REAL-TIME TRACKING WITH A 3D-FLOW PROCESSOR ARRAY[*]

D. Crosetto

Superconducting Super Collider Laboratory[+]
2550 Beckleymeade Ave.
Dallas, TX 75237

Abstract

The problem of real-time track-finding has been performed to date with CAM (Content Addressable Memories) or with fast coincidence logic, because the processing scheme was thought to have much slower performance. Advances in technology together with a new architectural approach make it feasible to also explore the computing technique for real-time track finding thus giving the advantages of implementing algorithms that can find more parameters such as calculate the sagitta, curvature, pt, etc. with respect to the CAM approach. The report describes real-time track finding using new computing approach technique based on the 3D-Flow array processor system. This system consists of a fixed interconnection architecture scheme, allowing flexible algorithm implementation on a scalable platform. The 3D-Flow parallel processing system for track finding is scalable in size and performance by either increasing the number of processors, or increasing the speed or else the number of pipelined stages. The present article describes the conceptual idea and the design stage of the project.

1. INTRODUCTION

In order to have better rejection at the Level-1 trigger, based on the calculation of additional parameters with respect to CAM, a 3D-Flow parallel processing system approach has been investigated.

A competitive solution in performance to the CAM approach is derived by using not only a parallel-processing solution, but also a processor with a special architecture presently not available on the market. These features are contained in the 3D-Flow processor and consist of high-speed communication ports (a large number of them to allow fast communication in six directions), and standard arithmetic operation as in regular processors. In addition, the processor also perform some special instructions to more efficiently execute high energy physics algorithms, FIFOs at the input port, to derandomize the processor clock with an external device clock (at the detector), and data-driven types of operations. Highlights of the proposed scheme are depicted in Section 3.2.

2. 3D-FLOW PROCESSOR

The 3D-Flow processor, Figure 1(a) and 1(b), is a programmable, data stream pipelined device that allows fast data movements in six directions with digital signal-processing capability. The design of the processor has been completed, and 225 hours of consultancy from industry have checked the feasibility of the 3D-Flow idea. A total of 6000 lines of VHDL code, describing the behavior of the single units and their interconnection, allows one to simulate algorithms and check the timing of all signals in the circuit. A table format of Microsoft Excel sets the input/output conditions at the external pins of the processor at each state. Other formats are used to download data-memory values into the processor. Program memory, thresholds and counter values have also been provided.[1, 2]

The 3D-Flow operates on a data-driven principle. Program execution is controlled by the presence of the data at five ports (North, East, West, South, and Top) according to the instructions being executed. A clock synchronizes the operation of the cells (a prototype will be made at 60 MHz). With the same hardware one can build low-cost, programmable, Level-1 triggers for a small and low-event-rate, or high-performance, programmable Level-1 triggers capable of executing more complex algorithms.

[*] Presented at the Workshop on B Physics at Hadron Accelerators, June 2, 1993, Snowmass, Colorado.

[+] Operated by the Universities Research Association, Inc., for the U.S. Department of Energy under Contract No. DE-AC35-89ER40486.

At each input port of the 3D-Flow processor there is a FIFO that de randomizes the data from the calorimeter to the processor array. North, East, West, and South ports are 12-bit parallel bidirectional on separate lines for input and output, while the Top port is 12-bit parallel input only, and the Bottom port is 12-bit parallel output only. North, East, West, and South ports are used to exchange data between adjacent processors belonging to the same 3D-Flow array (stage).

Top and bottom ports are used to route input data and output results between stages under program control. Each 3D-Flow cell consists of a Multiply Accumulate unit (MAC); arithmetic logic units (ALUs); comparator units; encoder units; a register file; an interface to the Universal Asynchronous Receiver and Transmitter (UART), used to preload programs and to debug and monitor during their execution; data-memory to be used also as a look-up table to linearize the compressed signal, to remove pedestals, and to apply calibration constants; and a program storage surrounded by a system of three-ring buses. At each clock, a three-ring bus system allows input data from a maximum of two ports and output to a maximum of five ports. During the same cycle, results from the internal units (ALU, *etc.*) may be sent through the internal ring bus to a maximum of five ports. The architecture of the 3D-Flow Processor cell is shown in Figure 1(a), the input/output in Figure 1(b).

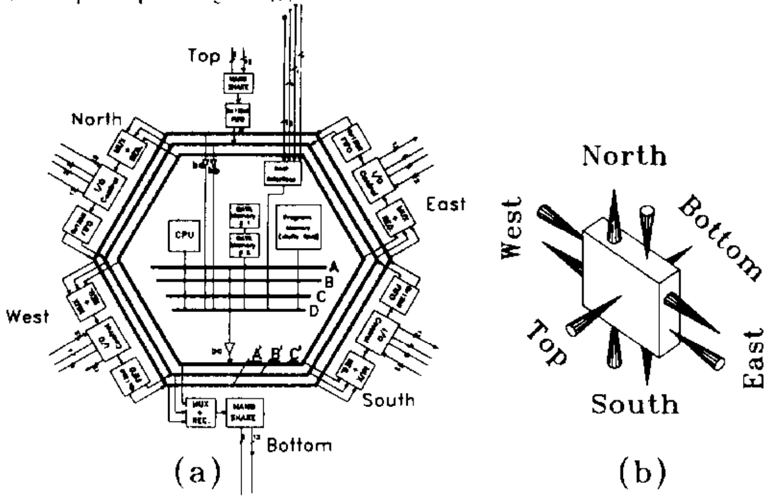

Figure 1. 3D-Flow Processor.

3. REAL-TIME TRACK FINDING AND REJECTION

3.1 Lookup-tables versus computing techniques

Real-time tracking techniques, used to date in experiments are of the look-up table type. The look-up table technique has a very fast response, but it requires a large amount of memory and is limited to recognizing only tracks that have been prerecorded into the memory.

Another technique, not used very much to date, is the computing technique that offers the advantage of implementing algorithms rather then just relying on a coincidence. One reason that it has not been used is its much lower performance with respect to look-up tables.

The following approach of real-time tracking, with the 3D-Flow parallel-processing system offers a fast and programmable response that may solve the problem in some real-time tracking applications.

Figure 2 depicts how information from different sub-detectors is sent into the 3D-Flow parallel-processing system.

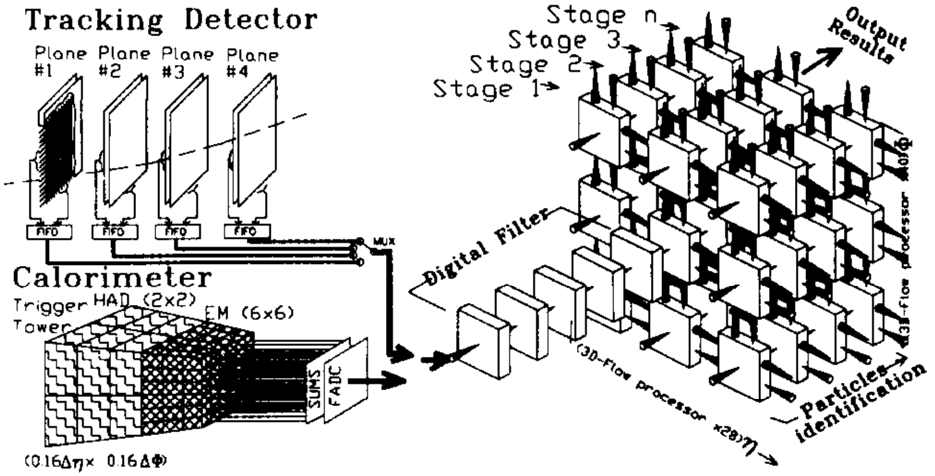

Figure 2. 3D-Flow system receiving data from different sub-detectors.

In the track-finding application, a number of 3D-Flow processors are used for each "plane." Depending on the complexity of the algorithm and the number of tracks expected in a given area, the user decides the most convenient price/performance segmentation of the "plane" in smaller areas, each of which sends the information to a 3D-Flow processor.

As an illustrative example, Figure 3 shows the mapping of the strip (wire) signals to a 3D-Flow processor array, while Figure 4 shows the mapping of the signals from a subset of a tracking detector into a 3D-flow processor. Thus, if we have a "plane" (consisting of several subplanes "x", "y", "u", "v") of 512 wires or strips and we know *a priori* from Monte Carlo simulation that the number of expected tracks is not greater than 10, then a convenient segmentation would be a 11 × 11 3D-Flow processor array for each "plane," such that each one receives as input a small fraction of information of the entire plane (*e.g.*, 48 wires or strips of each subplane).

We assume that an approximate vertex point has been located in a first step of the Level-1 tracking program.[3] For each detector plane there is a 2-dimensional 3D-Flow processor array; for successive detector planes there are successive arrays (or stages).

Each 3D-Flow processor takes the x and y coordinates from a hit on the first plane and computes the predicted coordinates on the next plane by a straight-line extrapolation. If curved tracks are expected in one or two dimensions, the processor in the next array should look for a hit in a wider region of interest. In the next plane, and in the corresponding small area, the 3D-Flow processor checks whether the predicted x and y coordinates lie in its region of operation. If so, the processor should find a hit which may come close to a straight-line predicted value (or deviates by a relatively small amount if a curvature is expected). The processor calculates, for this track, the new slopes (in x and y), the sagitta, the momentum (P), and the transverse momentum. The results of the calculation are passed on to the 3D-Flow processor that will operate on the corresponding area element in the next plane.

If the calculation to see whether the predicted x, y coordinate pair lies in the operating region of the individual 3D-Flow processor shows that it does not, the processor then forwards the received quantities to the adjacent 3D-Flow processor in the same array (or stage). The processor that finds that the predicted coordinates match its operating area then checks for continuity of the track in that plane by searching for a hit in its region. If the hit is found, the processor calculates the momentum, etc., and the result is forwarded to the next processor array (or stage), and so on.

3.2 Tracking Detector Versus 3D-Flow Processor Array

The tracking detector versus the 3D-Flow processor array is shown in Figures 3 and 4.

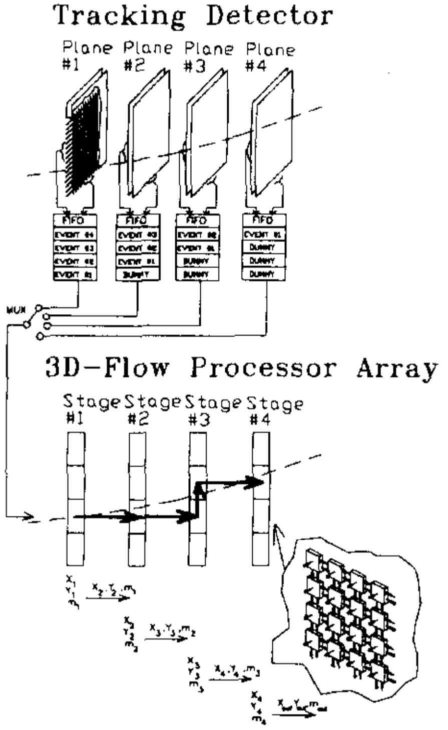

Figure 3. Tracking detector versus 3D-Flow processor array.

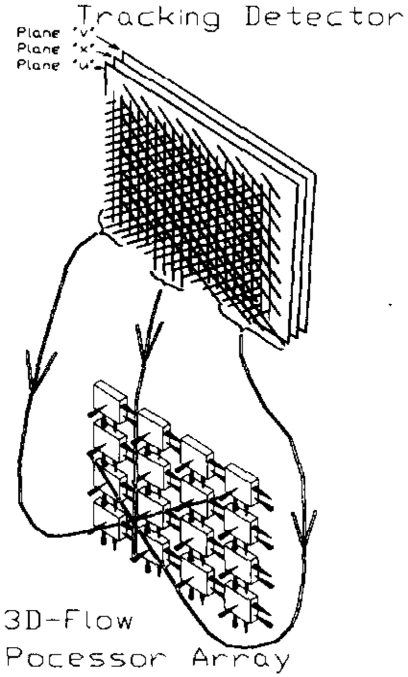

Figure 4. Mapping of signals from a subset of a tracking detector into a 3D-Flow processor.

The signal from the wires of the central plane is sent to the input of all 3D-Flow processors of the second column, as are the signals from the wires of the other planes to the other processors as shown in Figure 4.

3.3 Timing and Synchronization

Depending on the amount of computing required to calculate the unknown parameters and the number of hits per plane, the user selects an appropriate segmentation of the plane and associates it to a 3D-Flow processor array. Note that the high communication speed of the 3D-Flow processor allows the exchange of data between adjacent areas, thus allowing a system with no boundary limitation.

In Table 1 the four columns represent the activity of the processors in the four arrays (or stages): the rows indicate (from top to bottom) the timing sequence: the activity at each timing sequence for the arrays is indicated in the corresponding row.

Since each processor has the capability to simultaneously move data and perform calculations, two columns have been reserved for each processor array in order to indicate these activities. For example, row "zero" indicates that data event #1, from detector plane #1, is moved to processor stage (or array) #1: and row "one" indicates that the received data of event #1, from plane #1, is processed in the processor stage (or array) #1, at the same time that the processor is receiving the data of event #2 from plane #1, and so on for row "three," *etc.* Following this sequence, by row "eight" the results of event #1 are ready for output. At this time, the pipe is full, and all the processors are performing the two operations of moving and computing on data from different events.

Table 1. Timing of "data moving" and "data processing" on each 3D-Flow stage. Results are moved in sequence after computing.

Time	Detector Plane #1 = 3D-Flow array # 1		Detector Plane # 2 = 3D-Flow array # 2		Detector Plane # 3 = 3D-Flow array # 3		Detector Plane # 4 = 3D-Flow array # 4	
	3D-Flow Processing	3D-Flow Data mover	3D-Flow Processing	3D-Flow Data mover	3D-Flow Processing	3D-Flow Data mover	3D-Flow Processing	3D-Flow Data mover
0		in EV1-PL1-ST1						
1	Computing EV1-PL1	in EV2-PL1-ST1		in EV1-PL2-ST2				
2		Res EV1 to ST2						
3	Computing EV2-PL1	in EV3-PL1-ST1	Computing EV1-PL2	in EV2-PL2-ST2		in EV1-PL3-ST3		
4		Res EV2 to ST2		Res EV1 to ST3				
5	Computing EV3-PL1	in EV4-PL1-ST1	Computing EV2-PL2	in EV3-PL2-ST2	Computing EV1-PL3	in EV2-PL3-ST3		in EV1-PL4-ST4
6		Res EV3 to ST2		Res EV2 to ST3		Res EV1 to ST4		
7	Computing EV4-PL1	in EV5-PL1-ST1	Computing EV3-PL2	in EV4-PL2-ST2	Computing EV2-PL3	in EV3-PL3-ST3	Computing EV1-PL4	in EV2-PL4-ST4
8		Res EV4 to ST2		Res EV3 to ST3		Res EV2 to ST4		Res EV1 to Out

4.0 CONCLUSIONS

The 3D-Flow system provides an alternative to real-time finding lookup-table technique with a relative fast track finding computing technique. The advantages among the two techniques is the less amount of memory required by the computing technique, thus lower cost. Additionally, it allows calculation of more parameters, *e.g.* sagitta, pt, *etc.* in order to achieve better rejection.

ACKNOWLEDGMENTS

I would like to acknowledge W. Selove and T. Pal for the fruitful discussions.

REFERENCES

1 D. Crosetto, "3D-Flow processor for a programmable Level-1 trigger," Nuclear Science Symposium (NSS), Medical Imaging Conference (MIC), Orlando, Florida, October 26-31,1992.

2 D. Crosetto, "Digital Programmable Level-1 Trigger with 3D-Flow Assembly," SSCL-PP-445, August, 1993. *Submitted to Nuclear Instruments and Methods in Physics Research.*

3. W. Selove, "Impact Parameter Trigger and Vertex Detector for Forward Collider," Univ of Pa. Report UPR-222E, Aug. 31, 1993. Submitted to Proceedings of the Workshop on B Physics at Hadron Accelerators, Snowmass Colorado, June 1993.