Implementation of the Content Addressable Memory (CAM) as an Encoder for the Silicon Track Card (STC)

Arvindh Lalam

Rm. 312, FAMU-FSU College of Engineering,
2525, Pottsdamer Street, Tallahassee, FL – 32310, USA
lalam@eng.fsu.edu
Reginald Perry

Rm. 341, FAMU-FSU College of Engineering,
2525, Pottsdamer Street, Tallahassee, FL – 32310, USA
perry@eng.fsu.edu

Abstract. The current paper details the design of the *hit word generator* for the Silicon Track Card (STC), by using the integrated Content Addressable Memory (CAM) of the Altera's Advanced Programmable Embedded MatriX (APEX) device. The *hit word generator* module receives a 46-bit word, *comparator word*, representing the output from a set of 46 comparators. The *hit word generator* can be described as an *encoder* with 46 inputs and 6 outputs. However, due to multiple active inputs in the 46-bit *comparator word*, a combinational logic *encoder* cannot be used. This paper introduces a new scheme that uses the Ternary CAM to efficiently encode multiple active inputs. This module also generates 32-bit words called '*hits*' for each of the encoded outputs.

1. Introduction

There has been extensive research directed towards investigating properties of the fundamental particles that constitute matter. In the forefront of this research is the TeVatron [1] collider located at Fermi National Accelerator Laboratory (FNAL), Batavia, Illinois, in which protons and anti-protons are accelerated to high energies and are then made to collide into each other. These collisions are analyzed to study the properties of sub-atomic particles and to detect presence of new particles. The D0 detector is a general-purpose detector that is part of the TeVatron complex. Though particle beams with high beam intensity are used in the TeVatron [2], a very small fraction of the proton anti-proton pairs actually collide and a still smaller fraction of these collisions result in events that are of interest to physicists. The number of rare events that are of interest, like the generation of the top quark, are in the order of one in 10 billion collisions. Thus, the objective of the detector is to identify these rare events among billions of events occurring every second during the course of collisions between the protons and anti-protons. This depends on how well the

detector eliminates unwanted events. In Run I of D0 collider that was carried out between 1992-1996 [2], events were recorded at a rate of 3.5 Hz from a total collision rate of 0.5 to 1.0 MHz. For Run II, the D0 detector is being upgraded to operate with a ten-fold improvement in beam intensity [2] and twenty-fold improvement in the amount of data collected [3]. This is accomplished with the addition of several tracking detectors such as central fiber tracker (CFT) and silicon micro-strip tracker (SMT) [2]. In addition to the tracking detectors, the D0 detector also contains fast electronics called *triggers*, divided into three levels. The SMT consists of rectangular silicon p-n junctions placed in four layers. On generation of an electron-hole pair due to the passage of charged particles, the charge is collected by the junctions also known as *strips* and deposited in an array of 32 capacitors [4]. These are connected to Analog-to-Digital Converter (ADC) and the digitized data is sent through an optical link [4] to the Level 2 and Level 3 D0 trigger. The Level 1 *trigger* of the D0 detector has an input rate of 7MHz and sends output at a rate of 10KHz to the Level 2. The Silicon Track Trigger of the Level 2 (L2STT) [5] identifies groups of charges, *clusters*, and reconstructs paths taken by the particles generated during the collisions. The STC finds the clusters and generates the hits. The *hit word generator* module is the part of STC that is responsible for generating the *hits*. These hits are used by the Track Fit Card (TFC) for reconstructing the path taken by the charged particles.

This project is a collaboration between researchers from the Department of Electrical and Computer Engineering, Florida Agricultural and Mechanical University - Florida State University (FAMU-FSU) College of Engineering, researchers from High Energy Physics (HEP), Florida State University and High Energy Physics, Boston University (BU). The current project is based on the specifications provided by the engineers at the Boston University (BU) and a preliminary STC module designed using the Very High Speed Integrated Circuit (VHSIC) Hardware Description Language (VHDL) [4]. The Altera Quartus II is used for synthesis, simulation, place and route and configuration of the device. Other third party Electronic Design Automation (EDA) tools are also used for synthesizing and optimizing modules written in VHDL.

2. D0 Trigger

The "triggers" associated with the D0 detector carry on the task of digitally sieving events for particular occurrences that are of interest to physicists. This system is divided into various levels, each of which performs event selection to some extent, thus decreasing the accept rate for next immediate level. Effectively, the output rate of the final level decreases to 50 Hz from a total collision rate of 7MHz. The time available to process events in each of the stages increases from the Level 1 to the Level 3, thus prospective events can be examined more closely at higher levels.

2.1. Level 1

Level 1 analyzes the data from tracking detectors, locates the *clusters* of energy and

identifies hit patterns[3]. The Level 1 contains 128 trigger bits, each of which is set on occurrence of a particular combination of trigger terms [2]. An example of triggering combination is a track candidate in the CFT having momentum more than a particular threshold [4]. If any of the 128 trigger bits are set, then the Level 1 sends a Level 1-accept signal to the Level 2 for subsequent processing. The output rate from Level 1 to Level 2 is 10KHz.

2.2. Level 2

The Level 2 improves the accept rate of the events by a factor of ten. This level has access to more refined information than the Level 1 and processes data in two stages. All the preprocessors from the Level 2 send data to the Level 2 global processor, which makes a decision of selecting or rejecting the events. The L2STT is organized into the Fiber Road Card (FRC), the STC and the Track Fit Card (TFC) [4] as shown in Fig 1. The FRC receives information from the Level 1 CFT and generates particle trajectory information understandable to the STC, called *roads* as shown in Fig 2 [3]. The data from various layers of the CFT are used to define a *road*, which passes through the SMT layers. The roads are sent to the STC, whose operation is discussed in detail in Section 3. The TFC uses eight TI-320C6203 300MHz Digital Signal Processors (DSP) for reconstructing the trajectory parameters using the data obtained from FRC and STC. Reconstructed parameters are then sent to the L2 Central Tracking Trigger (L2CTT) [3].

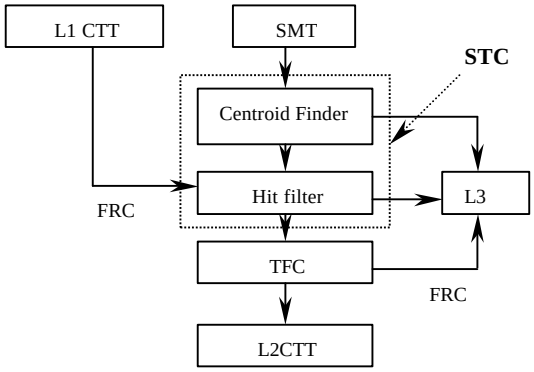

Fig. **1**. Block diagram of the STT

2.3. Level 3

Level 3 is the final level of the D0 trigger. Upon receipt of a Level_2 accept, Level 3 receives data from Level 1 and Level 2 modules for further processing. This stage is implemented in software unlike other levels and uses parallel fast processors to achieve the processing rate required [2]. Output rate of this final stage is 50 Hz. Output of Level 3 is stored on tape for later application [2].

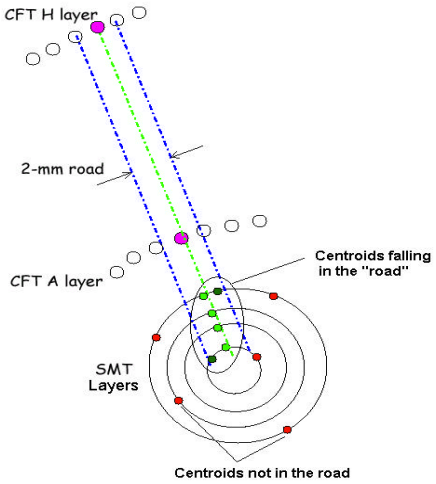

Fig. 2. Diagram showing a *road* and *centroids* [3].

3. Silicon Track Card (STC)

The STC card is part of the L2STT as shown in Fig 1. The STC card consists of *control logic* and eight STC modules, also called *channels*. The *control logic* designed by engineers at BU acts as an interface between the STC channels and the rest of the STT. The charges identified in the SMT are sent to the STC in digitized form, called *strip* information. The information sent by Level 1 CFT is used by the FRC to define *roads* each of which represents a path 2-mm wide as shown in Fig 2. The function of each STC is to organize the strip information into groups called *clusters* and finding the centers of these clusters called *centroids*. In addition, STC performs the task of identifying the *centroids* that fall into any of the roads. Fig 2 shows a group of clusters identified by STC and highlights the clusters that fall into the roads. The STC constitutes a main data path, miscellaneous memory block and L3 buffers as shown in Fig 3.

3.1. Main data path

The Main Data Path is indicated in Fig 3 as shaded regions. This has three major parts as described below:

Strip Reader Module. This module accepts the SMT strip information in the form of a byte stream arriving at a rate of 53MHz and formats it into an 18bit word. Input byte stream alternatively contains strip address and strip data along with error bits, which are set for erroneous bytes. Strip address and data bytes are stored along with respective error bits into an 18-bit *strip word*. These *strip words* are read out at a

frequency of 32 MHz. There is a possibility that some of the *strips* will be damaged or will need gain and offset compensation. Look Up Tables (LUTs) are used to identify bad strips and to perform gain and offset compensation for good strips. Bad channel LUT identifies damaged strips, while gain offset LUT holds the gain and offset compensated data for each strip. The valid data words thus obtained are stored in a FIFO for later use by the clustering module.

Centroid Finder Module. The *centroid finder* module organizes the strips into *clusters* and finds the center of the cluster, *centroid*. This module organizes strips such that the strip with the highest data is placed in the center while the strips immediately before and after this are arranged on either side in the same order. The clustering module can find either a five-strip or a three-strip cluster. An event can have more than one cluster depending on the type of strips and on the continuity of the strip addresses.

The centroid finder algorithm is implemented as an asynchronous module that takes the strip data from the *clusters* determined by the clustering module. This calculation is centered on the second strip. The final value has an additional two bits added for precision and is stored in a *centroid FIFO*. This module uses the following expressions to calculate the centroid for five-strip and three-strip clusters [4] with D1, D2, D3, D4 and D5 representing strip data:

$$\text{Five strip centroid} = \frac{-\text{D1} + \text{D3} + 2\text{D4} + 3\text{D5}}{\text{D1} + \text{D2} + \text{D3} + \text{D4} + \text{D5}}$$

$$\text{Three strip centroid} = \frac{-\text{D2} + \text{D4}}{\text{D2} + \text{D3} + \text{D4}}$$

Hit Filter. While the *centroid finder* module calculates *centroids*, the *hit filter* receives the road information in the form of digital words. The *road* information contains two words, upper-word and the lower-word, representing the two sides of a road. A comparator is designed to store the upper-word and the lower-word of a particular *road*. When a centroid is given as an input to the comparator, it checks to see if the centroid falls between the upper and lower words stored in it. A centroid that fits in a comparator is said to be falling in the *road* and is called a *hit*. A comparator that has a *hit* gives an output of '1'. Since only one *road* can be stored in a single comparator, 46 comparators are used to store a maximum of 46 roads. After all the roads are stored in the comparators, a centroid is read from the *centroid FIFO* and is given as an input to the *comparator* module. The *comparator* module gives out a 46-bit word that is directly sent to the CAM modules of *hit word generator*. The functionality of *hit word generator* is discussed in Section 5. The *Hit filter* module uses handshaking signals to find if *hit word generator* is still busy. After *hit word generator* writes the *hit words* of the centroid, *hit filter* reads the next centroid and this process continues until the *centroid FIFO* is empty. Since the STC card contains eight STC channels, a contention scheme is used by the control logic for reading out the hits from the channels.

3.2. L3 Buffers

In addition to clustering and finding centroids, the STC also needs to buffer intermediate information throughout the processing of an event. The buffering modules are selectively activated for the required channels. L3 buffer module also allows buffered data to be read selectively. Since there are a total of eight channels that process data, a contention scheme very similar to the one used for hit readout is used to control data transfer from L3 buffers. When an L3 block is ready for readout, the corresponding STC pulls up its *busy* signal and waits for data bus to become available. When bus becomes available, *block* signal is set high to block the bus from being used by other channels until whole block of data is read. Types of data that each of channels can store are hits, raw data, corrected data, strips of the cluster and bad strips.

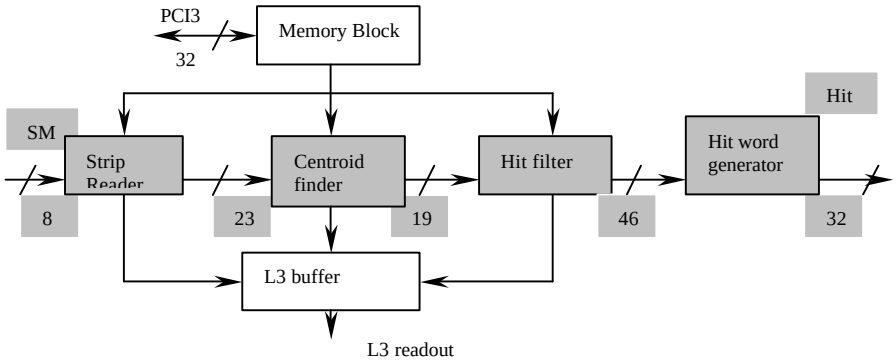

Fig. 3. STC and Main data path

4.Content Addressable Memory (CAM)

A Random Access Memory (RAM) block accepts an address of the memory and returns the content in a single clock cycle. However, for searching a word in the RAM, its has to sequentially check the addresses and thus takes a number of clock cycles. The Content Addressable Memory (CAM) is a type of memory that allows searching, by using the content of the memory rather than address of the memory [6]. For example, considering the instance shown in Fig 4(a), where the CAM contains data 4,7,8 and 1, gives an output of '1' when one of these words is given to the input. The number of clock cycles required for a CAM to search the content is small and constant, while the number of clock cycles a RAM requires to search is proportional to the number of memory words stored. CAMs are extensively used for applications that require reverse-lookup of the data. In addition, CAMs are also used for applications that require fast searching and matching of the data. While a simple CAM can hold logic levels of '0' and '1', Ternary CAMs can also hold *don't care*. A CAM containing *don't care*, also represented as a *X*, returns a match for all logic

levels. The ternary CAMs allow multiple words to be written into one location and thus require fewer memory locations to represent multiple words. The current project uses the integrated ternary CAM available in the APEX devices.

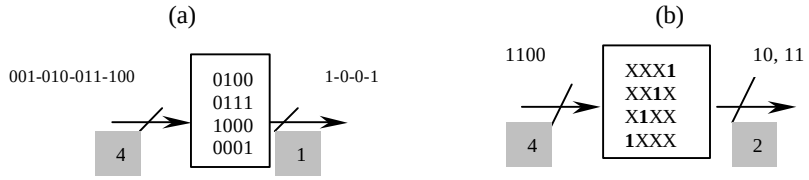

Fig. 4. **(a)** A Simple CAM memory block showing a match output; **(b)** Encoded output from a MASK CAM.

4.1 APEX CAM

CAMs were traditionally available as discrete components that can be externally connected to the core module. Since the external signals have to travel on the PCB, they invariably have a time delay associated with them. However, the integration of the CAMs to the Programmable Logic Devices (PLDs) drastically reduces the time delay and saves the board space on the PCB. In Altera's Quartus II, the APEX CAM is implemented by using the Altera's altcam megafunction [6] and the memory associated with the Embedded System Blocks (ESB) [7] of the APEX devices. The APEX CAM can be configured to accommodate any configuration between 32 X 5120 and 5120 X 32. The Quartus II software cascades ESBs to implement wider and deeper CAMs, however, wider CAMs cannot provide encoded output.

The APEX CAM can support don't cares [6] and thus allows designer to efficiently use the memory resources. The contents of the CAM can be written either during power-up or during the normal operation of the CAM. A memory initialization file(.mif) or a intel hex file can be used to initialize the memory during power-up. *don't cares* can also be written into the CAM memory using the internalization files. In the current implementation, an .mif file is used to initialize the memory. The APEX CAM can be used in three modes as detailed below:

Single Match Mode. In the single-match mode, the APEX CAM requires only one clock cycle to return the data location. However, the APEX CAM gives erroneous values in case duplicate data is present. In this mode, the CAM can accommodate as many as 32 words with 32 bits each [6] in one ESB on the APEX device.

Multiple-Match Mode. In the multiple-match mode, the CAM needs two clock cycles to return a valid data location. This mode supports duplicate data words, but can accommodate 32 words with only 31 bits in each word, in a single ESB [6].

Fast Multiple Match Mode. In this mode, the CAM takes only one clock cycle to return a valid address. However, the CAM can accommodate only 16 words with 32 bits in each word [6].

5. Implementation of Hit Word Generator

Hit word generator receives the 46-bit *comparator word.* Each bit in the comparator word corresponds to a comparator, which stores a *road* received from FRC. Thus, the position of a bit in the 46-bit word is indicative of the corresponding *road* number. Therefore, position of an active bit (bit holding value '1') when encoded, gives the number of the road that the *hit* corresponds to. This block encodes all the roads in which a centroid made a 'hit'. The 32-bit *hit word* containing the encoded road number, strip information of the centroid and details of this event is generated. The data format of *hit word* is shown in Table 1.

Table 1. Data format for the *HIT WORD*

	31..27	26	25..24	23	22..16	15..13	12..9	8	7	6	5	4	3..0
HIT	TRACK		DE/DX		SEQ ID	HDI	CHIP	STRIP					

TRAILER	11110		EVENT	No. of HITS	SERR	MM	RERR	EERR	

For the case where there can be only one comparator bit holding a value '1', a *selected signal assignment* in VHDL or a simple encoder, implemented using combinational logic, would suffice. However, more than one comparator bit is expected to hold a value '1'. In a sequential approach to this problem, each bit has to be sequentially checked for a *hit.* This solution when implemented in a Finite State Machine results in a huge overhead. In addition, the state machine cannot be stopped until all the 46 bits are checked, even if there are no hits in the later roads.

The APEX CAM working in m*ultiple-match mode,* can be efficiently used as an encoder in this context. The mask property of the APEX ternary CAM is used in this implementation. All the bits except the bit corresponding to the address of the memory word are written as *don't cares*. For example, in the case of an 4 x 4 CAM in *multiple-match mode*, shown in Fig 4(b), the written word for address 3 is "1XXX" and written word for address 2 is "X1XX". A logic level '1' in either the bits 3 or 4, will result in a match, irrespective of other bits. Output shown in Fig 4(b) is encoded, thus matching address locations can be read in encoded format. In this mode, CAM effectively encodes the *multiple active input* and reads out the encoded words.

In case of a 46-bit comparator word, using one 46-bit CAM will stretch the APEX CAM width beyond one ESB width and output will not be available in encoded format [6]. In order to overcome this problem, 46 bits are split up between two CAMs as shown in Table 2. Implementation of this scheme is shown in Fig 5. This will require only two ESBs and output can be obtained in encoded format from each of CAMs. The *hit word generator* first starts reading output from the CAM_31 until there are no more matches. After reading the CAM_31, *hit word generator* starts reading the output from CAM_15 and adds "011111" to its output to compensate for the offset.

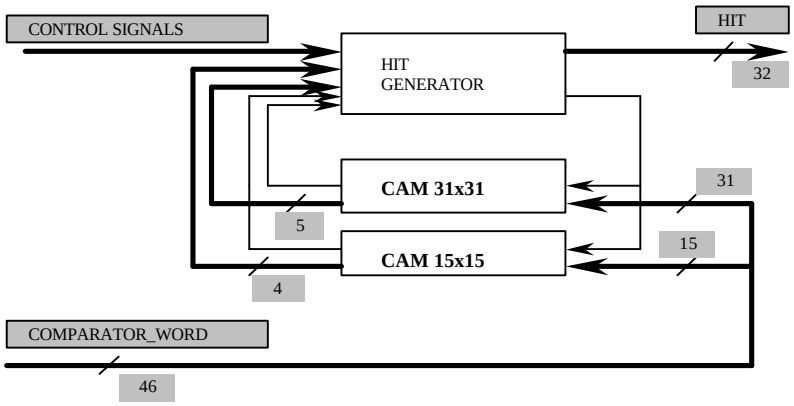

Fig. 5. Hit Word Generator using two APEX CAMs

Table 2.. Distribution of 46 bit word across two CAMs

		45 44 43 42 41……..28 29 30 31	30 29 28 27 26 ……..4 3 2 1 0
			30----0
0	**0**	x x x x x . x x x x x	x x x x x . x x x x x 1
1	**1**	x x x x x . x x x x x	x x x x x . x x x x 1 x
2	**2**	x x x x x . x x x x x	x x x x x . x x x 1 x x
		x x x x x . x x x x x	x x x x x
		x x x x x . x x x x x	x x x **CAM 31** x x x
		. . . .	. . . 30…0 . . .
		x x x x x . x x x x x	x x x x x
		x x x x x . x x x x x	x x x x x
28	**28**	x x x x x . x x x x x	x x 1 x x . x x x x x x
29	**29**	x x x x x . x x x x x	x 1 x x x . x x x x x x
30	**30**	x x x x x . x x x x x	1 x x x x . x x x x x x
		14 ------0	
3	**0**	x x x x x . x x x 1	x x x x x . x x x x x x
1	**1**	x x x x x . x x x 1 x	x x x x x . x x x x x x
3		x x x x x x	x x x x x . x x x x x x
2		x x x x x x	x x x x x . x x x x x x
		x x x **CAM 15** x x	x x x x x . x x x x x x
		. . . 14….0 . .	. . . .
		x x x x x	x x x x x . x x x x x x
		x x x x x	x x x x x . x x x x x x
43	**12**	x x 1 x x . x x x x x	x x x x x . x x x x x x
44	**13**	x 1 x x x . x x x x x	x x x x x . x x x x x x
45	**14**	1 x x x x . x x x x x	x x x x x . x x x x x x

After the last hit is written into the hit FIFO, hit word generator writes a trailer. From Table 3, it can be observed that performance of the block that uses CAM is far greater than that of the sequential block. The performance of the blocks is compared in terms of the number of clock cycles taken by each module.

Table 3. Chart showing clock cycles taken by *hit word generator*.

	6 roads		46 roads		6 distributed roads	
	not EOF	EOF	not EOF	EOF	not EOF	EOF
With CAM block	10	13	50	53	10	12
Sequential search	32	42	232	242	150	160

6. Conclusion

The STC module was successfully tested at an operating frequency of 32 MHz with the new *hit word generator*. As observed in Fig 5, usage of CAM for the block improved the performance of the *hit word generator* by 3 to 15 times. The time taken by the complete module has decreased to around 70% with the usage of the CAM.

Acknowledgements

We thank The United States Department of Energy and the National Science Foundation for supporting this work. We thank Horst Wahl, Physics Department, Florida State University, for his continuous support and encouragement. We thank the engineers at Boston University for helping us with the hardware test. We also thank Altera Corporation for providing the software and the chips for the prototypes.

References

[1] Fermi National Accelerator Laboratory, http://www.fnal.gov/pub/about/whatis/history.html, December 12, 2001.

[2] G.C. Blazey, "*The D0 Run II Trigger*", D0 collaboration, Department of Physics, Northern Illinois University, DeKalb, Illinois.

[3] W. Taylor, "*An Impact Parameter Trigger for the D0 Experiment*", D0 collaboration, State University of New York, Stony Brook, NY.

[4] S. Lolage, "*VHDL design and FPLD Implementation for Silicon Track Card*", Masters Thesis, Florida State University.

[5] W. Taylor's talk, "*The D0 Silicon Track Trigger*", International Electrical and Electronics Engineers National Science Symposium, 2000, Lyon, France.

[6] Altera Corporation, "*Implementing High-Speed Search Applications with Altera CAM*", 101 Innovation Drive, San Jose, CA.

[7] Altera Corporation, "*APEX 20K: Programmable Logic Device Family*", 101 Innovation Drive, San Jose, CA.