

A Fast, First Level, $R\phi$ Hardware Trigger for the D0 Central Fiber Tracker Using Field Programmable Gate Arrays

Brad Abbott¹, Bob Angstadt², Fred Borcharding², Marvin Johnson², Manuel Martin²

¹New York University,

²Fermilab, P.O.Box 500, Batavia, Illinois 60510

Abstract

An $R\phi$ trigger was developed using the eight doublet layers of axial fibers in the new Central Fiber Tracker for the D0 Upgrade Detector at Fermilab [1]. This trigger must be formed in less than 500 nsec and distributed to other parts of the detector for a level 1 trigger decision. The high speed is achieved by using massively parallel AND/OR logic realized in state-of-the-art field programmable gate arrays, FPGAs. The programmability of the FPGAs allows corrections to the track roads for the as-built detector and for dynamically changing the transverse momentum threshold. To reduce the number of fake tracks at high luminosity, the narrowest possible roads must be used which pushes the total number of roads into the thousands. Monte Carlo simulations of the track trigger were run to develop the trigger algorithms and a vendor specific commercially available simulator was used to develop and test the FPGA programming.

I. INTRODUCTION

Every 132 nsec one or more interactions will occur in the upgrade D0 detector producing from 10 to 100 charged particles. These particles travel radially outward and are detected in 8 barrel shaped double layers of scintillating fibers. A trigger for this system must sample the output of the fibers and recognize groups of fiber hits as real particle tracks above a transverse momentum threshold, P_t . The trigger must have a small latency since it must produce a result in under 500 nsec and it must be pipelined to be able to look at the tracks from each crossing. Also it must be capable of handling many channels because there are approximately 38,400 fibers in the trigger.

The trigger must also be flexible with respect to the realization of the calculated roads. The as-built detector will have systematic differences in the placement of the fibers due to construction constraints and random differences due to limitations in construction control. To meet these criteria field programmable gate arrays, FPGAs, are used to form the trigger logic. They can operate well above the 7 MHz data input rate, can process many channels in parallel, can be reprogrammed to reflect differences in fiber locations, and have propagation times in the 100 nsec range.

The detector is divided into 80 pie shaped wedges, called sectors. A sector, therefore, includes fibers from each layer. Every track above a P_t of 1.5 GeV is contained within two sectors. The 480 fibers from one sector of the detector are processed on a single PC board. Each sector is further divided into four equally sized wedges called cells, each of which contains 120 fibers. The track finding is done

independently within each cell. Because tracks curve cross the boundaries of cells, fibers from adjacent cells must be combined with those of the home cell to form a seamless trigger. The neighbor cell fibers needed for curved tracks down to 1.5 GeV add another 192 fibers giving a total of 312 fibers per cell. Note that of the 192 fibers, 96 come from each side of the home cell to accommodate both positive and negative curvature tracks. The fibers for one cell are input into a single FPGA and four FPGAs are placed on one PC board. To reduce the number of I/O channels, the 312 input fibers are time multiplexed 4:1 onto 78 input pins.

The central tracker has a solenoidal magnetic field which bends the charged tracks in the radial or $R\phi$ plane of the tracker. The transverse momentum of the tracks is proportional to the inverse of the radius of bend and one can calculate roads which tracks of a given initial ϕ direction and radial momentum, P_t , must follow. The realization of these roads in the FPGAs is conceptually very simple. If a given road passes through A layer fiber bin a and B layer fiber bin b and so forth out to H layer fiber bin h then the road can be programmed as an 8-fold AND in the FPGA as;

$$\text{Trig}[a,b\dots h] = A[a] \text{ AND } B[b] \text{ AND } \dots H[h]. \quad (1)$$

The number of equations needed depends on three factors. The first factor is the minimum P_t . The lower the P_t threshold the more the tracks bend so more roads are needed. The road number grows approximately as the square of the P_t . Lowering the P_t by a factor of 2 requires 4 times as many equations. The second factor is the size of the fiber bins. A change of the bin size by a factor of 2 results in a factor of 4 change in the number of equations. The third factor is any variation in radial or angular position of the fiber along its length in Z. When the radius and ϕ position of a fiber remains constant in Z, the fiber projects onto a circle in the $R\phi$ plane. When either of these varies the fiber projects onto some other more complicated shape which is larger than the simple circle. Also note that the fibers can then seem to overlap. The effect of these placement errors were studied. These studies indicated that for placement errors as small as $1/10^{\text{th}}$ the fiber diameter the number of equations needed to retain high efficiency is starting to be significantly impacted [2]. The number of equations greatly increases as the placement uncertainty increases.

II. FINDING TRACK CANDIDATES

The fiber trigger is formed in four stages. First the signals from the fibers are discriminated and distributed to the trigger logic. Second the individual fiber signals are formed into bins in each of the eight layers and compared to a list of roads to find track candidates. The third stage takes the track

candidates which were found in ϕ and Pt bins and loads the found tracks into four output buffers. Each word in this buffer contains the ϕ and Pt bin address for one track. In the fourth stage the four buffers from stage three are merged into one buffer and the information is sent to the other detector elements as required.

A. Doublet Finding

Each of the eight layers is made up of two single layers. One of the layers is staggered by $\frac{1}{2}$ a fiber relative to the other so that there are no gaps. All tracks pass through a fiber in the inner, the outer or both layers.

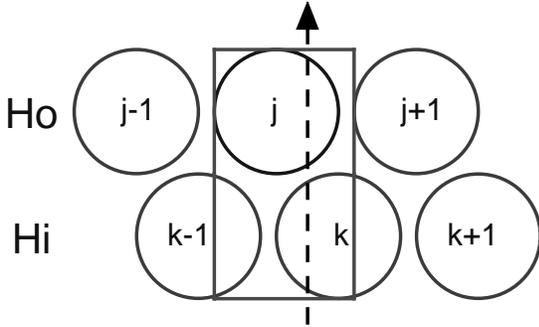


Figure 1: Cross sectional view of single layer. Hi is the H layer inner singlet layer, Ho is the H layer outer singlet layer. The box is the doublet bin and the dashed line represents a track segment.

The first part of finding a track is to form bins in each doublet layer by combining individual fiber hits in the inner and outer singlet layers. For this design one inner fiber is combined with one outer fiber forming a non-overlapping doublet bin one fiber wide.

The equation for doublet bin j is:

$$HL[j] = (\text{NOT}(\text{Ho}[j+1]) \text{ AND } \text{Hi}[k]) \text{ OR } \text{Ho}[j]; \quad (2)$$

The ‘OR’ combines the inner and outer layers. The ‘AND’ with the ‘NOT’ makes the bins non-overlapping. As stated above this design results in a bin one fiber wide, but with different logical combinations of the inner and outer fibers the bins could be as fine as $\frac{1}{4}$ fiber wide or as course as desired.

B. Track Finding

The doublet bins are combined to form a track by requiring a hit in each of the eight layers. A trigger requiring 8-of-8 can be used because the efficiency for each doublet layer is over 99%. We also determined the resources required to relax the track requirement to any n-of-8 and found that four times the number of FPGA logic cells are needed for all values of n less than or equal to 7.

The list of roads is found independently using both an analytical calculation and a special particle tracing Monte Carlo. In each case a table of bin numbers is generated. This table is then used by a text manipulation program to convert each row into an equation formatted in the hardware descriptive language, HDL, used to program the FPGAs.

These equations are of the form:

$$\begin{aligned} T1013172227323945 = & \text{AL}[10] \text{ AND } \text{BL}[13] \text{ AND} \\ & \text{CL}[17] \text{ AND } \text{DL}[22] \text{ AND } \text{EL}[27] \text{ AND} \\ & \text{FL}[32] \text{ AND } \text{GL}[39] \text{ AND } \text{HL}[45]; \end{aligned} \quad (3)$$

where the index numbers (10, 13, ...) depend upon the details of the design. In the next step the several terms that share the same A layer doublet number, here 10, and the same H layer number, here 45, are OR’ed together:

$$\begin{aligned} \text{Trig_a10h45} = & T1013172227323945 \text{ OR } T10\dots45 \text{ OR} \\ & \dots \end{aligned} \quad (4)$$

The resulting terms are then OR’ed together in groups that share the same H layer index but differing A layer indexes to form Pt bins. A straight line corresponding to an infinite momentum track drawn from the center of the detector and through the center of an H layer bin passes through just one A layer bin. That bin is defined as the zero offset bin for the H layer bin. The different Pt bins can then be defined with respect to their relative offset from the zero A layer bin. For the above example the zero A layer for an H layer bin value of 45 is 17. Therefore the offset for A layer bin 10 is 7. Eight Pt bins are formed, four for positive and four for negative curvature tracks.

There are eleven ϕ bins per cell and eight Pt bins per ϕ bin. Therefore the output from this stage is 88 terms from each of the four FPGAs. Each term is TRUE if a track was found or FALSE if it wasn’t. These output terms are multiplexed 4::1 onto 22 output pins on each FPGA for transmission to the next stage.

III. LOADING TRACK BUFFERS

As discussed above the track finding stage outputs a matrix of terms which is 44 long, corresponding to the 44 ϕ bins, by 8 wide, corresponding to the 4 Pt bins of each polarity. This array must be searched in decreasing Pt order looking for any entries that are TRUE. As each TRUE pin is found the ϕ and Pt bin addresses for that pin are loaded into a register. This is basically a serial problem which must be solved in parallel hardware. If it were done serially the process would take at least $44 \times 8 = 352$ steps. To get a result every crossing a serial processor would have to make 352 steps within the 7.6 MHz crossing frequency which requires a clock rate of over 2 GHz. Alternatively, the problem can be solved in a very short time using FPGAs utilizing a tree structure with many parallel branches. However, this method requires significant resources.

The solution is formed in 8 steps, 6 of which are done in the next set of four FPGAs. Each of these FPGAs works on 44 ϕ bins and 2 Pt bins. In the first step the 88 pins are grouped into sets of 4 ϕ pins which are input into truth tables. Each table has sixteen lines, one line for each possible combination of four binary inputs. Each of these lines creates a four deep buffer with the ϕ addresses of the TRUE inputs. It also inserts the number of TRUE pins into a counter. At this point it is not necessary to attach a Pt address to the buffer elements since they are all in the same Pt bin. Since the ϕ address of the four entries are adjacent it is also possible to only load in the two least significant address bits. Keeping the width of the buffer

elements narrow at this stage reduces the resources needed. After the first step there are 22 buffers each of which is 4 words deep holding from 0 to 4 ϕ addresses which are each 2 bits wide.

entries of both buffers are bit extended to hold the appropriate Pt bin address. Then the negative buffer is appended to form a single buffer. (Which is in reality a positive track and which a negative is of course arbitrary.) The output of this stage is one of four sets of buffers holding the results for two Pt bins.

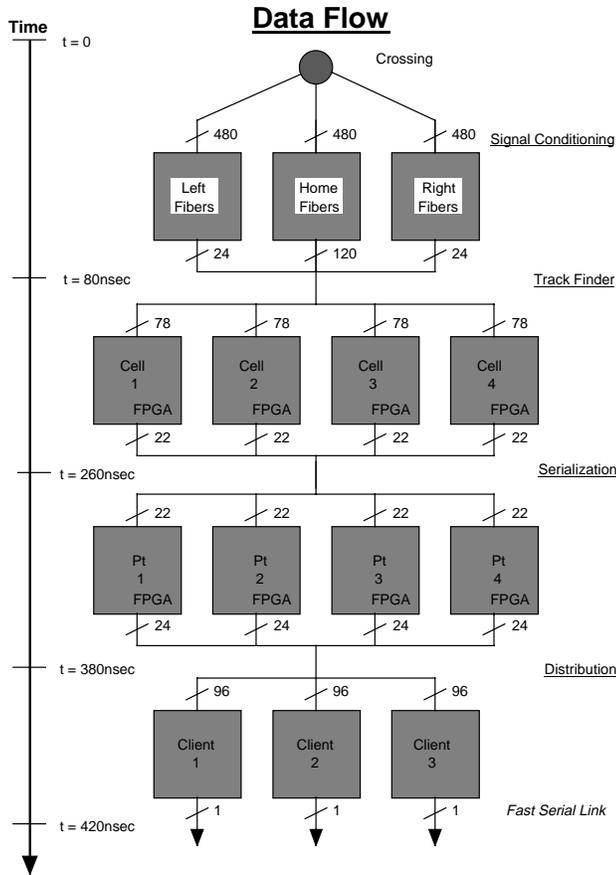


Figure 2: Flow chart for the level 1 trigger formation. The data flows from the top of the chart at $t = 0$, through each of the four stages of the trigger and is sent to the other detectors after about 420 nsec.

The next step combines the 22 four deep buffers by pairs into 11 buffers that are 6 deep. First the ϕ addresses are extended just enough to keep them unique. Then the occupied words of the first input buffer are moved into an output buffer. Next the remainder of the output buffer is filled with the contents of the second input buffer. The word count of the output buffer is the sum of the word counts of the two input buffers. The maximum buffer length is limited to 6 which means the information for any more found tracks is dropped from this step on. However, the track counter is 4 bits wide and counts to a maximum value of 16.

Geant Monte Carlo simulations of the upgrade detector have shown that even for the busiest events with multiple underlying interactions, the number of tracks found per cell is most probably zero or one and rarely is more than a few.

This process is repeated two more times and results in two six deep buffers. Of these final buffers one is for the positive value of this Pt bin and one is for the negative value. The

IV. DISTRIBUTION OF TRIGGERS

The fourth and final stage starts by combining the 4 buffers from the previous stage into a single buffer. It also pads the addresses of the buffer entries so that each entry is 16 bits wide. Next a header word is composed which gives information about the entries in the track buffer, the crossing number and other diagnostic information. The seven words of track information are then ready to be sent via fast serial link to other detector elements for a final level 1 decision. The data are also moved to a pipeline buffer of up to 32 crossings deep for read out to the level 2 and level 3 on a level 1 trigger accept.

V. HARDWARE

Data flow starts with a crossing as shown at the top of figure 2. Within the signal conditioning stage the VLPC[3] converts the photons from the scintillating fibers into an analog signal which is discriminated. The discriminated signals are then multiplexed in four time slices and sent to the next stage on the home board and across the back plane to the neighbor boards. The times four multiplexing is essential for the back plane signals in order to keep the number of connections to a manageable level. The times four multiplexing is also used for the transfer of the signals on the PCB between each of the following stages to reduce the number of on board traces and the pin counts of the FPGAs.

The finding of track candidates is done in a set of four FPGAs. Each FPGA is used to find the tracks in a single cell as explained above. The data is input in four time slices and latched inside the FPGA. Simulations of the gate array indicate the output becomes stable for all channels 160 nsec later and remains stable for another 110 nsec. Anytime from 160 nsec to 270 nsec after input the output can be multiplexed out to the next stage.

The track finder logic for one cell requires about 35,000 user gates in the FPGA, which is about 70% of the largest available chip from Altera Corporation, the vendor for which the simulations were run. Also it should be noted that the tracking simulation was done with a Pt threshold of 3 GeV. Changing the threshold to 1.5 GeV would require about 4 times the number of gates and would not fit into the hardware as outlined here.

The loading of the track buffers discussed in section III is done in another set of four FPGAs. In its simulation new data arrives every 132 nsec, the output data is good after 90 nsec and remains good for about 50 nsec. The logic requires about 30,000 user gates in each FPGA, which is about 60% of the usable gates.

VI. SUMMARY

We have developed a fast, pipelined trigger design which is based upon the use of large FPGAs. This design can look at the information from 38,400 channels and produce a list of found tracks giving their azimuthal address and Pt value. It finds the tracks for each 132 nsec spaced beam crossing in under 500 nsec. The design has been simulated in vendor software for specific FPGAs and meets our requirements on speed and flexibility.

VII. REFERENCES

- [1] Information on the D0 Upgrade detector can be found on the WWW at : <http://d0sgi.fnal.gov/hardware/upgrade/upgrade.html>. *Note that this link can be followed to other pages which contain links to PS versions of many of the following references.*

S. Abachi et al., "THE D0 UPGRADE," *FERMILAB-CONF-95-177-E*, Jul 1995. 54pp. Submitted to *International Europhysics Conference on High Energy Physics (HEP 95)*, Brussels, Belgium, 27 Jul - 2 Aug 1995.
- [2] Fred Borcharding, "Level 1 Trigger Design for the D0 Upgrade Central Fiber Tracker," *D0 Note* 2359, November 21, 1994.
- [3] R. Angstadt and Fred Borcharding, "D Zero Central Hardware Trigger Preliminary Implementation Studies of the 'Base Line Design' ", *D0 Note* 3058, August 15, 1996.
- [4] Alan Baumbaugh, Fred Borcharding, Marvin Johnson, Jesse Costa, Lourival Moreira, Sidhindra Mani, Steven Glenn and David Pellett, "Electronics Design Specifications for the D0 Upgrade Scintillating Fiber Detector with a Level 1.0 Trigger", *D0 Note* 2139, July 19, 1994
- [5] Fred Borcharding, "Level 1 Trigger Design for the D0 Upgrade Central Fiber Tracker," *D0 Note* 2359, November 21, 1994.
- [6] R. Angstadt and Fred Borcharding, "D Zero Central Hardware Trigger Preliminary Implementation Studies of the 'Base Line Design' ", *D0 Note* 3058, August 15, 1996.
- [7] Fred Borcharding, "A Study of the Effects of Fiber Placement Errors on the Level 1 CFT Trigger," *D0 Note* 2504, March 17, 1995
- [8] VLPC stands for "Visible Light Photon Counter" a solid state device which operates at cryogenic temperatures.
- [9] M. Adams et al., "A Detailed Study of Plastic Scintillating Strips with Axial Wavelength Shifting Fiber and VLPC Readout," *Nucl. Instrum and Meth.* {A366} 263 (1995) , *FERMILAB PUB-95/027-E*
- [10] Suyong Choi, "Characterization of Visible Light Photon Counters," *DPF'96*, August 10-16, 1996.