EUROPEAN ORGANIZATION FOR NUCLEAR RESEARCH

CERN/ECP 92–7
19 August 1992

EVALUATING PARALLEL ARCHITECTURES FOR TWO REAL-TIME APPLICATIONS WITH 100 kHz REPETITION RATE

J. Badier[1], R.K. Bock[2], Ph. Busson[1], S. Centro[3], C. Charlot[1], E.W. Davis[4], E. Denes[5], A. Gheorghe[6], F. Klefenz[7], W. Krischer[2], I. Legrand[6], W. Lourens[8], P. Malecki[9], R. Männer[7], Z. Natkaniec[2], P. Ni[4], K.H. Noffz[7], G. Odor[5], D. Pascoli[3], R. Zoz[7], A. Sobala[9], A. Taal[8], N. Tchamov[2], A. Thielmann[10], J. Vermeulen[11] and G. Vesztergombi[5]

1 Ecole Polytechnique, Palaiseau, France
2 CERN, Geneva, Switzerland
3 Dip. di Fisica and INFN Padova, Italy
4 North Carolina State University, Raleigh, USA
5 Central Research Institute for Physics, Budapest, Hungary
6 Institute of Atomic Physics and Polytecnic Institute, Bucharest, Romania
7 Inst. of Computer Science V, University of Mannheim, and Interdisciplinary Center for Scientific Computing, Heidelberg, Germany
8 Utrecht University, The Netherlands
9 Institute of Nuclear Physics, Cracow, Poland
10 GEVA Ltd, Warsaw, Poland
11 NIKHEF-H, Amsterdam, The Netherlands

ABSTRACT

In the context of Research and Development (R&D) activities for future hadron colliders, competitive implementations of real-time algorithms for feature extraction have been made on various forms of commercial pipelined and parallel architectures. The algorithms used for benchmarking serve for decision making and are of relative complexity; they are required to run with a repetition rate of 100 kHz on data sets of kilobyte size. Results are reported and discussed in detail. Among the commercially available architectures, pipelined image processing systems can compete with custom-designed architectures. General-purpose processors with systolic mesh connectivity can also be used. Massively parallel systems of the SIMD type are less suitable in the presently marketed form.

Submitted to IEEE Transactions on Nuclear Science

Mac/Word/8024F/RKB/sj

1. INTRODUCTION

The EAST Collaboration [1] (RD-11) has made it one of the first major goals to explore possible commercial implementations of second-level "intelligent" triggering architectures. Initially, we have restricted our investigations to the "feature extraction" part, i.e. to the task of converting raw front-end-formatted data to quantities (features) meaningful from the physics point of view. Features are interesting in restricted areas of individual subdetectors only. Such "Regions of Interest" (RoI) are indicated by a preceding first-level trigger, operating at the collider's bunch crossing frequency of ~ 60 MHz. Feature extraction can in a natural way make use of multiple devices operating in parallel, on different RoIs and for different subdetectors. The physics features extracted from fine-grain local data in multiple detector windows, are subsequently combined into global decisions, by correlating data from different RoIs and subdetectors. The global decision is not directly addressed in this note.

The assumed trigger context, borne out by physics simulations, and universally assumed also in all trigger and data acquisition discussions around the collider projects LHC and SSC, is the following: after a reduction of the initial event rate by a first-level trigger, relying on calorimeter windows and muon identification, the event passing rate is of the order of ~ 100 kHz. At this rate, reduction "algorithms" of some complexity will be required to reduce event rates further. They are based on data from the first-level trigger and on additional local detector data that may be extracted from some buffering device or intercepted "on the fly" as data pass from one system part to the next (e.g. as they are pushed from geographically spread front-end pipelines to data concentrators). We assume for the architectures studied that external hardware units, i.e. local buffers and the first-level trigger, send control signals and local data into the architecture in question: the first request feature extraction algorithms to start, the second come after a defined delay. This is usually described as a "push" architecture, as opposed to a readout in which the device executing the algorithm also manages the readout ("pull" architecture).

We have explored, in particular, the possibilities of introducing commercially available local solutions at this stage. Other ideas to solve the same problem have been discussed: in the technical proposal of the SDC detector [2], the implementation of second-level triggers does not stress the need for access to fine-grain data, and custom-made solutions for implementing the algorithm are discussed exclusively. Other investigations [3] concentrate on transmitting all detector data needed in second-level trigger calculations via a switching network of high bandwidth to a "farm" of general-purpose processors.

The results presented here include solutions with the general characteristics described above, studied as part of the EAST (embedded architectures for second-level triggering) project [1]. The present paper concentrates on a discussion of results from a

systematic study of possible architectural solutions for two feature extraction tasks. These were defined as characteristic second-level trigger tasks in terms of physics goals, detectors and triggering algorithms, and with fixed assumptions about detector and first-level trigger electronics. Benchmark implementations of such algorithms have been done on competing architectures, on available hardware where possible, by detailed simulation otherwise. Such isolated hardware implementations of algorithms, or simulation, demonstrate an architecture's possibilities and limits, and permit to evaluate a first approximation of cost (although this report does only give rough estimates for cost). Practical implementations including data input show the problems which will have to be solved in embedding architectures in the data flow of future detector electronics. They will be one of the next goals of EAST, for some of the more promising alternatives, along with studies of the multi-RoI operation with global decision taking.

2. THE BENCHMARK ALGORITHMS

In a workshop held at CERN in October 1991 [4], we have defined two pilot tasks for our benchmarking of second-level trigger algorithms. Both definitions arose from close collaboration with R&D projects (RD-1 and RD-6) pursuing the corresponding detector developments. The algorithms, or rather the problems together with a possible algorithmic solution, are defined in internal EAST Notes [5]. The pilot tasks and algorithms reflect a certain state of detector development, frozen for our purpose. They are strictly limited to feature extraction in a single RoI, leaving the connection between possible multiple windows in the same detector, or the correlation between features in different detector subsystems for the same RoI, to a global second-level decision unit. It is, however, one of the criteria used for judging an implementation to discuss possible connections to such a global decision unit. At the level of feature extraction, the cuts to be applied to the features, and the logical relations between results from different truth values of cuts, must remain flexible to account for different physics goals (e.g. "jets" from τ leptons) or varying luminosity.

Very briefly, the pilot tasks are the following:

(a) A calorimeter algorithm based on an indication from a first-level trigger window of a region of 16×16 towers. Each tower of our calorimeter is subdivided into 2×2 electromagnetic cells, one hadronic and one mixed (wedge-shaped) cell. In our simulation the RoI of 256 towers corresponds to $\Delta\eta \times \Delta\Phi = 0.5 \times 0.5$. The objective of the algorithm is to find features, i.e. decision variables, suitable to distinguish electrons from pions or from hadronic jets, or even pions from jets. Inside the RoI, a near-circular region has first to be defined in which the peak energy deposition is fully contained (cluster area). The second moment of the

cluster radius in two dimensions, or energy sums in different ring-shaped zones and longitudinal volumes of fine granularity, are then calculated over the cluster area as the most complicated variables defining isolation, as used in off-line programs. They are likely to contain all relevant information for electron/pion/jet discrimination that cannot be explored in a first-level algorithm. They also allow to refine the positional resolution, inside the RoI.

(b) A tracking algorithm based on a straw geometry as pursued in a transition radiation detector (TRD), operating in a magnetic field. Radial straws at constant $\Delta\phi$ are arranged in planes perpendicular to Z, which coincides with the magnetic field axis. Tracks leave a hit when traversing a straw chamber, whose pulse height is indicative of particle mass. Again, the assumption is that a preceding step selects a RoI of TRD data for further treatment. In the projection natural for the TRD, i.e. in the Z/Φ plane which corresponds closely to the readout coordinates straw/plane number, tracks appear as straight lines, with the slope $d\Phi/dZ$ corresponding to p_T, the position along Φ indicating azimuth, and start and end point crudely indicative for Z. The algorithm then recognizes patterns of digitizings that correspond to high-momentum tracks, taking into account the pulse height distribution of digitizings for identification of electrons. The algorithm consists either of making histograms along Z (simplest, without a result on p_T), or along roads of different $d\Phi/dZ$ (with an indication of p_T). With a detailed knowledge of the geometry, the precise positions of straws can be used to refine the histogramming and get a further improvement in p_T.

For both pilot tasks, the algorithms are simple enough to be reprogrammed or even hardware-implemented in dependence on the architecture to be studied. For a clear definition, algorithms are given in a high-level language, but some freedom was left so that equivalent but different algorithms are also acceptable. Data sets for testing were also provided. They contained data values inside RoIs, for signal and background collisions, at different levels of luminosity (minimum bias background). A full simulation of physics background, i.e. of detectors with a first-level trigger filter, was not available (it is being done now in collaboration between EAST and the groups working on Letters of Intent for complete detectors). This shortcut was taken so that the benchmarks could be completed in a reasonable time; it does not invalidate the implementation work in any way, if the algorithms are considered as meaningful. On the other hand, this data does not allow to derive a judgement on the value of the second-level trigger for the detector system.

3. TARGET ARCHITECTURES

3.1 MaxVideo

The MaxVideo system [6] is a family of many powerful and special–purpose VME boards. Properly connected via point-to-point links or a general switch, they can perform the type of algorithms needed for feature extraction in second-level triggering applications. They are manufactured by Datacube Inc., Peabody. A flexible and expandable bus system (MAXbus) allows to define arbitrary data paths between the different modules at a rate of 10 or 20 Mbytes/s/port (fig. 1).

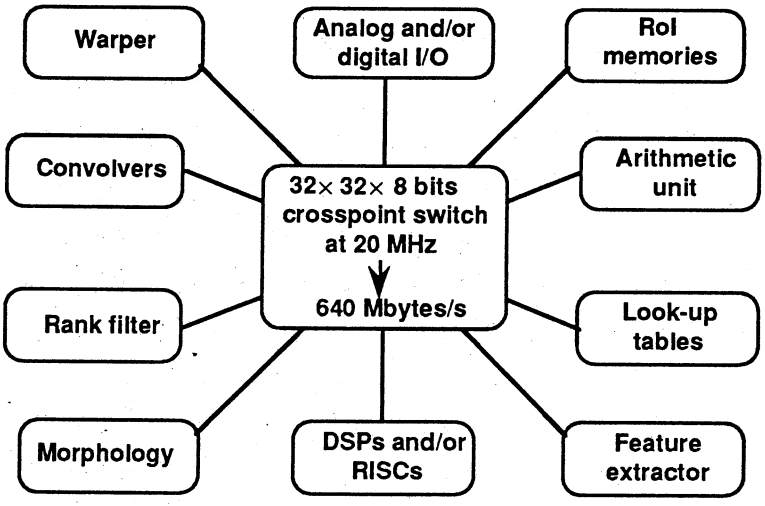

Fig. 1

Regions of interest can be specified in the internal memory, from 1 × 1 to 4096 × 4096 pixels and the processing time is proportional to the number of pixels in the RoI. Examples of interesting modules are the triple-ported **ROI memories** from 512 × 512 × 16 bits to 32 Megapixels. **Convolvers** exist from 3 × 3 up to 16 × 16 arbitrary coefficient kernels and a large kernel convolver for 2 × 256 × 64 with 3 coefficients only. This corresponds to a processing power of 180, 5120 and 320 000 million operations per second, respectively. There are **rank filters** and **morphological processors**. Very interesting is the **warper**. It allows arbitrary geometrical transformations and could be very useful as part of a router. Other examples are programmable crosspoint switches, pointwise ALUs, a module to count the occurrences of many different events and to extract the coordinates of a specific event (**feature extractor**). Many modules contain **look-up tables** (LUTs) (from 256 × 8 bits to 65536 × 16 bits, i.e. an arbitrary function F(X,Y) can be performed on two 8-bit images X and Y).

To fill the gap between these special-purpose hardware modules and more general software solutions, for higher-level decisions on the output of one or more of these

systems, for random access and/or higher precision etc., **DSP** and/or **RISC processors** (e.g. i860) with a MAXbus interface are available.

3.2 DAVIS (Datawave)

The Datawave processor [7] from Intermetall GmbH, Freiburg im Breisgau, Germany, a daughter of the ITT semiconductor group, is a single-chip implementation of a programmable data-driven wavefront array processor. Figure 2 shows the processor architecture: 16 identical cells are integrated on one chip in a 4 × 4 array with nearest-neighbour communication. Chips can be cascaded in one or two dimensions.

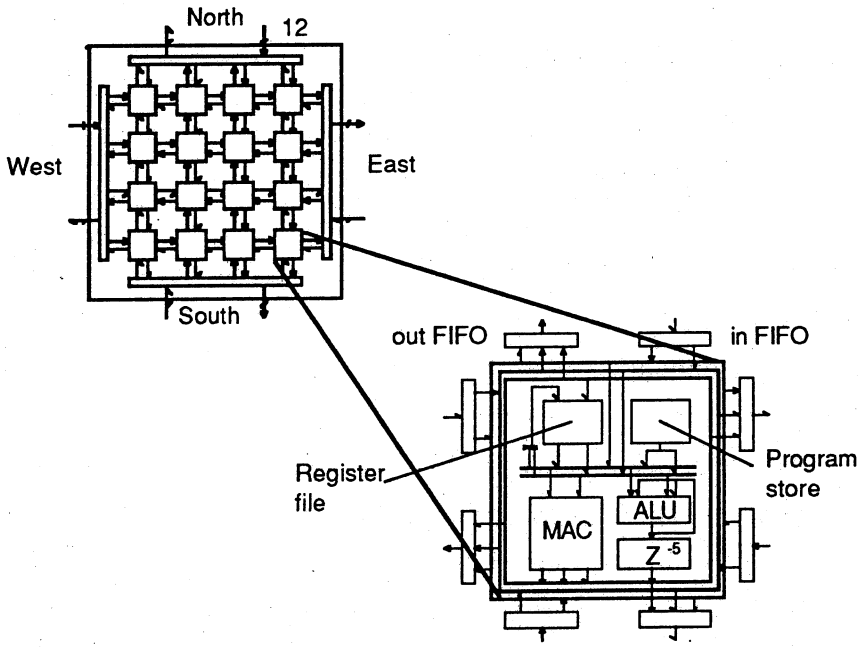

Fig. 2

Each cell is a RISC processor with a program store for 64 instructions (of 48 bits each). This long instruction word makes possible, in the same instruction cycle, the execution of arithmetic or logical operations, data transfers to 5 different destinations and conditional or unconditional branches. It is a deeply pipelined 12-bit architecture with a multiplier-accumulator of 29 bits length, an ALU, 16 registers etc. Each cell can e.g. start a multiply-accumulate in each cycle; thus, for the present prototypes running at 125 MHz, the peak performance is 250 Mops per cell. Because all data paths are 12 bits wide and a word can be transferred on-chip every clock-cycle and off-chip every second cycle, the intercell transfer rates are 1500 Mbytes/s or 750 Mbytes/s, respectively.

3.3 MasPar

The MasPar MP-1 computer [8] is a commercial compute-accelerator based on a massively-parallel SIMD architecture. The system is attached to a DEC 5000 workstation which serves as a front end, where all development tools execute. A parallel application

may execute on the data parallel unit (DPU) only, or on a mixture of front end and DPU. The MP-1 is programmable in Fortran 90 and MPL (based on C, but enhanced with data parallel statements). A source level debugger is available for fast and efficient debugging and profiling.

A SIMD computer, such as the MP-1, has one instruction stream which is sent to all processors by the array control unit (ACU). A processor either executes or ignores an instruction sent by the ACU, based on local computation or its processor number.

The MP-1 is built using reliable and proven VLSI technology (cycle time 80 ns). High performance is achieved through its large number of processing elements (PEs), of 4-bit word structure, with 80 ns cycle time, and 16 Kbytes of memory each. Systems are configurable from 1024 to 16 384 PEs.

For communication purposes, the processors are arranged as a two-dimensional mesh with (toroidal) wrap around, at 1.25 Mbyte/s for each line. Processors fetch a data element from another processor over the same distance and direction at a cumulated maximum rate of 20 Gbytes/s. The global router, a slower but more flexible communication mechanism, implements random processor-to-processor communication, similar to a telephone system. It is used for irregular communication patterns. Its maximum achievable total communication speed is limited to 1.4 Gbytes/s. The global router is also connected to the I/O buffer. Communication between processors and the I/O buffer cannot overlap with computation. Transfer into (and from) the I/O buffer takes place in parallel with processor computation. Several I/O devices, such as the HIPPI, disk array controller and high speed graphics interface, are currently under development.

3.4 Blitzen

The Blitzen system is based on chips, full custom VLSI components, having an array of 128 simple PEs operating in SIMD fashion. Massively-parallel systems can be built from these components. Since the chips individually have significant I/O capability, they may also be used in custom configuration for applications with high data rate requirements. The chips have been fabricated and a system prototype with 1024 PEs has been built. Blitzen is described in detail in ref. [9]. A brief description of an individual PE and the array architecture is given here.

Each Blitzen PE is a bit-serial processor with six single-bit registers, a variable length shift register, and a 1 Kbit of RAM. Arithmetic and logical operations are performed by a full adder and a logic block. Four single-bit paths interconnect PEs in a two-dimensional, eight neighbour grid. A four-bit I/O bus, shared among 16 PEs, provides a path to pads of the chip for connection to external RAM or other devices. Processing elements are organized on a chip in an 8×16 array. Chips represent the

building blocks for configuring large arrays of PEs. Eight such chips are needed to form a 1024 PE array as proposed for the full TRD benchmark solution.

The Blitzen system has three local control features: masking, address modification, and conditional operation. While masking is common in SIMD machines, the other two are not. Masking is simply the ability to enable or disable individual PEs.

Local address modification in SIMD machines has been called MIMD addressing or indirect addressing. Under program control, the global address sent from the control unit to every PE can be optionally modified in each PE using data from the shift register. This enables parallel data-dependent table access. It was used in forming the horizontal sliding-sum for TRD histogramming. In general, it provides greater flexibility in developing algorithms.

Local conditional operation allows one instruction from the control unit to produce different results in different PEs. Under program control, certain instructions test the state of a register, then produce the normal result or the complementary result. A major benefit is the ability to perform addition in some PEs and subtraction in others, simultaneously based on a data-dependent decision within each PE.

Architectural features involving more than one PE are the interconnection scheme that moves data between PEs and the I/O scheme that provides data movement between the array and external devices.

Processing elements are arranged in a two-dimensional array with an X-shaped grid of interconnections as shown in fig. 3. The cross-point of the X is a wire junction such that paths between PEs can be formed in eight compass directions. All PEs route the same direction in one processing cycle.

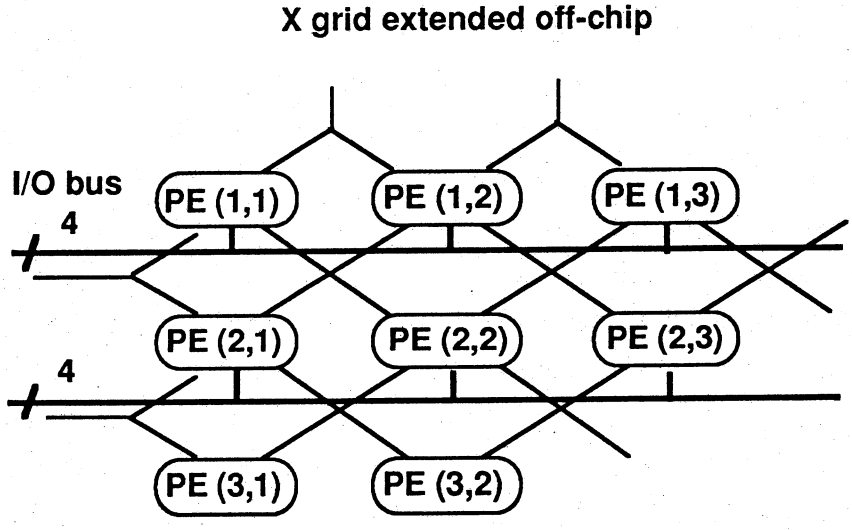

Fig. 3

Data I/O can be the limiting factor in the performance of a data parallel machine. In Blitzen, the array is segmented along chip boundaries to provide many edges for I/O. If 128 chips are used to make a massively-parallel array, the system can transfer 4096 bits in one cycle. The I/O system bandwidth potential, with a 20 MHz clock, is 10 240 Gbytes/s.

For I/O purposes, the RAM gives access to four-bit data items. For processing purposes, there is further selection to a single bit, consistent with the bit-serial processors. An I/O broadcast feature is implemented for distributing a common value to all PEs on an I/O bus.

Each PE has a local (on-chip) memory of 1024 bits. The memories of 16 PEs are connected to a bus which can be attached to high-speed video RAM (VRAM) devices as the next level (off-chip) in a hierarchy. Current technology has 1 Mbit VRAM chips organized 256 K × 4. Using just one chip per I/O bus provides storage of 64 Kbits per PE.

3.5 Associative string processor (ASP)

The ASP architecture is based on studies in the early 1970's at Brunel University; processor chips are now available from Aspex Ltd, Uxbridge [10]. The ASP SIMD architecture attempts to take advantage of the latest advances in technology; at present, multi-chip modules are under development. The arithmetic unit of the APE (associative processing element) is a single-bit full adder. Processors are connected along a single linear string, but the weight given to associative addressing, and fast skipping of processor rows or entire chips, outweigh the restrictions in communication.

The ASP customer is, in principle, offered a building block in VLSI, and given a high degree of freedom in adapting overall design parameters: the number of processors, processor internal resources, I/O and overall system design, can be chosen to fit the needs of the application. In practice, we are basing ourselves on a design pursued in the MPPC [11], with 64 APEs on a chip and 2048 processors and a single 32-bit wide serial I/O port on a board. Each processor has an internal RAM of 64 bits. In addition, there is a set of 6 "activity bits" to be used as control/status bits in the implementation of algorithms. The processor has a parallel comparator for 32 data and 12 activity bits, and the activity registers provide the associative function. Through activity registers, processors are tagged for execution by globally comparing the data on the data (and activity) bus by means of a masking facility. The result of the comparison sets a match flag and activity bits in each APE, and is also output to a single-bit line common to all processors, to allow the controller to find out if any of the processors had a match.

All processors are connected to a communication network. The network supports two modes of communication: an asynchronous mode primarily intended for exchange of activating signals, and a synchronous mode where data or activating signals are shifted from one processor left or right, to other processors on the string.

A controller broadcasts data, activity bits and control to all the APEs simultaneously. The normal sequence of instructions is to activate APEs according to a given criterion expressed as matching bits; then perform some operation on the data in the activated processors.

I/O to the APEs is via the controller's broadcast of data, and filling data into the active APE(s). This serial mode consists of filling 32-bit data words into and taking results out of one APE in 100 ns. If this is too slow, a possibility exists to exchange data words in parallel over the network to all APEs simultaneously. This allows a trade-off between the number of processors on the chip and the I/O buffer size.

The programming of the ASP is done in a Modula-2 environment, but detailed knowledge of the architecture is required for efficiency. Libraries exist for the most basic functions and for common image processing operations. A simulator is provided on a SUN workstation, or in a limited version on a PC.

3.6 Enable

Enable is a development project in the EAST Collaboration, to build a systolic array processor based on field-programmable gate arrays (Xilinx), for the specific task of triggering the transition radiation detector of RD-6. The Enable machine will be directly plugged into the TRD front-end readout electronics (prototype) via a router and a HIPPI serial-link interface. The Enable machine operates like a systolic processor array. The pixel image is shifted one column (of constant Φ) at a time into the rectangular mesh-connected array of processing elements, which execute on every operand the same matching function with pre-programmed patterns. According to the underlying deterministic space-time sorting and shuffling procedure, all matching data are routed to corresponding histogram channels. Operations are executed in parallel, on identical hardware, for low and high-threshold data. Peaks in low-threshold data (i.e. tracks) are then found, and a comparison of contents in high and low threshold result in the electron likelihood. The systolic data flow is maintained throughout, from data input up to the activation of the trigger lines.

The system can be scaled by changing the number of standard boards used. One board handles images of 60 pixels height and provides, according to the benchmark algorithm, m × k histogram channels, where m is the number of different bin offsets (= 20) and k is the number of patterns (= 15). A processor prototype board development

is under way; this board will serve for testing and evaluating the TRD emulation using EAST's hardware emulator (called SLATE). Simulation studies have shown, that the systolic machine will execute the TRD benchmarks with only minor deviations from the classification results shown off-line. The data flow has been tested on a clock-cycle basis with a hardware simulator. The prototype will be functionally testable later in 1992.

3.7 iWarp

iWarp is a scalable system architecture that meets the needs of high-performance embedded processing applications such as signal and image processing. It is based on the ideas of H.T. Kung [12], first put forward in 1978, according to which many problems can best be functionally decomposed for parallel execution in systolic processors. Intel has developed the commercial iWarp system [13] under a DARPA and Carnegie Mellon University sponsored cost-sharing contract. The iWarp custom VLSI component forms the basic building block for constructing iWarp systems that can be scaled over a performance range of 20 Mflops to 20 Gflops with a computational density of 1 Gflops per cubic foot.

A single iWarp component and 18 memory chips form one complete processor (called a cell) of the iWarp system. The cells in a configured system are connected in a two-dimensional toroidal mesh topology. Communication bandwidth between cells is 40 Mbytes/s, full duplex, simultaneously in all four directions. In other words, each iWarp cell supports a 320 Mbyte/s data rate. This addresses the most common limitation of existing parallel architectures and communications performance.

iWarp supports both the message-based communication model of current parallel systems, and a "systolic" communication model particularly well-suited to signal and image processing applications. The systolic model allows the computational element to use the data directly from the communications pathway without sacrificing precious memory access. Data from external sensors can flow, or be pumped, through the array of processors as it is used simultaneously by the hardware on a word-by-word basis, and imposes no additional overhead on the computation.

iWarp high-level programming tools support both FORTRAN and C languages with communications extensions for both message passing and systolic programming models. The iWarp cross-development environment is hosted on Sun 3 and Sun 4 model workstations along with system debug capability, application libraries and parallel-application development tools. Additionally, for program developers and small application needs, a single board array is provided that plugs into the host workstations.

The compute power of each cell in an iWarp system is 20 Mflops single precision and 10 Mflops double precision. Since the floating point arithmetic elements are not

pipelined, the computational unit provides scalar machine efficiency. At the heart of the cell architecture is a shared 128-word register file that supports 15 access operations in a single instruction. For example, during a single 100 ns compute cycle, the following operations can occur simultaneously in a cell:

- four incoming words received by the "communications agent" from the pathway (including two data reads by the "computation agent");
- four outgoing words sent by the communication agent over the pathway (including two data writes by the computation agent);
- two memory address calculations (independent base address plus offsets);
- two accesses to memory (read two, or read one and write one);
- one floating point multiply (read two, write one back);
- one floating point add (read two operands, write one back);
- loop count decrement and branch operation.

iWarp systems are based on a quad cell board with four iWarp chips and 0.5 Mbytes or 2.0 Mbytes of memory for each cell. Memory expansion modules provide the option for expansion to 1.5, 4.5 or 6 Mbytes/cell. The cards are held in a card cage assembly (CCA) which can support up to 16 cards (64 cells) and provides clock signals, cooling and integrated power supplies. The self-contained units can be embedded in larger systems or configured in a "system cabinet".

4. BENCHMARK CRITERIA

Quite different from the conventional benchmarking of main frame computer systems, which typically use very large application programs in a high-level language (dusty decks), compiler performance or adherence to portability standards are not part of our evaluation criteria. The main criteria for this real-time benchmark are:

(a) Algorithm execution time, separated into the two aspects decision time (the time interval between successive decisions) and latency (time interval for a given event between start of data input and output of results): this assesses the overall feasibility of a given architecture to contribute as a second-level trigger device (target numbers are 100 kHz for frequency, and of the order of 1 ms for maximum latency).

(b) Practical solutions to the high bandwidth input: this addresses a typical bottleneck for many commercial systems which are targeted at compute-intensive problems, and also challenges the flexibility of architectures or their manufacturers in interfacing to specific user constraints.

(c) Possible constraints on the order of input data; this aspect is relevant as we deal with architectures that typically achieve performance by high parallelism with distributed memory, or by pipelining data in a certain sequence: we assess here how much of the custom-made data selection has to be loaded with tasks that would be, in a general-purpose device, part of the algorithm itself.

(d) Interfacing to a high-level decision-making unit and to the (physicist) user: flexibility with respect to algorithm parametrization, and hardware possibilities of passing results (physics features) to a global device for overall decision making, are critical parameters in assessing the embedding difficulties of an architecture.

5. BENCHMARK RESULTS

Table 1 is an oversimplification, and should only be read in conjunction with the discussion in sect. 6. Some work is still in progress, so that numbers and implementation details will evolve in the future. More detailed discussions are given in several internal EAST Notes [14].

Table 1

Architecture/ Algorithm studied	Measured (best possible) execution time [μs]	Latency estimate [μs]	Comments
Pipelined			
MaxVideo/TRD	20.0 (10.0)	22.5	single system
MaxVideo/Calo	51.0 (12.8)	83 (21)	single system
DAVIS/TRD	3.0	6.0	
DAVIS/Calo	14.3	31.3	
SIMD			
MasPar/TRD	741 (16.3)	820	2400 (16 K) PE
MasPar/Calo	782 (12.2)	854	256 (16 K) PE
Blitzen/TRD	147 (14.8)	148	1024 (256) PE
Blitzen/Calo	76.6 (59.4)	80	256 PE
ASP/TRD	33.5 (9.5)	50 (25.5)	10 (2) systems of 2048 APEs each
ASP/Calo	20.6	not available	5 systems of ≤ 2048 APEs
Other			
Enable/TRD	6.5	6.9	I/O limited
iWarp/TRD	16 (9)	< 32	12 processors
iWarp/Calo	33 (17)	90	24 processors

6. DISCUSSION BY INDIVIDUAL ARCHITECTURE

6.1 MaxVideo

MaxVideo is a fully commercial system. All numbers were achieved on installed hardware.

Algorithm restrictions: the full algorithms were implemented for the calorimeter, all results are in full agreement with the off-line algorithm. For the TRD, only histogramming along Z was performed. The TRD times given correspond to two images of size 240 × 25 (1-bit) pixels, in Z and Φ. Multiple histogram scans along different slopes dΦ/dZ could be performed in parallel identical modules at very little cost in time, but this was not programmed.

Implementation details: for both algorithms, to avoid "flyback" time, the images were transmitted in a single line.

Comments on timing: timings given are for the MaxVideo10 (10 MHz) modules. An immediate factor 2 is achieved when moving to MaxVideo20, of which we have only an incomplete version. The algorithm execution time for the calorimeter was achieved with a single convolver board used twice; chaining up two of them would give another factor two (best time).

Input solution and data order: an interface HIPPI/Maxbus is under construction. It is possible to keep up with the algorithm speed at 20 MHz. Larger RoIs treated in MaxVideo, however, would increase the decision time. For both algorithms, line-by-line input is necessary.

Output solution and flexibility in output features: a general-purpose processor can be embedded in MaxVideo and connected via the internal video bus (commercial product). Output features are only limited by the operations available on standard MaxVideo cards.

6.2 DAVIS

The results were achieved using a proprietory simulator.

Algorithm restrictions: the full algorithms were implemented for the calorimeter, all results are in full agreement with the off-line algorithm. For the TRD, only histogramming along Z was performed. The times given correspond to two images as above. Multiple histogram scans along different slopes dΦ/dZ could be performed in parallel identical modules, but data splitting possibilities are unclear.

Comments on timing: timings given are for a chip assumed to run at 125 MHz.

Input solution and data order: optimal input is assumed, no interface can be discussed (no board-level product defined). For both algorithms, line-by-line input is necessary.

Output solution and flexibility in output features: cannot be discussed at the present stage.

6.3 MasPar

All results were obtained using a commercial system with 4096 processing elements (PEs), installed at the University of Basel.

Algorithm restrictions: the full algorithm was implemented for the calorimeter, all results are in full agreement with the off-line algorithm. For the TRD, only histogramming along Z was performed. The times given correspond to an image size of 240×10 pixels of two bits each, in Z and Φ. Multiple histograms or general pattern matching have not been explored.

Implementation details: the use of memory (off-chip) is avoided, all operations are done in registers.

Comments on timing: they are given for an implementation in MasPar's high-level parallel language MPL. A factor of two is estimated to be achievable by the company's experts, by resorting to machine language. Times given are for a single-event algorithm; this does not make good use of the massive parallelism of the machine. A total of 256 PEs is used for the calorimeter, 2400 PEs for the single-event TRD algorithm. A "best time" was therefore explored for multi-event decisions (64 events in both algorithms), with obvious consequences for overall latency and event buffering throughout the detector. In the case of the TRD, this version implied a different data storage order in the algorithm, hence a longer time for the set of events than achieved on a single event.

Input solution and data order: an interface from HIPPI into MasPar is under design (one per system). This would be an obvious bottleneck. For both algorithms, an input order is necessary which optimizes the transfer from the buffer memory into the PEs via the internal "global router". The details of this optimization have not been explored.

Output solution and flexibility in output features: a general-purpose host processor (DECstation) is part of the system. Limits in transfer rates are given by VME.

6.4 Blitzen

Results were achieved on a software simulator, assuming the processor clock frequency of the existing prototype chips (20 MHz).

Algorithm restrictions: for TRD, the benchmark was first implemented exactly as specified, histogramming along 4 slopes dΦ/dZ; the time given corresponds to an image size of 240 × 16 pixels in Z/Φ. The "best" time corresponds to histogramming along Z, using data precompacted (15 to 4 bits) in a LUT like in other implementations. In this approach, 8.55 μs are needed for the histogramming, additional 6.2 μs for the electron discrimination.

For the calorimeter, the full benchmark algorithm (16 × 16 towers) was implemented. The following results were obtained:

- with 13-bit input data, 76.6 μs (including 19.2 μs for loading),
- with 12-bit input data, 59.4 μs (including 14.4 μs for loading).

Implementation details: 1024 PEs were used (8 chips) for the first TRD approach, only 256 PEs for the "best" TRD and calorimeter algorithms.

Input solution and data order: one or several interfaces HIPPI-to-Blitzen could easily be designed; using a video RAM as intermediate buffer, and multiple parallel lines to each chip, data rates could be matched. Row-wise input is acceptable.

Output solution and flexibility in output features: a general-purpose processor can be easily connected via the VME bus.

6.5 ASP

Results were obtained using a proprietary software simulator [15], assuming an execution time slot of 20 MHz (existing chips run at 25 MHz).

Algorithm restrictions: for the TRD, local pre-histogramming along Z (LUT reducing every 16 bits along Z into a 4-bit count) was assumed to be done outside the ASP. The information loss related to this packing operation is local, and does not preclude the subsequent histogramming at different slopes dΦ/dZ. Apart from increased time for data transmission, histogramming with unpacked data and for larger slopes increases the execution time to 46 μs. For the calorimeter, the algorithm implemented uses different test statistics from most other implementations, but has been shown to be qualitatively equivalent. It is based on applying thresholds to six different energy sums (central peak, near neighbourhood ring, wider neighbourhood, e.m. and hadronic), i.e. on simple convolutions. The widest areas considered are 6 × 6 towers, the window slides dynamically over all possible positions in the RoI. The resolution for some energies is refined to the e.m. tower size, i.e. a fourfold number of cells is considered.

Implementation details: a full analyzed image size of 80 × 240 in Φ/Z has been assumed for the TRD. Different slopes are done in parallel, in independent ASP substrings. This assumes a parallel data copy into several identical ASP systems of 2048

APEs each. For the calorimeter, the six sums (= convolutions) are done in parallel in five independent ASP strings, with different precision used in the sums. The total number of APEs is 6400, the largest substring is 2048. Here, too, data has to be carefully replicated into multiple strings/APEs, to achieve efficient execution.

Comments on timing: timings given are for the existing board design of MPPC, ignoring data duplication and the input bottleneck related to the existence of a single 32-bit/25 MHz channel for 2048 processors (below). The "best time" (TRD) assumes only very crude histogramming along Z, or tracks that span no more than three bins in Φ. For the calorimeter algorithm, the logical connection between the results of the individual convolutions is not included in the timing.

Input solution and data order: the times given are for algorithm execution only. Input can proceed in overlap only if one assumes two identical systems, with a simple switch upstream. A solution for the data replication in the calorimeter case is not given; this would lead to a custom-designed unit(*).

Output solution and flexibility in output features: not discussed.

6.6 Enable

Results were achieved on a home-written software simulator, assuming an implementation on Xilinx 4000 series chips at 40 MHz.

Algorithm restrictions: the full algorithm of ref. [5] is implemented, using variable slopes. This systolic special-purpose processor array is, beyond that, conceived for general pattern matching of an image of fixed width (Φ). The array is fully scalable in image size along Z (latency), and in the number of patterns scanned (replication = cost). Assumed for latency are 240 Z values and 16 different patterns.

Comments on timing: reducing the algorithm to a single zero-slope pattern (histogram along Z), would decrease execution time.

Input solution and data order: one interface from HIPPI for each of the high and low threshold images, with a 20-bit parallel input ($n\Phi$) is foreseen. Today's HIPPI at 25 MHz will then take a minimum of 9.6 μs to transmit 240 columns at maximum speed, and will constitute the limiting factor.

(*) A separate timing has been performed for a TRD algorithm including the router function. The router is a data reformatting unit necessary to present to the second-level trigger architecture a RoI in a suitable and invariable data order, independent of the modularity and readout order of the detector front-end electronics. In the present TRD prototype, the router also reduces the information from 3 bunch crossings into a single two-bit signal for each straw. The additional time for logical connection of time slices and data interleaving is 5 ms. The factor of 3 in required bandwidth would, obviously, constitute an increased I/O challenge to the ASP design.

Output solution and flexibility in output features: a general-purpose processor can be easily connected via the VME bus. No other solution is discussed.

6.7 iWarp

Results were achieved on an 8×8 processor array, using machine-language coding for the critical program parts which use domains of the array as stages of a data-driven pipeline.

Algorithm restrictions: the full calorimeter algorithm was implemented; for the TRD, instead of variable slope histograms, two partitions in Z were made. The difference in position of the two partition peaks is a measure of p_T. However, results are not 100% equivalent due to the limited statistics in each partition.

Comments on timing: compared to high-level non-systolic formulation, a large factor was gained in time. The calorimeter algorithm was implemented as a 3-stage pipeline of 8 processors each, the TRD algorithm as a 2-stage pipeline of six processors each. The pipeline speed was shown to scale with the number of processors up to the available width (8). This confirms Kung's conjecture for linear speed-up with the number of processors, in systolic systems [12]. We consider an extrapolation by a factor 2 (for width 16/12) as permitted (best value).

Input solution and data order: a serial interface with specifications like HIPPI is being offered. Data sets have to come in suitable order to match the architecture, so that some prerouting is likely to be necessary.

Output solution and flexibility in output features: the iWarp computing element being a general-purpose processor, there is no conceivable limit to algorithm complexity, other than execution time and cost. A system performing multiple feature extraction algorithms for different subdetectors, in parallel, and subsequently using one processor to take the global decision, is arguably the solution of maximum ease of implementation, with the flexibility where it serves physics best.

7. COMMERCIAL, TECHNOLOGICAL AND COST ARGUMENTS

Our target architectures span systems that are based on commercial parts at very different levels. At one end, we use a custom-designed system making use of commercial tools and low-level programmable components only (Enable), at the other extreme, we envisage a very general architecture of independent general-purpose processors connected by a powerful and high-bandwidth data network (iWarp/MIMD), or a large-scale massively-parallel system (MasPar/SIMD), both marketed for "grand challenge" applications and corresponding to what the major manufacturers of high-performance

computing systems are advertising to become replacements of supercomputers. In between, we consider commercial modular image processors marketed for quite different applications (MaxVideo) or components which may come on market in other contexts and will require custom embedding (DAVIS, Blitzen and ASP).

From today's point of view, the extrapolation into the future of general markets, and an evaluation of the likelihood to remain close to the edge of technology, favour the low-level field-programmable gate array components of Enable, the image processing modules of MaxVideo, and the MIMD systems of high-performance computing.

Cost estimates, at this stage, must be superficial. Systems that are custom designed, around general or specific components (Enable, DAVIS, Blitzen and ASP), will show a low component cost (like of the order of 50–100 KCHF per feature extraction window, possibly much less), but cost will be dominated by detector-dependent parts and by the investments in manpower. MaxVideo will need much less interfacing work, and a one-window system can be estimated around 100–150 KCHF, high-level processor included. Much less clear is the pricing of the general-purpose parallel systems: a MasPar system with 4 K PEs or a 64-processor iWarp both have a list price of the order of 1 MCHF, but no detailed discussions concerning scaled-down versions have taken place. It could be argued, of course, that the generality of a MIMD system with HEP-suitable connections would constitute an ideal general and scalable approach for multiple subdetectors, for both general interfacing, local feature extraction and global decision making; a comparatively high price tag could be defendable with such an argument.

8. BENCHMARK CONCLUSIONS

Benchmarking is an attempt to reduce a multi-dimensional problem to very few parameters, and to measure these for competing systems using representative problem formulations. The procedure necessarily gives results of a limited validity, and many other factors will enter decision making. This has been found true over a long history of "dusty deck" benchmarking of mainframes. Benchmarks in a real-time environment like ours give more freedom to the implementer, and have no comparable history. When studying the detailed discussion above, it becomes clear that human ingenuity plays a role nearly as important as architectural details: the individual contributions to our study have made use of the allowed freedom to a different degree. The balance between computing power, communication bandwidth, and memory characteristics of future systems is a delicate parameter, and precise detector requirements will have to be available for taking the decisions concerning future trigger systems.

Nevertheless, we believe that our studies are the most significant that can be done under the given conditions. They certainly allow to draw some first general conclusions:

(a) Some of the commercial devices used in this study, can meet today the requirements in execution speed, latency, flexibility as stated for our feature extraction algorithms. They, or comparable successors, can be considered serious candidates for embedding in future readout systems for particle detectors.

(b) Pipelining coupled with a (near-) synchronous data flow appears to be the natural way to achieve our goal of 100 kHz algorithms. This is true independently of using a clock-driven [12] synchronization (Enable and MaxVideo) or a data-driven [16] programmed pipeline (DAVIS and iWarp). Pipelining is based on a functional decomposition of the algorithm; the simple parallelism of SIMD systems (all processors execute the same function), therefore, does not seem to correspond to this problem formulation. A very general asynchronous multi-event "farm" approach (based on general-purpose processors with message-passing communication), on the other hand, necessarily leads to large bandwidth and memory requirements and causes long latency (single-RoI algorithms execute typically in milliseconds in a modern RISC processor).

(c) Massive parallelism implemented with simple processing elements does not correspond ideally to the problems benchmarked, and is difficult to put to use (MasPar). When scaled down to a limited number of processors, custom-embedded into the data flow, SIMD concepts appear to be usable, although presently only the simplest algorithms have been implemented, for which programmed machines may seem an overkill. It would appear then, that a SIMD-based trigger architecture could take advantage only of the basic multiprocessor chip, pipelining multiple small SIMD systems. Such scaled-down SIMD systems, however, are not on the market. All processor control (issuing of instructions) including software, all parallel I/O, and all monitoring devices, would then have to be custom developments of HEP laboratories.

(d) All commercial parallel systems favour short-range communication, and run into bottlenecks with global sums as needed in histogramming or in the computation of isolation criteria. Only the Enable machine or other custom-designed processors (e.g. convolvers in image processors) seem in a position to cope with the summing problems efficiently in our benchmarks. A different example: the overall energy vector sums to tag neutrinos, needed in the first-level trigger, cannot be found efficiently in any of the architectures considered (also ref. [17]). In our benchmarks, the histogramming/summing problem was circumvented in several implementations by postulating an order of input allowing to form bit sums by a preprocessing step implemented as a table look-up.

(e) The desire for utmost flexibility in algorithms finds its limit in that part of the program which has to do with data communication: arguably, it is of no advantage

to have a programmable processor array, if the data are driven through this array in a fixed sequence, so that only few algorithmic choices remain possible. Only in this way does it seem possible to reach decision frequencies as postulated for second-level triggers. Moreover, although programmable, several systems (DAVIS, Blitzen and ASP) are so only at a comparatively low level, requiring special expertise to take advantage of the specific details of the architecture in question. Typical feature extraction algorithms will eventually have to be firmly coded in hard or soft form, with only critical parameters remaining under control of a custom-designed user interface.

Acknowledgements

We express our gratitude to J. Nelson and G. Greer (Intel Corp.), H. Figel (MasPar Distributor, Zürich) and R. Frank (Universität Basel) for active participation in porting the benchmark algorithms to iWarp and MasPar, and for providing free access to the two systems.

REFERENCES

[1] EAST (RD-11), Proposal and Status Report: CERN/DRDC 90–56 (1990) and CERN/DRDC 92–11 (1992).

[2] SDC 92–201 Solenoidal Detector Collaboration, Tech. Design Report, Chapter 8 (April 1992).

[3] E. Barsotti et al., Switch-based data acquisition system, Symposium on Detector R&D for the SSC, Fort Worth (Oct. 1990).

[4] Algorithm and Architecture Workshop 14–15 October 1991, EAST Note 91–16 (1991) (copies of transparencies).

[5] Algorithm and data definition:

J. Badier, R.K. Bock, C. Charlot and I. Legrand, Benchmarking architectures with Spacal data, EAST Note 91–10 (1991);

P. Bialas, J. Chwastowski, P. Malecki and A. Sobala, Benchmarking with data from the transition radiation detector, EAST Note 91–11 (1991).

[6] MaxVideo documents: User Manual and Reference Guide, Datacube Inc., Peabody.

[7] U. Schmidt, Data-driven array processor for video signal processing, IEEE Transactions on Consumer Electronics (1990).

[8] MasPar documents: System Overview (PN 9300–0300), MPL Reference Manual (PN 9302–0000), MPL User's Guide (PN 9302–0100).

[9] D.W. Blevins et al., Blitzen: a highly integrated massively-parallel machine, J. of Parallel and Distributed Computing 8/2 (1990) 150.

[10] R.M. Lea, The ASP: a cost-effective parallel microcomputer, IEEE Micro, Vol. 8, No. 5 (1988).

[11] MPPC project: Status Report CERN/DRDC 90–76 (1990).

[12] H.T. Kung and C.E. Leiserson, Algorithms for VLSI processor arrays (1980), eds C. Mead and L. Conway, Introduction to VLSI systems (1980);

H.T. Kung and C.E. Leiserson, Systolic arrays (for VLSI), Sparse Matrix proc. (1978);

W.M. Gentleman and H.T. Kung, Matrix triangularization by systolic arrays, proc. SPIE, Vol. 298, Real-time Signal Processing IV (1981).

[13] S. Borkar et al., iWarp: an integrated solution to high-speed parallel computing, Supercomputing '88, IEEE Computer Society and ACM SIGARCH, Orlando, Fla (1988) 330.

C. Peterson, J. Sutton and P. Wiley, iWarp: a 100 Mops microprocessor for multicomputers, IEEE Micro (June 1991) 26.

Intel documents: Introduction to iWarp, User's Guide, Programmer's Guide.

[14] Benchmark Results Workshop 11–12 May 1992, EAST Note 92–16 (1992) and accompanying notes:

R.K. Bock, H. Figel and I. Legrand, The MasPar computer as second-level trigger architecture for calorimeter windows, EAST Note 92–04 (1992);

G. Greer, I.C. Legrand and J. Nelson, A systolic cluster identification algorithm implemented on the iWarp system (Draft), EAST Note 92–08 (1992);

A. Sobala, Benchmarking with data from the transition radiation detector implementation on the MasPar computer, EAST Note 92–10 (1992);

S. Centro, E. Davis, Ping Ni and D. Pascoli, A parallel algorithm for feature extraction from transition radiation detector data: benchmark results using the Blitzen parallel processor, EAST Note 92–11 (1992) (also DFPD92/EI/28 Dip. Fisica, Padova);

A. Thielmann, The ASP benchmarks for the second-level trigger (TRD), EAST Note 92–12 (1992);

R.K. Bock, B. Greer, I.C. Legrand and J. Nelson, Systolic histogramming technique for track finding on the iWarp system, EAST Note 92–13 (1992);

G. Vesztergombi and G. Odor, ASP algorithm for second-level TRD triggering, EAST Note 92-14 (1992);

F. Klefenz et al., Benchmark results for the Enable machine, EAST Note 92–15 (1992);

A. Gheorghe and W. Krischer, Second-level trigger algorithms on some pipelined signal processing architectures, EAST Note 92–17 (1992).

[15] VASP-SIM simulator Version 1.2, Aspex Microsystems Ltd, Uxbridge.

[16] S.Y. Kung, VLSI array processors, Prentice Hall (1988).

[17] S. Lone et al., Fine-grain parallel compter architectures in future triggers, Nucl. Instr. and Meth. A289 (1990) 507.