CORE
Metadata, citation and similar papers at core.ac.uk
Provided by CERN Document Server

DISSERTATION

The Hardware Track Finder Processor in CMS at CERN

ausgeführt zum Zwecke der Erlangung des akademischen Grades eines Doktors der technischen Wissenschaften

eingereicht an der Technischen Universität Wien
Fakultät Elektrotechnik

von

Alexander Kluge, CERN/PPE, CH-1211 Genf, Schweiz
geb. 15. April 1968, Wien

Wien, im Juli 1997

Begutachter

Prof. Dr. techn. Franz Seifert

Prof. Dr. techn. Christian Fabjan

Kurzfassung

Für das für 2005 geplante Hochenergiephysikexperiment CMS (Compact Muon Solenoid) am CERN in Genf wurde ein Myon-Trigger- und Impulsmeßsystem (Track Finder Processor) entwickelt. Der ‘Track Finder Processor’ gestattet es, die physikalische Relevanz von hochenergetischen Kollisionen abzuschätzen und nur die Messdaten von interessanten Ereignissen den entsprechenden Analyseeinheiten zuzuführen.

Die Daten von mehr als 200000 Detektorzellen werden benutzt, um den Ort, bzw. die Spur von Myonen aufzufinden und deren Transversalimpuls zu messen. Alle 25 ns wird ein neuer Spurensatz generiert. Die Bestimmung von Flugbahn und Transversalimpuls der Myonen am Wechselwirkungspunkt benötigt insgesamt 350 ns. Die 'pipeline'-Architektur verarbeitet neue Spurensätze mit der geforderten Rate von 40 MHz, um das Gesamtsystem totzeitlos betreiben zu können.

Im Rahmen dieser Arbeit wurden die Spezifikationen und das Gesamtkonzept des Prozessors detailliert ausgearbeitet. Simulationen wurden ausgeführt, um die am effizientesten einsetzbare Meßmethode und Implementierungstechnologie auszuwählen. Zusätzlich wurden schon existierende Systeme bewertet und deren Spezifikationen mit denen des ‘Track Finder Processors’ verglichen. Die klassische Methode bei Hochenergiephysikexperimenten ist, nach vorgegebenen Spurensätzen bzw. Bitmustern in den Meßdaten zu suchen und deren Eigenschaften zu bestimmen, wobei vordefinierte Muster mit den tatsächlich auftretenden verglichen werden (Mustervergleich). Die hohe Zahl der Datenkanäle des ‘Track Finder Processors’ und die komplexen Anforderungen an die örtliche Detektorauflösung erlauben es jedoch nicht, diese Methode zur Anwendung zu bringen. Ein sogenannter Spurverfolgungsalgorithmus wurde entworfen, der ausgehend von einzelnen Spurelementen ganze Spuren durch den Detektor zusammenzusetzen vermag. Anstatt große, vorsimulierte Vergleichsdatensätze zu speichern, wird ein Algorithmus zur Spurensuche und Impulsbestimmung direkt angewandt. Dadurch wird die Umsetzung in Hardware ermöglicht, womit die für das Gesamtexperiment vorgegebenen Anforderungen erfüllt werden. Der Meßalgorithmus wurde in digitaler Logik implementiert. Ausführliche VHDL-Hardware Simulationen wurden durchgeführt, um den Algorithmus und seine Hardware-Repräsentation zu optimieren. Ein FPGA (Field Programmable Gate Array)-Prototyp und ein Testsystem wurden entwickelt. Eine Machbarkeitsstudie wurde durchgeführt, um zu zeigen, daß der Prozessor mittels heute erhältlicher Technologie als ‘Application Specific Integrated Circuits’ (ASICs) realisiert werden kann und den technischen Anforderungen durch das Experiment gewachsen ist.

Abstract

The work covers the design of the Track Finder Processor in the high energy experiment CMS (Compact Muon Solenoid, planned for 2005) at CERN/Geneva. The task of this processor is to identify muons and measure their transverse momentum. The track finder processor makes it possible to determine the physical relevance of each high energetic collision and to forward only interesting data to the data analysis units.

Data of more than two hundred thousand detector cells are used to determine the location of muons and measure their transverse momentum. Each 25 ns a new data set is generated. Measurement of location and transverse momentum of the muons can be terminated within 350 ns by using an ASIC (Application Specific Integrated Circuit). A pipeline architecture processes new data sets with the required data rate of 40 MHz to ensure dead time free operation.

In the framework of this study specifications and the overall concept of the track finder processor were worked out in detail. Simulations were performed in order to select the most appropriate measurement method and implementation technology. Already existing systems were evaluated and their specifications were compared with those of the track finder processor's. The classic method in high energy physics experiments is to search for predefined tracks or bit patterns in the measurement data and to determine their properties. The predefined patterns are compared to the found patterns. The high number of data channels of the track finder processor and the complex requirements to the spatial detector resolution do not permit to employ a pattern comparison method. A so called track following algorithm was designed, which is able to assemble complete tracks through the whole detector starting from single track segments. Instead of storing a high number of track patterns an algorithm for track finding and momentum measurement is employed directly. This enables to realize a hardware implementation within the requirements given by the experiment. The algorithm was translated to the level of digital electronics. Comprehensive simulations, employing the hardware simulation language VHDL, were conducted in order to optimize the algorithm and its hardware implementation. An FPGA (field programmable gate array)-prototype and a test system was designed. A feasibility study to implement the track finder processor employing ASICs was conducted. The study proves that the track finder processor can be implemented using today's technology.

Acknowledgements

This dissertation is dedicated to my parents, Irene and Kurt.

I would like to thank Professor Franz Seifert for supervising my dissertion.

Professor Christian Fabjan I wish to thank for his advice and for being co-supervisor.

I want to thank Dr. Friedrich Szoncso and Dr. Claudia Wulz for entrusting me the task to plan an important part of the experiment and for giving me autonomy during this work.

I owe Dr. Friedrich Szoncso thanks for reading and criticizing my dissertation.

To Dr. Norbert Neumeister and Dipl.-Ing. Torsten Wildschek I express my thanks for many valuable suggestions and the repeated assistance in Physics.

I especially appreciated the teamwork with Dipl. Ing. Torsten Wildschek.

The seemingly insurmountable problems with the operation of VHDL simulators I could only solve with the help of Serge Brobecker.

When it was necessary to prove that the VHDL simulations could also be transformed into hardware, a prototype was built and I relied on numerous pieces of advice by Dipl.-Ing. Anton Taurok.

The collaboration with Stefan Puttinger during the trigger test was of decisive significance for success.

To my whole family, especially to my parents, Irene and Kurt, I am thankful for having had the chance to carry out my studies. For this reason my dissertation is dedicated to my parents.

My special thanks certainly goes to the Austrian Nation for making my stay at CERN possible by the doctoral scholarship which I received.

CARPE DIEM

Usually scientific papers are comprehensible only to a small readership. In order to serve also readers not familiar with physics or electronics an additional feature is provided in the next pararaph. Like the ‘Hitchhiker’s Guide to the Galaxy’ this document contains a very important phrase printed in large capital letters on one of the first pages.

‘DO NOT PANIC’

Table of Contents

Structure of this work:

Chapter 1 gives a short introduction to high energy physics and the applied data acquisition methods.

Chapter 2 covers the environment of the track finder processor. The experiment setup is explained shortly and the data acquisition of the system is described in more detail.

Chapter 3 deals with track finder processor specifications. The system requirements are compared to previously implemented systems.

In chapter 4 both the realisation methods and the implementation technology possibilities for the track finder processor are discussed and the most suitable are selected. At the end of the chapter the realisation feasibility of the chosen realisation method is proven.

In chapter 5 the algorithm and its implementation in hardware is discussed comprehensively.

Chapter 6 deals with the design of a FPGA (field programmable gate array) prototype.

Chapter 7 covers the problem of final implementation employing ASICs (Application Specific Integrated Circuit). A feasibility study is presented where system partitioning and timing analysis show that the system is implementable within the given requirements even with today's technology.

Chapter 8 presents an error detection and location scheme for the entire track finder processor system.

Chapter 9 concludes the work and gives further perspectives of the track finder processor project.

The research and development work for the track finder processor project has mainly been conducted by a physicist, Dipl-Ing. Torsten Wildschek, and me. The responsibilities for the project were shared between the two of us. Torsten Wildschek was responsible for the physics aspects of the processor while my duty was to carry out research and development regarding the hardware aspects of the processor. Since this document concludes the results concerning the hardware aspects of the track finder processor special emphasis has been given to the description of the hardware. However, in order to understand the entire project also physics studies (see chapter 4.3., "Extrapolation method - Feasibility from physics point of view") and the design of the surrounding environment (see chapter 2, "Track finder processor environment") are described, although, only to a certain minimum expense. Of course the design of the track finder processor also influenced the general concept of the first level muon trigger (see chapter 2.3., "CMS first level trigger"). For further information on topics not directly related to the hardware of the track finder processor refer to the literature given in the descriptions.

Of course in some aspects of the design process a close team work was established between Torsten Wildschek and me. This is especially true for working out the specifications of the track finder processor, described in chapter 3.1., and finding a

appropriate track finding strategy which fulfils the needs from physics point of view and is found to be implementable in hardware (chapter 4.1.). However, once the requirements from physics point of view were understood the hardware design process was carried out rather independently by me.
This included

- to perform simulations in order to support chosing the implementation methodology,
- to work out a suitable implementation technology,
- to work out a track finding algorithm implementable in hardware,
- to design the overall concept of the processor,
- to design the detailed architecture of the processor and components,
- to simulate the behavior and optimize the hardware architecture and algorithm,
- to design a FPGA-prototype,
- to design a trigger test bench,
- and to perform a feasibility study for a final implementation.

(see chapters 4.1., “Feasibility study: Feature extraction methods”, 4.2., “Implementation technologies”, 5, “Detailed architecture and functionality of the CMS track finder processor”, 6, “FPGA prototype layout and test”, 7, “Feasibility study for ASIC implementation of the CMS track finder processor”, 8, “Error detection and location”).

1 Introduction

In the following chapters a short overview of techniques in experimental high energy physics is given. The chapter is aimed at readers not familiar with high energy physics experiments. Chapter 1.1. gives a short introduction to high energy physics experimental methods. In chapter 1.2. methods for data acquisition are described.

1.1. Short introduction to high energy physics experimental methods

At CERN (European Laboratory For Particle Physics) in Geneva a new particle collider LHC (Large Hadron Collider) has been designed and is going to be built. The LHC project [1] consists primarily of the addition to the already existing LEP (Large electron positron collider) tunnel of a superconducting magnet system with two beam channels. While the LEP collides electrons and positrons with an centre of mass energy of up to 200 GeV, the LHC is designed to bring two proton beams into collision resulting in 14 TeV centre of mass collisions.

Given the size of the beam and the number of particles within the beam a probability of their collision can be given. However, not all collisions occur with the same impact. The transverse momentum p_t (momentum perpendicular to the beam direction) of the particles emerging from a collision is a measure for the power of the impact. Thus one wants to observe as many collisions with a high number of particles having as high a transverse momentum as possible.

At one of the experimental areas the experiment CMS (Compact Muon Solenoid) [2] is planned to be installed (see fig. 2.2). The aim when designing detectors for collider experiments is to cover a full 4π-area around the interaction point, in order to ensure the detection of all produced particles. As constructing a ideal sphere-shaped detector system poses problems, a cylindrical shaped form is realized to cover the interaction point.

Generally seen two types of detectors exist. One type are detectors which measure the energy of particles and are called calorimeters [3]. The second type of detectors measures the position of charged particles. Wire chambers belong to this type. They are described shortly as they are used by the track finder processor.

Wire chambers are based on the principle of ionisation. The basic configuration consists of a container with conducting walls. It is filled with a gas. Along the axis of the cylinder a conducting wire is suspended to which a positive voltage relative to the walls is applied. When charged particles traverse the chamber volume, gas molecules are ionized and electron-ions pairs are being created. Under the action of the electric field, the electrons will be accelerated towards the anode and the ions towards the cathode. Is the applied electric field high enough, an ionisation avalanche is triggered. Thus the deposited charge on the wire is amplified and can be detected by the read out electronics.

A multi wire proportional chamber comprises a number of single wire chambers arranged next to each other. It allows to cover a large measurement area.

Drift tubes are wire chambers where the drift time of the electron avalanche is used to determine the exact position of incident. The drift time is the time difference between the arrival of a particle at the chamber and the deposit of the charge at the wire. It depends on many parameters (gas, field). The resulting drift speed is in the order of 50 µm/ns.

Applying drift tubes allows to determine the position of particles with a resolution in the order of 100 µm. However, a typical problem of drift tubes is the so called left-right ambiguity of the position measurement. A single drift cell does not give any information if the particle crossed on the left or right hand side of the wire.

1.2. Data acquisition in high energy physics experiments

In the first section basic structures of data acquisitions are described, as they are used for the experiment CMS.

The second section covers basic track finding methods.

1.2.1. Data acquisition architectures

When colliding protons against each other only a small fraction of occurring interactions involve high energy quark-quark collisions. There will be a large number of so called soft collisions which are of no particular interest.

In LHC some type of interactions are supposed to be examined which have a probability to occur once in 10^{11} interactions. As experiments become more and more complex the amount of data taken for one interaction is very large. In case of CMS the number of channels is about 10^8. The resulting compressed data stream has a size of 1 MByte per event. Recording all events on a permanent storage device and performing off-line analysis at a later time to evaluate interactions of physical relevance is out of question. The data amount per second produced by CMS yields 1 MByte · 40 MHz =

$4 \cdot 10^{14}$ bit·s^{-1}. Thus it is obvious that a selection of interesting events has to be performed prior to writing the data to a permanent storage device such as a tape drive.

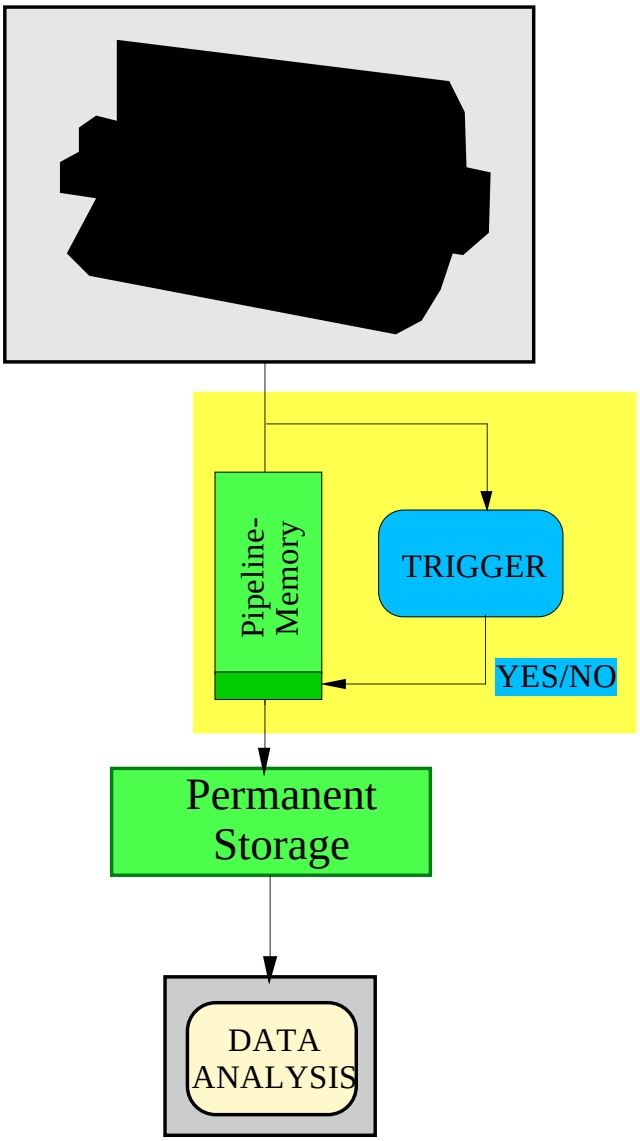


Fig. 1.1.: Data Acquisition using a trigger to select interesting data on-line.

Trigger devices evaluate the occurring interactions. They select all good event candidates and reject most of the back ground events. Features are extracted from each event. Features are variables containing the relevant physics message that can be used for the final decision of the trigger. When a decision based on the extracted features is positive the events are recorded, otherwise discarded. Features for trigger decisions can be, for example:

- track multiplicity of an event,
- direction of particles,
- deflection or curvature of particles to measure momentum,
- type of particles,
- deposited energy in the detector,
- missing energy to detect particles which leave the detector without interaction with matter (as neutrinos), but also
- event topology for expected interactions.

During the processing time of the trigger all detector data are stored in a pipeline memory. If the trigger decision is positive the data will be extracted from the buffer and written onto the permanent storage device. There it is available for off-line analysis (see fig. 1.1).

However, in modern high energy physics experiments the number of read-out channels and the complexity of the ongoing interactions do not allow to perform a trigger decision in one go using all available data in a reasonable amount of time. In CMS the number of channels is 10^8 and the required reduction rate of the trigger is as high as 10^7. That means in average only one event out of 10^7 is considered worth being stored. However, the dead time, where a trigger processes data from an event and thus is not able to evaluate subsequent events, must not occur.

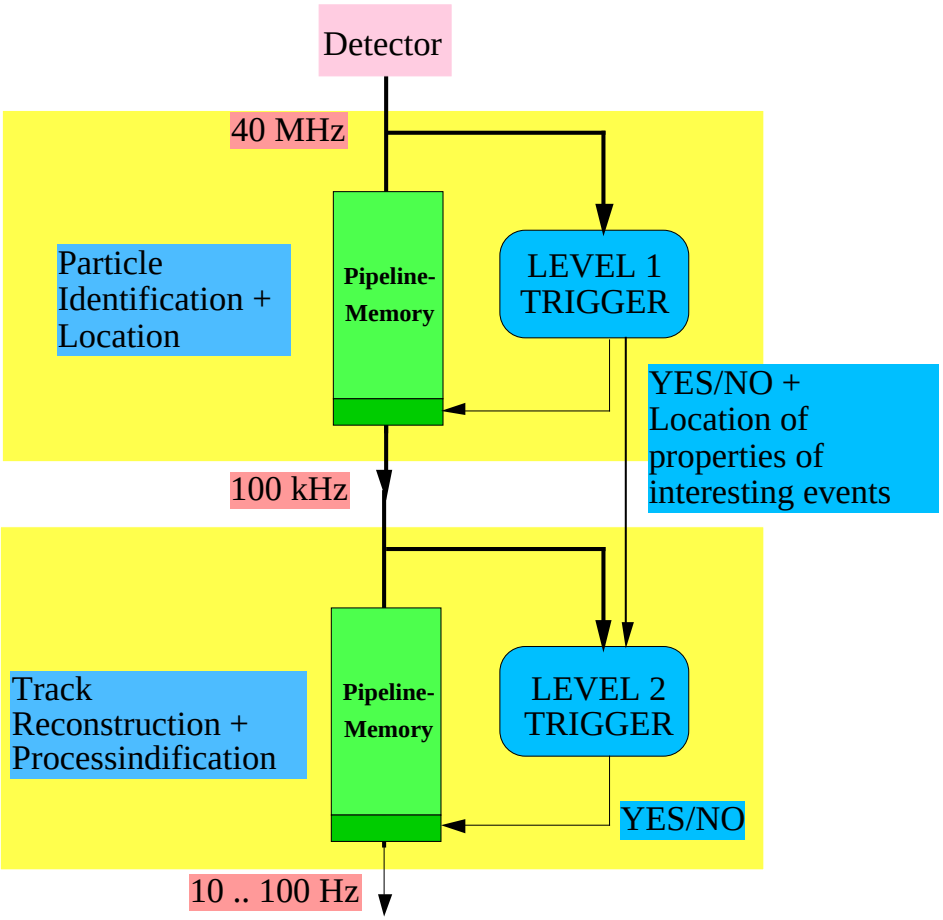


Fig. 1.2.: Two level trigger architecture.

Hence a two or more level trigger scheme is applied (see fig. 1.2). A first level trigger performs an event evaluation based on a subset of all available data. If the first level trigger releases a positive trigger decision, the event data is forwarded to the next trigger stage. In this way the event rate is reduced for the subsequent trigger level offering more decision time for this level. The higher the trigger level the more complex but also time consuming the algorithms become. However, as long as the processing time for a certain level is smaller than the corresponding bunch crossing period, no dead time is introduced to the data read-out. Deviations from the medium event rate must be compensated for by buffers between the various trigger levels.

When the bunch crossing period for a certain trigger level is smaller than the processing time, data from the subsequent interaction arrive at the inputs of the trigger before the previous event was processed. That means the trigger introduces dead time and cannot evaluate each subsequent event. In proton-proton colliders, as the LHC, where the possibility of interactions during a bunch crossing is very high, dead time of the trigger inevitably introduces inefficiencies. For lepton colliders, as the e^+e^- collider LEP/CERN, where the probability of collisions in a bunch crossing is far below 1, dead times of the trigger system usually do not cause severe inefficiencies. This is because the probability of a second interaction within the trigger processing time is low.

In case of LHC it is vital to avoid any dead time. However, the bunch crossing (bx) period is 25 ns and it is obvious that no decision, even not on the first level, can be found during that time. Two architectures are possible to come by this problem:

- ‘Round Robin’- architecture and
- pipeline architecture.

The ‘Round Robin’ architecture employs a number of parallel running processors. Each processor executes the entire algorithm for one single event. After each occurring event the data is sent to another processor which happens to be idle. When a processor finished its task it puts out the calculated data and is free to obtain the next data set. The number of processors necessary equals the execution time for one event divided by the bunch crossing period. This scheme has the advantage that commercial processors may be employed. However, the hardware effort is high, since the architecture foresees parallel running identical processors. Moreover routing the data stream into the corresponding processor again introduces propagation time and hardware effort.

A more elegant solution, especially when employing custom designed digital circuits, is the pipeline architecture. The implementation of the algorithm is partitioned in small calculation steps. The processing time of each step does not exceed the bunch crossing period. The intermediate results of each step are stored in flip-flops. When applying a central synchronous clock to the circuit equal to the bunch crossing rate (see fig. 1.3) the data stream proceeds though the structure accepting a set of input data each clock cycle and outputting a result each clock cycle. The processing time corresponds to the number of processing steps times the clock period. Most algorithms implemented in custom designed digital circuits can be splitted into sufficiently small steps. The advantage is the reduced hardware amount. No processor duplication must be performed. Higher processing times might be a result of storing the intermediate results in the latches.

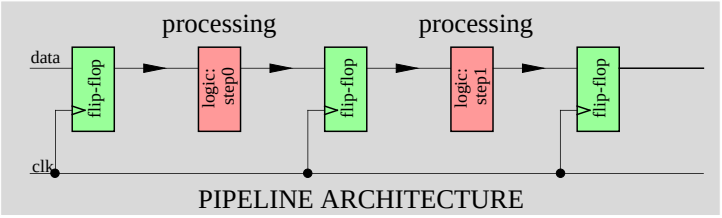


Fig. 1.3.: Pipeline architecture.

1.2.2. Track finding in high energy physics

The decision finding process of a trigger can be based on several features. However, here only the methodologies for track finding in position measuring detectors are shortly described. For more detailed information refer to [4].

A large number of methods is known and has been applied in software implementations. Only a few have been realized in hardware applications. The different methods can be classified as global or local. When comparing the available methods to each other one should consider the advantages of a hardware implementation compared to a software application. The obvious advantage is the possible processing speed of any given logic or arithmetic operation. However, the other advantage is that in hardware systems the processor architecture can be adapted exactly to the required needs. In other words one can take advantage of parallel working units. This again reduces the overall computing time. It is obvious that not all track finding methods can be implemented in hardware in an efficient way.

A local method evaluates a subset of the entire data set first and attempts to find a correlation to the remaining data. Local methods often have to make fruitless attempts in order to find track candidates and thus use the same data entity in different combinations. When used in software applications the computing time increases more rapidly than linearly with the number of data entries. However, when implemented in hardware different combinations can be evaluated in parallel. This requires to foresee a number of parallel working hardware blocks. The number of hardware blocks also increases more rapidly than linearly with the number of data entries. Since different combinations are processed in parallel computing time does not increase necessarily with the number of data entries.

A global algorithm evaluates the entire data set at once. The computing time of a global method should be proportional to the number of data entities (points) in the event. Applied in a hardware application a global method is very efficient as long the data set is reasonable small. However, as data set size increases due to the high number of inputs and the necessary interconnect within processing units a global method requires a high number of processing units.

1.2.2.1. Local methods

Track following method

An initial track segment is selected. The next step is to predict a point by extrapolation into the next chamber. The extrapolation may be of zero order simply by choosing the nearest neighbour, first order (straight line), second order (parabola), and possibly higher orders, or other track forms such as circles or helices.

In high resolution detectors where a large number of possible points may occur applying this method in hardware can be time consuming or requires a high number of processing units. In a hardware implementation two options are given. Either one attempts to extrapolate from one point to the points in the adjacent chamber one after the other (in the same way as it is done in software) or one attempts to extrapolate from one point to all other points at the same time in parallel. In the first case the advantage of parallelisation is entirely lost and thus a higher processing time will be the result. In the latter case the possibility must be provided to extrapolate from one point to all other points at the same time. However, if the number of possible extrapolation target points exceeds a reasonable amount a combination of the two options might have to be considered.

Track road method

An interpolation between points is used to predict extra points on the track. By using initial points at both ends of a track - and possibly in the centre of curved tracks - a simple model of the track is used to predict the positions of further points on the track, by defining a road around the track model. If a sufficient number of hits is going to be found to be within this road a track is considered to be formed.

The track road method is slower than the track following method. Sometimes it is the only workable method available, particularly in the case of widely spaced detector planes, where the redundancy in the coordinate measurement can be very low. However, even when there is sufficient redundancy the road method can sometimes be

superior in performance and speed when compared with the track following method, for example in drift chambers with left-right ambiguity.

However, parallelisation of hardware processing units may prove to be cumbersome when the maximum number of possible reference points from which the track model is derived from exceeds a certain limit. The track road method can be applied efficiently in hardware when points in a reference plane with a low multiplicity are available. For instance this could be the outermost plane and the interaction point. However, being dependent on the proper functionality of a specific detector plane is a severe restriction to the track finding performance.

Track segment method

The track candidates are formed in two steps. Short track segments are built from points, normally inside sub divisions of the detector. As a result one obtains the position of the cluster and a direction value (track segment). Several track segments are connected to each other using track following or track road methods.

The great advantage of this method is its speed, compared to using all points per track directly. It is therefore appropriate for detectors with a very high point density. In addition, the left-right ambiguity of drift chambers can be resolved at the track segment level.

From point of view of hardware implementation the track finding method has an important feature, namely reducing the data set size (by creating a fixed number of track segments) while maintaining the full detector resolution. As a result the number of points to be processed for the actual track finding algorithm is reduced. Track segments in different detector regions can be formed in parallel. Then a track following or track road method is applied on the reduced data set.

1.2.2.2. Global methods

Combinatorial method

All measurements giving a possible track candidate are fitted to a track model. If the requirements for a valid track are met, a track is found. This method is very time consuming. Applied in a hardware system this is only a viable solution when the maximum number of possible points is low.

Histogramming method

A set of *n* functions of the measured hit coordinates is defined. The function is chosen in a way that, applied to hit coordinates of the same track, it delivers identical (or at least similar) values. In most times it is attempted to define functions which directly yield the desired property of the track. The function values are entered in a n-dimensional histogram. Hit coordinates triggered by one track will cause a peak in the histogram.

This method is known to be used intensively in software applications. Serially the software processes count the entries into the histogram. Afterwards the filled histogram is evaluated. Employing this method in hardware using parallelisation proves cumbersome when the dimensions of the histogram and the maximum number of

entries get too large. Assuming a two dimensional histogram of the size 20 times 20 with a maximum number of entries of 16 already requires 400 counters each accepting up to 16 entries at the same time.

Pattern matching

A predefined set of valid track patterns is compared to measured values. If a match is detected a valid track is found. The method can only be applied if the number of patterns to be stored and compared can be kept within reasonable limits.

In hardware applications this method is known to be applied intensively. It is a fast and efficient method. However, as detector resolution and number of detector cells get too high the number of necessary inputs of the hardware system and the number of patterns too store can get prohibitively high.

Neural artificial networks

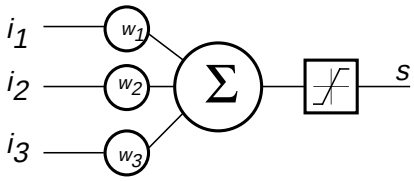


Fig. 1.4.: Block diagram of a neuron.

The sensory and cognitive abilities of biological neural networks, like the human brain, are still not reached by even the most powerful electronic systems. However, recently more and more applications to implement neural networks are being undergone. The neuron is an information processing unit that is fundamental to the operation of a neural network. Fig. 1.4 shows the model of a neuron: Input signals are fed to a set of synapses or connecting links, each of which is characterized by a weight of its own. An adder sums up the weighted input signals and provides it to the activation function. This limits the amplitude of the output of a neuron. A layered neural network is a network of neurons organized in the form of layers. [5,6]

2 Track finder processor environment

2.1. Experiment CMS - Compact muon solenoid

In this chapter the experiment is described in a simplified way in order to support understanding of the environment and specifications of the track finder processor. More detailed information about the detector may be found in [2, 7, 8].

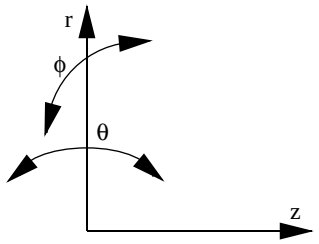


Fig. 2.1.: System of coordinates.

Fig. 2.1 illustrates the system of coordinates for the detector used in the following chapters. The angle θ frequently is replaced by the pseudo rapidity η. Equation 2.1 shows the definition for η. For directions perpendicular to the beam axis η equals 0. Directions parallel to the beam axis have a pseudo rapidity η of +∞ or -∞.

$$\eta = -\ln\left(\tan\left(\frac{\theta}{2}\right)\right) \qquad \text{(Eqn. 2.1.)}$$

The detector will be built around a high-field superconductive solenoid leading to a compact design for the muon spectrometer, hence the name Compact Muon Solenoid (CMS). The solenoid has an inner radius of 3 m generating a uniform magnetic field of 4 T parallel to the beam axis. The magnetic flux is returned through a 1.8 m thick iron yoke instrumented with muon chambers. The magnetic field in the return yoke is 1.8 T. Fig. 2.2 shows a three-dimensional view of CMS. The overall dimensions of the detector are: a length of about 20 m, a diameter of 14 m and a total weight of 12000 tons. The total number of measurement channels is about 10^8.

From the inside of the detector at the interaction point to the outside particles traverse

- the beam pipe,
- the inner tracker,
- the crystal electromagnetic calorimeter,
- the hadronic calorimeter,
- the superconductive coil, and
- muon chambers and return yoke.

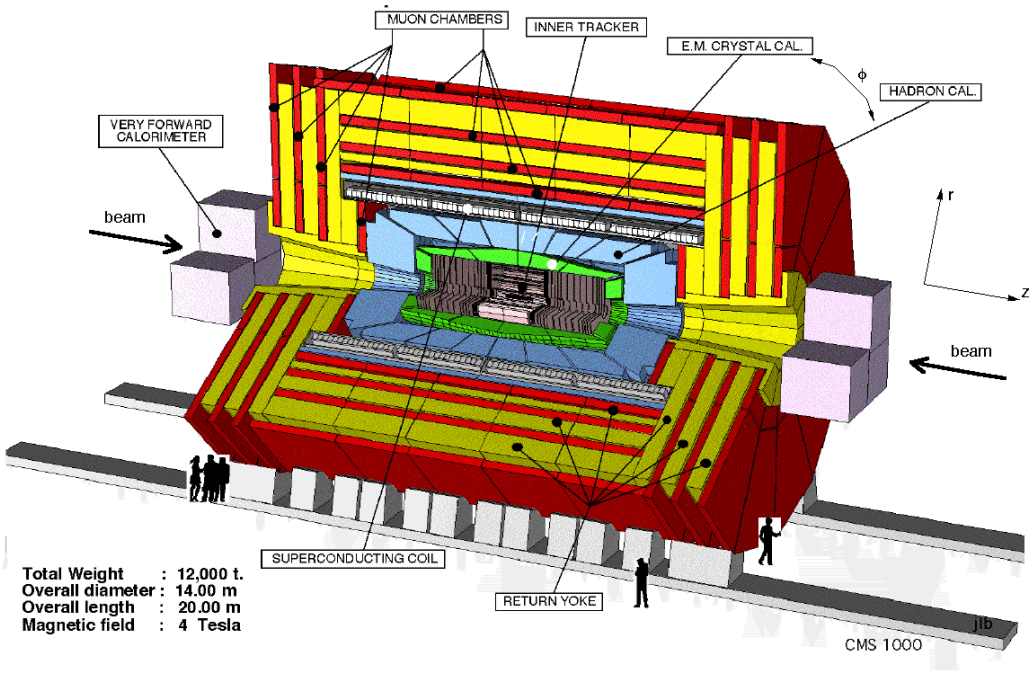


Fig. 2.2.: Three dimensional view of the detector CMS (Compact Muon Solenoid).

The inner tracker provides accurate position information of all charged particles. The number of particles and superimposed tracks close to the interaction point require the inner tracker to have a high granularity in order to be able to ensure pattern recognition. The three employed detector systems, silicon pixel detectors, microstrip detectors and micro strip gas chambers, achieve a spatial resolution in the order of 15 to 40 µm.

The electromagnetic calorimeter measures the energy of photons and electrons. It is built using lead tungsten ($PbWO_4$) crystals arranged in a projective geometry with respect to the interaction point. The crystals have a physical size of 23 cm length and 20.5 x 20.5 mm^2 square section on the front face. The size of the front face corresponds to the spatial granularity of the detector.

The hadronic calorimeter surrounds the electromagnetic calorimeter and acts in conjunction with it to measure the energies and directions of hadrons and to help measuring the missing energy. The active elements of the calorimeter consists of plastic scintillator tiles. Layers of these tiles alternate with layers of copper absorber to form the sampling calorimeter structure. The tiles are arranged in projective towers with fine granularity. The granularity is 160 x 160 mm^2.

The muon detector fulfils three basic tasks: muon identification, trigger, and momentum measurement. The muon detector is placed behind the calorimeters and the magnet coil. It consists of four muon stations interleaved with the iron return yoke plates. A system of drift tubes (DT) is applied in the barrel region, while cathode strip chambers (CSC) cover the forward region. In addition resistive plate chambers (RPC) cover the entire muon detector.

The barrel part of the detector is divided in five wheels z-direction and twelve 30° sectors, resulting in 60 (ηϕ) detector segments (see fig. 2.3.a). Each sector comprises four measuring stations, MS1 to MS4. Every station contains a module of drift tubes.

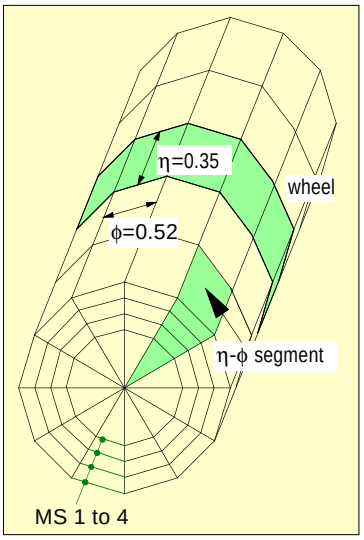


Fig. 2.3.a: Barrel muon detector segmentation.

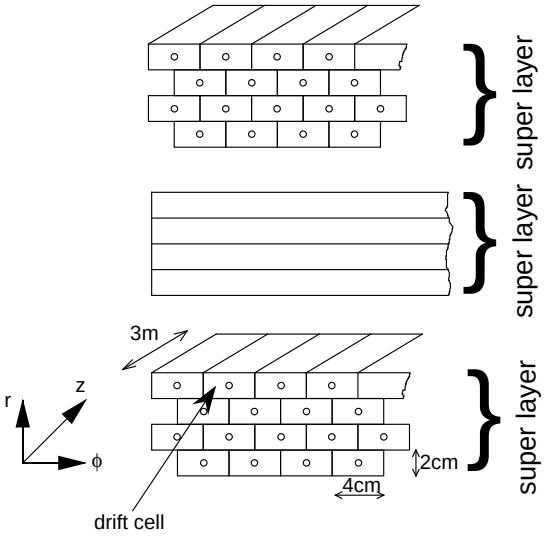


Fig. 2.3.b: Three superlayers form a drift tube station.

A drift tube cell has a cross section of 4 x 1.1 cm^2. The drift time is about 400 ns. A spatial resolution of about 200 μm is achieved. The total number of drift wires is about 200000. Fig. 2.3.b illustrates the arrangement of the drift tubes. The chambers consists of twelve layers of drift tubes arranged in three superlayers of four planes each. The outer two superlayers (rϕ) measure the ϕ-coordinate in the bending plane. The middle superlayer (rz) measures the z-coordinate along the beam line. The chambers are 2.56 m long and the width increases from 2 m in the inner station to 4 m in the outer station. The tubes within a superlayer are staggered by half the width of a tube.

In the endcap regions the muon detector comprises four muon stations. Muon stations are mounted vertically (fig. 2.3). Each muon station contains cathode strip chambers. The cathode strip chambers contain six layers with cathode strips oriented radially to measure the azimuthal coordinate. The wires are strung perpendicular to the central radial line. [2]

2.2. Trigger and data acquisition of CMS

The LHC environment presents challenges to the trigger and data acquisition which are far more demanding than those encountered by the past and present experiments at other facilities. [9]

These advanced demands arise because of a

- high bunch crossing rate, 40 MHz
- high channel number in CMS, >10^8 channels

- high collision rate (up to 18 interactions per crossing), 10^9 Hz
- high complexity of the events, overlap and pileup
- and high reduction rate of the trigger. 10^7

The bunch crossing (bx) rate in LHC is 40 MHz, corresponding to 25 ns between two subsequent crossings. On average there will be 18 interactions per bunch crossing, causing overlap of the interactions within one bunch crossing, but also interfering with subsequent interactions (pileup). Thus the resulting interaction rate is about 10^9 interactions per second. The trigger and data acquisition must reduce the input rate by a factor of 10^7 to 100 Hz. This is the rate permanent storage devices, such as tape drives, are able to cope with. [2]

CMS has more than 10^8 channels. Considering the raw data from all the detectors and the bunch crossing rate of 40 MHz the resulting data rate for the entire experiment is in the order of 10^{15} bit s^{-1}. This data rate corresponds to an amount of data processed by $2 \cdot 10^9$ simultaneous playing compact disc players. Assuming a large music store has 100000 compact discs in stock 80000 stores would have to play all their discs at the same time to generate an equal data rate as caused by the detector CMS [10]. Table 2-1 shows the number of channels and the resulting amount of data after zero suppression for each of the detector systems. Even after zero suppression the data size for one crossing is still 1 MByte, resulting in a compressed data rate of $3.4 \cdot 10^{14}$ bit s^{-1}. The occupancy gives the average fraction of detector channels whose detector cells are being hit.

Detector	No. of channels	Occupancy [%]	Event size [kB]
Pixel	$8 \cdot 10^7$	0.04	100
Inner Tracker	$1.6 \cdot 10^7$	0.8	700
Calorimeters	$6.4 \cdot 10^5$	5-10	100
Muons	10^6	0.1	10
total	$9.8 \cdot 10^7$	-	~1000

Table 2-1: Number of channels, occupancy and zero suppressed events size of detectors.

The data reduction rate is performed in three steps. The three steps form a series of progressively more complex, but also more time consuming levels. By reducing the event rate, each level offers the subsequent level more time for its decision. Although the data rate is reduced in three steps, only two physical units (physical levels) are employed. The first level is based on custom, pipelined hardware processors. The second and third level are based on network switches and processor farms.

The level-1 trigger comprises the front-end electronics which generates the trigger primitives (detector data for the trigger decision). The level-1 trigger operates on a subset of data (muon detector and calorimeters). The event rate is reduced to 100 kHz. The entire trigger system operates dead time free. That means the first level trigger generates a decision for each event every 25 ns. Due to the limited storage capacity of detector read-out the decision must be available 3.2 µs after the corresponding bunch crossing.

The level-2 trigger is provided by an on-line processor farm and reduces the trigger rate by a factor of 10 to 10 kHz. After a positive Level-2 decision, the remainder of the full crossing data is requested for further processing by this farm for the final (Level-3) decision, which puts out a positive decision with a frequency of up to 100 Hz. [2]

In the first level trigger only a subset of data is used to identify the particles and perform an energy evaluation on a macro-granular detector information. In level two the information about location and type of particle is known from level-1 and a finer granularity is used to refine the level-1 measurements. Employing track reconstructions and more precise energy measurements permits evaluating the event topology. In level-three a full process identification is conducted.

No high energy physics experiment in the past had as many channels as CMS. Moreover, none of the previous experiments worked at a bunch crossing frequency of 40 MHz. See table 2-2 for a comparison of the channel number, event rate, bunch crossing rate, event size, and data rate for the experiment UA1 (SPS/CERN), H1 (HERA/DESY), and CMS (LHC/CERN) [10,12].

Experiment	**UA1**	**H1**	**CMS**
Tracking [channels]	10^4	10^4	10^8
Calorimeter [channels]	10^4	$5 \cdot 10^4$	$6 \cdot 10^5$
Muons [channels]	10^4	$2 \cdot 10^5$	10^6
Bunch crossing rate [ns]	3400	96	25
Raw data rate [bit·s^{-1}]	10^9	$3 \cdot 10^{11}$	$4 \cdot 10^{15}$
Tape write rate [Hz]	10	10	100
Event size [byte]	100k	125k	1M

Table 2-2: Data acquisition parameters for UA1 (1982), H1 (1992) and CMS. [10]

2.3. CMS first level trigger

The task of the CMS first level trigger is to release the L1 accept signal. In case of a positive decision the detector data are extracted from the data pipeline and forwarded to the second level trigger. In case of a negative decision they are discarded. The requirements on the hardware implementation are severe. Every 25 ns another event has to be evaluated. The L1 trigger processing time is limited to 128 bx.

The CMS First Level (L1) Trigger has to reduce the data rate from 40 MHz to less than 100 kHz. This yields a rejection rate of 400. The data reduction is based on the recognition of primitive physical objects such as leptons, hadrons and photons. The final decision is made as a function of the event topology and cuts on the kinematical parameters transversal Energy E_t and transversal momentum p_t. The L1 trigger has to process its algorithms in a fixed time. The processing time must be independent of the occurring event. The large amount of data to be processed by the L1 trigger requires

more time than one bunch crossing. Moreover the L1 trigger must not introduce dead time to the trigger system. Hence the trigger has to operate in a pipelined way. In order to render the size of the data pipeline within reasonable limits the calculation time of the L1 trigger is limited to 128 bunch crossings (bx) or 3.2 μs. The strict requirements on L1 trigger latency do not permit to use tracker information in the L1 trigger. Hence only muon detectors and calorimeter data participate in this system. Still the number of channels to be processed in each crossing is higher than 10^6.

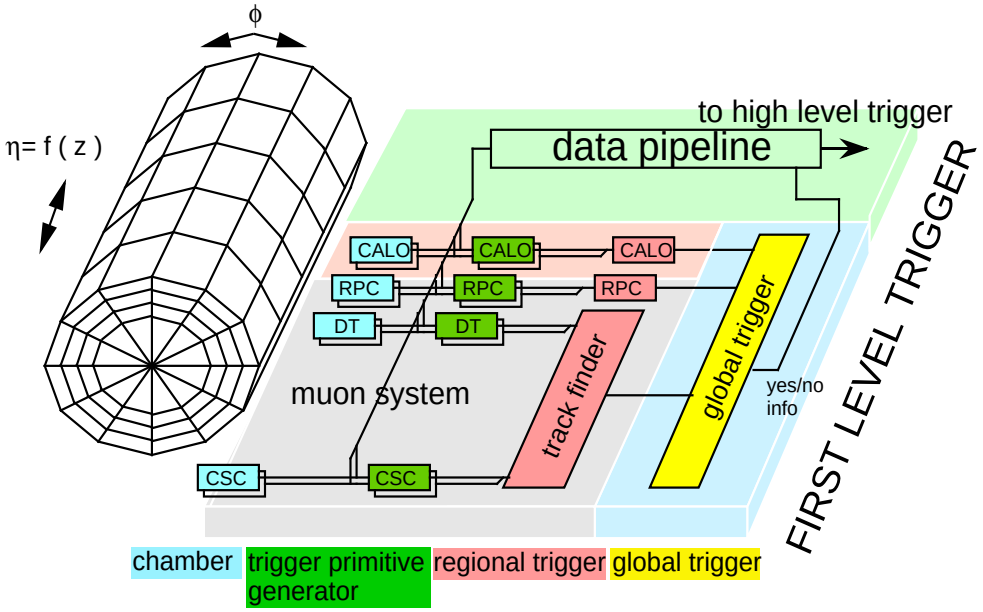


Fig. 2.4.: Block diagram of CMS First Level Trigger.

The complexity of the events does not allow to process all the input data in one logical unit. Fig. 2.4 shows a block diagram of the CMS L1 trigger. Four main blocks can be seen, the calorimeter trigger, the muon trigger, the global trigger and the data pipeline. The calorimeter trigger measures the energy of hadrons, electrons and photons. The muon trigger determines the location and the transverse momentum p_t of muons. The main principle of the L1 trigger is that none of the sub triggers (muon, calorimeter) applies any cuts on the detector data. It is the global trigger which combines the sub trigger information and finds a final decision.

Both sub trigger systems have a common structure. They can be subdivided into three groups; the trigger primitive generators (TPG), the regional triggers, the global trigger. The trigger primitive generator processes the digitized chamber raw data. It performs zero suppression and forwards the data to the regional triggers. These units attempt to find particles and perform energy or momentum measurements. The information of both sub triggers is combined in the global trigger. It releases the ‘level 1 accept’ signal.

2.3.1. CMS global trigger

The global trigger combines the information from the calorimeter trigger and the muon trigger and delivers the final trigger decision (L1 accept signal). The L1 accept signal is transmitted to the data pipeline in order to discard the data of the corresponding event or to forward all detector information to the higher level triggers. Additionally the

information which the L1 trigger decision is based on, including all sub trigger output data and trigger settings, is made available to the second level trigger. This information serves the second level trigger to identify regions of interest.

2.3.2. CMS level 1 trigger latency

The level 1 trigger latency is a basic parameter of CMS. It determines the amount of data storage that must be provided in the front end electronic systems, which are located near the detector. Specification of an unnecessarily long latency can make the design of the data pipeline more complex and costly. Specification of too short a time can make it impossible for some detector system to contribute fully to the trigger, or, even worse, can mean that the data from some detector cannot be captured for subsequent read-out.

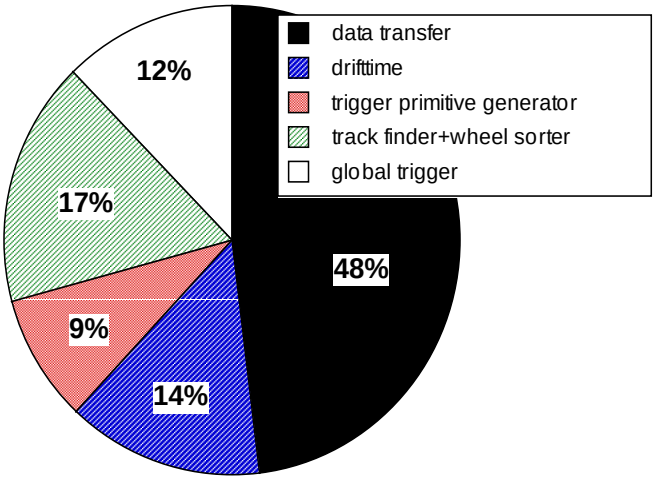


Fig. 2.5.: Latency distribution of the drift tube muon trigger system and global trigger.

There are many contributions to the latency. These include time of the flight of the particles to the detector - this number can be as high as 30 ns; propagation of signals within the sensitive elements of the detector - in case of drift tubes up to 400 ns; signal processing and trigger primitive generation times; cable runs within the detector hall and from the detector hall to the control room - the longest expected cable run is as long as 90 m (450 ns); time to process regional trigger algorithms; time to make the global trigger decision; time to distribute the L1 trigger accept signal back to the crates in the front end - 90m (450 ns); and time to distribute the signal to the relevant locations within the crates. Fig. 2.5 shows the distribution of the latency caused by processing time, cable delay and synchronization for the track finder system and global trigger. 48 % of the latency is due to cable delays, synchronization and the transmission from the detector to the processing units in the electronics barracks and vice versa. The charts show that an appreciable portion of the latency is in fixed cable delays.

2.3.3. CMS muon trigger

The muon trigger performs only measurements on the particles found in the muon system. It does not make a decision whether the detector data of a given event is to be discarded or to be retained. The tasks of the CMS muon trigger are to identify muons, determine their location, determine the bunch crossing in which they occurred and measure the transverse momentum. The system is based on dedicated trigger detectors (resistive plate chambers) and precision muon chambers (drift tubes and cathode strip

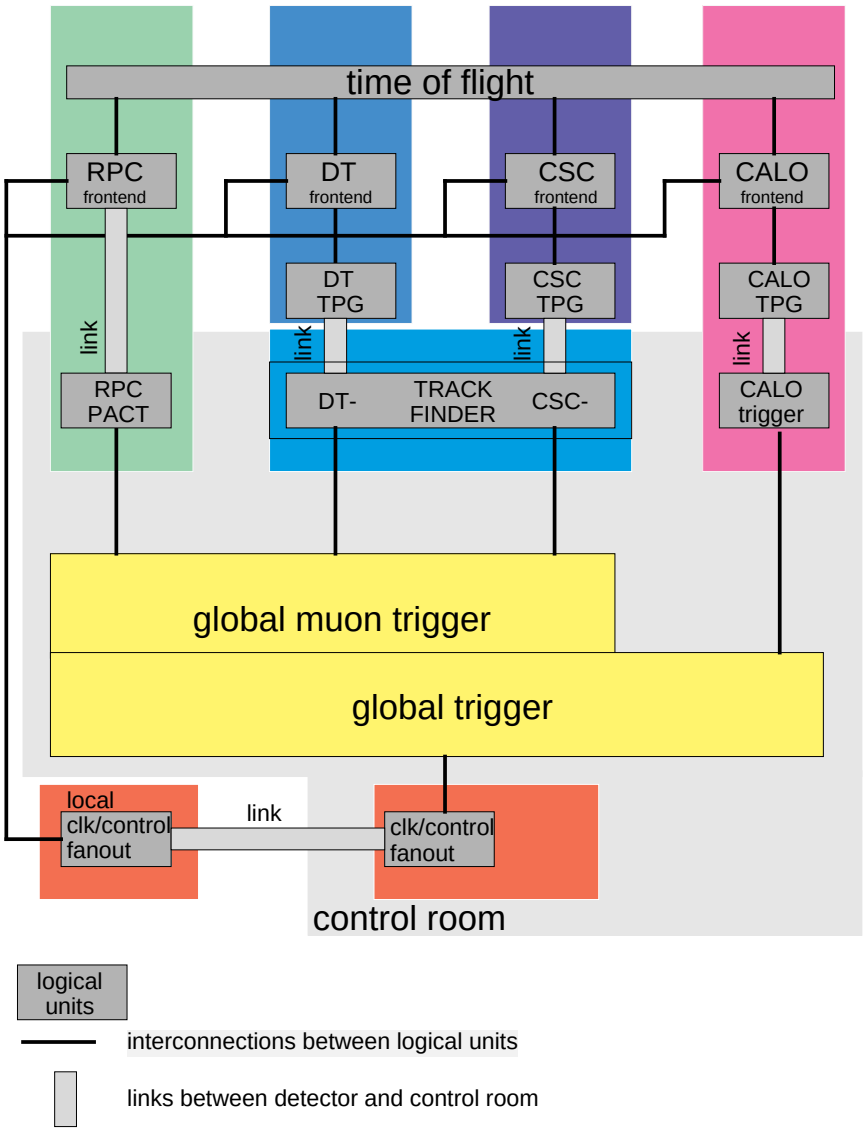


Fig. 2.6.: Detailed first level trigger block diagram

chambers). The RPC system as well as the muon chamber trigger system are designed to work independently of each other. In most large scale high energy physics experiments fast dedicated trigger chambers recognize a muon and precise muon chambers provide a sharp momentum cut. The possible difference in CMS is that often in other experiments muon chambers are used in the second level trigger whereas in CMS the chamber information is already used in the first level trigger. This is well suited to the overall concept of the CMS trigger which does not have a dedicated level two hardware. This is possible due to high integration of logic processing units and fast calculation speed of today's electronic components. Since the two trigger components work in the same level one can take advantage of their complementarity with respect to their measurement performance. For further information about advantages of the two component muon trigger system refer to [11].

The system has to cover the entire geometrical range of the muon system of up to |η|=2.4. The latency of the system must not exceed 128 bunch crossings. The system must not introduce any dead time to the detector read-out. Both systems put out information about muons with the highest transverse momentum p_t in the entire detector. [13]

The CMS muon trigger comprises three main components; the pattern comparator trigger (PACT) [2,14,15] based on resistive plate chambers (RPC), track finder processor (TF) based on drift tube chambers (DT) and cathode strip chambers (CSC) and the global muon trigger. A detailed block diagram of the L1 trigger is shown in fig. 2.6. All blocks on the shaded background are located in the control room 90 m apart from the detector.

DT- and CSC-trigger primitive generators first process the information of the chamber locally. As a result up to two vectors per muon station, the track segment (position and angle) is delivered. Track segments from different stations are collected by the track finder which combines them to form muon tracks. The transverse momentum p_t is assigned. The track information is forwarded to the wheel sorter which selects the four highest p_t muons in each detector wheel.

2.3.3.1. Drift tube trigger primitive generator (DT-TPG)

The task of the drift tube trigger primitive generator is to perform a data compression

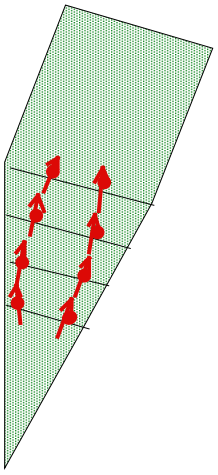


Fig. 2.7.a: Two track segments per chamber are given out.

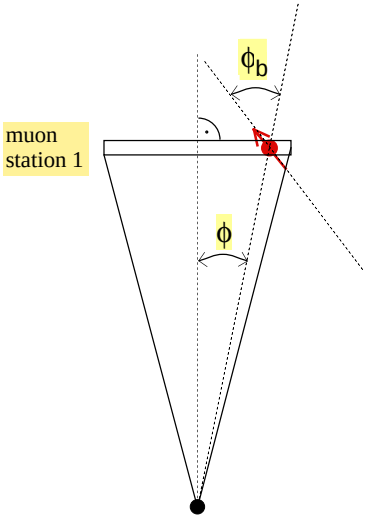


Fig. 2.7.b: A track segment consists of the spatial coordinate ϕ and the crossing angle ϕ_b.

for further processing by the track finder. Since the probability to find more than two tracks in a detector segment is negligible [16] the primitive generator suppresses data from all drift cells which are not hit (zero suppression) and puts out information of up to two track segments per chamber in each detector segment (see fig 2.7.a). A track segment consist of a position and an angle measurement. The position gives the location of the hit in the chamber. The angular value is the angle of the crossing track with respect to the detector radius (see fig. 2.7.b).

In case more than two track segments are found in a chamber the drift tube electronics selects those two track segments with angles closest to the radial direction (muons with high momentum have a low bend angle).

The maximum drift time of the drift tube chambers in CMS is about 400 ns or 16 bx. Thus a single drift cell does not allow to identify the bunch crossing in which a particle crossed the muon detector. This problem is solved by applying the so called mean timer method:

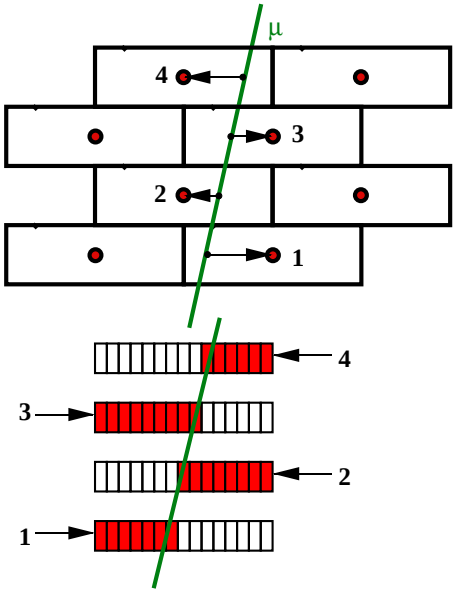


Fig. 2.8.a: illustrates the meantimer technique.

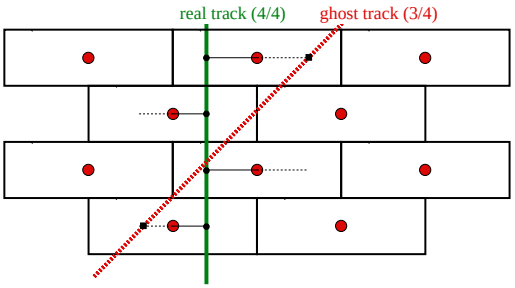


Fig. 2.8.b: The example shows a real track and a ghost track.

Each of the four muon stations in the barrel is equipped with twelve layers of drift tubes [2]. They are arranged in quartets called superlayers SL. Two of the superlayers measure the rϕ-coordinate, one measures the z-coordinate. The drift tubes of a superlayer are staggered (see fig. 2.3.b). The tracks in each superlayer are identified by bunch and track identifier BTI [17]. The signals from four drift tube layers are each connected to a register. They are clocked with the bunch crossing frequency and have a depth equal to the maximum drift time T_{max}. The wire signals are shifted in the same direction as the electrons drift in the chamber. The hits will be aligned in the shift register at the time T_{max} after the corresponding bunch crossing. The aligned 'shift register images' reflect the images of the tracks (see fig. 2.8.a). Thus the track position and direction is given and can be calculated. The position resolution is 1.25 mm and the angular resolution 60 mrad. However, possible inefficiencies in the chamber system require to accept track images consisting of only three out of four hits. This and the 'left-right' ambiguity of drift tubes makes it possible to find ghost tracks. In fig. 2.8.b such a ghost track is illustrated. Both the real 4 out of 4 track and a ghost 3 out of 4 track will be found. 4 out of 4 hits are marked as high quality triggers (HTRG). Conservatively a low quality trigger (LTRG) indicates 3 out of 4 tracks.

BTIs find track segment candidates. The track correlator TC attempts to form a correlation between the track segments of the inner and outer superlayer. Amongst all track segments the track server TS finds the ones exhibiting the lowest angle closest to the radial direction.

The BTI is used in both chamber views, rϕ and rz, to find track segments giving at least three out of four hits. A wire cluster of nine wires is connected to the BTI. The principle of correlation is shown in fig. 2.9. The angles of track segments in the outer and in the inner superlayer are compared to the computed value using the positions of the two found track segments. If the angles are found to be equal within a programmable tolerance the correlation will be considered successful and the correlation quality bit of the track segment will be activated. Only one track segment (position and angle) is given out. Due to the correlation the track angle can be calculated with a more accurate precision of 10 mrad using the spatial coordinates of the correlated track segments.

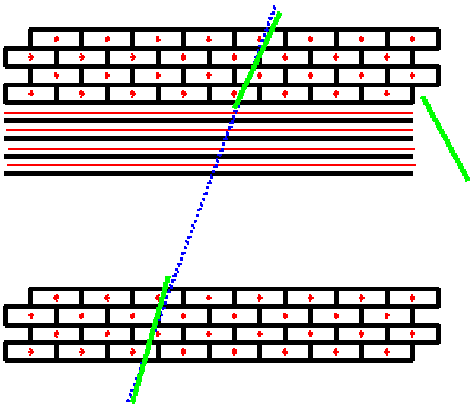


Fig. 2.9.: Correlation between track segments in the inner and outer superlayer of a station.

If the track segments cannot be correlated the track segment with the higher quality will be given out. Uncorrelated high quality triggers are forwarded without any filtering. Low quality triggers are filtered, because ghost tracks belong to this class. The ghost suppression is performed by making a time coincidence with the track segment from the rϕ-plane and the track segment of the rz-plane.

The last stage is the trigger server. Amongst all track segments the trigger server finds the ones exhibiting two track segments with angles closest to the radial direction again. The track segments are sent to the track finder. The spatial value is transmitted in an eleven bit word. Eight bits are reserved for the angle.

The DT-chamber system is divided into twelve ϕ-segments, five wheels in z-direction and four stations in r-direction, see fig. 2.2. Thus the entire system comprises 12 x 5 x 4 = 240 stations. Hence 480 track segments are delivered to the regional trigger, the track finder.

2.3.3.2. Cathode strip chamber trigger primitive generator (CSC-TPG)

A description of the CSC-TPG cannot be given at the present moment. Design is not advanced sufficiently. However, output format (track segments) is identical to the drift tube trigger primitive generator. Baseline information can be found in [18,19].

3 Track finder processor overview

In chapter 3.1. the track finder specifications are discussed.

Chapter 3.2. gives an introduction to previously implemented and planned first level triggers in comparable experiments in the whole world. Comparisons between the track finder specifications and the features of other first level trigger systems are conducted.

3.1. Track finder processor specifications

In this chapter the track finder processor specifications are discussed. Although they are listed here one after the other the process of optimizing the appropriate features was determined by a long adaptation phase. Some of the requirements and parameters influenced each other. Some of the specifications had to be refined in a later design phase.

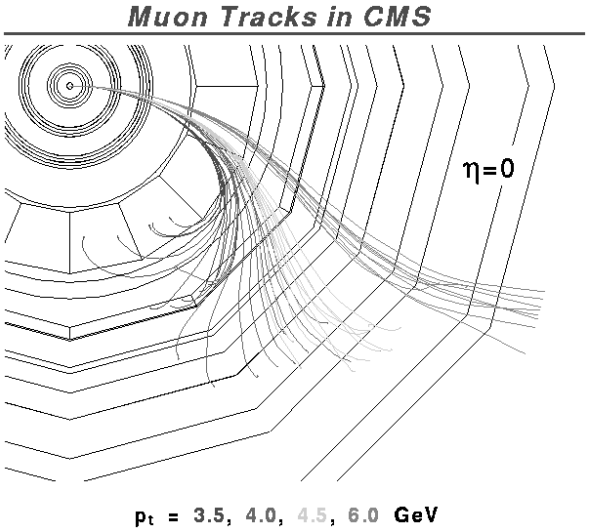


Fig. 3.1.: Muon tracks with different p_t-values in the magnetic field of CMS. Due to the change of magnetic field direction in the return yoke of the magnet the curvature of muon tracks changes.

The task of the track finder processor is to find muon tracks originating from the interaction point and to measure transverse momentum p_t and their location in ϕ and η. The system is based on the drift tube chamber system in the barrel of the detector. However, it must be designed as flexible as possible in order to allow an application in the forward region, where CSC deliver the hit information.

In fig. 3.1 muon tracks are illustrated. As shown muons with a transverse momentum of as low as 3.2 GeV/c (for η=0 and c velocity of light) reach the muon detector. In the forward region even muons with transverse momenta of 2.0 GeV/c penetrate the muon stations. Clearly

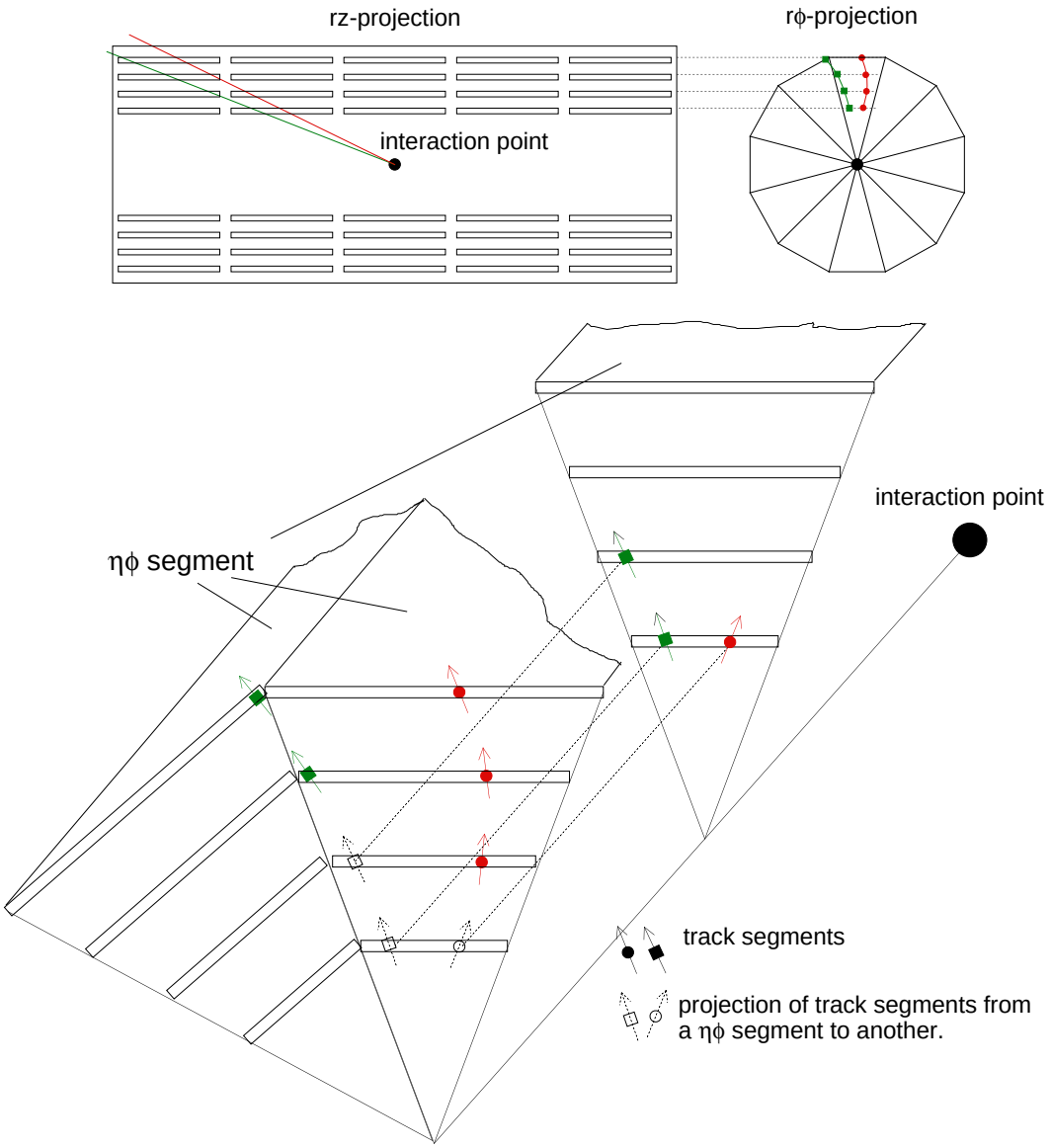


Fig. 3.2.: shows muons in the detector crossing segment boundaries in ϕ as well as in η. The three dimensional view of the detector segments and the track segments illustrates that the track finder processor first has to generate the projection of the track, respective its track segments, onto the rϕ-plane in order to obtain all track segment data of a certain track. This includes merging neighboring ϕ-segments as well as neighboring η-wheels.

one can see that low momentum muons are deviated very strongly by the magnetic field and thus cross detector segment boundaries quite regularly. On the other hand high p_t muons resemble almost a straight line and thus pose a challenge to the track finder to determine the track's curvature and its momentum. As illustrated in fig. 3.2 the track finder has to find matching track segments, link them together to a full track and measure their momentum. Hereby only muons emerging from the interaction point should be taken into account.

P_t-measurement

The second level trigger of CMS is designed to process events with a rate of 100 kHz at the maximum. If the first level trigger output rate exceeds this rate the second level trigger will fail to process events. As a consequence it is vital to be able to limit the first level muon trigger rate. Simulations for physics events producing muons have been

conducted. They allow to estimate the rate of muons in the detector in dependence of their transverse momentum p_t [2, 22].

Approximately one half of the level one trigger bandwidth is expected to be filled by triggers involving the calorimeter system and the other half is expected to be filled by triggers involving the muon trigger system. However, since an even division of level one bandwidth among sub triggers is not anticipated, flexibility to accommodate for variations in rates of the sub triggers is planned. In order not to exceed the maximum output rate foreseen for the muon sub triggers one selects only high transverse momentum muons and cuts off low transverse momentum muons just to obtain the required rate. Due to uncertainties of the estimates, changing beam conditions and different trigger settings it is necessary to be able to set a muon transverse momentum threshold in a range between 2.0 GeV/c and 140 GeV/c. The rates are also dependent on the beam condition. One expects the rate estimations to be accurate by a factor of two. The p_t-range between 2.0 GeV/c and 140 GeV/c is divided into 25 p_t-classes (see table 3-1). The step from one p_t-class to the next corresponds to a change in output rate of factor 1.5 [23,24]. This factor is quite well suited to accommodate the rate estimation uncertainties. A factor exceeding the value two would cause inefficiencies in output rate adaptations. The p_t-value is given out in a 5 bit number. A sixth bit indicates the charge of the particle.

As the number of events to be recorded is limited one wants to capture only those events containing high transverse momentum muons. This requirement corresponds to a sharp cut off at the momentum threshold. That means one wants to record only a little number of muons with transverse momentum lower than the threshold and vice versa. In other words one wants to have a transverse momentum measurement resolution as good as possible.

The relative measurement resolution (standard deviation) σ_{pt}/p_t of the transverse momentum measurement is a trade-off between technical feasibility and physics requirements. The transverse momentum p_t is derived from the curvature κ_c of the muon tracks. The resolution of κ_c is directly proportional to the resolution of the measured bend angle derived from the track segment coordinates [20, 21]. The chamber can deliver a resolution of as low as 200 µm [2]. Obviously this accuracy cannot be used in the trigger level. A spatial measurement resolution σ_{ts} of 0.3 mrad is considered implementable in hardware [16,17] and is provided for the track segment measurements by the trigger primitive generators. 0.3 mrad corresponds to approximately 1.25 mm in station one and two and 2.5 mm in station three and four. Using a spatial resolution of σ_{ts} enables to achieve a p_t measurement resolution σ_{pt}/p_t in the order of 35 %. For further details refer to [22, 25, 26].

ϕ-measurement

Measurement in ϕ can be given out very accurately since the position of the track segments in ϕ is known well. The 1.25 mm binning of the track segments corresponds to a ϕ-resolution in the first station of 0.018° or 0.3 mrad. That means the track position in ϕ can be given in the same resolution. The first level global trigger cannot process the sub trigger data with such a good resolution. Thus the track finder processor partitions the total of twelve detector segments into 256 bins using an eight bit number. Hereby a ϕ-resolution of 1.4° or 25 mrad is provided.

p_t range	5 bit-p_t code	p_t range	5 bit-p_t code	p_t range	5 bit-p_t code	p_t range	5 bit-p_t code
no muon	0	4.0-5.0	8	17.0-20.0	16	70.0-80.0	24
reserved	1	5.0-6.0	9	20.0-25.0	17	80.0-100.0	25
reserved	2	6.0-7.0	10	25.0-30.0	18	100.0-120.0	26
reserved	3	7.0-8.0	11	30.0-35.0	19	120.0-140.0	27
2.0-2.5	4	8.0-10.0	12	35.0-40.0	20	140.0-∞	28
2.5-3.0	5	10.0-12.0	13	40.0-50.0	21	reserved	29
3.0-3.5	6	12.0-14.0	14	50.0-60.0	22	reserved	30
3.5-4.0	7	14.0-17.0	15	60.0-70.0	23	reserved	31

Table 3-1: p_t-classes [23].

η-measurement

The non-projective chamber geometry with respect to the rz-view does not allow a precise measurement of the η-coordinate of the track. The only information the η-coordinate can be derived from is the place where a particle crossed detector wheel boundaries. The η-value is given in a 2 bit code.

Quality

Due to inefficiencies not all chambers deliver track information at all times. Then a track must be identified and the transverse momentum measured using only the available data. Obviously the measurement resolution is impaired and measurement errors are higher. Thus the track finder has to give information about the quality of the data the track measurement is based to the global trigger. This is done by a two bit number.

Dead time free

The first level trigger architecture must permit a dead time free operation. That means that even in case of a positive level accept signal release the processor must stay operational for the subsequent events. The trigger system must be capable of accepting data of each single event, with a data repetition rate of 40 MHz. It is obvious that the trigger cannot perform its task on a data set of one event within one bunch crossing.

Processing time - latency

It is important to keep the processing time of the track finder processor as low as possible, because during the time an event is evaluated in the level one trigger all corresponding detector data must be stored in the data pipeline. In CMS the number of detector channels is as high as 10^8. Most of them belong to the tracker. The tracker signals are analogue and are only converted into a digital number after a positive level one trigger accept signal. Thus the tracker data is going to be stored in an analogue memory. Since they are expensive the storage depth is limited to 128 steps. As thoroughly discussed earlier the level one trigger latency is dominated by transmission of signals from and to the detector and by signal propagation between processing units. Thus only a smaller part is left over for the execution of the algorithms. Fig. 3.3 shows the latency distribution between the different system components. For the track finder

processor 21 bunch crossings or 525 ns are foreseen. The first level trigger decision must be provided within 128 bunch crossings. Obviously one cannot design a first level trigger system for exact this latency number. Changes at a later time as adaptations to newly found requirements or changes of detector geometry and the resulting changes in cable length cannot be accounted for. Thus a contingency of about eight bunch crossings is foreseen.

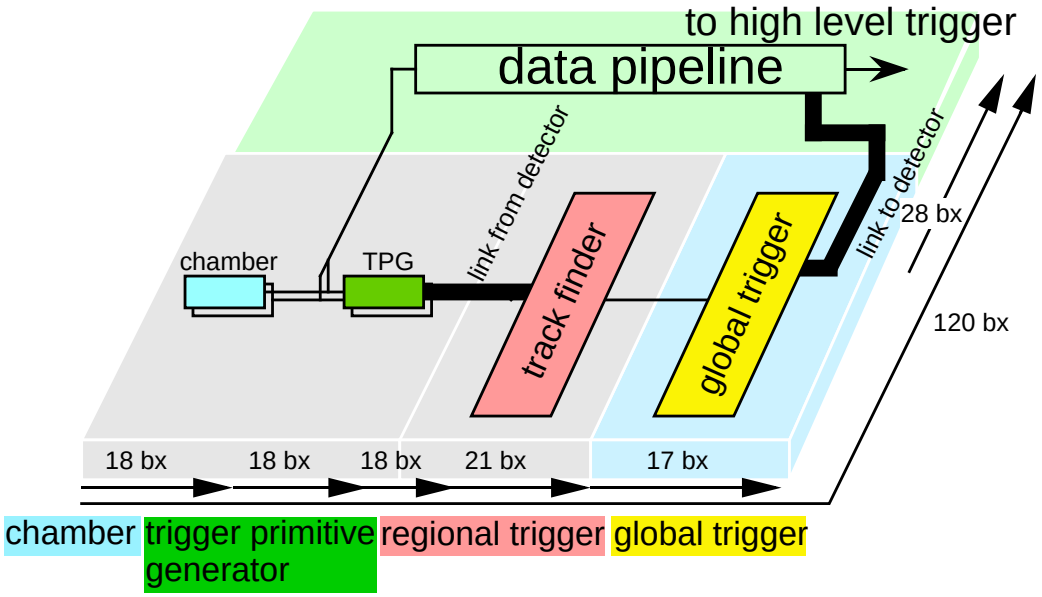


Fig. 3.3.: Latency of drift tube trigger components.

Programmability

The trigger system must be flexible enough to permit changes in the algorithm and changes in the detector geometry. In order to optimize the track finding and measurement algorithm simulations have to be employed. Obviously they are based on several assumptions. Simulation results may be refined in a later phase of optimization or at the time beam data will be available.

Another point is the detector geometry. Even if the designed chamber geometry is not going to be changed misalignment of the chambers must be accounted for. The trigger uses a resolution of 1.25 mm. However, it is impossible to install all chambers (of the dimensions 2.56 m x 2 m) within an accuracy of 1.25 mm with respect to each other. The chambers are installed within a coarser accuracy and a laser based alignment system provides information to the trigger electronics how much each chamber is displaced from the nominal position.

Output segmentation

The trigger system must output the information about four muons with the highest p_t in a detector wheel.

The mapping of the detector geometry onto the hardware level must be realized in a way where information exchange between processing units is minimized. This is especially vital with respect to latency but also to the physical dimensions of interconnections. Although the number of channels to evaluate exceeds 200000 in the

barrel the hardware expense of the track finder system must be kept as small as possible. Table 3-2 concludes all track finder processor specifications.

track finder processor specifications
outputs the four highest p_t muons per wheel: p_t, location, quality
p_t measurement range: 2.0 - 140 GeV/c; resolution: ~ 35 %
ϕ-measurement: 2 mrad
η-measurement: as good as possible
dead time free
programmability: algorithm settings and chamber alignment
modular and flexible for use in the CSC and overlap region
processing time: < 21 bunch crossings or 525 ns.

Table 3-2: Track finder processor specifications.

output	**number of bits**
p_t + charge	5 + 1
ϕ	8
η	2
quality	2

Table 3-3: Output format.

Compatibility to forward and overlap region between cathode strip chambers CSC and drift tube chambers DT

The track finder is supposed to be applied in the barrel region (equipped with drift tube chambers) but also in the forward region (equipped with cathode strip chambers) and in the resulting overlap region between the two systems. This requires the cathode strip primitive generator to deliver the same output format as the drift tube trigger primitive generator. However, as the design of the cathode strip chamber electronics is not evolved yet to a satisfying extent, in this work only the barrel drift tube track finder processor is described. However, from point of view of hardware it can be extended into the forward region of the detector.

3.2. Trigger environment and implementation in other high energy physics experiments

This chapter deals with previously implemented trigger systems in modern high energy physics experiments. Focus is given to muon and track finding systems in similar detector and accelerator environments. The features of the

systems are discussed in the following chapters. At the end of each description the systems are compared to the track finder processor. Chapter 3.2.6. concludes all systems and differences.

A large number of previously implemented trigger systems in high energy physics experiments exist. However only those are described which are similar to the drift tube track finder in a way. Small fixed target experiments are ruled out. Moreover only experiments at accelerators with requirements to the data acquisition similar to those of LHC are regarded. These are colliders with a high bunch crossing frequency. The most recently implemented trigger systems at large scale experiments at modern colliders whose requirements are in some way similar to those of the track finder processor are at the:

- Experiment D0 at the collider Tevatron (FERMILAB),
- Experiment CDF at the collider Tevatron (FERMILAB), and
- Experiment H1 at the collider HERA (DESY).

Tevatron [27] is a high-energy colliding beam accelerator operating at Fermilab, Batavia (Illinois). It collides 900 GeV protons with 900 GeV antiprotons accelerated on a 6.3 km circular path. At the collision points the experiments D0 [28] and CDF [29] are located. The bunch crossing period is 3.5 μs.

HERA [30] is a high-energy colliding beam accelerator operating at DESY, Hamburg. It consists of two intersecting storage rings situated in a tunnel of 6.3 km circumference, one containing electrons of 27 GeV energy and the other protons of 820 GeV energy. The beams of particles collide at two points where experiments are mounted: H1 [30] in the north area and ZEUS [31] in the south area. The bunch crossing period is 96 ns.

3.2.1. Muon trigger electronics at D0 (Tevatron, Fermilab)

The muon trigger electronics [32, 33] is embedded in the level 1 and level 1.5 trigger of the experiment. Due to the relatively long bunch crossing periods it does not employ a pipelined structure. While the level 1 trigger operates dead time less (execution time is less than the bunch crossing time) the level 1.5 trigger will need some tens of microseconds for its analysis thereby exceeding the time between bunch crossings.

The task of the level 1 muon trigger electronics is to find probable track segments in proportional drift tube chambers and then match these track segments. The number of muons in a detector sector is counted and compared to a given threshold.

The magnetic field is parallel to the drift tubes, so that the bending will take place in the drift coordinate. From the inner part of the detector the muon crosses three stations, each consisting of three or four staggered drift tube layers of up to 24 and 72 cells per detector segment respectively. The trigger system has to cope with a total of 15000 drift cells in the entire detector. They are read out by binary signals thus obtaining a spatial resolution of the cell size between 3 and 10 cm.

Logical ORing of neighbouring detector cells is applied to further degrade the measurement resolution. This enables to employ a pattern match using PALs to find

track segments and match the track segments of the three stations together. The output is only the number of found tracks.

In the level 1.5 trigger a higher spatial resolution is used to perform a crude transverse momentum estimation. Hereby the track segment match is refined by employing a content addressable memory (see also chapter 4.1.1.). The p_t measurement is conducted by means of a memory based lookup table.

The calculation time, however, exceeds 3.5 μs and thus introduces dead time to the system. Moreover since the content addressable memory performs the search looping through all possible track segment combinations the execution time is dependent on the number of found track segments. The features of the system are concluded in table 3-4.

D0 muon trigger
bunch crossing period = 3.5 μs
searches locally for track segments and matches regionally to tracks
parallel architecture
15000 trigger cells
pattern match
no pipeline architecture
coarse spatial resolution used (3 - 10 cm or higher)
counts only found muons
level 1.5 introduces dead time: coarse p_t measurement
PALs, content addressable memories and lookup tables are employed

Table 3-4: D0 muon trigger features.

3.2.2. Track finder electronics at CDF (Tevatron, Fermilab)

The trigger system of CDF [34, 35] employs two levels of hardware trigger logic and a level three trigger software farm. Level one incurs no dead time; level two does. The rejection factor provided by level one and two is about 10^3 - 10^4. In the first level trigger only the presence of particles is determined.

The track finder [36] identifies the existence of tracks in the central tracking chamber [37] (but not their positions). Only the level two trigger bases its decision on the presence and position of track segments in the muon chambers that are associated with high p_t tracks in the central tracking chamber.

The track finder for the CDF tracking chamber [38] employs a 19 stage digital pipelined processor to find high momentum muon tracks. It has eight programmable thresholds between 2.5 and 15 GeV/c. A search for all high momentum tracks in the chamber system can be completed in a time of 3 μs.

The central tracking chamber is composed of nine superlayers. The track processor uses only the rϕ-information (bending plane). Each superlayer consists of twelve layers

of drift wires. In total, the trigger system uses 4400 drift wires as binary inputs, resulting in a spatial trigger resolution of about 10 mm.

The hardware implementation employs a pattern matching method. The patterns are defined by a hit in the outer layer, the interaction point and a range of momenta p_t. A total of 32 roads are defined for each wire in the outer superlayer.

The processor architecture performs in a pipelined manner. The pattern comparison is implemented by means of memory based lookup tables. The best track candidate is chosen from among the 32 roads as the track with the most hits and the track with highest p_t. The presence of a track above a programmable threshold releases a muon level 1 trigger.

The system is designed using discrete components, such as AS TTL (multiplexers, priority encoders, buffers, registers), PLA, and CMOS SRAMS. The maximum clock frequency is 25 MHz.

The execution time is dependent on the number of hits in the event. In general, the trigger decision is made within 3 µs. Table 3-5 concludes the system features.

CDF track finder
bunch crossing period = 3.5 µs
applies pattern match using memory based lookup tables
serial, but pipelined architecture
4400 trigger cells
coarse spatial resolution used (10 cm or higher)
measures p_t, but puts out only number of particles above threshold, no spatial coordinate of the track
dead time free
discrete components (ALS TTL, RAM, PLA)

Table 3-5: CDF track finder features.

3.2.3. The fast pipelined vertex finder for the H1 experiment based on multiwire proportional chamber signals

Based on a total of 1920 pads from multiwire proportional chambers [39] the H1 vertex finder [40] reconstructs the position of the collision along the beam axis of the collider. It employs custom designed gate arrays as well as field programmable gate arrays and a memory based lookup table. The H1 detector uses a 4-level trigger scheme. The first level operates dead time free with a decision time of 2.3 µs. Given the bunch crossing frequency of 10.4 MHz at HERA, this requires a pipelined design, performing the trigger algorithm in 24 steps.

The trigger operates on the cathode pad signals of two cylindrical multiwire chambers, situated around the beam axis, as well as one planar chamber (see fig. 3.4). Each chamber consists of two independent layers.

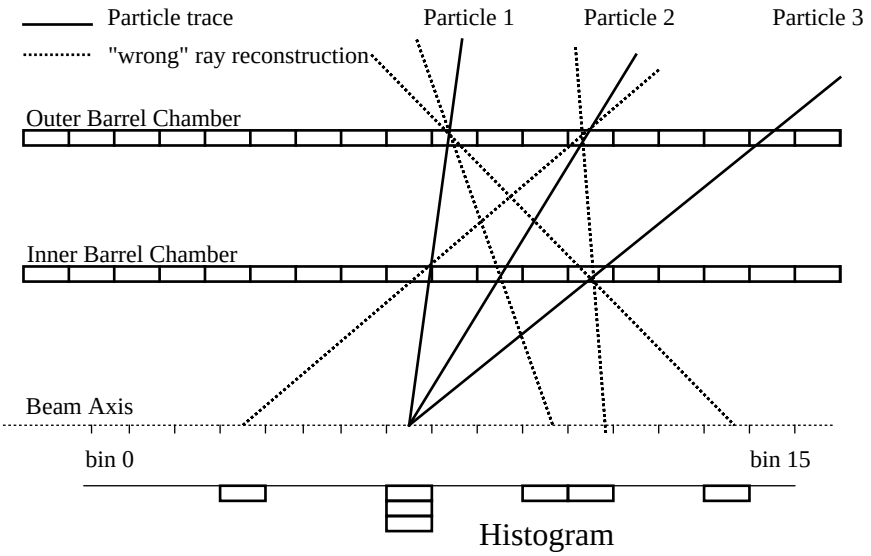


Fig. 3.4.: Vertex reconstruction by building a histogram [40].

A ‘ray’ is defined as the coincidence of four chamber pads, which can be connected by a straight line in the rz-plane. The number of such rays is entered into a 16 bin wide histogram, each bin related to the origins of its respective rays along the beam axis. Inevitably one also finds combinations not originating from a true particle track. However, these wrong rays generally produce a flat distribution in the histogram, whereas at the true vertex location a peak is found. The 16-fold ϕ-segmentation of the detector corresponds to 16 independently built histograms, one for each segment. Once a peak in the histogram is found the location of the peak within the histograms corresponds to the location of the interaction point of the given event.

The rayfinder electronics is distributed over 256 printed circuit boards, each building one histogram bin for one ϕ-segment. For each bin, about 120 combinations of chamber pads have to be examined. Mask programmable gate arrays in 1.5 µm CMOS technology are employed. The rayfinder electronics requires 2112 gate array chips.

Field programmable gate arrays and memory based lookup tables are used to calculate the sum of all inputs, the peak, the number of entries in the peak bin and the of a peak of a histogram. Table 3-6 concludes the H1 vertex finder features.

H1 vertex finder features
bunch crossing period = 96 ns
applies histogram method
parallel and pipelined architecture
1920 trigger cells
measurement resolution of the vertex position: 54 mm no other track properties
measures only the position of the track vertex
dead time free
gate array and FPGA

Table 3-6: H1 vertex finder features.

3.2.4. The drift chamber track finder for the first level trigger of the H1 experiment

The charged particle track finder [41] is based on the hit information of the central drift chamber of the H1 experiment. The track finder is embedded in the experiment's first level trigger and consists of a 24 step pipelined hardware logic with a decision time of 2.3 μs and no dead time.

The signals of the detector are synchronized with a sampling frequency f_s (up to 20.8 MHz) and sent into a storage buffer (shift registers). The shift registers have a depth corresponding to the maximum possible drift time in the chamber plus the decision time of the trigger logic, and each of its bits corresponds to a hit in a drift time slice of width $\tau_s = 1/f_s$. This constitutes a mapping of the space grid with a bin width of $\Delta x = v_{drift} \cdot \tau_s$ into a time grid with time slice τ_s. A similar systematic is employed for the drift tube trigger primitive generator (see chapter 2.3.3.1.).

The track finding logic works fully parallel and uses 870 drift tube wires. The chamber system [42] consists of 10 layers of drift tubes. It samples the pattern of hits and tries to match it with a predefined set of bit masks, which represent track candidates in the transverse momentum range of $|p_t| > 400$ MeV/c.

The track finding logic counts the number of track candidates for positive and negative curvatures with p_t values above two programmable transverse momentum thresholds. In addition, a topological correlation between tracks (such as two tracks being back to back) can be programmed.

The hardware implementation employs field programmable gate arrays (FPGA). The shift registers are placed in 37 XILINX type [43] gate arrays. 42 FPGAs are needed to house about 700 trigger patterns, 16 patterns in each chip. Two FPGAs are used to form the multiplicity for two momentum ranges and two charges. Table 3-7 concludes the H1 track finder features.

H1 track finder
bunch crossing period = 96 ns
applies pattern match
parallel and pipelined architecture
820 trigger cells
trigger resolution of the chamber: ~2.5 mm
counts only tracks above p_t threshold, no position measurement
dead time free
employs FPGA

Table 3-7: H1 track finder features.

3.2.5. The first level trigger for the H1 forward-muon spectrometer

The first level forward-muon spectrometer trigger [30] is based on drift tube chambers with a maximum drift time of 1.2 μs. It looks for particles whose tracks point back to the interaction region. The system works in a pipelined manner and introduces no dead time to the H1 level one trigger. The latency of the system is 22 bunch crossings or 2100 ns.

The chamber system consists of six drift chamber planes, three on either side of an iron toroidal magnet. Each plane has to drift tube layers, with one layer displaced by half a cell. Four planes are used for the trigger. The total number of drift wire inputs is 112.

The trigger system comprises three parts:

- The trigger primitive generation: In each plane the two layers are correlated and a position measurement of found and pointing tracks is given out. Moreover the bunch crossing of the traversing particle is identified.
- Correlation of two track segments of the chambers after the magnet and before the magnet (see fig. 3.5). This is done separately for the tracks in planes before and after the magnet.
- Correlating the track segments from before and after the magnet to a full track.

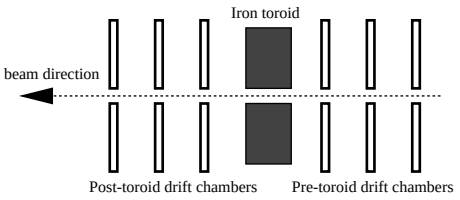


Fig. 3.5.: H1 spectrometer [30].

For the trigger primitive generation chamber hits of the two layers in each plane are fed into shift registers with a clock period of 48 ns. The scheme is similar to the drift tube trigger primitive generator discussed in chapter 2.3.3.1. Using a coincidence matrix it selects tracks originating from the interaction point and identifies the bunch crossing the hits belong to. The output is available after a fixed time after the corresponding bunch crossing and indicates the spatial coordinate of the pointing track. The spatial resolution achieved is 2.4 mm. However, the resolution is not used for the correlation process. Because of the use of OR gates the resolution is degraded to a value between 20 and 60 mm.

Correlation of track segments from planes before and after the magnet is performed again using a pattern match method by means of a coincidence matrix. The programmable matrix selects only combinations which possibly come from the interaction point and puts out their spatial coordinate.

Assembling the correlated track segments pairs from before and after the magnet to full tracks is again done with a coincidence matrix.

The output of the system to the final decision module and the higher level triggers is a crude measurement of the location of the trigger tracks. The hardware implementation of the 32 x 32 coincidence matrices for the trigger primitive generation and the correlators employs semi-custom ASIC in 1.5 μm CMOS technology. The execution time of the trigger algorithm is less than 2.3 μs. Taking into account the drift time in the chambers, and the cables from the chamber to the trigger and from the trigger to the central trigger logic of the experiment, the time available for the trigger

to make its decision is much shorter. It is as low as 300 ns. Table 3-8 concludes the H1 forward-muon spectrometer trigger features.

H1 forward-muon spectrometer trigger
bunch crossing period = 96 ns
applies pattern match
parallel and pipelined architecture
122 trigger cells
trigger resolution of the chamber: >2.4 mm, but not used for track finding (20-60 mm)
gives out only location of the tracks no transverse momentum measurement
dead time free
no magnetic field within chamber system: straight tracks
employs semi custom ASIC

Table 3-8: H1 forward-muon spectrometer trigger.

3.2.6. Conclusion: Other high energy experiment triggers

While the previously discussed trigger systems all have impressive features none of them combine all requirements in one system as they are necessary for the drift tube track finder of CMS. One of the most striking differences is the operating frequency requirement. In LHC the bunch crossing period is 25 ns. Due to the large physical dimensions and the high required operating frequency signal delays on the cables will contribute to the challenges of the trigger system. The level one trigger system is a synchronous setup running on a single clock. The large physical dimensions of the detector and its read out environment results in a highly distributed level one trigger system. The detector is about 10 m long and 14 m in diameter. The control room where all data are transferred to including the level one trigger data is about 90 m apart. Data propagation along the transfer path from the detector to the control room takes as much as 400 ns or 16 bx time. A dedicated clock distribution system [44] has been developed.

Previously implemented triggers with the highest operating frequency are the H1-triggers at HERA (DESY). They operate at a clock period of 96 ns. However, the system complexity in terms of number of channels to be processed does not exceed 2000 channels. In case of the H1 vertex finder the number of channels is 1920. In the track finder processor system of the drift tube chambers $2 \cdot 10^5$ channels must be processed.

The large number of channels to be evaluated in the CMS first level trigger in combination with the high bunch crossing frequency yields an immense data rate to be coped with. The drift tube trigger primitive generator sends its data with a rate of about $4 \cdot 10^{11}$ bit $\cdot$ s^{-1}. One should not forget that the drift tube trigger primitive generator already performs a zero suppression on the raw chamber data. D0 is the experiment which was found to have the highest number of input channels into the trigger. In D0

the level one muon trigger performs its operations on 15000 channels. The bunch crossing period at D0 at the Tevatron is 3.5 μs, being slower by a factor of $1.4 \cdot 10^2$ compared to LHC. The data rate for the D0 level one muon trigger thus yields only $4 \cdot 10^9$ bit $\cdot$ s^{-1}. This is less by two orders of magnitude.

Due to the strong magnetic field (4 T inside the coil; 1.8 T in the yoke) and the large dimensions of the detector the resulting high bending power for charged particles, causes a strong curvature of the particle tracks. This is, in combination with the high chamber resolution, the large number of channels in the drift tube chamber system and the large transverse momentum measurement range (2.5 GeV/c to 140 GeV/c), the reason for a high number of chamber channels to be observed at the same time to find all parts of a full track. Due to the strong magnetic bending the tracks are likely to cross detector segment boundaries. Information exchange between processing units is unavoidable. In no other trigger system the number of channels was as high as 200000 with a trigger resolution of the chambers of as good as 1.25 mm. In the D0 muon trigger the number of channels is 15000 but there the used spatial resolution of the chambers is between 30 and 100 mm and no p_t measurement is conducted in the first level. The H1 track finder achieves a 2.5 mm spatial resolution but bases its measurements on only 820 channels and simply the number of found tracks above a certain p_t threshold are given out.

None of the already implemented systems puts out the transverse momentum p_t and the position of the tracks in both coordinates η and ϕ as the track finder does. While some systems give out the transverse momentum p_t (like the D0 muon trigger) they do not measure the location of the track. However the D0 muon trigger performs the momentum measurement in the level 1.5 trigger which introduces dead time and is asynchronous to the bunch crossing frequency. Most systems only count the number of identified muon tracks. Some of them put out the number of tracks above a given p_t threshold. An example is the H1 track finder which operates on 820 wires only.

The processing speed of previously implemented trigger systems is also higher than the track finder processor execution time requirement. A large variety of trigger systems do not work dead time free. Even those working dead time free have a long calculation time. This is possible due to the longer bunch crossing frequency of the accelerators (muon trigger at D0 executes its algorithm within the bunch crossing period). When the trigger execution time exceeds the bunch crossing period and it has to work dead time free, a pipelined architecture is applied in most of the cases. This requires the data of the detector to be stored during the decision time of the trigger. If only a small number of channels are to be read out the storing poses no problem. However, in CMS, the total number of channels is in the order of 10^8 and the majority of these channels are analogue signals. As discussed earlier they are going to be stored in analogue memories. Due to the cost of analogue memories the memory depth is limited. As a consequence the processing time of the track finder processor must not exceed a given time. (see also chapter 2.3.2. and fig. 3.3).

Each single requirement of the drift tube track finder, as operating frequency, chamber complexity (number of channels), measuring p_t in strong magnetic field (strong curvature of tracks, thus large overlapping), and output format, is more challenging than those of already implemented high energy physics trigger systems. Of course there are commercial systems available which are equal or even better performing than the track finder processor with respect to a single requirement. An

example would be the large number of channels in modern image processing application, but there the processing time is found to be in the µs or ms range. Faster operating frequencies may be found in modern microprocessor systems. Then, however, either the system complexity is found to be at modest extent or the execution time of algorithms is long since a high number of processing steps are necessary.

Generally seen all the tasks of the track finder processor already have been performed previously but never within the first level of a high energy physics trigger. The highly sophisticated tasks of the processor as measuring p_t and location of the tracks were left to higher level triggers or off-line triggers and thus in a detector environment hardly comparable to those of CMS with respect to speed, precision and number of detector channels. The track finder is no trigger system in the classical way. It does not apply a cut on data. Furthermore it measures the particle's properties in a highly sophisticated way and thus it is to regard as a high performance instrument delivering precision measurements until now only available after off-line analysis. In table 3-9 the features of the trigger systems discussed in the previous chapters are compared to those of the track finder processor's.

Trigger	bx-period (ns)	method	output	number of channels resolution	magnetic field	technology
D0 Muon Trigger L1	3500	pattern match	number of found muons, no position	15000, 30-100 mm	yes	PAL
D0 Muon Trigger L1.5	3500	pattern match	crude p_t calculation, no position asynchronous introduces dead time	15000, 30-100 mm	yes	PAL, RAM, content addressable memory
CDF Track Finder	3500	pattern match	number of particles above threshold, no position measurement	4400, ≥ 100 mm resol.	yes	discrete components (ALS TTL, RAM, PLA)
H1 Vertex Finder	96	histogram method	position of track vertex	1920	no	gate array and FPGA
H1 Track Finder	96	pattern match	number of tracks above p_t threshold, no position measurement	870, 2.5 mm	yes	FPGA
H1 Forward Spectrometer trigger	96	pattern match	location of the tracks, no transverse momentum measurement	122, 20 - 60 mm	no, two straight track segment	semi custom ASIC
CMS drift tube track finder	**25**		**transverse momentum p_t, location in ϕ and η**	**200000, 1.25mm**	**yes**	

Table 3-9: Features of previously implemented large scale first level triggers compared to the track finder specifications.

4 Evaluating the CMS track finder processor

In Chapter 4.1. several realisation methods are discussed. Their possible application is evaluated and the most suitable algorithm is selected. Although this chapter deals with finding the best realisation method one of the selection criteria is the implementation feasibility in hardware.

Chapter 4.2. examines the implementation technologies for the chosen algorithm.

In chapter 4.3. physics simulations are presented which prove the functionality of the chosen algorithm.

4.1. Feasibility study: Feature extraction methods

This chapter discusses several feature extraction (or measurement) methods with respect to the possible application in the track finder processor. Their advantages and disadvantages are discussed in detail. The method best meeting the given requirements is selected for further investigations. Most methods are known to having been used in software applications and off-line triggers. Special focus was given to find an implementation method which may be employed using today's technology. This is important to ensure that the trigger system and its functionality or its implementation does not rely on the progress of any technology. This also makes it possible to build a prototype and test it before the actual start of the experiment.

In the following sections four implementation methods and their possibility of application are discussed:

- Pattern comparison method
- Histograms
- Artificial Neural Networks
- Track segment method (Extrapolation)

For each of the described methods the advantages and disadvantages are highlighted. The task is to find a method which meets the requirements of the measurement resolution and which is implementable using technologies available today. The

specifications on resolution, timing and hardware extent together form the selection criteria to chose an algorithm. When evaluating a measurement method first the effects of the resolution requirements to the hardware architecture are examined. The amount of hardware and the execution time with today's available technologies are estimated. Based on these estimates the decision whether further investigations are to be made or discontinued is taken. In chapter 4.1.5. all methods discussed are compared directly. The best method is chosen for further investigations.

4.1.1. Pattern comparison

The classical method in high energy physics hardware implementations is to search for predefined tracks or bit patterns. Predefined patterns are compared to the actual ones occurring (pattern comparison).

Each crossing charged particle generates a hit pattern. Due to the finite resolution of each detector many different tracks may cause the same hit pattern. The number of valid hit patterns of all different tracks is limited. One can store them in a sufficiently large memory. The number of valid patterns depends on the detector resolution and its geometry. The actually occurring patterns are obtained by simulation or beam data. Properties, like transverse momentum p_t, are attached to each stored pattern. In case the detector data resembles a stored pattern one can deduct the corresponding track properties.

Pattern Comparison - Direct method

Usually pattern comparison methods are implemented in hardware without the use of separate trigger primitive generators. That means the detector signal is digitized and forwarded directly to the pattern comparison unit. No zero suppression is performed. This is referred to as *direct pattern comparison* method. It is applicable especially when using detector cells with binary output, indicating the hit of the corresponding sensitive element. The output of an entire chamber is a bit stream of a length proportional to the length of the chamber. The position of the bits within the bit stream corresponds to the position of the detector cell within the chamber. However, systems with drift chambers which output a drift time or a hit coordinate respectively can employ direct pattern comparison methods too. The output of the drift tube trigger primitive generator can be coded into a bit stream using an '1 out of N' decoder, indicating the location of a hit.

Using a direct pattern comparison method for the track finder processor

In chapter 3.1. the required spatial resolution is shown to be 0.3 mrad or 1.25 mm in station one and two and 2.5 mm in station three and four.

Considering the different length of the chambers they must be divided into about 1600 bins of 1.25 mm and 2.5 mm respectively. For pattern comparison methods the critical parameter is first the number of bins N_{binmin} one has to look at in all stations in order to find all possible occurring tracks originating from a given reference cell and secondly the number of patterns P_{patt} to be stored (see fig. 4.1). The number of bins N_{binmin} equals the minimum bit number to be sent to the integrated circuit. Reasonable numbers for I/O pins of an integrated circuit for this application are in the order of 300 pins.

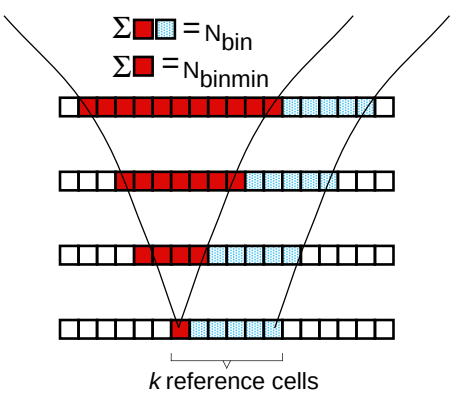


Fig. 4.1.: Calculation of number of bins N_{bin}.

In order to estimate N_{binmin} one selects a reference bin, for instance a bin in station one, and investigates all possible tracks originating from this single cell in this station. The maximum spread of muons originating from the reference cell yield the detector area to evaluate. Given this area and the detector granularity allows to calculate the minimum number of detector cells N_{binmin}. Obviously the tracks with the widest spatial spread are tracks with low transverse momentum p_t. Fig. 4.2 shows the deflection of muon tracks between station one and two ϕ_{21} over their transverse momentum p_t for η=0. After station two the bending angle decreases again due to the opposite direction of the magnetic field in the return yoke. In average muons with p_t less than 4 GeV/c will not reach stations three and four [22]. Low p_t muons hit only two stations each consisting of two rϕ–superlayers with four drift tube layers each. Thus track patterns of such muons consist of only 16 bits, while high p_t muons involve 32 bits. Using equation 4.1 allows an estimation of the required number of bits N_{bin} for k reference cells in station one (fig. 4.1). n_{SL} is the number of superlayers a muon with $p_t > 4$ GeV/c can hit. n_{SLd} stands for the number of superlayers muons with $p_t < 4$ GeV/c can reach. ϕ_{n1max} is the maximum spread (medium value plus one σ, see fig. 4.2) of muon tracks between station one and n ($\phi_{21max} = 0.12$, $\phi_{31max} = 0.1$, $\phi_{41max} = 0.1$). $\Delta\Phi$ is the spatial resolution of the chamber. n_{DT} is the number of drift layers per superlayer.

$$N_{bin} = n_{SL} \cdot n_{DT} \cdot \left(\frac{\phi_{41max} + \phi_{31max}}{\Delta\phi} + k \right) + n_{SLd} \cdot n_{DT} \cdot \frac{\phi_{21max}}{\Delta\phi} \qquad \text{(Eqn. 4.1.)}$$

N_{bin} equals 28000 for k=1 (= N_{binmin}). That means 28000 bits must be observed at the same time in order to find all possible muon tracks originating from a single reference cell. The total number of reference cells in a chamber is about 1600. In order to keep the number of integrated circuits per detector segment below 100 a chip must work with 160 reference cells (k=160). The number of required I/O bits per chip increases to 33000. It is obvious that this is far from being a viable solution.

Pattern matching - indirect method (trigger primitive generator)

When using high resolution position detectors direct pattern matching methods require a high hardware expense. Trigger primitive generators are applied. They perform zero suppression and output the track data in a compressed format. The amount of data to be processed by subsequent stages using pattern comparison methods is reduced dramatically. In case of the drift tube trigger primitive generator (DT-TPG) the data reduction factor is about 300. When track finding is performed by pattern matching after employing a trigger primitive generator it is called *indirect pattern matching*.

In each station the drift tube trigger primitive generator (DT-TPG) [16,17] delivers up to two track segments consisting of a spatial and angular coordinate. While the amount of data delivered is reduced the precision is fully maintained. Data of eight drift

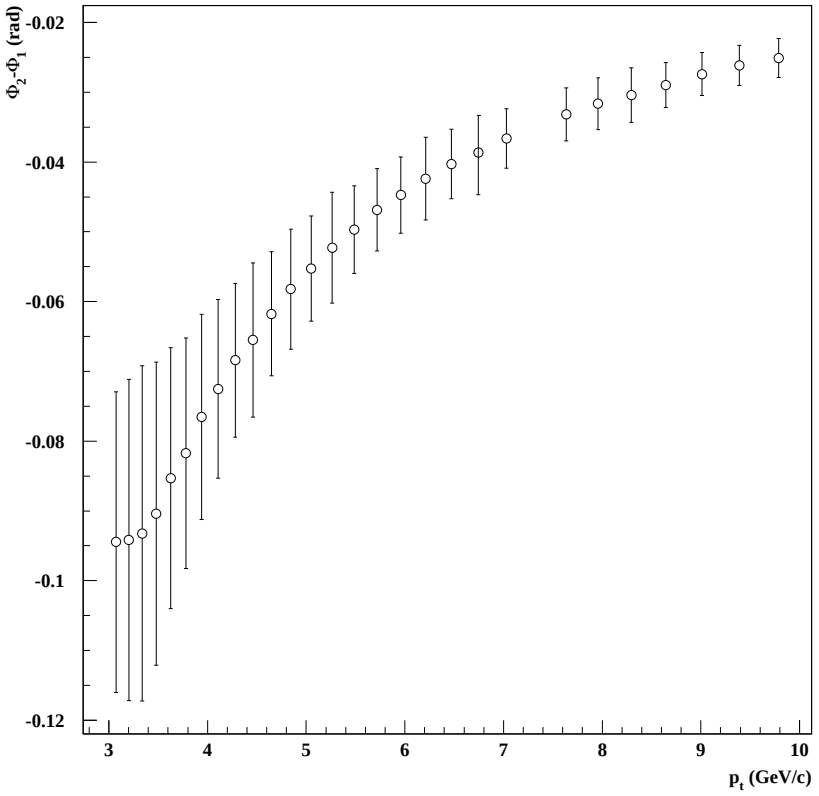


Fig. 4.2.: Deflection of muons between stations one and two ϕ_{21} over their transverse momentum p_t for pseudo rapidity η=0. The error bars give the $\pm\sigma$ deviation from the medium value.

tube layers of the two rϕ-superlayers in a station are compressed by one drift tube trigger primitive generator. Unlike in the direct pattern match the track patterns do not consist of bit streams with each bit indicating a hit. Hit coordinates represent valid track patterns. The drift tube trigger primitive generator delivers four 19 bit words, totalling to 76 bits. In each station two track segments are given out. Two sets of 76 bits (152 bits) must be evaluated per detector segment. Due to the long pattern words a pure combinatorial solution employing simple gates can be ruled out. Memory based lookup tables are very popular for high speed trigger applications. However, lookup tables with an address width of 152 bits and a memory depth of 2^{152} are not feasible. On the other hand this huge storage capability is not necessary because only a small fraction of 2^{152} combinations are valid muon tracks. A possible solution are content addressable memories (CAM).

Pattern matching using content addressable memories (CAM)

A content addressable memory or associative memory [45] is a memory that does not address its stored data by its location but by its content. That means a pattern is the input and the memory compares this pattern to its stored content. The output can either be only the information if a match is detected or the location (address) of the pattern. However, it has to be stated that content addressable memories are available commercially only with small storage capacity. Examples can be found in [46,47]. The storage depth is about 1024 x 48 bit words.

Applications for pattern matching with a content addressable memory in a high energy physics trigger can be found in [48,49]. The memories are used to store possible track patterns. It is not sufficient that the content addressable memory only confirms the existence of a pattern in its memory. The address of the pattern must be found. The address must allow to deduct track properties like transverse momentum and location of the track unambiguously.

In all first level triggers calculation speed is a key issue. Thus all comparisons between the event patterns and all prestored patterns must be done in parallel. A VLSI implementation of an associative memory architecture is proposed in [50,51]. Each memory cell is large enough to hold one pattern and has built logic in to compare its content to the event pattern. Fig. 4.3 shows the block diagram of the memory. Each row

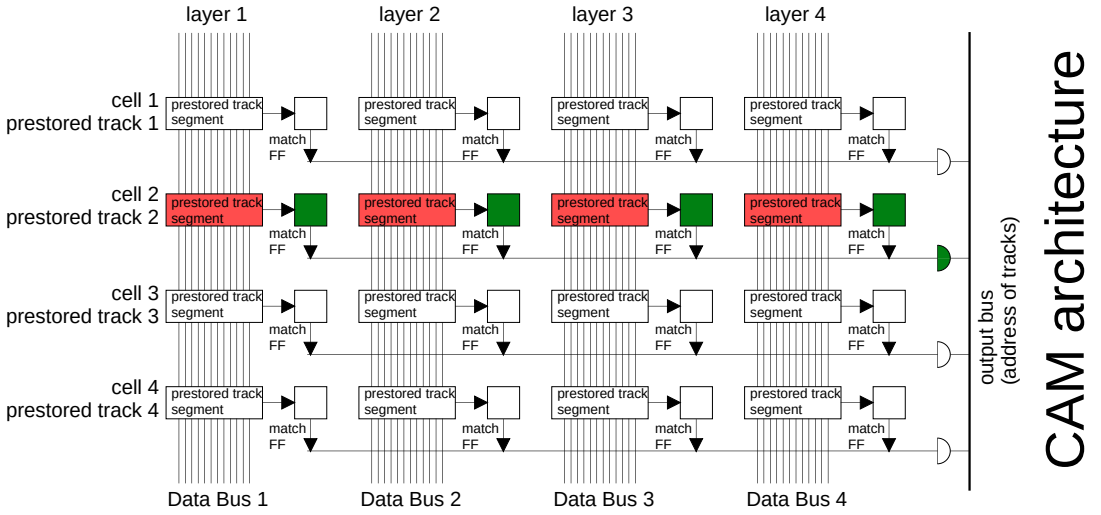


Fig. 4.3.: Content addressable memory architecture.

represents one cell. The cells are divided into four words with one word per detector station (track segment). All the words in a cell define an entire pattern by specifying one track segment per layer. The data buses connect all word memories of a layer with each other. When looking for a pattern all track segments of a station are loaded sequentially onto their corresponding data bus. All the word memories compare its contents to the incoming track segment data in parallel; in case of equality a hit flip-flop is set. After all track segments have been loaded, any cell with all flip-flops set contains a track pattern of a track of the last event, because all track segments of this hit pattern are present in the event. The corresponding '*found*' signal is activated. The addresses of the track patterns are sent to the output bus sequentially. The processing time of this scheme is proportional to the number of track segments per station to be loaded into the content addressable memory and the number of found track candidates given out by the memory.

In [52] the AMchip, a content addressable memory ASIC, applied in the fast trigger system in CDF [48] is described. It employs 121000 MOS transistors on a 100 mm^2 die in 1.5 micron technology. It carries 128 patterns, each five 12 bit words long.

H.W. den Bok et al. [49] describe the application of an associative memory employing field programmable gate arrays (FPGA). Each memory is able to recognize 24 patterns. The 24 '*found*' signals are encoded into a 5-bit binary number by a priority encoder. In response to an active '*found*' signal, the priority encoder returns the address

of the recognized track. If more than one '*found*' signal is active the track address produced is the one with the highest priority.

A disadvantage of the content addressable memory scheme is that it does not allow a pipelined design. Thus several memories must be used in parallel in order to cope with the data rate and to provide dead time free function ('round robin' scheme). In order to know if a content addressable memory can be applied for the track finder processor it is important to investigate the total number of track patterns to be stored.

Using the content addressable memories for the track finder

The particle tracks differ from the ideal mathematical shape due to collisions with the matter the particles traverse (multiple scattering and energy loss, see chapter 4.3.1.). Thus it is not possible to precisely predict the number of possible track patterns. However, simulations have been performed. The simulations have been conducted in such a way that muons were aimed at a single reference cell of the size Δx in station one (fig. 4.4). The track finder processor must be able to assign each muon to one of the p_t-classes, shown in table 3-1, unambiguously. For each p_t measurement class the different possible track patterns were counted.

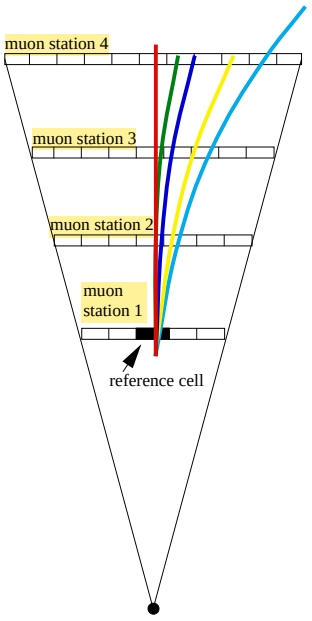


Fig. 4.4.: Principle of pattern number estimation.

Fig. 4.5 shows the number of patterns to be stored for p_t classes 10-12, 25-30, 40-50 and 50-60 GeV/c. Parameters are the spatial resolution Δx and angular resolution $\Delta\phi_b$ of the track segment measurements. Pattern numbers for Δx = 2.5 mm, Δx = 5 mm and angular resolution of $\Delta\phi_b$ = 10 mrad are shown. However, although the angular value is necessary to identify the p_t class a muon belongs to (in case one or two station hits are missing, see chapter 4.3.4.) the number of patterns was estimated not considering the angular measurement at all; $\Delta\phi_b = \infty$. Fig. 4.5 shows that for the p_t class 10-12 GeV/c, spatial resolution Δx =5 mm and no angle measurements the pattern number is as high as 400. Station one is 2 m wide and a binning of 5 mm yields 400 reference cells. Hence the number of patterns as high as for this p_t class is 160000. Naturally the number of patterns for the nine p_t classes below will be at least as high. Thus a crude estimation of the pattern numbers for p_t classes < 12 GeV/c and Δx =5 mm yields a number of $1.6 \cdot 10^6$. Note that the effects of the angular measurements are not considered and the actual number is considerably higher. Estimations for spatial resolution of Δx =2.5 mm were conducted for higher p_t. However, as stated in chapter 3.1., a granularity of Δx = 1.25 mm is needed. For p_t classes between 25-30 GeV/c and 50-60 GeV/c one finds 150 patterns. These values applied to the p_t classes from 12 GeV/c to 60 GeV/c - 9 p_t classes - yield a pattern multiplicity of (2000 mm / 2.5 mm) · 150 · 9 = 10^6. In all the number of patterns to be stored is higher than $2 \cdot 10^6$. However, a detector granularity of 1.25 mm for pt > 12 GeV/c is necessary. The actual number of patterns is then even higher for these p_t classes.

Taken a modern commercial content addressable memory with a storage capacity of about 1000 patterns one would need at least 2000 memory units. Moreover the

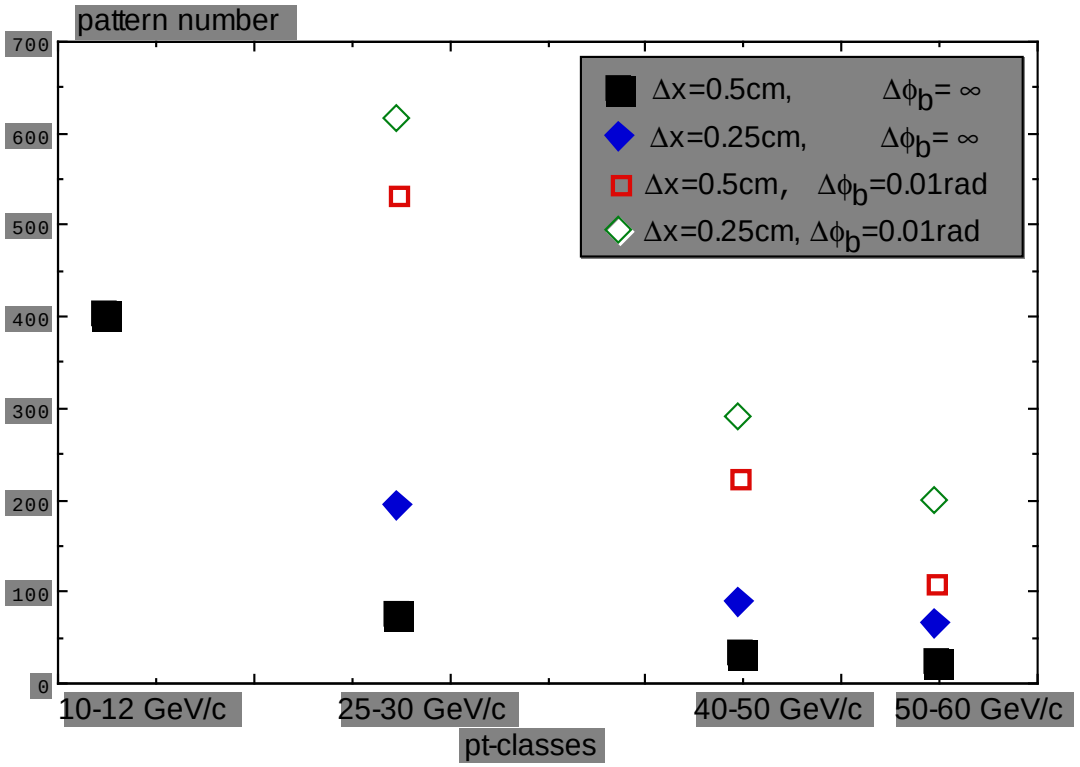


Fig. 4.5.: Pattern number estimation for four p_t measurement classes and various spatial resolutions.

calculation speed advantage of the content addressable memory solution is lost since the highest rank track must be found among those given out by all 2000 units.

Possibility to reduce the number of patterns

The rϕ-projection of the muon system in CMS resembles almost a circle. The deviation from the circular shape is small (3.4%). Thus one can assume that identical muon tracks cause identical patterns regardless whether they are located in the outer area of the chambers or in the centre. That means the relative difference of spatial coordinates within one pattern is equal for all patterns caused by the same muon track independent of its location within the chamber. The absolute values of the spatial coordinates of patterns of two identically shaped muon tracks hitting different reference cells are translated by their distance. Instead of defining a different set of valid patterns for each reference cell one can just observe the relative locations of the track segments to each other. Hereby one calculates the relative offset between the hits in the reference station and the other stations.

All position values of track segments of station two, three and four are subtracted from the spatial coordinate of a track segment in station one. Referring to station one's track segment one obtains three words indicating the particle's deviations of the track segments in stations two, three, and four. Thus patterns consist of only three (station-) words. For the second track segment in station one the same mechanism is to applied.

The number of patterns to be stored is reduced by a factor of at least 400 (number of reference cells in station one for Δx = 5 mm). However, one has to cope with inefficiencies in the chambers. In case the hit in the reference chamber is missing the memory will not find the track. As shown later in chapter 4.3.3. a valid track must consists of at least two track segments. Hence eleven valid track classes exist: T1234,

T123, T124, T134, T234, T12, T13, T14, T23, T24 and T34 - the digits denote the stations involved in the track. Thus additional memories with reference to stations two and three must be foreseen. Each of the two track segments in the reference stations must be equipped with a memory. Hence 22 memories must be employed. Even if all patterns of a track class would fit into one content addressable memory each, the total number of memories would be (number of track segments · number of track classes · number of calculation cycles = 2 · 11· 10 =) about 220 per detector segment. (The number of calculation cycles is assumed to be about 10. This includes the time to input all track segment data serially.) Among the output of all the memories the two highest ranking tracks must be selected. This is time consuming because p_t values have to be compared to each other. Moreover since a complete track involving all stations is found by all memories one and the same track is given out more than once. However, these tracks cannot be easily distinguished from tracks close to each other with transverse momenta of the same class. Multiple tracks cannot be filtered out at all.

Conclusions: Pattern comparison methods for the track finder processor

The low granularity of the detector and the high required p_t resolution confined with the high number of detector channels, the non-projective chamber geometry and high bending power of the detector does not allow to perform track finding and p_t assignment within the available calculation time using a pattern matching method. On one hand the amount of patterns is far too high; on the other hand the resolution and complexity of the detector requires too much data to be processed at the same time.

4.1.2. Histogram method for track finding

Histogramming has a broad spectrum of application in off-line analysis of high energy physics data. One hardware application of this method is found in the z-vertex trigger for the H1 experiment at HERA, see also chapter 3.2.3. [40].

The specifications of the z-vertex trigger are not comparable to the track finder processor specifications neither with respect to the complexity of the detector system nor the processing speed.

Using the histogram method for the track finder

When employing a histogram method one has to find adequate histogram functions. In case of the track finder the best function values would be transverse momentum $p_t = f(\phi,\phi_b)$ and location $\phi_{track} = f(\phi,\phi_b)$ as a function of spatial and angular track segment coordinates ϕ,ϕ_b. In all stations but station three transverse momenta can be derived from the bending angle ϕ_b. In close vicinity to station three the bending angle ϕ_b has a zero crossing. Thus transverse momenta cannot be deduced from track segments in station three (see also chapter 4.3.2.). Moreover the resolution of the angle measurement ϕ_b is not sufficient for the measurement requirements on the p_t measurement (see chapter 4.3.4.). Hence a method were the desired quantities, p_t and ϕ_{track}, are assigned by a function and detectable directly from the histogram can be ruled out.

An alternative possibility is to find functions of the hit coordinates (ϕ and ϕ_b) giving a calculated hit coordinate in a reference plane. Since such a function for station three cannot be found station three has to be the reference plane. Function values are spatial

and angular coordinate of the tracks in station three. They are entered in the two dimensional ϕ-ϕ_b-histogram. A peak will form in the histogram when track segments (ϕ,ϕ_b) enter the same histogram bin and thus come from the same track. The location of the peak corresponds to the location ϕ_{track} of the track but not to the transverse momentum *pt*. That means the histogram method can be used only to find tracks but not to assign a transverse momentum. In addition to the number of entries for each bin one has to store the relative address of the track segments which caused the entry. Once the peak in the histogram is found one can select the track segments of the perceived track(s) using their addresses stored with each bin. They are used to find the hit coordinates to calculate the transverse momentum p_t. However, the compactness of the histogram method is lost.

For the task of assembling track segments to a complete track the full ϕ-resolution (0.3 mrad or 11 bits) and ϕ_b-resolution (10 mrad or 8 bits) is not needed. Assuming that eight bits for ϕ and five bits for ϕ_b are sufficient, the histogram still has a dimension of size 256 (ϕ) times 32 (ϕ_b). Each histogram bin has to store the number of entries and the addresses of at least four track segments which caused the entries. This means the peak finder has to find the highest entry in a (256 times 32 =) 8192 bin histogram. Given the timing constraints this is also not practicable.

Conclusion: Histogram method for the track finder processor

While proceeding in such a manner is a common approach in software solutions a hardware implementation does not seem practicable. The size of the histogram requires a huge amount of logic units. Moreover the calculation time would by far exceed the required maximum latency.

The strength of the histogram method, namely producing a histogram with bins of the desired features, can not be exploited to the full extent. No function can be found for all input data which produces the transverse momentum p_t and location η, ϕ. Therefore the architecture looses its compactness.

4.1.3. Artificial neural networks

The compact representation of detector hits is given by an analogue hit position value (derived from a drift time of a drift chamber as in the case of the drift tube trigger primitive generator). This opens the possibility to employ neural nets. A comprehensive discussion about theory and application of neural nets can be found in [5].

Recently intensive research was conducted on the application of neural nets in high energy physics [6]. This includes software and off-line triggers as well as hardware triggers. The use of neural nets in LHC experiments was studied [53].

In [54] software algorithm using neural nets to find tracks in four layers of drift tubes is described. The input are the drift times of 14 drift tubes. The output is the slope of the traversing track. The technique was found to work for both simulated and real data from a collider experiment. It resolved the left-right ambiguity of drift tube systems and was found to cope with noise sufficiently.

The neural trigger in the experiment WA92 at CERN [55] is the first actually operating hardware trigger application of neural nets in high energy physics. It has the task of evaluating events already accepted by the WA92 standard trigger in order to identify a certain particle decay and to accept them into a special data stream for early analysis. The neural chip ETANN [56,57] of Intel is employed. In time multiplexed mode the number of inputs can be as high as 128. Using no hidden layers the response time is 3 μs, and roughly 8 μs with one hidden layer.

An application study of INTEL's ETANN chip was performed for drift chamber track finding for the D0 collider system at the Tevatron [58]. The system is described in [57]. Track finding was studied with neural nets in a prototype four-layer muon chamber which had eight cells. The task is to determine the intercept and slope of muon tracks. The drift times are measured by time to voltage converters. The voltages are proportional to the drift times (or distances). The analogue drift times are fed into the neural network chip. The intercept and the slope are expressed as peaks in the distribution of outputs. 32 outputs are used for the intercept and the second set of 32 for the slope. Using the distributed output method instead of having two outputs where each of them is proportional to the intercept and slope respectively, increases the measurement resolution to 6.25 mm for the intercept range between 0.0 mm and 200 mm with a chamber resolution of 0.5 mm. The slope outputs represent 50 mrad bins between -0.8 rad and +0.8 rad. The muon trigger is in the second level of the D0 trigger system [59] and must perform calculations within several microseconds. If the neural networks were incorporated into an upgraded muon trigger system they would provide an improvement in the track resolution.

In several more high energy physics experiments neural networks are considered for application and some are already in use (CDF [60], CP-LEAR [61], H1 [62,63], NEMO, WA92 [55]). The research results are quite promising. However, it has to be said that until now no first level trigger was employed using neural nets only. This is firstly due to the relatively long processing time and to the limited complexity of implementable algorithms.

Neural networks implemented in software are becoming more popular recently. However, only a few hardware realizations exist today and even less are commercially available. In addition those available are very slow with respect to the requirements of LHC detectors. The response time of typical commercially available digital neural networks is found to be between 1 to 10 μs. However, the industrial chips were not designed to meet the requirements of fast propagation time. Within the high energy physics community investigations and developments are in progress to increase processing the speed of neural nets.

A recent application of a digitally programmable analogue neural network [64] is reported in [62]. This chip has 70 analog inputs and one output. The processing time in the described application is as low as 50 ns. A fast real-time tracking system provides a 16 bin histogram of track vertices along the beamline. These 16 bins are fed into the chip, which then gives yes/no answer to whether there is a good peak in the proper range. It has to be said that in this experiment a conventional method using digital logic (comparator arrays) was originally employed for this task. The advantage of the neural network solution is the fast processing time.

Using neural networks for the track finder

Each track segment can be seen as two independent coordinate values; ϕ with eleven bit resolution and ϕ_b with eight bit. As shown later at least 48 track segments must be evaluated per detector segment to see all tracks originating from a given detector segment. Hence a neural network would have to have 96 inputs. Both variables ϕ and ϕ_b with their resolutions can be accommodated within a digital neural network. The outputs of transverse momentum p_t, location ϕ, η, and quality q can be assigned to an output of a neural net each.

Conclusion: Neural networks for the track finder processor

Concluding the state of the art of hardware implementation of neural nets it must be said that today it is not sufficiently advanced. Still the most ambitiously implemented algorithms are at a level of moderate complexity. One must not forget that the requirements on the track finder system are severe with respect to processing speed, failure control, reprogrammability and complexity of measurement algorithm. Due to the non-projective geometry of the drift tube chambers the number of input channels to the track finder is large. However, as designs of neural net implementations evolve, especially with respect to processing speed and number of inputs, they can become a possible solution for first level triggering and thus for the track finder. Since one of the implementation goals of the track finder is to use today's available technology the neural net solution was not decided to be investigated further. However, this promising method of image processing and the progress of hardware implementation should carefully be observed in the future. A field of further studies would be a neural network algorithm for the track finder in case an implementable hardware solution should arise within the next years.

4.1.4. Track following method using extrapolation

The basic principle described in [4] and in chapter 1.2.2.1. is to attempt to match track segments caused by the same track together. This is done by extrapolating into the next station from a track segment using the spatial and angular measurement. The extrapolated hit coordinate is compared to the actually measured track segments. If the difference is found to be within a given threshold the extrapolation is considered successful and two track segments have been matched together. The appended track segment is used as a new source track segment for the next extrapolation.

A software application of the extrapolation method is found in the Omega particle detector system [65]. The aim of the program was to find track and vertex (interaction point) of the particles. The hits in the various layers are connected together subsequently from the outermost detector layer to the centre. Missing hits are replaced by their predictions thus avoiding to extrapolate from one chamber to the next but one. Of course this trick is done only for a low, limited number of missing planes. The computing time is proportional to a number between n and n^2 while n is the number of hit coordinates.

A software track finding strategy for superlayered drift chambers for operation at the SSC [66] using the track following method is described in [67]. The drift chamber system consists of superlayers arranged in concentric cylindrical layers surrounding the interaction point. Each superlayer is composed of eight drift tube layers, thus a

superlayer delivers eight drift times. The track segments (for each superlayer, hit location and bend angle) are derived from these drift times. Two track segments are linked together using a parabolic extrapolation. If the two track segments are found to lie on the parabola within a given maximum deviation the link is considered established. Track segments which belong to each other and form a complete track are called chains. The procedure of creating a chain entails the formation of a new link between a track segment of an existing link and a track segment of the next chamber. A new link is established if the two track segments involved lie on a parabola and are consistent in curvature with the parent track segment link. The process is repeated until either the innermost chamber is reached or more than a given number of layers without acceptable hits are encountered or more than one acceptable link is found from a given parent link. Once a complete chain is found it is validated. The entire track must lie on a circular track model.

Software triggers or software track finders employ track following methods quite often. In off-line applications complex algorithmsare implemented since high calculation speed is not the main requirement. However, a hardware solution employing such a method does not exist. Considering the recent progress of digital technology with respect to processing speed and integration capability it was decided to investigate possible applications of a track following method for the track finder.

Using track following methods for the track finder

Due to their high calculation time and large amount of hardware complicated algorithms like high order fits as described previously must be ruled out. It was decided to investigate a track following method where single data entities (track segments) are matched to each other to form track segment pairs. Track segment pairs are finally assembled to full tracks.

The pairwise matching of track segments is performed by extrapolation. Thus the method is referred to as extrapolation method. The bending angle given by the track segments is used to calculate the deflection of the particle from one station to the next. This is possible because the magnetic field and the distance from the interaction point are known. Particles not originating from the interaction point will be filtered out because the extrapolation fails. Using the spatial coordinate of the track segment measurement allows to calculate an extrapolated hit coordinate in the next chamber. If the difference between extrapolated value and the actual hit coordinate is found to be within a given threshold the extrapolation is considered successful.

Is it feasible to perform an extrapolation with reasonable hardware complexity and within short enough a time?

As discussed earlier there are two possibilities to cope with the high bunch crossing rate. Since introducing dead times into the trigger system is not permissible each processing unit must be able to accept new data every bunch cycle (25 ns). This can either be achieved by the ‘round robin’ architecture (where a number of identical processing units take turns in accepting new data) or by employing a pipelined system (where no processing step lasts longer than the system clock period). In most hardwired custom designed digital processors the processes can be split into several smaller operations which allows the use of a pipeline architecture. As a consequence the

hardware effort is reduced. In this context it is required to perform the calculation of the deviation and all other subsequent processing steps within 25 ns each.

Even if material effects (multiple scattering and energy loss, see chapter 4.3.1.) are not taken into account calculating the deflection by means of arithmetic logical units is very costly and takes too much time. The calculated deviation $\phi_{deviation}$ is a function of the bending angle ϕ_b. It is common practice to perform complex calculations by means of memory based lookup tables. The data resolution of the bending angle and the threshold requires a 256 times 8 bit memory (eight bit input eight bit output) with an access time of considerably less than 25 ns. Static RAM with these specifications are available commercially at low cost. The advantage of using memories is that energy loss effects can be taken into account without large excessive.

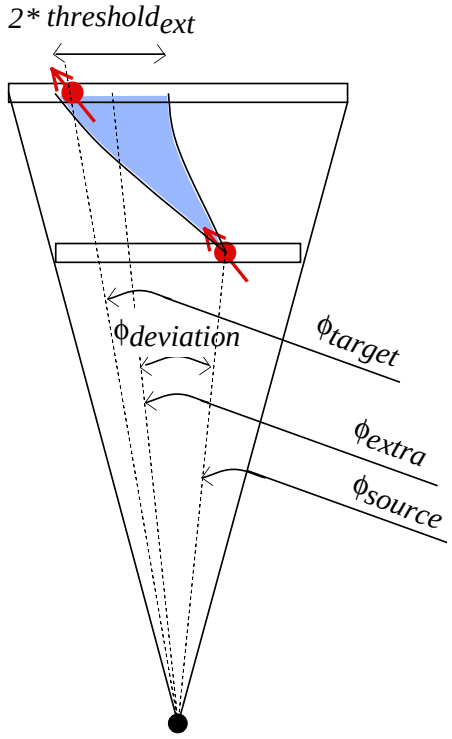


Fig. 4.6.: If a track segment is found to be within the extrapolation window given by ϕ_{extra} and threshold$_{ext}$, the extrapolation is considered successful.

After the deviation $\phi_{deviation}$ is determined one has to compare the extrapolated hit coordinate ϕ_{extra}, derived from the source track segment coordinate ϕ_{source} and the deviation $\phi_{deviation}$ (by simple addition) with the actually measured track segment coordinate ϕ_{target}. The required eight bit arithmetic operations can be performed easily within 25 ns (see fig. 4.6).

Although the resolution of the spatial coordinates ϕ is eleven bits only eight bits are necessary to form the track segments pairs. This is due to the fact that the deviation $\phi_{deviation}$ has a resolution of eight bits. Thus maintaining the higher accuracy for the spatial coordinates ϕ is not meaningful. Moreover the high resolution is only needed for the determination of the transverse momentum p_t. The track can be assembled without loosing performance with a lower resolution of eight bit. When assigning a p_t-value the higher ϕ-resolution is used.

How many extrapolations have to be conducted at the maximum?

As in any track following method missing hits cause severe problems. In software solutions there are several possibilities to come by this problem. One is discussed earlier in the example of the Omega Particle Detector system [65]. Missing hits are replaced by their predictions. For one track candidate this is allowed only in a limited number of chambers. Another is described in [67]. The search starts by connecting a pair of track segments which potentially belong to the same track. The process is repeated by forming a new link between the inner hit and the next superlayer thus building a chain of connected hits. For chains that have not been followed over their full potential length, a second (global) phase is entered where the track found so far is used to define a road within which one looks for further hits.

Implemented in hardware the described methods are too time consuming and require too much hardware. Extrapolations between all stations pairings must be carried out in parallel. The resulting track segment pairs are assembled to full tracks in a second processing step. As described later a valid track consists of at least two track segments belonging together. Hence extrapolations between six station pairs; 1-2, 1-3, 1-4, 2-3, 2-4 and 3-4; are necessary. Since particles cross detector segment boundaries extrapolations have to be conducted from a given track segment to track segments in the same detector segment as well as to at least five neighbour segments (see figure 4.7). Thus in a detector segment 144 extrapolations have to be performed (6 (stations pairs) · 6 (detector segments) · 2 (source tracks segments per chamber) · 2 (target tracks segments per chamber)). Given the integration capability of digital technology this number is reasonable.

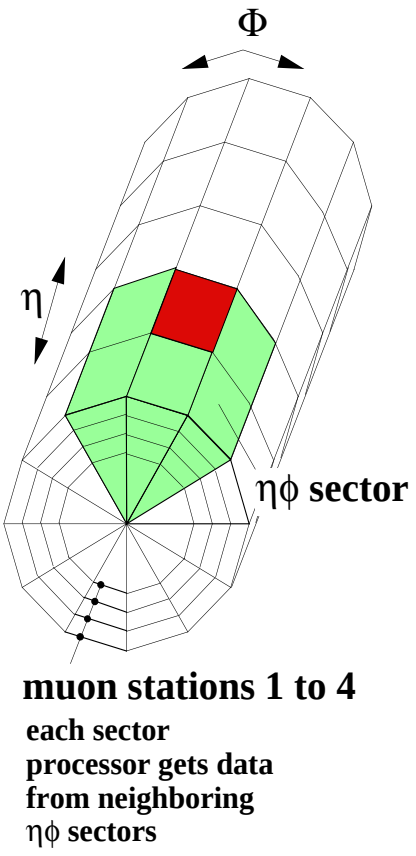


Fig. 4.7.: A muon can cross sector boundaries.

Beginning from the inside (or the outside) of the detector track segments pairs which belong together are assembled to full tracks. This is done by looking for a target track segment being a source track segment of another successful extrapolation. Since extrapolations are done for all station pairings tracks are also found in case one or two hits are missing.

While pattern matching methods, histogram methods or neural networks usually deliver the wanted track property directly, the extrapolation method requires three steps;

- extrapolation or pairwise matching of track segments to full tracks
- assembling track segment pairs
- assigning (or measuring) the track properties like transverse momentum and location.

After linking the track segment pairs to full tracks the information which track segments are involved is still available. Thus the track segment data can be used with its full resolution to assign the transverse momentum p_t. Again calculating the momentum via the track curvature is too cumbersome. Memory based lookup tables may be used.

All other previously mentioned methods (pattern match, histogram method, but also neural networking) work in a global or direct data processing way. The input data is directly set into relationship to the wanted features. Very often an advantage in this case it complicates the implementation. When performing a pattern match one looks for track segment combinations which belong to one track (track finding). For this task a lower resolution may be used, but after a pattern has been recognized the link to the original track segment data is not available anymore. Thus patterns must be formed by

track segment data with full resolution so that each pattern allows directly the determination of the wanted features. Therefore the number of patterns is very high. In case of the histogram method the problem is similar.

However, in case of the extrapolation method the process of measuring the transverse momentum p_t is partitioned in three steps; pairwise matching of track segments by extrapolation, assembling track segment pairs to full tracks, and assigning (or measuring) transverse momentum p_t. This approach allows for a flexible use of the resolution in each step according to the requirements. The extrapolation and track assembly is performed with reduced resolution. As a consequence the hardware effort is smaller and the execution time shrinks. For assigning the transverse momentum p_t the full resolution of the track segment measurements is still available.

Conclusion: Track following method (extrapolation) for the track finder processor

Although the algorithm of the extrapolation method is probably the most complicated among those discussed in the previous chapters the implementation is based on a simple idea and thus is the most feasible one. Moreover hardware implementation of the extrapolation method promises to ensure a functionality with no restrictions with respect to measurement accuracy of the transverse momentum. The features given by the DT trigger primitive generator can be fully exploited. The extrapolation method was decided to be the base line for the track finder processor. It was suggested to the CMS collaboration and published in [2, 25, 26, 68, 69].

4.1.5. Conclusion: Track finding methods

While the method of pattern matching theoretically may be applied for the track finder the pattern multiplicity is too high. This is due to the high required p_t resolution and the resulting high chamber resolution.

The theoretically possible features of the histogram method cannot be exploited to the full extent for the track finder. It would have to be adapted with a loss in performance. Moreover the required p_t resolution would require a too complex hardware layout.

Neural networks seem to be the promising technology of the coming time in image processing applications. However, today's available logic does not allow to implement a large scale trigger system. Moreover it is not obvious that a complex algorithm can be implemented in an artificial neural network now or within the next few years.

The extrapolation method allows to fully exploit the features of the drift tube trigger primitive generator. Employing a principle which makes use of the zero suppression of the trigger primitive generator allows to implement a system with reasonable hardware effort providing a precise transverse momentum and location measurement of the particles within a small time limit.

The extrapolation method will therefore be further pursued. However, it has to be proposed that progress of technology shall be observed carefully. In a few years time technology may be advanced enough for implementing alternative track finding and feature extraction methods. However, as has been shown in the previous chapters the architectural effort for the pattern matching and histogram method is prohibitive. The

Method	Method applicable	Hardware complexity	Necessary technological progress	Remarks
Pattern comparison	yes	very high	very high	too large number of patterns, too large number of logic bins (high resolution of the chamber)
Histogram	no	very high	very high	too high resolution of the chambers, too high hardware complexity,
Neural networks	?	low	high	promising method, but technology not advanced enough, too slow, algorithm (probably) too complex
Extrapolation	**yes**	**low**	**none**	**full resolution of the chamber can be used,** **employs today's technology,** **fast,** **simple implementation principle**

Table 4-1: Conclusion of the properties of the previously discussed feature extraction methods.

advantage of the extrapolation method employing only a fraction of architectural resources is tempting. Thus using the time before the start of the experiment to optimize the algorithm and its hardware realization is a very prudent option. This is especially true as the trigger system of the experiment is a key point of success or failure. Table 4-1 concludes the properties of the previously discussed feature extraction methods.

4.2. Implementation technologies

Now once the track finding method is chosen possible hardware implementations must be evaluated. In this chapter they are described and their features discussed. Note that it is necessary to find an implementation strategy which allows to achieve total functionality of the track finder system employing today's technology. The first level trigger is a crucial part of the CMS data acquisition. The success of the whole experiment depends in great deal on the trigger system. The functionality of the trigger directly influences the data quality. It is important not to be dependent on future developments and progress of implementation methods. A technology must be found which reliably processes the chosen algorithm with the given specifications. There must be enough time to test and improve the system before the start of the experiment. However, as other technologies improve sufficiently and the chosen track finding methods can be implemented without restrictions, they certainly cannot be ruled out in advance. Moreover, if progress of alternative technologies advances in time even the track

finder method may be adapted.

Two possibilities to implement high speed digital algorithms seem obvious. One is to employ commercially available processors. In this approach one attempts to use the fixed machine architecture of the processor in an effective way. However, in some cases the processor architecture is hardly suited to the algorithm. On the other hand a microprocessor is able to process any computable function with a maximum of flexibility.

The opposite extreme is to design custom circuitry. In this approach the entire machine structure is tailored to the application. Operation is efficient and fast. The drawback is that the circuitry is limited to a single or small number of specific algorithms. The flexibility of reprogramming the processor is reduced to a minimum.

In the following sections the features of commercial digital signal processors (DSP), programmable (reconfigurable) logic and application specific integrated circuits (ASIC) are discussed.

4.2.1. Commercial digital signal processors

Commercially available digital signal processors are not designed to meet criteria of high processing speed with respect to first level trigger applications. The processors are capable to process complicated operations on a large amount of data, exhibiting a relatively long processing time in the order of microseconds. Use of digital signal processors is not common in high speed data acquisition applications. However, an example of the use of a digital signal processors is described for the total energy trigger of DELPHI [70]. Digitized trigger data, corresponding to the energies deposited in 24 sectors of the detector, are read out, added and compared with pre-defined thresholds. The calculation is completed in about 3 microseconds. It should be noted that in case of the detector DELPHI [71] the bunch crossing period is 22 μs and the trigger system [72,73] is not dead time free. However, the function of the track finder must be realized in a dead time free manner.

In [74] the application of a commercial image processing system for triggering in future LHC experiments was studied. It is obvious that such a system may only be applied if at all for the second level trigger of CMS. The event rate is estimated to be up to 100 kHz at the second level trigger input. The study shows a successful implementation of several critical feature extraction algorithms at the desired system frequency. However, the detector design of future LHC experiments does also include second level trigger tasks for which commercial modules will not provide a solution [74].

Conclusion: Using commercial digital signal processors for the track finder

Due to the low calculation speed and the inflexible hardware architecture of commercially available digital processors their use in first level trigger applications for LHC experiments can be ruled out at the present moment.

4.2.2. Programmable logic - Field programmable gate arrays

Using field programmable gate arrays (FPGA) offers advantages in two ways: flexibility of software and performance of hardware processors.

In the last ten years the application of field programmable gate arrays (FPGA) became common in the design of custom specific electronic products. FPGAs are based on a matrix of configurable programmable logic blocks (CLB) which are connected with each other by a programmable interconnection matrix [75].

Standard logic integrated circuits (LSTTL,...) are more and more being replaced by FPGAs. The high integration capability at relatively low cost permits a compact system design. The advantage of reprogrammability of FPGAs ensures a large degree of flexibility whenever a fast change of algorithm is desired. The modified gate array structure can be loaded into the integrated circuit via a low bandwidth bus within a few fractions of a second. Moreover test procedures (test vector loading, connectivity checks, boundary scan [76]) can be performed automatically using the same access method. The amount of necessary glue logic is minimized. Interconnections between logic circuits are greatly reduced. However, the processing speed of FPGA is smaller than some faster families of discrete logic circuits. In certain gate array architectures the routing delays between logic units within the integrated circuit are not constant. Routing may even become a problem in large chip designs [43]. Digital designs which want to take advantage of the high integration capability must be implemented carefully to fulfil the required timing constraints. Very often the designer has to interfere manually with the normally automatic place-and-route process.

In high energy physics triggers the field of applications for FPGAs is large. An example employing FPGAs in a first level trigger design is the fast pipelined trigger for the H1 experiment at HERA [40]. It is also discussed in chapter 3.2.3. Note that at HERA the bunch crossing frequency is 96 ns (compared to 25 ns at LHC). Trigger algorithm and chamber geometry are not as complex as in case of the track finder processor of CMS.

Using FPGAs for the track finder processor

As demonstrated previously implemented systems the application of FPGAs in 40 MHz dead time free environments is already feasible. Although the integration capability of modern FPGAs is high still their possible application is largely dependent on the architectural complexity of the entire system. This is true for both

- the aspect of implementing the trigger (or measure) algorithm in a hardwired system and
- the aspect of mapping the *process space* onto the hardware level.

The *process space* is given by the spatial range of the measured processes and the experimental time which has to be observed or followed to obtain a complete image of the entire processes. The *process space* may be divided into several segments with the aim to share a minimum of data between the segments. This is important because often the feature extraction requires evaluation of large amounts of data and performance of complex algorithms. In most hardware solutions the operations cannot be performed on all data at once. Systems must be partitioned in hardware segments where each of the

segment processors performs operations and measurements on a data subset. A global unit collects all segment data and performs a final evaluation.

The task of implementing the algorithm in hardware has consequences on the number of necessary gates or logic blocks. If the realization of digital operations belonging to each other exceeds a certain gate count the logical unit must be partitioned into several physical units. This bears potential disadvantages.

- Firstly the input data may have to be delivered to all circuits that are now distributed over several separated integrated circuits. This causes fanout problems and timing misalignments on the system board level.
- Secondly data interchange between processing units takes away I/O pins. The pins are lost for input and output data transfer. As a consequence less input data can be fed into one circuit thus limiting the amount of logic that can be placed in the circuit. A compromise between amount of logic and I/O must be found to optimize these system parameters. Moreover data transmission provokes deadtime. When fast pipelined systems are implemented the time for driving output signals, transmitting them to the next state (on the same board) and accepting them (in the subsequent flip flop stage) takes as much time as a system clock period in many cases. No time is left to perform an operation and to deliver data into the next integrated circuit within one pipeline step. Additional deadtime is caused as a consequence. Entire processing units have to wait for the neighbour-data to arrive. Storage of intermediate results during a wait state requires additional (memory-) logic.

The aspect of mapping the *process space* onto the hardware level has similar consequences.

In most experiments the time when a process occurs is known with a precision of less than 100 ps. The timing and control system makes available a precise clock that is able to reference all synchronized processes to about +/- 150 ps. In case of LHC every 25 ns an event occurs and all interesting particles have left the detector by the arrival of a subsequent collision 25 ns later. Hence one of the *process space* segment boundaries is defined by the bunch crossing period. Once the general flight direction is known the tracks of particles originating from the interaction point (vertex) can be estimated within certain limits. If there is no external magnetic field force influencing charged particles (magnetic field) tracks are straight lines originating from the interaction point. Often the detector chamber geometries are adapted to the track topology. Chambers may be designed such that their surface is perpendicular to a straight line originating from the interaction point. Such detectors are known to have *projective geometries*. Hardware, connected to projective chambers, may be designed responding to this evident gain of simplicity and processing time. Data exchange between processing units is automatically minimized because tracks hit chambers perpendicularly. Hardware segmentation then resembles detector segmentation and hence also *process space* segmentation. Conservatively the entire *process space* is mapped onto the hardware level with a minimum of overlap to other segments.

If for certain reasons the *process space* cannot be reflected by the hardware geometry in an elegant way, the overlap of *process space* segments must be duly increased. Input data must be shared between several parts of the system. In other words logic segments must overlap in order to be able to find all possible particle tracks. This

decreases processing speed due to the necessary data transfer and the increased amount of data input into each processing unit.

In CMS calorimeters as well as resistive plate chambers (RPC) [2] have projective geometries in both projections, $r\phi$ and rz. Drift tubes do not have projective geometries in the rz plane (see fig. 5.17on page 85).

Given the geometry of the drift tube system one can predict a large amount of necessary interconnections between processing units. Although the integration capability of FPGA is considerable one cannot expect to be able to place all necessary logic inside one single FPGA. An example for a state of the art (1997) field programmable gate array is the XILINX XC4025 [43]. It contains 1024 cellular logic blocks or a equivalent of 25000 gates. The number of gates in an FPGA can not be compared directly to the gate number in an ASIC. This is due to an inefficient use of gates within the FPGA in order to retain a maximum of programming flexibility. The maximum I/O pin count of the device is 256. As described in chapter 6 the logic of the track finder processor fits into 19 XILINX devices.

When employing FPGA the time for driving and accepting I/O signals compared to the execution time of logic operations is high. This means when working at an operation frequency of 40 MHz at least one cycle is lost to the transmission from one FPGA to another.

Another critical point is the relatively low I/O pin count of available FPGA products. The chamber geometry and the high resolution require a high number of input bits to be made available to logic units. Interchanging data between several integrated circuits is not feasible due to the slow intercommunication. However, input pin multiplexing is a possible solution. Time sharing with a pin reduction factor of 2 requires a I/O bit rate of 80 MHz. However timing specifications of FPGA products reveal that I/O-frequencies for synchronous designs in excess of 50 MHz are not yet available.

FPGA-arrays - Programmable active memories

An architecture which employs an array of FPGAs, the so called Programmable Active Memory [77], eludes the lack of available gates and I/O pin count but preserves the flexibility of reprogrammability.

The structure of cellular arrays (employed inside each FPGA) is extended to the system level. Cellular arrays consist of an array of programmable logic cells joined by fixed or programmable interconnections [75]. The architecture of cellular arrays was discussed firstly in the mid 1960s [78,79,80,81].

Programmable active memory designs are synchronous circuits: all registers are updated on each cycle of the same global clock. The maximum speed of a design is directly determined by its critical path. This of course varies from one design to another. The critical point when employing FPGAs remains the I/O bandwidth of each component. Of course the more complex a programmable active memory system the longer the propagation time of the signals throughout the entire system. This is especially true when data exchange between logic cells is required to be large.

Very often programmable active memory systems can be programmed by a high level language as C or C-like languages. A compiler translates the programme

directives, provides the interconnection matrix programming files and finally the place and route files for the logic devices. The aim of the software programme developments is to create an environment where programmers can design the algorithms on an abstract level without thinking about hardware implementations.

Several studies were undertaken to investigate whether programmable active memories may be employed in LHC triggers. In [77] two systems are evaluated. The first, DECPeRLe-1 [82], is a multiple purpose programmable active memory whilst the second, Enable++ [83,84,85,86], is a system especially tailored for a specific application in the second level trigger of experiment ATLAS/LHC [87].

The DECPeRLe-1 employs a 4 by 4 matrix of XC3090 (XILINX) chips. Each chip has 16 direct connections to all of its four nearest neighbours. Interconnections are done via two 16-bit buses. The system has four 32-bit wide external connections which are used for establishing real time links at rates up to 33 MHz.

The Enable++ system employs a 2 by 2 FPGA-processor matrix. The processors are connected with each other through a crossbar switch. In addition to the nearest neighbour interconnections different global connection topologies can be achieved. Each FPGA-processor consists of a 2 by 2 FPGA matrix (XILINX XC4013), with each processor connected to its nearest neighbour. In addition 96 high-speed 128 kByte synchronous SRAM chips are located at the very interconnections between neighbouring FPGAs. A scheme like that is suitable for pipelined lookup tables (FPGA1-RAM-FPGA2) with up to 17 input bits and 16 output bits. The computing array is connected via crossbars to a bus interface. This circuit decouples the synchronously operating array from asynchronous bus protocols on the backplane and buffers incoming or outgoing data. Connections to the backplane are done using eight 40-bit buses which can be operated synchronously up to the full system speed of 50 MHz.

Both systems were evaluated for the application in the second level trigger of experiment ATLAS [77,86]. Note that the event rate at the second level trigger of both experiments CMS & ATLAS is only 100 kHz. However, the requirements on the feature extraction algorithms of the second level trigger in ATLAS are comparable to the ones of the first level trigger in CMS. The event rate at the first level is 40 MHz. In experiment ATLAS the feature extraction in the silicon tracker, transition radiation tracker and a calorimeter were tested using the two systems.

All systems were found to be suitable for the second level trigger. However, the execution time of the processors was in the range of 10 μs.

Conclusion: Using FPGAs and/or programmable active memories for the track finder processor

The concept of programmable active memories has been presented as a viable alternative for application in the second level triggers at the ATLAS experiment. Two already realized devices, DECPeRLE-1 and Enable++, demonstrated the proper functionality at an event rate of 100 kHz. However, the present state of the art of reconfigurable logic, FPGA or arrays of FPGAs, does not allow the design of a first level trigger. Whilst it presently appears to be possible to design a synchronous system employing FPGAs with a clock frequency of 40 MHz the data transfer bandwidth

within processing units is still limited. Hence the execution time of trigger algorithms is too high to run first level trigger applications on programmable active memories.

However, in all branches of the electronics industry other than the consumer market, the use of FPGAs is on the rising edge despite the fact that their cost is higher than the one of ASICs in volume production. In 1992/93, FPGAs were the fastest growing part of the semiconductor industry [77], increasing its output by 40% compared with 10% for chips overall. As a consequence, FPGAs are at the leading edge of silicon devices.

Today's largest SRAM based FPGAs incorporate about 25000 gates. FPGAs exhibit a maximum I/O count of about 300 pins and operate synchronously with a frequency of about 50 MHz. Already now announcements are being made for reconfigurable products employing 100k gates and more with an I/O pin count of 375. These devices operate at an internal frequency of up to 100 MHz. An example is published in [88]. Predictions are being made that the leading edge FPGA of 1996 operating at 50 MHz will work with a frequency of 200 MHz by the year 2001 [77]. Taking into account this rapid development a final implementation of the track finder processor employing reconfigurable logic devices may be considered in the future. However, it was decided to focus development efforts on a technology available now in order to be able to verify system functionality today.

4.2.3. ASIC (Application Specific Integrated Circuit)

It is obvious that a system of the size and data processing capability like the track finder processor certainly may not be designed using discrete digital circuits. Note that in each bunch crossing the drift tube trigger primitive generators forward 10 kbit data to the track finder processor. This corresponds to a data rate of about 400 $Gbit \cdot s^{-1}$. Recently the production of application specific integrated circuits (ASIC) became cheaper and commercial software tools for development and simulation are available also to smaller customers. The advantage of using ASICs is the huge number of usable gates and the low execution time.

Interconnections between processing units are very often the limiting factor in digital systems as they are time consuming and push cost [89]. Processing units share the input data or one unit fans out data to multiple subunits. If a maximum of logical circuitry could be assembled within the same package the data transfer problem is reduced to an interconnection problem within the integrated circuit. This is less time and power consuming. Moreover the error rate due to external interconnections (connectors, crosstalk) is greatly reduced. Maximizing the size of processing units will reduce the total connectivity requirements. Whilst the amount of available gates or logic cells within an ASIC is generally sufficient, it is the limitation on the I/O pin count which does not permit true *one chip* solutions. The most obvious way to increase the input data rates into a chip is multiplexing of input pins. The I/O bit rate then becomes higher than the internal operating frequency.

The drawback when employing ASICs is the limited flexibility with respect to changes of an implemented algorithm. Of course one can foresee several sets of algorithms or variations of one algorithm in the hard wired ASIC processor. However, the basic architecture of an ASIC system is fixed and cannot be changed. Memory

based configuration cells for different setups or lookup tables may provide a certain degree of architectural flexibility.

On the other hand it has to be stated that at the first level trigger one does not have to expect *major* changes of the algorithm once the experiment has started. The track finder processor does not apply any cuts nor does it determine the relevance of an event depending on the event topology. Its task is to extract features of tracks. Hereby raw frontend data are converted to quantities (features) that are meaningful from the physics point of view. Features are variables containing the relevant physics message like cluster or track parameters - transverse momentum and location - that can be used for a later decision of the level one global trigger. Due to the fixed architecture of the detector and the muon system the kind of features to be extracted will basically remain unchanged. In the specific case of the track finder the task will always be muon momenta measurement and the location of the particles situated on what is believed to be a consistent track. New trigger algorithms may be necessary in the future at the level of the global trigger when now unknown physics processes need to be covered.

The track finder processor must provide a certain degree of programmability with respect to the chamber configuration and performance. In case of misalignment of the chamber system the trigger processor must be able to compensate for this effect. Dead chamber parts or noisy channels will require some sort of masking.

ASICs with a typical propagation of less than 300 ps are available commercially. I/O pin counts of more than 300 pins are state of the art. The gate count of modern ASICs exceeds 300000 gates. An I/O frequency of 250 MHz can be achieved [90].

Conclusion: Using application specific integrated circuits (ASIC) for the track finder

All given requirements can be met easily using an ASIC system. Due to the high integration capability and the high processing speed the ASIC solution allows an implementation of the track finder algorithm with no restrictions to performance and functionality even with today's available technologies. Chapter 7 proposes an implementation of the track finder system using ASICs. Required hardware needs and the estimated processing speed are discussed.

4.2.4. Conclusion: Implementation methods

Table 4-2 shows the properties of the discussed implementation technologies.

Commercially available digital signal processors (DSPs) are not suited for application in first level trigger systems. Their fixed processor architectures and limited input bandwidth are designed for complex algorithms running processes in a comparably high time. Industry might however produce dedicated image processing systems but it cannot be expected that specifications, in particular the bandwidth requirement, will meet the criteria of first level trigger processing.

Programmable configurable logic components like field programmable gate arrays (FPGA) or programmable active memories (PAM) are subject to ambitious technological advance. In the near future it may reasonably be expected that component development will allow all the stringent requirements given by the experiment's first level trigger to be met.

Technology	Commercial availability	Speed	Necessary technology progress	Remarks
Digital Signal Processors	yes	very slow	very high	complex algorithms feasible, but too slow; (fixed) architecture not adaptable to requirements; pipelined structure not possible on large dataset
Field Programmable Gate Arrays FPGA	yes	slow	little	cheap, reprogrammable; too few I/O pins and too small implementable logic -> boundary problem; large technology progress to be expected
Programmable Active Memories	limited	slow	little	flexible -> programmable hardware architecture; too slow; large technology progress to be expected;
Application Specific Integrated Circuit ASIC	no	very fast	none	very fast; high integration capability; technology of today advanced enough; limited flexibility

Table 4-2: Properties of the discussed implementation technologies.

However, today's application specific circuits (ASIC) are advanced enough for an implementation of a first level trigger system for CMS. Due to the high industrial demand for easily producable and affordable components ASICs will be available with even more advanced technology to the customer in the future.

Given the present state of the art the ASIC solution is the only possibility today. An ASIC-based solution for the track finder processor implementation has already been proposed to the CMS collaboration.

In chapter 6 the implementation of a (necessarily slower) prototype employing FPGAs is described and results are discussed. Chapter 7 deals with a possible high speed ASIC implementation and discusses options for realization.

4.3. Extrapolation method - Feasibility from physics point of view

This chapter deals with the extrapolation method and its performance with respect to the physics requirements. The feasibility of the pairwise matching of track segments is discussed. The requirements for the matching of track segment pairs to full tracks and the measurement of the transverse momentum p_t are shown. However, only the basic principle and results of the calculations and

simulations are shown as they are important for the design considerations of the hardware implementation. A comprehensive discussion by the project's team introduced on page xi of the physics aspects of the algorithms may be found in [22, 25, 26].

Note that the extrapolation method employs a three stage processing scheme. First track segment pairs are formed. Secondly they are linked together to full tracks and thirdly the transverse momentum p_t is measured. This is entirely different from other track finding techniques which employ a global single step method such as neural networks or pattern matching (see fig. 5.2).

4.3.1. Particles in magnetic field and matter

In the following the basic principle of extrapolating a track segment found in one chamber into the subsequent chamber (calculating $\phi_{deviation}$) is described. Particle trajectories in a magnetic field $\boldsymbol{B}$ must satisfy the equation of motion given by Newton's law and the lorenz force

$$\frac{d\boldsymbol{p}}{dt} = Q\boldsymbol{v} \times \boldsymbol{B}, \qquad \text{(Eqn. 4.2.)}$$

where Q is the particle's charge, $\boldsymbol{p}$ its momentum and $\boldsymbol{v}$ its speed. $\boldsymbol{p}$ equals

$$\boldsymbol{p} = m \cdot \gamma \cdot \boldsymbol{v}, \qquad \text{(Eqn. 4.3.)}$$

with m being the particle's mass and

γ the relativistic auxiliary function with c being the speed of light.

$$\gamma = \frac{1}{\sqrt{\left(1 - \frac{v^2}{c^2}\right)}} \cdot \qquad \text{(Eqn. 4.4.)}$$

Neglecting the energy loss of the particle due to the matter it traverses the solution of equation 4.1 is a straight line for a field free region; a helix is found when traversing a uniform magnetic field at some angle. Assuming the magnetic field is parallel to the z-coordinate and a particle is emerging from the interaction point at the origin of coordinates with a momentum $\boldsymbol{p} = \begin{bmatrix} 0 & p_y & p_z \end{bmatrix}$ the resulting particle track can be expressed by equations 4.5, 4.6 and 4.7. x, y and z are the spatial coordinates. t is the time.

$$x(t) = \frac{p_y}{Q \cdot B} \cdot \left(1 - \cos\left(\frac{Q \cdot B}{m \cdot \gamma} t\right)\right) \qquad \text{(Eqn. 4.5.)}$$

$$y(t) = \frac{p_y}{Q \cdot B} \cdot \sin\left(\frac{Q \cdot B}{m \cdot \gamma} t\right) \qquad \text{(Eqn. 4.6.)}$$

$$z(t) = \frac{p_z}{m \cdot \gamma} \cdot t \qquad \text{(Eqn. 4.7.)}$$

The track is a helix with its axis in magnetic field (z-) direction. The projection of the trajectory onto the xy-plane is a circle with radius of curvature given by

$$r_c = \frac{p_y}{Q \cdot B} \quad . \qquad \text{(Eqn. 4.8.)}$$

This relationship allows to calculate the momentum p_y from the radius of the curved track. The projection of the trajectory in the yz-plane has sinusoidal shape. It is given in equation 4.9 for particles originating from the interaction point at $z_0 = 0$.

$$y(z) = \frac{p_y}{Q \cdot B} \cdot \sin\left(\frac{Q \cdot B}{p_z} z\right) \qquad \text{(Eqn. 4.9.)}$$

However, the sine's argument is small for muons reaching the muon detector in CMS - it is in the order of $3 \cdot 10^{-2}$ for $z < 7$ m. Thus sine(x) $\cong$ x and y(z) can be approximated by equation 4.10.

$$y(z) \cong \frac{p_y}{p_z} \cdot z \qquad \text{(Eqn. 4.10.)}$$

Thus the yz- and also the xz-projection of the track are almost straight lines.

The coordinates of the interaction point, the magnetic field and the coordinates of a hit ϕ in a station (see chapter 2) together with the incident angle ϕ_b define the trajectory of the particle. Since the detector exhibits almost full radial symmetry the relative deviation $\phi_{deviation}$ from one chamber to the next is solely dependent on the bending angle ϕ_b in the chamber and can therefore be determined. The situation is more complicated in certain regions where the magnetic field is no more homogeneous, particularly in the forward region (see fig. 2.2).

Energy loss in matter

Fig. 4.8.: Muon track in magnetic field in matter.

Since muons, when traversing the muon detector, have to cross the yoke of the magnet they interact with the iron and thus loose energy. This energy loss in matter is caused by Coloumb interaction [20] with the atomic electrons. Due to this effect every time an interaction with matter occurs a certain amount of momentum is transferred and lost. The kinematic equations above assume a constant momentum of the particles. Due to the energy loss of the crossing muons the projection of the particles in the xy-plane will diverge from the ideal circular shape and will resemble a spiral shape with decreasing radius (see fig. 4.8). The amount of energy lost by a charged particle that has traversed a fixed thickness of absorber will vary due to the statistical nature of its interactions with individual atoms in the material. The energy loss per traversed distance in matter *dE/dx* can be calculated as an averaged value for each material, particle type, and particle energy.

δ-rays

Is the energy transfer between incident particle and atomic electron high enough for causing ionization the emerging electron is called δ-ray. However, collisions with small energy transfer are much more likely to occur. Anyway δ-rays are very unfavourable for muon identification and momentum measurement. The δ-electrons will also be detected by the drift chambers which causes additional hits not belonging to a true muon track (see fig. 4.9).

Multiple scattering

So far the particle's trajectory was considered the to be continuous. However, the force on the incident particle due to charged particles in the absorber can also introduce small deflections in the trajectory. The Coulomb field of the incident particle provides a momentum for the charged particle in the medium. The incident particle receives an oppositely directed momentum of the same magnitude. It will therefore be deflected by a small angle. It is possible for a particle to traverse a block of matter after having made a large number of small angle collisions. This effect is referred to as multiple scattering. Since each individual small angle scattering is a random process, one expects the mean scattering angle of a particle's track with respect to the incident direction to be zero. [20]

All three effects mentioned, i.e. energy loss, δ-rays, and multiple scattering, can be described mathematically. The effects which are dependent on particle type, particle energy, and type of matter can be calculated. Further information my be found in [20, 91, 92].

Taking into account kinematical equations, energy loss, and multiple scattering one can predict a hit coordinate in a station using the track segment measurement of a previous station.

Once a track is identified and all track segment measurements are known the particles transverse momentum p_t can be determined using the relation between the radius of the curvature and the momentum given in equation 4.11 (derived from equation 4.8.):

$$p_t = Q \cdot B \cdot r_c \qquad \text{(Eqn. 4.11.)}$$

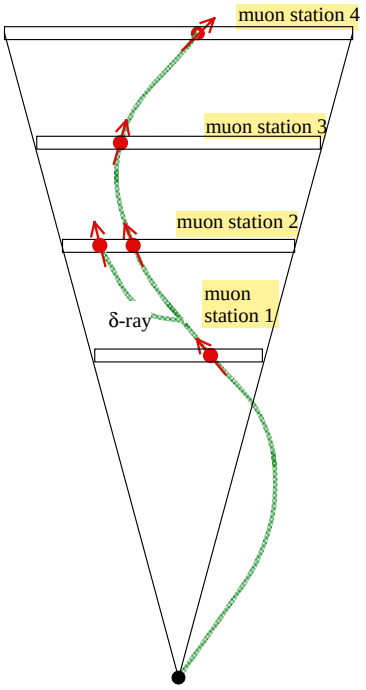


Fig. 4.9.: A δ-ray triggers a second track segment in station two.

For assigning a p_t value to a track material effects etc. must be considered. However, this work only deals with the technical implementation of the trigger. A thorough discussion on the properties of measurement algorithms can be found in [22].

4.3.2. Pairwise matching - Extrapolation

The pairwise matching is based on the principle of extrapolation. Using the spatial coordinates ϕ_{source} and the angular measurement $\phi_{b,source}$ of the source track segment an extrapolated hit coordinate ϕ_{extra} in another chamber may be calculated. If a target track segment is found to be at the extrapolated coordinate within a certain extrapolation threshold $threshold_{ext}$ the match is considered successful (see fig. 4.6).

$$\phi_{extra} = \phi_{source} + \phi_{deviation}(\phi_{b,\,source}) \qquad \text{(Eqn. 4.12.)}$$

$$threshold_{ext} \geq \left|\phi_{extra} - \phi_{target}\right| \qquad \text{(Eqn. 4.13.)}$$

Extrapolation feasibility

Simulations have been conducted to prove the feasibility of the extrapolation between the stations [25]. Fig 4.10 shows the relation between the bend angle ϕ_b in the source station and the relative deviation of the particle track between target- and source-station $\phi_{deviation} = \phi_{target} - \phi_{source}$ for several station pairs. The graphs show unambiguous relationships proving the feasibility of extrapolation between these station pairs. The same condition can be found in all other station pairs except for those extrapolating from station three. Fig. 4.11 shows the situation when extrapolation is done from station three. No unambiguous relationship can be found. Moreover, for

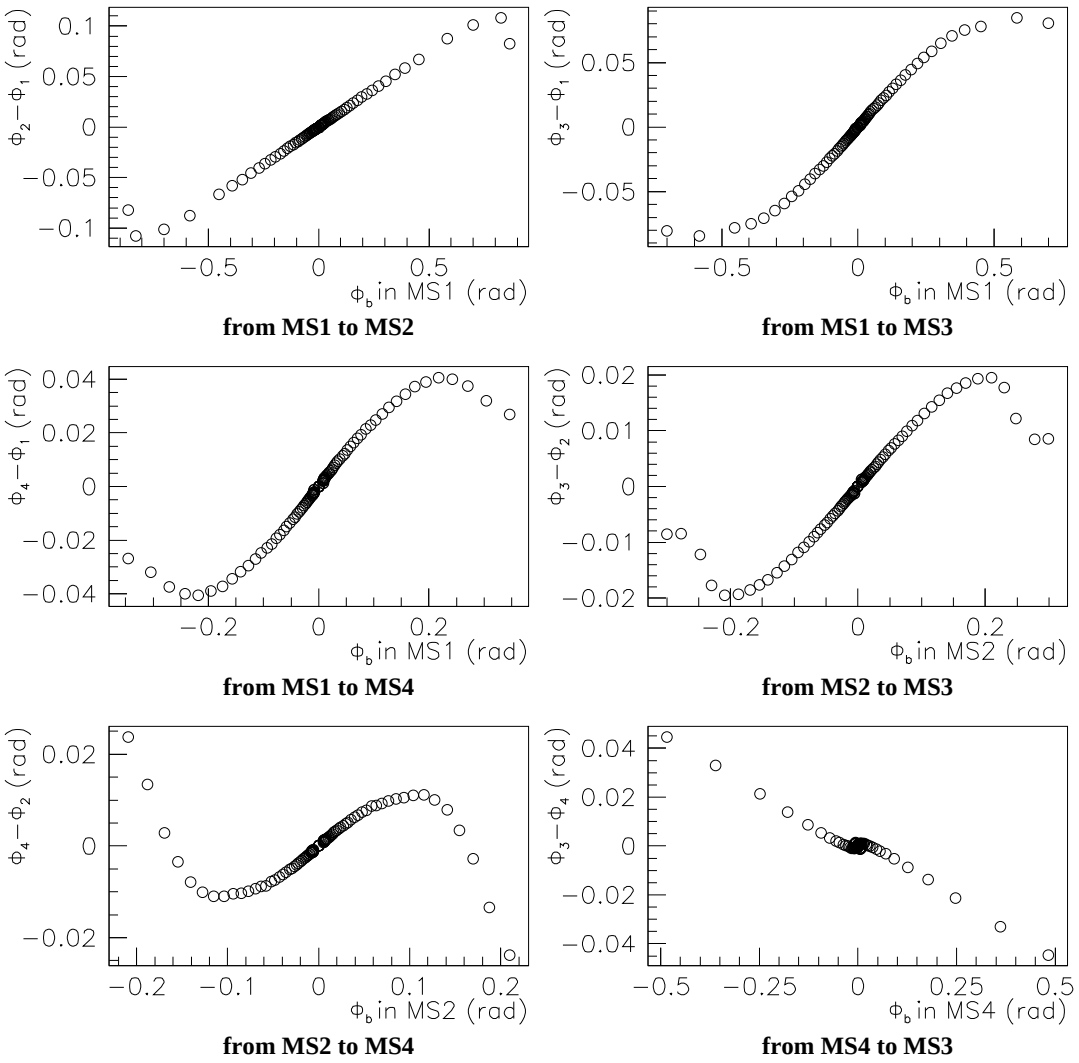


Fig. 4.10.: Relationship between bend angle measurement ϕ_b in the source station and deflection of a muon ϕ_{target}-ϕ_{source}.

small bend angles no prediction can be done at all. This effect is caused by the zero crossing of the bend angle. However, since all other extrapolations are feasible these problems can be circumvented by extrapolating towards station three instead off station three (see left part of fig. 5.2).

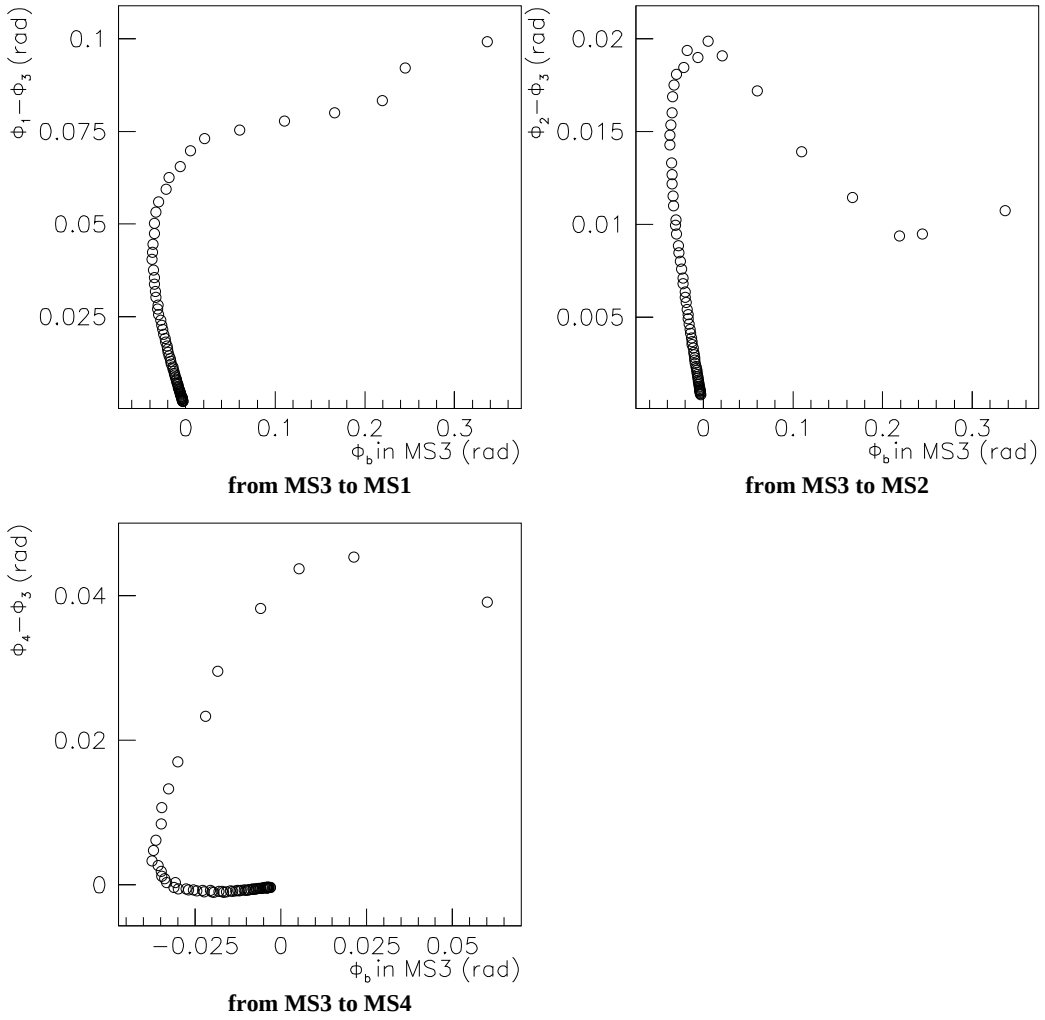


Fig. 4.11.: Relationship between bend angle ϕ_b in station 3 and deflection of the muon is unambiguous.

4.3.3. Track segment assembling

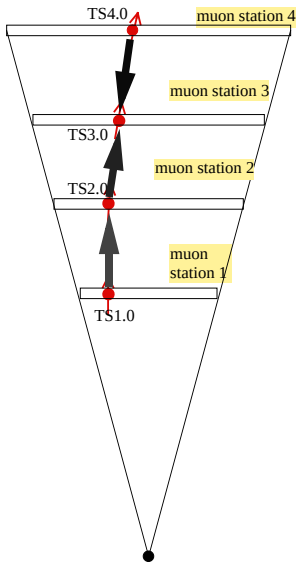


Fig. 4.12.: Matching track segment pairs are combined to a full track.

Once track segment pairs are found they are linked together to full tracks. Track segment pairs of different chamber pairs may be matched to each other if they have one track segment in common and the resulting track segment triplet (or quadruplet) forms a continuous line through the muon system. This scheme is illustrated in fig. 4.12.

Acceptance study

However, due to geometrical inefficiencies and chamber failures missing hits must be accounted for. Simulation studies for the acceptance of muons were conducted for tracks consisting of at least three out of four track segments and for at least two out of four track segments. Fig. 4.13 illustrates the results. When simulating the requirement of three out of four matching track segments only about 82% of all muon tracks are found. An acceptance of more than 95% is achieved when the requirements are loosened to two out of four matched track segments. Consequently the track finder accepts tracks consisting of only two matching track segments. As a consequence even a single track segment pair is already considered a valid track.

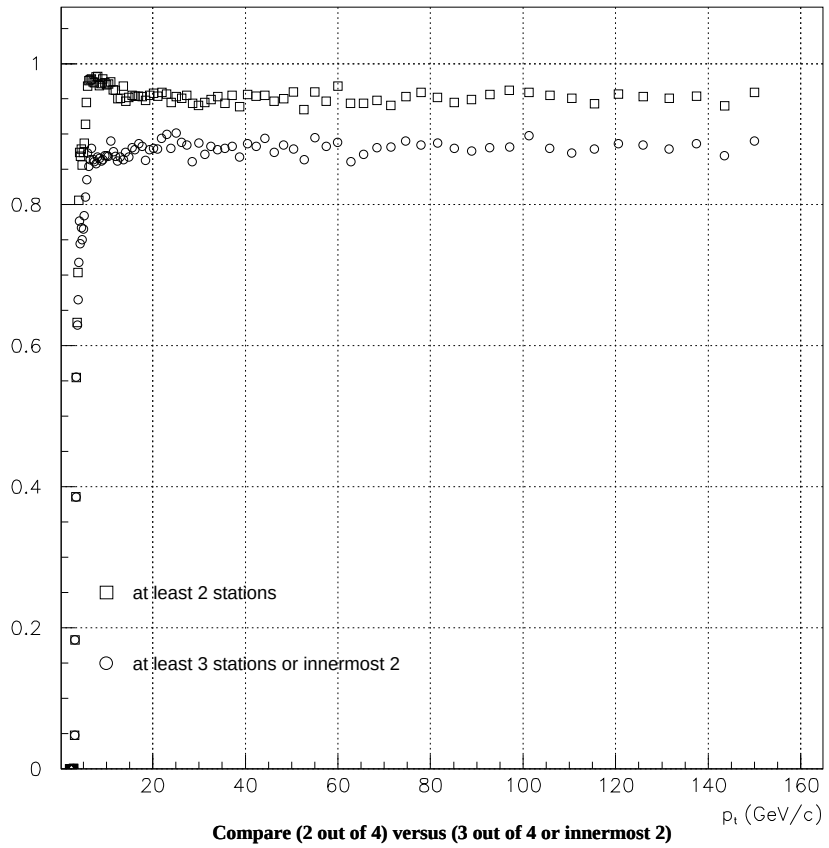


Fig. 4.13.: Acceptance study for track consisting of two and of three out of four track segments.

4.3.4. P_t - assignment

Transverse momentum measurement can be done in several ways once the track segments are matched to a full track [25].

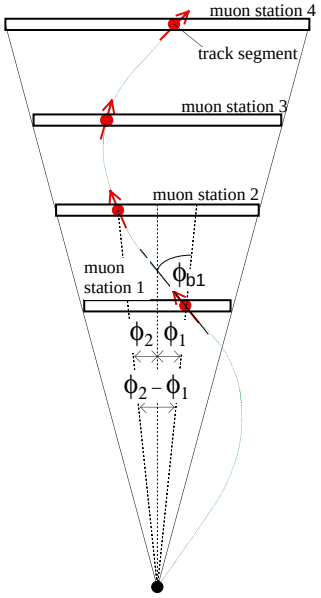


Fig. 4.14.: Using the deflection ϕ_2-ϕ_1 or the bending angle ϕ_b to determine p_t

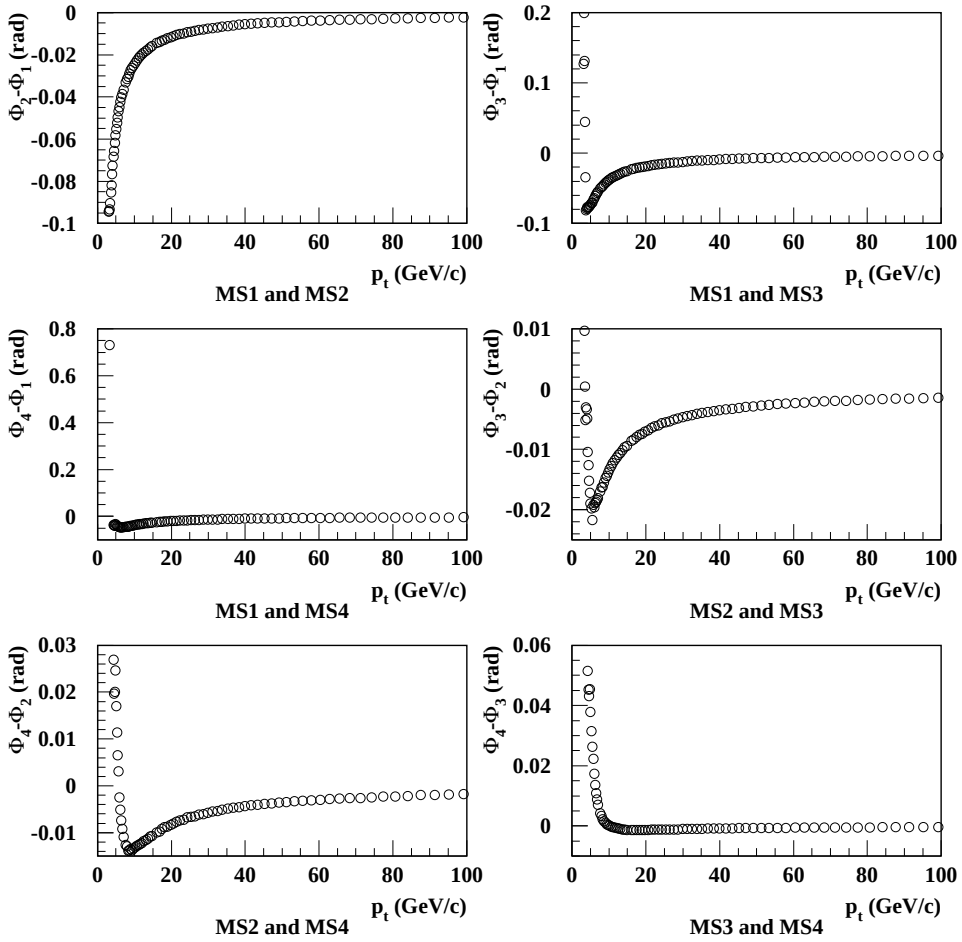


Fig. 4.15.: Difference of azimuthal hit coordinates ϕii-ϕi (deflection of the muon between two stations) over transverse momentum p_t for several station pairs

The method uses the deflection of the muon track between two stations as a measure for the transverse momentum (see fig. 4.14). Hereby two spatial coordinates are sufficient. Fig 4.15 illustrates the relation between transverse momentum and track deflection between two stations. One expects the absolute value of the difference in bending angle to decrease with increasing p_t. However, due to the zero crossing of the bending in station three the absolute value of the difference of the angles first rises (except for ϕ_2-ϕ_1). An unambiguous relationship between difference of bending angles and transverse momentum p_t is shown. However, for some station pairs this method does not deliver the required accuracy. In such a case the angular measurement of a single track segment is used to measure p_t. Fig. 4.16 shows the relation between transverse momentum p_t and bend angle ϕ_b measured in one station.

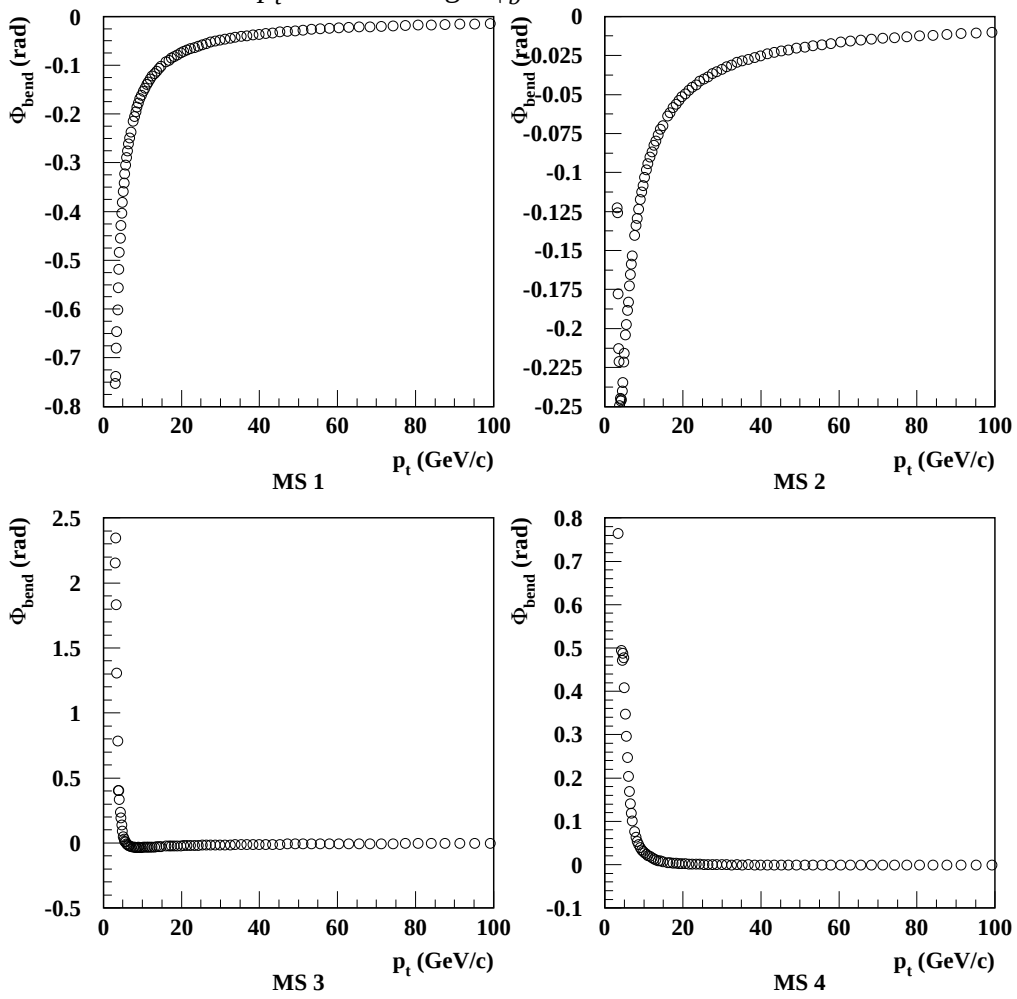


Fig. 4.16.: Bend angle in various stations for μ^+ over transverse momentum p_t.

As described earlier the drift tube trigger primitive generator also delivers a quality information indicating the measurement resolution of the track segments. These quality bits are used to select the most efficient algorithm in order to provide the most accurate p_t-measurement.

The presented simulations show that extrapolation, track assembling and p_t assignment is possible. The extrapolation scheme is a sophisticated method to find muon tracks in CMS.

5 Detailed architecture and functionality of the CMS track finder processor

The basic architecture of the track finder processor is described. The mapping of the chamber structure onto the hardware level is discussed. Every single part of the processor model is discussed in detail. Although this model is technology independent main focus was given to the implementation feasibility. Due to the bending of the tracks and the non-projective geometry of the chamber system muons cross detector boundaries. This requires a large amount of interconnection between processing units. Using the hardware description language VHDL a simulation of the processor model was conducted. The model was used to prove the functionality of the algorithm. Moreover it served to optimize the system partitioning with respect to the amount of interconnections between processing units. Using the VHDL model a FPGA prototype was designed (see chapter 6). In Appendix A an introduction to hardware simulations is given.

5.1. Logic segmentation

The barrel part of the CMS drift tube system consists of 4 muon stations, divided into 5 wheels along the z-axis and into 12 sectors along the azimuthal angle, giving a total of 60 $\eta\phi$-detector segments (see fig. 5.1). The drift tube muon system is described comprehensively in chapter 2 and [2].

Processing of the entire muon data of the detector within one logical unit is impossible and also unnecessary. The amount of 10 kbit data per event yields an input data rate of about 400 Mb/s. The large amount of data to be processed causes a severe integration problem. When splitting up the processor in several physical units an interconnection problem between those units arises. Fortunately, as shown later, a muon track covers only a small number of detector segments.

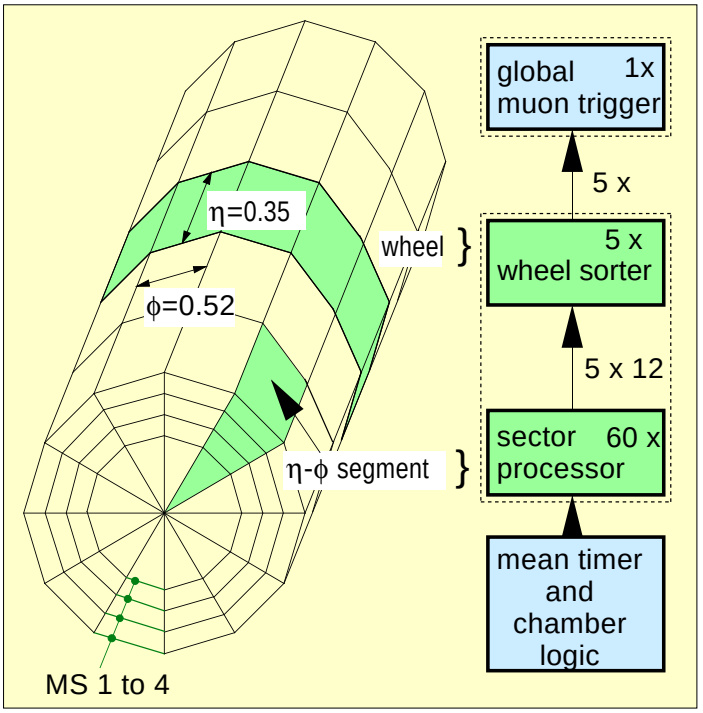


Fig. 5.1.: Logical segmentation of the track finder processor.

In order to render communication between processing units possible at a minimum extent, the logical structure of the chamber system is mirrored inside the track finder processor hardware. In fig. 5.1 the logical segmentation of the track finder processor is shown.

Like the detector itself it is divided into five wheel sorters ($\Delta z \times \Delta\phi = 2.56$ m x 2π). Each wheel sorter is subdivided into twelve sector processors with a segmentation $\Delta z \times \Delta\phi = 2.56$ m x 0.52 rad (30°).

A sector processor matches the track segments identified by the meantimer logic and tries to form complete tracks. If the sector processor succeeds it assigns a transverse momentum p_t to each track, tests whether the track extrapolates to the interaction point and finally forwards up to two tracks to the wheel sorter. Tracks which traverse more than one detector segment are given out by the sector processor of the detector segment from where the track originates.

Of the 12 times 2 possible tracks identified by the sector processors of one wheel only the four tracks with the highest p_t are retained by the wheel sorter. All the information on these tracks - p_t, charge, η, ϕ, quality - is forwarded to the global muon trigger. The latter combines the track finder processor information with the trigger information given by the RPC-system [2, 93].

5.2. Track finder processor algorithm

Fig. 5.2 illustrates the principle of the track finder algorithm. It is divided into three

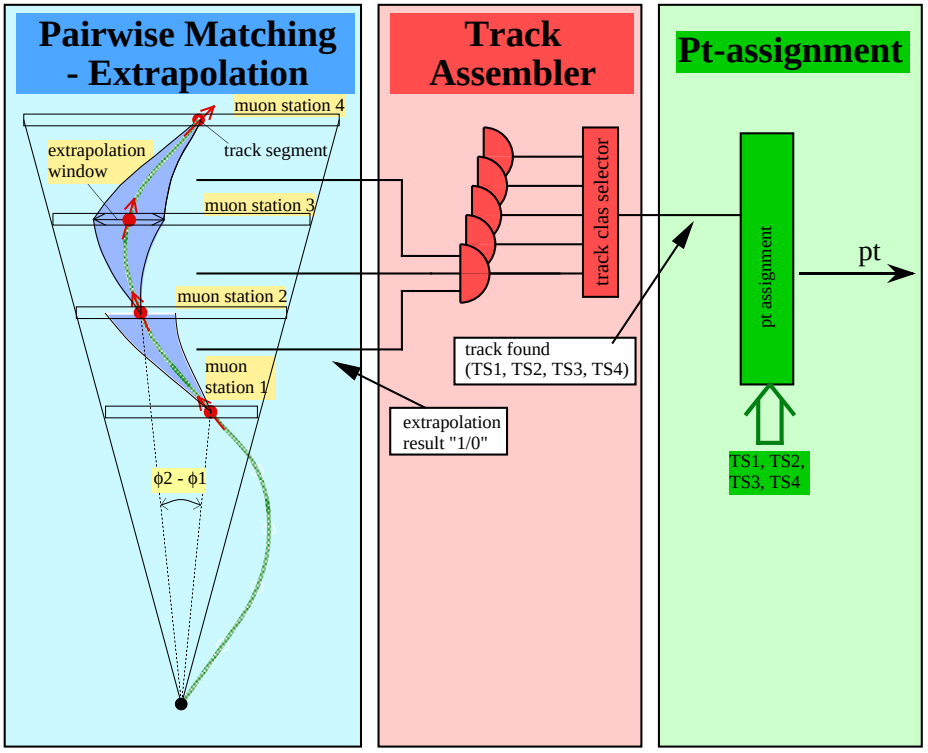


Fig. 5.2.: Principle of the track finder algorithm (3-Step scheme).

steps. In the first step track segments from different chambers are matched to track segment pairs by extrapolation. Later the track assembler links track segment pairs to form complete tracks and forwards the track segment data of the two highest ranking tracks to the assignment units where transverse momentum p_t and location of the tracks are determined.

In fig. 5.3 a simplified block diagram of a sector processor is displayed. The sector

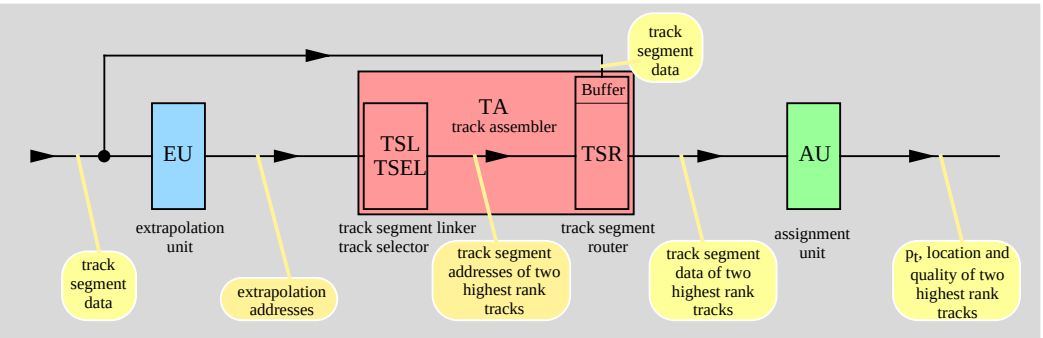


Fig. 5.3.: Block diagram of a sector processor.

processor is divided into three parts - the extrapolator (EU), the track assembler (TA), and the p_t-, η- and ϕ-assignment units (AU).

The extrapolation unit EU attempts to match track segment pairs of distinct stations using the extrapolation criteria described earlier. When track segment pairs meet these criteria the information is forwarded to the track assembler TA. All extrapolations between all station pairs are carried out in parallel.

Since tracks may cross detector segment boundaries the information of the extrapolation units of the neighbouring detector segments are also routed to the track assembler TA. It evaluates all extrapolation results in order to find up to two tracks with the innermost track segment in its own detector segment (TSL, TSEL). It forwards the relative track segment addresses of the track segments of found tracks to the track segment router TSR. During the execution of the track assembler algorithm the track segment data is stored in a buffer memory. The relative addresses are used by the track segment router TSR to extract the corresponding track segment data out of the buffer memory.

The track segment data are forwarded to the p_t-, η- and ϕ-assignment units (PAU, ηAU, ϕAU).

Short description of the track finder algorithm

In the following a short overview of the track finder algorithm is given. The various

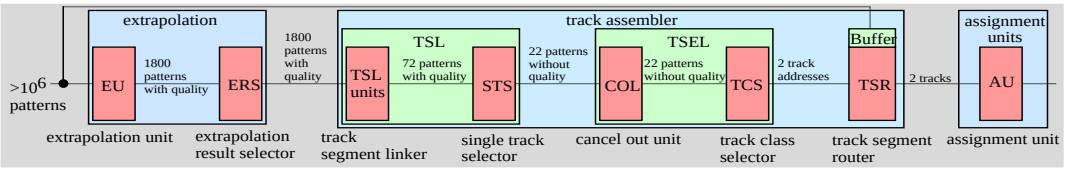


Fig. 5.4.: Detailed block diagram of a sector processor.

processing units are described in detail later. Fig. 5.4 shows a more detailed block diagram. Fig. 5.5 illustrates how the algorithm reduces the number of possible track candidates from more than 10^6 to two.

The extrapolation units (EU) match track segment pairs to each other. As a result the information *er* which track segments belong to each other and the quality *eq* of the matched track segments is given out. In total more than 1800 possibilities exist to assemble valid tracks (fig. 5.5 b, c).

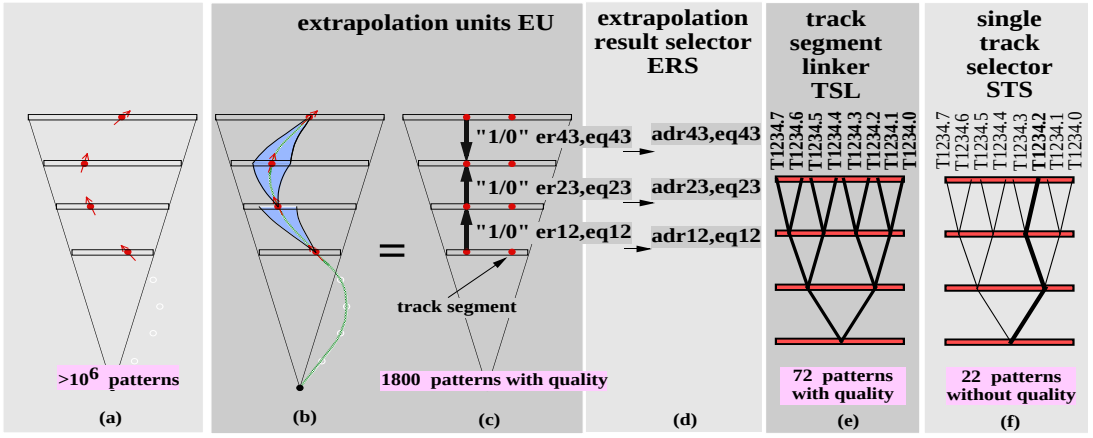


Fig. 5.5.: Reduction of possible track candidates by the track finder algorithm.

The extrapolation result selector (ERS) selects the two best extrapolations for each source track segment. It outputs the relative address *adr* of the track segments as well as the extrapolation quality *eq* (fig. 5.5 d).

The track segment linker (TSL) attempts to link track segments together starting from the innermost track segment. As the extrapolation result selector (ERS) delivers up to two extrapolation addresses per source track segment more than one track candidate may be found originating from the innermost track segment (fig. 5.5 e). In order to cope with inefficiencies of the chamber system a given number of track segment linker modules start in stations other than station one. In general the track segment linking scheme reduces the number of track candidates from 1800 to 72. Still a quality information remains attached to each track candidate.

For each innermost source track segment the single track selector (STS) retains only the track candidate with the highest extrapolation quality. A total of 22 track candidates survive (fig. 5.5 f).

The cancel out units (COL) validate track patterns which are part of longer track patterns. An example is a track consisting of track segments in station one and two which are found to be equivalent to a track containing segments from all four stations.

The track selector (TSEL) selects the two highest ranking tracks out of the remaining 22 track candidates and forwards the relative addresses of the matched track segments.

The track segment router (TSR) uses these relative addresses mentioned above to output the corresponding track segment data. It extracts the track segment data from the buffer memory.

The assignment units (AU) use the track segment data of up to two tracks to determine the track properties.

In the following chapters all elements of the track finder processor are described in full detail.

5.2.1. Extrapolation unit (EU)

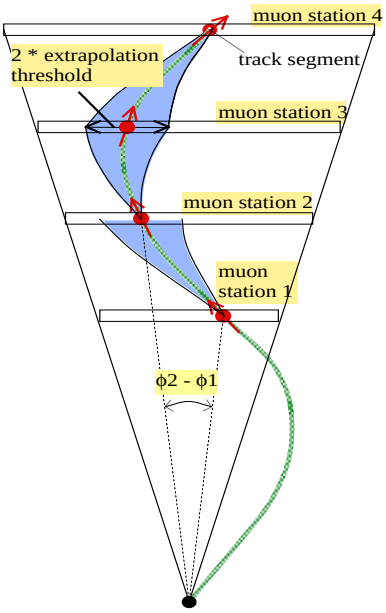


Fig. 5.6.: Pairwise matching of track segments by extrapolation.

The extrapolation unit attempts to join a track segment with track segments of other stations. The matching of two track segments will be carried out by extrapolating one of the track segments to the station of the second track segment. If the difference between the extrapolated position value and the measured value of the second track segment is below a specific threshold the extrapolation is considered successful (see fig. 5.6). The extrapolation unit assigns to each possible track segment pair an extrapolation result and an extrapolation quality.

Due to the bending a muon can cross detector segment boundaries in ϕ-direction. Thus when joining two track segments it is necessary to involve the track segments of the neighbouring ϕ-sectors. Fig. 4.2 on page 39 shows the maximum deflection for strongly bent low p_t muons to be below the dimensions of two detector segments ($2 \cdot 0.52$ rad or $2 \cdot 30°$). This means a muon is very unlikely to cross two detector segment boundaries.

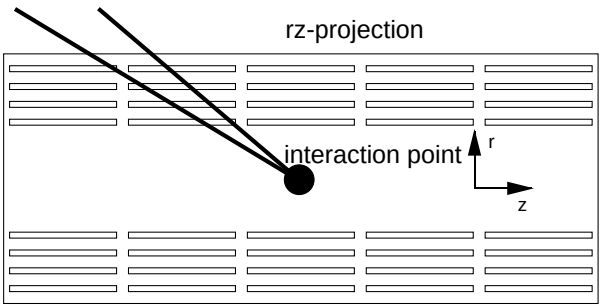


Fig. 5.7.: As seen in the rz-view muons can only traverse two wheels.

The non-projective geometry of the chambers with respect to the muon tracks in the rz-view requires to examine also the neighbouring wheels. Fig. 5.7 illustrates that a muon originating from the interaction point does not cross more than one wheel boundary. The deviation from a linear track in the rz-plane is small. The track can be approximated by a straight line. Thus when joining track segments it is sufficient to look into the corresponding ηϕ-segment and its directly adjacent neighbours. As the muon track in the rz-projection is almost a straight line the particles will not change their flight direction with respect to the z-direction of the detector. Checking the wheel contrary to the flight direction is not necessary. In all wheels but the central wheel the flight direction of the muons in z-direction obviously is outgoing. Thus in all wheels but the central wheel the sector processor algorithm examines six ηϕ-segments (fig. 5.8). When extrapolating from the centre of the detector (wheel 0) the target track segments of the nine adjacent detector segments have to be evaluated.

In every station the trigger primitive generator delivers up to two track segments. The extrapolation unit has to attempt to match each source track segment to twelve (18 when considering the centre wheel, respectively) track segments of the next station (fig. 5.8). **From now on the sector processor of an outer wheel is described. For the sector processor in the centre of the detector (wheel 0) the number of target track segments and thus the number of extrapolations has to be increased accordingly.**

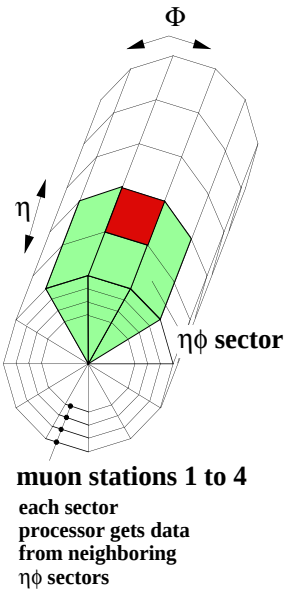


Fig. 5.8.: At least five neighbouring detector segments must be evaluated.

Simulation of trigger acceptance requires to accept tracks with at least two out of four track segments (see chapter 4.3.3.). For the matching process six station pairings are necessary; 1-2, 1-3, 1-4, 2-3, 2-4 and 4-3. (Remember that the extrapolation from station three is not possible. The extrapolation is carried out the other way round.) In total twelve source track segments exist per detector sector. The sector processor incorporates twelve separate extrapolation units. Each extrapolation unit extrapolates one source track segment to another station and compares the extrapolated value to twelve target track segments. As a consequence 144 comparisons are carried out in the processor.

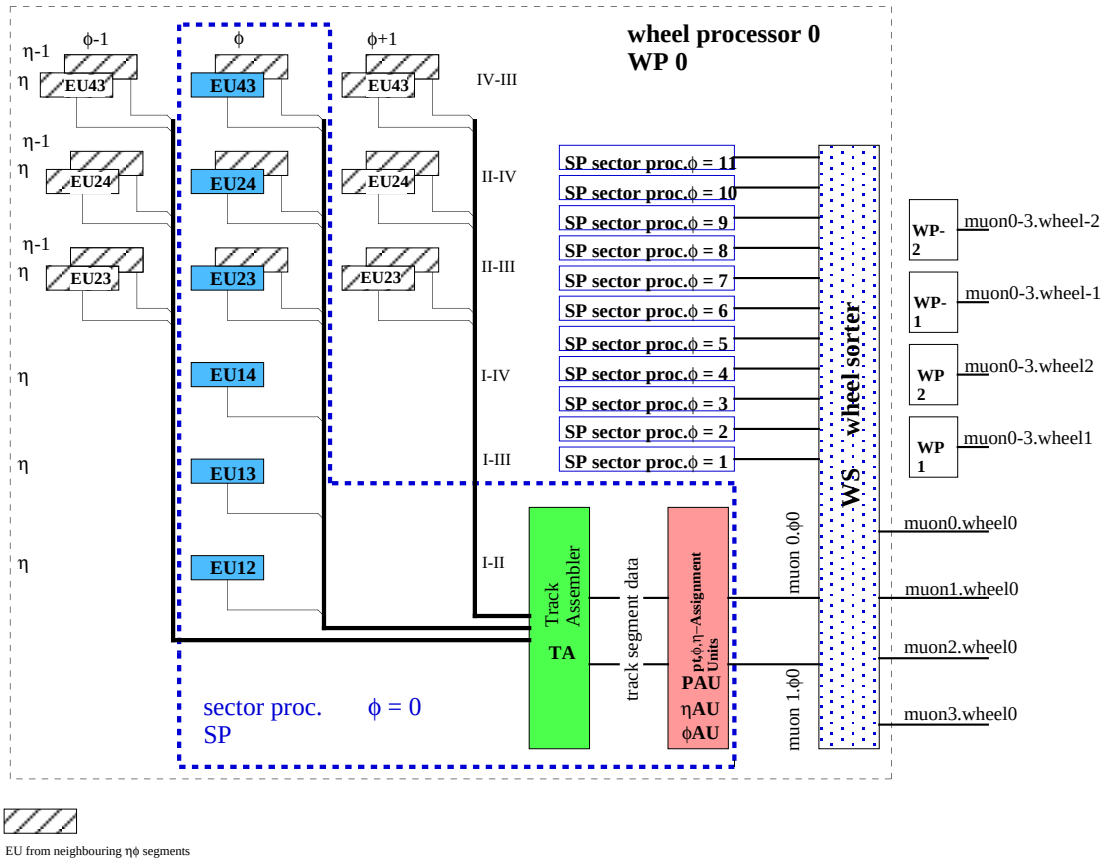


Fig. 5.9.: Block diagram of the entire track finder processor system.

In order to process tracks crossing detector segment boundaries the information of neighbouring extrapolation systems must be made available to the sector processor. The output of the neighbouring extrapolation units EU23, EU24 and EU43 from the five neighbouring detector segments are routed to the sector processor.

A sector processor searches for tracks with the first track segment within its detector segment. The output of the neighbouring extrapolation units EU12, EU13 and EU14 is not needed. The block diagram of the entire track finder system is illustrated in fig. 5.9.

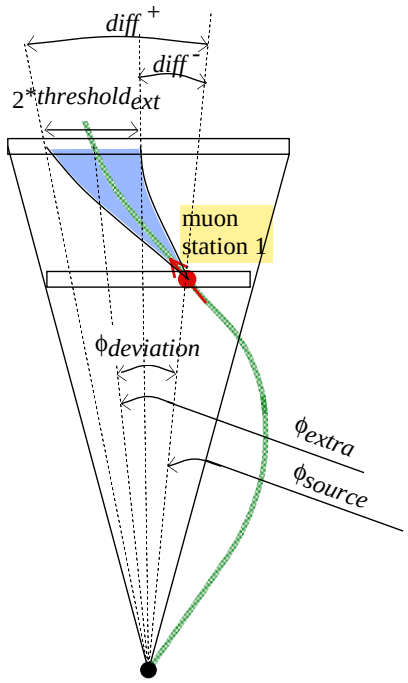


Fig. 5.10.: Extrapolation from station one to station two.

Using the extrapolation method allows to calculate the medium deviation $\phi_{deviation}$ (difference between extrapolated hit position ϕ_{extra} and source hit position ϕ_{source}, see fig. 5.10) and an extrapolation threshold $threshold_{ext}$. The extrapolated hit position ϕ_{extra} is compared to the hit position of each possible target track segment ϕ_{target}. If the difference is found to be below the extrapolation threshold $threshold_{ext}$ the extrapolation is considered successful. The extrapolation result bit is set. An extrapolation quality word is assigned to each successful extrapolation; it is derived from the track segment qualities given by the drift tube trigger primitive generator.

The hardware implementation of the extrapolation using the medium deflection $\phi_{deviation}$ and the extrapolation threshold $threshold_{ext}$ proves cumbersome because the absolute value of the difference of the extrapolated hit position ϕ_{extra} and the actual hit position ϕ_{target} needs to be calculated. In addition two arithmetic operations ($\left|\phi_{source} + \phi_{deviation} - \phi_{target}\right| < threshold_{ext}$) have to be conducted before the actual threshold comparison. These operations are quite time consuming. A window comparator is more efficient. It checks for the difference *diff* between hit position in the source chamber ϕ_{source} and the hit position in the target chamber ϕ_{target} to be within a lower and an upper limit, $diff^+$ and $diff^-$, which greatly reduces the number of calculation steps (see fig. 5.10).

It is too time consuming to obtain the extrapolation limits by means of digital arithmetic logic units. The values $diff^+$ and $diff^-$ are therefore predetermined and stored in memory based lookup tables (EXT). The bend angle ϕ_b of the source track segment is used to address two RAM-based lookup tables which output the upper and the lower limit ($diff^+$ and $diff^-$) of the difference of the position values of the two track segments in question. At the same time the actual difference *diff* between the hit position in the source chamber ϕ_{source} and the hit position in the target chamber ϕ_{target} is calculated. A window comparator checks if the difference *diff* is found to be within the given limits ($diff^+$ and $diff^-$). In case of a successful extrapolation the extrapolation result bit *er* extrapolation quality word *eq* is assigned. In the FPGA prototype, described in chapter 6, the extrapolation quality is a one bit word; it is set to '1' if both track segments of the matched pair have the quality status 'correlated' (see chapter 2.3.3.). For the final

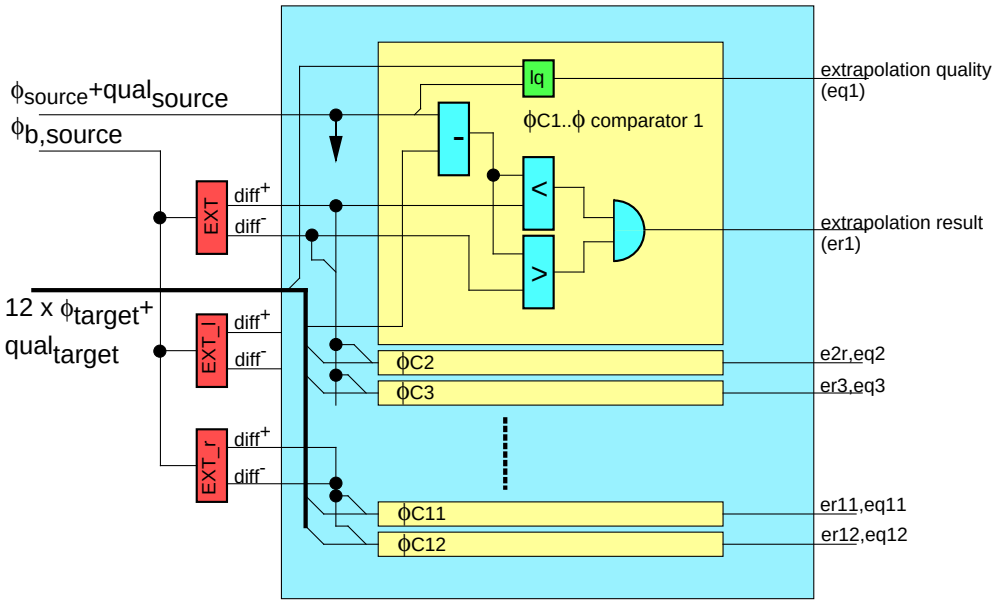


Fig. 5.11.: Extrapolation unit extrapolates from one track segment and compares the extrapolated value to twelve target track segments. It outputs the extrapolation result (er) and the extrapolation quality (eq) for each comparison.

implementation further simulations have to show whether this single quality bit is sufficient. Each extrapolation unit outputs the extrapolation results and the extrapolation quality bits for each of the twelve target track segments (see fig. 5.11).

A problem arises when subtracting the hit coordinates ϕ_{source} and ϕ_{target} of track segments of different ϕ-segments. The origins of coordinates are different and thus the difference *diff* is shifted by the chamber dimension in ϕ (0.57 rad). The obvious solution were to add or subtract this offset from the target track segment position ϕ_{target}. The disadvantage is the additional calculation delay. In order to avoid this delay the offset compensation is carried out already during the calculation of the extrapolation limits *diff+* and *diff-*. Accordingly two further extrapolation lookup tables must be introduced; one is used for the extrapolations to ϕ-sectors on the right hand side (EXT_r) and the other one for the left hand side (EXT_l). The lookup tables EXT_l and EXT_r deliver the offset-compensated extrapolation limits $diff\_l^+$, $diff\_l^-$, $diff_r^+$and $diff\_r^-$(fig. 5.11).

5.2.1.1. Extrapolation result selector (ERS)

The extrapolation result selector selects the two best extrapolations per source track segment and outputs the relative address of the target track segment.

The task of the extrapolation result selector is to reduce the data stream from the extrapolation units to the track assembler. The probability to find more than two successful extrapolations originating from one single track segment is negligibly small. However, the extrapolation units deliver the extrapolation result and the extrapolation quality for each of the twelve possible track segment pairs. The extrapolation result selector simplifies track segment assembly (see chapter 5.2.2.1. for explanation).

The selection criteria are the extrapolation qualities and the relative location of the target track segment with respect to the source track segment. Firstly the extrapolation

result selector evaluates the extrapolation qualities; secondly the target track segment is checked for its origin which may be either the same detector segment or a neighbouring segment.

For each of the two target track segments the extrapolation result selector ERS outputs:

- Relative address of the target track segment. In case the extrapolation was unsuccessful a certain code is given out.
- Extrapolation quality. The extrapolation quality the selection was based on.

The hardware implementation uses a set of priority encoders.

5.2.2. Track assembler (TA)

The task of the track assembler is to find the two tracks in a detector segment exhibiting the highest number of matching track segments and the highest extrapolation quality. It outputs the track segment measurements belonging to these tracks. The track assembler TA consists of the track segment linker TSL, the track selector TSEL and the track segment router TSR.

The task of track segment linker is to link the track segment pairs formed by the extrapolation units to full tracks. It forwards all track candidates to the track selector which selects the two highest ranking tracks and gives out the relative addresses of the matching track segments. The track segment router extracts the corresponding track segment data from the data pipeline using the relative addresses of the track segments.

5.2.2.1. Track segment linker (TSL)

To solve this problem again a pattern matching method may be considered. The output of the extrapolations units (extrapolation result and quality) can be used. A valid track consisting of *n* track segments is identified by a pattern of *n-1* extrapolation result bits. As illustrated in fig. 5.2 this task can be performed using simple gate array logic. Each valid track segment combination is recognized by its dedicated AND gate. A priority encoder (pattern selector) retains only the two highest ranking patterns and gives out their code. This code unambiguously identifies the track segments contained by the found tracks. Fig. 5.12 shows examples of valid track segment pair patterns. In total there are about 1800 valid patterns. On one hand this relatively high number of valid patterns is due to particles crossing detector segment boundaries. On the other hand this high number arises due to possible tracks consisting of less than four track segments. Even a single track segment pair has to be accepted as a full track. There are eleven valid track classes overall: T1234, T123, T124, T134, T234, T12, T13, T14, T23, T24 and T34 (the digits denote the stations belonging to the track).

A track consisting of four matching track segments, T1234, is assembled by three matching track segment pairs (1-2/2-3/3-4). However, both the combination of track segment pairs ‘1-2/2-3’ and ‘2-3/3-4’ each form an additional valid track (T123 and T234) consisting of three matching track segments. Furthermore even each single track segment pair (1-2,2-3,3-4) forms more valid track segment pair patterns. This is

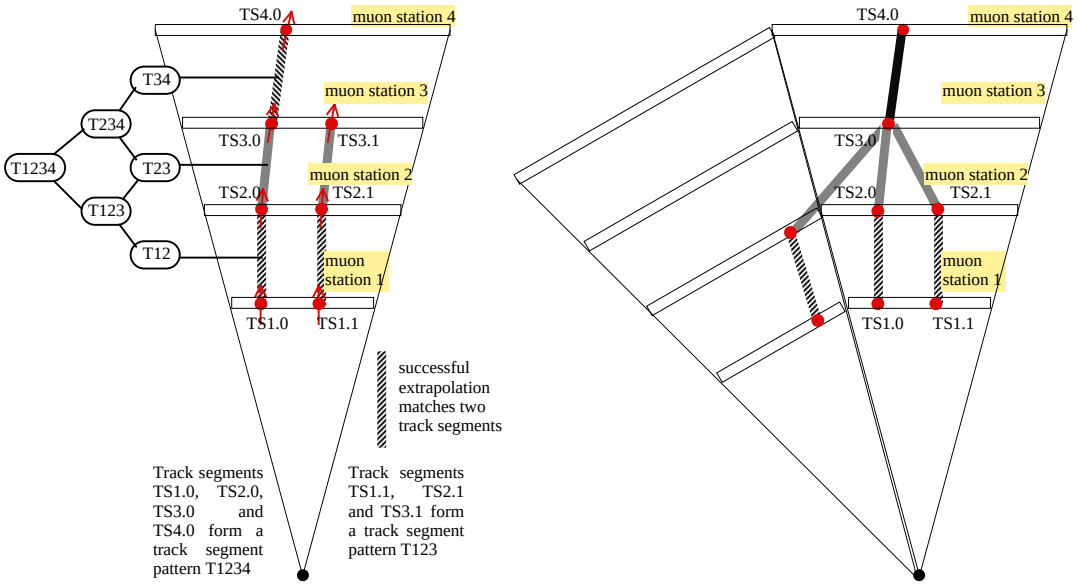


Fig. 5.12.: Example of track linker patterns.

illustrated in fig. 5.12. These sub patterns must be suppressed in case of a complete track T1234. Basically the recognition of 1800 patterns of a length of up to four bits is easily configured by hardware. Problems arise when the sub patterns need to be suppressed. Every single ‘short track’ (consisting of only two or one track segment pairs) is a sub pattern of a vast number of longer tracks (see fig. 5.12). This fact raises a severe interconnection and integration problem within the track segment linker. In order to find tracks overlapping across detector segment boundaries the extrapolation results of the detector segment neighbours also have to be evaluated. Sub patterns have to be cancelled out even by tracks originating from a different detector segment.

As discussed later the extrapolation qualities must be taken into consideration in order to find the best track. This problem is covered in detail in chapter 5.2.2.2. It is not sufficient to recognize the track pattern and to mark its appearance. Furthermore the attached quality information must be provided to the pattern selector. The latter must select the highest ranking pattern with respect to the number of involved track segments and with respect to extrapolation quality. This process is time consuming and complicated because the pattern selector would have to compare quality words of 1800 patterns to each other.

Considering these requirements the hardware implementation using a pattern matching method for the track segment linker proves impossible. Each short pattern might be vetoed by a longer track. Simple four fold AND gates to recognize full patterns are by far not sufficient.

Prior to extrapolating more than 10^6 patterns would have to be recognized and their properties given out. Applying the extrapolation method reduced the number of patterns to some 1800. Even at this stage a pattern comparison method in the track segment linker is not feasible. A further simplification step is necessary. Since the probability to find more than two successful extrapolations from a given source track segment is very low it is possible to reduce the maximum number of successful extrapolation results given out by the extrapolation units to two. A further processing unit, the extrapolation result selector ERS, is introduced between extrapolation unit and track segment linker (fig. 5.4, see chapter 5.2.1.1.) for further data reduction.

Dynamic track segment linking with indirect addressing

The track segment linker comprises the track segment linker units and the single track selectors. Each of the track segment linker units is responsible to find tracks of a certain track class originating from one track segment. The track segment linker units forward their track candidates to the single track selector. There is one single track selector for each track segment linker unit. The single track selectors retain only the track candidate with the highest extrapolation quality. The linking result and the relative track segment addresses of all involved track segments for each track class and source track segment are given out.

In the track segment linker each event is evaluated if the pattern is a member of the valid track segment pair pattern space. Since the scheme works comparable to the indirect addressing in a microprocessor and links the track segment pairs without comparing it to a predefined set of patterns it is called ‘dynamic track segment linking with indirect addressing’. The basic principle is to assemble all tracks from their innermost starting points in a serial way.

The extrapolation result selector delivers the addresses of the target track segments of previously formed track segment pairs. In order to append a second track segment pair to form a track consisting of three track segments one has to look if an extrapolation from the target track segment of the first track segment pair to another track segment was successful. One looks at the output of the extrapolation result selector of the first track segment pair. The obtained extrapolation address denotes the source track segment for the track segment pair to append. If the extrapolation result selector of this source track segment gives a valid target segment address, this address indicates the address of the linked track segment. A track consisting of three track segments is formed. The scheme is repeated once more in order to form tracks comprising four track segments.

One track segment linker unit attempts to find a track candidate for one track class originating from one single track segment. There are eleven track classes. Two track segments in each station are delivered by the drift tube trigger primitive generator. As a consequence ($2 \cdot 11 =$) 22 track segment linker units are being applied and form one track segment linker.

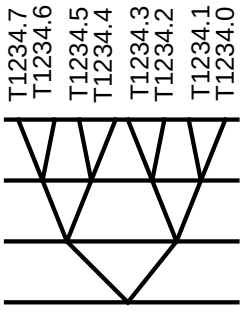


Fig. 5.13.: Eight track candidates of the track class T1234 emerge from a single track segment.

Every extrapolation result selector ERS delivers up to two addresses. The theoretically possible number of track branches originating from the same track segment is 2^{n-1}. n is the number of involved track segments (see fig. 5.13). The sum of all valid track branches yields 72. For every possible track branch all track segment addresses are given out. The linking result, indicating the presence of a track candidate, can be derived from the track segment addresses of the outermost station. If it equals the code for a valid track segment, this track candidate has been “found”.

The result of this scheme is a priority ordered bit stream of 72 bits. The dynamic track linking scheme reduces the number of patterns from about 1800 to 72 by a factor of 25 without sacrificing measurement accuracy.

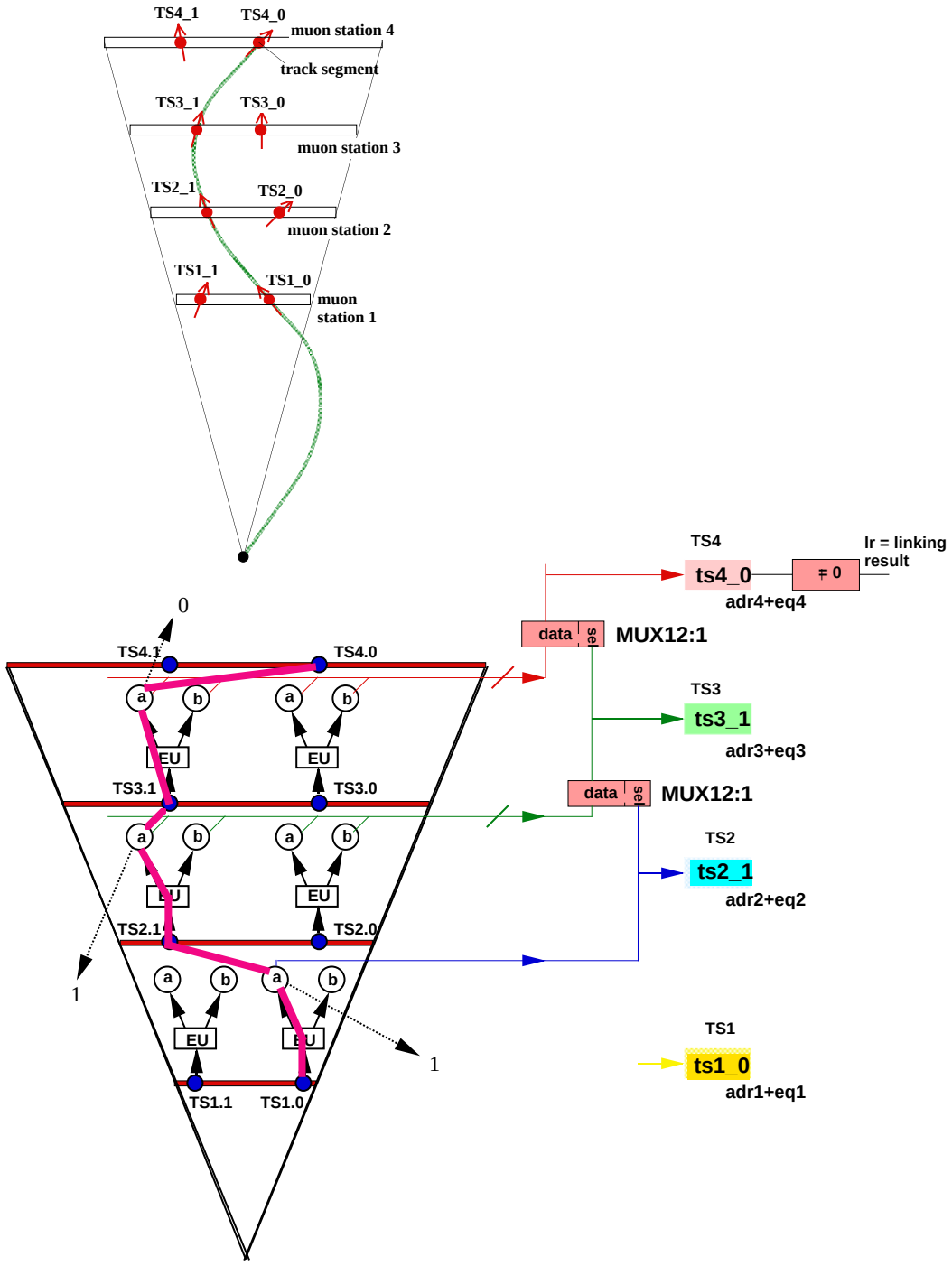


Fig. 5.14.:Principle of track segment linker operation.

An example is given in fig. 5.14. It assumes a track to be comprised by track segment 0 in station one, track segment 1 in station two and three and track segment 0 in station four. As a consequence the extrapolation units and extrapolation result selectors of the extrapolation from track segment 0 in station one to station two give the address of track segment 1 (in station 2). The extrapolation units of track segment 1 in station 2 give out the address for track segment 1 in station three and so on.

As illustrated in the figure the extrapolation addresses *a* guide the track linker unit from the innermost track segment to the outermost.

The hardware implementation employs multiplexers (see fig. 5.14). The extrapolation addresses of the first track segment pair are connected to the select input of a multiplexer. All target addresses of extrapolations of the next station pair are routed to the data input. The address on the select inputs selects the according target track segment address among twelve. Thus a twelve to one multiplexer is employed. The output of the multiplexer is the address of the appended track segment. This principle resembles the indirect addressing in a microprocessor. The advantage of this method lies in the speed of multiplexers implemented in hardware. Only a small number of logic gates is required. Although the scheme proceeds in a serial way (with up to two multiplexer stages in a row) it is much faster than a pattern comparison approach. An important issue for the inevitable error monitoring is the clear and simple concept of the scheme.

A detailed hardware description may be found in appendix B.

5.2.2.2. Single track selector (STS)

The single track selector retains only one track originating from one source track segment by selecting the track with the highest extrapolation quality.

For two reasons only one track originating from the same track segment is tolerated. Firstly multiple hits in the drift chamber caused by δ-rays will trigger several valid track candidates (see fig. 5.15). Secondly it is time consuming to select the two best patterns amongst 72 if the extrapolation quality has to be included in the selection process.

However, it is very unlikely to find a track segment measurement caused by two different muons but with exactly the same location. Thus the ‘single track selection’ STS may safely be applied. The single track selector retains one track branch per source track segment.

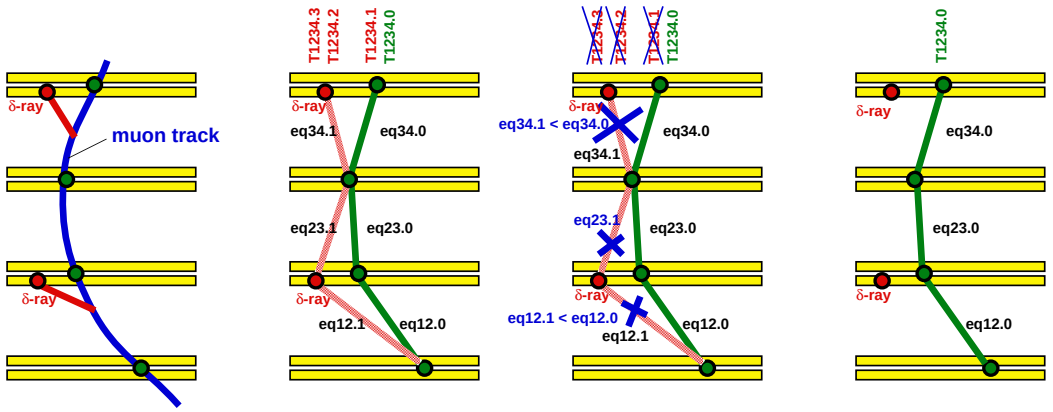


Fig. 5.15.: Principle of single track selection operation.

Fig. 5.15 illustrates the principle of the single track selection. δ-rays triggered track segment measurements in station two and four. Four track candidates are being provided by the track segment linker units. The single track selector cancels branches with lower extrapolation quality *eq* and retains only the track consisting of track segments with the highest quality attached to.

Simulation showed that in most cases track segments caused by δ-rays have an uncorrelated track segment quality status [22]. This is due to the limited range of the electrons in matter. Hence track segment pairs with a track segment caused by δ-rays are likely to have low extrapolation quality.

Starting from the outermost branching point (in case of the tracks T1234 this is the extrapolation between station 3 and 4) the branch with the lower extrapolation quality is cancelled. In case of equality the branch with the higher index *b* (T1234.b) is removed. After selecting the highest ranking branch 34 the single track selection is applied again to the branches 23 and 12.

The number of possible tracks is now reduced to 22. Only one track per track origin is given out. Applying the single track selection reduces the number of valid patterns from 72 to 22.

The hardware implementation employs multiplexers and comparators (see fig. 5.16). The extrapolation qualities of the branches are compared. The comparison result controls the multiplexer. It routes the address and the linking results of the remaining branch through the multiplexer.

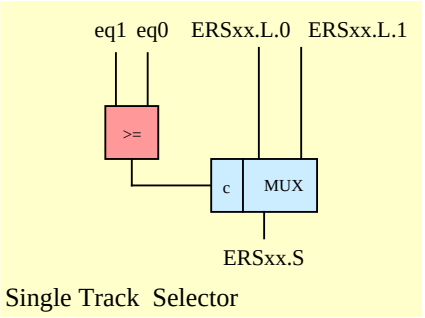


Fig. 5.16.: Block diagram of a single track selector.

All remaining linking results - a 22 bit stream with each bit indicating if the corresponding track segment combination was found - and the track segment addresses are forwarded to the track selector. However, there is no quality information attached to the track candidates anymore. The single track selector already includes the quality information into its decision.

Simulation showed that for up to 75 % of muon tracks at least one station delivers a second track segment. Assembling the tracks without the single track selection scheme would cause the track finder processor to output a second track in all these cases. VHDL simulation of the processor revealed that application of the single track selector decreased the fraction of double tracks given out to only 4 %. However, this reduction is performed without any loss of real tracks. Certainly algorithms may be found which reduce the number of double tracks further but this might be at the cost of efficiency when two real tracks close to each other need to be treated.

5.2.2.3. Track selector (TSEL)

The track selector retains only the two highest ranking track candidates amongst the 22 provided. It consists of the cancellation units and the track class selector.

The previous stage, the track segment linker, provides all possible track candidates. However, some of the track candidates may be entirely part of a longer track candidate. For instance this is the case if a track containing track segments from all four stations, T1234, is being found. In addition tracks containing some of the same track segments (like T123, T234, T12,...) will be found. The track with the highest number of track

segment is referred to as mother pattern while the others are called sub patterns. The cancellation unit suppresses the sub pattern in presence of their mother patterns and removes tracks of different track class originating from one track segment. The inputs are the 22 linking result bits from the track segment linker. The linking result bit of a certain track class is masked off if this track is part of a longer track. The cancellation unit outputs the masked 22 linking result bits to the track class selector, which selects the two highest ranking track candidates and forwards the relative address of the track segments.

Cancellation units (COL)

The basic principle of the cancellation scheme is to alter only the linking result bits in presence of a mother pattern. The track segment addresses remain unchanged. The hardware effort is minimized. The cancellation units are divided into three groups: cancellation sub pattern, cancellation single track and cancellation down.

Cancellation sub pattern checks the presence of a mother pattern. Cancellation single track attempts to distinguish between multiple tracks caused by δ-rays or by tracks close together and removes tracks caused by δ-rays. Cancellation down finds double tracks caused by double hits in the innermost chamber.

The hardware implementation employs simple logic gates and comparators. A more detailed description is given in appendix B.

Track class selector (TCS)

The task of the track class selector is to find the highest ranking track amongst all track candidates. Applying the extrapolation method, the dynamic linking of track segment pairs and the single track selection reduces the number of track candidates to 22. No quality information must be taken into account anymore at this stage. Thus a simple priority encoder may be used. The linking result bits are connected to the data input in the order of their rank. The first criterion is the number of track segments the track consists of. The second criterion prefers tracks with track segment of stations one and two, because the momentum can be measured with a higher precision in station one and two [22]. The track class selector puts out the relative addresses of the track segments of up to two tracks.

5.2.2.4. Track segment router (TSR)

The track selector delivers the relative addresses of matching track segments. The track segment router uses the track segment addresses to select the corresponding track segment data. During the processing time of the track assembler the data is stored in a buffer memory. Shift registers and multiplexers are used.

5.2.3. Assignment units (AU)

Once the track segment data is available to the assignment units, memory based look up tables are used to determine the transverse momenta of the particles. The momentum and the location of the tracks as well as a quality information about the track finding is given out.

5.2.3.1. P_t-assignment unit (PAU)

As mentioned in chapter 4.3.4. the resolution for determining p_t is dependent on the track segment quality. A control circuit evaluates the various track segment qualities and determines the p_t measurement algorithm [22]. In case transverse momentum p_t is measured using two spatial coordinates, the difference between the position values is sent to the memory based p_t assignment look up tables. If p_t is assigned using only an angle of a track segment this angle is routed to the same look up table. In both cases the information, namely which type of p_t measurement was chosen, is made available to the lookup tables. p_t is measured with a resolution of 5 bit, one additional bit indicates the charge of the particle.

5.2.3.2. ϕ-assignment unit (ϕAU)

The high ϕ-resolution of the track segments allows to output a ϕ-measurement of the track very easily. The ϕ-value of the innermost track segment of the track is given out with a reduced resolution of 8 bits for the whole 2π range. This is equivalent to a binning of 1.4° or 25 mrad.

5.2.3.3. η-assignment unit (ηAU)

The non-projective chamber geometry with respect to the rz-view does not allow for

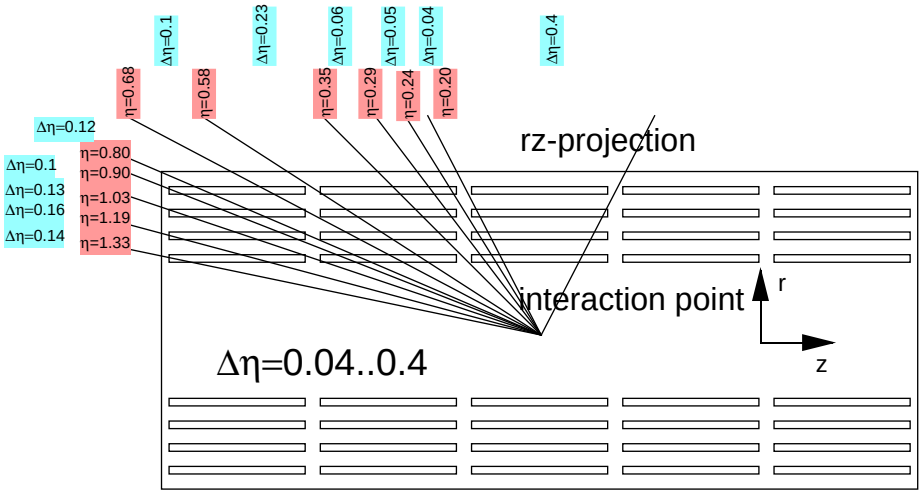


Fig. 5.17.: rz-view of the drift tube chamber system.

a precise measurement of the η-coordinate of the track. As illustrated in fig. 5.17 the only information the η-coordinate can be derived from is the place where a particle crossed detector wheel boundaries. A coarse, location dependent resolution in η can be achieved. Moreover the resolution is dependent on the number of track segments a track consists of. The η-measurement uses a 2 bit code which indicates between which stations the muon crossed the wheel boundaries.

5.2.3.4. Quality assignment unit (qAU)

A quality code for each track is given out. One bit indicates if the transverse momentum has been assigned using the difference of two spatial measurements or by using one track segment angle only. The p_t measurement using two spatial coordinates yields a measurement with a better resolution. Further two bits are reserved to output the number of track segments involved in the track. Since a valid track may consist of two track segments only the number of track segments a track contains represents a quality measure.

5.2.4. Wheel sorter (WS)

The task of the wheel sorter is to select the four muons with the highest p_t in a wheel amongst the 24 muons from the twelve sector processors. This is done using a dedicated ASIC-sorter chip [94]. The latency for sorting four muons amongst 24 is 150 ns or 6 bunch crossing (bx). One bunch crossing is reserved for resynchronisation.

5.3. Conclusion: Processor architecture and simulation

The architecture was simulated using a behavioural VHDL simulation. The simulation was used to prove the functionality of the algorithm and to optimize the hardware partition.

The processor uses only simple logic blocks as multiplexers, comparators and subtractors. Therefore the architecture proves to be simple and easy to survey.

The efficiency curves show the performance of the entire track finder processor system (fig. 5.18) for p_t-thresholds p_t = 20, 40, and 50 GeV/c. In high energy physics experiments muon triggers are set to conditions that permit finding a muon with at least a certain momentum. Thus the efficiency curve is a quality measure of the trigger. It gives the relative number of particles identified with a momentum higher or equal than the threshold over the actual momentum. The steeper the curve the sharper is the momentum cut.

The efficiency reaches at least 95 %. Not all muons are being identified because of geometrical inefficiency of the detector and inefficiency of the chamber system [16].

The obvious bottleneck of the track finder processor is the limitation to be able to process only up to two successful extrapolations per source track segment (extrapolation result selector). However, as VHDL simulations showed, in only 2 % of the double muon events more than two successful extrapolations occured. Again they were caused by δ-rays. The extrapolation result selector is optimized in a way to select the track segment pairs with the highest quality. The extrapolation result selector

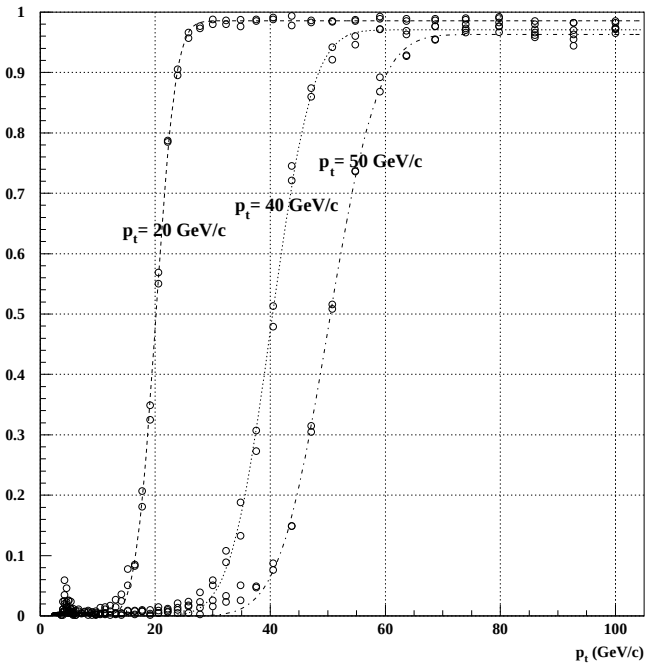


Fig. 5.18.: Efficiency curves for p_t = 20, 40 and 50 GeV/c.

discards the pairs caused by δ-rays. In this way the extrapolation result selector does not influence system performance.

6 FPGA prototype layout and test

In this chapter the design of an FPGA-prototype of the sector processor is described. 19 FPGA circuits (XILINX 4000) and 19 RAM based lookup tables are employed in one sector processor. The main limitation were the relatively low I/O pin count of the available packages. The design of the prototype showed clearly that it is difficult to cope with the amount of bits to be processed. The interconnection between processing units poses a problem and causes additional delay. However, the necessary gate array logic can be placed easily in the programmable units.

6.1. Implementation strategy

One important key point was to prove that the VHDL model of the processor can implemented in hardware in a reasonable way.

The main reason for the prototype was to build a hardware system capable of the same logical functionality as an ASIC based system. XILINX-FPGAs [43] were used. Restrictions had to be accepted with respect to the clock frequency as well as the number of processing steps.

The relatively low I/O pin count turned out to be the main limitation of the available FPGA packages. Data transfer between the FPGA-circuits must be time multiplexed. As stated in earlier chapters today's field programmable gate arrays do not allow the implementation of synchronously working digital systems with clock frequencies of more than 50 MHz. Double allocation of input pins would limit the system clock to 25 MHz. For the prototype test a system clock of 20 MHz was chosen.

The sector processor board was designed in VME-9U standard. All lookup tables and FPGAs are loaded via the VME bus interface. This solution ensures maximum flexibility. Even a change of the logic configuration of the programmable units remains possible. Parts of the system can be altered or switched off without changing the hardware of the processor.

6.2. Block diagram of implementation

In fig. 6.1 the block diagram of the processor is shown. The system partition is illustrated by the boxes. 19 XILINX4000-type field programmable gate arrays (FPGA) were employed to house the necessary logic. The theoretically available gate count of all 19 FPGAs yields 240000 gates or some 10000 cellular logic blocks. However, only 60 to 80% of the total logic resources (cellular logic blocks - CLB) of the FPGA were occupied by the track finder logic.

The task of system partitioning had to be done by hand. For programming the FPGA automated translation tools were employed. The VHDL-code was translated directly to the gate array level. However, it has to be stated that this proceeding was only possible for non time critical functions. Time critical functions were translated manually to the gate level.

The total execution time is 29 clock cycles. However, as mentioned earlier, a large fraction of the processing steps is consumed by the transmission between the integrated circuits. At least one clock cycle is lost for each transmission.The number of processing steps without transmission time is only 20 cycles. This is the number of clock cycles for an operating frequency of 20 MHz. In order to achieve this frequency, additional pipeline steps (flip-flop stages) had to be introduced. This was not due to slow processing of the logic units but merely due to routing delays between the units. Obviously one can achieve a smaller number of processing steps by reducing the operating frequency and removing a given number of flip-flop stages.

The design of the prototype board showed clearly that FPGAs of today are not suited to the track finder requirements. Both the I/O count of the packages and the data propagation do not allow the final design of the track finder system employing today’s FPGA.

In spite of that the functionality of the implementation of the track finder, the momentum measurement algorithm and the hardware structure mapping of the detector geometry onto the hardware level was proven to be adequate to the system requirements. Moreover it could be shown clearly that the VHDL-model can be implemented in hardware with a minor extent.

6.3. Test configuration

Each sector processor must communicate with its adjacent neighbours. However, only one track finder processor prototype is available. Neighbour processors are simulated logically using their VHDL-representation. In fig. 6.2 the test setup is shown. The hardware prototype receives test patterns from the test bench. Simulated chamber data as well as some simulated neighbour processor data are loaded at the same time.

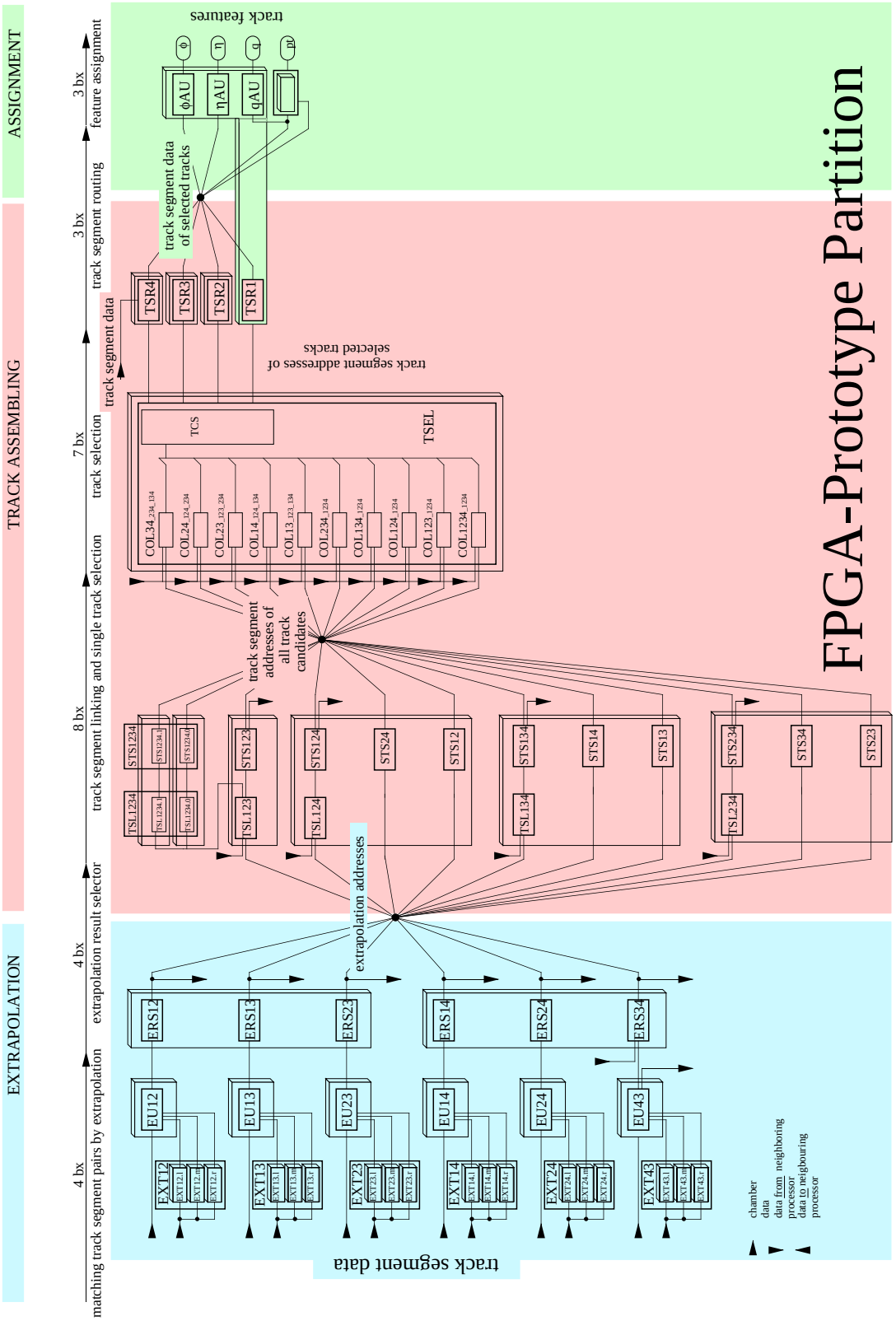


Fig. 6.1.: Block diagram of a sector processor. The shaded boxes represent the processing steps (extrapolation, track assembling and assignment units). The small boxes stand for a physical unit (FPGA and RAM).,

For test pattern procurement a two-step simulation is run. Firstly detector data simulated by generating physics events [22] is processed using the VHDL model. Running the VHDL-simulation provides two more datasets. The FPGA prototype test requires all three datasets, namely input pattern data, neighbour processor data, and output data. The first two, containing up to 10^5 physics events (equivalent to as many bunch crossings) are stored off-line in the input partition of the test bench memory. The test bench transfers detector data and inter-processor data to the FPGA-prototype using a synchronously clocked transfer. The transfer part plus the data processing by the FPGA-prototype is what may be called the actual test at full speed. The output is stored in the output partition of the test bench memory. Comparison with the VHDL simulation data is done off-line after the actual hardware test.

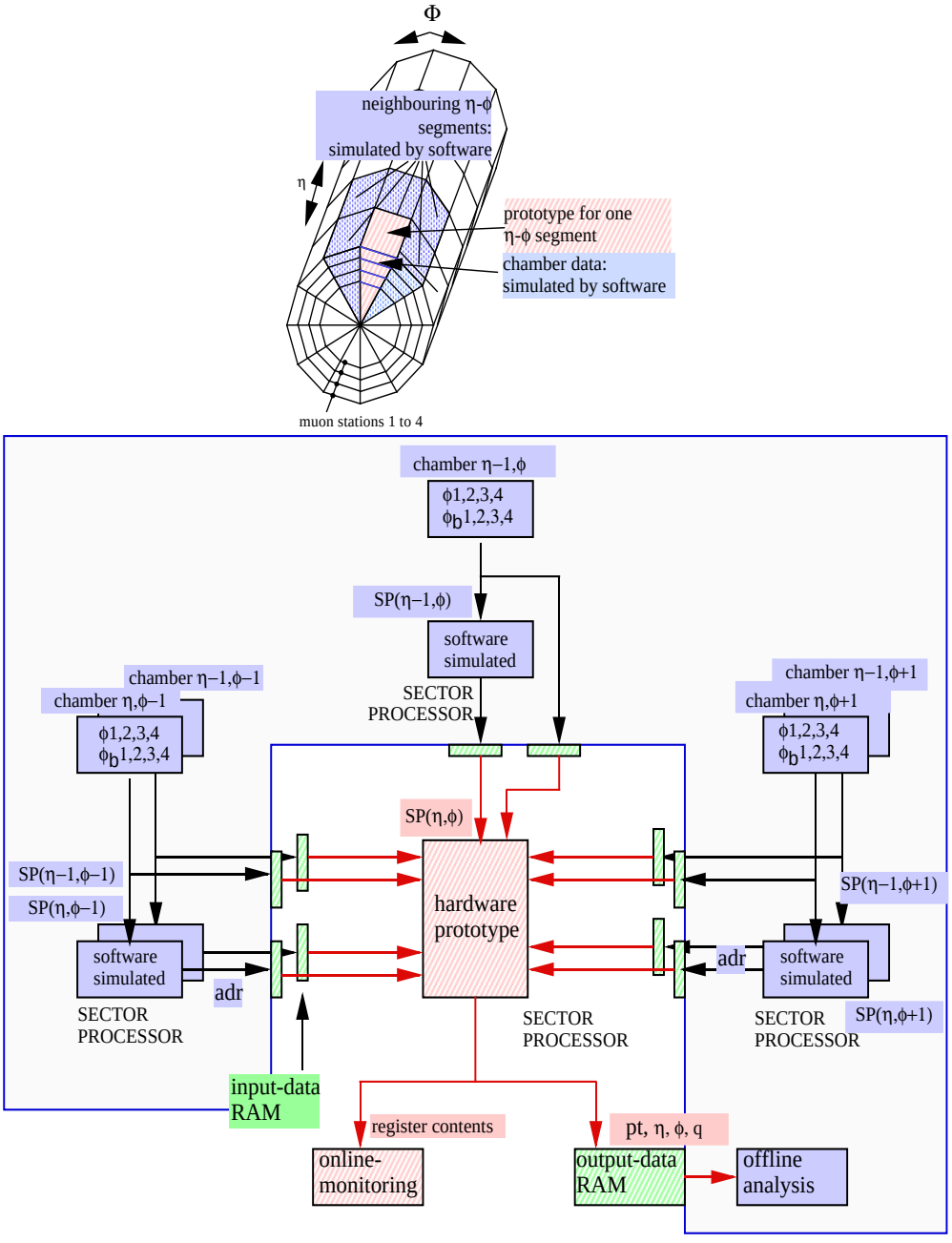


Fig. 6.2.: Test configuration of the FPGA-prototype.

The trigger test requires provision of test patterns with a data width of 1276 bits. This number includes chamber data as well as data coming from neighbouring processors (38 track segments each 22 bits; 440 bits of intermediate results from the neighbour processors). Input data passing to the processor are time multiplexed. Test patterns need to be sent in two parts. Pattern generators with a width of 638 bits are required for transmitting data using a 40 MHz clock. A flexible and versatile test system was conceived. 32-bit wide fast dual port memories [95] which can be used at frequencies up to 100 MHz on one port were employed; this corresponds to a sustained data transfer rate of 400 MB/s on the fast port. Multiplexer boards are fitted between the memories and the FPGA-prototype in order to be able to double the number of bits. The multiplexer boards are custom-designed for use with the fast dual port memories.

The dual port memories may conveniently be loaded via their second port which is a full VME-interface. A common clock signal allows synchronous transfer using the second port, which also triggers the output sequence of the test patterns.

Output data of the FPGA-prototype as well as some intermediate results for error detection (see chapter 8) are read out using another set of multiplexers and dual port memories. In total the prototype delivers 140 bits per processed event. After processing all test patterns the FPGA-prototype processor’s output is read to the work station via the VME-link and analysed off-line.

7 Feasibility study for ASIC implementation of the CMS track finder processor

In order to meet the given requirements at the present state of technological advance the only possible solution for a full-scale, full-speed implementation of the track finder processor is to employ application specific integrated circuits (ASICs). A possible implementation using ASICs is presented. In particular system partitioning and processor timing are addressed.

7.1. Implementation strategy and problems

The technological progress in producing integrated circuits is very fast. At present 0.5 µm CMOS processes are standard. 0.35 µm technologies are expected to be publicly available in the near future [96]. However, it will be shown that 0.7 µm CMOS technology is sufficiently advanced to meet the given requirements. System implementation is only simpler when benefit may be drawn from additional technology improvements.

When implementing an ASIC-based design general system parameters and requirements have to be compared to the vendor's specifications in the first place. The system must be partitioned into several logical units. Units must match the target technology.

The major factors that affect system partitioning with ASICs are:

- type of ASIC design
- type of logic
- maximum die size or gate count
- speed requirements
- power dissipation
- I/O per chip available
- package (and prototyping) costs

Designers must consider ASIC family capabilities in at least two phases of an ASIC development - first, during vendor selection, and second during design partitioning. A rough estimate of requirements is performed which should provide a good guideline for

correct vendor and ASIC family selection. Later, during detailed system design, the ASIC family capabilities are considered when deciding the final system partitions.

Two main approaches to ASIC designs are considered: gate array architecture and standard cell architecture [97].

Gate array architecture

In a gate array the ASIC vendor prefabricates ASIC transistors into arrays and then fixes the array position on a given wafer (or die). Interconnecting these transistors with metal stripes achieves logical functions during the metallization process. Typically, a designer selects and interconnects logical elements (also known as cells) from a gate array library; the latter have a predefined transistor design. Sometimes a vendor will also supply a set of hard macros (super macros) with a predefined circuit layout and hence predictable performance parameters.

Gate arrays have become the dominant form of ASIC technology in terms of quantities sold. A combination of ever-increasing gate densities, low cost, first pass successes, and fast turn-around times continues to make gate arrays the volume winner over the standard cell architecture.

Gate arrays only require customization of the wafer's interconnection layers i.e. one to three masks. This allows a vendor to fabricate every gate array wafer identically. Approximately four fifths of the whole fabrication process remain unchanged which helps boosting consistency and reliability.

Standard cell architecture

Each cell may differ in a standard cell ASIC. The vendor optimizes the transistors for area and performance, predefines each logical element, and makes available "super-macros". Like in the gate array approach design proceeds by selecting the proper library cells and by defining their interconnection.

The difference between gate array and standard cell ASICs lies mainly in the nature of the cell. Whereas the gate array cell consists of an array of identical transistors, the standard cell consists of transistors of different size. Transistors are optimized for the cell's function. The standard cell gets smaller and faster for a given function than the gate array. However, the desirable characteristics of the standard cell require a completely new design at every layer of the chip. On the contrary, gate array chips differ only in the top metal-related layers.

Estimating die size, timing and power dissipation

The core active area, the periphery active area, and interconnect area determine the required die size for a standard cell design and the gate count for a gate array respectively. Timing is dependent on the implemented logic, the routing and the fan-out of transmitted signals. Power dissipation is mostly dependent on the leakage current, switching current, current associated with charging and discharging of capacitive loads, and dc current that comes from transistors in on-state. To facilitate these estimates the vendor supplies formulas that use various aspects of a logic-level design to predict these areas.

Obviously the estimates are crude at the beginning of the design phase. They can only be refined when the target technology is chosen and comprehensive cell library informations are available. Software simulation tools allow to enter the desired functionality of the circuit and to calculate the corresponding ASIC parameters.

7.2. System partition, gate count and timing estimation

In the case of the track finder processor the decision which technology is going to be applied certainly also depends on an economical point of view. Hence when the target technology is chosen at the time of a possible final implementation the system partitioning must be refined in order to achieve the most effective and hence cheapest way for this technology. Two main approaches may be considered:

- small number of large (powerful) chips or
- large number of small (and more universal) chips.

In the first the processing part is contained within a few units. Hence the interconnection problem and its influence on the execution time is at its lowest. The drawback is a higher production cost. In some cases the more expensive chip design is compensated for by fewer interconnections which lowers peripheral cost.

In the latter case a large number of smaller, more universal chips is employed. Some restrictions apply. Obviously not every design scheme can be translated easily into a hardware model. A breakup into small identical or similar processing units is not always feasible. In case this possibility becomes available (parallel processing, ALUs, etc.) production cost of small chips is lower. However, the amount of interconnections between processing units increases. So does the processing time. If the design cannot employ entirely identical units, the universal chip must be designed such as to permit a change of the operating mode. The programmability of hardwired architecture chips is done using multiplexers or tri-state buffers to reroute signals to different processing blocks. As a consequence additional processing delay occurs.

As mentioned above the system partitioning depends mainly on the target technology. In the following one possible solution for the track finder processor employing only two types of large powerful of ASICs is discussed. It is clear that different technologies suggest different system partitioning. However, the gate count and propagation delay estimate will not change very much whichever technology comes into play. The ASIC-implementation study was conducted for a gate array architecture:

- gate array Motorola H4C series 0.7 µm [98]: available number of gates ranging from 18000 to 317000 (equivalent to a die size of 10 mm^2 up to 180 mm^2) and I/O package pins from 80 to 447 pins. The typical gate delay propagation time is 365 ps.

It should be noted that the 0.7 µm technology is already outdated by the 0.5 µm process providing a higher gate count and even smaller propagation time.

The first step is to partition the entire system. Again the limiting ASIC parameters are gate count (die size) and I/O pin count. In case of the track finder processor it is not to be expected that the amount of logic to implement will exceed the available die size or gate count. However, the data amount to be loaded into the integrated circuits requires many I/O pins. Packages with more than 400 pins are available; even more important, the I/O bit rate can be increased to 80 MHz and higher. This enables to design a synchronously running system using a 40 MHz-clock. Multiplexed I/O data transfer could be a way of solving the loading problem.

The partitioning was performed for the track finder system for six and for nine neighbouring detector segments (see chapter 5.2.1.). This mostly influences the number of necessary I/O pins on the integrated circuits. In the following gate count and processing time delay are given only for the system with nine neighbouring detector segments. Obviously gate count and amount of delay are much smaller for the system coping with only six neighbouring segments.

The mentioned ASIC system parameters are based on estimates using the formulas provided by the vendor. The gate propagation time in CMOS ASICs is dependent on a constant term t_{p0} plus a term for charging the capacitive load C_{Lout} on the output of the gate. The process and environment parameter *PTV* considers process variations, operating temperature and circuit supply voltage (equation 7.1).

$$t_p \approx (t_{p0} + k \cdot C_{Lout}) \cdot PTV \qquad \text{(Eqn. 7.1.)}$$

In CMOS technologies both rise and fall time, propagation times (t_{pLH0}, t_{pHL0}) and rising and falling load sensitivity factors (k_r,k_f) can differ from each other considerably. Typical values for t_{pLH0}, t_{pHL0}, k_r, k_f and *PTV* for the Motorola H4C-0.7 µ series [98] are given in table 7-1.

macro	t_{pLH} [ns]	t_{pHL} [ns]	k_r [ns/pF]	k_f [ns/pF]	C_{lout} Fan-out =1 [pF]	C_{lout} Fan-out =10 [pF]	t_{pLH} for Fan-out =1 [pF]	PTV best case, δ=25°, Vcc=5.5 V	PTV worst case, δ=70°, Vcc=4.5 V
2 fold NAND	0.17	0.12	1.38	0.94	0.3	2.01	0.58	0.72	1.59

Table 7-1: Typical macro delay properties Motorola H4C-0.7 µ series [98].

For each of the processing blocks of the track finder processor a typical case (temperature $\delta = 25^\circ$ and supply voltage V_{cc} = 5.0 V) and worst case (temperature $\delta = 70^\circ$ and supply voltage V_{cc} = 4.5 V) propagation time estimate was conducted. The gate count was estimated using the tables given for vendor supplied macros.

Fig. 7.1 shows the block diagram of the track finder processor and the suggested system partitioning. Each grey box stands for an integrated circuit. Two ASIC types, a number of memory based lookup tables and field programmable gate arrays FPGAs are employed:

Table 7-2 illustrates the properties of the employed units.

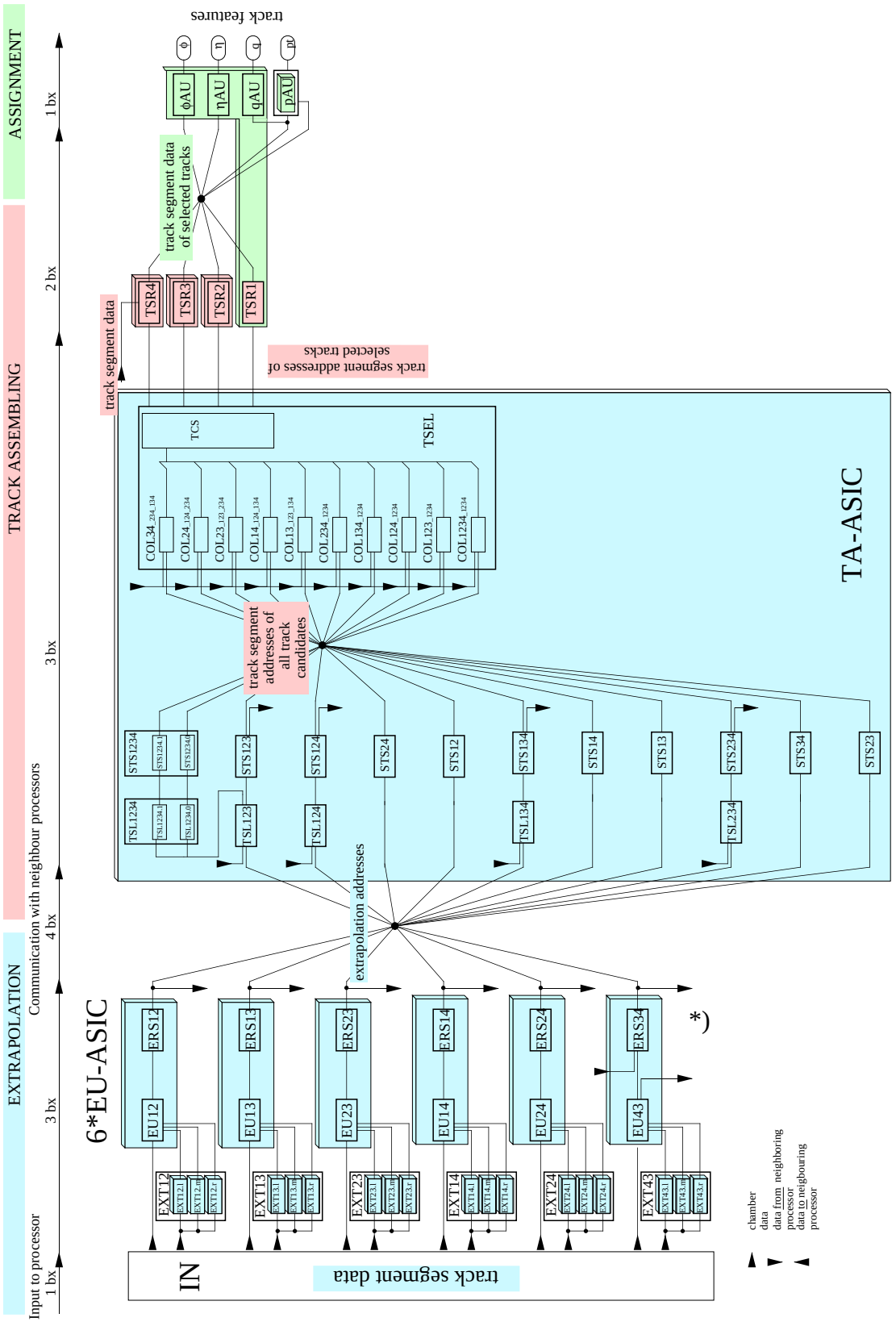


*Fig. 7.1.: ASIC-system partition. *)...see point (4) in text.*

(1) The input block IN prepares the data to be input into the ASIC (time multiplexing). Standard line drivers are used.
(2) The extrapolation value calculation EXT is performed using SRAM-based lookup tables.
(3) The extrapolation-ASICs EU-ASIC, containing the extrapolation units EU and the extrapolation result selectors ERS for a station pair, perform the extrapolation comparisons and give out the two addresses of matched target track segments.
(4) The track assembler-ASIC TA-ASIC, containing all track segment linker TSL, single track selectors STS, cancellation units COL and track class selector TCS, assembles matched track segment pairs and outputs the relative address of the track segments of the found tracks.
The extrapolation from station four to three works in the opposite direction with respect to all other extrapolations. The extrapolation selector ERS34 must find the two track segments in station three TS3 which are the target of the extrapolation from station four. Extrapolation results from the neighboring segments must be combined in the processor and evaluated in an extrapolation result selector ERS34. In order to reduce inter-chip transmission ERS34 is located within the track assembler-ASIC TA-ASIC.
(5) Track segment routing TSR employs field programmable gate arrays.
(6) All assignment units, except the p_t-assignment unit PAU, fit into field programmable gate arrays.
(7) The p_t-assignment unit PAU is realized using a SRAM based lookup table.
(8) IN-EU-neigh., OUT-EU-neigh., IN-TA-neigh., OUT-TA-neigh. stand for the I/O units for the data transfer between sector processors of different detector segments.

The timing specification for the entire system including the wheel sorter requires the execution to be completed within 21 bunch crossings (bx = 25 ns; see chapter 3.1.). As stated in chapter 5.2.4.) the wheel sorter needs up to seven bx. As a consequence 14 bx are left for the track finder processor itself. The table shows that the ASIC-based system can accomplish this requirement.

As can be seen in the table the gate count does not pose a problem to the integrated circuits. However, the I/O pin count is high in some units. As already mentioned earlier, the problem may be solved by multiplexing input pins with an integer multiple of the chip's operating frequency. If all data are input into and output from the integrated circuit within one bunch crossing, one cycle latency is added by the I/O process. The necessary multiplexing factor m_{IO} may be derived from the fraction of the number of input bits n_{input} and available I/O pin count p_{IO}. The resulting I/O bit rate b_{IO} is the product of multiplexing factor $m_{I/O}$ and the operating frequency f_{op} (equations 7.2. and 7.3).

$$m_{IO} = \frac{n_{input}}{p_{IO}} \qquad \text{(Eqn. 7.2.)}$$

$$b_{IO} = m_{IO} \cdot f_{op} \qquad \text{(Eqn. 7.3.)}$$

The input synchronization and buffer stages IN have to prepare the processor's input data for insertion into the ASIC. Compensation for the phase shift between the input

unit name	type	size	I/O count for 9 neighboring detector segments	I/O count for 6 neighboring detector segments	processing time typical, PTV=1.0 25˚, supply=5.0 V	processing time worst case, PTV=1.49 70˚, supply=4.5 V	latency contribution in bx
IN-chamber	Latch, MUX	-	1484	1004			1
EXT	SRAM	256 x 18	-	-	8.5 ns	8.5 ns	0(*)
EU-ASIC	ASIC	9000 gates	402	286	23.8 ns	35.5 ns	3(**)
IN-EU-neigh.	Latch, MUX	-	700	408			1
OUT-EU-neigh.	Latch, DEMUX	-	120	88			1
IN-TA-neigh.	Latch, MUX	-	12	12			1
OUT-TA-neigh.	Latch, DEMUX	-	120	120			1
TA-ASIC	ASIC	32000 gates	942	626	23.9 ns	35.6 ns	3(**)
TSR	FPGA	6 x 2 x 22 = 264 bit RAM	406	272			2
AU except PAU	FPGA	8000 gates	262	250			1
PAU	SRAM	128k x 8	-	-	10 ns	10 ns	
OUT-track finder	Latch	-	38	38			0(***)
total	VME board	9U	2474	1670	-	-	14

Table 7-2: : Properties of ASIC based system parts,
(*) does not add to latency, because EXT works parallel to EU-ASIC
(**) Latency is calculated for an I/O-frequency of 80 MHz (multiplexing factor $m_{I/O} = 2$)
(***) Transmission and synchronisation included in wheel sorter latency.

channels must also be done. Although a first compensation step is already performed by the transmission unit from the detector to the control room the input stage of the track finder processor must be able to shift the phase of the incoming signals up to 0.5 bx (12.5 ns) relative to each other. Input data is arriving at the control room at a certain time after the according bunch crossing. The full dataset arrives in parallel format at a rate of 40 MHz. Input transfer to the ASICs is done time multiplexed. A total latency of 1 bx (25 ns) is foreseen for all input stages IN. If the way of transmitting the data from the detector to the track finder processor is adapted in a way that the necessary data format is provided already by the transmitting stage one half of a bunch crossing can be gained.

All extrapolation lookup tables EXT are designed using SRAMs of the size 256 x 18 bits. Employing SRAMs with an access time of less than 8.5 ns allows the design of one EXT unit using a single RAM. The EXT unit is responsible for providing extrapolation values of two track segments (one after the other) within one bx. Such devices are available today [99]. The access time of the extrapolation lookup tables

EXT does not contribute to the total latency because EXT works in parallel with the extrapolation ASIC.

The extrapolation-ASIC EU-ASIC must have a maximum of 402 input bits (again considering the most complex case of nine neighbouring detector segments). Applying a multiplexing factor m_{IO} of two gives an I/O pin count of 201 pins and an I/O bit rate b_{IO} of 80 MHz. Using a modern ASIC technology these requirements can easily be met. The execution time corresponds to three bunch crossings (75 ns).

The track assembler-ASIC TA-ASIC must have 942 input bits, again considering nine neighbouring detector segments. Applying a multiplexing factor m_{IO} of two yields an I/O pin count of 471 pins with a I/O bit rate b_{IO} of 80 MHz. By today's standard such an I/O count is technical feasible but at very high cost. One option is to increase the multiplexing factor m_{IO} to three which gives 314 I/O pins and an I/O bit rate of 120 MHz. Although an I/O bit rate of 120 MHz is fully feasible some additional complexity is added when using an odd multiple of the clock. It has to be noted that the increased transfer frequency would only apply on the board level. The transmission of data from neighbouring processors onto different boards may be performed at a lower rate.

However, there still is a solution to further reduce the necessary number of input bits (and pins) at the track assembler-ASIC input. Up to now track assembling was considered progressing only in one direction, namely from the inside to the outside of the detector. Obviously, when starting in the centre wheel particles may diverge in both z-directions, making it necessary to assemble track segments taking into account nine neighbouring detector segments (see fig 5.17). However, when performing the track assembling from the outside towards the inside of the detector the possible z-direction of the tracks is determined and hence only six neighbouring detector segments must be taken into account; in the centre wheel only three. In addition background noise in the chambers of station four is considerably lower because punchthrough [100] of particles other than muons is smaller in this region.

The drawback of outside-to-inside algorithms for track assembly is a more complicated single track selection STS. Anyway, the decision in which direction the tracks will be assembled must be postponed until more detailed simulation results of the cathode strip chambers in the forward region will be available. It might as well be possible that the requirements regarding the forward and overlap region between drift tubes DT and cathode strip chambers CSC completely rule out one track assembling direction because of chamber geometry. Anyway, the track assembling-ASIC needs only 626 input bits for six neighbouring detector segments. A multiplexing factor m_{io} of two is sufficient to get done with 313 input pins and an I/O bit rate of 80 MHz. The process is terminated within three bunch crossings (75 ns).

The track segment router TSR may be realized employing field programmable gate arrays FPGA. The necessary I/O bit count of 406 and 272 respectively cannot be implemented using FPGAs with a clock frequency of 40 MHz. However, the task of a track segment router can be distributed to two parallel units, each having only one half of the inputs. The outputs may be combined employing tri-state buffers introducing no additional latency. A track segment router latency of two bx (50 ns) may be achieved.

The assignment unit AU, except the p_t-assignment unit PAU, is designed using two sets of two parallel FPGAs. Data I/O (up to 262 bits) can be split up between the FPGAs.

The p_t-assignment unit PAU is designed by a SRAM of the size 128k times 8 bits. Using a RAM with an access time of 8.5 ns allows to use only one RAM to assign the momentum to two tracks within one bx. Within one bx the assignment process is terminated. An additional bx is reserved for driving the signals up to the sorting stages.

The total latency including input and output buffers plus synchronization yields 14 bunch crossings or 350 ns. Adding seven bunch crossings for the wheel sorter and synchronisation yields 21 bunch crossings for the entire ASIC-built track finder system. This corresponds exactly to the required value as stated in chapter 3.1. on page 23. Employing ASICs for the track segment routers TSR and assignment units AU instead of FPGAs further reduces the processing time. Removing all FPGAs from the design and replacing them by ASICs is a possibility to reduce the I/O bit rate for the track assembler-ASIC. It then becomes possible to sacrifice a bunch crossing to read in data into the chip at a lower frequency (e.g. at 80 MHz). However, this would require two track assembling-ASIC to work in parallel.

The gate count for the two suggested ASICs (9000, 32000; see table 7-2) is very low and does not pose any implementation problem. This paves the way for envisaging an implementation of all functions of the two ASICs (EU and TA) within one mode-switchable ASIC. A mode pin must be reserved to invoke the corresponding function.

One sector processor needs at least six EU-ASICs and one TA-ASIC. Consequently the barrel track finder system, consisting of 60 sector processors, employs 360 EU-chips and 60 TA-chips. Using a mode-switchable 420 equal ASICs would be sufficient.

It should be noted that this estimation is based on a worst case study using today’s technology. It was shown that even now the requirements can be fully met with no restrictions to the functionality of the track finder processor.

In Appendix C alternatives to the base line system partitioning can be found.

Modern ASIC technologies support even faster inter-ASIC communication by providing dedicated system solutions. An overview of a system solution is given in appendix D.

8 Error detection and location

The track finder processor is embedded in the trigger system of the experiment CMS. Any error inside the track finder processor will directly influence datataking and thus the performance of the entire experiment. As a consequence any error inside the trigger system must be detected as soon as possible. This is done using a sophisticated monitoring system. Once an error has been monitored the error must be located. This is achieved by means of an error location scheme.

8.1. Error detection

The first level trigger with its subsystems is a very complicated digital processing unit. Simulations down to the level of single logical gates theoretically prove the correct behaviour of the implemented systems. Once implemented in hardware all units are going to be tested for their functionality. Requirements are met if system behaviour is identical to what has been simulated before.

However, as the trigger systems have direct influence on the datataking process, it is more than vital to prove at all operating times that the systems perform as required. Any error must be detected as soon as possible because the data taken with an impaired trigger system cannot be relied on and may even have to be discarded.

In previously implemented high energy experiments the trigger often needed to be monitored by analyzing off-line data. This implies two major difficulties. When the data is taken using the trigger which is to be tested, only the data of those events are available off-line which the trigger system selected. All events which have met the trigger conditions but erroneously not recorded cannot be accessed by this method. For test purposes the trigger system must perform normal processing but its decisions are in no way linked with datataking. In fact there is a random selection of events. However, trigger response is recorded. The trigger decision for these events is used to check trigger performance. In general trigger testing off-line results in late error reporting.

A more sophisticated monitoring coverage is achieved when trigger input data and trigger output data is compared directly with each other using an on-line system. The on-line system recalculates the trigger decision for a sample of input data. The result is compared with the trigger output. In this way errors can be detected fast. The data

acquisition controller can be notified immediately. All events, be they with positive or negative trigger decision, can be examined. However, when choosing the event sample randomly the fraction of events tested with positive trigger results over all events tested equals the reciprocal value of the trigger reduction rate. As a consequence only a very small sample of events which are supposed to have a positive trigger result can actually be looked at.

While an on-line system allows to monitor the trigger even during its normal operation the information derived will only allow to evaluate the correctness of the trigger result. Error detection must be done elsewhere.

However, as the track finder system is a largely distributed system employing a high number of processing units, more information about the entire system status must be available in case of an error. The track finder works in a pipelined manner which imposes a strong boundary condition. In addition to connection and hardware failures the occurrence of synchronization and timing errors must be covered. Status information about the entire system becomes a necessity.

A status monitoring system was designed for the track finder processor. Fig. 8.1 shows the system layout. Traffic on all data paths between integrated circuits may be loaded into the status shift register at the board's operating frequency. Using a shift register allows analysis of several subsequent events. As the system works in a pipelined manner corresponding register contents belonging to one event leave the processing units at a different time. Delay units *del* are foreseen to ensure that data from the same events have the same location in all shift registers. After the monitoring run data can be read out via a low bandwidth link and compared to the data provided by an on-line computer. Latter recalculates the corresponding trigger response (and also all register contents).

In case of an error it is important to know all inputs and outputs words of the various processing units (register of the processor). An on-line computer recalculates the track finder output and all intermediate results (i.e., the register contents) for the corresponding event. The processing unit causing the error can now be identified.

When an error occurs several preceding and subsequent register contents are available to the analysis units. Synchronization and time delay errors may be detected this way.

Also follow-up errors which occur only after a certain event, history may be traced this way. Some errors occur under certain circumstances only. They can be tracked down only after comprehensive analysis of the error conditions, the amount of data to be processed usually exceeding the computing power available online. An error (word) trigger solves the selection problem. Register contents are compared to preset values TRIG (see fig. 8.1). Throughout the processor predefined patters should match register contents. In case this comparison proves to be true the monitoring registers are stopped in a way that a programmable number of preceding and subsequent events become available for further analysis.

This feature may also be used to test events generating only positive trigger results or events containing at least one muon. Monitoring trigger conditions are set to trigger on a certain output pattern (for instance a pattern corresponding to $p_t \neq 0$). A central

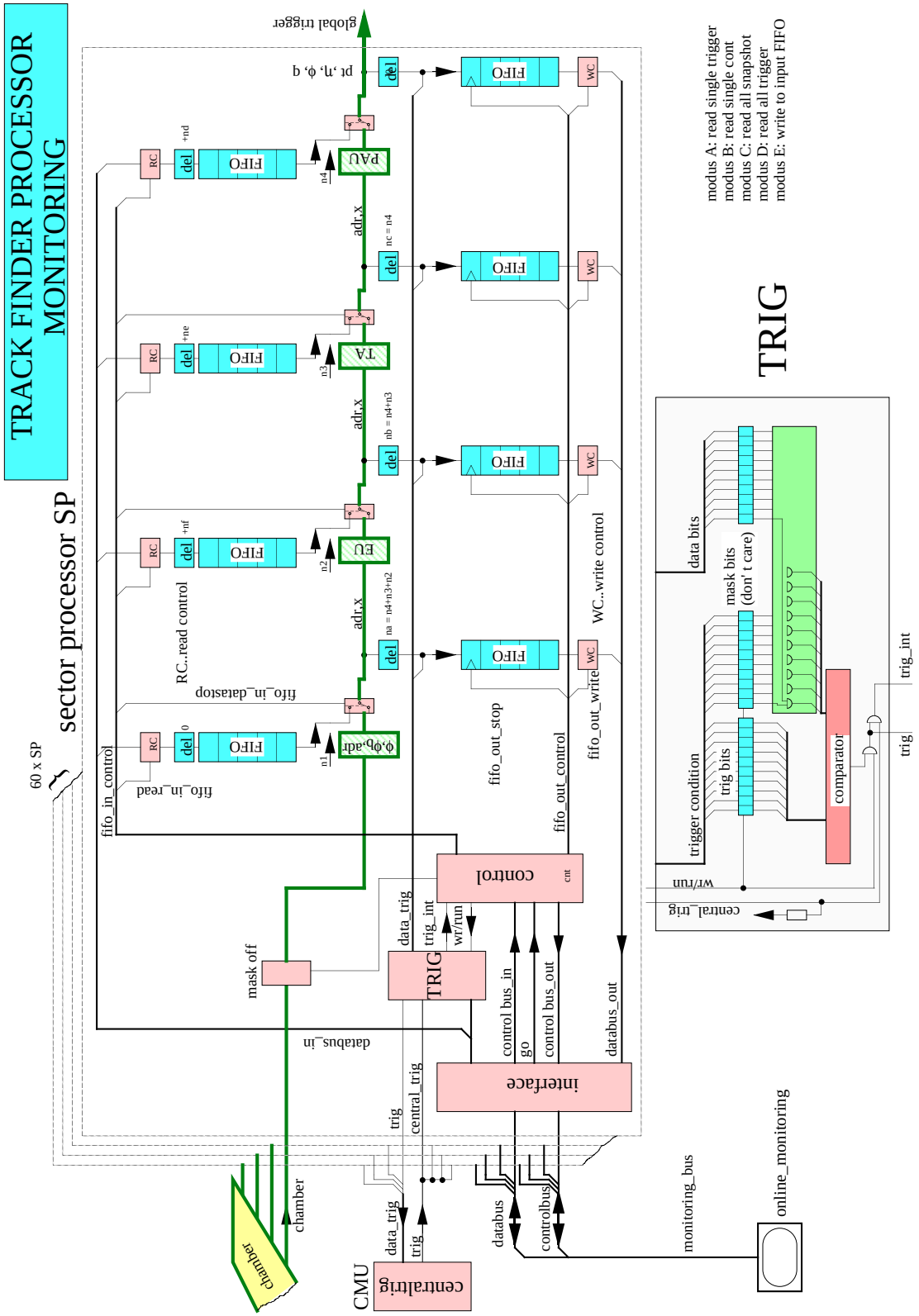


Fig. 8.1.: Track finder processor monitoring system.

monitoring unit CMU allows to set trigger monitoring conditions for all sector processors in the entire system.

The monitoring system works independently from the processing units. During normal operation of the trigger system it supplies basic data that is processed further downstream. General trigger monitoring and, looking at the whole experiment, detector monitoring uses error detection facilities like the one described above as its basic building blocks.

8.2. Error location

Errors detected by the monitoring system must be located.

The monitoring system outputs the register contents. Measured data are compared to data processed by the error analysis computer. The error location can be determined to be between the last register with correct contents and the first register with false contents. Either the processing unit (integrated circuit) or the connections to and from the units are malfunctioning. For errors of this kind the IEEE boundary scan architecture [101] is employed to further track down the error. The boundary scan architecture is especially adapted to:

- test the interconnections between integrated circuits once they are installed on a printed circuit board;
- test the integrated circuit itself.

The circuitry defined by this standard allows test instructions and associated test data to be fed into a component and, subsequently, allows results of execution of such instructions to be read out. All information (instructions, test data, and test results) is communicated in a serial format.

All integrated circuits are tested in a dedicated environment before they are installed on the board. After final assembly on a printed circuit board access is limited to the interconnections accessible from outside. It is impossible to test components in the same way as before. In-circuit test features must be provided during system layout.

To ensure that built-in test facilities can be used or that pre-existing test patterns can be applied, a framework is needed that can be used to convey test data to or from the boundaries of individual components so that they can be fully tested. The IEEE boundary-scan in conjunction with a dedicated test access bus provides such a framework.

The boundary scan technique involves the inclusion of a shift-register stage (contained in a boundary-scan cell) adjacent to each component pin so that signals at component boundaries can be set and observed using scan testing principles. [101]

Fig. 8.2 illustrates an implementation example for a boundary-scan cell which can be applied for an input and output connection to the circuit under test. Dependent on

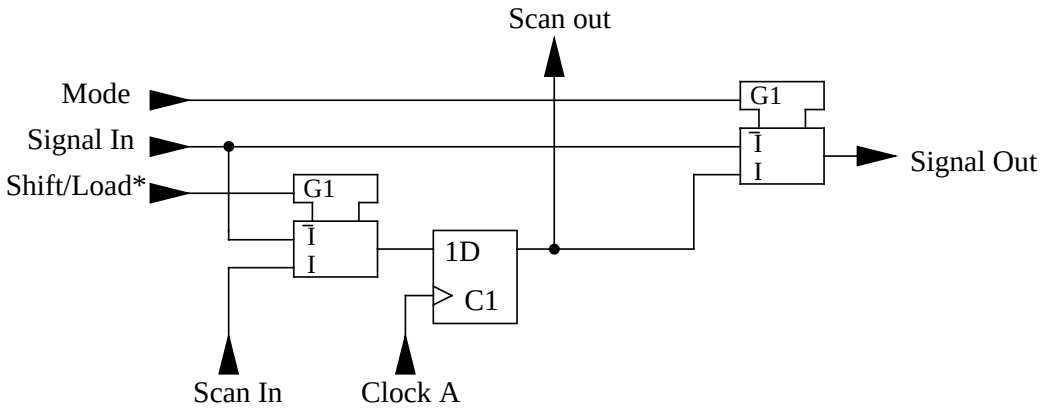


Fig. 8.2.: Boundary scan cell.

the multiplexer control signals data can either be read from the input *Signal In* into the boundary scan register. Data is now accessible on the *Scan Out* pin (mode “read output pins of a circuit”) or data can be sent from *Scan In* to *Signal Out* and from there directly into the core of the circuit (mode “send test pattern”). The data path during normal operation runs from *Signal In* to *Signal Out*. The scan register can be loaded during normal operation of the circuit.

The boundary-scan cells for the pins of a component are interconnected such that they make up a shift-register chain around the periphery of the circuit to be tested. This path is provided with serial input and output connections; an appropriate clock; necessary control signals (see fig. 8.3).

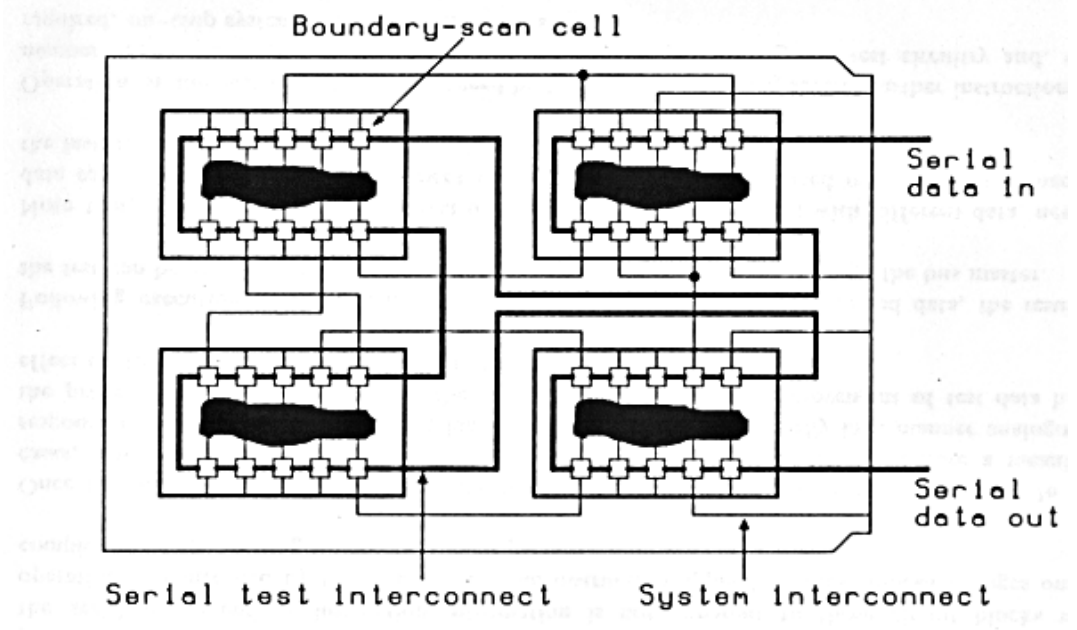


Fig. 8.3.: Boundary scan cells form a shift register chain. [101]

In case all the components in a circuit are equipped with a boundary-scan register the resulting path through the complete circuit can be used to trace mainly two things:

- Boundary scans permit testing of interconnections between the components. Test data can be shifted into all the boundary-scan register cells associated with component output pins and loaded in parallel through the component interconnections into cells associated with input pins.
- Boundary scans also permit complete functionality tests of each component on the board.The boundary-scan register can be used to load stimuli data into the inputs of a circuitry and to examine the data on the output of the system.

During these two test modes normal operation of the circuit is not possible anymore. However, this is not considered a limitation because an error has already been detected. The defective unit or board cannot be used any more for the decision taking process. Error search may usually be performed in an undisturbed way, whilst the remaining system is kept operational.

9 Conclusion and further perspectives

In chapter 3 the track finder processor specifications are compared to already existing systems. It is shown that no previous implemented system is suitable to be applied in the track finder processor environment. Chapter 4 demonstrates that conventional methods applied in hardware triggers fail for the track finder. Especially the most common approach, the pattern comparison, must be ruled out because of the large hardware extent. Instead, an algorithm based feature extraction method, the track segment method or extrapolation method, is introduced. It is shown that the algorithm copes with the track finder specifications. The algorithm can be implemented with a minimum of hardware. Several target technologies are considered. Advantages and disadvantages as well as technological progress necessary for an implementation are highlighted. Chapter 5 covers a detailed description of algorithm and its implementation. Using VHDL simulation the algorithms and its hardware representation were optimized. Simulation shows clearly that the simplicity of the design concept, namely reducing data flow in subsequent steps (by extrapolation, track assembly and property assignment) and selecting the highest ranking track candidate after each reduction step without sacrificing measurement accuracy, proves to be an efficient method. In chapter 6 the hardware of a FPGA-prototype is described. The algorithm is implemented in hardware with a small effort. The prototype clearly shows the proper functionality of the implemented system. Instead of storing a high number of track patterns (pattern comparison method) the problem is brought back to the algorithm level. Using simple logic modules, such as multiplexers, comparators, subtractors and logic gates, proves to be an important key point for the success of the design. However, it is pointed out that the number of bits to be processed in parallel poses a challenge to the hardware implementation. Chapter 7 covers a possible final implementation using ASICs. It is shown that the algorithm can be placed easily into an ASIC based hardware system that meets the given requirements. Processing speed and simplicity of the design are kept. Although simulations and prototype design gave encouraging results it is vital to keep in mind that the finally implemented and running system will be subject to failures at some point. Thus chapter 8 is fully dedicated to error detection and location. A minimum specification for an error detection system is described.

However, it has to be stated that some important implementation issues are not tackled at all. One example would be the synchronisation of the entire track finder processor to the first level trigger environment and to the clock system of the entire experiment. One should not forget that the track finder processor and much more the first level trigger are geographically largely distributed and complex systems. Distances for data transfers are found to be in the 100 m range. The system works

synchronously using only one clock referenced to the accelerator. Signals of each processing step are to arrive (and to be digested) synchronously at predetermined times. However, problems of this kind are beyond the scope of this work.

The experiment CMS is scheduled to start in 2005. The time until the actual start must be used to complete the design of CMS. Prototypes of detector parts and electronics will have to be built and tested. The prototype results will provide valuable information for the development of the final implementation.

As a consequence the design of the track finder processor, as it is introduced in this work, cannot at all be regarded as terminated. Although both the simulation and the prototype already delivered sophisticated results the track finder design presented here represents only a first step towards final implementation. The work suggests an algorithm and an implementation method. However, as the surrounding environment of the processor will evolve more and more both the specifications and the implementation of the track finder processor will have to be refined accordingly.

In the future it will be necessary to study changes in the track finder environment carefully. This is especially true for the cathode strip chamber based trigger primitive generator. Simulations of this system and the resulting design specification will have direct influence on the track finder processor. Physics simulation will have to be conducted to adapt the algorithm to the forward region of the detector where cathode strip chambers are supposed to deliver the trigger primitives [22]. Since cathode strip chamber trigger primitives are required to be similar to the drift tube chamber primitives the adaptations of the algorithm are not expected to be severe. However, chamber geometry and particle rates are different in the forward region. This may make it necessary to adopt changes to the mapping of the process space onto the hardware level. This implies a modification of number and hardware partition of the processing units. Specifically, due to the higher particle rate in the forward region it might be unavoidable to accept more than two track segments per station in a detector segment. The concept of the algorithm will remain the same. Only the number of processing modules will change. However, the number of track segments to be processed increases significantly. As discussed in previous chapters the number of bits to be processed is already on the verge of feasibility when using a pipelined design. On one hand the chip packages have a limited I/O pin count. On the other hand the number of connections to a printed circuit board is limited due to geometrical reasons. These problems can be solved using a mixed hardware architecture. Pipelined design is combined with a ‘round robin’ based parallel architecture. Several cycles are used to read data into a processing unit. Within the processing unit the data are processed in a pipelined way. According to the number of read cycles a certain number of units must be used in parallel. Two possible hardware partition schemes are discussed in appendix C.

Once the specification of the trigger primitive generators will be settled it becomes necessary to build a prototype of the processor (or part of it) using the technology of the final implementation. By today’s standards this will be an ASIC-based technology. However, technological advance could pave the way for something beyond ASICs (e.g. faster FPGAs).

Appendix A

A.1. Hardware simulations

As in any design it is very important to discover inconsistencies as early as possible. Later in the design phase a correction might be costly or even impossible. Since the track finder system is a complicated distributed digital system designing and prototyping the system gets impossible without dedicated verification techniques. During the design process simulations of the system were conducted using the hardware description language VHDL [102]. In this chapter the technique of using VHDL to design a system is described.

In designing physics experiments, simulations have been used for a long time to study the feasibility of the experiment in terms of the expected detector response. In such simulations the detector and data acquisitions systems, or parts of it, are modelled and the probable output from different physical reactions are studied. Particles are generated corresponding to the real events that are expected in the experiment. These particles are then followed through the detector, at all stages using the known laws of physics to determine the effects of their interactions with matter. New particles are created along the way in different processes. Some of the particles will, at some stage, reach a detection element where an electrical signal is generated to be further analysed by the data acquisition system. Following this, the effect of different trigger algorithms can be investigated [103].

Once the first and mostly crude requirements have been formed, an efficient implementation onto hardware level must be found. Simulations of the physical processes are fully common for a long time. However, with the growing complexity of electronic systems involved in high energy physics experiments, simulation becomes important in the field of designing electronics as well. The process of finding the specification for a certain system and the process of mapping the requirements onto a hardware system is closely related to each other. Specifications must be worked out with respect to the feasibility of their implementation. Often it is not obvious from an estimation point of view if a certain requirement can be met employing the available technologies. Simulations of the system level but also the hardware and gate level respectively must be conducted in order to refine specifications. An iteration process of physics simulation and hardware simulation is being performed converging in the final specification of the performance of the system and the resulting hardware requirements.

When performing design studies two main methodologies may be considered.

Top-down and bottom-up design methodology

In the top-down the top hierarchy is worked out first and the architecture is refined afterwards until the lowest level is reached. Bottom-up works the other way round. Basic building blocks are combined in order to form a more complicated system. The top-down method is in most ways the more preferred method as it allows to focus to the overall system aspects at first. In most designs both methods have to be used in close relationship to each other. When applying top-down method the system is splitted into a number of sub units. However, as the requirements of a system touch the limit of feasibility for a given technology it is not obvious whether a certain sub unit is implementable at all. Bottom-up methodologies must be employed to investigate the feasibility of a certain building block. The higher level descriptions have to be modified to implement the refined sub units into the system. New low level investigation might have to be performed again. An iteration process of top-down and bottom-up design processes leads to the final implementable design.

The main advantage of using hierarchical design methodologies is that the design can be optimized at the different levels. An optimization method may be applied especially adapted to each level. However, inefficiencies introduced at a higher level in most cases cannot be compensated for at a lower hierarchy level.

Verification

Hierarchic design methods can only be applied successfully if prove can be produced that the model resembles the same functionality in all levels. When changing from one level to the next simulations must be conducted to ensure no inconsistencies are introduced by the newly created level. This is achieved by comparing the results of the levels to each other and by checking that the new level still meet the requirements of the entire system.

Simulation

When implementing a system one encounters several levels of simulating the hardware model:

- Requirement simulation,
- implementation independent simulation, and
- implementation dependent simulation.

Requirements simulation are used to study the feasibility of a proposed system in order to arrive at a specification. An example for a requirement simulation is the simulation conducted to estimate the necessary number of patterns to be stored for the pattern comparison method (see chapter 4.1.1.).

In implementation independent simulations the overall system aspects are studied without relation to any specific implementation in hardware. They are intended as a way to investigate certain aspects of potential designs and to pinpoint weak parts.

In implementation dependent simulations, on the other hand, the system has been partitioned and the simulations are more closely connected with real hardware.

Obviously very often combinations of the several simulation classes are applied. For each of the simulation steps an according description of the system is used. Dependent

on the level of abstraction the description represents the system in the following ways from high to low level:

- Behavioral description: the description reacts more or less the same way the final design will without assuming any specific structure for the final implementation. Timing information is not included in most cases, since it cannot be known until the final design is completed and the technology chosen.
- Register transfer level describes the data flow through the model and the operations performed on the data described.
- Structural description already includes a structure of several connected behavioral described parts.
- Gate level description represents the design as connected logic gates. Such a description can be used for implementations in several technologies, thus the description is not yet completely technology dependent.
- Switch Level description is a mixture of logic and physical simulations, where all transistors are modeled as simple switches but where some physical effects have been included.
- Analogue description represents as a set of coupled differential equations and the simulations are performed by solving these equations numerically.

The lower in the design hierarchy the larger get the need of computing resources. In complicated, large systems it will not be possible and also not useful to simulate the entire system in the analogue or switch level but also the gate level representation. On the other hand carefully examining critical parts in a low level description while the other system components are represented in a less costly description is certainly useful.

A.2.VHDL-VHSIC hardware description language

The track finder processor was designed and optimized using the IEEE standard hardware description language VHDL. In the following a short introduction to the description language is given. However, for comprehensive information and the application about VHDL is has to be referred to the literature [102, 104, 105].

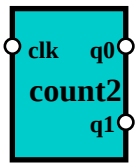
In the first place VHDL was intended as a way to write specifications to communicate designs from one contractor to another, as well as documenting the designs and providing for reuse of the designs in future products. 1987 IEEE decided to standardize the language and approved it as IEEE standard 1076-1987. It was reviewed 1992 resulting in an improved version, called VHDL’92.

VHDL is a language for describing digital electronic systems. The advantages can be summarized as follows:

- Public availability, because it is an official IEEE standard.
- Design methodology and design architecture independent: e.g. top-down, bottom-up, library based methods are supported. Design technologies, like synchronous versus asynchronous can be described.

- Technology and process independent: The description of a system can be done in a way entirely independent from the final implementation technology. However, once a technology is chosen the model can be refined in a way to support VHDL-input to the implementation tools.
- Wide range of description models: Simple behavioral models of an architecture may be refined down to the gate level with some restrictions even to the switch level.
- Design exchange: VHDL is a standard, and as a result, VHDL models are guaranteed to run on any system that conforms to that standard.

The basic concept of VHDL relies on two parts; the entity and the architecture.



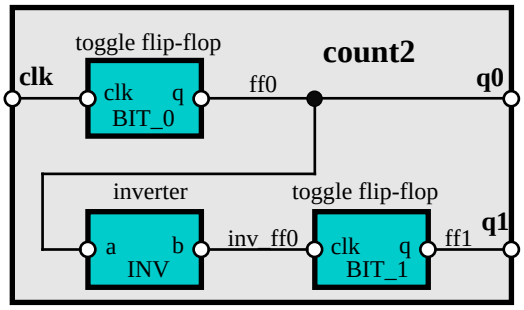
a)

```
entity count2 is
   port (clk:      in std_logic;
         q0, q1: out std_logic);
end count2;
```

b)

```
architecture vhdl_behavioral of count2 is
begin
  count_up: process (clk)
    variable count: integer:= 0;
  begin
    if (clk'event and clk = '1') then
      count := (count + 1) mod 4;
      q0 <= bit'val(count mod2);
      q1 <= bit'val(count / 2);
    end if;
  end process count_up;
end vhdl_behavioral;
```

behavioral description



c)

```
architecture structure of count2 is
  component toggle_flip_flop
    port (clk: in std_logic; q: out std_logic);
  end component;
  component inverter
    port (a: in std_logic; b: out std_logic);
  end component;
  signal ff0, ff1, inv_ff0: std_logic;
begin
  bit_0: toggle_flip_flop port map (clk => clk, q =>
ff0);
  inv: inverter port map (a => ff0, b => inv_ff0);
  bit_1: toggle_flip_flop port map (clk => inv_ff0,
                                    q => ff1);
  q0 <= ff0;
  q1 <= ff1;
end structure;
```

structural description

Fig. A.1.: VHDL descriptions of a 2 bit counter.

The entity can be seen as the interface of a system or system part to its environment. The inputs and outputs are defined there. Fig. A.1a illustrates an entity for a two bit counter. No statements about the functionality are given.

The architecture declaration defines the functionality of an entity. Signals are used to connect the various entities within an architecture. In the begin of a design the behavioural description specifies the desired function of a component. Fig. A.1b shows the VHDL representation of the counter in the behavioural representation. When design evolves and the structure of the system becomes apparent the model may be refined to a structural description. The VHDL description is given in fig. A.1c.

The VHDL description of a system is supposed to reflect the functionality of the real hardware with respect to both the logic states and the timing. All components in a hardware system work parallel to each other. However, VHDL simulations also must be able to be executed on non-parallel machines. Thus VHDL supports a two stage model of time. The two stage model is referred to as simulation cycle (see fig. A.2). After the start of the simulation first all signals are updated and thus set to their initial state. This first step, the updating process, is terminated when all signals (between processing units) which are scheduled to obtain new values at a current simulation time are updated. During the second step, the process execution, all processes (of the various entities) are carried out. Of course this is done by the simulation host in a sequentiell way. The second stage is completed after all processes have been executed. The simulation clock is set to the next simulation time at which a signals is scheduled to change its value and the simulation cycle is started again. The simulation is terminated when no further signal changes are scheduled [105].

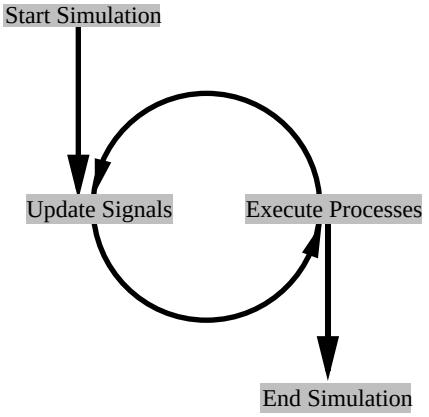


Fig. A.2.: VHDL simulation cycle.

As a consequence of this model there is always some delay between the time a process is carried out (and puts data on a data path) and the time when the signal reflects that value. Even if there is no delay assigned to a specific output a delta delay is used. This delay requires to pass a new simulation cycle before the signal is updated. This scheme ensures that all processes are carried out in a pseudo-parallel way. Omitting the two stage model would make the simulation dependent on the order of execution of the various processes.

Several tools are available which aid the designer to apply VHDL to implement a digital system.

- Graphic tools are available which allow to describe the functionality of instances in a graphic way (machine diagrams or flow charts). The software tools automatically provides a VHDL code.
  Another kind of VHDL graphic tools aid the designer to describe the structure of the system. Hereby the functionality of the components are described as VHDL code, while the structure (the connections between the instances) are given in a graphical way. Obviously the structure is easier to survey using a graphic approach than to describe the connections via a VHDL code only.
- Simulators run the VHDL code and output the responses of the entities. Comprehensive debugging facilities are available. Very often descriptions programmed in other languages than VHDL as VERILOG may be executed in combination.
- Syntesizers allow to directly translate the VHDL code to the gate level description adapted to a specific technology. Both logic synthesis, where a direct correspondence between VHDL code and hardware implementation exists, and high level synthesis, where the software description is interpreted in terms of equivalent hardware are available.

During the design of the track finder prototype it was found that the most efficient way to optimize and implement the hardware was to use graphic tools to describe

structure of the system only. Syntesizers were used intensively to map the VDHL code onto technology dependent gate level descriptions ready for implementation in the target technology. However, high level synthesis were consumed only with care. This is because the implementation, e.g. in field programmable gate arrays, touched the limits of what is possible today. The automated translation tools were not found to be sufficiently effective as to use the technology in a most efficient way with respect to partitioning the functions on the available space and more severe with respect to timing constraints.

In high energy experiments the influence of trigger systems was investigated already before using simulation methods. However, conventional languages as FORTRAN and C were employed. Of course the functionality of the logic was only described on a behavioural bases. As electronic systems become more and more complicated and are not easy to survey, more focus must be given to producing a hardware related simulation model also for physics simulation purposes. Obviously there is no sense in attempting to simulate the entire data acquisition system on a gate level for physics simulations. The used models must resemble the same functionality as the hardware. The necessity of keeping the same data format used in the hardware simulation (bits, bit vectors or in some cases integer values) become apparent when considering digitization errors or overflow problems. One can envisage combined simulation constructs; simulations using a conventional programming language and a hardware description language.

A combination of both, the detector simulation in FORTRAN and the VHDL simulation was used to optimize the track finder processor.

Appendix B

B.1. Example of the hardware implementation of the track segment linker modules.

The hardware implementation is illustrated on the example of one of the two

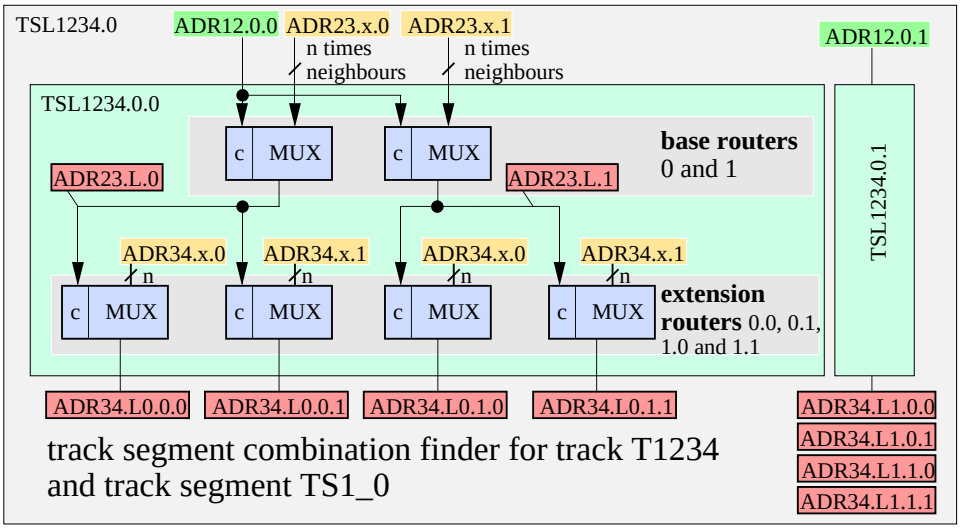


Fig. B.1.: Block diagram of track segment linker unit TSL1234.0.

TSL1234 which looks for tracks with track segments in station one, two, three, and four originating from track segment 0 in station one, see fig. B.1. The extrapolation result selector ERS12.0 of source track segment 0 gives out the two extrapolation addresses *ADR12.0.0* and *ADR12.0.1*. Thus track segment linker unit TSL1234.0 (for TS0) is sub divided in two parts (TSL1234.0.0 and TSL1234.0.1), each attempting to find a full track T1234 using one of the two extrapolation results as source track segment pair.

The control input of the first multiplexer, the base router, is connected to the first ERS address *ADR12.0.0* (base address). This is the first address of the extrapolation from track segment 0 in station one to the track segments in station two. All ERS addresses of the next station *ADR23* (including the addresses of the detector segment neighbours) are routed to the data inputs of the multiplexer. As each of the extrapolation result selectors ERS23 deliver two addresses, two multiplexers are employed in the baser router, giving already two possible track branches. Accordingly the output of the base router are the two addresses of the linked third track segments *ADR23.L.0* and *ADR23.L.1*. In order to append a track segment pair 3-4 one more multiplexer level, the extension routers, are employed. Hence there are two addresses

ADR23.L, two extension routers are necessary. Accordingly *ADR23.L.0* and *ADR23.L.1* respectively are connected to the control inputs; all *ADR34* are routed to the data inputs. The output of the extension routers are the addresses of the appended track segments of station four (*ADR34.L.0.0, ADR34.L.0.1, ADR34.L.1.0* and *ADR34.L.1.1*, see fig. B.1). An unsuccessful extrapolation is signalized by the ERS using an certain address code. If this code is applied to one of the multiplexer select inputs, it forwards this code to its outputs notifying the subsequent units of the lack of suitable track segment pairs.

For the base address *ADR12.0.1*another set of base routers and extension routers is necessary. Altogether the routers for the addresses *ADR12.0.0* (TSL1234.0.0) and *ADR12.0.1* (TSL1234.0.1) form one track segment linker unit (TSL1234.0) for tracks T1234 originating from track segment 0 in station one.

Of course there is a second track segment linker unit (TSL1234.1) necessary for the second source track segment in station one for track T1234.

The linking result bit is obtained by the track segment addresses of the last multiplexer stage. If it equals the code for a successful extrapolation the linking process for this track segment combination was successful too and the linking bit is activated.

For all tracks consisting of three track segments only base routers are applied. Tracks assembled only by track segment pairs do not use any routers at all.

B.2. Several cancel out schemes

Cancel Out Sub Pattern

The principle is described on several examples. Tracks of class T123 must be suppressed in presence of a track T1234 which originates from the same track segment as track T123. This is done simply by means of a logic gate.

The situation is more complicated in the case of a track T234. It must be suppressed if a track T1234 consists of the same track segments in station two, three and four as the track T234. However, if a track T1234 is built up by the same track segment in station two as the track T234, track segments in station three and four will also be identical. This is due to the identical linking scheme for tracks T1234 and T234. Hence it is only necessary to identify a track T1234 which uses the same track segment in station two. Note, that also tracks originating in different detector segments come into question. In all there are twelve track candidates T1234 which have to be investigated. The track segment addresses *ADR2* of all candidates T1234 are compared to the address of the track segment in station two of track T234. In case of a match the corresponding linking result bit is masked off.

All other track classes are cancelled out in a similar way. The less track segments a track class incorporates the more cancel out conditions are to apply. For example in case of a track T23 the track classes T123, T234 and T1234 must be investigated.

Cancel out single track

The task of this unit is to distinguish if multiple tracks from different classes are triggered by δ-rays. The single track selector prevents only from double tracks of the same class originating from the same track segment. Multiple tracks of different classes triggered by δ-rays are removed in this stage.

An example of track segment pair patterns of different classes caused by δ-rays is

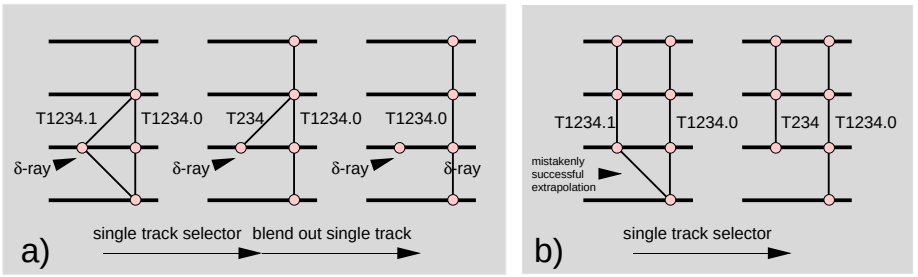


Fig. B.2.: Principle of the cancel out single track mechanism.

shown in fig. B.2a. Fig B.2b shows two particles close together. One of the tracks is missing a track segment in station one. The extrapolation from track segment in station one to the track segment in station two of the other track mistakenly was successful. In both cases (fig. B.2a and b) the single track selector retains one track T1234 and one T234. However in each cases a track T234 remains. In one case, the δ-ray case (fig B.2a), the track T234 is to be cancelled out, while in the second case track T234 has to be kept. The implementation principle is to check whether the track segments of the two remaining tracks T1234 and T234 are identical after the branch - in this example track segment 3. Again comparators and simple logic gates are employed.

Cancel out down

Cancel out down scheme removes double tracks which are identical except of their source track segment. An example of such tracks is illustrated in fig. B.3. A δ-ray caused a double hit in chamber one and thus two tracks T1234 are found.

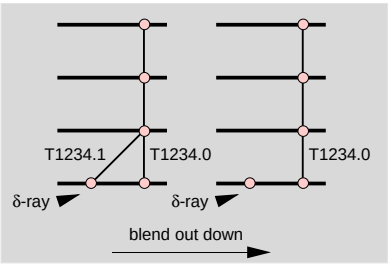


Fig. B.3.: Principle of cancel out down operation.

Cancel out down checks the extrapolation qualities of the extrapolation between station one and station two and cancels the track T1234 with the lower quality. Again comparators are employed. The hardware extent of this feature is large. It has to be noted that the probability of such configurations is low. This is due to the fact that track segments caused by δ-rays are unlikely to have a bend angle which extrapolates to the track segments in the next chamber. Further simulations and test beam results have to show if this feature is necessary.

Appendix C

C.1. Alternatives to the base line ASIC-system partitioning

In the following two alternatives to the base line system partitioning explained in chapter 7.2. are introduced. The presented schemes are supposed to guide appropriate ASIC technology selection at the time of final implementation.

A radical alternative to the baseline system partitioning is to implement all processing units from extrapolation EU, extrapolation value calculation EXT and extrapolation result selection ERS over track assembling TA and track segment routing TSR in one single integrated circuit. Two variants of this possibility are being introduced.

Variant 1:

The chip incorporates the equivalent of one track assembler chip, six extrapolation chips, the track segment router and assignment units (see chapter 7.2.).

The estimated gate count for one out of three track segment routers TSR is 1000 gates. The assignment units AU, except the p_t-assignment units, employs about 4000 gates. The total gate count is 93000 gates (see table 7-2). The extrapolation value calculation EXT employs SRAMs. If all processing parts are contained within an ASIC the total execution time is very fast, since no interchip transmission adds to the latency. Thus one can afford to employ only one RAM for one source track segment instead of three. Remember that in order to cope with the different origins of coordinates in the neighbouring detector segments three lookup tables are employed (see chapter 5.2.1.). The additional calculation time for subtracting a constant value from the track segment coordinates is below 7 ns. Employing RAMs as fast as to deliver extrapolation values for two extrapolations within one bunch crossing requires to implement only six RAMs of the size 256 x 18. This totals up to a memory bit count of about 28 kbit. A standard cell technology is more suited to implement a RAM within an ASIC. Using the standard cell technology ES2 0.7µm [106] the estimated required die size for this amount of RAM equals 15 mm^2. Using the size of a single gate in the ES2 0.7µm technology of $4 \cdot 10^{-4}$ mm^2 yields an necessary active die area of 54 mm^2 including the RAM, which is reasonable.

However, the amount of input bits into the system is very large, because all track segment data must be made available to the circuit. The total number of bits yields 1692 (6 neighbours) and 2496 (9 neighbours) respectively. Assuming a multiplexing factor

m_{IO} of two and an I/O pin count of 300 requires at least three bx (or four for nine neighbours) to read all data into the chip. Thus three (or four chips) must be used in parallel.

The total execution time for the chip is 57.3 ns or three bx. The delay of I/O multiplexing of chamber data (0.5 bx), EU-neighbour data (1bx), TA-neighbour data (1bx) and output data (0.5 bx) adds up to three bunch crossings. The data input process into the ASIC takes up to four bunch crossings. The p_t-assignment units requires a RAM of the size 128k x 8 (1 Mb). It cannot be contained by the ASIC. It will add another bunch crossing latency. For driving the output one bunch crossing is reserved. All this totals up to twelve bunch crossings.

Variant 2:

Up to now it was always considered to execute all extrapolations in parallel before starting the track assembling. Another implementation method combining the architectural structure of the extrapolation units and the track assembling units can be envisaged. For each track class a track searcher is employed. Beginning from the inner part of the detector extrapolations to the next chamber are being conducted and the address of up to two successful extrapolation targets are given out (block A in fig. C.1). Multiplexers MUX route the track segment data of the extrapolation targets to the subsequent processing stage, block B. Using the target track segments as new source track segments the extrapolations to the next stations are being performed in block B, resulting in the addresses of the newly appended target track segments. The next step extrapolating from station three to four is not possible, as explained in chapter 4.3. Thus extrapolations from station four to three must be performed parallel to the extrapolations between stations one and two. For each track segment in station three the addresses of matching track segments in stations four must be provided. Block C then routes the corresponding track segment address and track segment data of station four to its outputs.

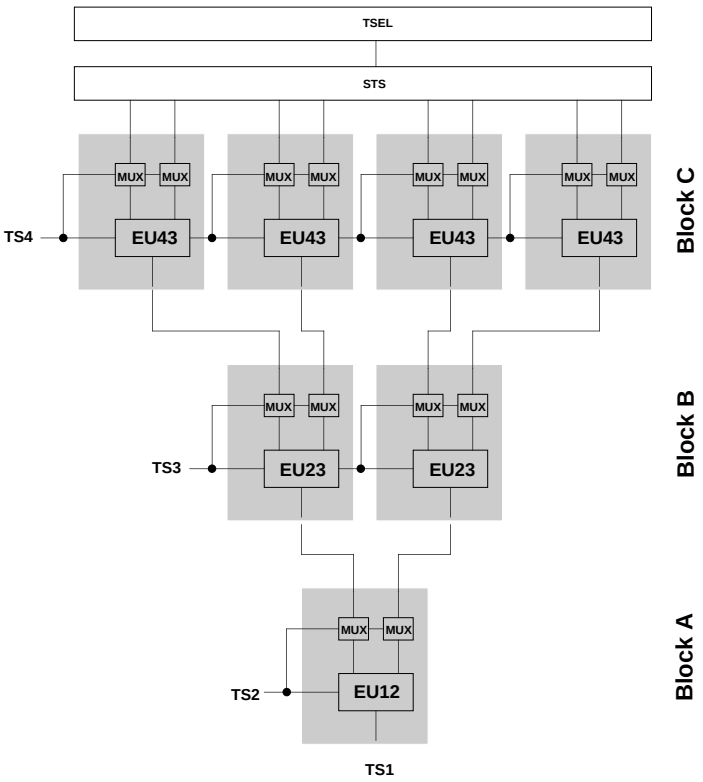


Fig. C.1.: Block diagram of variant 2.

After all extrapolations have been performed the single track selection STS is applied and data is forwarded to the track selector TSEL, track segment router TSR and assignment units AU as in the previous explained architecture.

The advantage of this scheme is that all data necessary for track assembling in a given detector segment are produced within the given track finder processor and thus are available without any interconnections between several boards. This is a major advantage since interconnections cause additional processing delay and consume immense physical space. The only interconnections necessary between processors are those for the cancel out units (total 132 bits I/O).

The drawback, however, is that the extrapolations are conducted in a serial way and the execution time for each subsequent station pair sums up.

Possible system partition for variant two

Two possibilities are open: One puts all necessary logic within one chip, another splits the system up into several small units with similar design features.

The first method is certainly favourable with respect to system complexity and execution time. However, the amount of input bits into the system is large. The total number of bits yields 1136 (6 neighbours) and 1616 (9 neighbours) respectively. Assuming a multiplexing factor m_{IO} of two and an I/O count of 300 requires at least two bunch crossings (or three for nine neighbours) to read all data into the chip. Two (or three) chips must be used in parallel.

As it is not clear at the process begin what track segment the extrapolation has to start from the extrapolation value calculation EXT cannot be dispersed from the chip. However, in this scheme a total of 38 extrapolations are necessary. Employing RAMs as fast as to deliver extrapolation values for two extrapolations within one bunch crossing requires to implement 19 RAMs of the size 256 x 18. This totals up to a memory bit count of about 90 kbit. Using the standard cell technology ES2 0.7µm [106] the estimated required die size for this amount of RAM equals 50 mm^2.

In addition the chip incorporates the equivalent of one track assembler chip and 19 extrapolations chips. This results in a gate count of 200000 gates. Using the size of a single gate in the ES2 0.7µm technology of $4 \cdot 10^{-4}$ mm^2 yields an necessary active die area of 135 mm^2 including the memory. Although this area is technically feasible by now die sizes of such dimensions should be avoided. Applying a more sophisticated technology as 0.5 or 0.35 mm^2 in the future will decrease the necessary active gate area.

The track segment router TSR and assignment units (except the p_t-assignment units PAU) can be incorporated by the integrated circuit as well, saving overall latency compared to the base line partitioning scheme. A rough execution delay estimation for this scheme yields 76.8 ns or 4 bunch crossings. The delay of I/O multiplexing of chamber data (0.5 bx), TA-neighbour data (1bx) and output data (0.5 bx) adds 2 bx. The data input process itself takes up to three bx. For assigning p_t using a SRAM and driving the outputs one bx is being reserved each. This totals up to eleven bx.

The other possibility to partition this system is to split the system up into several small units with similar design features. All blocks contain extrapolation units EU,

multiplexers MUX, single track selector STS and track selector TSR. Using various modes, units are switched on or off.

Compared to the single chip design the I/O pin count of the chips is reduced.

The drawback, however, is the necessity of frequent transmissions between ASICs and the resulting increased delay. For each inter-ASIC transmission at least one half of a bunch crossing must be accounted for. Compared to the previous partitioning scheme, employing only one ASIC, additional four inter-ASIC transmissions must be performed. This totals up to 13 bx.

Appendix D

D.1. System solutions for high speed multi ASIC systems

Usually when designing fast digital synchronously running systems the clock skew (caused by slow rise time of the clock signal) and clock path delay (insertation delay) causes severe problems to synchronize the entire system. The insertation delay is the delay from rising edge of the external system clock to the rising edge of the clock on any given flip-flop in the ASIC.

CMOS devices can display quite a spread of input threshold voltages. Thus a long rise time of the clock (skew) causes a difference in trigger time [107]. Usually in a CMOS design, the clock signals will drive the heaviest capacitive loads. The clock signals feed the entire chip and are often distributed throughout the chip architecture. The ASIC vendors provide clock distribution techniques within a chip to minimize the disadvantageous effects. However, still the ASIC devices on a board must be synchronized to each other compensating for the insertation delay.

To address these problems, a phase locked loop PLL can be added to each ASIC

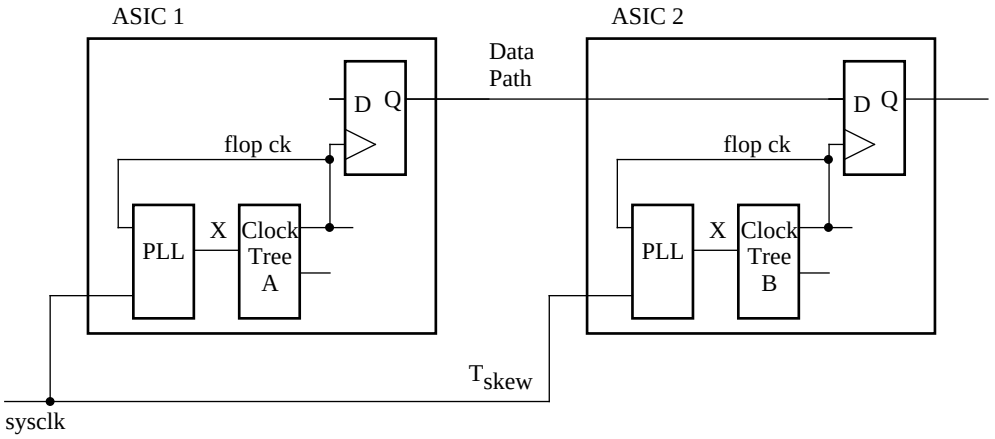


Fig. D.1.: Clock distribution using on-chip PLL [90].

device to reduce the effects of insertion delay differences (see fig. D.1). The PLL will synchronize the rising edge on *SYSCLK* such that it will be simultaneous with a rising edge on *flop ck*. If the PLL is used on each ASIC device all *flop ck* signals on every ASIC will be synchroneous within the error of the PLL. The PLL will compensate for differences in insertation delays from ASIC-to-ASIC as well as PTV variations [90].

Analog phase locked loop APLL on the ASIC provide on-chip frequency synthesis, which allows the board clock frequency to be multiplied up to the desired on-chip clock frequency [108]. This feature allows to realize a synchronous I/O bit rate other than the board clock frequency or the double of it using the rising and falling edge of the clock. Analog phase locked loops APPL with a frequency of up to 250 MHz are available on the ASICs. This allows an chip-to-chip interface of up to 250 MHz with today's technology [109].

Bibliography

[1] CERN, LHC Large Hadron Collider, CERN Publication, European Laboratory for Particle Physics, June 1990.

[2] CMS, The Compact Muon Solenoid, Technical Proposal, CERN/LHCC 94-38 LHCC/P1, 15 December 1994.

[3] C. W. Fabjan, Experimental Techniques in High-Energy Physics, Addison-Wesley Publishers, 1987.

[4] R. K. Bock et al., Data analysis techniques for high-energy physics experiments, Cambridge University Press, 1990, ISBN 0 521 34195 7.

[5] Simon Haykin, “Neural Networks A Comprehensive Foundation”, 1994.

[6] H. Kolanski, Application of Artificial Neural Networks in Particle Physics, DESY 95-061, April 1995, ISSN 0418-9833 or
Nucl. Instr. and Meth. A367 (1995), 14-20.

[7] Seez C.J., CMS, A general purpose detector for the LHC, Nucl.Instrum.Meth. 344 (1994), 1-10.

[8] Denegri D., The CMS Detector and Physics at the LHC, CERN-PPE/95-183, CMS Technical Note TN/95-167, November 16, 1995.

[9] A. J. Lankford, Trigger and Event Assembly for LHC Experiments, First Workshop on Electronics for LHC Experiments, Lisbon, CERN/LHCC/95-96, 1 October 1995, 313-318.

[10] R.E Hughes-Jones et al., Triggering and Data Acquisition for the LHC, Fifth Internation Conference on Electronics for Particle Physics, May 1995.

[11] W. Smith, G. Wrochna, Complementarity of the Two Components of the CMS Muon Trigger System, CMS Technical Note TN/95-014, February 15, 1995.

[12] S. Cittolin, Trigger and Data Acquisition at Large Hadron Collider, CERN School of Computing, Egmond aan Zee, Sept 9 1996.

[13] CMS Muon Trigger Group, CMS Muon Trigger Preliminary specifications of the baseline trigger algorithms, CMS TN/96-060, May 7, 1996.

[14] Czyrkowski et al., RPC Based CMS Muon Trigger, Progress Report, CMS TN/93-111, October 15, 1993.

[15] M. Andlinger et al., Pattern Comparator Trigger (PACT) for the muon system of the CMS experiment, Nucl. Instr. and Meth. A370 (1996) 389-395.

[16] M. De Giorgi et al., Design and Simulations of the Trigger Electronics for the CMS, First Workshop on Electronics for LHC Experiments, Lisbon, CERN/LHCC/95-96, 1 October 1995, 222-227.

[17] M. De Giorgi et al., Design and Simulations of the Trigger Electronics for the CMS Muon Barrel Chambers, CMS TN/95-01, January 12 1995.

[18] J. Hauser, Baseline Design for the CSC-based Endcap Muon Trigger, CMS TN/95-013, March 8, 1995

[19] A. Lee and P. Padley, CMS Endcap Muon Motherboard Logic, Rice Univerisity, not published yet.

[20] R. C. Fernow, Introduction to experimental particle physics, Cambridge University Press, 1986, ISBN 0 521 301 70 X.

[21] T. Wildschek, Simulation of the Silicon Tracker/Vertex Detector of the ATLAS Experiment at the Large Hadron Collider at CERN, Diploma Thesis, Technische Universität Wien, 1993.

[22] T. Wildschek, An algorithm for Track Finding in the CMS Muon Trigger, Dissertation, Technische Universität Wien, 1997.

[23] G. Wrochna, Momentum scale - a proposal, private communication, http://cmsdoc.cern.ch/doc/mu_tr/docs/MUTRGUIDE.HTML#MuonTrigger

[24] CMS, Muon Technical Design Report, in preparation.

[25] A. Kluge et al., Track Finding Processor in the DTBX Based CMS Barrel Muon Trigger, First Workshop on Electronics for LHC Experiments, CERN/LHCC/95-56, October 1, 1995.

[26] A. Kluge et al., Track Finding Processor in the DTBX Based CMS Barrel Muon Trigger, Second Workshop on Electronics for LHC Experiments, CERN/LHCC/96-39, October 21, 1996.

[27] L. M. Lederman, The Tevatron, Scientific American, March 1991.

[28] S. Abachi et al, The D0 Detector, Published in Nucl.Instrum.Meth.A338:185-253,1994.

[29] CDF Collaboration, The Collider Detector at Fermilab (CDF), IEEE Trans. Nucl. Sci. 33, No. 1, February 1986, 40.

[30] T. Ahmed et al., A pipelined first-level trigger for the H1 forward-muon spectrometer, Nucl.Instrum.Meth.A364 (1995), 456-472.

[31] ZEUS collaboration, The ZEUS Detector Technical Proposal 1986.

[32] M. Abolins et al., The Fast Trigger for the D0 Experiment, Nucl.Instrum.Meth.A289 (1990), 543-560.

[33] M. Fortner et al., The VME-Based D0 Muon Trigger Electronics, IEEE Trans. Nucl. Sci. 38, No. 2, April 1991, 480.

[34] D. Amidei et al., The Trigger System for the Collider Detector Facility, IEEE Trans. Nucl. Sci. 33, No. 1, February 1986, 63.

[35] D. Amidei et al., The CDF Trigger, Nucl.Instrum.Meth.A265 (1988), 326-335.

[36] D. Amidei et al., A Two Level Fastbus Based Trigger System for CDF, Nucl.Instrum.Meth.A269 (1988), 51-62.

[37] F. Bedeschi et al., Design and Construction of the CDF Central Tracking Chamber, Nucl.Instrum.Meth.A268 (1988), 50-74.

[38] G. W. Foster et al., A Fast Hardware Track-Finder for the CDF Central Tracking Chamber, Nucl.Instrum.Meth.A269 (1988), 93-100.

[39] K. Müller et al., H1 internal report H1-04/87-61, DESY, Hamburg,1987 or Nucl. Instr. and Meth. A312 (1992) 457.

[40] S. Eichenberger, A Fast Pipelined Trigger for the H1 Experiment at HERA Based on Multiwire Proportional Chamber Signals, Doct. Thesis Universität Zuerich, 1993 or
S. Eichenberger, A fast pipelined trigger for the H1 experiment based on multiwire proportional chamber signals, Nucl. Instr. and Meth. A323 (1992) 532-536.

[41] Th. Wolff et al., The Drift Chamber Track Finder for the First Level Trigger of the H1 experiment, Nucl.Instrum.Meth.A323 (1992), 537-541.

[42] J. Bürger et al., The Central Jet Chamber of the H1 Experiment, Nucl.Instrum.Meth.A279 (1989), 217-222.

[43] XILINX, The Programmable Logic Data Book,1994, U.S.A.

[44] B. Taylor, Optical Timing, Trigger And Control Distribution For LHC Detectors, IEEE Trans. Nucl. Sci. 41 (1994) 1294-1299.

[45] T. Kohonen, Content Addressable Memories, 2nd Ed., Springer-Verlag, 1987.

[46] Cypress Semiconductor, Data Sheet Advanced Information Cy7C915 1k x 42 SmartCAM.

[47] Music Semicoductors, Data Sheet MU9C1640 CacheCAM, June4 1993.

[48] S. R. Amendolia et al., Study of a Fast Trigger System on Beauty Events at Fixed Target Experiments and Colliders, Nucl. Instr. and Meth. A289 (1990) 539-542.

[49] Henk W. den Bok et al., Track recognition with an associative pattern memory, Nucl. Instr. and Meth. A300 (1991) 107-114.

[50] M. Dell'Orso, L. Ristori, VLSI Structures for Track Finding, Nucl. Instr. and Meth. A278 (1989) 436-440.

[51] S.R. Amendolia et al., The AMchip: a VLSI associate memory for track finding, Nucl. Instr. and Meth. A315 (1992) 446-448.

[52] S.R. Amendolia et al., The AMchip: a full-costum CMOS VLSI associative memory for pattern recognition, IEEE Transactions on Nuclear Science, Vol. 39, No.4, 1992, 795.

[53] Th. Lindblad, Using software and hardware neural networks in a Higgs search, Nucl. Instr. and Meth. A356 (1995) 498-506.

[54] B. Denby, E. Lessner, Test of Track Segment and Vertex Finding with Neural Networks, Fermilab-Conf-90/68, 1990.

[55] C. Baldanza, Results from a neural Trigger Based on the MA16 Microprocessor, Int. J. Mod. Phys. C6 (1995)567 or DFUB95/2, 1995.

[56] Intel Corp., 80170NX Specification booklet, June 19910085-18-X.

[57] B. Denby et al., Drift Chamber Tracking with Neural networks, IEEE Trans. on Nuclear Science, Vol. 40, No. 4 , August 1993.

[58] C. Brown et al., D0 Muon System with Proportional Drift Tube Chambers, Nucl. Instr. and Meth. A279 (1989), 331.

[59] M. Abolins et al., A High Luminosity Trigger Design for the Tevatron Collider Experiment at D0, IEEE Trans. on Nuclear Science, Vol. 36, No.1, February 1989, 384-389.

[60] C. Loomis and J. Conway, "Using an Analog Neural Network to Trigger on Tau Leptons at CDF", Proceedings of AIHENP95 , 1995.

[61] G. Athanasiu, P. Pavlopoulos and S. Vlachos, "A neural network trigger system for the CP-LEAR experiment", Proceedings of AIHENP95.

[62] S. Schiek, G. Schmidt, Application of a high speed analog neural network chip for first level triggering at the H1-Experiment at HERA, Proc. of the "International Conference on Artificial Neural Networks" ICANN'95, Oct. 9-3, 1995, Paris, France, Vol. 2, pp.363-368, ISBN 2-910085-18-X.

[63] Fent et al., The Realization of a Second Level Neural Network Trigger for the H1 Experiment at HERA, AIHENP'96, Lausanne, Switzerland, Sept.2-8, 1996.

[64] K. Hoen et al., 70 input 20 nanosecond pattern classifier, The 1994 IEEE International Conference on Neural Networks, Vol. 3, 1854-1859.

[65] J.C. Lassalle, Trident: A Track and Vertex Identification Program for the CERN OMEGA Particle Detector System, Nucl. Instr. and Meth. A176(1980) 371-379.

[66] J. Peoples, THE SSC PROJECT, Rome 1988, Proceedings, Particle accelerator, Vol. 1 1988, 237-241.

[67] M. E. Mermikides, A Track Finding Strategy for Superlayered Drift Chamber Design Studies, FSU-SCRI-90T-14, The Florida State University Tallahasse, Florida, Sept. 11, 1990.

[68] A. Kluge et al., Der österreiche Beitrag zum Experiment CMS am CERN, Österreichische Physikalische Gesellschaft Fachausschußtagung für Kern- und Teilchenphysik, Weyer, September1996.

[69] A. Kluge et al., Der Hardware Track Finder Processor in CMS am CERN, Österreichische Physikalische Gesellschaft Fachausschußtagung für Kern- und Teilchenphysik, Lindabrunn, September1997.

[70] P. Battaiotto et al., A Fastbus Module for Trigger Applications Based on a Digital Signal Processor and on Programmable Gate Arrays, INFN/AE-90/10, Servizio Documentazione dei Laboratori Nazionali di Frascati, 17 Sept. 1990.

[71] P. Aarnio et al., The DELPHI Detector at LEP, Nucl.Instrum.Meth.A303:233-276,1991.

[72] S. Quinton et al., An Overview of the First and Second Level Trigger of DELPHI, IEEE Trans. Nucl. Sci. 36 (1989) 390-394.

[73] V. Bocci et al., Architecture and performance of the DELPHI trigger system, Nucl. Instrum. Methods Phys. Res., A : 362 (1995), 361-385.

[74] R. K. Bock et al., A commercial image processing system for triggering in future LHC experiments, Nucl. Instr. and Meth. A356 (1995) 304-308.

[75] S. A. Guccione, Programming Fine Grained Reconfigurable Architectures, Doct. Thesis at University of Texas, Austin, May 1995.

[76] IEEE, IEEE Standard Test Access Port and Boundary-Scan Architecture, IEEE Std 1149.1-1990, Institute of Electrical and Electronics Engineers, Inc., New York, 1990.

[77] D. Belosloudtsev, Programmable active memories in real-time tasks: implementing data-driven triggers for LHC experiments, Nucl. Instr. and Meth. A356 (1995) 457-467.

[78] W. H. Kautz, Cellular logic-in-memory arrays, IEEE Transactions on Computers, vol. C-18, 719-727, August 1970.

[79] W. H. Kautz et al., Cellular interconnection arrays, IEEE Transactions on Electronics Computers, vol. C-17, 443-451, May 1968.

[80] R. C. Minnick, Outpoint cellular logic, IEEE Transactions on Electronics Computers, vol. EC-13, 685-698, December 1964.

[81] R.C. Minnick, A survey of microcellular research, Journal of the ACM, vol. 14, no.2, 203-241, 1967.

[82] P. Bertin et al., Programmable Active Memories: A Performance Assessment. Technical Report, Digital Equipment Corporation, Paris Research Laboratory, 1993.

[83] F. Klefenz et al., Proc. IEEE Nucl. Sci. Symp., San Francisco, CA, USA, 1993, 62-64.

[84] F. Klefenz er al., Proc. Comp. in High Energy Physics, Annecy, France, CERN Rep. 92-07 (1992), 799-802.

[85] F. Klefenz et al., ENABLE - A Systolic 2nd Level Trigger Processor For Track Finding and e/π Discrimination for ATLAS/LHC, IEEE Transactions on Nuclear Science, Vol. 41, No. 4, August 1994.

[86] H.Högl et al, A second generation FPGA processor, EAST note 94-35, ATLAS, CERN or
H. Högl et al., Enable++: A General-Purpose L2 Trigger Processor, First Workshop on Electronics for LHC Experiments, Lisbon, CERN/LHCC/95-96, 1 October 1995, 335-339 or
Proc. IEEE Nucl. Sci. Symp., San Francisco, CA, USA, 1995.

[87] ATLAS, ATLAS Technical Proposal for a General-Purpose pp Experiment at the Large Hadron Collider at CERN, CERN/LHCC/94-43, LHCC/P2, December 15 1994.

[88] D. Bursky, Advanced 100-kgate FPGA Family Takes On Gate Arrays, Electronic Design, May 1 1995, 51-58.

[89] G. Appelquist et al., An R & D Programme for Alternative Technologies for the ATLAS Level 1 Calorimeter Trigger, RD27 note 36, ATLAS DAQ-NO-32, 16 January 1995.

[90] Motorola Semiconductor, ASIC Clock Distribution Using a Phase-Locked-Loop (PLL), Application Note, AN1509, 1992, Motorola INC.

[91] D. H. Perkins, Introduction to High Energy Physics, 2nd Edition, 1982, Addison-Wesley Publishing Company, ISBN 0 201 05757 3.

[92] R. L. Gluckstern, Uncertainties in Track Momentum and Direction, due to Multiple Scattering and Measurement Errors, Nucl.Instrum.Meth.24 (1963), 381-389.

[93] N. Neumeister et al., CMS Global Trigger, CMS TN/97-009, January 20, 1997.

[94] Robertis G. De and Ranieri A., The Sorting Processor Project, CMS TN/95-028, March 6, 1995.

[95] S. Cittolin et al., Dual Port Memories in LHC Experiments, CMS-RD12, Technical Note 95-04, March 24, 1995.

[96] G. Declerck, CMOS and BICMOS IC Design, Future Trends in Technology for VLSI, EPFL Lausanne, September 6 1996.

[97] James Wall and Anne Macdonald, The NASA ASIC Guide , 1993 , Jet Propulsion Laboratory California Institute of Technology and National Aeronautics and Space Administration.

[98] Motorola Semiconductor, H4C Series Design Reference Guide, Customer Defined Arrays 0.7 Micron $L_{eff,}$, Motorola INC, 1993.

[99] Cypress Semiconductor, Cypress Data Book Cy7C178, 32k x 18 Synchronous Cache RAM, 1995.

[100] C. Albajar, Measurement of Hadronic Shower Punchthrough in Magnetic Field, CERN-PPE95-61, May 9 1995, Submitted to Zeitschrift für Physik.

[101] IEEE, IEEE Standard Test Access Port and Boundary-Scan Architecture, IEEE Std 1149.1-1990, Institute of Electrical and Electronics Engineers, Inc., 345 East 47th Street, New York, NY 10017, USA, May 21, 1990.

[102] IEEE, IEEE Standard VHDL Language Reference Manual, IEEE Std 1076-1987, IEEE Inc., New York, USA, 1987.

[103] G. Appelquist, Using Hardware Simulations in the Design of Large Scale Trigger and Data Acquisition Systems, Fysikum, Stockholm University, 1992.

[104] R. Lipsett , C. Schaefer, C. Ussery, VHDL: Hardware Description and Design, Kluwer Academic Publishers, ISBN 0-7923-9030-X, 1992.

[105] P. Ashenden, The VHDL Cookboook, First Edition, Dept. Computer Science, University of Adelaide, South Australia.

[106] European Silicon Structures, ES2 ECPD07 Library Databook.

[107] P. Horowitz and W. Hill, Art of electronics, Cambridge , Cambridge Univ. Press, 1989, 1125 p.

[108] Motorola Semiconductor, Analog Phase Locked Loop for H4CPlus, H4Eplus and M5C Series Arrays, Application Note, AN1522, 1994, Motorola INC.

[109] Motorola Semiconductor, H4CPlus Series Design Reference Guide, Customer Defined Arrays 0.6, Motorola INC.

LEBENSLAUF: Dipl. Ing. Alexander Kluge

Geburtsdatum und Ort: 15. April, 1968 in Wien
Staatbürgerschaft: Österreich

ADRESSE:

CERN/PPE
CH-1211 GENEVE 23
SCHWEIZ
Telefon: +41/22/767/7372
E-mail: alexander.kluge@cern.ch

AUSBILDUNG

Matura (Mai 1987)

Höhere Technische Bundeslehranstalt für Elektronik und Nachrichtentechnik, Wien.

Diplomstudium Elektrotechnik an der Technischen Universität Wien

(Dezember 1993)
Diplomarbeit: 'Ein schneller Trigger für ein Experiment am CERN; Institut für Hochenergiephysik Wien und Institut für Meßtechnik an der Technischen Universität Wien.

Doktorratsstudium (- September 1997)

'The Hardware Track Finder Processor in CMS at CERN' durchgeführt am CERN; Institut für Angewandte Elektronik an der Technischen Universität Wien und CERN, Genf.

BERUFSERFAHRUNG und FORSCHUNGSGEBIETE:

Entwicklung von analogen und digitalen Schaltkreisen bei einer Elektronikfirma.

(Mai 1987- Jänner 1989)**,** Technisches Büro Seufert, Gablitz, Österreich.

Entwicklung und Bau eines Myontrigger-Systems für das Experiment RD5 am CERN.

(März 1990 - September 1990; Juli 1991 - August 1991)
Computersimulationen bewiesen die richtige Funktion des entwickelten Triggeralgorithmus. Den Ergebnissen der Simulation entsprechend wurde ein elektronisches System entworfen. Dieses wurde durch ein Computerüberwachungsprogramm andauernd auf seine richtige Funktion hin überprüft.

[Alexander Kluge, Ein schneller Trigger für ein Experiment am CERN, Techn. Univ., Diplomarbeit, 1993]

Entwicklung eines Myonmeß-Systems für das Experiment CMS am CERN.

(Mai 1994 - September 1997)
Das System soll Myonen mittels 200000 Detektorzellen aufspüren und deren Ort sowie Transversalimpuls bestimmen. Mehrere Realisierungsmethoden wurden betrachtet. Die ökonomischste wurde ausgearbeitet. Mittels der Hardwarebeschreibungssprache VHDL wurde sowohl der Meßalgorithmus als auch die entwickelte Hardwarearchitektur auf ihre richtige Funktion getestet. Alle Simulationen fanden auf UNIX-Plattformen statt. Ein Protoyp des Systems wurde entwickelt, um an einem Teststand die Simulationsergebnisse mit tatsächlich gemessenen Daten vergleichen zu können. Die Ergebnisse dieses Projektes wurden in einer Dissertation zusammengefaßt.
[CMS, Compact Muon Solenoid, Technical Proposal, CERN-LHCC 94-38;
[1], [2]]

PROGRAMMIERKENNTNISSE

UNIX, VHDL, Fortran, Pascal;
Grundkenntnisse: C und Assembler

SPRACHKENNTNISSE:

Deutsch, Englisch:	fließend
Französisch:	Grundwissen

Publikationen

[1] A. Kluge, T. Wildschek, Track Finding Processor in the DTBX Based CMS Barrel Muon Trigger, First Workshop on Electronics for LHC Experiments, CERN/LHCC/95-56, October 1, 1995

[2] A. Kluge, T. Wildschek, Track Finding Processor in the DTBX Based CMS Barrel Muon Trigger, Second Workshop on Electronics for LHC Experiments, CERN/LHCC/96-39, October 21, 1996

[3] A. Kluge et al., Der österreiche Beitrag zum Experiment CMS am CERN, Österreichische Physikalische Gesellschaft Fachausschußtagung für Kern- und Teilchenphysik, Vortrag Weyer, September1996.

[4] A. Kluge et al., Der Hardware Track Finder Processor in CMS am CERN, Österreichische Physikalische Gesellschaft Fachausschußtagung für Kern- und Teilchenphysik, Vortrag Lindabrunn, September1997.

[5] F. Bakker et al., The construction and performance of single-layer honeycomb strip chambers in the TRACAL detectors of RD5, Nucl. Instrum. Methods Phys. Res., A : 330 (1993), 44-54.,

[6] Veröffentlichungen der RD5 collaboration am CERN.

[7] Veröffentlichungen der CMS collaboration am CERN.

CURRICULUM VITAE:
Dipl. Ing. Alexander Kluge

PERSONAL DATA

Name:	Alexander Kluge
Date of Birth:	April 15, 1968
Nationality:	Austrian

ADDRESS

CERN/PPE
CH-1211 GENEVE 23
SWITZERLAND
Phone: +41/22/76/77372
E-mail: alexander.kluge@cern.ch

EDUCATION

College Exam (Matura, May 1987)

Technical College for Electronics and Telecommunication (HTL), Vienna.
Special focus was given to the practical design of analog and digital circuits.

Dipl. Ing. Degree in Electronics Engineering (December 1993)

Diploma Thesis: ‘Ein schneller Trigger für ein Experiment am CERN (A fast trigger for an experiment at CERN)’, Institute for High Energy Physics, Vienna and Institut für Messtechnik at the University of Technology Vienna.

Ph.D. Student of Applied Electronics (May 1994 - September 1997)

Doctoral Thesis: ‘The Hardware Track Finder Processor in CMS at CERN’
Institute for Applied Electronics at the Technical University of Vienna and
CERN, Geneva.

PROFESSIONAL EXPERIENCE

Development of analog and digital systems at a telecommunication company (1987-1989).

Technisches Büro Seufert, Gablitz, Austria.

Design and development of a muon trigger system for the RD5-Experiment at CERN. (1992)

The system is capable of distinguishing between muons and pions in a time of less than 250 ns. It bases on honeycomb-chambers located in a sampling calorimeter. Computer simulations proved the functionality of the algorithm. A NIM-based modular trigger system was developed. An on-line real time monitoring for the trigger and the chamber has been realized.
[Alexander Kluge, Ein schneller Trigger für ein Experiment am CERN, Techn. Univ. Wien, Diploma Thesis, 1993]

Design of a muon track finder processor for the experiment CMS at CERN. (May 1994 - September 1997)

The system identifies muons and measures their transverse momentum in a time of less than 350 ns. It is realized in a pipelined manner to cope with the LHC data rate of 40 MHz. Several realisation methods were evaluated and the performance of the chosen base-line design was tested using a VHDL-based hardware model. A prototype has been designed and tested.
[CMS, Compact Muon Solenoid, Technical Proposal, CERN-LHCC 94-38; [1, 2, 3,4]

PROGRAMMING SKILLS

VHDL (hardware description language), Fortran, Pascal, UNIX and Assembler.

ELECTRONIC DESIGN AUTOMATION TOOL KNOWLEDGES

Cadence schematic entry and VHDL entry (Concept),
Cadence VHDL and VERILOG Simulator and Synthesis (Leapfrog, Opensim),
XILINX (-FPGA) Programming and Simulation tool (XACT),
Synthesis tools: ABEL, Exemplar, Galilieo,
Layout program: Powerpads,
Routing tool: Specctra.

LANGUAGE KNOWLEDGES

German, English, basic knowledge of French.

OTHER EDUCATIONS

Course: 1996 Making presentations.
Course: 1997 Communicating effectively.
Course: 1997 Communicating effectively in a team.

HOBBIES

Skiing, Mountaineering, Paragliding, Windsurfing, Mountain biking, Travelling

OTHER EXPERIENCES

Winters 1988-1993: Working as ski instructor in Austrian ski schools
Summer 1985 and 1986: Working at an elevator firm as construction worker (8 weeks).
Summer 1984: Working at an elevator firm as draftsman (4 weeks).

PUBLICATIONS

[1] A. Kluge, T. Wildschek, Track Finding Processor in the DTBX Based CMS Barrel Muon Trigger, First Workshop on Electronics for LHC Experiments, CERN/LHCC/95-56, October 1, 1995

[2] A. Kluge, T. Wildschek, Track Finding Processor in the DTBX Based CMS Barrel Muon Trigger, Second Workshop on Electronics for LHC Experiments, CERN/LHCC/96-39, October 21, 1996

[3] A. Kluge et al., Der österreichische Beitrag zum Experiment CMS am CERN, Österreichische Physikalische Gesellschaft, Fachausschußtagung für Kern- und Teilchenphysik, Weyer, September 1996.

[4] A. Kluge et al., Der Hardware Track Finder Processor in CMS am CERN, Österreichische Physikalische Gesellschaft, Fachausschußtagung für Kern- und Teilchenphysik, Lindabrunn, September 1997.

[5] F. Bakker et al., The construction and performance of single-layer honeycomb strip chambers in the TRACAL detectors of RD5, Nucl. Instrum. Methods Phys. Res., A : 330 (1993), 44-54.,

[6] Member of RD5 collaboration at CERN.

[7] Member of CMS collaboration at CERN.