DESIGN OF A SECONDARY-VERTEX TRIGGER SYSTEM FOR A HADRON COLLIDER*†

D. Husby, Fermilab
and
P. Chew, K. Sterner, W. Selove
University of Pennsylvania

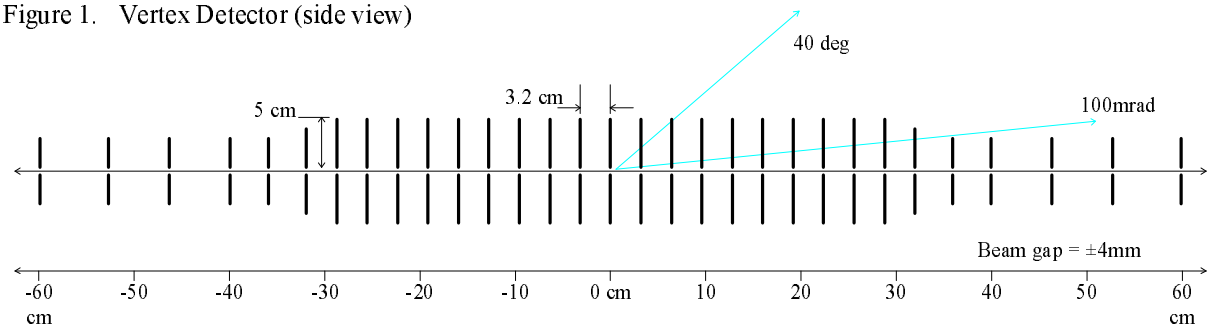
ABSTRACT

For the selection of beauty and charm events with high efficiency at the Tevatron, a secondary-vertex trigger system is under design. It would operate on forward-geometry events. The system would use on-line tracking of all tracks in the vertex detector, to identify events with clearly detached secondary vertices.

The Tevatron currently produces about 500 times as many B-$\overline{B}$ pairs per second as the expected rate from the SLAC and KEK B-factories. With the completion of the Main Injector, the rate will increase by another factor of 10 or so. These production rates offer a potentially extremely important capability for highest sensitivity for B physics. A critical component in exploiting this capability is a suitable trigger system. The trigger must be able to reject minimum bias events by a factor of order 1,000:1, while retaining B events with high efficiency. Clearly a secondary-vertex trigger, if highly efficient, would provide the greatest possible sensitivity.

We have designed a secondary-vertex trigger for a forward-geometry hadron collider experiment at the Tevatron. The vertex detector geometry is patterned after that originally proposed by Schlein [1], and operated by him in a test at the S-p-$\overline{p}$-S [2]. Figure 1 shows an overall side view of the vertex detector and its 31 detector stations. Planes are mounted inside the beam pipe; they are retracted during a beam store and then brought down to about 4 millimeters from the beam.

Figure 1. Vertex Detector (side view)



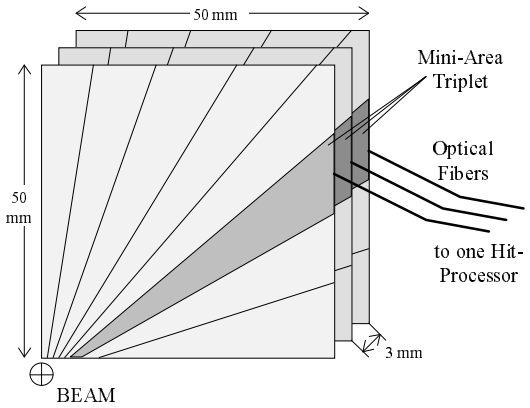
The primary job of the trigger processor is to analyze every event at interaction rates of 5 MHz or more, and quickly discard minimum bias events while at the same time recognizing and retaining B events with high efficiency. To this end, we have made several major modifications to Schlein's system. The modifications are designed to provide very fast on-line tracking with momentum information on each track. The momentum information is needed at trigger level in order to do a vertex search using only those tracks which can be extrapolated accurately to the vertex region[3]. Tracks of low transverse momentum will give large smearing of the vertex region, and can cause confusion by producing false secondary vertices. To prevent this smearing, and

yet do a vertex search at high speed, we wish to determine the pt of each track, approximately, so that tracks of very low pt can be left out of the vertex search. Momentum information of sufficient accuracy can be obtained, for the forward geometry, purely from the vertex detector if a suitable magnetic field is present. We note that the momentum information also makes possible a very rapid estimate of the effective mass of the daughter particles at a decay vertex, adding to the ability of the trigger to separate reconstructible B events from other events such as charm events.

In order to obtain fast on-line tracking with momentum information, the trigger system has several important features. Among the most important are the application of a dipole magnetic field at the vertex detector, the arrangement of triplets of planes at each station, the division of station area into mini-areas to facilitate a very high degree of parallelism in processing the individual hits on a track to obtain 3-d space points, and the sorting of hits into groups in azimuthal slices, to facilitate very rapid tracking[4].

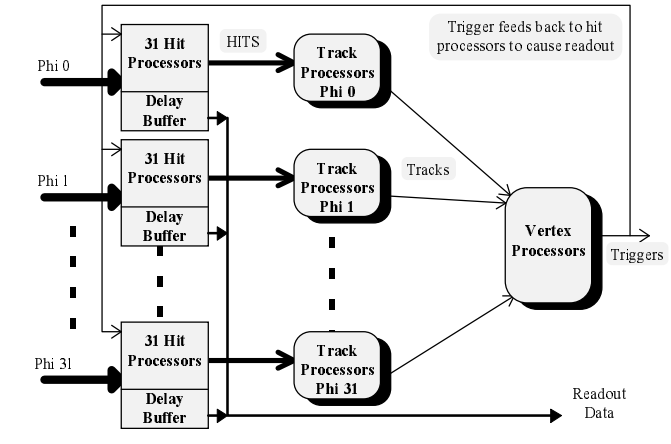
Each of the detector stations consists of a triplet of planes. Ideally each one is a pixel plane to facilitate very fast tracking algorithms [5]. The area of each plane is subdivided into 32 phi slices arranged as 8 "mini areas" in each quadrant as indicated in Figure 2. In this way the average number of tracks passing through one phi slice in one station is brought down to less than one track per interaction for minimum bias events. Thus the tracking problem in each phi slice of each station is a "mini-tracking" problem, and each such phi slice delivers individual station hits which can all be processed in parallel by hit-processors. For the design shown in Figure 2 the data rate on each fiber from each phi slice is approximately 3 hits per microsecond, at luminosity 10^{32}. The hits are fed on 992 separate fiber optic triplets to the trigger electronics.

Figure 2.
Division of Pixel Plane in one quadrant of a station.



As shown in figure 3, the trigger electronics consists of three levels of processors. At the first level, each fiber triplet feeds a single hit processor where raw hit data is processed independently, summarized, and then passed to the next level where it is merged with other hits within the same phi slice to form tracks. Similarly, tracks are processed independently (within a phi slice) and merged at the third level to form vertices. At each level, data is combined and summarized, reducing the bandwidth from about 100 Gigabytes per second at the front end to about 100 Megabytes per second at the output.

Figure 3. Trigger Electronics
Data flow through three levels of processors.



Raw Data from Detector 100 GByte/Sec	992 Hit Processors 330ns/Hit	Hit records 3 GHz 24 GByte/Sec	2048 Track Processors 10 uS/Track	Track records 200 MHz 3.2 GByte/Sec	128 Vertex Processors 16 uS/Vertex	Triggers 10 kHz Readout 100 MB/sec

Initial versions of programs have been written for the hit processor and for the track processor. At luminosity 10^{32}, there will be approximately one interaction per bunch crossing, with a group of 33 bunch crossings occurring at a repetition period of 132 nsec.

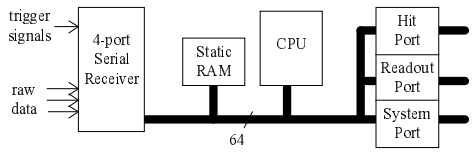
The hit processor has to analyze 3 hits per microsecond, on average, during this 33-bunch burst. Initial programming of this operation, by D. Crosetto, for the special purpose processor he has designed [6] gives a processing time of 15 to 20 clock cycles per mini-track [7]. For a commercial processor, with a single operation per clock cycle, it is likely to take somewhat longer.

The track processors have to analyze some 200 tracks per microsecond. An initial program has been written for the tracking operation, and gives a processing time, for a commercial microprocessor, appreciably smaller than 10 microseconds per track. Thus to keep up with the data rate, some 2000 track processors are needed.

Hit Processing

At the predicted hit rate of 3 MHz per hit processor, there's not much time to do detailed processing. For each hit, the processor must do four tasks: extract coordinates, find segments, assemble hit records, and place the raw data into a delay buffer for later readout. Figure 4 show a block diagram of a hit processor.

Figure 4. Hit Processor block diagram



Extract coordinates: When a track passes through pixel plane, it generally hits a cluster of pixels-- a center pixel and a one or two pixels on either side as shown in figure 5. Accurate hit coordinates are extracted by finding the center of mass of a cluster.

Figure 5.
Hit segments in a triplet

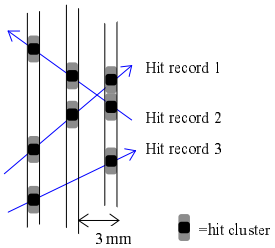


Figure 6.
Fields in a hit record

Size	Function
8 bits	Time Stamp
5	Station Number
3	Serial Number
2	Flags
1	Direction
12	Y Slope
15	X coordinate
15	Y coordinate
3	Cluster Size

Find segments: It is possible for several tracks to pass through a triplet during one interaction (although in most cases, there will be less than three). A simple segment finder cycles through the list of hits and assigns them to "mini-track" segments. Each segment is represented by a hit record.

Assemble hit records: A hit record is a 64-bit word that contains enough information to describe a track segment's 3-d position and direction, as well as route it to the proper track processor. As shown in figure 6, it contains several fields that must be merged into a single 64-bit word. The first 5 fields are used by the routing circuitry to deliver the hit record to its final destination in a track processor.

Manage delay buffer: Each hit processor must hold raw data in a FIFO delay buffer for a few tens of microseconds. When an 'accept' trigger is generated, the processor will transfer data for the accepted event to the readout port. Most of this is done as a DMA operation and requires only a few CPU cycles per trigger.

Track Processing

Hit records are passed through a switching network which sorts them by time stamp and direction, (and possibly by slope), and places them in a buffer in a track-processor. The result of this sorting is that all of the hits for a track are brought together in a single processor. The buffer may contain more than one track, but it will contain complete tracks (as long as they don't cross phi boundaries) which all belong to the same interaction.

Figure 7. Track Processor Block Diagram

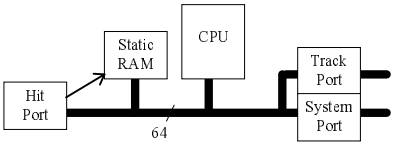


Fig. 8. Track buffer

Addr	Station	Serial#	Flags	
0	0	0	1	(valid)
1	0	1	1	(valid)
2	0	2	3	(end of list)
3	0	x	x	(un-used)
4	0	x	x	
5	0	x	x	
6	0	x	x	
7	0	x	x	
8	1	0	1	
9	1	1	3	
A	1	x	x	
B	1	x	x	
...				
F0	30	0	3	
F1	30	x	x	
F2	30	x	x	
.				
F7	30	x	x	

Fig. 9. Track Record

Size	Function
8 bit	Time Stamp
5	Phi Number
3	Serial Number
2	Flags
1	Direction
16	X-Slope
16	Y-Slope
16	X-intercept
16	Y-intercept
10	Momentum
8	py
3	py flags
6	Track quality factor
6	Vertex quality
12	spare
128	total

There are 64 track-processors per phi slice (2048 total), allowing an average of 10 microseconds for processing each track. A load-leveling circuit dispatches new events to the least-busy processor in order to minimize latency.

The format of the track buffer (shown in figure 8) is designed to speed up the track finding algorithm by pre-sorting hits by station number. The buffer has 248 locations arranged as space for 8 hits from each of the 31 stations. The station-number field and serial-number field are used as an index into the buffer when the hit is written. The flags are used by the software when parsing the buffer.

The track processor computes the track momentum and pt, and also a tentative location of the source point if the track originates on the beam line. For tracks with transverse momentum (py) below an adjustable cut value, a flag is set in the track record output (shown in figure 9) which is subsequently used by the vertex processor to assign low weight, or zero weight, to this track in the vertex search.

Vertex Processing

Track records from all 32 phi slices are brought together in the Vertex level. They are sorted by time stamp into vertex buffers. There are 128 vertex processors, allowing an average of 16 microseconds to process each event. When a clearly identifiable secondary vertex is found, and after a rough effective-mass calculation identifies the event as a promising B candidate event, a trigger signal is sent to the rest of the system to cause data for that event to be saved.

Switching Network

When data passes from one processing layer to the next, it must be merged, sorted, and load-leveled. Within each phi slice, for example, there are 31 hit processors emitting hit records that are used by 64 track processors. Data from all 31 hit processors must be combined together and sorted by time stamp. All of the records with the same time stamp must be delivered to a single destination processor. Destination processors must be chosen so that no single processor is swamped with too much work. In addition, all of this must be done at a sustained rate of 10 nanoseconds per hit record.

While this is an engineering challenge, it is not impossible. At one extreme such a network could be implemented as a single 64-bit wide bus with 31 masters and 64 slaves running at 100 MHz. At the other extreme it could be implemented as 1,984 slower point-to-point links.

The implementation that we have chosen seems to be an optimal middle ground. We divide the bandwidth into four 32-bit buses running at 50 MHz. This keeps all of the parameters within reasonable limits: The bus speed, bus width, and fanout are well within the capabilities of TTL-level drivers. The pipeline cycle speed, 40 ns, is reasonable for low-cost programmable gate arrays. The physical size, 8-16 processors per PC board, is reasonable for a typical 9U-VME-sized module.

As shown in Figure 10, processors are grouped into clusters. Hit processors are grouped into four clusters of 8 CPUs. An eight-way merger merges 64-bit hit records onto a single 32-bit bus. Buses from four mergers feed data into a 4x4 switch that routes words to one of four output buses based on the lower 2 bits of their time stamps. Each output bus feeds a cluster of 16 track processors via a load-leveling buffer manager.

The switching elements are implemented in Field Programmable Gate Arrays (FPGAs). Commodity chips like the AT&T ORCA can implement wide data paths, complex logic, and multiple FIFO buffers on a single chip. The merger shown in figure 11, for example, is implemented in a single ATT 2C12 chip. It includes eight FIFOs and circuitry for inserting data fields into a 64-bit record.

The sorting switch is implemented using three FPGAs. Each chip is a 12-bit slice of the 32-bit data path. The switch, shown in figure 12, contains a 16-word FIFO at each intersection of four input buses with four output buses. On every 40 ns cycle, a word is transferred from each input to one of four FIFOs in its column. The destination FIFO is selected by the lowest two bits of the word's time stamp. Outputs of four FIFOs in each row are merged together onto a single bus and sent to the output pins.

The FIFOs are necessary to smooth out the data flow. Since it is possible for all four input buses to be sending data

Figure 10. Implementation of switching network

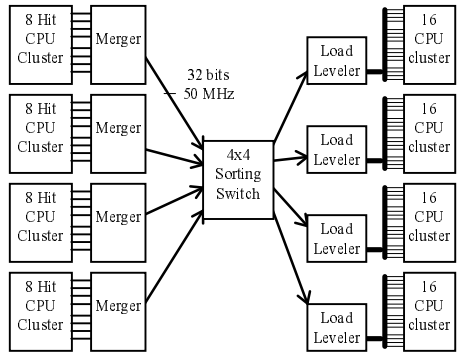


Figure 11. 8-way Merger Chip

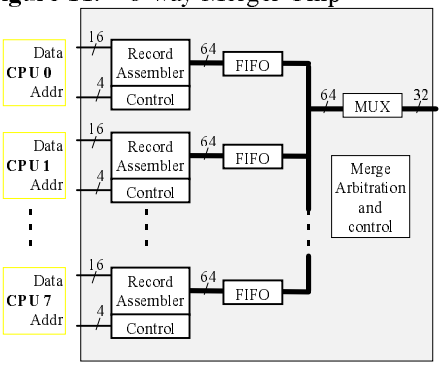
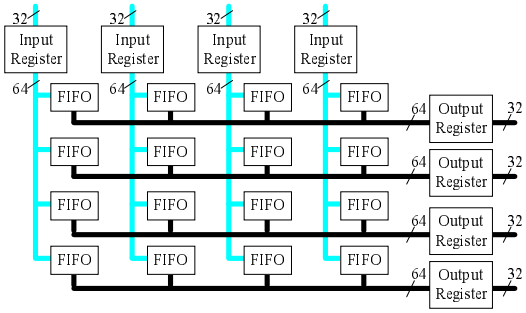


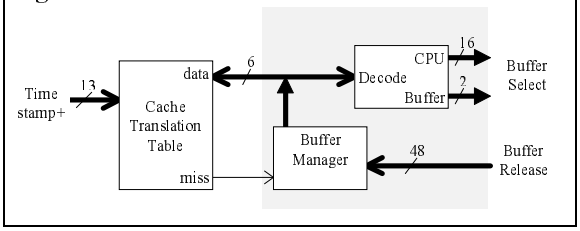
Figure 12. 4x4 Sorting Switch



to the same output bus, some of the data will have to wait. The FIFOs allow the input buses to continue sending data at the full rate as long as the average rate to all outputs is the same. The FIFOs are 16 words deep, so the switch can accept a sustained burst of 20 hits on each of four inputs directed at the same output before a FIFO overflows. Most of the burstiness is eliminated by using the lower two bits of the time stamp to select an output. A time stamp represents 132 nanoseconds. A given output will receive only the data from every fourth time stamp.

The load leveler (figure 13) performs a similar smoothing function. It keeps track of which processors are busy, and assigns new events to the least-busy processors. An event (interaction) is defined as all hit records with the same time stamp. The load leveler keeps a translation table that translates time stamps (and other sorting fields) to buffer addresses. When a hit record with a previously unseen time stamp is received, it causes a cache miss which causes the buffer manager to select a free buffer and write its address into the translation table. Subsequent hits with the same time stamp will all be directed to the same buffer. When a processor has finished processing an event, it releases the buffer so that the manager can use it for another event.

Figure 13. Load Leveler Circuit

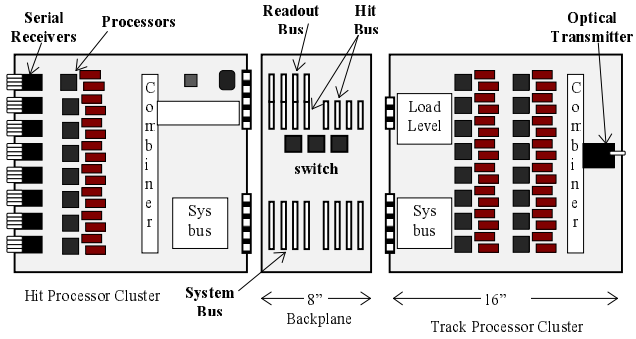


Each of the 16 processors in a cluster has three buffers for a total of 48 buffers. This allows a processor to be working on one event while two others are being transmitted. The buffer manager chooses processors based on the number of buffers in use. It fills all processors with one event before sending a second event to any processor.

Packaging

Clusters of 8 or 16 processors along with their merger and/or load leveler are put on a single PC board. These boards, shown in figure 14, plug into a switching backplane that contains the sorting switch as well as system bus and readout bus. The whole package of 8 processor boards and backplane make a box that is 8" x 16" x 16". This box is called a Phi-box, since it processes all of the tracks for one phi section.

Figure 14 Physical Modules of a Phi Box



Similar boxes are used to do vertex processing. As shown in figure 15, the trigger system is made up of 32 Phi-boxes and 4 Vertex-boxes. Track records from each of the four clusters within a Phi-box go to a different Vertex-box via a high speed optical link.

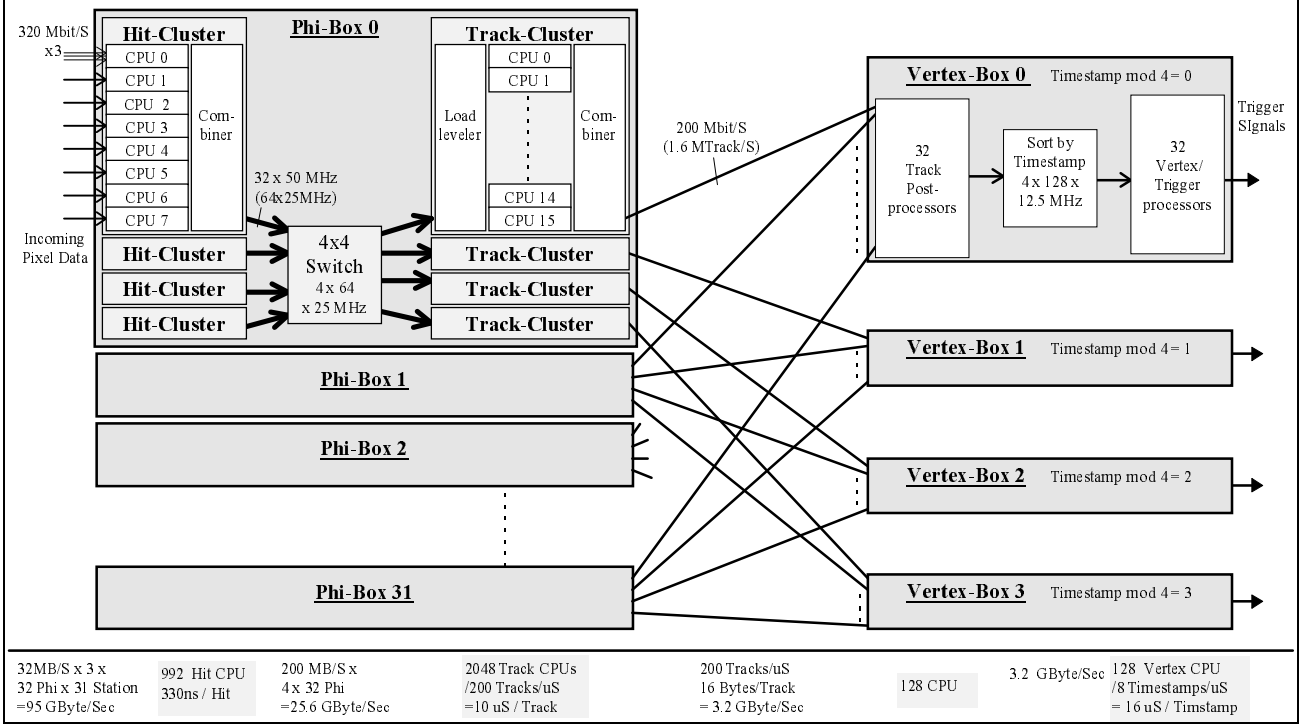
Prototype

A prototype phi-box is being built by the Fermilab Electronic Systems Engineering group. The prototype will include a single hit cluster and track cluster and a full 8-slot backplane with VME or PCI system interface. It will be a full working prototype, except that it will not have the serial links or readout bus. Work on this prototype began in March, 1995 and is expected to be completed by late 1996.

The microprocessors used in the prototype are from the MIPS 4600 family. They are the same type of processor found in the Silicon Graphics Indy and Challenge systems. They can execute 133 Million instructions per second and 44 MFlops. Since this particular processor, the Orion 4650, is targeted for the multimedia market, it will sell for under $50 per chip.

The choice of processor will be re-evaluated when work on the production system begins. Since microprocessors are doubling their performance every two years, there is a significant advantage in waiting as long as possible before committing to a processor. One likely candidate is a processor currently being proposed by Dario Crosetto [3]. His processor is specifically designed for physics trigger applications. Very long instruction words and multiple function units make it well suited for the type of processing done by high speed triggers. In addition, his design includes four processors on a single chip, making it possible to fit thousands of processors in a reasonable amount of space.

Figure 15. Block diagram of trigger system



Acknowledgments:

We express our thanks to J. Butler and H.H. Williams for their assistance and support. E. Barsotti, D. Crosetto, M. Johnson, S. Mani, and S. Shapiro have provided much helpful information.

REFERENCES:

* Paper presented by D. Christian at the Hiroshima Symposium on Development and Application of Semiconductor Tracking Detectors (October 10-13, 1995). An earlier version of this paper was issued as a Fermilab document, "FERMILAB-Conf-95/170, Univ. of Pa. Report UPR-230", and was published in the "Proceedings of the International Conference on Electronics for Particle Physics", LeCroy Corporation, May 1995.

† Work supported by U.S. Department of Energy under contract DE-AC02-76CHO3000 at Fermilab and partially supported under contract DE-ACO2-76-ERO-3071 at the University of Pennsylvania.

[1] S. Erhan, *et al.* (COBEX Collaboration)
"COBEX, A Dedicated Collider B Experiment"
Nucl. Inst. Meth. **A333** (1993) 101.

[2] J. Ellett, *et al.* (P238)
"Development and Test of a Large Silicon Strip System for a Hadron Collider Beauty Trigger"
Nucl. Inst. Meth. **A317** (1993) 28.

[3] W. Selove
"Impact Parameter Trigger and Vertex Detector for Forward Collider"
Proceedings of the Workshop on B Physics at Hadron Accelerators (Snowmass Workshop, 1993), Fermilab Report CONF-93/267, P. McBride and C.S. Mishra, editors.

[4] W. Selove and K. L. Sterner
"Data Transfer and Trigger Processor System for a High Rate Heavy-Quark Collider Detector"
Univ. of Pa. Report UPR-227E, August 1994.

[5] A suitable pixel plane design, with fast data-driven readout and with high precision coordinate determination, has been under development by S. Shapiro et al.:
S. Shapiro, D. Cords, S. Mani, et al.,
"Progress in the Design of a Data Push Architecture for an Array of Optimized Time Tagging Pixels"
SLAC Report SLAC-PUB-6249, June 1993.

[6] D. Crosetto
"Digital Programmable Level-1 Trigger with 3D-Flow Assembly"
Superconducting Super-Collider Report SSCL-PP-445, August 1993.

[7] D. Crosetto, private communication.

The authors can be contacted via email at husby@fnal.gov.