, this is not felt to be a serious restriction. Fixed point multiplication is done by hardware in 3 cycles (315 ns), but division is implemented in microcode. A floating-point processor was not foreseen originally, again because we felt it was not really needed in event-selection applications. Other applications may however profit from a user microprogrammable floating-point unit so that, for instance, a FFT algorithm could be run at maximum speed. Such a unit is being designed. MICE has a simplified Unibus, so that simple peripherals can be attached without difficulty. A Camac interface is provided for program loading into MICE, for reading of results and for debugging.

MICE has a 1 K writeable control store, 5/8 of which is used for the emulation of the PDP11. A unique feature of the machine is its user micro-programmability. User-written microprograms can be invoked from within PDP11 code, in a way similar to a subroutine call. Depending on the program, an extra factor of 2-5 in speed can be gained by microcoding. On MICE the microcoding can be done for selected parts of the program only. Frequently 90 % or more of a program's execution time is spent in 10 % or less of the code. Such a small piece of code can then be replaced by microcode and the overall speed increased by a factor 2 or more, with a minimal effort.

It is not surprising that these characteristics of MICE were attractive to a number of experiments. At present MICE has been in use in three experiments and three others have definitely decided to use it.

The following three examples of the use of MICE highlight one user-aspect each.

MICE in Computer-Aided Tomography. In this application[10] MICE collects the data for every positron annihilation and reconstructs the line of flight of the two photons. The intersection points of this line with twelve planes is calculated and 2-dimensional histograms built up in a number of Camac modules. These histograms will undergo a further treatment off-line (e.g. FFT and its inverse) to produce the final tomographs.

The program running in MICE was written in PL11 and was able to handle 2500 events/second. The inner loop of the program was then recoded into 20 microinstructions. This gave an improvement of a factor 2 in overall speed. The rate of events that can now be handled in real-time (5000 s^{-1}) should be compared with the rate (6000 s^{-1}) at which histogramming can be done off-line on an IBM 370/168. The rate obtained matches the limit of both the Camac hardware and the processing in MICE.

The microcode was designed, written and debugged in a couple of days by an expert of MICE. It is highly optimized in the sense that the possibilities of pipelining and parallelism of MICE are fully exploited. This example shows that with a modest effort considerable improvements in speed can be obtained, even if a slow I/O system imposes constraints.

MICE in a track selection application. The WA1 neutrino experiment provides another example of the use of MICE. Here it is used to select cosmic ray muons which are closer than 250 mrad to the horizontal direction. These muons, collected in the time between neutrino bursts are used for calibration. MICE reads the data needed via a ROMULUS branch, makes the necessary checks on wordcounts, etc. and suppresses the zero data words, converts the wire number into a space coordinate and stores these numbers in an array. All this is done on the fly, by a program written in PDP11 assembly language. As ROMULUS delivers one word every 1.5 µs and MICE takes only 630 ns to reject a zero data word, there is plenty of time left for treating the non-zero data (≃ 5 %). When the read-out is finished - after 1.6 ms - the analysis program written in Fortran finds the horizontal track, or rejects the event. The overall event rate is limited by the time - 50 ms - to read an accepted event into the NORD 10 computer. The processing time in MICE is negligible : it does not exceed 400-500 µs on average. 40 % of all events are rejected almost immediately by applying simple criteria. 7.5 % of the total number of events are accepted after a modest amount of work. One third of the events are sufficiently complicated that more work has to be done to detect 2.5 % of acceptable tracks amongst them.

A peculiar effect caused by the FIFO-buffers resulted in a decrease (from ≃ 100 to 64) of the total number of events read during a 6 second period, but in the critical regions of the detector ten times more tracks are now found than before.

The example shows that MICE can be usefully employed with a modest programming effort, ≃ 350 lines of assembly code and ≃ 250 lines of Fortran.

MICE in a SC experiment. MICE was used during one week of data taking in an experiment with a ^{12}C beam. MICE simply replaced a PDP11/04, did the Camac read-out via its Unibus extension and sorted the events into five classes. Full buffers were transferred to a PDP11/34. An existing stand-alone program was used. It needed adaptation to replace the DMA transfers from Camac by programmed transfers. The total effort spent was 2 programmer days. The data taking rate was improved by a factor 4.

Emulators : 168/E

At present the only on-line application at CERN of the 168/E is in the central detector of UA1. In December a single processor was used to test the algorithms. The results will be given by S. Cittolin[2]. As far as the hardware is concerned the present plans are to have 2 processors installed end April. The CPUs are built at CERN, the memory boards with increased capacity for data will come from Saclay. The event size is such that a data memory of 128 Kwords is needed. A smart Camac module with a programmable sequencer will read out the 5 Remus branches in parallel at 0.5 µs/word, using the S1 pulse only. The data is sorted into the appropriate blocks and sent to one of the 168/Es for track processing. This arrangement will also require new interfaces to the 168/Es to adapt to an improved version of the data bus.

This application of 168/Es exploits an old idea in event multiprocessing : a complete event is treated by one processor and the next event is sent to another processor. The present plans are for two 168/Es but presumably later extensions are foreseen. It will be interesting to see how such a multiprocessing scheme behaves under real-time conditions.

Off-line applications of 168/E. Although not really the subject of this talk, it is interesting to describe briefly the present status of off-line processing with 168/Es. At end 1981 2.2×10^7 events had been processed, which corresponds to 2000 hours of 370/168 CPU-

times. The machines are now grouped in an off-line pool, which will eventually consist of 7 processors : 2 systems with 3 machines and one system with 1 machine. A system includes a PDP11 and the MOSTEK memories for overlaying. The two large systems are intended for production, the smaller for tests and software development. Four user groups use the facility at present : EMC, SFM, R807 and Asterix. Full track reconstruction for SFM may need upgrading of some of the machines to 64K, to make space for the magnetic field map.

3081/E. In May 1981 plans were presented for an improved version of the 168/E[11], now baptized 3081/E. In principle an agreement was reached with SLAC to collaborate in this development. The project should get off the ground after this spring.

Multi-processor Systems

Systems built from a number of identical and cheap processors should in principle be capable of very high throughput, if the intercommunication and I/O problems can be satisfactorily solved. Many high-energy physics experiments do use some sort of a distributed processor system, but most of them have grown in an ad-hoc manner by adding something to an already existing configuration. Rather few attempts have been made to design a modular multiprocessor system which, with a modest effort, can be adapted to a variety of real-world situations.

Computer scientists recognize that multiprocessing still presents many problems for which no general solution has yet been found. It is then probably wise to adhere to a guiding principle : Simplicity. When we manage to keep things simple, particle physics may profit from the improved price/performance ratios of the modern 16-bit microprocessors. Systems which do not make conscious use of the fact that data is structured in events - or worse, which are upset by it - will have little chance of success in experiments. Simplicity also means that a multiprocessor system be configured once and for all together with the data-acquisition system and that no attempts be made to implement concepts such as dynamic reconfiguration or dynamic task allocation. They only add complications and overheads in our environment. These remarks do not contradict the fact that multiprocessor configurations have emerged rather naturally and without too much difficulty in those cases where the tasks were restricted to data-acquisition and pre-processing. Examples abound in these applications.

The scene changes however when we consider event-selection. Obviously the final decision can only be taken by one processor alone. For colliding beam experiments, if the total task is too large to be performed on a single processor in the time available between successive triggers, only two possibilities are left for implementation : a "hierarchical" system or a "collegial" system. The latter is based on a mutual agreement "I'll see this event through from beginning to end; you others take care of the events that occur while I am busy". We saw an example in UA1.

The FAMP[12] system, developed in Amsterdam, and using Motorola M68000s, is a hierarchical system. Slave processors perform subtasks and report their results to a supervisor via messages deposited in dual-port memories. The supervisor can at any moment take a decision and stop further processing by the slaves with an interrupt.

In experiment NA11 a FAMP system of 1 supervisor and 2 slaves is about ready to operate. For UA1 the implementation of a second level decision process for the muon chambers is well advanced. The final system will consist of 6 slave processors and one supervisor. Data from drift tubes will be used, truncated to a limited precision. Each slave processor will search for muon tracks in a part of the detector and for one projection only. The supervisor will take a decision on the basis of the results for two projections. The programs running in the slave processors are written in assembly language and occupy at present 2 K. They make extensive use of large look-up tables for the definition of cones in which to look for tracks and for the transformation of drift times into bit patterns which can be ANDed to detect a straight track. For the next run, the decision time is expected to be a few milliseconds. This time should be considerably reduced when more a-priori information will become available from the first level trigger.

Conclusion

In this review, the examples of the use of event selection processors at CERN came mainly from fixed target experiments. What then is the relevance for colliding beam experiments ?

In the absence of a burst structure, events cannot be buffered for long periods and the rates that can be handled are directly proportional to the speed of the processing system (including data acquisition). Processing speed is thus a very important factor. Very few physicists seem however to be ready to sacrifice everything else for the sake of speed. So easy programming, easy interfacing and easy adaptation to changing experimental conditions are equally important. The first point, easy programming, will prevail when the events become very complex at higher energies. The availability of good information from the first level trigger will then be a necessity. Much time can be saved if a program knows where to search for tracks.

When we require good programming capability and speed together, the choice of processors narrows down considerably. In my opinion, of the systems mentioned in this review only three then remain : MICE, 168/E and FAMP. The first two seem rather expensive for use in large quantities but they are very well suited when they can handle the job alone. For multiprocessor configurations it is obviously much more attractive to use the relatively cheap microprocessors. But the number of processors that can be made to work together constructively is limited, either by overheads or by the impossibility to divide the job into independent tasks.

For a simple experiment my preference still goes to a single processor system for event selection, but then, are there still simple experiments ?

Acknowledgements

I would like to thank my colleagues at CERN who so willingly gave me the information I needed to update my knowledge. In particular I would like to thank the following persons : C. Bizeau, J. Bourotte, S. Cittolin, L.O. Hertzberger, D. Holthuizen, D. Jacobs, M.F. Letheren, D. Townsend, H. Verweij and H. Wahl. All errors and omissions are the author's responsibility.

References

1. Proceedings of the Topical Conference on the Application of Microprocessors to High-Energy Physics Experiments, Geneva, May 1981, CERN 81-07. (hereafter to be referred to as TCAMHEP, CERN 81-07)

2. S. Cittolin, these proceedings.
3. C. Verkerk in TCAMHEP, CERN 81-07, p. 395.
4. E. Barrelet, R. Marbot, P. Matricon : "Camac Booster", (CAB system), in H. Meyer (ed.), Proc. Real Time Data Handling and Process Control, Berlin, October 79.
5. E. Barrelet et al., Phys. Letters 94 B (1980) 541.
6. F. Bourgeois, Modular trigger logic techniques at the CERN Omega Spectrometer, IEEE Trans. Nucl. Sci NS-27, 594 (1980).
7. G. Lütjens, in TCAMHEP, CERN 81-07, p. 236.
8. T. Lingjaerde, in TCAMHEP, CERN 81-07, p. 454.
9. J. Anthonioz-Blanc et al. in TCAMHEP, CERN 81-07, p. 266.
10. A. Jeavons et al., in TCAMHEP, CERN 81-07, p. 276.
11. D. Lord et al., in TCAMHEP, CERN 81-07, p. 341.
12. L.O. Hertzberger et al., in TCAMHEP, CERN 81-07, p. 70, also p. 83.