System Architecture for CMS L1 Tracking Trigger and work plan for Vertical Slice System Demonstration

by the L1 Track Finding Group

Abstract

In this document we will describe a track trigger architecture and system demonstration as a proposal for Phase 2 tracking trigger R&D for CMS. This will be a living document, serving multiple purposes. It is intended to define the tracking trigger project and organize the efforts towards the Technical Proposal, the Vertical Slice Demonstration System, and the TDR (Technical Design Report). It will be written in such a way to make it easy not only for new groups to learn, but also to communicate with all the relevant tracker, trigger and physics groups.

Contents

Executive summary

A key goal of the CMS HL-LHC upgrade is to maintain physics acceptances for all basic objects (leptons, photons, jets and MET) at the Level 1 (L1) Trigger level. As such, the introduction of a L1 Tracking-Trigger is of strategic importance. Consequently, the design of the Phase-II CMS Tracker must allow for an effective implementation of the tracking trigger. Although the construction of the Phase-II Tracker will take many years, it's design must be finalized soon. A silicon-based L1 tracking trigger has never been realized at this scale and thus it is imperative that the feasibility of the trigger be demonstrated before the design of the Phase-II Tracker is finalized. Silicon-based Level-2 tracking trigger systems were successfully implemented in the past and are being actively explored in the present. Experiences of these systems will serve as useful inputs in the design of the CMS L1 tracking trigger, however the higher occupancies anticipated in HL-LHC operation and the low latencies required at L1 (about $10\ \mu$s) present us with a unique set of challenges.

Motivated by these challenges, several CMS institutions have carried out a focused R&D program to advance the state-of-the-art in hardware-based pattern recognition and track reconstruction. We have attempted to address the issues of occupancy and latency by developing a "full-mesh" ATCA data formatting system, higher density AM chips and new algorithms for hardware-based track finding based on FPGAs. The long-term goal of this R&D effort is to develop these critical technologies to the point where we can ultimately propose them as a viable solution to the problems of HL-LHC L1 track triggering. Given the progress made by the R&D program in the last few years, we believe it is now time to take the important next step of establishing a Vertical Slice Demonstration System. This system will comprise a full tracking trigger path and will be used with simulated high-luminosity data to measure trigger latency and efficiency, to study overall system performance, and to identify potential bottlenecks and appropriate solutions.

The full-mesh ATCA architecture we have proposed for the CMS L1 tracking trigger permits high bandwidth inter-board communication. The full-mesh backplane is used to time-multiplex the high volume of incoming data in such a way that I/O demands are manageable at the board and chip level. The resulting architecture is scalable, flexible, and open and it enables us to undertake an early technical demonstration using existing technology. The ATCA architecture will allow us to explore and compare various pattern recognition architectures and algorithms within the same platform. Given that Advanced Mezzanine Card (AMC) specifications are designed to work with both ATCA and microTCA, the architecture naturally allows for the long-term integration of Tracker DAQ (AMC based) and tracking trigger activities.

The proposed architecture and system demonstration has been well received in the Phase-II Tracker Upgrade community and we are now working to better define the concept. In this document we will describe the tracking trigger architecture and system demonstration as a work plan we propose for CMS Phase-II R&D. This *living document* is intended to define the tracking trigger project and to organize efforts leading to the Technical Proposal, the Vertical Slice Demonstration System and the Technical Design Report. The proposed architecture for the system demonstration, however, is by no means the final system. As stated above, it is meant to demonstrate technical feasibility of a L1 Tracking Trigger with the state of the art technology and is possible to be implemented in 2 to 3 years from now. It will identify the issues and help find the solutions, guiding the community to gain experience towards the final system.

1   Introduction

The physics reach of the HL-LHC experiments will depend critically on the ability of their trigger systems to discriminate between interesting rare events and background. For example, the CMS muon trigger will reach an unacceptably large trigger rate at high luminosity due to the number of hits in the muon detectors. The trigger rate can be reduced to an acceptable level if tracks can be found in the inner detector and matched to the muon candidate. In order to be viable at Level 1, such a trigger decision would need to have a latency of the order of a few microseconds or less and therefore, despite the progress in computing technology expected for the next few years, given the complexity of the problem and the extremely crowded event structure, will need to be implemented mostly using specialized hardware.

Hardware-based pattern recognition for fast silicon-based triggering on charged tracks was first developed for the CDF Silicon Vertex Trigger (SVT) at the Fermilab Tevatron in the 1990's. The method used there [1] was based on a massively parallel architecture - the Associative Memory - to identify patterns efficiently at high speed, and has provided an effective solution to fast track triggers in a hadron collider environment. The Associative Memory approach was successfully used in CDF at trigger Level 2 and the same approach is now being implemented for Atlas (FTK), also at Level 2, albeit with a much improved hardware architecture implemented with modern technology. Since the Associative Memory approach is so far the only proven approach to tracking trigger in a hadron collider environment, it is chosen here as the baseline in our R&D program for CMS. However, the design of an Associative Memory system capable of dealing with the much higher complexity of the HL-LHC collisions, within the much shorter latency required by Level 1 triggering, poses significant technical challenges. An aggressive R&D program is currently ongoing to demonstrate its feasibility as well as to explore other approaches to the pattern recognition problem.

Track reconstruction typically consists of two steps: pattern recognition followed by track fitting. Pattern recognition involves choosing, among all the hits present in the detector, those hits that were potentially caused by the same particle. This stage produces a set of hits of interest. Track fitting involves extracting track parameters from the coordinates of the hits of interest. When time constraints are not so stringent, track reconstruction is implemented in software, often using processors running in the upper levels of a data acquisition system. However, software algorithms running on standard CPUs are typically not fast enough for these extreme applications. As will be described below, in the traditional Associative Memory approach, the pattern recognition stage is performed by Associative Memories, while track fitting is done using a simplified least squares fitting algorithm using a linear expansion of the analytical expression of the track trajectories around the hit locations in the detector, running on FPGAs. In what follows we will assume this approach as our baseline but we will make sure that the architecture we are proposing lend itself to testing and comparing other possible solutions.

1.1   The Challenges for tracking trigger at L1

Current estimates show that only a few microseconds will be available for track finding and fitting at Level 1. Two difficult challenges one has to face are data dispatching and pattern recognition. Data dispatching is where the stubs from many thousands silicon modules must be organized and delivered to the appropriate eta-phi trigger towers. Due to the finite size of the beams luminous region in z and the finite curvature of charged particles in the magnetic field, some stubs must be duplicated and sent to multiple towers in an intelligent way. This is especially challenging for Level 1 track trigger. Since all this must be done within a very short time (of the order of a microsecond). Communication between processing elements in different towers requires very high bandwidth and very low latency.

To get a feeling of the complexity of the pattern recognition task, we can use the Atlas FastTracK (FTK) [10] project as an example. Since the design requirements of the FTK system are now known from extensive simulations, numbers from FTK, in some case, can be used as an order of magnitude estimate for CMS. The original CDF SVT system, in operation from 2001 to 2005, had a total of 384,000 Associative Memory patterns, while the ATLAS FTK system for the Level 2 trigger effectively requires about 1 billion patterns in order to handle a luminosity of $3 \times 10^{34} \ cm^{-2}s^{-1}$ [10]. This is three orders of magnitude more Associative Memory patterns than the SVT. The Level 1 Track Trigger upgrade for CMS has the advantage of having the p_T stub finding done upstream, therefore it may not require as many patterns as that for FTK. However, the pattern recognition engine has to run at much higher speed to be usable at Level 1. It should be noted that larger available pattern density in the hardware is beneficial as it reduces some difficulties associated with track fitting, thus leading to lower overall latency.

In addition to the challenges above, extremely fast and effective track fitting is also required. Extensive R&D and experimentation of innovative ideas is needed in this area.

1.2 On going R&D activities at CMS

The architecture we have recently proposed for the CMS L1 tracking trigger system is based on ATCA with full-mesh backplane. The large inter-board communication bandwidth provided by the full-mesh backplane is used to time multiplex the high volume of incoming data in such a way that the I/O bandwidth demands are manageable at the board and chip level, making it possible for an early technical demonstration with existing technology. The resulting architecture is scalable, flexible and open. For example, it allows different pattern recognition architectures and algorithms to be explored and compared within the same platform. Also, given that AMC specifications are designed to work with both ATCA and MicroTCA, this architecture allows a natural long term integration of TK-DAQ (AMC card based) and TK-TRIG (ATCA based).

The proposed architecture and system demonstration concept has been well received within the tracker Phase 2 upgrade community and work is in progress to better define the concept. At the same time, establishing international collaborations within CMS to work on this project is essential. One of the main activities in the coming year (FY14) will consist of extensive simulation efforts, by physicists, to establish technical specifications based on Phase 2 physics goals. At the same time, students and postdocs from all groups will be offered a unique opportunity to develop hardware experience by getting involved with the design, construction and commissioning of the vertical slice demonstration over the next few years. Some of the groups, for the longer term, are also interested in getting involved with the development of the algorithms for the post-track-finding stages and of the interfaces with the global trigger.

This document presents a proposed architecture to demonstrate the track reconstruction at L1 for CMS. As already mentioned above, the proposed demonstrator is by no means the final system and serves the purpose of demonstrating feasibility and identifying bottlenecks of the L1 tracking trigger. This will be a living document and is intended to define the tracking trigger project and organize the efforts towards the Technical Proposal, the Vertical Slice Demonstration System, and the Tracker Technical Design Report (TDR). It will be written in such a way to make it easy not only for new groups to learn, but also to communicate with all the relevant tracker, trigger and physics groups.

2 CMS Track Trigger System Overview

2.1 Overview

The extremely high luminosities foreseen at the HL-LHC pose many unique challenges to a possible track trigger system based on silicon detectors, this is especially true at Level 1. As mentioned earlier, the total Level 1 trigger latency has to stay within about 10 microseconds, with only a few microseconds or so available for track processing. This includes data transfer, trigger tower formation, pattern recognition, track fitting and any necessary further processing such as, for example, vertexing.

Many unique challenges must be faced at the different stages of the processing chain: first, data need to be transferred out of the tracker at the necessary speed, stubs from thousands of silicon modules must be formatted, organized into $\eta - \phi$ trigger towers, duplicated and shared across tower boundaries as needed, then we need to perform pattern recognition and track fitting, and finally process all the tracks reconstructed by the previous stages to form an intelligent trigger decision. A coherent system design for a Level-1 track trigger will include all these aspects.

In this document we will make the working assumption that there will be a total of 15K detector modules/fibers, each fiber with 3.25 Gbps payload bandwidth capability. The detector will be partitioned into 48 trigger towers, 6 in η and 8 in ϕ. Each trigger tower will therefore handle 312 modules/fibers on average. The cabling of the modules will need to be optimized for trigger requirements. For simplicity, we will assume that the FEDs are upstream and receive the fibers from the modules and pass the relevant data to the track trigger system even though the architecture could allow the FED to reside in the same ATCA shelf as the track trigger data input boards on dedicated AMC ATCA carrier boards. The focus of this document is the Vertical Slice Demonstration System, not the DAQ readout, so FED details are not discussed here (they belong to TK-DAQ). The FED interface will need to be defined for demonstration purposes, even though the actual FEDs do not have to be involved in the demonstration.

2.2 Tracker-Trigger interface

2.2.1 Block Synchronous Data Transfer scheme

The found stubs are sent from the modules using a block synchronous data transfer scheme, which tolerates random occupancy fluctuations while bonding latency. The current plan is to have the data from 8 consecutive beam crossings as one block. The front-end designers are still investigating different format variants for robustness against rate fluctuations, ease of implementation, impact on power consumption, etc. While choosing the 8 crossings scheme as our current working assumption, our strategy is to design the downstream components to be flexible enough to handle different possible formats. There are many studies we still need to do to understand the system requirements far beyond data transfer.

2.3 Track Trigger System Architecture

2.3.1 Tracker geometry and Trigger Towers

Detailed studies have been done for the BE tracker geometry with different trigger tower partitions, and the 6 (in η) x 8 (in ϕ) = 48 trigger tower partition has been chosen as the default baseline configuration (see Figures 1 & 2).
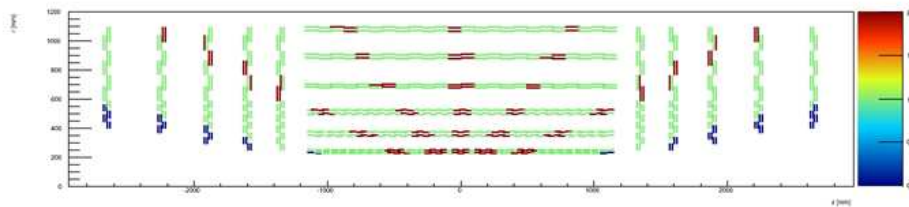
Figure 1: Six sectors in η. Note that the symmetry around $\eta = 0$ will provide for easier cable grouping

Stubs from the 15K silicon modules must be delivered to the correct trigger towers. Due to the finite size of the beam's luminous region in z and the finite p_T curvature of charged particles in the magnetic field, some of the stubs, coming from the neighborhood of some tower boundary, must be delivered to multiple towers to avoid efficiency gaps. Detailed studies have been performed on data sharing assuming the default 48 tower partition with

Figure 2: Eight sectors in phi

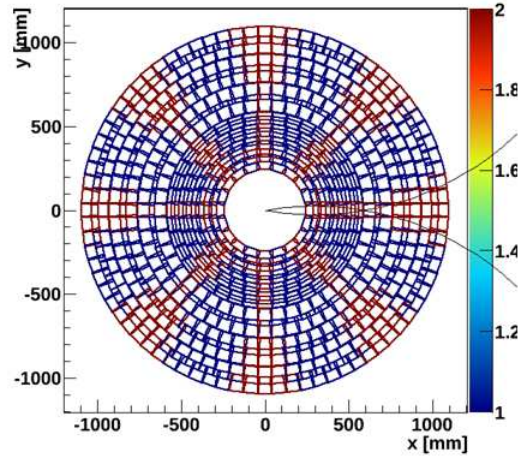a minimum p_T of 2 GeV and track origin smearing in $z \pm 7$ cm. Figure 3 shows the number of trigger towers that stubs from a given module should be delivered to under these conditions. When a stub is in the middle of the trigger tower, it will have to be delivered to only one tower (to the native trigger tower). When a stub is at the boundary in phi or eta (but not both), it will have to be delivered to two towers. If a stub is at both the boundaries in eta and phi, it will have to be delivered to four towers. Note that four towers is the maximum number of towers any stub must be delivered to.
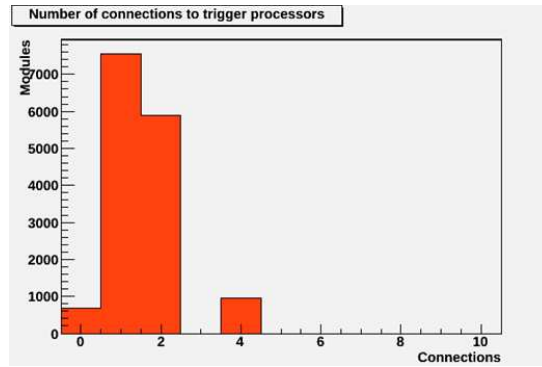

Figure 3: Distribution of the number of trigger towers each module needs to be connected to. Entries at zero are from modules that do not participate in triggering

2.3.2   Formation of Trigger Towers

The phase II tracker is being designed and optimized keeping in mind the trigger requirements. The resulting formation of the 48 Trigger towers is shown in Figure 4, where the lines indicate all needed interconnections among the trigger towers. The unique feature of this arrangement is that any given trigger tower only needs to be connected and share stubs with its immediate eight neighbors. This inter-connection structure will be used as the basis of the proposed trigger system architecture. Detailed studies have shown that this architecture is not sensitive to the variations of minimum Pt threshold, nor to the track origin in z (studied up to +- 15 cm).

Our hardware design process followed a bottom up approach whereby we studied various track trigger architectures by first studying the trigger tower formation and data sharing needs. Please note that this track trigger architecture for CMS L1 is rather different from that of FTK for ATLAS L2. In the case of FTK, the ATLAS tracker was not designed for track triggering purposes, and the cabling was optimized for readout and not for trigger. The data sharing between processing nodes is largely asymmetric and highly dependent upon upstream cabling and detector geometry. FTK chooses 64 trigger towers (16 in ϕ and 4 in η), and the inter-connections among the trigger towers are much more complex as a result, see Figure 5 for comparison.
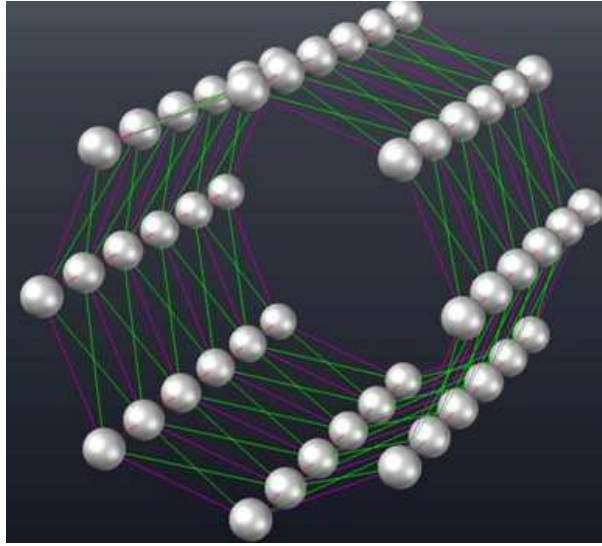
7

Figure 4: Conceptual view of the proposed CMS phase II L1 tracking trigger towers. The formation is organized as 48 trigger towers (6 η x 8 ϕ). Because the phase II tracker is being designed for tracking trigger purposes, it is possible to arrange the towers in such a way that data sharing only requires communication with immediate neighbor towers. Each node in this diagram represents a trigger tower processor engine. Within each processor engine crate the full mesh backplane is used for time multiplexing of the incoming data, while the data sharing between towers is handled with inter-crate fiber links.
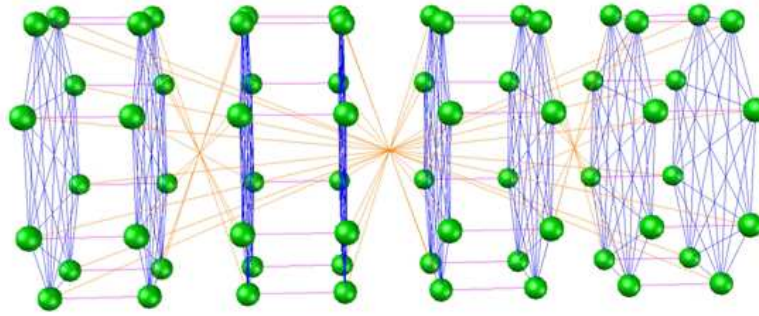

Figure 5: Conceptual view of the ATLAS L2 Fast Tracker (FTK) Data Formatter architecture. The FTK system is organized as 64 trigger towers (4 η x 16 ϕ), represented here by the green nodes. Because the existing silicon tracker was not designed for triggering, the data sharing among trigger towers is complex as indicated by the lines connecting the nodes. This requires the use of a full mesh backplane (shown in blue) for data sharing. Orange lines represent inter-crate links.

2.3.3 System architecture

The tower processor platform must support large numbers of fiber transceivers, which are used for receiving input links and sharing data between neighboring towers. A flexible, high bandwidth backplane is also required to quickly transfer data between boards. The boards should be large enough to support pattern recognition engines and fiber connections. Given these requirements, we conclude that a full mesh 14 slot ATCA shelf is a natural fit for the tower processor. If we were building the system today using existing technology, we would propose a L1 Tracking Trigger system comprised of one ATCA shelf per trigger tower with possibly an additional shelf acting as a second stage processor, as shown in Figs. 6 and 7 . Connections between tower processor shelves are limited to eight nearest neighbors. With the technology quickly progressing, we expect that the size of the system can be significantly reduced in the future.

The fundamental processing element is a pattern recognition mezzanine (PRM) card shown on Fig. 8 which performs both track finding and fitting. Time multiplexed data transfers into several parallel PRMs can reduce bandwidth requirement to manageable level. PRM's using different approaches to track finding and fitting may be tested and compared within the same overall high-level system architecture and data dispatching scheme.

An ATCA shelf is typically an air-cooled 13U rack mounted chassis consisting of 14 slots. The first two slots
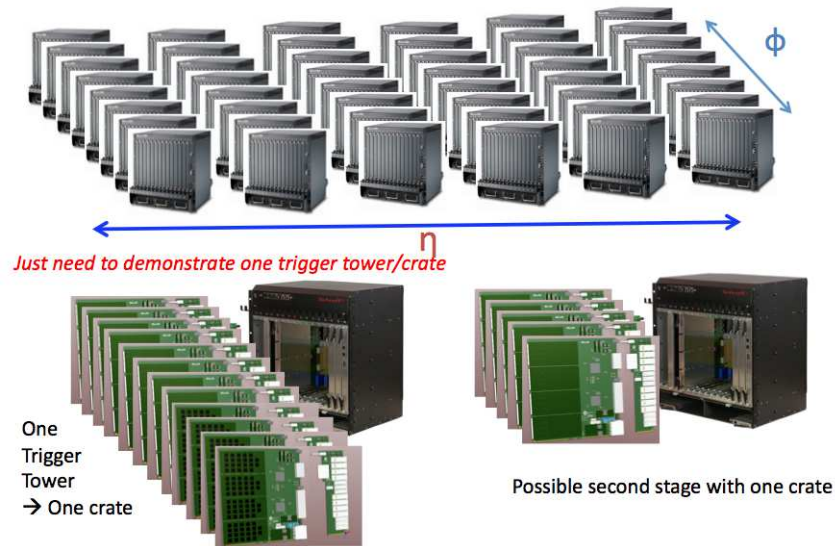
Figure 6: Possible system configuration with today's technology (smaller in the future)

48 tower inputs per time slice
each board could see the whole event
Track Duplication removal
More boards can be used to share data
Track Pt with MET related ...
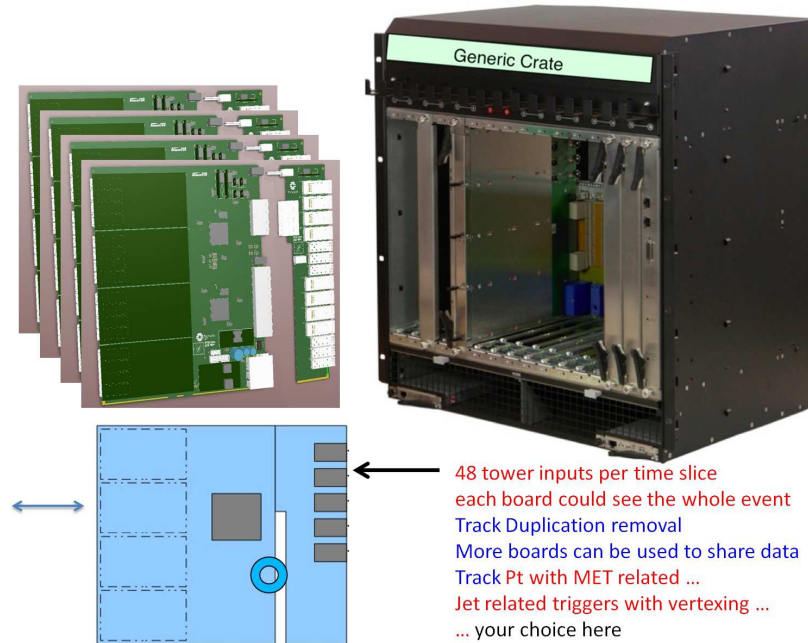Jet related triggers with vertexing ...
... your choice here

Figure 7: The second processing stage shelf.

are reserved for Ethernet switch blades. Switch blades may include a fast CPU and are often used for controls and other system functions. The remaining 12 slots are used for processor or payload blades. In a full mesh ATCA backplane each pair of slots is directly connected with a multi-lane bidirectional serial channel capable of supporting sustained 40 Gbps data transfers. A modern "40G" full mesh ATCA shelf has a total aggregate bandwidth of over 7 Tbps, not including external I/O.

The generic processor blade is shown in Figure 9. The front board measures 8U x 280mm and is designed around a single FPGA. This FPGA connects directly to the full mesh backplane fabric, mezzanine cards, and fiber transceivers located on a rear transition module (RTM). For the most part communication channels are high speed serial point to point links and are directly supported by SERDES transceivers in the FPGA.

2.3.4 Architecture Flexibility

A major advantage of the full mesh backplane is that it effectively blurs the distinction between boards, thus enabling system architects to experiment with different shelf configurations. Components can be roughly categorized
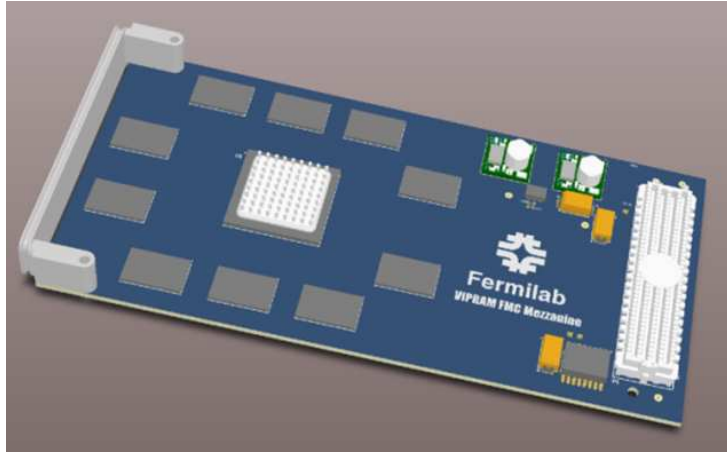
9

Figure 8: Concept of a pattern recognition mezzanine for testing different pattern recognition algorithms.
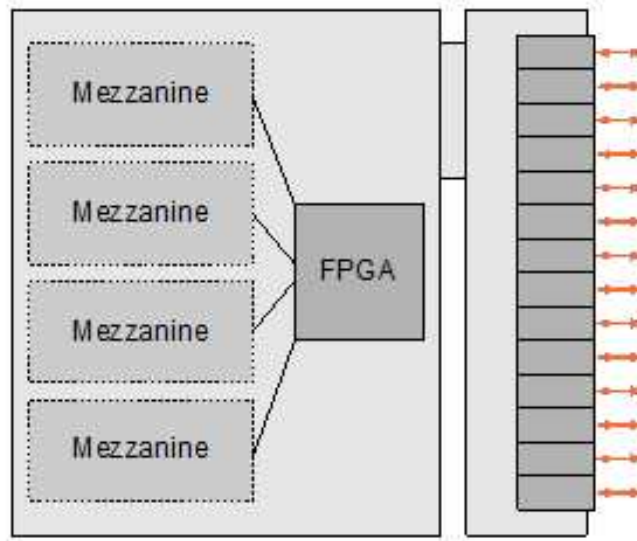

Figure 9: Generic processor blade

into functional blocks such as input link receivers, pattern recognition engines, inter-crate gateways and output formatters. While these functional blocks usually represent different boards in the crate it is important to note that the architecture is flexible enough to accommodate multiple functions in a single board. In the following sections we briefly illustrate two kinds of tower processor systems made possible by the flexibility of the full mesh architecture.

N DIB and M PRM configuration ($N + M \leq 12$) The most straightforward tower processor architecture consists of N data input boards (DIB), which receive input links and perform zero suppression. A DIB may be built using the generic ATCA processor blade (Figure 9) if the data is coming from FEDs. It may also be possible to use a generic ATCA carrier board and several FED AMC mezzanines directly [ref] if FED AMC card will include the DIB functionality (to pass the data for L1 track trigger to PRBs). After zero suppression, the N DIBs transfer the event data to M number of pattern recognition boards (PRB), which contain Mx4 pattern recognition mezzanine (PRM) cards. Data transfers from the DIBs to the PRMs are time multiplexed, thereby the bandwidth requirements can be significantly relaxed.

Data entering the PRB can be time multiplexed again and transferred to the four PRMs to further reduce bandwidth requirements and allow for longer processing times. The full mesh backplane fabric supports any variant of these configuration (assuming that $N + M \leq 12$), and different variant may have different demands on hardware. Example variations are sketched on Fig.10 and summarized in Table 1.
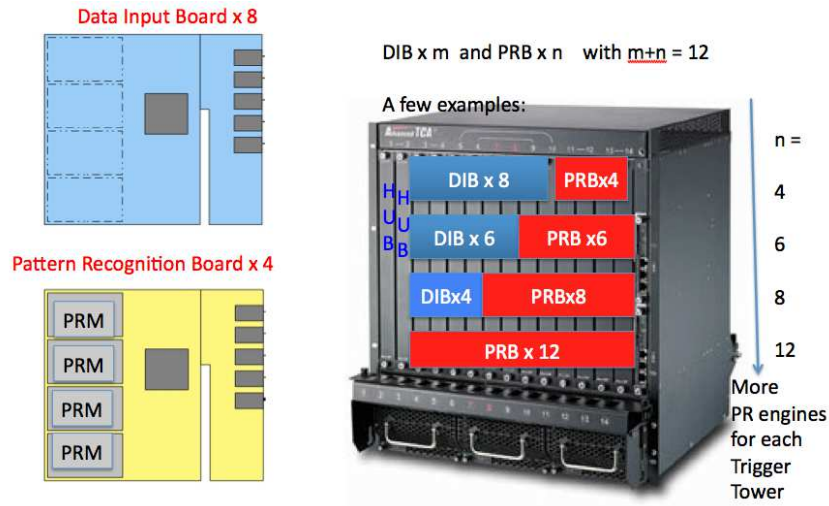
10

Figure 10: System flexibility: many configurations possible & being studied to select the right one for demonstration purpose

DIB/PRB/PRM Count	Fabric Channel BW (minimum)	PRM Input BW (minimum)
8/4/16	20 Gbps	40 Gbps
6/6/24	20 Gbps	27 Gbps
4/8/32	20 Gbps	20 Gbps

Table 1: Data sharing between towers occurs on the PRB board level. Each PRB connects to the corresponding PRB in the eight nearest tower processor shelves. The above numbers assume 500 32-bit stubs per event (every 25 ns). An example of special configuration with eight DIBs and four PRBs will be used as a simple example in Section 3.

275 **DIB/PRB combo configuration** In the limit of N=0 and M=12 from the "N DIB and M PRB" configuration, the
276 DIB and PRB functionalities can be combined into one blade design, shown in Figure 11.
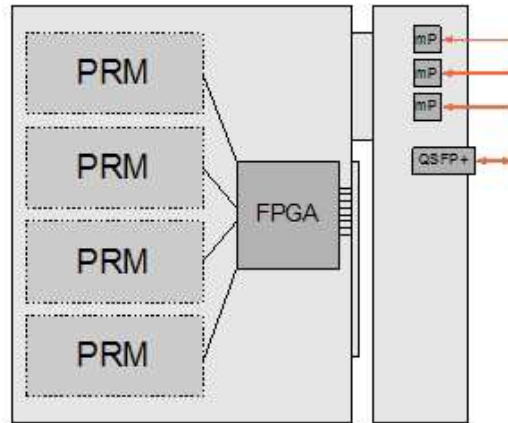
Figure 11: Combined blade design

277 A tower shelf would then consist of 10 Processor blades, one Gateway blade (for data sharing), and one Collector
278 blade (for tracks found). These three different blade functionalities can be implemented in the same hardware.
279 Backplane transfers are described in a series of fully pipelined sequences shown in Figure 12.

280 This processor architecture uses every channel in the full mesh backplane. By using the full mesh fabric more
281 effectively we are able to decrease the channel bandwidth requirement from 20 Gbps down to 6 Gbps with no
282 significant latency increase.

283 In summary, the architecture proposed here for the CMS L1 tracking trigger system is based on ATCA with
284 full-mesh backplane. The large inter-board communication bandwidth provided by the full-mesh backplane is
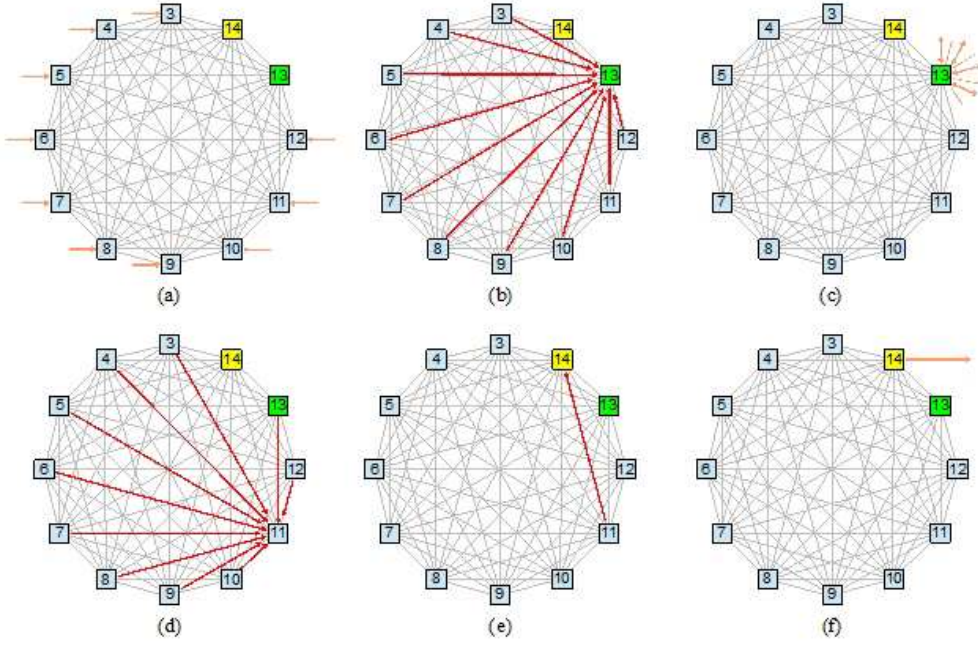
Figure 12: Backplane transfers sequences using the combined blade design. First, the input fibers are received on the Processor blades (a). Each Processor blade then transfers a portion of the input data to the Gateway blade (b), where it is exchanged with neighboring towers over fiber links (c). The Processor blades and Gateway blade transfer the event (including neighbor data) to the target Processor blade in a time multiplexed, round robin scheme (d). Results from the Processor blades are then transferred to the Collector blade (e) for any final formatting and processing before transmission downstream (f).

used to time multiplex the high volume of incoming data in such a way that the I/O bandwidth demands are manageable at the board and chip level, making it possible for an early technical demonstration with existing technology. The resulting architecture is scalable, flexible and open. For example, it allows different pattern recognition architectures and algorithms to be explored and compared within the same platform. Also, given that AMC specifications are designed to work with both ATCA and MicroTCA, this architecture allows a natural long-term integration of TK-DAQ (AMC card based) and TK-TRIG (ATCA based). In Section 3, we will describe the concept of how to build a vertical slice system demonstration based on this flexible architecture.

12

3 Vertical Slice Demonstrator System

3.1 Overview and methodology

The flexible architecture described above lends itself to an early technical demonstration of the system. The main goal of the demonstration system is to identify possible problems in the architecture design and, hopefully, find solutions. We would study, measure and optimize trigger latency and efficiencies at different stages of the system using hardware prototypes being developed. This will involve extensive simulation work, to guide the hardware implementation and to compare actual measurements with expectations. A possible Vertical Slice Demonstration System is shown in Figure 13. Each stage is described in more detail in the following sections.

Although the architecture is flexible enough to allow for different configurations, for the sake of clarity and simplicity, in what follows we will often use the specific configuration with eight Data Input Boards and four Pattern Recognition Boards as an example. We will decide only at a later time which specific configuration we will actually use for the demonstration system.

This demo system will be implemented in stages, at mezzanine level, board level, crate level and multi crate level. These different stages would naturally proceed in sequence, from the bottom up. This way, we will have the opportunity to learn along the way about the performance of the different components of the system before having to decide exactly how the whole thing will be cabled up. Also, the extra crate, with three neighbor towers, will need to come into play only at a very advanced stage, towards the end when the system dynamics needs to be demonstrated.
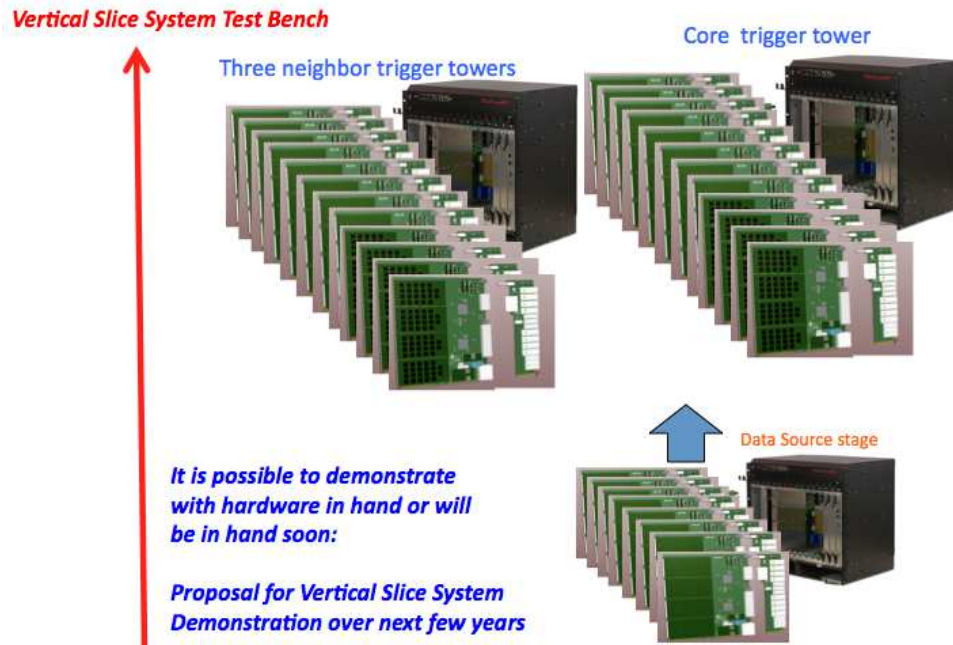
Figure 13: Vertical slice test bench principle.

3.2 Data Source Stage

The Data Source mimics the data flow out of the upgraded Phase II outer-tracker running at the HL-LHC. It will drive 300+ fibers (one/module) to the trigger tower under study exactly as the data were coming from the real detector at high luminosity and full speed. Each fiber connection will transmit data at 3.25 Gbps payload bandwidth, in the same way as the actual modules in the future real system. The data will be derived from simulation, appropriately formatted, stored into on-board memories, and then played back at full speed.

3.3 Data Input

The Data Input Blade (DIB) is responsible for receiving data from the upstream detector electronics (or Data Source output) and transferring them to the PRBs. Up to about 40 fiber links will be received by each DIB. These input links may terminate on the RTM or mezzanine cards. Input fiber links are nominally 3.25 Gb/s payload

13

bandwidth. The Data Input Board will perform zero suppression, pack the stubs into a new format and send them to the PRBs. Current estimates indicate that a rate of about 200 stubs per event per trigger tower, which yields a data rate of roughly 256 Gb/s (200 stubs*32 bits/25 ns) entering each trigger tower on average.

As an example, in the configuration with eight DIBs and four PRBs, each DIB will be receiving an average of about 256/8 = 32 Gb/s of stub data (after zero suppression). Each of the eight DIBs in the shelf sends data to four PRBs in a round-robin, time multiplexed fashion. Since data is sent to four PRBs, these transfers can take place at a quarter of the input rate, or 32/4 = 8 Gb/s, as shown in 1.

In Figure 13, the ATCA shelf devoted to the "core" trigger tower is shown equipped with 8 DIB boards and 4 PRB boards while the shelf devoted to the "neighbor" towers is equipped with 8 PRB boards. In general, each tower needs to share data with 8 neighbors but 3 are sufficient in the demonstration system to test all possible data sharing cases (eta, phi and "diagonal"). Simulated data corresponding to three neighbor tower are delivered from PRB boards in the "neighbor" shelves to the corresponding PRB boards in the "core" shelf.

3.4   Pattern Recognition Board

Using again the special configuration above (8 DIB + 4 PRB), each PRB will be receiving 64 Gb/s stub data on average. While receiving the data and sending them to the mezzanine cards, each PRB will exchange data with the corresponding PRB, processing the same time slice, in the neighboring tower for data sharing in the overlap regions. In this case, each PRB can use four 40Gb/s links (QSFP) for the connections in the eta and phi directions, and four 10Gb/s links (SFP+) can be used for data sharing in the "diagonal" directions (see Figure below for the shelf interconnections). The PRB FPGA drives data received from the DIBs to the Pattern Recognition Mezzanine (PRM) boards. This can also be done in a 4x time multiplexed fashion. The 4x time multiplexed transfers from the PRB FPGA to the PRM would require a bandwidth of about 16Gb/s this way. This also means that each PRM must contain all of the patterns for the trigger tower in this special configuration.
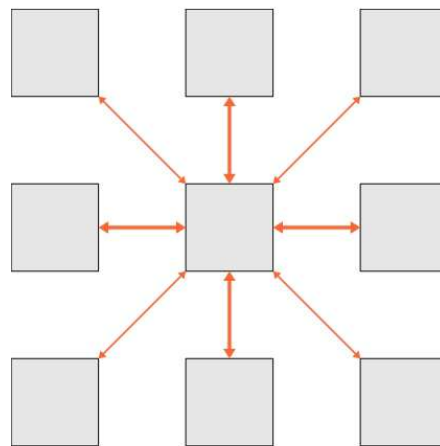
Figure 14: ????

3.5   Pattern Recognition Mezzanine Card

Each PRB supports four Pattern Recognition Mezzanine (PRM) boards. These boards are based on the FMC standard and support high speed LVDS and SERDES connections to the PRB FPGA. In one possible incarnation, each PRM will contain an FPGA, on board memory to act as Data Buffer, and an array of pattern recognition devices. In our example configuration, we need to support 16 Gb/s between the PRB FPGA and the PRM. PRM's using different approaches to track finding and fitting may be tested and compared within the same overall high-level system architecture and data dispatching scheme.

The FPGA-PRAM channel bandwidth needs will be a fraction of the PRM input bandwidth, because only relevant stubs will be sent to the relevant pattern recognition chip covering the relevant regions of the trigger tower (see Figure 15).

14

Figure 15: PRM working principle

4 Simulation

Simulation software plays an essential role in the development of a tracking trigger for CMS as it provides crucial input to the system design and hardware development. It is, therefore, imperative to have a robust and efficient simulation framework at the early stage of the project. The efforts to develop simulation software for AM-based tracking trigger implementation can be divided into the following tasks:
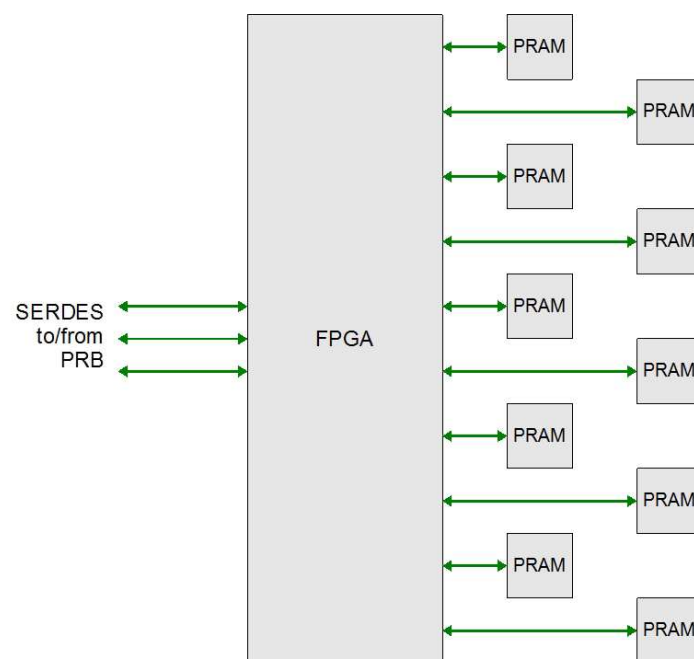
1. **Data Format/Flow**: this part of the framework defines data formats and simulates the flow of the data at all stages from the tracker front-end electronics (transmitting hits associated with stubs) to the Layer-2 of the system (transmitting L1 tracks). This package should include detailed emulation of every stage of the data transmission including:

   - Front End → Data Input Boards
   - Data Input Board → Pattern Recognition Board
   - Pattern Recognition Board → PR Mezzanine
   - Transmission of matched patterns from PR chip to the track fitting FPGA
   - Transmission of L1 tracks to downstream

2. **AM simulation**: this part of the framework defines geometry of the trigger towers, this package is close connected to the previous and following packages and provides the expected output geometry used in the construction of the hits and further on the roads.

3. **AM Pattern Bank Generation**: generates pattern bank, performs pattern matching and is interface-able with track fitting simulations. There is an existing framework in CMSSW, which serves this role and the Lyon/Padova/Kolkata groups provide development and support for it. The main effort here should be aimed at improving the current performance of the package, namely addressing limitations related to speed and memory consumption.

4. **Track Fitting**: this package is closely connected with the core simulations and provides functionality for emulating the FPGA-based track fitting stage. The package should be modular, allowing different algorithms to be compared. It should allow individual users to quickly access matched roads and associated hits, implement novel track fitting algorithms and test their performance.

5. **Integration**: in order to have a complete bit-level emulator of the tracking trigger system, integration of the packages outlined above is necessary. The work here is to integrate, streamline and validate functionality of the full emulator.

6. **Fast simulation**: based on the SVT and FTK experience, it is reasonable to expect that the development of the full software suite outlined above will take a long time (years). On the other hand, improvements in the performance of CMS due to the availability of a tracking trigger need to be quantified on a much shorter time scale. It may, therefore, be advisable to invest in the development of a "light" tracking trigger simulation framework. The goal of this package would be to mimic the L1 tracking trigger performance by using offline hit/track collections and applying parametric efficiencies, fake rates and resolutions.

7. **Vertical Slice Demonstration System Firmware and Software**: In addition to the emulators, a number of software and firmware packages will have to be delivered for each hardware component of the demonstrator. This includes hardware access software, low-level board validation software, integration software and monitoring software. The firmware for each component includes: core functionality firmware, validation firmware and algorithmic firmware.

5 Work Breakdowns and Resources

One of the main purposes of this document is to define the tracking trigger R&D project and organize the efforts. In this section, we will describe possible work breakdowns and work packages for the Vertical Slice System Demonstration. The main work involved can be roughly divided into seven areas, and is shown in Figure 16 below. These seven areas can be in the future organized as seven Work Packages shown in Figure 17. The work breakdown and resources will be one of the main discussion topics with interested groups at the tracker upgrade week.
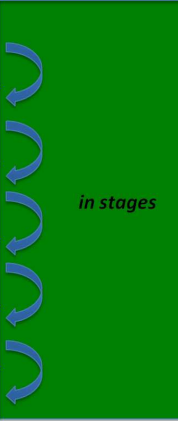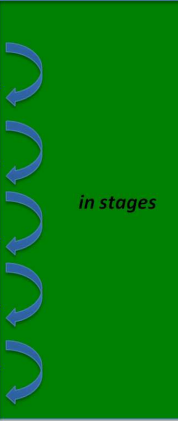
Areas \ Tasks	System Design Specifications (t1)	Hardware Implementation (t2)	Vertical Slice Integration (t3)
Data Source stage	Format/flow/bandwidth	Firmware Specs Hardware Testing	
Data Input Board	Format/flow/bandwidth	Firmware Specs Hardware Testing	
PRB (Pattern Recognition)	Format/flow/bandwidth	Firmware Specs Hardware Testing	in stages
PRM (Mezzanine)	Format/flow/bandwidth	Firmware Specs Hardware Testing	
AM (or new approach)	Interface spec	Design/Testing	
Track Fitting	Interface spec	Design/Testing	
Simulation	System Level Simulation	Hardware Emulation	System Emulation

Figure 16: Work packages

Areas \ Tasks	System Design Specifications (t1)	Hardware Implementation (t2)	Vertical Slice Integration (t3)
Data Source stage	Work Package 1		
Data Input Board	Work Package 2		
PRB (Pattern Recognition)	Work Package 3		Vertical Slice Integration
PRM (Mezzanine)	Work Package 4		
AM (or new approach)	Work Package 5		
Track Fitting	Work Package 6		
Simulation	Work Package 7		

Figure 17: Work packages

17

	System Design Specifications (t1)	Hardware Implementation (t2)	Vertical Slice Integration (t3)
Tasks / Areas			
Data Source stage	TK-DAQ CMS-UK (?)	?	
Data Input Board	TK-DAQ CMS-UK (?)	?	
PRB (Pattern Recognition)	Northwestern/FNAL /INFN	Northwestern/FNAL	*in stages*
PRM (Mezzanine)	FNAL/INFN/CERN	FNAL/INFN/CERN	
AM (or new approach)	FNAL/INFN	FNAL/INFN	
Track Fitting	Lyon/INFN/Cornell/ FNAL	Lyon/INFN/Cornell/ FNAL	
Simulation	Lyon/FNAL/INFN/CERN/Florida/Cornell/...	Group working on the hardware	

Figure 18: Current involvments

6 Conclusion

References

[1] M. Dell Orso, L. Ristori - *"VLSI Structure For Track Finding"* - Nucl.Instr. and Meth. A278 (1989), 436

[2] J. Adelman *et al.* - *"Real time secondary vertexing at CDF"* - Nucl.Instr. and Meth. A569 (2006), 111

[3] J. Adelman *et al.* - *"The Silicon Vertex Trigger upgrade at CDF"* - Nucl.Instr. and Meth. A572 (2007), 361

[4] T. Kohonen - *"Content-Addressable Memories"* - 2nd edition, New York, Springer- Verlag (1987)

[5] K. Pagiamtzis, A. Sheikholeslami - *"Content-Addressable Memory (CAM) Circuits and Archtectures: A Tutorial and Survey"* - IEEE Journal of Solid-State Circuits, Vol. 41, No. 3 (2006), 712

[6] A. Annovi *et al.* - *"A VLSI Processor for Fast Track Finding Based on Content Addressable Memories"* - IEEE Transactions on Nuclear Science, vol. 53, no. 4 (2006), 1

[7] A. Annovi *et al.* - *"The GigaFitter: A next generation track fitter to enhance online tracking performances at CDF"* - Nuclear Science Symposium Conference Record (NSS/MIC), 2009 IEEE (2009), 1143

[8] T. Liu *et al.* - *"A New Concept of Vertically Integrated Pattern Recognition Associative Memory (VIPRAM)"* - FERMILAB-CONF-11-709-E, Proceedings of TIPP 2011 conference, Volume 37, 2012, Pages 1973

[9] T. Liu *et al.* - *"Development of 3D Vertically Integrated Pattern Recognition Associative Memory (VIPRAM)"* - FERMILAB-TM-2493-CMS-E-PPD-TD

[10] - *"FTK: A Hardware Track Finder for the ATLAS Trigger"* - Technical Proposal (2010)

A State of the art and on-going R&D

Motivated by the challenges described above, over the past few years, a number of institutions in CMS have been working on R&D projects related to the development of fast tracking based on the Associative Memory and tracklet-based approaches.

The AM device solves the combinatorial problem, inherent to this kind of pattern recognition algorithms, by employing a massively parallel architecture to compare each detector hit to a large number of pre-calculated geometrical patterns simultaneously. Then, the selected patterns are processed using fast FPGAs to perform track fitting. Since each pattern corresponds to a very narrow "road" through the detector, the usual helical fit is much simplified and fast by using a pre-calculated set of parameter values for the center of the road and applying corrections that are a linear function of the actual hit positions in each layer.

INFN has been working together with the Atlas FTK team exploring the possibility of using FTK associative memory chips for CMS applications. Development of the simulations for the Associative Memory approach in CMS was started by the Pisa and Lyon groups with significant progress made towards making the machinery work in the standalone mode. Now, together with Padova, they are migrating the tool into CMSSW. Lyon group has been also focusing on the possibility of using a new algorithm, based on Hough's transforms, to replace the conventional SVT-style, linearized track fitting algorithm.

Fermilab carried on a focused R&D program to advance the state-of-the-art of pattern recognition and track reconstruction for fast triggering. Specifically, Fermilab has been developing a new Data Formatting system based on full-mesh ATCA and exploring new Associative Memory VLSI structures based on new technologies both 2-dimensional and 3-dimensional. The 3D approach to Associative Memory implementation is particularly appealing because adding the "third" dimension opens up the possibility for new architectures that would dramatically increase the achievable density of patterns and the pattern recognition power.

The long-term goal of this R&D efforts is to develop all necessary critical technologies to the point where we can ultimately propose them as a viable solution to the CMS L1 tracking trigger problem for HL-LHC. Major progresses have been made recently. INFN has done the initial measurement of AMchip03 latency, Lyon's group had already initial success with the simulation of the Associative Memory and has performed some detailed studies on an algorithm based on Hough's transform. Fermilab has successfully tested the first Data Formatting system components (ATCA motherboard nicknamed Pulsar-IIa, RTM and mezzanine cards) and they all work well, actually better than expected. Fermilab is now ready to test the first Associative Memory 2D prototype chips expected to be delivered at the end of 2013. Pulsar II hardware design is flexible enough to be used as the main workhorse to build the Vertical Slice Demonstration System.

A.1 PRAM and its Development

As mentioned earlier, the CDF SVT-style Associative Memory chip (from now on we will call it PRAM, Pattern Recognition Associative Memory, to emphasis its purpose for HEP) is a departure beyond conventional CAMs. Like conventional CAMs, PRAMs store address patterns and look for matches between incoming hits and those addresses for a given detector layer. At this level, the match is expected to be either exact (Binary CAM) or partial (Ternary CAM) and an array of Match Flags is the typical output. A PRAM has an array of Match Flag Latches which capture and hold the results of the match until reset for the next event. As the hits from the various layers of the detector for the same event arrive, the PRAM is looking for more than simple matches from one candidate address to one or more stored address patterns. The PRAM organizes stored address patterns into roads, which are linked arrays of several stored address patterns from different detector layers. Each stored address pattern in a road is from a different layer in the detector system and these linked arrays represent a path or road that a particle might traverse through the layers of the detector (hence the name "road"). The ultimate goal of the PRAM is to match real particle trajectories to those roads. Like a conventional CAM, a PRAM flags a match when a candidate address matches a stored pattern address for a given detector layer. However, before the PRAM does anything with that match, it must find matches in all (or majority of) the elements (layers) that constitute a road.

It should be emphasized that compared to commercially available CAMs, such as Network Search Engine, the PRAM has the unique ability to search for correlations among input words received on different clock cycles. This is essential for tracking trigger applications since the input words are the detector hits arriving from different layers at different times. They arrive at the chip without any specific timing correlation. Each pattern has to store each fired layer until the pattern is matched or the event is fully processed. Even in the case of a level-1 trigger application, which is largely synchronous, this feature will still be important. One unique feature of this approach is that the pattern recognition of the event is done as soon as the last hit arrives, which makes the approach a

promising candidate for L1 track trigger. However, the requirements for L1 track trigger application will be very different from that for L2, and the system interface of the chip has to be fully redesigned and the performance has to be optimized.

The PRAM pattern density can be improved by optimizing the design in single-layer chips (2D), using custom cell designs with smaller feature size technology. There is an R&D effort by INFN using 65 nm technology to improve design for Atlas FTK application for L2 trigger (AMchip05 or 06). INFN has now in hands a 65 nm version prototype, developed for FTK purposes, which could be used for initial testing. INFN AM05 has been submitted recently, and the AM06 will be submitted in the Spring 2014. INFN AM06 for FTK with 128 Kpatterns is expected to become available in 2015.

There is also an on-going R&D effort at Fermilab using both conventional 2D and the emerging 3D technology to design future generation of PRAM chip [8],[9] specifically for the needs of the L1 CMS tracking trigger needs (ProtoVIPRAM series). The first 2D prototype chip (ProtoVIPRAM01) is expected to arrive by the end of 2013.

A.2   Track Fitting

The traditional CDF SVT/FTK-style track fitting stage can be used to benchmark the performance of this stage. We are exploring other new approaches as well, such as Hough transform algorithm and tracklet-based algorithm. All track fitting algorithms can be implemented in FPGA on the PRMs, therefore they can be studied and compared directly using the same vertical slice demonstration setup. More detailed description of the two new approaches will become available in this section later. As a reference, the traditional SVT/FTK-style track fitting is described below [10].

For a region of detector sufficiently small, a linear approximation gives helix parameters close to those of full helical fit. In other words, for a road narrow enough that a helical fit can be replaced by a simple linear calculation, each of the 5 helix parameters (p_i) can be calculated as the vector product of prestored constants (a_{ij}) and the hit coordinates (x_j): $p_i = a_{i0} + \sum_{i=1}^{N} a_{ij}x_j$ where N is the number of coordinates on the track, one for each SCT layer and two for each pixel layer. Since there are more than 5 coordinates, there are additional linear equations that correspond to constraint equations, again where the constants are prestored. There are (N - 5) such equations.

This linear approximation gives near offline resolution for regions considerably wider than a single road. A single set of constants will be used for each sector of the detector. The width of the sector at each silicon layer is the size of a physical detector module. Per sector, 5(N + 1) constants are needed for the helix parameters, and (N - 5)(N + 1) constants are needed for the constraint equations. The total number of fit constants (FC) per sector is thus N(N + 1).

Will add some description here about CDF Gigafitter performance with FPGAs [7] etc. also with FTK most recent performance with modern FPGAs.

B   Pattern recognition and track fitting approaches

B.1   Introduction to associative memory approach

The SVT approach at CDF was the first hadron collider experiment in HEP to incorporate a fast secondary vertex track trigger [2], [3] to find all tracks produced in each collision and precisely measure their properties within about 30 microseconds from the time of the collision. The SVT significantly extended the physics reach of CDF opening the way to measurements that would have been impossible otherwise.

The CDF SVT-style associative memory approach, later adopted for FTK, consists of two sequential steps of increasing resolution. In step 1, pattern recognition is carried out by a dedicated device called the Associative Memory (AM) which finds track candidates in coarse resolution roads. When a road has stubs on all layers or all except one, step 2 is carried out in which the full resolution hits within the road are fit to determine the track helix parameters and a goodness of fit. Tracks that pass a 2 cut are kept. If there are hits in all layers and the 2 fails the cut but is not extremely large, the track is refit a number of times with the hit on one of the layers dropped each time. This "majority recovery" allows for the loss of a single hit/stub due to detector inefficiency with a random hit picked up instead.

Note that there has been more than a decade of experience in HEP with this approach from both CDF and Atlas, though all with silicon-based tracking trigger for Level 2 where trigger latency requirement is much more relaxed.

516 Therefore, one of the key issues for L1 implementation or demonstration will be the overall tracking trigger latency.
517 Both the pattern recognition at the AM stage (hits delivery) and track fitting stage will have to be fast enough.

B.1.1 From pattern recognition to track fitting

519 The pattern recognition step is usually the most time consuming task, because one needs to test many combinations
520 of hits to find those that potentially come from the same track and typically these tests are done sequentially (e.g.
521 in software). The associative memory approach allows the testing of all hit combinations in parallel against a
522 set of known patterns. To illustrate the concept of patterns in the associative memory approach, one can use an
523 oversimplified case with a simple detector consisting of six detector layers, as shown in Fig. 19. A charged track
524 crossing the detector would produce a set of hits (pattern). A finite set of distinct patterns can be generated this
525 way using valid tracks for a given experiment, and such sets are often called the pattern bank.
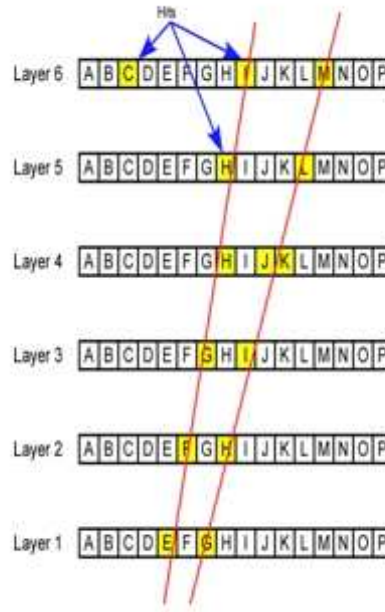

Figure 19: Tracks in a tracking detector

526 The Associative Memory (AM) architecture is based on Content Addressable Memory (CAM) cells [4, 5] to effi-
527 ciently identify track patterns (roads) at high speed using coarse-resolution 'hits' recorded in the tracking detector.
528 A block diagram [1] of the Associative Memory architecture is shown in Fig. 2 (for a case with four detector
529 layers). Each pattern (shown as a cell) is composed of four hit coordinates each of which is stored in the CAM
530 word for a given layer, and only four patterns (cell 0 to 3) are shown in Fig. 20.

531 An incoming hit from a given detector layer is transmitted to the corresponding layer and the hit coordinate is
532 compared against the stored words for all patterns in parallel for that layer. Any match to each incoming hit will
533 be latched for that layer an d for that pattern until reset (to rearm for next event). This process is repeated for all
534 the incoming hits for each detector layer as the hits arrive one after the other. As soon as all hits from the same
535 event are received, the hit matching stage is done and all latched matches for each pattern (or cell) will be fed into
536 a majority logic stage where a fired road will be found if the number of matched layers reaches a programmable
537 threshold.

538 Associative Memory is sometimes called PRAM, Pattern Recognition Associative Memory. The AM method
539 solves the combinatorial challenge inherent to the pattern recognition by exploiting massive parallelism of associa-
540 tive memories that can compare tracking detector hits to a set of pre-calculated patterns simultaneously. The found
541 patterns or "fired roads" are then processed using fast FPGAs to perform track fitting with full detector resolution
542 using all combinations of the "hits of interest" from the fired roads. Because each pattern or road is narrow enough,
543 the usual helical fit can be replaced by a simple linear calculation. The track fitting stage for each matched pattern
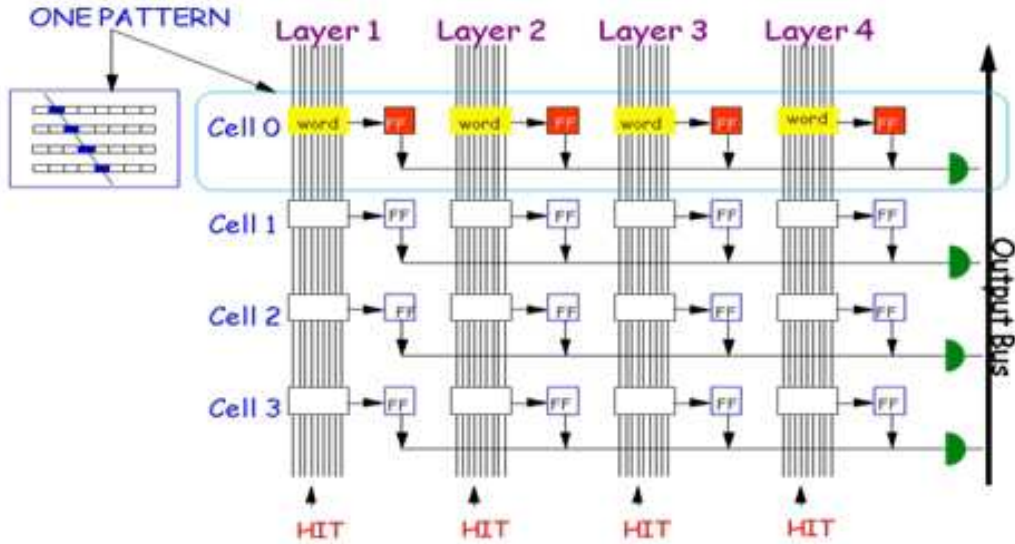544 is much simplified and can be very fast [7].

Figure 20: A block diagram on an associative memory chip. The CAM cells are shown as white (or yellow) boxes. The majority/glue logic is shown to the right as semi-circles.

B.2   Trigger Tower, Sector, superstrip, and pattern banks

We have described the concept of Trigger Tower in Section 2. Here we will describe the concept of sector, superstrip and pattern banks for the AM approach.

The pattern generation procedure starts by identifying first the sectors, which consists of one silicon module in each detector/logical layer. The list of sectors is determined with a training procedure that uses a large number of single muon (negative and positive) events, to cover all possible combinations of modules compatible with tracks coming from a region around the nominal beam spot position. The concept of sector is important because it is the region in which the linear track fit with one set of constants is performed (see later track fitting stage).

Each pattern is made of one superstrip per layer. A superstrip is a group of strips/pixels, and is therefore heavily constrained by the detector itself. Once the superstrip definition has been set, its position information is coded in a N-bits word: the superstrip address. It is important to realize that the AM-based pattern recognition is using these addresses, and is therefore independent from the detector geometry. As the addresses are transmitted to the AM independently for each layer, layer number doesn't have to be in the address word.
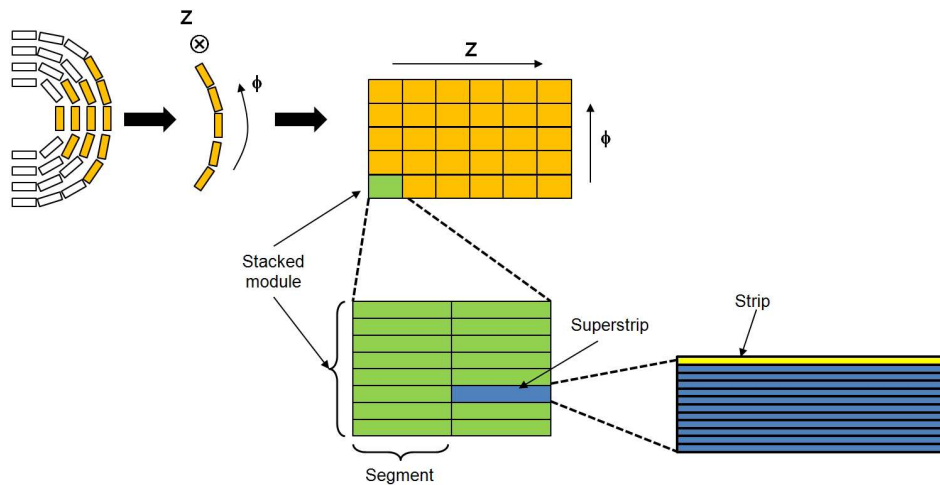

Figure 21: Geometric definition of a superstrip (barrel example).

In order to understand the address definition, Fig. 21 shows how a superstrip is defined in the barrel part of the
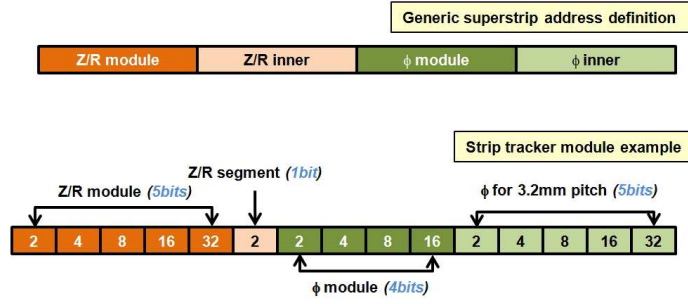
24

Figure 22: Address definition of a superstrip.

tracker. Figure 22 shows the address definition of a superstrip. The number of bits necessary reflects the superstrip granularity and is constrained by the maximal word size acceptable by the AM chip (default is 15 bits). A first set of bits provides the module number along the corresponding coordinate: 5 bits for Z (one could have up to 24 modules in Z in one sector) and 4 for ? (up to 11 modules per sector in the outermost layer). Then, for each coordinates, a second series of bits provide the superstrip position within the module. Fig. 6 shows a tracker strip module (in green), divided into 2 segments in Z and 1024 strip in ?. Therefore in order to describe all the position, one would need only 1 bit for Z, and 10 bits for ?. In practice, a certain number of strips are grouped to form the superstrip, in our case 32, thus leading to 5 in the address. For the endcap the coding is the same, with z being replaced by r (equivalence is made between disks and layers, ladders and rings).

C   Simulation tool development

C.1   The importance of simulation tools

AM-based L1 tracking is a relatively complex task requiring a precise modelization before going to hardware implementation. This modelization is done via a dedicated simulation framework comprising:

- The model of the new tracker geometry implemented within the official CMS software chain (CMSSW)

- The definition of the trigger tower

- The definition and the production of the pattern banks for each trigger tower

- The emulation of the AM-based pattern recognition within CMSSW

- A set of track fitting procedures compatible with an FPGA implementation, coded within CMSSW

The following sections of this part are providing a description of the different steps necessary to setup this modelization.

C.2   Tracker geometry and trigger tower definition

The geometry description and the trigger sector definition is provided by a standalone program: the TkLayout tool [?]. For the geometry, this tool builds the xml files necessary to the CMSSW simulation stage. Then, using rates obtained from simulation and L1 tracking trigger requirements (in particular the minimal p_T required for the tracks), the TkLayout tool can be used to optimize the trigger towers definitions. The goal here is to define the largest possible towers with a reasonable input data rate per layer, and to minimize the data sharing between towers. After performing this task, the TkLayout tool provides a simple text file containing, for each sector, a list of the tracker modules belonging to that sector. This file defines the structure of the track trigger system, and is hence is the basis of the L1 tracking stage.

C.3   Pattern bank generation

Bank generation is done via a dedicated standalone C++ program [?]. The main bank generation procedure is sketched on Fig. 23. In order to produce a bank, two inputs are needed: a large simulated sample of single particle gun events (traditionally muons) covering the whole L1 track trigger acceptance requirements, and the TkLayout file providing the trigger towers definition.
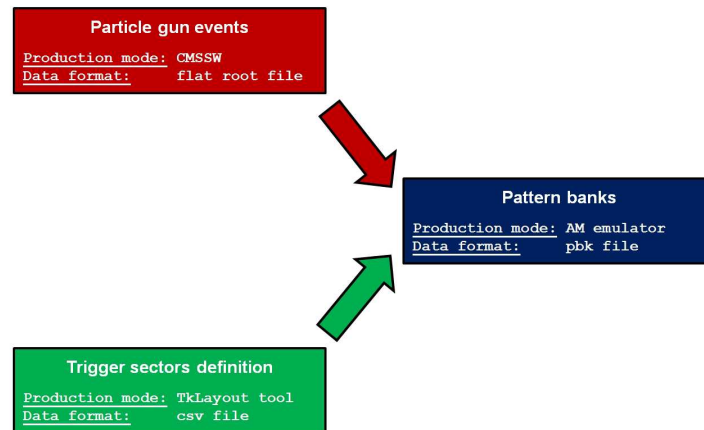

Figure 23: Pattern bank generation principle

Based on these inputs, the code is able to produce a pattern bank for any trigger tower. Bank parameters can be set in the configuration file of the generation job. A complete description of the available option is provided in [?].

The bank generation algorithm, as it is implemented in the emulator, is summarized on Fig. 24.
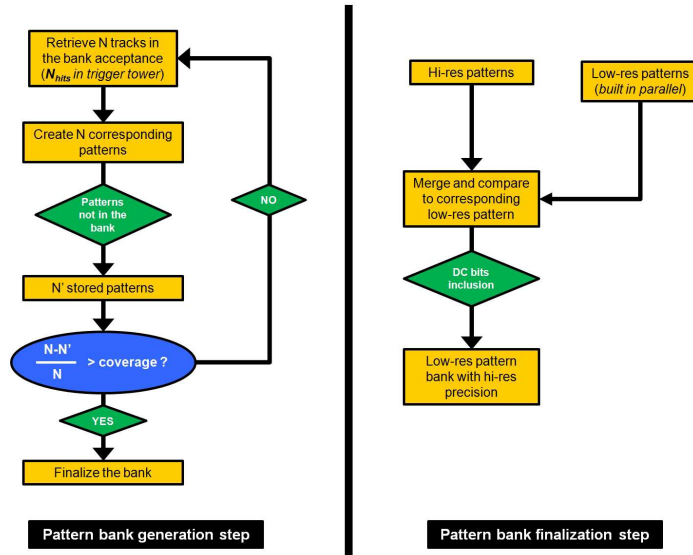
Figure 24: Pattern bank generation workflow: from track sample to pattern bank

This algorithm can be divided into two steps: bank production and bank finalization. As shown on the left diagram, the bank production is performed iteratively. An iteration starts by the collection of a set of N tracks satisfying the pattern bank acceptance requirements in the input data sample (in our case $N = 40000$). This acceptance requirement is defined by the parameter N_{hits}, which corresponds to the number of different layer/disks in which the particle has induced stubs in the trigger tower considered[1].

A track can be used to produce a pattern only if its N_{hits} value is equal to the pattern size set in the job configuration. Of course the pattern made from the track is stored in the bank only if not already in. Iteration ends when N tracks have been collected. At this point, the proportion of tracks collected which were leading to an already stored pattern is computed. If this proportion, defined as the bank coverage, is larger than the coverage initially required, the iteration stage stops.

Once the required coverage has been reached, the bank is finalized. The procedure is described in the right part of Fig. 24. At this point, it's not one, but two banks which have been built. Indeed, each new pattern is automatically linked to a low-resolution mother pattern. The resolution of this mother pattern is direclty related to the number of DC bits. The width of the low-res patterns is indeed $2^{N_{DC}}$ larger than the width of the high-res ones. For example, if one requires 2 DC bits, and built pattern with a superstrip size of 32 strips, the width of the low resolution patterns will be 128 strips.

The low-res bank building follows the same procedure than the hi-res bank. But in addition, each low-res patterns is linked to a set of hi-res patterns. Before storing the low-res pattern in the banks, hi-res patterns belonging to it are merged. The merging result is then compared to the low-res pattern, and the hi-res granularity is kept whenever possible. On the other hand, if all the super-strips are used in a given layer, corresponding DC bits are applied.

Using this procedure provides the possibility to get high-resolution precision for the price of a low resolution bank.

C.4 Pattern recognition

C.4.1 Software principle

The pattern recognition is done using the same code than the bank generation. This code has been interfaced with the CMS software framework (CMSSW). It uses the official CMSSW stubs and stores the patterns as official CMSSW track seeds. For debugging purposes, the pattern recognition code can also work standalone, starting from plain rootuple containing the stubs of a given event.

The code is working for a given sector. The complete pattern recognition of a given event therefore requires to run in parallel jobs for every trigger tower. The data merging is performed afterwards, so that the final output contains

---

[1] This number should not be confused with the total number of stubs induced by the particle. If a track induces two stubs in a given layer (for example in an overlap area), these stubs will just count as one in N_{hits}.

for each event the pattern recognition outcome for all the towers.

C.4.2 Estimation of the pattern recognition efficiencies and fake rates

The basis of the pattern recognitionefficiency is the number N of matchable tracks. This number is depending on the size of the patterns stored. For patterns made of 6 superstrips, if a missing hit is accepted, any track crossing at least 5 layers in the tower will matchable.

The number of layers/disks crossed by the particle within one tower is therefore an important parameter. From now own, it will be mentioned in all the notations. For example, N_5 corresponds to the number of tracks crossing at least 5 layers/disks in one of the trigger tower.

Starting from there, the efficiency for a given threshold could be defined as the number of tracks matched in a pattern divided by the total number of tracks:

$$\epsilon_i = \frac{N_i^{matched}}{N_i} \tag{1}$$

Then, we define the fake rate as the proportion of patterns which are not containing a good track.

$$\rho_i = \frac{N_i^{badpatterns}}{N_i^{patterns}} \tag{2}$$

This fake rate definition doesn't account for the duplicated patterns. This value is measured by the redundancy,which corresponds to the average number of patterns activated by each good track. The ideal pattern bank for a given threshold i is the one for which $\epsilon_i = 1$ and $\rho_i = 0$.

Such a bank exists, this is the one for which each possible track in the detector corresponds to one pattern. However, the size of such bank will be prohibitive. On the other side, one could imagine the simplest bank with 1 single pattern defined by the whole detector. Such a bank will have $\epsilon = 1$, $\rho = 0$ and $r = 1$, but ϵ_{hits} will be close to 0. Such a solution would be highly inefficient, as TF would be impossible to process. Therefore the optimal solution lies in between, and one realizes that a parameter will be very important: the size of the patterns. The patterns will have to be sufficiently small in order to get a reasonable ϵ_{hits}, but also sufficiently large in order get a reasonable pattern bank size.

Finding the optimal bank is a complex exercise which will be done for the TDR, and hence is beyond the scope of this preparatory note. In this document we present preliminary results obtained using a simple set of pattern banks, which where produced using a large sample of particle gun $\mu^{\pm}$ produced within the following phase space:

- $0 < \eta_0 < 2.2$.

- $1.95 GeV/c < p_{T_0} < 100 GeV/c$, generated randomly in $1/p_T$.

- $-15cm < z_0 < 15cm$.

- $-1mm < d_0 < 1mm$.

In order to cover correctly the whole pT phase space the sample was divided into three pT ranges: 1.95 to 5 GeV/c, 5 to 20, and 20 to 100. Patterns were made from 32 to 256 strips-wide superstrips (we generate 32 strips banks and used 3 DC bits) in order to get banks of 1M patterns for each trigger tower. The total number of patterns necessary to cover all the tracker in this configuration is 64 Millions.

The following table summarizes the overall efficiencies and fake rates obtained for simple single particles with $p_T > 2 GeV/c$ using the banks described previously, for $N_{hits} = 5$.

Particle type	Muons	Pions	Electrons
Efficiency (ϵ_5, in %)			
Fake rate (ρ_5, in %)			

Table 2: Stub repartition in the tracker, measured with PU=200 events.

Figures 25 and fig:PRELE shows a summary of the pattern recognition outcome for muons and electrons respectively. The broader turn-on for electrons is mainly due to multiple scattering.

Figure 25: Pattern recognition efficiency for single muons

Figure 26: Pattern recognition efficiency for single electrons

In order to evaluate the fake rate, two types of events were used. Both were overlaid with an average of 140 minimum bias events. In the first set, a single muon was added to minbias contents, whereas in the other sample, a four tops final state was added. This last sample could be considered as a worst case scenario for tracker occupancy, and is therefore particularly useful for studying the maximum rates per modules.

The pattern recognition efficiency was also tested, as shown on Fig. **??** for the 4 tops events. As expected, the efficiency is robust against pileup. If a pattern is matched, it will always match independently from the stubs you add around it. What will change is the fake proportion. The fake rates, for both samples are summarized in the following table:

Particle type	Muons	Pions	Electrons
Efficiency (ϵ_5, in %)			
Fake rate (ρ_5, in %)			

Table 3:

C.4.3 Bank efficiency definition

C.5 Further developments and improvements

29

D Track Trigger Performance Requirements

D.1 Tracking performance

D.2 Performance on trigger objects

D.3 Selected physics cases

30