ALICE TRD
Global Tracking Unit (GTU)

EDR Documentation

May, 2004

Prof. Dr. Volker Lindenstruth, Jan de Cuveland

Kirchhoff Institute of Physics
University of Heidelberg

Contents

List of Figures

List of Tables

1 Introduction

This document has been compiled as a preliminary version to form a part of the documentation for the GTU Design Review taking place in Heidelberg, 2004.

It consists of selected chapters taken trom the diploma thesis of J. de Cuveland and translated into English by courtesy of R. Eidsness.

In this document, the design of a Global Tracking Unit is presented, which represents the final part of the ALICE TRD trigger system. This unit three-dimensionally reassembles the track segments found in the detector's drift chambers, calculates the momentum of the originating particles from the course of the reconstructed tracks, and finally brings about the trigger decision. The objective of this thesis is to describe a hardware architecture which is able to perform the required calculations in less than 2 µs by means of massive parallelism. The analysis focusses particularly on an efficient implementation with regard to the utilized FPGA technology, while retaining high detection efficiency and reconstruction precision for relevant particles. The results show how a Global Tracking Unit meeting the experiment's real-time requirements can be realized.

2 Technical aspects of the GTU development

This chapter summarizes the working process for the development of the GTU design and the tools used in the process.

Before the GTU is implemented as a hardware description the requirements must be precisely specified and the principle computing procedure must be selected. At first a simulation is made in order to test the principle applicability of a procedure and to compare the different variations of computing in a simple way. The first part of this chapter is on the implementation of the simulation in the program environment of Root.

Starting from the simulation a hardware suitable model of GTU is developed. Therefore the computing procedure previously simulated will be transformed into an efficient hardware description. The form of the hardware description is the contents of the second part of this chapter.

2.1 Simulation with Root/AliRoot

For simulation, reconstruction, and analysis the detector systems of the ALICE experiment use a programming environment called *AliRoot*. AliRoot uses the *Root*[1] system as a basis for all the applications by putting the image of the function and geometry of the ALICE detectors on Root classes. Root is an object oriented program environment for data analysis and simulation which bases on C++. It is designed especially for the large amounts of data which are generated during LHC. Because of the C++ interpreter C++ can not only be used as a programming language for its own classes but as a fast script language as well. The various Root classes are documented in [BR03].

The simulation we made within the framework of this project contains parts of the computing of the LTUs plus the data transfer (see chapter 3) and also the total functions of the GTU (Chapters 4 and **??**).

During the implementation of the simulation the analysis code was separated from the real function as far as possible.
The function was implemented in the form of 11 C++classes which are deduced from the basic class of the framework `TObject`. The eleven classes amount to 4030 lines of source

[1] Root ist frei verfügbar und kann unter `http://root.cern.ch` im Quelltext und für diverse Systeme vorkompiliert heruntergeladen werden.

code. They are compiled into a library which can be integrated by the C++Interpreter CINT of the Root system
The analysis code was created in the form of C++scripts for the interpreter CINT each script dealing with a certain question. The scripts generate either an issue in text form, a Root data file or direct graphics in PostScript format.
A third group of programs uses the geometry data of the detector in order to generate value tables for the hardware implementation see chapter 4 and **??**). There is a total of 7500 source code lines in this simulation.

The first classes are integrated into the Root environment and the programming is done matching the style of (*Coding Conventions*) the AliRoot projects [Hri00]. The classes however are no part of AliRoot (at the moment). They instead use simulation data sets which are generated during the AliRoot simulation and are saved as a Root file. The advantage of this procedure is that the simulation does not rely on the sudden changes of the AliRoot environment. The disadvantage is that the data used might be outdated. However as the simulation during this phase is first of all for the concept design therefore it causes no problem if it is not according to the development of AliRoot. It is however important to recognize that the quantitative results presented in this chapter were slightly different from the ones that used the latest data. For the detector geometry the simulation uses the level of the technical design report [ALI01]. In chapter 7 there is a summary of the most important changes which occur during the process when the detector is planned.

2.1.1 Event data sets used

The simulation uses the Monte-Carlo procedure with the data of accidental collision events. Since the accelerator and detector are still under construction it is necessary to use synthetically generated events.

The event files used are derived from a HIJING[2] event generator which generates the particles required and their momentum by using accidental values for a collision event. One of the parameters for the generation is the multiplicity density[3] $\frac{\mathrm{d}N_{\mathrm{ch}}}{\mathrm{d}\eta}$ of the event. As the theoretical prediction of the expected multiplicity density differs from one another and at the same time the efficiency of the trigger depends decisively on the multiplicity density we use event files for various multiplicity density values (see table 2.1).

The traces which are interesting for the trigger are not yet contained in the HIJING-Data The events have been enriched by 50 for the simulation process of GTU Υ particles of which each decays into a pair of high energy electron and positron which can be detected. Hence

[2] "Heavy Ion Jet INteraction Generator"

[3] The *Multiplicity density* $\frac{\mathrm{d}N_{\mathrm{ch}}}{\mathrm{d}\eta}$ how many loaded particles $\mathrm{d}N_{\mathrm{ch}}$ are produced during the collision of a pseudo rapidity interval. $\mathrm{d}\eta$ Pseudo rapidity is a function of the production angle θ to the beam axis. It is defined as $\eta = -\ln(\tan(\frac{1}{2}\theta))$ and is a good approximation for relativistic particles for rapidity [BV98]. Rapidity is a relativistic velocity measurement without a unit $\tanh^{-1}\beta$, at which $\beta = \frac{v}{\mathrm{c}}$ with particle velocity v and light velocity c. The angle range which covers the TRD, amounts to a pseudo rapidity range of about $-0.9 \leq \eta \leq 0.9$. On average multiplicity density of $\frac{\mathrm{d}N_{\mathrm{ch}}}{\mathrm{d}\eta} = 8000$ 14,400 charged particles would be produced in the solid angle of the detector.

multiplicity density $\frac{\mathrm{d}N_{\mathrm{ch}}}{\mathrm{d}\eta}$	0	2000	4000	8000
added $\mathrm{e}^+/\mathrm{e}^-$-couples	50	50	50	50
number of events	100	40	40	40
name tag	signal	quarter	half	full

Table 2.1: The event file sets of the simulation.

the number of searched particles is far beyond the number expected in the simulation as a matter of fact we only expect an interesting trace only at very few events. Only when we do it like this we can already with a small number of event files statistically proof the detection probability.

We took the event files from a more abstract "theoretical" simulation of the detectors from Dr. B. Vulpescu. They already contain information on the raw track on the LTU-plane where the increase of the influence of *tilted Pads* is not corrected (see section 3.2.2). For every track segment there are among others the number of the detector module, the number of the pad row, the points of intersection y_0 and y_1 with the drift chamber[4] and for the control contain information on the causing particle. The compressed files of the 220 test events in total are together 6.4 GByte. They form as a reference data set the basis for all the simulation results presented.

2.2 Hardware description in VHDL

Starting from the Root-simulation we develop in this work a hardware realization of the GTU. For the definition of the architecture the hardware description language VHDL[5] is used. With the help of VHDL the single function blocks of a diagram of connections can exactly be described on various abstraction planes. As an introduction into VHDL as well as a reference [LWS94] is useful.

The VHDL description which was generated during this work consists of 22 modules with a total of 8019 source code lines. Together with the test bench files for the verification of the single modules we put up a bit more than 10,000 lines of VHDL source code.

The single function blocks of the design and the complete model were simulated on a functional base with *ModelSim* of *Model Technology*.

From the hardware description in VHDL one can generate the configuration data for a programmable logistic part (*Synthesis*). The 90 track combiner units (TMUs) that form

[4] The LTUs uses the y-coordinate at the drift chamber outside as an intercept on an axis of a track segment. It is more exact than the coordinate at the inside of the chamber which was calculated with its increase. Contrary to this in the track segment files the y-coordinate on the chamber inside shows to be more exact. The reason is not to be found in the physics of the detector but in the way the track segment file was generated. To compensate for these deviations we have used the coordinates of the chamber inside without emphasizing on this point in the following.

[5] "Very high-speed integrated circuit Hardware Description Language"

the GTU (see paragraph **??**), shall be built from programmable logic units. Therefore the use of a large FPGAs per TMU is planned.

A FPGA is an integrated circuit which consists inside of many thousand identical programmable cells – so called*Logic cells* – which are placed in a matrix on a chip. In the spaces between there are the *Routing Channels*, where the connection of the single cells takes place by the use of very fine structured connecting segments. The connection of the single segments is through programmable switches. Each of the logic cells typically consists of a programmable value table which has an aberration of four entrances on every exit plus one D-Flip-Flop. Furthermore there are connections to clock networks and an additional logic for special applications. Besides the logic cells most of the FPGAs have additional system units which can for instance be built in as memory units. After the FPGAs is switched on the programming of the SRAM-memory elements from the logic cell is required and wirings from an external memory with configuration data.

In order to generate the configuration data for a FPGA the VHDL system description is converted into an elementary logic description by a translation program. This elementary description is translated into a network list for the FPGA which contains the logic cells and its connections among each other. When finally the cells are placed and wired one can generate a configuration data stream for the FPGA.

By the appropriately chosen programs the steps run almost automatically. For the transformation into net lists of various FPGAs we used *LeonardoSpectrum* of *Mentor Graphics*. With *Quartus II* of *Altera* the cells of this producer where placed and wired.

The contents of the hardware description is shown in the chapters 4 and **??**. Chapter 6 summarizes the results of the Root simulation as well as the results of the functional simulation and the synthesis of the hardware description.

3 Interface to the GTU

The simulation of GTU which was done within the framework of this project includes also parts of the computations in the LTUs plus the data transfer to GTU. In the first paragraph the interface between LTU and GTU is specified. In the second paragraph the computations designed for the LTUs are described as they are the basis for the simulation. Here our point of emphasis is to carry out efficient computations by the CPUs.

3.1 Interface between LTUs and GTU

The computations in the LTUs are carried out by micro processors which are freely programmable and which can carry out through an assembler program any kind of computation within the time frame of the processing time available. By the configuration network of the detector it is possible to assign an own independently working program to each of the 250,000 CPUs. Furthermore there are numeric constants available within the CPU register which can also be separately configured for each CPU.

The program in the trigger selection mode must choose the sums which were pre-calculated during the drift time by the pre-processor and compute from these the parameter axis section and the increase of the balance even. Afterwards these parameters must be corrected by a certain effect (see paragraph 3.2). From the gradient of the track segment a first estimation of the transverse momentum of the particle is derived. For track segments with an estimated transverse momentum above a given threshold value of $p_{\mathrm{t,\,min}}^{\mathrm{LTU}} = 2.3\,\mathrm{GeV/c}$ the parameters will be put together to a certain data word which is transferred to the net intersection point. Dr. V. Angelov generated a first version of such a program. It was adjusted according to the requirements of this chapter.

It is necessary for the specification of GTU to define the responsibilities of the computations between the LTUs and GTU and to fix the kind of the parameterization and the format of the data word which shall be transferred.

The transfer is done under two aspects. At first there should be as much computing as possible decentralized in the LTUs to reduce the complexity of the GTU. The CPUs in the LTUs have processing units for the basic computing which can be used to discharge the GTUs of computing which must be generated for all of the track segments. The CPUs are clocked about three times faster than the GTU basing on FPGA technology which might result in time saving. On the other hand is the data word which shall be transferred per track segment fixed on a width of 32 bit. All the data which are important for the global track reconstruction shall be coded with the exactness required.

In the GTU the following 3 sizes are used for the connection of track segments:

- the z-coordinate which is projected in direction to the point of collision onto the middle plane of the pile of the pile
- the y-coordinate which is projected in direction to the point of collision onto the middle plane of the pile of the pile
- the angle of the track segment to the vertex direction[1]

For the reconstruction of the transverse momentum and the kind of particle the following magnitudes are used:

- the y-coordinate at the outer surface of the chamber which was not projected
- the probability measured from the charge for one electron, a pion respectively

In order to find the ideal transfer form and exactness for the data one must localize the computing required in the LTUs, in the GTU respectively.

3.1.1 Splitting the Computing

After the linear regressions are finished one gets two values in the LTUs: The y-coordinate of the intersection point of the even with the chamber surface in units of the pad width relative to the position of the LTU and the increase $\frac{\mathrm{d}y}{\mathrm{d}x}$ of the even. Instead of the in-

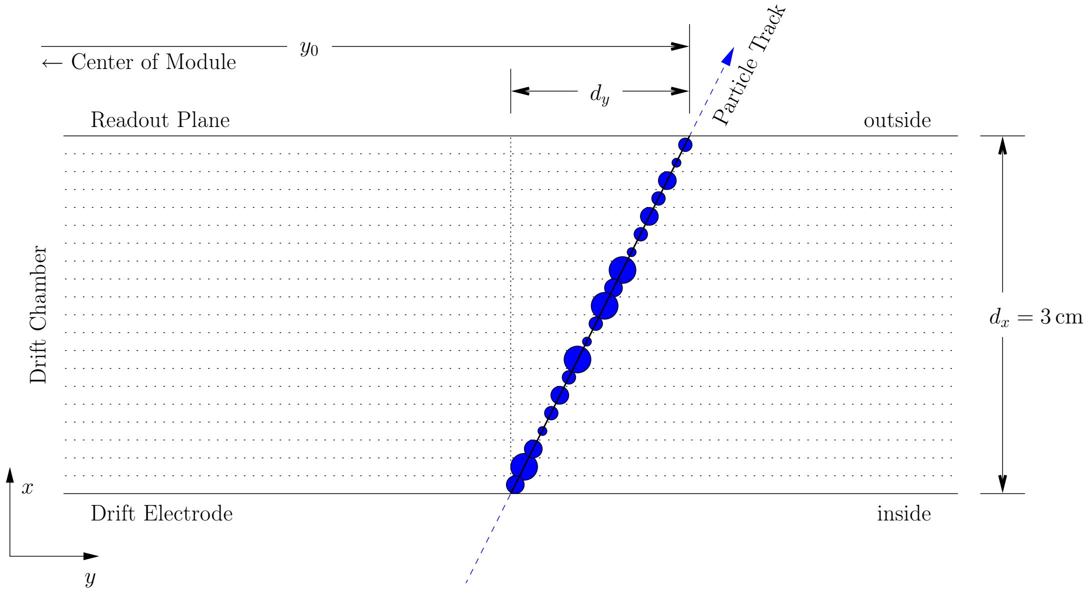

Figure 3.1: definition of y-position and deflection within the drift chamber.

[1] *vertex direction* we call the direction from one measuring point to the primary point of interaction in the center of the detector. It is the direction which would have the trace of a particle with unlimited large transverse momentum.

crease $\frac{\mathrm{d}y}{\mathrm{d}x} = \frac{d_y}{d_x}$ in the following we look at the *deflection* d_y within the drift chamber (see figure 3.1). This is equivalent since the thickness $d_x = 3\,\mathrm{cm}$ if the drift chamber is constant.

For every single track segment a total of the following computing is required:

1. correction of the deflection by the Lorentz angle (see paragraph 3.2.1)
2. correction of the deflection by the influence "tilted Pads" (see paragraph 3.2.2)
3. comparison of the angles between particle track and vertex direction with a threshold (see paragraph 3.2.3)
4. transformation of the charge measured into a probability for an electron rsp. a pion
5. translation of the local y values into global detector coordinates
6. projection of the track segment on a middle plane (for a union with other track segments)
7. computing of the angle between particle trace and vertex direction

In order to reject uninteresting tracks already on LTU level the matters 1–3 must be carried out directly within the LTUs. The remaining calculations can principally be localized as well in the LTUs as also in the GTU.

The transformation process of the charge measured into a probability for an electron respectively pion (number 4) is mainly a check up on a constant value table in which the values are filed standardized according to the distribution of the probability (figure **??**). The operation should be carried out in the LTUs as the image is strongly non-linear and otherwise the resolution of the result would be distributed across the value range strongly inhomogeneous – if there was a constant bit width of the value transferred –.

figure **??** shows that the deviation between the medium pulse frequency for electrons and pions is larger in the second half of the drift time. Therefore it might be useful to split the total charges into various parts of the drift time. At present we test the different possibilities. If the operations are implemented into the LTUs and if there is only the resulting probability transmitted changes of the procedure are possible without the necessity to increase the transferred bit width. For the value table there is configurable memory space for the CPUs.

It is not of any importance for the GTU how the single channels are allocated to the LTUs however it is the absolute position of the track in relation to a stack of modules significant. The transformation into global coordinates[2] (number 5) should happen within the LTUs as this releases the GTU from this duty without big disadvantage. The y-position is transmitted in absolute length units instead of the position relative to the module width or the numbers of the MCMs in the pad rows and the position relative to the channels of the MCMs. As the inner detector modules are less wide than the ones on the surface (95.6 cm width in plane 0 compared to 117.8 cm in plane 5) the range where the y-coordinate occurs

[2] The word *global coordinate* is not related to the complete detector but always to a stack of modules.

is smaller and the full range of values is at the same kind of scaling during the transmission process not totally used. The losses in precision amount to only 18.8 % of the value range – which is 0.30 bit. even at the inner plane.

In the GTU the projected y-coordinates are needed just as well as the non-projected. The non-projected values are needed for the more precise reconstruction. As the space within the data word is not sufficient for the (redundant) transmission of both values one might choose to transmit the non-projected position instead and to compute the projection (number 6) in the GTU.

Now we need in the GTU the increase of the track segment for the projection of the y-coordinate plus the angle opposite the vertex direction is used as a criteria for the assembling of the track segments which can be calculated by the increase (and y-coordinate). As the data word again has not enough space for the transmission of both values the computing of the angle (number 7) is also done in the GTU.

3.1.2 Contents of the Data Words Transmitted

The kind of values which are transmitted the data words results from the splitting of the computing. Each of the data words contains the following values:

- The y-position (in meters) of the point of intersection of the track with the chamber outside as a pre-marked distance from the chamber center.
- The absolute deflection of the track in the chamber i. e. the difference (in meters) between the y-position at the chamber outside and on the chamber inside.
- The z-position (without unit of measure) as a number of the pad row (0–15)
- The probability (between 0.0 and 1.0) on the basis of the charge measured of the track being from an electron

An important decision is the choice of exactitude which is used for the transmission of the values. Under the conditions given this choice seems not to be in [ALI01, p. 111] optimal.

For the transmission floating point numbers seem to be inefficient. In order to use the full range of values the values are multiplied with a resolution which is assessed for each of the sizes and are round off for the conversion into integral number.

The exactitude of the y-positions is decisive for the p_t resolution during the reconstruction. Therefore a very high resolution is desired. The maximum exactitude of the detector is at $\delta y_\mathrm{detector} \approx 400\,\mu\mathrm{m}$. One could however choose a transmission resolution of $1/200\,\mu\mathrm{m}$ in order to avoid a quantification error of the transmission. The values of the y-position are in a range of [3] $-64.32\,\mathrm{cm} \leq y \leq 64.32\,\mathrm{cm}$. For the transmission of the value one needs

[3] The difference to the width of the outside of the detector module quoted above results from the neglect of the distance between the modules in order to keep the transmission format free from the minimal changes at the chamber geometry..

name	Symbol	resolution	value range	bits
part of the axis	y	$1/160\,\mu\text{m}$	$-643.2\,\text{mm}$ .. $643.2\,\text{mm}$	13
deflection	d_y	$1/140\,\mu\text{m}$	$-8.8\,\text{mm}$ .. $8.8\,\text{mm}$.	7
pad row	z	1	0 .. 15	4
e^--probability	p	$1/0.39\,\%$	0 .. 1	8

Table 3.1: Contents of the data word which will be transmitted to the GTU for a track segment.

13 bit. In order to optimally use the value range during the transmission with 13 bit we choose a resolution of $1/160\,\mu\text{m}$.

Each of the optical connection transmits the track data of half a module at which the modules are divided in y direction. Because of this the real value range of both lines of a module is different from one another. However in the middle of the chamber both the value ranges intersect which causes the sign of y. to be not implicitly clear. To save re-computing of the GTU not each of the value ranges of the different halves is singly adjusted.

For $p_{\text{t, min}}^{\text{LTU}} = 2.3\,\text{GeV/c}$ and $B = 0.4\,\text{T}$ the maximum deflection angle $\alpha_{\text{max}} = 6.38°$ (see paragraph 3.2.3). Together with the maximum angle between vertex direction and chamber normal of 10.0° a maximum value range of $-0.881\,\text{cm} \leq d_y \leq 0.881\,\text{cm}$ is derived. The deflection d_y is not used for the reconstruction but only for the projection and the window criterion during the comparison of the track segments. If one would try to quote the value for $\delta\alpha$ then exactly to 10 % of the threshold (see paragraph 4.1.2) one would need an exactitude of $157\,\mu\text{m}$ – equal to 7 bit. The exactitude of the detector however is only about $\delta d_y \approx 400\,\mu\text{m}$. For the purpose of using the total value range optimally during the transmission with 7 bit a resolution of $1/140\,\mu\text{m}$ was chosen.

There are 4 bit necessary for the number of the pad row. From the 32 bit there are 8 bit left for the coding of the electron probability of a track segment into 256 levels. Table 3.1 summarizes the results.

3.2 Computing in the LTUs

From the adjustment of an even to the ADC values one derives the point of intersection y_0 with the chamber outside and the deflection in y direction $d_{y,\,\text{raw}}$. The computing which are meant to be carried out subsequently in the LTUs is shown in the following subsections. It is especially shown how the operations can be implemented efficiently.

3.2.1 Correction of the Deflection by the Lorentz Angle

The homogeneous $\vec{B}$-field, which penetrates the detector in longitudinal direction influences also the drift of the ionization electrons in the chamber.

In the presence of the $\vec{B}$-field the drift direction does not run along the electrical field but the electrons are deflected in y direction. During this period the drift velocity of the electrons is minimized. Figure 4.2 3.2 illustrates the effect. The drift angle Ψ_L is called *Lorentz angle*. If $\vec{E}$ and $\vec{B}$ are vertical to each other it is given by

$$\tan(\Psi_\mathrm{L}) = \frac{\mathrm{e}\tau B}{m}$$

(see [Kle92]). Then τ is the medium time between two collisions with gas molecules. The Lorentz angle is by τ dependent on the composition of the gas mix and the strength of the $\vec{E}$-field. In the experiment is probably amounts to $\Psi_\mathrm{L} = 7°$.

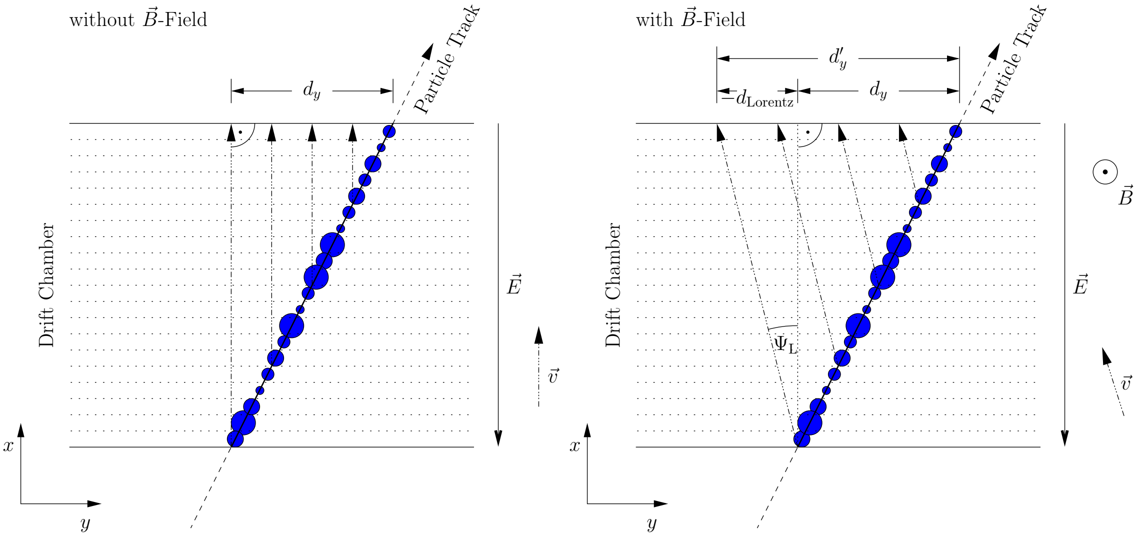

Figure 3.2: drift direction of the electrons in the chamber without (left) and with (right) $\vec{B}$-field. The additional field changes the direction around the Lorentz angle Ψ_L and hence the deflection measured by the distance d_Lorentz.

Whereas the y-position measured on the outside of the chamber does not influence the effect the deflection d_y measured must be corrected. A charge cluster from the chamber interior reaches the pad row on the outside permuted by

$$d_\mathrm{Lorentz} = -\tan(\Psi_\mathrm{L}) \cdot d_x = -3.68\,\mathrm{mm} \tag{3.1}$$

. A charge cluster on the chamber outside is not deflected. In order to correct the effect one must add the constant value d_Lorentz within the LTUs to each of the deflections measured independent of their value.

$$d'_y = d_{y,\,\mathrm{raw}} + d_\mathrm{Lorentz}$$

This means the Lorentz angle correction is a correction of the deflection respectively the gradient of the track segment.

For the correction of the deflection it is unimportant that by the Lorentz power the drift velocity of the electrons is minimized and the length of the drift distance is enlarged, as long as the drift time is chosen by the LTUs equivalently respectively the gradient is scaled.

3.2.2 Correction of the Deflection by the Influence of the "tilted Pads"

The cathode pads at the chamber outside are not exactly square but purposely off-centered slightly to the parallelograms opposed on the various detector planes. ("tilted Pads"). The tilting angle is $\beta_{\text{tilt}} = 2°$ (see figure 3.3). By this grouping it is possible for the later data analysis to improve the resolution in z direction for tracks that run through by combining various planes. Without the "tilted Pads" the tracking in z direction could not be determined more exact than on the width of a pad row ($d \approx 10\,\text{cm}$) at ideal projective geometry.

At the online analysis (trigger calculation) "tilted Pads" are of no use because of the short computing time available. However it is nevertheless important to recognize the geometry of the pads at the computing as it counterfeits the results of the measuring y-position and the deflection.

The counterfeit of the y-position occurs because charges within the limits of two pads at equivalent real y-coordinate depending on the z-position are projected onto the one or the other pad. The error in y amount to maximal $\pm\frac{d}{2} \cdot \tan(\beta_{\text{tilt}}) \approx 1.8\,\text{mm}$ and can not be corrected on LTU plane as the exact z-position is unknown.

As the y-position measured with "tilted pads" is not independent any more from the z-position also the gradient $\frac{\mathrm{d}y}{\mathrm{d}x}$ does change. With the help of the vertex presumption this effect can be corrected in the LTU [Vul03].

Instead of the real y-coordinate one measures with "tilted Pads" the value

$$y' = y \pm (z - z_{\text{row}}) \cdot \tan(\beta_{\text{tilt}}). \tag{3.2}$$

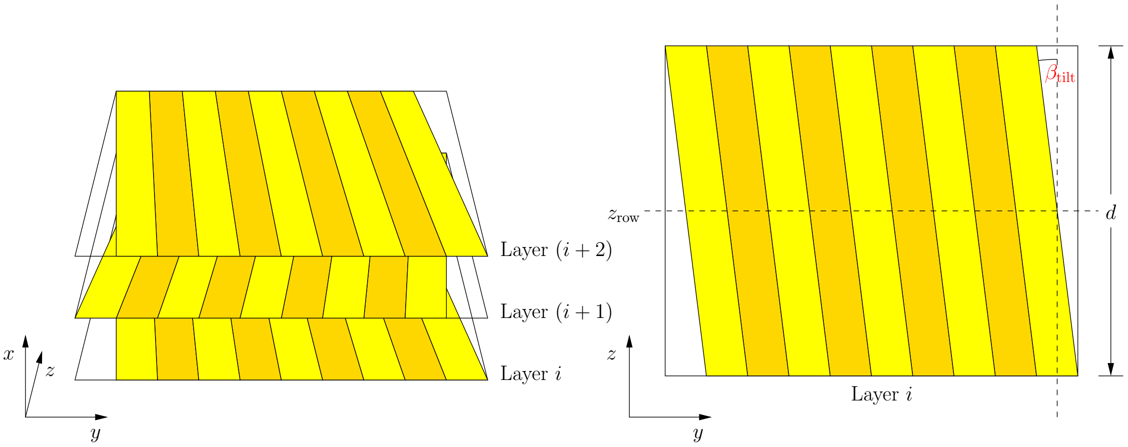

Figure 3.3: cathode pads for three pad rows which are super imposed. For the increase of the spatial resolution in z direction the pads are alternating tilted by an angle of ±2° in the various positions ("tilted Pads").

Here z_{row} is the z-coordinate in the middle of the pad row. Instead of $d_y = y_{\text{outer}} - y_{\text{inner}}$ one measures accordingly

$$\begin{aligned} d'_y &= y'_{\text{outer}} - y'_{\text{inner}} \\ &= d_y \pm (z_{\text{outer}} - z_{\text{inner}}) \cdot \tan(\beta_{\text{tilt}}) \\ &= d_y \pm \Delta z \cdot \tan(\beta_{\text{tilt}}). \end{aligned} \tag{3.3}$$

By using the primary vertex presumption we can determine as is shown in figure 3.4 Δz in good approximation from the position of the pad row as

$$\Delta z = \frac{z_{\text{row}} \cdot d_x}{x_0}$$

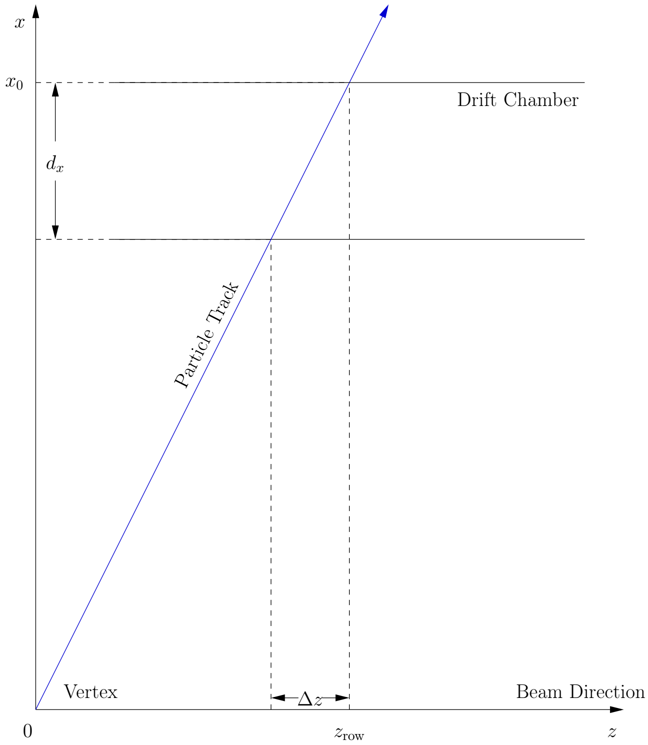

Figure 3.4: In the x-z-plane the particle tracks run straight. For particles which come directly from the primary vertex the deflection Δz in the chamber can be calculated directly from the coordinate x_0, z_{row} of the pad row.

For every pad row of the detector the value is

$$\begin{aligned} d_{\text{Pad Correction}} &= \pm \frac{z_{\text{row}} \cdot d_x}{x_0} \cdot \tan(\beta_{\text{tilt}}) \\ &= \pm 1.048\,\text{mm} \cdot \frac{z_{\text{row}}}{x_0} \end{aligned} \tag{3.4}$$

with

$$\begin{aligned} x_0, z_{\mathrm{row}} &: \text{coordinate of the pad row} \\ d_x &: \text{thickness of a chamber } (3.0\,\mathrm{cm}) \\ \beta_{\mathrm{tilt}} &: \text{angle of the pads against the } z \text{ axis} \end{aligned}$$

pre-calculated and saved as a sum together with d_{Lorentz} in the LTUs of this row. During the trigger selection the LTUs add as a correction always the constant value to the deflection measured. Together they result in two (locally constant) correction terms:

$$d_y = d_{y,\,\mathrm{raw}} + d_{\mathrm{Lorentz}} + d_{\mathrm{Pad\ Correction}} \tag{3.5}$$

That means both the corrections base on the addition respectively subtraction of an adequate common constant.

3.2.3

Only track segments with sufficiently big transverse momentum respectively sufficiently low deflection against the original direction shall be transmitted. Therefore we need not calculate the angle against the original direction right away. Instead the maximal deflection angle α_{max} is converted into a maximal deflection $d_{y,\,\mathrm{min}}$ rsp. $d_{y,\,\mathrm{max}}$.

Connection between Deflection Angle and Transverse Momentum

Figure 3.5 shows at first the connection between the angle α of the track segment against the vertex direction and the radius r of the circular particle orbit. in the upper right-angled triangle is

$$r = \frac{d/2}{\sin(\alpha)} \tag{3.6}$$

with the distance between the track segment and the beam $d = \sqrt{x_{\mathrm{m}}^2 + y_{\mathrm{m}}^2}$.

The transverse momentum p_{t} of the particle results if the magnetic induction is known B from the radius of the particle orbit in the x-y-plane as[4]

$$p_{\mathrm{t}} = \mathrm{e} \cdot r \cdot B = 0.30\,\mathrm{GeV/c} \cdot \frac{r}{\mathrm{m}} \cdot \frac{B}{\mathrm{T}}. \tag{3.7}$$

with the elementary charge e written $\mathrm{e} = 0.30\,\frac{\mathrm{GeV/c}}{\mathrm{m \cdot T}}$ which is adequate in this connection.

The maximum deflection angle α_{max} hence is calculated from the minimal transverse momentum $p_{\mathrm{t,\,min}}^{\mathrm{LTU}}$ by

$$\alpha_{\mathrm{max}} = \arcsin\left(\frac{\sqrt{x_{\mathrm{m}}^2 + y_{\mathrm{m}}^2} \cdot \mathrm{e}B}{2 \cdot p_{\mathrm{t,\,min}}^{\mathrm{LTU}}}\right). \tag{3.8}$$

In the simulation the value $p_{\mathrm{t,\,min}}^{\mathrm{LTU}} = 2.3\,\mathrm{GeV/c}$ is used that results in a maximal deflection angle $\alpha_{\mathrm{max}} = 5.47°$. for example for a pad laying on the outside.

[4] see [Gre00]. The connection is obvious if the Lorentz power on a single charged free particle being vertical to the $\vec{B}$-field is equated to the centrifugal power.

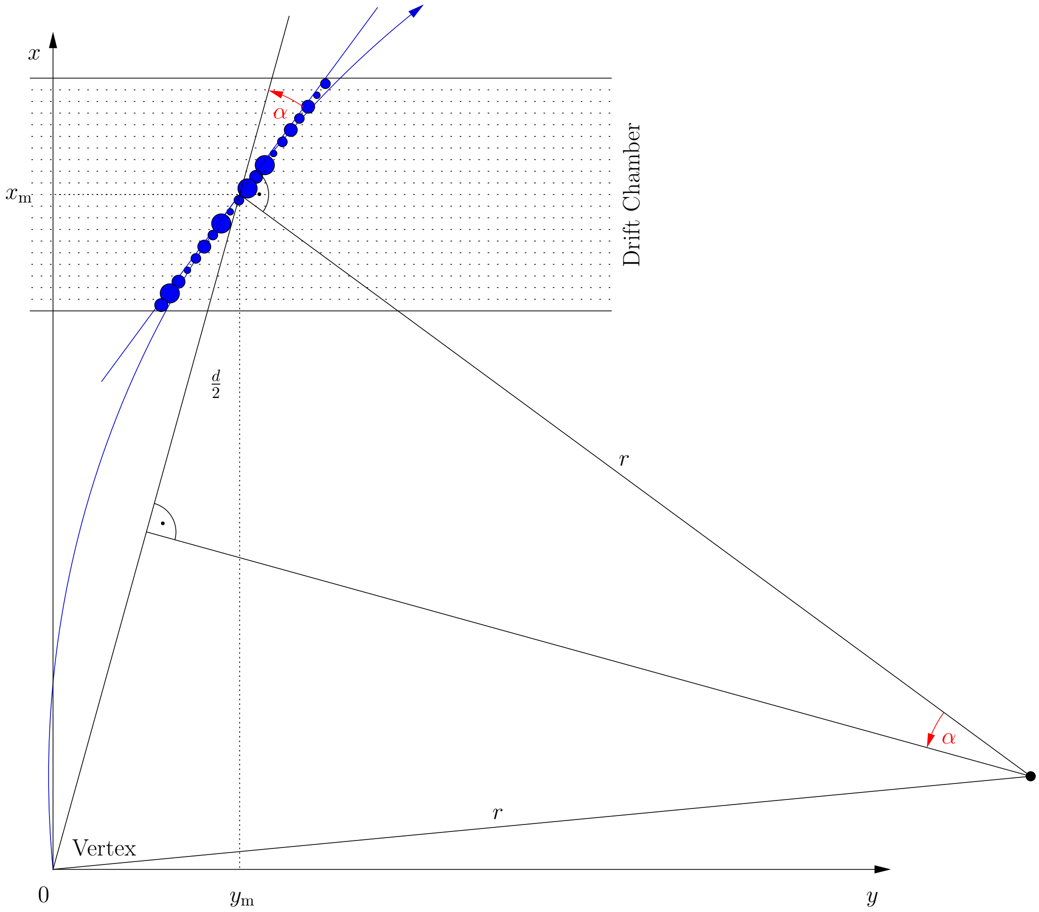

Figure 3.5: Scheme projection of the circular particle orbit with relation to a detector module. From the angle α of the track segment against the vertex direction one can figure the radius r of the particle orbit.

Connection Between Deflection and Deflection Angle

In order to calculate from the deflection d_y the angle α to the vertex direction one need the x-position of the chamber and the y-position of the track segment (see figure 3.6). This means

$$\tan(\varphi_\text{Vertex}) = \frac{y_0}{x_0} \qquad \tan(\varphi_\text{Track}) = \frac{d_y}{d_x}$$
$$\alpha = \varphi_\text{Track} - \varphi_\text{Vertex} = \arctan\left(\frac{d_y}{d_x}\right) - \arctan\left(\frac{y_0}{x_0}\right). \tag{3.9}$$

The sizes x_0 and d_x are constant for every chamber. y_0 is part by part seen constant within the framework of the resolution necessary. Then with the equations (3.6), (3.7) and (3.9) the transverse momentum p_t can be written as an uniquely reversible function of the deflection d_y. Instead of the condition $|p_\mathrm{t}| \geq p_\mathrm{t,\,min}^\mathrm{LTU}$ one can at thresholds $d_{y,\,\mathrm{min}}, d_{y,\,\mathrm{max}}$

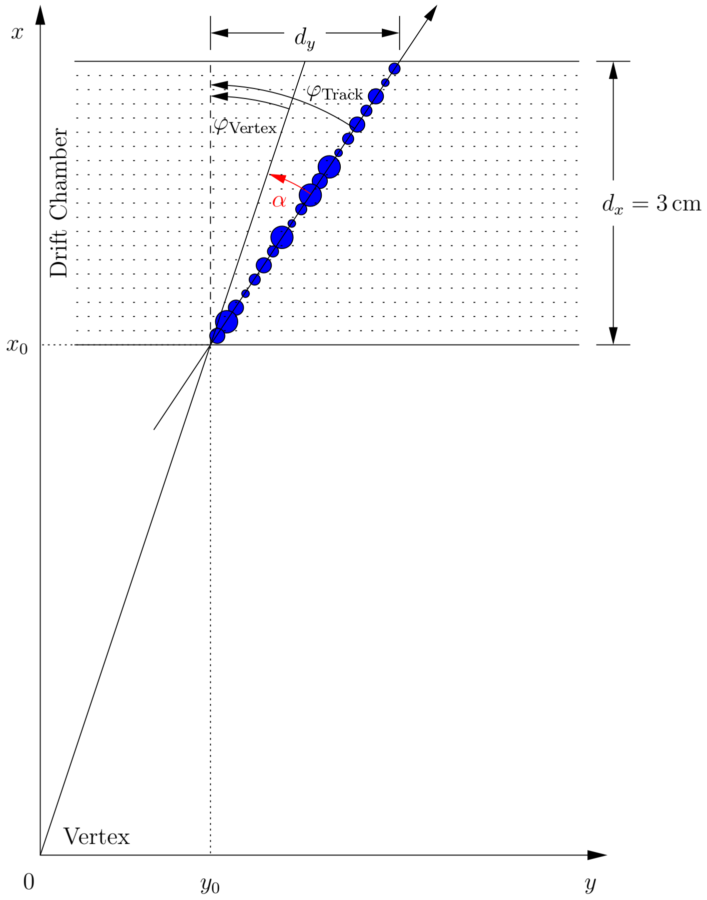

Figure 3.6: scheme projection of the particle orbit through a detector module. One can figure out the angle α of the track segment to the vertex direction from the deflection d_y.

chosen accordingly proof on the condition

$$d_{y,\,\text{min}} \leq d_y \leq d_{y,\,\text{max}}$$

(see figure 3.7).

The minimal and maximal deflection $d_{y,\,\text{min}}$ respectively $d_{y,\,\text{max}}$ will hence result from the maximum deflection angle α_{max} by

$$\begin{aligned} d_{y,\,\text{min}} &= d_x \cdot \tan\left(\arctan\left(\frac{y_0}{x_0}\right) - \alpha_{\text{max}}\right) \\ d_{y,\,\text{max}} &= d_x \cdot \tan\left(\arctan\left(\frac{y_0}{x_0}\right) + \alpha_{\text{max}}\right). \end{aligned} \tag{3.10}$$

Each of the threshold values $d_{y,\,\text{min}}, d_{y,\,\text{max}}$ is for every LTU pre-calculated for their position and some different y_0 values in its y range according to the equations (3.8) and (3.10) plus set up as a value table in the memory of the LTU. During the processing of the track

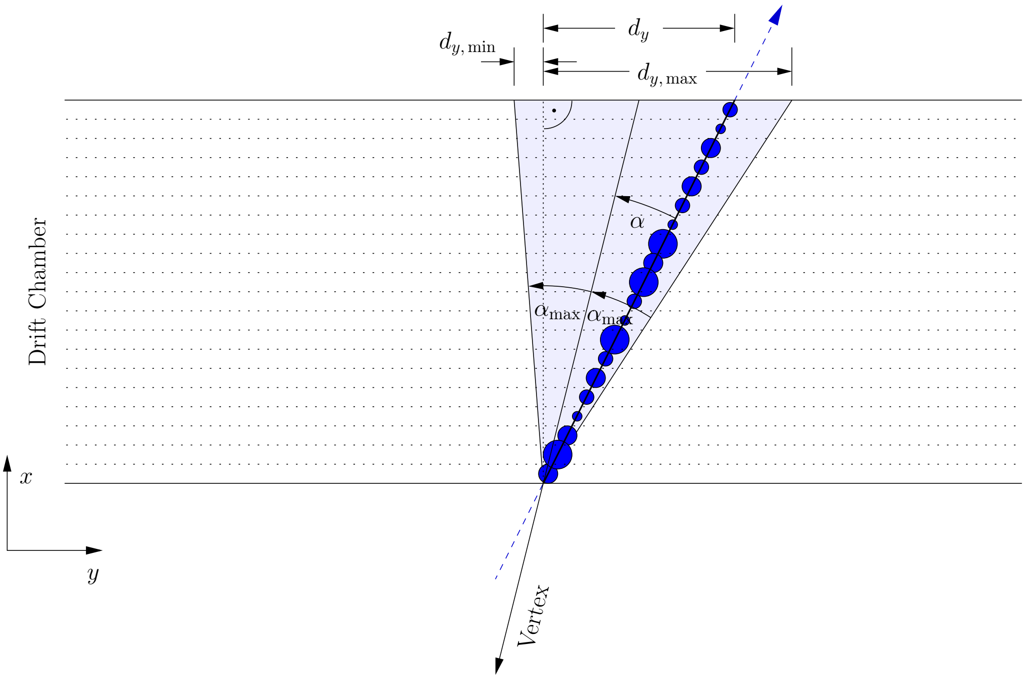

Figure 3.7: Scheme projection of the threshold values for the minimal respectively maximal deflection. instead of α with α_{max} one can compare directly d_y with $d_{y,\,\mathrm{min}}$ and $d_{y,\,\mathrm{max}}$.

segment the adequate threshold values are selected out by the present y-position from the table. The comparisons with the deflection d_y measured show whether the track segment is transmitted to the selection network or not.

Both corrections and the selection procedure can very efficiently be implemented in the CPUs of the LTUs as the computing consist of only three additions respectively comparisons. The description of the in paragraph 3.1.1 to the GTU shifted calculations is part of the following chapter.

4 Combining of Track segments into tracks

In this and the following chapters computing procedures are developed for various functional parts of the GTU and then translated into a hardware description. The procedures were chosen under an efficiency aspect for the realization of the hardware. Here the parallel procedure is an decisive criterion because of the low amount of computing time available. Sequential respectively iterative algorithms should be avoided. Especially time-consuming methods like the Kalman filtering are insufficient for the realization of the trigger.

Contents of this chapter is the combination of track segments of the detector to particle tracks. A basic procedure the applicability of which is visible in the Root simulation will be introduced in the first paragraph. Starting with the simulation a hardware transformation as fast and efficient as possible is developed in the following paragraphs. There the basic computing procedure is varied and optimized.

4.1 Basics

We need a criterion which delivers several track segments on the basis of the data that can easily be tested whether they belong to the same track or not.

Several track segments of the same particle have a similar angle α to the vertex direction. Their z-positions are on the x-z-plane on an even with the point of collision and their y-positions are on the x-y-plane approximately on an even the slope $\frac{\Delta y}{\Delta x}$ of which is equal to the one of the single track segments.

4.1.1 Projection onto the Middle Plane

A simple possibility to test the conditions is to project the track segments on a plane thought in common vertical to the x axis. Here we use in y direction for the projection the slope of the track segment, in z direction the vertex direction is projected. In order to keep the error rate of the position projected as small as possible and to keep the track segments from all the planes as similar as can be we use the middle plane of a module stack as a projection plane. Figure 4.1 shows the projection in the x-y-plane.

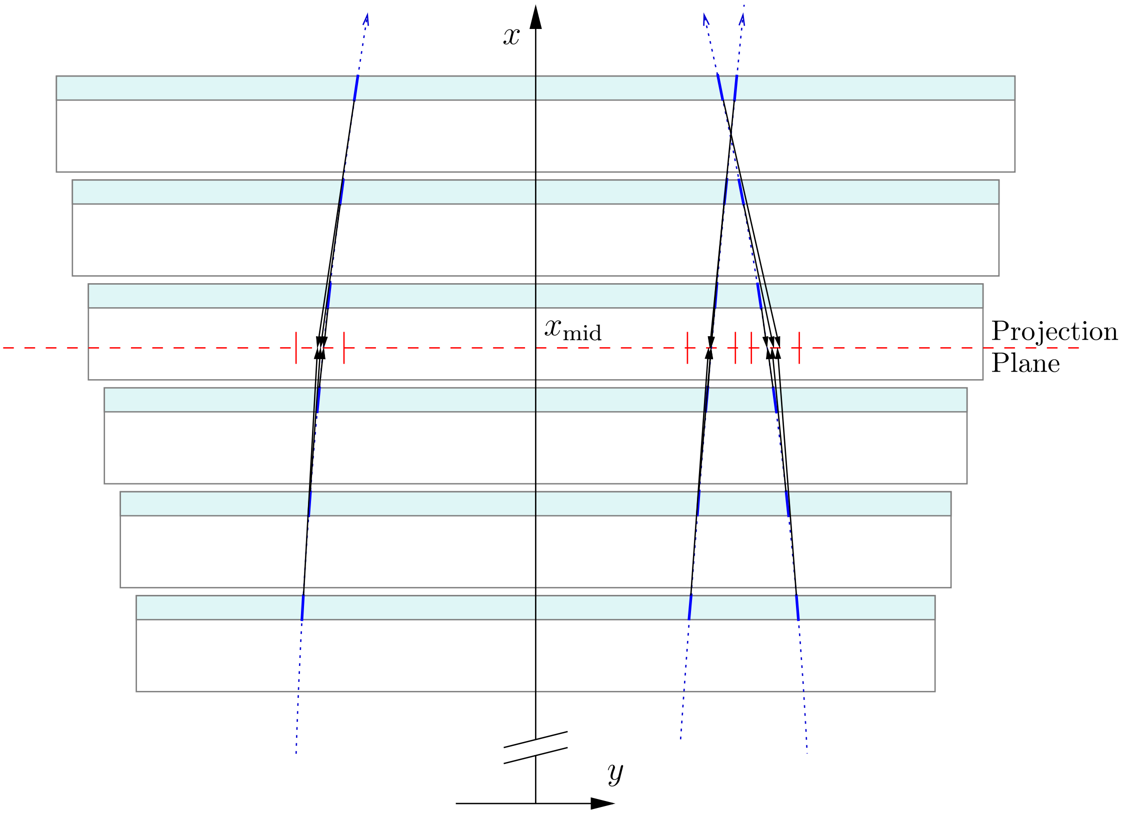

Figure 4.1: The track segments are projected on an imagined middle plane.

Computing of each of the Track Segments

For each track segment the following reference values are computed:

1. The y-coordinate projected in direction of the collision point

$$y_{\text{proj}} = y + d_y \cdot d_i \tag{4.1}$$

   with the constant:

$$d_i = x_i - x_{\text{mid}}$$

$$\begin{aligned} x_i &: x\text{-coordinate of the chamber outside in plane } i \\ x_{\text{mid}} &: \text{medium radius of the detector } (3.334\,\text{m}) \\ i &: \text{original plane of the track segment} \end{aligned}$$

2. The z-coordinate projected on the stack center plane into direction of the collision point

$$z_{\text{proj}} = z \cdot \frac{x_{\text{mid}}}{x_i} \qquad \text{(after the beam theorem)}$$

3. The deflection angle α of the track segment against the vertex direction. Instead

$$\alpha = \arctan\left(\frac{d_y}{d_x}\right) - \arctan\left(\frac{y}{x_i}\right)$$

one can approximately use.

$$\alpha = \frac{d_y}{d_x} - \frac{y}{x_i} \qquad (4.2)$$

As the values for the argument of the arc tangent are very small we can neglect the error here.[1] d_x is the thickness of a chamber (3.0 cm).

4.1.2 Window Criterion

A set up of track segments is characterized by the fact that their projections in the common plane are closely together and in the angle α are almost identical. Track segments which are only little different in the three values calculated are for this reason connected to one candidate. The limit values which are used for this reason in the simulation are summarized in table 4.1 [2] The GTU presumes a particle track to be found if there are at least four track segments from various planes within the three dimensional window.

size	max. divergence
y_{proj}	11.625 mm
z_{proj}	100 mm
α	0.05 rad = 2.86°

Table 4.1: The numerical values for the GTU-window criterion.

4.1.3 Hardware Realization

At a processing time of 1.4 µs and a desired tack rate of 40 MHz there is a total of 56 calculation frequencies about half of which are used for the connection of the track segments. If one wanted to compare each of the up to 40 track segments per module (see Figure 4.9) one after the other with all the track segments of the other planes one would have to carry out in every of the TMU up to $\frac{40\cdot5\cdot40\cdot6}{2} = 24,000$ coupled comparisons sequentially. This would consume about 1000 times the time we have.

One possibility is to put the track segments into a three dimensional histogram and to search for frequency points. The setting of the frequency point can be made directly during the data transfer. However the detection of the frequency points is very much work. We must add the number of track segments of the surrounding cells to the segment numbers

[1] At an exactness of 1/200 rad the simplified calculation results to in the simulation 81 % of the cases exactly the same result. In the other cases the divergence ±1 bit.

[2] The values are a little larger than the ones in [ALI01, p. 100]. They are chosen according to the “theoretical” simulation of Dr. B. Vulpescu.

in the histogram cells in order to find the track segments which are mapped onto the limit range of two cells. If we recognize in three dimensions also diagonal neighborhoods we must include 26 neighboring cells every time.

Therefore we use a different procedure within the framework of this project. The solution presented here uses the fact that three of the comparison values are very different from one another in relation to their exactitude and value range. Besides this it recognizes the succession of the track segments during the data transfer which allows to start the processing during the time of transmission.

The track segments are selected from every half of the modules of a fixed succession. During the transmission they are assorted according to the pad row numbers z in an increasing succession. Track segments with the same z are transmitted in successions of increasing y-coordinates. By the projection of the y-coordinate onto the middle plane in the GTU this sub selection is partly destroyed.

If the track segments are sorted in one of the variables it is not necessary to look at all the track segments for the control of the connection criterion in this variable but only the successive track segments.

The marking power of a variable for the track connection can be found if the window width is compared to the value range of the variable. The (projected) y-coordinate has by far the highest power of marking. The window width used amounts to only a $\frac{1}{110}$ of the value range. The deflection angle α has with about $\frac{1}{5}$ of the value range the smallest marking power. At the z-position the share is about $\frac{1}{16}$.

In order to use the sorting during the transmission we start with the comparison of the z values of the track segments. The pad row z as a curiosity has only 16 possible values. The explicit comparison of the (projected) z-coordinates hence can implicitly be realized by parallel recognition of all the possibilities. The procedure is shown in chapter 4.4.1.

As the (projected) y-coordinate has the highest marking power it is after all the main criterion for the track connection. If track segments are found which are sufficiently equal in z and y than the angle α of the track segments is compared.

4.2 Set Up of the Track Merging Units (TMUs)

The following paragraph describes the 90 identical TMUs the GTU consists of and the independent data of a detector stack are processed (compare paragraph **??**).

Figure 4.2 gives an overview of the set up of a TMU as it was developed within the frame of this work. In the input units the track segments of the LTUs of the six planes that are made parametric are received and further parameters are pre-calculated for the merging into tracks. The z-channel units finally control implicitly the equality of the track segments in z direction and assort it according to its y-coordinate. In the track finding units the track segments of the various planes are combined to tracks by comparing their y-coordinates and the deflection angle. The results of the track finding units are combined to a data stream

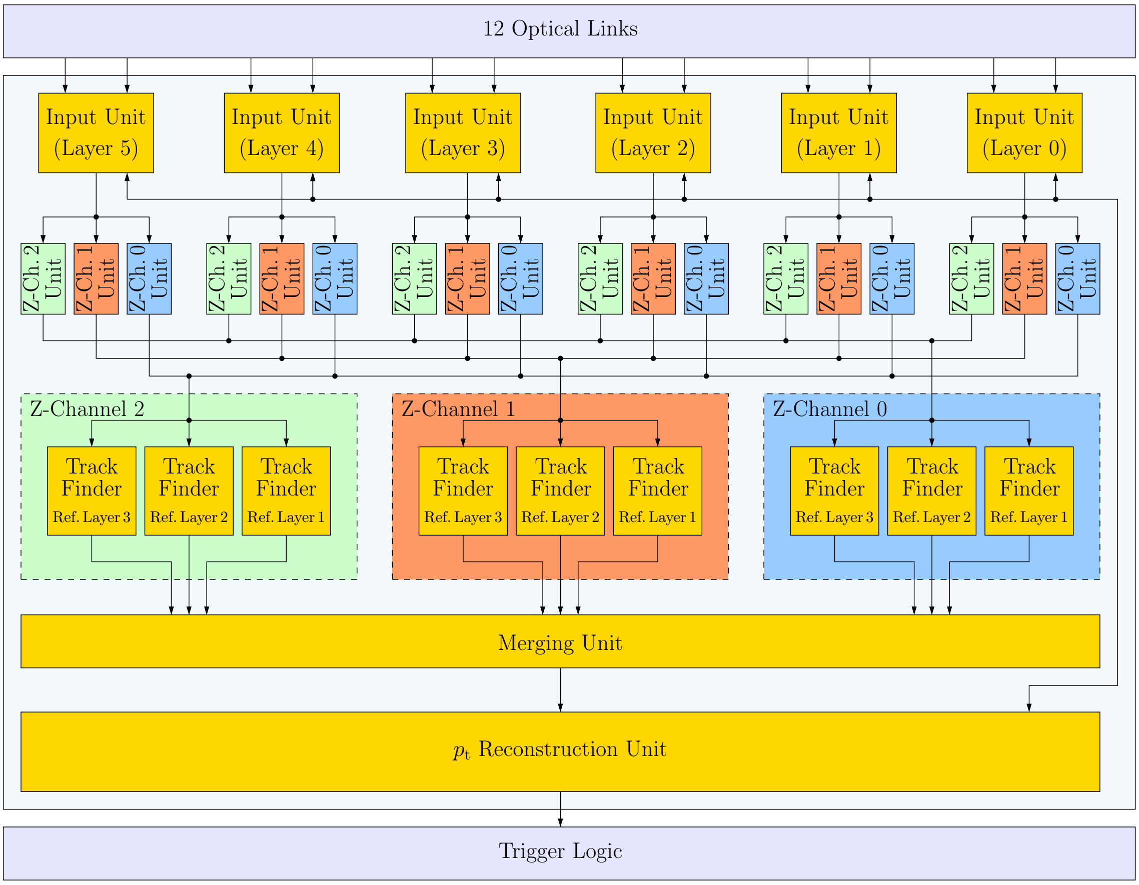

Figure 4.2: The set up of a TMU receives the track segment data from a module stack puts them together to tracks in several levels and reconstructs the transverse momentum of the particles generated. The data of the tracks found are transmitted into a simple GTU-internal trigger logic which receives the data of all of the 90 TMUs and merges them.

of tracks found. For every track the transverse momentum p_t of the particles generated is calculated within the reconstruction unit.

In the following paragraphs of this chapter the individual design units of the TMU are described. During the development of the units certain common basics were accepted:

- factor of the transmission time
  The processing of the track segment begins already during the transmission. It is our goal to be as far as possible with the computing of the present data.
- Parallelization
  Calculations which can be made at the same time independently from one another are normally not carried out sequentially but parallel by multiplication in the equivalent processing unit.
- low latency
  With all the design units special care was taken for a low latency. The unit keeps

data in the buffer only as long as necessary. If there are intermediate results they are transmitted to the next unit right away.

- no handshaking
  There is no handshaking between the individual units i. e. a unit can at any time transmit data to the next and is able itself to accept data in each clock cycle. If necessary this must be done with buffers. There is a control signal at each data signal which gives note per clock cycle whether a data word will be transmitted.
- bit-optimized signals
  The data signals which represent physical sizes are implemented as an integer number with minimal bit width by determining the resolution necessary for every signal and the value range. The numbers normally have a sign and are realized in two's complement presentation.
- generic design
  The bit widths and exactitude of all data signals are deduced from few constant parameters. If for instance the composition of the transmission data word shall be changed this is only necessary in one part of the design.

4.3 Input Unit

Each input unit combines the data of two optical receivers and calculates various parameters for the track reconstruction in the following unit. Values that are needed later are put into the storage and are transmitted in the form of their storage address. Figure 4.3 shows the input unit as a block system image.

4.3.1 Format of the Input Data

The data of the track segments are transmitted from the LTUs to the GTU as 32-bit words. The network intersection point handles each of the data words internally as 16-bit words where the lowest order value half-word is transmitted first. The end of the transmission is marked by the network intersection point by a 16 bit wide final word which is freely configurable. It must be defined in a way that if it is interpreted as the lower half of a data word it has an impossible value for the application. In Figure 4.4 the set up of the data word and the final word chosen are shown. The GTU in the final word tests only the position quoted.

The hardware implementation is according to table 3.1 calculated with integer numbers:

$$\mathtt{y} := \frac{y}{160\,\mu\mathrm{m}}, \qquad \mathtt{d} := \frac{d_y}{140\,\mu\mathrm{m}}. \tag{4.3}$$

In the following the individual components of the input unit are described.

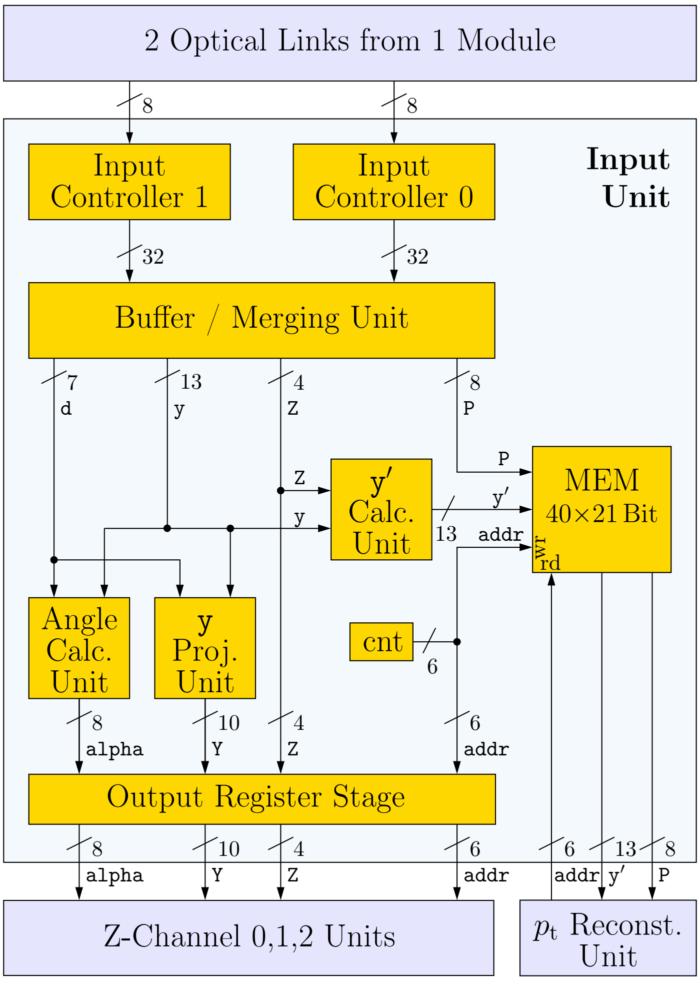

Figure 4.3: The set up of an input unit as a block system image. The input unit receives track segment data from a detector module and carries out the calculations which can be individually done for each segment.

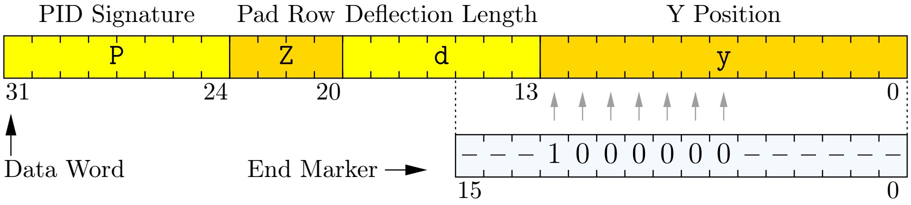

Figure 4.4: Composition of the 32-bit data words which are transmitted to the GTU. The final word contains only 16 bit and is characterized by the numbers quoted.

4.3.2 Input Controller Unit

The input controllers reassemble the individual bytes of the transmission network which arrive at 120 MHz and the double data rate *Double Data Rate*, DDR) to 32-bit words.

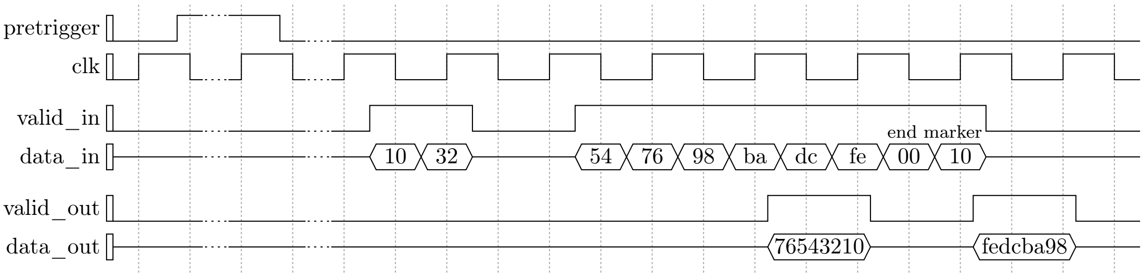

Figure 4.5: Example of the time behavior expected of a TMU of the DDR-input signals and the succession of the partial words. Here two track segments are transmitted. Between the two 8-bit data within a clock cycle the transmission may not be interrupted but between the 16-bit half words a random amount of pauses is allowed.

In Figure 4.5 the behavior of the input controller is shown for an example of transmission. By the `pretrigger` signal the control unit is set to receive data. Each time if 8-bit words are transmitted with common data and that are positioned to the decreasing and the subsequently increasing clock flank they are put together to 16-bit words. Between the 16-bit words interruptions are allowed. Valid data are shown by the signal`valid_in`. Two successive 16-bit words each combined to one 32-bit word which represents a track segment are forwarded to the next processing unit. By the 16-bit final word `0x1000` (see Figure 4.4) the end of the transmission is shown. Further data words are ignored by the input control unit.

4.3.3 Combination of the Data of the Two Input Units

In the following merging unit 32-bit words are combined to a data stream. The data are buffered in two FIFO-buffers[3]. After an ascending z-coordinate a data word is read out from each of the FIFOs and issued after being divided into its components.

The maximum amount of track segments per optical connection per event is important for the buffer sizes required. Figure 4.6 shows the distribution as a result of the simulation. Even at the highest amount of multiplicity very rarely there are more than 20 track segments per module half. This number is equivalent with the maximum fixed by the limitation of the transmission time.

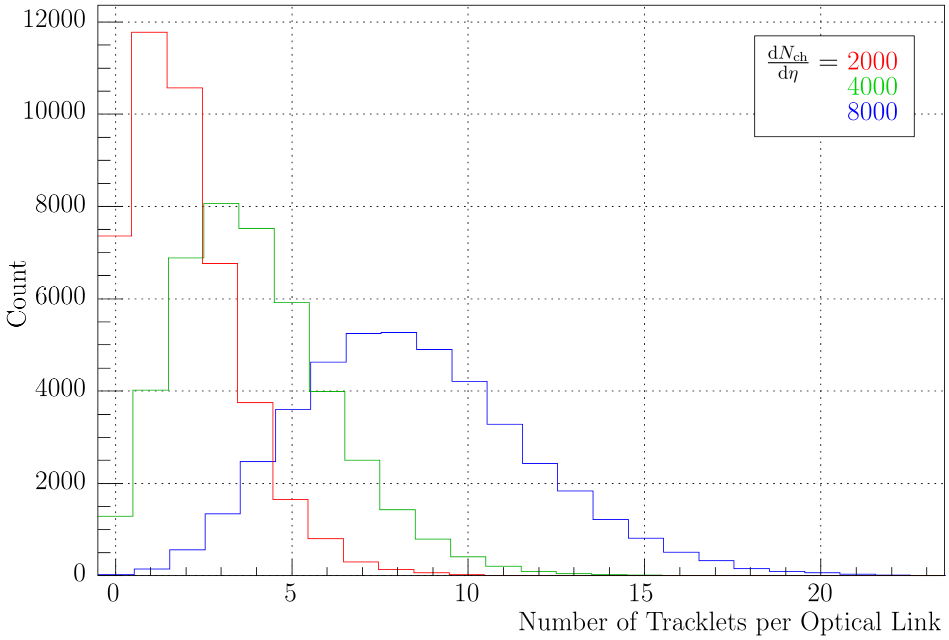

Figure 4.6: Distribution of the amount of track segments per optical connection for various multiplicity densities. The maximum amount of accepted track segments for example is in charge of the time which is necessary for the data transfer. A value of 20 is intended.

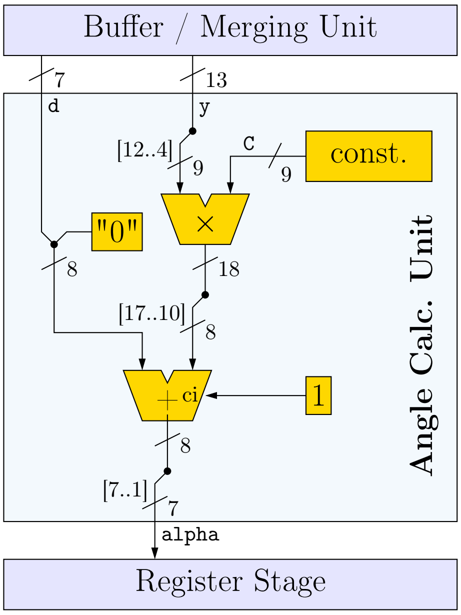

Figure 4.7: The set up of the computation unit for the deflection angle. By adding the y-coordinate which is according to the detector plane appropriately scaled the deflection of the track segment is in good approximation turned into the deflection angle against the vertex direction.

4.3.4 Calculation of the Deflection Angle

The angle calculation unit shown in Figure 4.7 recalculates the deflection d_y into the deflection angle α against the vertex direction. It implements equation (4.2) as[4]

$$\mathtt{alpha} = \left\lfloor \mathtt{d} + \frac{1}{2} \cdot \left\lfloor \left\lfloor \frac{\mathtt{y}}{2^4} \right\rfloor \cdot \frac{\mathtt{C}_i}{2^{10}} \right\rfloor + \frac{1}{2} \right\rfloor \approx \mathtt{d} + \frac{\mathtt{C}_i}{2^{15}} \cdot \mathtt{y}, \tag{4.4}$$

where the deflection $\mathtt{d}$ is not scaled to allow a calculation where only one addition and one multiplication with a constant are needed. The constant depending on the plane i is defined by

$$\mathtt{C}_i := -\frac{3\,\mathrm{cm}}{x_i} \cdot \frac{160\,\mu\mathrm{m}}{140\,\mu\mathrm{m}} \cdot 2^{15}. \tag{4.5}$$

The unit of the angle $\mathtt{alpha}$ hence is fixed by the relation $\frac{140\,\mu\mathrm{m}}{3\,\mathrm{cm}}$:

$$\mathtt{alpha} = \frac{\alpha}{0.004\bar{6}\,\mathrm{rad}} \tag{4.6}$$

4.3.5 Projection of the Y Coordinate

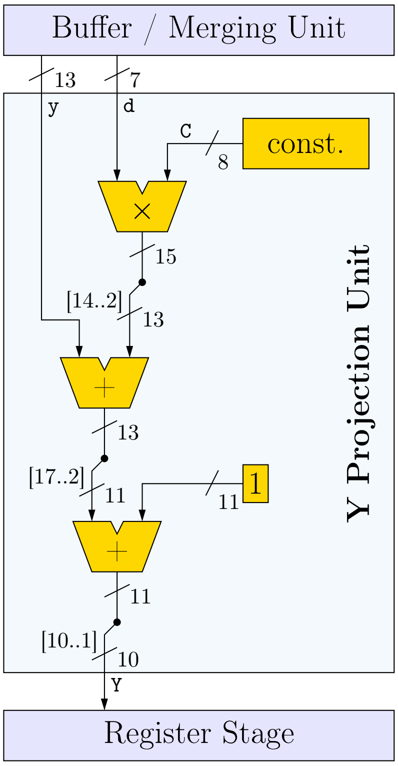

Figure 4.8: The set up of the calculation unit for the y-projection. To the y-coordinate an adequate multiple of the deflection – depending on the detector plane – is added. The second addition and the cut off of the last binary digits cause a round off of the projected y value.

[3] FIFO: first-in, first-out

[4] The signs $\lfloor\,\rfloor$ are the lower Gaußparentheses by which a number is rounded off to the next integer.

In Figure 4.8 the project unit is shown which projects the y-coordinate onto the imaginary middle plane. It implements equation (4.1) as

$$\mathtt{Y} = \left\lfloor \frac{1}{2} \left\lfloor \frac{1}{2^2}\mathtt{y} + \frac{1}{2^2} \left\lfloor \frac{\mathtt{C}_i}{2^2}\mathtt{d} \right\rfloor \right\rfloor + \frac{1}{2} \right\rfloor \approx \frac{1}{2^3} \cdot \mathtt{y} + \frac{\mathtt{C}_i}{2^5} \cdot \mathtt{d} \tag{4.7}$$

with the constant which depends on plane i

$$\mathtt{C}_i = -\frac{x_i - x_{\mathrm{mid}}}{3\,\mathrm{cm}} \cdot \frac{140\,\mathrm{\mu m}}{160\,\mathrm{\mu m}} \cdot 2^2. \tag{4.8}$$

The result is the y-coordinate projected

$$\mathtt{Y} = \frac{y_{\mathrm{proj}}}{1.28\,\mathrm{mm}}. \tag{4.9}$$

4.3.6 Computations and Buffering of the Reconstruction Parameters

For the calculation of the equalizing even by track segments according to equation Equation (5.8) the value y_i' equivalent to (5.4) is required. For this reason a table value is added into the $\mathtt{y}'$ calculation unit to the non-projected y-coordinate $\mathtt{y}$ which is dependent on the plane i and on $\mathtt{Z}$:

$$\mathtt{y}' = \mathtt{y} + \mathtt{C}_{i,\mathtt{Z}}.$$

The table values $\mathtt{C}_{i,\mathtt{Z}}$ result from the geometry of the detector by comparison with equation (5.4).

For each track segment the value of $\mathtt{y}'$ together with the electron probability $\mathtt{P}$ is put to a memory. The size of the memory depends on the simulation. In figure 4.9 the distribution of the amount of track segments per module is shown that amount to – two optical connections –[5] In order to file all the track segments also at a high multiplicity the depth of the memory is fixed to 40 inputs. The storage address comes from a 6-bit counter which is set to zero by the `pretrigger` signal and is enhanced by one for each of the track segments.

In the end the values put into the memory are needed for the reconstruction of the transverse momentum and the kind of particle of a track. At first they are not needed for the combination of the track segments into tracks. Numerous registers can be saved by using 6-bit addresses instead of 21 bit wide reconstruction data.

4.4 Z-Channel Unit

4.4.1 Z-Channels

As there are only 16 different possibilities for the numbers of the pad row it is obvious that the computations at this part should be done parallel.

[5] Here all track segments are counted which are determined for the transmission. It is not considered that some of the segments are omitted because of the limitation to 20 track segments as mentioned above.

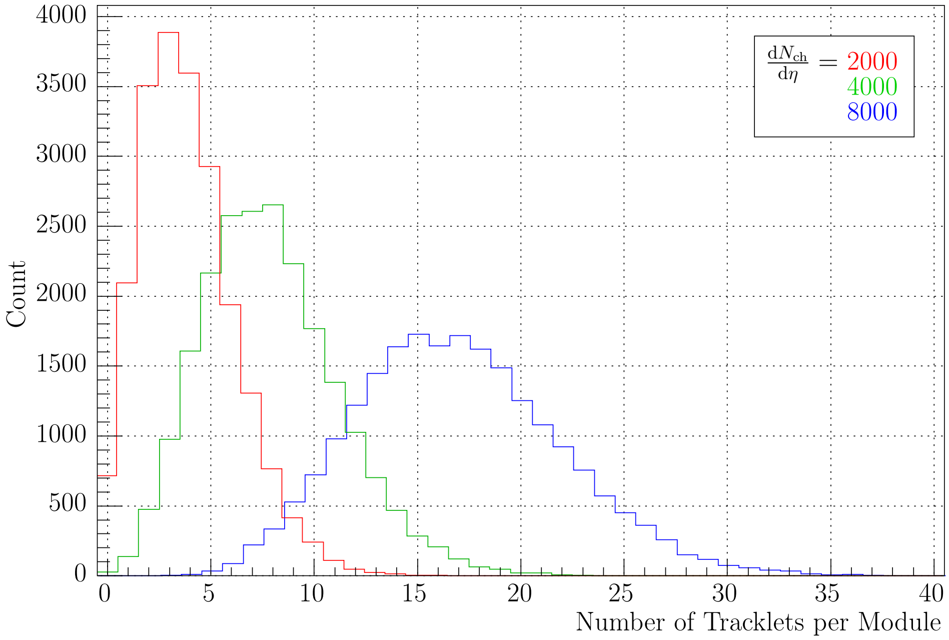

Figure 4.9: Distribution of the amount of track segments per detector module for various multiplicity densities. The maximum amount of track segments per module determines the size of the memory in the input units.

Instead of projecting the z-coordinate of the track segments to an imagined middle plane and to compare the distances between the segments on this plane with the half of the window width the window is projected onto the plane of the six detector modules where it is compared to the z-coordinates. Here as a basic plane for the projection not the imagined exact middle plane is used but one of the existing detector planes. In principal it is irrelevant which of the planes is chosen. It is however recommendable to use one of the two middle planes as a basic plane since then the range of intersection of the projections is minimized. As can be seen in the following this allows a very efficient implementation. For the further development the plane 2 was chosen.

Figure 4.10 shows the principle of the re-projection in an example. A particle the path of which in plane 2 intersects pad-row 6 (marked blue) can only intersect pad-row 6 or 7 if coming from the collision point because of the geometry of the detector in plane 3. For the other planes the equivalent is valid. During the projection a spatial insecurity of the primary vertex (collision point) in z-direction of ±20 cm is recognized.

If one would mainly recognize the track segments of this stack which are marked blue in figure 4.10 the z-coordinate would not have to be compared any further. It is already obvious by the selection of the track segment that they are sufficiently equal in their projected z-coordinate However one would only find tracks which run on plane 2 through pad-row 6 independent from the factor whether there was a track segment found on this plane or not.

In order to find all the tracks the selection described is now carried out parallel for all of

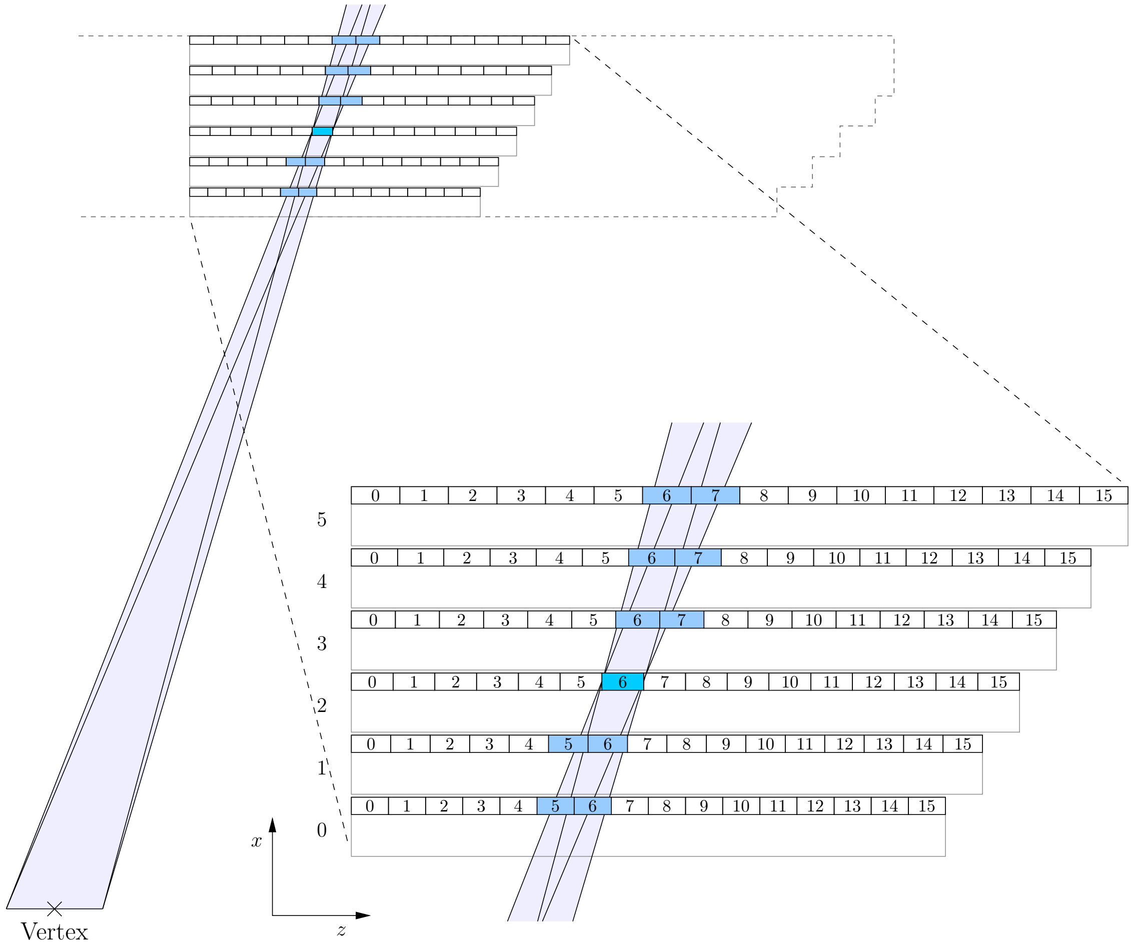

Figure 4.10: Sectional view of a module stack. A particle track can run through the detector modules of the various planes in different pad-rows. The range marked shows very clearly the pad-rows that can be reached by a particle from the vertex which runs on plane 2 through pad-row 6.

the 16 pad-rows of the basic plane. We do however not need 16 completely independent units for this. It is given from the geometry and the presumed insecurity of the vertex position that the project range in the other planes exceeds no more than three pad-rows and that the projection ranges which are about three units away do not overlap. As the track segments arrive assorted in z direction of the GTU the same unit can at first do the projection of the first pad-row and then the fourth, seventh and so on pad-row. Instead of using 16 independent units it is sufficient to use only three units of which each chooses the data of every third projection range for the processing.

The data of the three units are called *z-channels* 0, 1 and 2. The z-channel j contains the track segments from the projection range of the pad-row k with $j \equiv k \pmod 3$. Within the z-channel it is *Index* starting with one[6] the actual number of the projection (sub-channel).

[6] Pad-Rows that are irrelevant for a z-channel are called Index 0 in the hardware description.

Figure 4.11 shows the allocation of the track segment to the three z-channels for a detector stack. The distribution can be explained best with an example: If a track segment of plane 4 arrives from pad-row 3 in the GTU the index 2 is allocated to it in the z-channel 0 (blue).It does not occur in z-channel 1 (red) and in channel 2 (green) it gets index 1.

The analogy in the z-position is assured by totally separate parallel processing of the data of the three z-channels and combination of track segments with the same index only to tracks within the channels and the z-coordinates must not be recognized any more in the further calculation. Even though the combination logic is tripled this way the complexity of the problem is significantly reduced because of the fact that the comparison must now only be done within two dimensions.

4.4.2 Set Up of a Z-Channel Unit

The set up of the z-channel unit is shown in 4.12. The total of $6 \cdot 3 = 18$ z-channel units each work for one of the detector planes on one of the three z-channels. By comparison of the `Z` value with a *look-up table* (LUT) it is at first determined for every track segment

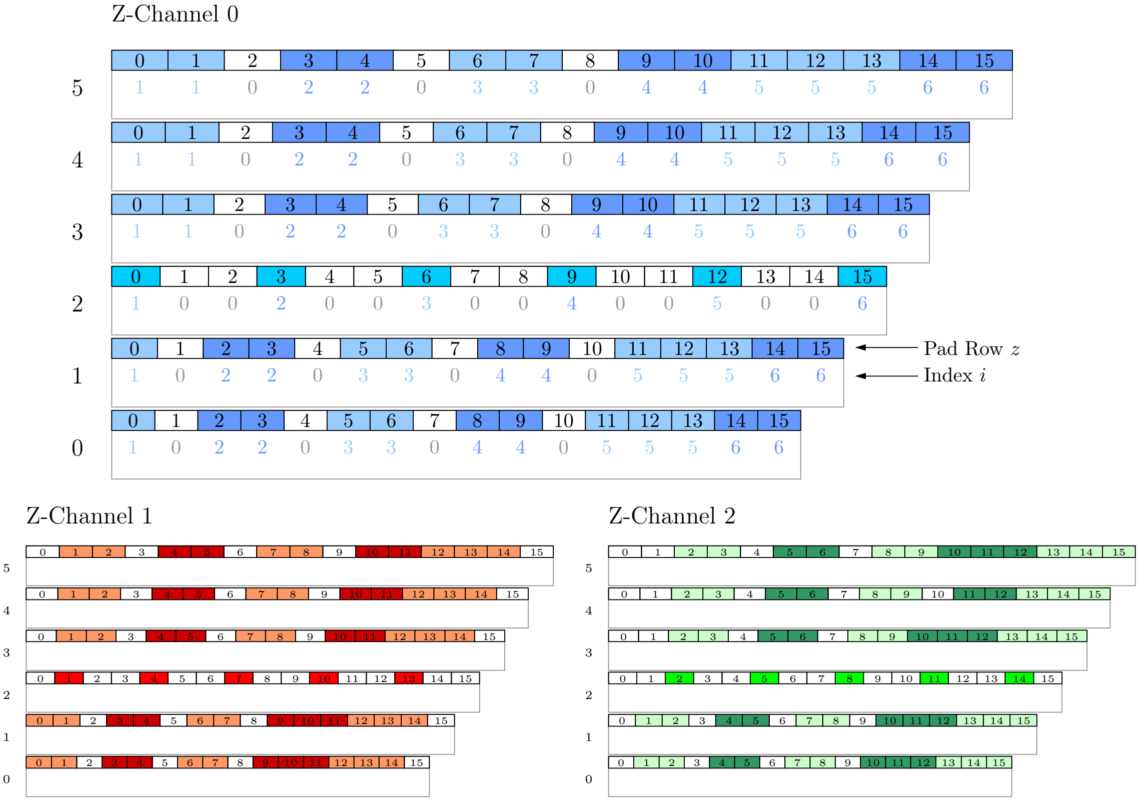

Figure 4.11: Distribution of the track segments into the three z-channels at the example of a detector stack. The track segments are allocated to z-channels each of which covers a certain angle range. According to the angle range the segment within the channel receives an sub-channel index i. If a track segment does not occur in the z-channel it is marked by $i = 0$.

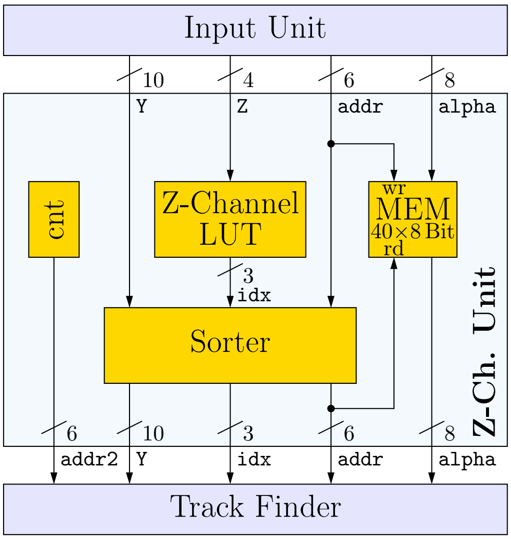

Figure 4.12: The set up of a z-channel unit as a block diagram. The unit contains all track segment data of a plane. With the help of a look-up table (LUT) the segments are selected for this Z-channel and by an index allocated to the sub-channels. The segments of each sub-channel are assorted by their y-coordinate and forwarded to the track finding unit.

whether and if yes with which of the sub-channel indices `idx` it belongs to this channel. Track segments with the same index are then assorted in the sorter according to their projected y coordinate.

In the memory that is parallel to the sorter (see figure 4.12) the `alpha` values of the track segments are filed which are not needed for the sorting. After the sorting they are called with their address relevant `addr`. This way 8 bit memory space each is saved in the sorter cells built of flip-flops.

As the original memory addresses `addr` after the sorting are not continuously ascending any more an additional number`addr2` is generated in another counter which numbers the track segments ascending in a new sequence. It is used as a write address from the succeeding memory in the track finding unit.

4.4.3 Z-Channel Selection Table

The selection table shows for every track segment whether it belongs to a certain z-channel and if yes which sub-channel index it has in it. It implements the allocation as is shown in figure 4.11 as an example. For the z-channel 0 shown the value table for instance in plane 5 is:

Z	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
idx	1	1	0	2	2	0	3	3	0	4	4	5	5	5	6	6

The index zero means that the segment does not occur in this z-channel.

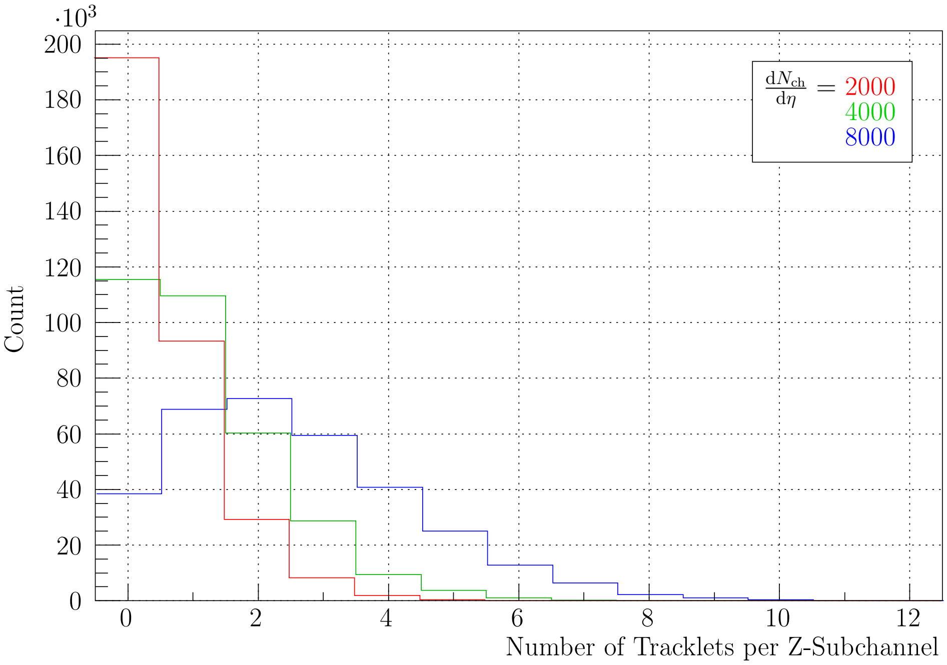

Figure 4.13: Distribution of the number of track segments per z-channel for various multiplicity densities. By the maximum amount it is fixed how large the buffer of the sorter needs to be.

4.4.4 Sorter and Buffers

The track segments which were originally assorted within a pad-row to y-coordinates are not assorted any more because of the projection onto the middle plane and by combining of the segments up to three pad-rows. The combination algorithm of the track finding unit however requires sorting in order not to have to compare all track segments with one another.

For the size of the sorter it is necessary to know how many track segments with the same index occur within a z-channel on one plane. Figure 4.13 shows the distribution in the simulation. Even at maximum multiplicity there are very rarely more than eight segments in the same z-channel. For this reason the number eight is used as the maximum depth of the sorter.

The sorter is optimized to minimal latency. After the last track segment of an index that has to be compared has already arrived in the subsequent clock cycle the output of the smallest element starts. In order to achieve this it is according to figure 4.14 set up from parallel plugged sorter cells. Each of the cells files a data word.

The sorter can take a data word independently from its condition at each clock cycle. New inputs at the same time are compared parallel with all the inputs present and set directly into the right place. Table 4.2 shows the way of processing of the sorter in an example. The contents of the first four sorter cells and the value at the input is given. The output here is always the data word of cell 0. The output is marked valid by the signal `valid_out`

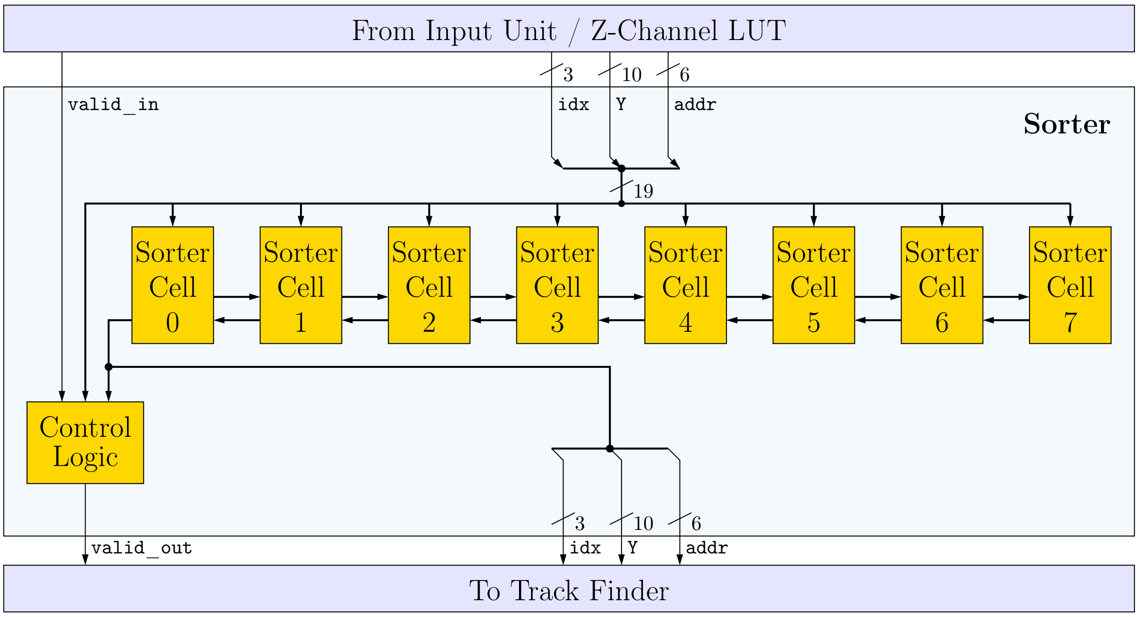

Figure 4.14: The set up of the sorter as a block diagram. It consists of eight identical sorter cells and one control logic. Incoming track segments are filed directly in one of the sorter cells therefore the segments of the cells are assorted ascending. Across the diagonal connections the track segments can be shifted between neighboring sorter cells. When all the segments of an index are in the sorter they will be issued one after the other assorted by y.

for the succeeding units if the latest track segment input had a different index and if there are all the track segments with the same index already present in the sorter.

Figure 4.15 shows the set up of a sorter cell. The comparison logic determines whether the new value is to be put left or right from the actual value. By a selection logic it is then decided whether the input value is taken in the next clock cycle, the actual input is kept or the input of the left or right connecting cell is taken. The decision tree of the selection logic is shown in figure 4.16.

4.5 Track Finding Unit

The track finding units recognize the track segments of each of the z-channels across all the planes and combine them to tracks. Therefore they at first compare the `Y` value and if equivalent additionally the value of the angle `alpha`.

4.5.1 Way of Function

Not all the values are compared in pairs but from every plane only a small section of the values assorted to the y-coordinate is recognized that is successively shifted to larger values. For efficiency reasons a randomly selected reference plane is presumed with the

 step	cell 0 (idx,Y)	cell 1 (idx,Y)	cell 2 (idx,Y)	cell 3 (idx,Y)	input valid	input (idx,Y)	output valid
⋮							
4	(1, -372)	(1, 89)	leer	leer	1	(1, 74)	0
	↓	↘					
5	(1, -372)	$\boxed{\text{(1, 74)}}$	(1, 89)	leer	0	–	0
	↓	↓	↓				
6	← (1, -372)	(1, 74)	(1, 89)	leer	1	(3, -141)	1
	↙	↙					
7	← (1, 74)	(1, 89)	$\boxed{\text{(3, -141)}}$	leer	0	–	1
	↙	↙					
8	← (1, 89)	(3, -141)	leer	leer	1	(3, -268)	1
		↓					
9	← $\boxed{\text{(3, -268)}}$	(3, -141)	leer	leer	1	(4, 221)	1
⋮							

Table 4.2: Example for the function of the sorter. The table shows a part of a fictive sorting procedure. The new value at the input is put right into the correct cell. Therefore the values in the sorter are always assorted ascending. If to an index idx all data words are in the sorter then at every clock cycle a word is put out. Depending on the constellation the values in the sorter are shifted to right of left or keep their position.

track segments of which the segments of the other planes are compared. Only tracks can be found that have a contributing track segment in the reference plane. This restriction can be neglected by setting up the complete unit triple parallel by determining each time a different reference plane. This way it is granted that tracks that produce a track segment in at least four of the six planes can at least be found in one of the units as a reference plane for the units 1, 2, and 3 are chosen.

Size of the Comparison Ranges. Two tracks that accidentally intersect the projection plane under various angles can overlap in the y ascending assorted succession. Therefore it is insufficient to recognize only the in y best fitting reference segment from each plane at the reference plane. Instead there are two succeeding track segments compared at the same time. In this way a track segment can also correctly be allocated (by its angle) if there is y“b”ecause of an error a more suitable segment.

It follows from the simulation that it is sufficient to restrict the recognition to only two in y fitting track segments. In table Table 4.3 there is for tracks looked for (electrons with $p_\mathrm{t} \geq 3.0\,\mathrm{GeV/c}$) given how many track segments are on the other planes within the y-connecting window (see table Table 4.4) seen from the reference plane. Here only the planes are recognized where definitely there is a fitting track segment. At a multiplicity density of $\frac{\mathrm{d}N_\mathrm{ch}}{\mathrm{d}\eta} = 2000$ for example there is in 98.662 % of the cases the track segment searched for is the only one within a y window. In 1.332 % of the cases there are two track segments within the window width one of which is exactly the right one. Only in 0.006 % of the cases there are three segments within the window width.

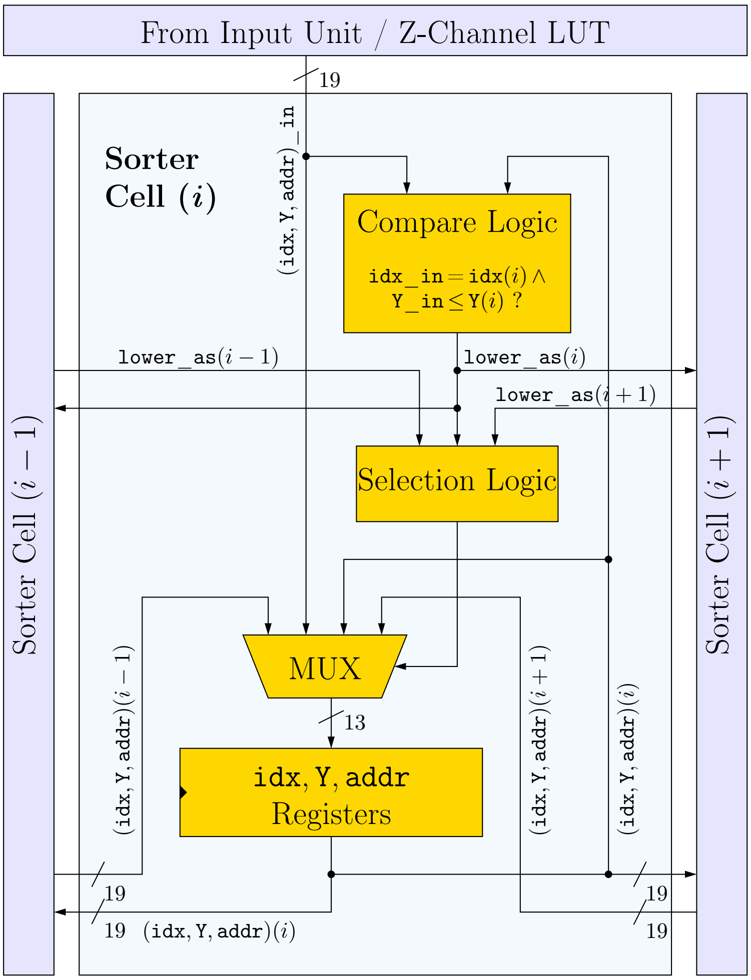

Figure 4.15: The set up of a sorter cell. The relevant data of a track segment are filed in the register. At every clock cycle the new data word laying at the input or the data word from a neighboring sorter cell can be taken. The decision is made from the selection logic on the base of value comparisons in this and the two bordering cells.

number of track segments	occurrence at $\mathrm{d}N_{\mathrm{ch}}/\mathrm{d}\eta =$ 0	2000	4000	8000
1	99.273 %	98.662 %	97.893 %	95.983 %
2	0.721 %	1.332 %	2.084 %	3.908 %
3	0.006 %	0.006 %	0.022 %	0.107 %
4	–	–	–	0.003 %

Table 4.3: Distribution of the number of track segments that are in y direction within the equality window for various multiplicity densities. As there are very rarely more than two track segments within the $y$window one can restrict the search for suitable track segments to checking only the first two in y fitting segments of the deflection angle.

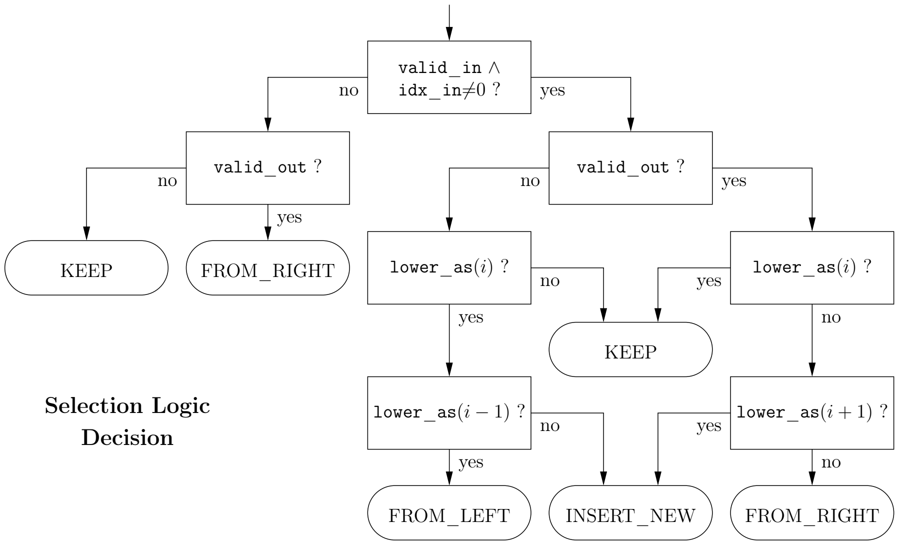

Figure 4.16: The decision tree of a selection logic in the sorter cell. Which of the track segments is taken depends on whether a new segment is present (`valid_in`), whether one has been put out (`valid_out`) and what the comparisons (`lower_as`) in this and the neighboring cells show.

Figure 4.17 illustrates the function of the track finding unit. The data rows assorted of the various planes are allocated that track segments that belong together are put to the common vision range. The reference plane s is marked with colors in the figure.

Set up. The track finding unit consists mainly of memories for the data rows and a strictly combinatorial logic which defines the turn of the data sequences and controls whether the unification criteria are given. The set up is shown in figure 4.18. As there are two succeeding data words recognized at once memories with two independent reading ports

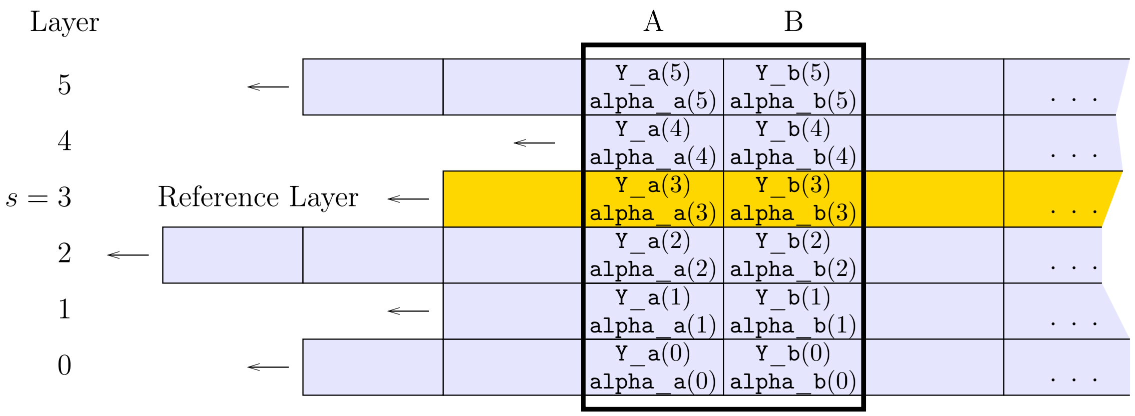

Figure 4.17: The track finding unit looks from every plane at two succeeding track segments. It shifts the data of the various planes in a way that track segments that belong together are put together in the vision range.

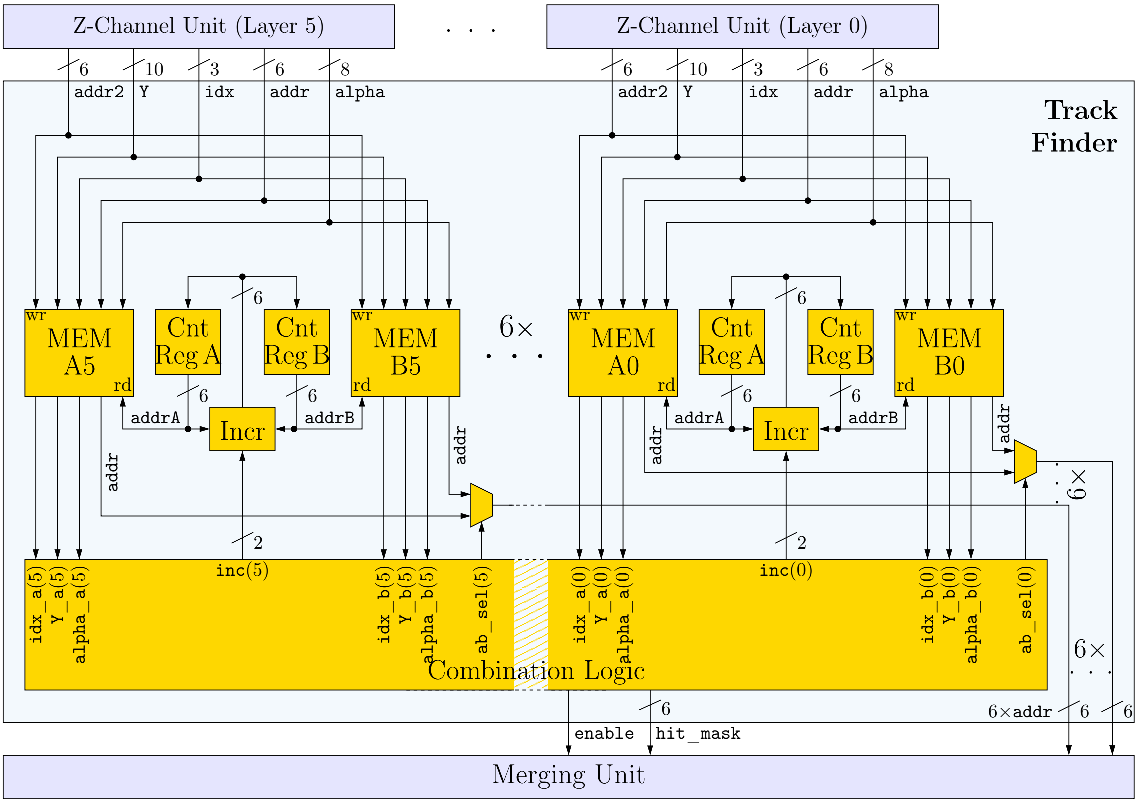

Figure 4.18: The set up of a track finding unit. The track segments of all of the six planes are taken parallel and are each filed in two memories. Per memory block there is one counting register for the reading address. The combination logic compares the values taken from the storage. It recognizes whether the track segments have one track in common and determines the rise of the counting register at every clock cycle.

are needed. Alternatively here the set up of two independent memories is shown which are described parallel. Each of the memories gets its data directly from the sorter of the relevant z-channel units. The write address for every plane comes from a common counter which is enhanced by one for each track segment. The reading address is according the position of the range recognized (cf. figure 4.17). With the exception of the complicated to handle start and end conditions which are not shown here addrB $=$ addrA $+1$. is always valid. At every clock cycle the vision range can be shifted by one or even two fields or not move at all. Equivalently addrA and addrB depending from the value of the signal inc are increased at every clock cycle by 0, 1 or 2. The combination logic receives from each of the twelve memories the three values idx, Y and alpha. If a track (*Hit*) is found the enable signal is activated for one clock cycle. The hit_mask shows at which of the planes there are adding track segments. The signal ab_select shows for every plane whether the adding track segment is at position A or B. The addresses of the adding track segments are forwarded to the next unit together with the hit mask. Additionally for the marking of the track – in the block diagram not visible – the index idx and the round off y-coordinate approx_y of the track segment are forwarded to the reference plane.

size	resolution	max. deflection	factor	name
y_{proj}	$1/1.28\,\text{mm}$	$11.625\,\text{mm}$	9	$\Delta\texttt{Y}$
α	$1/0.004\bar{6}\,\text{rad}$	$0.05\,\text{rad}$	11	$\Delta\texttt{alpha}$

Table 4.4: The numerical values for the GTU-window criterion as they are used in the hardware model. Due to the optimization of the hardware description at the resolution values are not smooth any more.

4.5.2 Combination Logic

The combination logic processes the signals[7]

$$\texttt{Y\_a}(i), \texttt{Y\_b}(i) \quad \text{und} \quad \texttt{alpha\_a}(i), \texttt{alpha\_b}(i) \qquad (\text{mit } 0 \leq i < 6).$$

Starting from the track segment A of the reference plane s the window limits are calculated

$$\begin{aligned} \texttt{Y}^{+} &= \texttt{Y\_a}(s) + \Delta\texttt{Y} & \qquad \texttt{alpha}^{+} &= \texttt{alpha\_a}(s) + \Delta\texttt{alpha} \\ \texttt{Y}^{-} &= \texttt{Y\_a}(s) - \Delta\texttt{Y} & \qquad \texttt{alpha}^{-} &= \texttt{alpha\_a}(s) - \Delta\texttt{alpha} \end{aligned}$$

with the numeric values of table 4.4. From them the Boolean variables are made

$$\begin{aligned} b_{\text{Hit A},\,i} &:= (\texttt{Y}^{-} < \texttt{Y\_a}(i) < \texttt{Y}^{+}) \wedge (\texttt{alpha}^{-} < \texttt{alpha\_a}(i) < \texttt{alpha}^{+}) \\ b_{\text{Hit B},\,i} &:= (\texttt{Y}^{-} < \texttt{Y\_b}(i) < \texttt{Y}^{+}) \wedge (\texttt{alpha}^{-} < \texttt{alpha\_b}(i) < \texttt{alpha}^{+}) \end{aligned}$$

which determine for each plane i whether the track segment A respectively B can be unified with the reference segment.

Are both variables wrong for a plane this does not mean there is no fitting segment on this plane because it might be that the wrong section of the plane had been recognized. To make clear whether the vision range has progressed enough

$$b_{\text{aligned},\,i} := (\texttt{Y}^{-} < \texttt{Y\_a}(i)) \vee (\texttt{Y}^{+} < \texttt{Y\_b}(i))$$

is calculated. The variable $b_{\text{aligned},\,i}$ is true if the data row of the plane i related to the reference plane *aligned* is progressed far enough. This is the case if the y-coordinate of the segment A is not more than half a window width below the reference segment. Additionally a plane is seen aligned if the y-coordinate of segment B is already above the window limit since in this case the shifting of the plane can not put a track segment into the vision.

The sums

$$\begin{aligned} N_{\text{Hits}} &= \sum_{0 \leq i < 6} \mathbb{1}_{\{b_{\text{aligned},\,i} \wedge (b_{\text{Hit A},\,i} \vee b_{\text{Hit B},\,i})\}} \\ N_{\text{uncertain}} &= \sum_{0 \leq i < 6} \mathbb{1}_{\{\neg b_{\text{aligned},\,i}\}} \end{aligned}$$

[7] The `idx` signals are in the following not explicitly mentioned. A difference in `idx` is taken like a big difference in `Y`.

are calculated[8] that N_{Hits} gives the amount of secure equalities and $N_{\text{uncertain}}$ the counts the planes with no safe proposition. A track is recognized found if

$$\texttt{enable} := \mathbb{1}_{\{N_{\text{Hits}} \geq 4 \wedge N_{\text{uncertain}} = 0\}}$$

is one. The signals

$$\texttt{hit\_mask}(i) := \mathbb{1}_{\{b_{\text{Hit A},\,i} \vee b_{\text{Hit B},\,i}\}}$$

$$\texttt{ab\_select}(i) := \begin{cases} 0(\text{A}) & b_{\text{Hit A},\,i} \\ 1(\text{B}) & \text{otherwise} \end{cases}$$

then show for each plane i whether a segment contributes to the track and if yes whether this stands instead of A or B. Always the first analogy is used. The alternative at a double analogy to select the “best” would require additional reference logic. As two fitting segments occur only very rarely at this point the additional complexity is omitted.

Shifting of the reference line. The combination logic now must decide for the data row of each plane how far this may be “shifted” underneath the vision range. The reference plane here has a special part. Its data row may only be shifted if it is absolutely obvious whether a track can be found to the actual segment at position A or not. In special cases the reference row can even be shifted two fields. Therefore the sum

$$N_{\text{way beyond}} = \sum_{0 \leq i < 6} \mathbb{1}_{\{\texttt{Y\_b}(s) + \Delta\texttt{Y} < \texttt{Y\_a}(i)\}}$$

calculated. $N_{\text{way beyond}}$ give the number of the rows which have progressed already far enough so that their segments even with segment B of the reference plane can not be unified to a track any more. The number of elements by which the reference row shall be shifted $\texttt{inc}(s)$ therefore is derived from the following algorithm:

```
if ($N_{\text{uncertain}} \neq 0$) and ($N_{\text{uncertain}} + N_{\text{Hits}} \geq 4$) then
  inc($s$) := 0
elsif ($N_{\text{way beyond}} \geq 3$) then
  inc($s$) := 2
else
  inc($s$) := 1
end if
```

The reference row is stopped if one of the other rows is not yet properly aligned and in addition the possibility exists that a fitting track is found as the number of possible analogies N_{Hits} together amount to at least four. Otherwise the reference row can be shifted. If in at least three of the other rows the `Y` values are already high enough to exclude a unification with segment B of the reference plane also this can be rejected by enhancing the reading address in the reference row by two. Otherwise it is enhanced by one in order to search for tracks to this succeeding segment during the next clock cycle.

[8] The symbol $\mathbb{1}$ names the indicator function the one is if its argument is true and otherwise takes the value zero.

Shifting of the other Rows. Now it must be decided how many fields the vision range can be shifted in the other rows. The decision is depending on `inc`(s) which is the behavior of the reference row. If this is shifted one field (`inc`$(s) = 1$) the shifting of row i is given through comparison with segment B of the reference plane which becomes a reference segment in the following clock cycle:

```
if (Y_b($i$) $<$ Y_b($s$) $-$ $\Delta$Y) then
  inc$'$($i$) $:= 2$
elsif (Y_a($i$) $<$ Y_b($s$) $-$ $\Delta$Y) then
  inc$'$($i$) $:= 1$
else
  inc$'$($i$) $:= 0$
end if
```

If the y-coordinate of the B segment of the plane viewed is to little to be unified with segment B of the reference plane which is the next reference plane both segments of the viewed plane can be rejected. If only segment A fulfils the condition then at least this one is irrelevant in the next clock cycle and the row can be shifted one field. Otherwise the row is stopped to totally be compared with the new reference segment in the succeeding clock cycle.

If the reference plane is shifted two fields (`inc`$(s) = 2$) it could be compared with a fictive segment C instead of segment B of the reference plane. This extension is rejected and also in this case the value determined before `inc`$'(i)$ is used.

This altogether results in `inc`(i) according to the following algorithm:

```
if (inc($s$) $\neq 0$) then
  inc($i$) $:=$ inc$'$($i$)
else
  if (Y_a$^-$ $<$ Y_a($i$) $<$ Y_a$^+$) then
    inc($i$) $:= 0$
  elsif (Y_b($i$) $<$ Y_a$^-$) then
    inc($i$) $:= 2$
  elsif (Y_b($i$) $<$ Y_a$^+$) then
    inc($i$) $:= 1$
  else
    inc($i$) $:=$ inc$'$($i$)
  end if
end if
```

If the reference row is not shifted the remaining rows shall only be aligned to the actual reference segment accordingly If the segment A fits already to the reference segment in relation to the y-coordinate the row will stay at its actual position. If both segments of the actual row have a y-coordinate that is too low then the row is shifted two fields. If segment B fits to the reference segment but not segment A then the row is shifted one field. If none of the three cases does apply i. e. neither segment A nor B fits to the reference segment and their y-coordinates are not to small then an accordance with the reference segment for this plane is ruled out. Afterwards it is proceeded as if the reference plane is shifted and as described before it is compared with segment B of the reference plane.

The selection procedure aims for a maximum enhancement of the reading address at every clock cycle for each plane in order to minimize the total latency of the unit.

The combination logic is a complex combinatorial part of the design. For the parallel value comparisons 39 10-bit adders, 22 7-bit adders, and 35 3-bit adders in every unit are needed.

4.6 Unification Unit

By dividing the data into z-channels and the parallel track search with various reference planes it might happen under certain circumstances that the same track is recognized several times. This disadvantage of the parallel procedure is eliminated by the unification unit. It unifies the data of the nine track finding units into one and this way assures that every track is only forwarded one by one.

In order not to compare all tracks coupled the principle of the pre-sorting is applied. In several (*merging stages*) the data are combined in a way that the same tracks succeed one another. The data prepared this way pass (*uniquifiers*) which recognize identical tracks in the sequence and reject it if they occur twice. Figure 4.19 shows the set up as a block diagram.

4.6.1 Unifying of various tracks of different reference planes

At first within the individual z-channels the tracks are combined which were found starting from various reference planes.

By a simple method it is possible to suppress most of the multiple findings already on the level of the track finding units: tracks of the track finding unit with the reference plane $s = 2$ are only forwarded if they have no contributing track segment on plane 3 as these are already recognized from the track finding unit with reference plane $s = 3$. According to this does track finding unit with $s = 1$ accept only tracks that have neither on plane 2 nor on plane 3 a segment. Nevertheless due to side effects double findings might happen as the window positions through the various reference planes are slightly different in the three units.

The track data are at first put into buffers (FIFOs). The succeeding unit compares at each clock cycle the oldest values of the three FIFOs and chooses the “smallest” to forward it

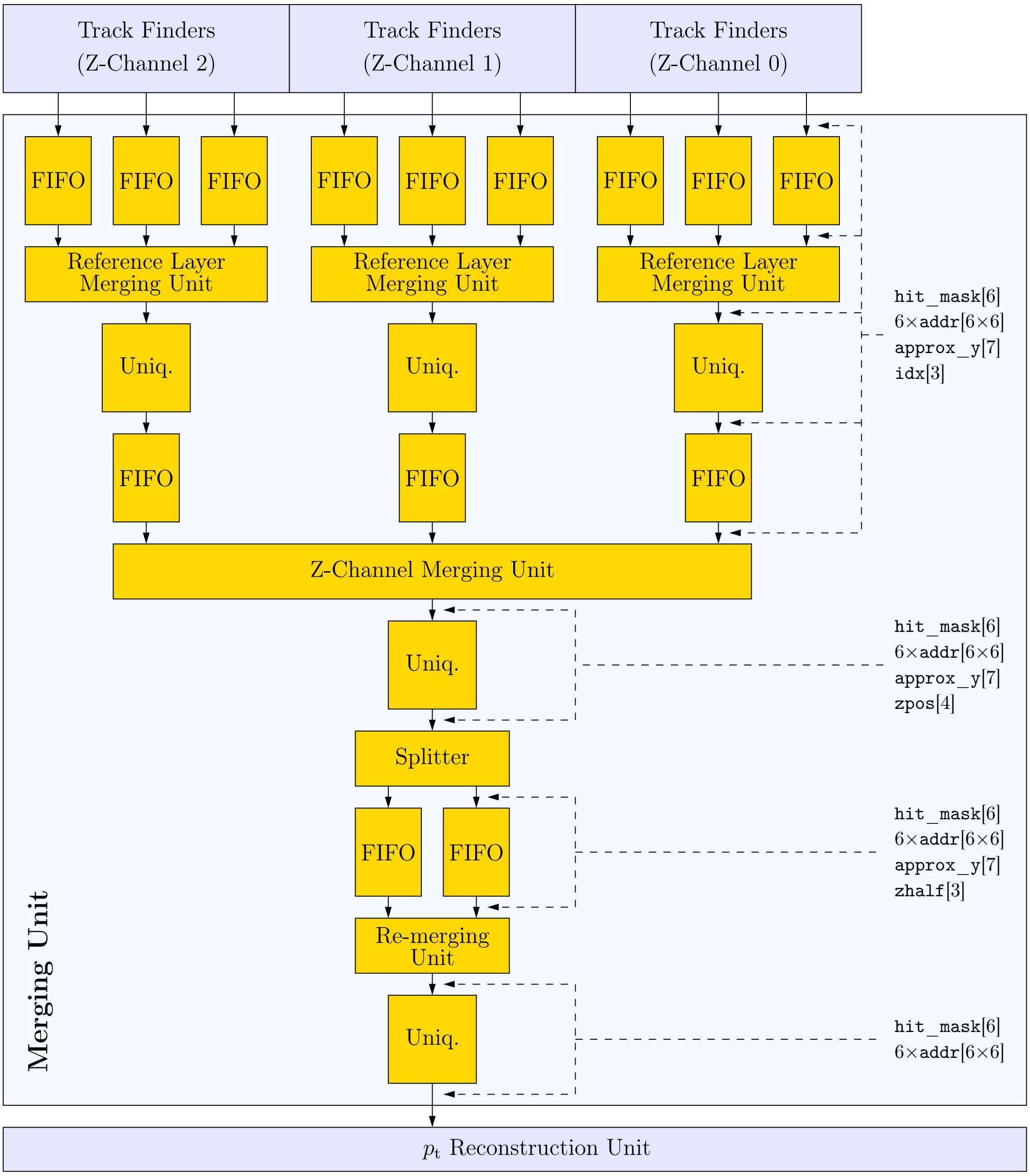

Figure 4.19: The set up of the unification unit. The arriving data are at several levels buffered each in FIFOs and unified in order. Double inputs are rejected here.

and to eliminate it from the memory. As an order criterion at first index `idx` is used and secondly the approximate $-coordinate `approx_y` is used:

$$
\begin{aligned}
&\texttt{track\_A} < \texttt{track\_B} \ :\Leftrightarrow \\
&\qquad\qquad \texttt{idx\_A} < \texttt{idx\_B} \vee (\texttt{idx\_A} = \texttt{idx\_B} \wedge \texttt{approx\_y\_A} < \texttt{approx\_y\_B}).
\end{aligned}
$$

The track data sets are assorted by sequencing the data sets with equal tracks. Pre-assorted like that the data come to the uniquifier.

4.6.2 (“Uniquifier”)

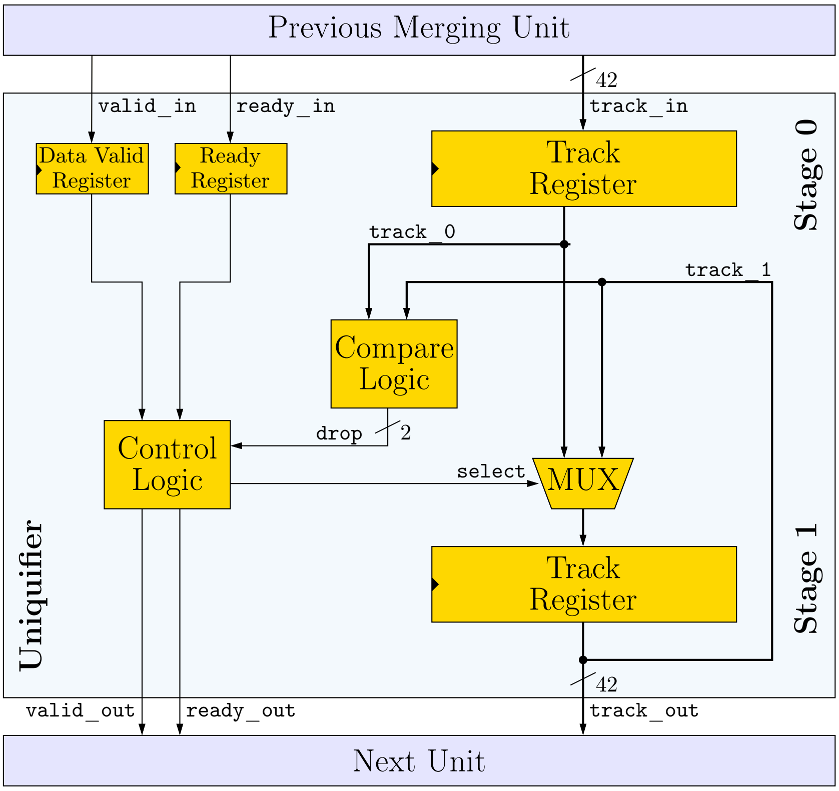

Figure 4.20: The set up of a uniquifier. It files in two register levels succeeding track data words for a coupled comparison. If they are equal one of the words is rejected in order to remove multiply found tracks from the data stream.

Figure 4.20 shows the set up of a uniquifier. Two succeeding track data sets are filed into both the register levels. The comparison logic tests whether one respectively which of the track data sets shall be rejected. At this point it is only worked at the addresses of the track segments. Two tracks are meant to be equal as soon as they have one common track segment. That track is rejected that has less contributing track segments. The track data set of level 1 is output as soon as the data of another track are in level 0 or the end of the data stream is reached.

`idx`	1	1	1	2	2	2	3	3	3	4	4	4	5	5	5	6
`zch`	0	1	2	0	1	2	0	1	2	0	1	2	0	1	2	0
`key`	0	1	1	2	2	3	3	4	4	5	5	6	6	7	7	8
`zpos`	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
`zhalf`	0	0	1	1	2	2	3	3	4	4	5	5	6	6	7	7

Table 4.5: Sorter key for the connection of the tracks. The table shows all the values for all z-channels according to which is assorted in the individual steps during the connection procedure.

4.6.3 Combining of the Tracks of Various Z-Channels

The next step is to combine the tracks from the three z-channels.

Tracks from six segments can only be found in a z-channel with six segments as the z-channels according to their construction do not intersect (see figure 4.11). In the other planes however the ranges of the z-channels intersect by the detector geometry and the presumed spatial insecurity of the interaction point that the track in one or the other neighboring z-channel can be found with four or five segments under some circumstances.

The combination is more complicated as there can be no sorting with simple connecting where the data with equal tracks are always succeeding one another. Instead a two-level procedure is necessary. at a first step the data of two neighboring z-channels are assorted connected after y-coordinates `approx_y` Double values are eliminated from the data stream. The remaining values are separated again afterwards and the data of two neighboring sub-channels are re- connected in the other possible way of coupling. Again the double values are eliminated so that finally all tracks found twice are wiped off.

The sequence of the connection in the individual steps is fixed by the order criteria by which every time the smallest element is chosen. The values of the sorter key

$$
\begin{aligned}
\mathtt{key} &:= \left\lfloor \frac{\mathtt{zch} + 3 \cdot \mathtt{idx}}{2} \right\rfloor - 1 \\
\mathtt{zpos} &:= \mathtt{zch} + 3 \cdot (\mathtt{idx} - 1) \\
\mathtt{zhalf} &:= \left\lfloor \frac{\mathtt{zpos}}{2} \right\rfloor
\end{aligned}
$$

are given in table 4.5. In a first step it is assorted ascending to `key`. At equal values there is always the y-coordinate `approx_y` is compared:

$$
\begin{aligned}
&\mathtt{track\_A} < \mathtt{track\_B} \;:\Leftrightarrow \\
&\qquad \mathtt{key\_A} < \mathtt{key\_B} \vee (\mathtt{key\_A} = \mathtt{key\_B} \wedge \mathtt{approx\_y\_A} < \mathtt{approx\_y\_B}).
\end{aligned}
$$

The *splitter* afterwards splits up the data to the lowest value bit of the value `zpos` into two FIFOs. At a new connecting procedure it is finally assorted to the value of `zhalf`.

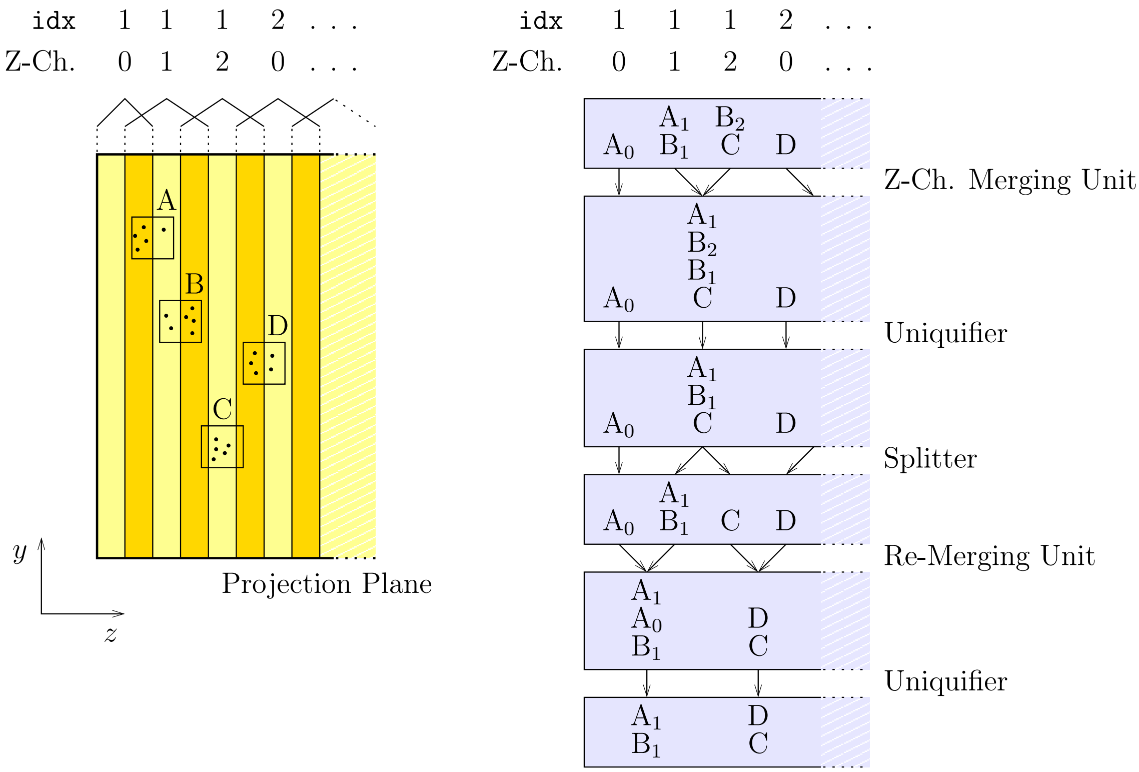

Figure 4.21: Example for the elimination of double data sets during the connection procedure of tracks found in the various z-channels. (see text).

Example

The procedure is shown in figure 4.21 in an example. At the left half the projection plane is shown schematically. The lines mark the projection range of a Z-sub-channel. The intersection ranges of neighboring sub-channels are marked stronger. The dots stand for projected track segments. The four tracks A, B,C, and D can be found. For example track A is with 5 track segments in the range of z-channel 1. It is however to be found with 4 track segments in z-channel 0. But Track D is only in the z-channel 0 as the three track segments in the intersection range with channel 0 are insufficient for an evidence.

In the right half of the figure 4.21 the step by step procedure during the connection of the tracks is shown. At first there are six tracks found. After the first step of connection procedure both tracks B_1 and B_2 are behind one another so that the double of track B can be eliminated by the unifier. The tracks are distributed again to their original positions. In the second connection procedure step (Re-Merging) both data sets from track A are behind one another so that here too the one with less tracks can be eliminated by the uniquifier. Finally there are only exactly four data sets left for the four tracks.

5 Reconstruction of the Transverse Momentum

After the track segments are combined to tracks in the GTU the transverse momentum p_t of the particle generated shall be reconstructed from the track course. The computation necessary and their hardware realization are the contents of this chapter.

The track segments of a candidate are in the x-y-plane together with the vertex on an arc of a circle the radius of which the transverse momentum can be calculated. From the various procedures for the adjustment onto an arc of a circle to data points (see e. g. [FWSW03]) most of them are no good for the use in the trigger calculation because of the high amount of computing.

The most likely adequate seems the *conformal mapping* where a circle is mapped through the origin onto a straight line by inverting in the complex numerical plane ($u = \frac{x}{x^2+y^2}, v = \frac{y}{x^2+y^2}$). The impact parameter of the line then is inverse proportional to the radius of the circle.

However even this method is quite energy consuming because of the divisions necessary. On the other hand the radii of the circle courses are large enough that they are still approximately even at GTU level. The simulation shows that it is sufficient to adjust an even to a track segment and to presume it as a tangent respectively a secant of the arc of the circle. This way one derives the even parameters a and b (see Figure 5.1).

5.1 Adjustment of an Even to the Track Segments

Without “tilted Pads” (see paragraph 3.2.2) the deflection in z-direction could be neglected and it would be sufficient to minimize the error sum of squares

$$\chi^2 = \sum_{i \in \mathfrak{I}} (y_i - (a + b \cdot x_i))^2 \tag{5.1}$$

with

y_i : y-coordinate of the track segment in plane i
x_i : x-coordinate of the chamber outside in plane i.

The set $\mathfrak{I}$ here contains the indices $0 \leq i < 6$ of the detector planes from which a track segment contributes to the track. That gives $4 \leq |\mathfrak{I}| \leq 6$.

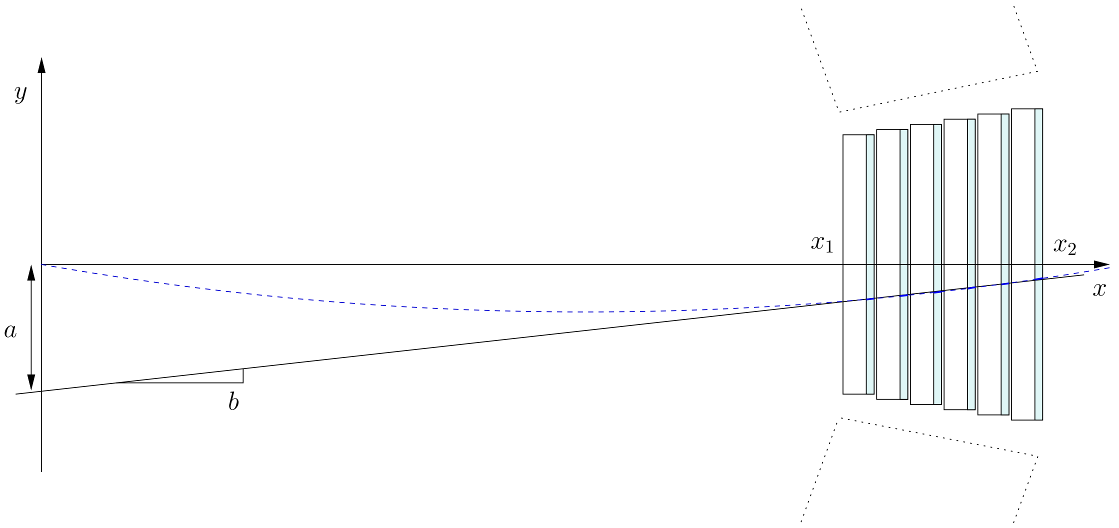

Figure 5.1: Adjustment of an even through the track segments. The gaps are given according to the scale in this draft.

With “tilted Pads” is y_i however can not be measured directly but according to (3.2) slightly depending on the z-position (see paragraph 3.2.2):

$$y_i = y_{\text{mess},\,i} + (-1)^i \cdot (z_i - z_{i,\,\text{row}}) \cdot \tan(\beta_{\text{tilt}}) \tag{5.2}$$

z_i here is also not the directly measurable z-position $z_{i,\,\text{row}}$ is the (known) z-coordinate of the middle of the pad-row. With this one gets:

$$\chi^2 = \sum_{i\in\Im} \left(y_{\text{mess},\,i} + (-1)^i \cdot (z_i - z_{i,\,\text{row}}) \cdot \tan(\beta_{\text{tilt}}) - (a + b \cdot x_i)\right)^2 \tag{5.3}$$

If one presumes the particles in x-z-plane coming from the origin then is

$$z_i = c \cdot x_i$$

for the constant c. If one also defines

$$\begin{aligned} y'_i &:= y_{\text{mess},\,i} - (-1)^i \cdot z_{i,\,\text{row}} \cdot \tan(\beta_{\text{tilt}}) \\ c' &:= c \cdot \tan(\beta_{\text{tilt}}), \end{aligned} \tag{5.4}$$

one gets:

$$\chi^2 = \sum_{i\in\Im} \left(-a - b \cdot x_i + c' \cdot (-1)^i \cdot x_i + y'_i\right)^2 \tag{5.5}$$

Because of the “tilted Pads” one must calculate a three dimensional linear regression instead of a two dimensional one. The parameter c' is necessary here to exactly determine a and b. However is its value not used in the succeeding calculations.

If one minimizes χ^2 from equation (5.5) by a, b and c' according to

$$\frac{\partial\chi^2}{\partial a} = 0, \qquad \frac{\partial\chi^2}{\partial b} = 0, \qquad \frac{\partial\chi^2}{\partial c'} = 0,$$

one gets a linear equation system in the three variables with the coefficients:

$$\left(\begin{array}{lll|l} \sum_i 1 & \sum_i x_i & -\sum_i(-1)^i x_i & \sum_i y_i' \\ \sum_i x_i & \sum_i x_i^2 & -\sum_i(-1)^i x_i^2 & \sum_i x_i y_i' \\ \sum_i(-1)^i x_i & \sum_i(-1)^i x_i^2 & -\sum_i x_i^2 & \sum_i(-1)^i x_i y_i' \end{array}\right)$$

If one resolves to a and b then for the coefficients is

$$a = \frac{1}{D}\begin{vmatrix} \sum_i y_i' & \sum_i x_i & -\sum_i(-1)^i x_i \\ \sum_i x_i y_i' & \sum_i x_i^2 & -\sum_i(-1)^i x_i^2 \\ \sum_i(-1)^i x_i y_i' & \sum_i(-1)^i x_i^2 & -\sum_i x_i^2 \end{vmatrix} \tag{5.6}$$

$$b = \frac{1}{D}\begin{vmatrix} \sum_i 1 & \sum_i y_i' & -\sum_i(-1)^i x_i \\ \sum_i x_i & \sum_i x_i y_i' & -\sum_i(-1)^i x_i^2 \\ \sum_i(-1)^i x_i & \sum_i(-1)^i x_i y_i' & -\sum_i x_i^2 \end{vmatrix} \tag{5.7}$$

with

$$D = \begin{vmatrix} \sum_i 1 & \sum_i x_i & -\sum_i(-1)^i x_i \\ \sum_i x_i & \sum_i x_i^2 & -\sum_i(-1)^i x_i^2 \\ \sum_i(-1)^i x_i & \sum_i(-1)^i x_i^2 & -\sum_i x_i^2 \end{vmatrix}$$

As the values for y_i are variable and quasi-continuous the x_i are determined constant by the detector geometry. The sums that contain no y_i are hence only depending on the index number $\mathfrak{I}$.

The composition of the set $\mathfrak{I}$ can be coded by a number k that a 1 at the n position of the binary notation of k means that the track segment at plane n is present. For $k = 23 = 010111_2$ hence for example $\mathfrak{I}_{23} = (0, 1, 2, 4)$. For a valid track candidate there are at least four of the six track segments required. Hence by a combinatorial way of thinking that there are 22 valid values for k respectively possible sets $\mathfrak{I}_k$.

From y_i independent terms can be pre-calculated and for all values for k be put in a look-up table. Rearranging from (5.6) and (5.7) results in

$$a = \sum_{i\in\mathfrak{I}_k} a_{ki} y_i', \qquad b = \sum_{i\in\mathfrak{I}_k} b_{ki} y_i' \tag{5.8}$$

plane $i =$	0	1	2	3	4	5
$a_{15,i} =$	-6,68	-6,41	6,16	5,93		
$a_{23,i} =$	-6,76	-0,00	-0,33		6,10	
$a_{27,i} =$	-5,54	-2,68		2,48	4,74	
$a_{29,i} =$	-6,76		-0,33	0,00	6,10	
$a_{30,i} =$		-6,93	-6,66	6,41	6,18	
$a_{31,i} =$	-5,41	-2,68	-0,27	2,48	4,87	
$a_{39,i} =$	-2,92	-5,55	2,69			4,78
$a_{43,i} =$	0,00	-7,01		-0,33		6,35
$a_{45,i} =$	-7,16		6,60	-6,36		5,92
$a_{46,i} =$		-7,01	0,00	-0,33		6,35
$a_{47,i} =$	-2,91	-5,42	2,68	-0,26		4,91
$a_{51,i} =$	-3,60	-3,45			3,08	2,97
$a_{53,i} =$	-6,76		-0,33		6,10	-0,00
$a_{54,i} =$		-5,74	-2,78		2,58	4,94
$a_{55,i} =$	-3,51	-3,45	-0,17		3,17	2,97
$a_{57,i} =$	-5,70			-2,55	4,87	2,37
$a_{58,i} =$		-7,01		-0,33	-0,00	6,35
$a_{59,i} =$	-3,59	-3,38		-0,16	3,07	3,06
$a_{60,i} =$			-7,18	-6,92	6,66	6,43
$a_{61,i} =$	-5,56		-0,27	-2,55	5,01	2,37
$a_{62,i} =$		-5,61	-2,78	-0,27	2,58	5,07
$a_{63,i} =$	-3,51	-3,37	-0,17	-0,16	3,16	3,05

Table 5.1: The coefficients a_{ki} for the linear regression. The values are without unit.

plane $i =$	0	1	2	3	4	5
$b_{15,i} =$	2,02	1,94	-2,02	-1,95		
$b_{23,i} =$	2,08	-0,16	0,05		-1,97	
$b_{27,i} =$	1,63	0,75		-0,84	-1,54	
$b_{29,i} =$	2,00		0,05	-0,15	-1,90	
$b_{30,i} =$		2,02	1,94	-2,02	-1,95	
$b_{31,i} =$	1,62	0,75	0,03	-0,84	-1,56	
$b_{39,i} =$	0,82	1,64	-0,91			-1,55
$b_{43,i} =$	-0,17	2,17		0,06		-2,05
$b_{45,i} =$	2,09		-2,09	1,86		-1,87
$b_{46,i} =$		2,07	-0,15	0,05		-1,97
$b_{47,i} =$	0,82	1,63	-0,91	0,03		-1,57
$b_{51,i} =$	1,01	0,97			-1,01	-0,97
$b_{53,i} =$	1,93		0,05		-1,83	-0,14
$b_{54,i} =$		1,63	0,75		-0,84	-1,54
$b_{55,i} =$	1,01	0,97	0,00		-1,01	-0,97
$b_{57,i} =$	1,62			0,69	-1,53	-0,78
$b_{58,i} =$		2,00		0,05	-0,14	-1,90
$b_{59,i} =$	1,01	0,97		-0,00	-1,01	-0,97
$b_{60,i} =$			2,02	1,95	-2,02	-1,95
$b_{61,i} =$	1,60		0,03	0,69	-1,54	-0,78
$b_{62,i} =$		1,62	0,75	0,03	-0,84	-1,56
$b_{63,i} =$	1,01	0,97	0,00	-0,00	-1,01	-0,97

Table 5.2: The coefficients b_{ki} for the linear regression. The values are in the unit m^{-1} quoted.

with

$$a_{ki} = \frac{1}{D}\left[\left(\sum_{j\in\Im_k}(-1)^j x_j^2 \cdot \sum_{j\in\Im_k}(-1)^j x_j^2 - \sum_{j\in\Im_k} x_j^2 \cdot \sum_{j\in\Im_k} x_j^2\right)\right.$$

$$+\left(\sum_{j\in\Im_k} x_j \cdot \sum_{j\in\Im_k} x_j^2 - \sum_{j\in\Im_k}(-1)^j x_j^2 \cdot \sum_{j\in\Im_k}(-1)^j x_j\right) \cdot x_i$$

$$\left.+\left(\sum_{j\in\Im_k} x_j^2 \cdot \sum_{j\in\Im_k}(-1)^j x_j - \sum_{j\in\Im_k} x_j \cdot \sum_{j\in\Im_k}(-1)^j x_j^2\right) \cdot (-1)^i \cdot x_i\right]$$

$$b_{ki} = \frac{1}{D}\left[\left(\sum_{j\in\Im_k} x_j^2 \cdot \sum_{j\in\Im_k} x_j - \sum_{j\in\Im_k}(-1)^j x_j \cdot \sum_{j\in\Im_k}(-1)^j x_j^2\right)\right.$$

$$+\left(\sum_{j\in\Im_k}(-1)^j x_j \cdot \sum_{j\in\Im_k}(-1)^j x_j - \sum_{j\in\Im_k} x_j^2 \cdot \sum_{j\in\Im_k} 1\right) \cdot x_i$$

$$\left.+\left(\sum_{j\in\Im_k} 1 \cdot \sum_{j\in\Im_k}(-1)^j x_j^2 - \sum_{j\in\Im_k} x_j \cdot \sum_{j\in\Im_k}(-1)^j x_j\right) \cdot (-1)^i \cdot x_i\right]$$

For the 22 possibilities now the coefficients a_{ki} and b_{ki} are pre-calculated. The values are to be seen in table 5.1 and 5.2. With this the values for the coefficients a and b from the y_i' according to (5.8) by at the most six multiplications each and five additions.

5.1.1 Estimation of the Intersection Points between the Even and the Circle Orbit

One reason for the inexactness of this procedure is that the adjusted even is neither a tangent nor a secant. It is rather in between the two lines (see Figure 5.2). For the precise reconstruction of the circle it would be necessary to know the x-coordinate of the intersection points between the adjusted even and the circle orbit. The "correct" choice of the points x_1, x_2 is not only dependent on the fact where the plane for the track segment for the calculation are taken from but also from the radius of the Circle orbit. Therefore it is not possible to determine the points in a trivial way. In the simulation good results are acquired with the two experimental approximation formulae

$$x_1 = x_{\mathrm{Layer}\,j} + x_{\mathrm{d}} \cdot \frac{n_{\mathrm{Hits}} - 1}{6} \qquad\qquad x_2 = x_{\mathrm{Layer}\,k} - x_{\mathrm{d}} \cdot \frac{n_{\mathrm{Hits}} - 1}{6} \tag{5.9}$$

with

j : Detector plane of the most inner track segment used

k : Detector plane of the most outer track segment used

n_{Hits} : Number of the track segments used

$x_{\mathrm{Layer}\,i}$: x-coordinate of the drift chamber outside in plane i

x_{d} : Distance of the two drift chamber outsides.

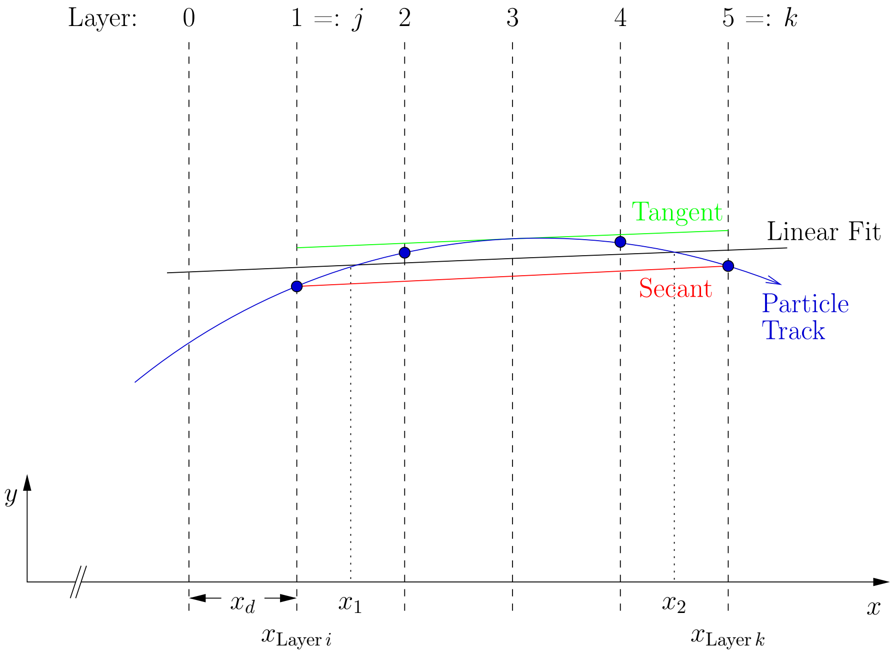

Figure 5.2: The compensation even through the track segments is between tangent and secant. The “correct” intersection points x_1, x_2 can not be determined in a trivial way. The curvature of the particle orbit is shown very much exaggerated to put it up more clearly.

As the radius of the particle orbit is very large the quantitative influence of the choice of x_1 and x_2 not very strong as is expected. If one uses the approximation quoted instead of the more simple estimation $x_1 = x_{\mathrm{Layer}\,j}, x_2 = x_{\mathrm{Layer}\,k}$ in the simulation the p_t-resolution is elevated by 1,0 % and the systematic divergence of the middle value of the error distribution from zero is decreased by 0,022 GeV/c (27 %).

5.2 Computation of the Circle Radius

From the three points $(0,0)$, (x_1,y_1) and (x_2,y_2) the radius of the circle orbit can be determined [Vul03]. According to Figure 5.3 (upper right angled triangle):

$$r = \frac{d_{12}/2}{\sin(\alpha)} \tag{5.10}$$

with

$$\alpha = \varphi_2 - \varphi_1 = \arctan\left(\frac{y_2}{x_2}\right) - \arctan\left(\frac{y_1}{x_1}\right) \tag{5.11}$$

and

$$\begin{gathered} d_{12} = \sqrt{(x_2-x_1)^2 + (y_2-y_1)^2} \\ y_1 = a + b \cdot x_1 \qquad\qquad y_2 = a + b \cdot x_2 \end{gathered} \tag{5.12}$$

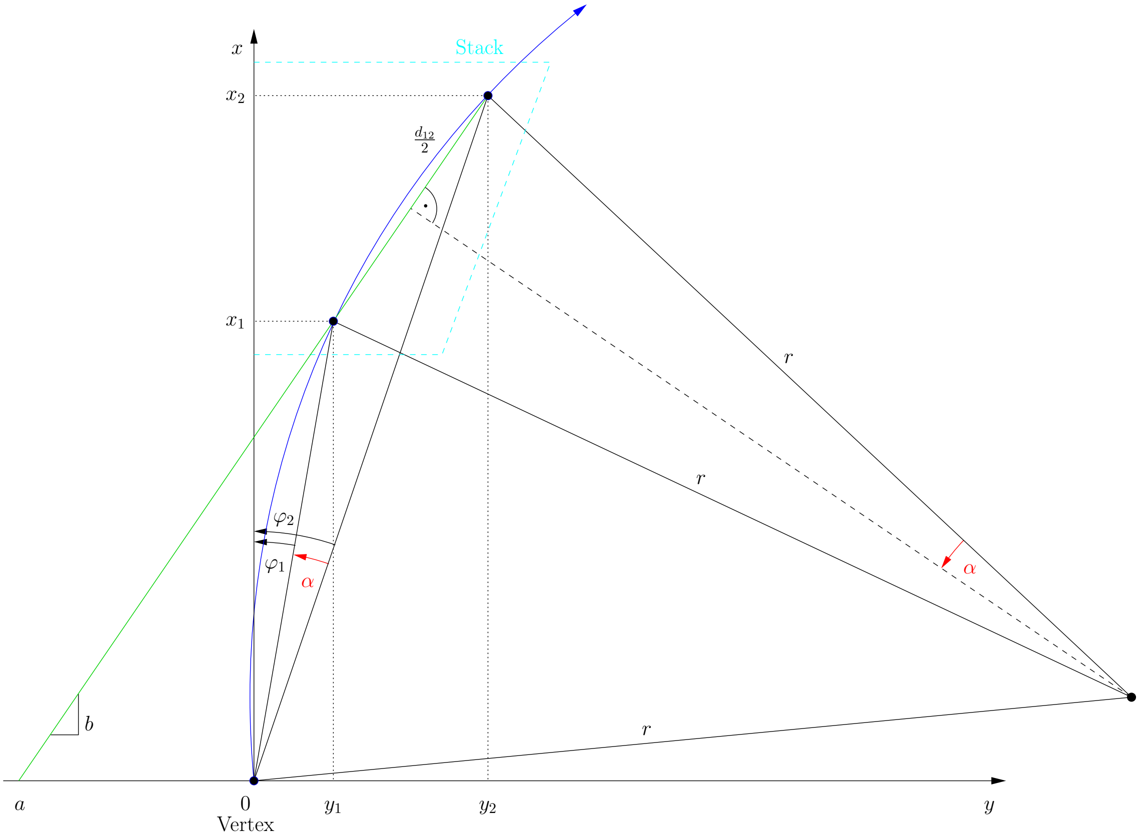

Figure 5.3: Draft of the calculation of the transverse momentum at GTU level.

With the help of this formula the radius of the particle orbit from the two coefficients a and b can be reconstructed. If the reconstructed orbit radius is r one gets the transverse momentum p_{t} of the particle according to (3.7) by multiplication with a constant.

5.2.1 Possible Simplifications at the Computing

The computing of the orbit radius according to the formula given seems to need much effort at the time given for the realization of a hardware. Therefore we try to simplify the computing by approximation without deteriorating the result significantly.

In order to test the procedures for the reconstruction of the transverse momentum for each change the simulation was completely conveyed and the error distribution of the reconstructed transverse momentum was analyzed. The width and the middle value of this distribution give an estimation for the exactitude respectively a systematic shifting during the computing.

The distance d_{12} can be written according to (5.12) as

$$d_{12} = \sqrt{1 + b^2} \cdot (x_2 - x_1)$$

There is almost no loss of exactitude if the root is developed to

$$\sqrt{1+x} \approx 1 + \frac{1}{2}x \qquad \text{(for small } x\text{)}$$

. As the angle α in the experiment is very small there is also no loss by estimation

$$\sin(\alpha) \approx \alpha \qquad \text{(for small } \alpha\text{)}$$

. The calculation is now:

$$d'_{12} = (x_2 - x_1)\left(1 + \frac{b^2}{2}\right) \tag{5.13}$$

$$\varphi_1 = \arctan\left(\frac{a}{x_1} + b\right) \qquad \varphi_2 = \arctan\left(\frac{a}{x_2} + b\right) \tag{5.14}$$

$$r' = \frac{d'_{12}}{2 \cdot (\varphi_2 - \varphi_1)} \tag{5.15}$$

Up to here no worsening of the results in the simulation can be recognized.

The estimation $\arctan(x) \approx x$ which is valid for very small x would lead to a radical simplification however, generates much worse results. It is better to develop $\alpha = \varphi_2 - \varphi_1$ according to Taylor. This is:

$$\alpha = \varphi_2 - \varphi_1 = a(b^2 - 1) \cdot \frac{x_2 - x_1}{x_1 x_2} + a^2 b \cdot \frac{x_2^2 - x_1^2}{x_1^2 x_2^2} + \ldots$$

This by re-arranging the computing is:

$$c_1 = \frac{x_1 x_2}{2} \qquad c_2 = \frac{x_1 + x_2}{x_1 x_2} \tag{5.16}$$

$$r'' = c_1 \cdot \frac{\frac{b^2}{2} + 1}{a(b^2 - 1) + a^2 b c_2} \tag{5.17}$$

The constants c_1 and c_2 are then only depending on the distribution of the segments onto the planes. The possible 22 values each c_{1k} and c_{2k} are made into the form of a table. The calculation of the radius is restored to the calculation operations of addition and multiplication as well as a division. If the radius shall merely be compared to a threshold value the division can be omitted.

Nevertheless a further simplification shall be examined. The values of the slope b in the simulation are as expected relatively narrow distributed around the value zero. If one neglects the slope completely it results in the simple coherence:

$$r''' = \lim_{b \to 0} r'' = -\frac{c_1}{a} \tag{5.18}$$

Therefore the look-up table for b_{ki} and c_{2k} can completely be omitted. The calculation is besides the multiplications for the determination of a from the y_i reduced to a few table operations.

However the p_t-resolution is worse in the variant of simulation which is close to the hardware configuration only by 5 % of 0,132 GeV/c. It occurs an additional systematic shifting of the error middle value of −0,073 GeV/c to −0,125 GeV/c which can be compensated by the addition of a constant in the mean.

In relation to the trigger function of the GTU the result is adequate. Therefore this algorithm is used for the implementation of the prototype.

5.3 Architecture of the Reconstruction Unit

In the reconstruction unit the computing presented before are shown implemented as a combination from arithmetic operations and table accesses. The exactitude respectively bit width of the signals and look-up table values here are chosen according to the simulation so that there is no proof deterioration of the results.

The unit consists of several pipeline levels that are completely separated by registers so that at each clock cycle one track can be taken for processing. The depth of the pipeline depends on the kind of results desired. Figure 5.4 shows the first four pipeline levels of the reconstruction unit.

In the first pipeline level(level 0) the reconstruction unit gets a track data set that consists merely of the memory addresses of the contributing track segments and a bit vector `hit_mask` that quotes which of the six segments definitely belong to the track. The segment addresses are transmitted to the memories in the input units which retransmit the values which were put there at the beginning `y′` and `P`[1]. The 6-bit vector `hit_mask` can not take all combinatorially possible 64 values as at least four segments are required for

`hit_mask`	`mask_id`	`hit_mask`	`mask_id`
001111_2 (15)	0	110011_2 (51)	11
010111_2 (23)	1	110101_2 (53)	12
011011_2 (27)	2	110110_2 (54)	13
011101_2 (29)	3	110111_2 (55)	14
011110_2 (30)	4	111001_2 (57)	15
011111_2 (31)	5	111010_2 (58)	16
100111_2 (39)	6	111011_2 (59)	17
101011_2 (43)	7	111100_2 (60)	18
101101_2 (45)	8	111101_2 (61)	19
101110_2 (46)	9	111110_2 (62)	20
101111_2 (47)	10	111111_2 (63)	21

Table 5.3: The Figure of the combinatorial possible values for the bit vector `hit_mask` which quotes the detector planes which contribute to a track on a continuous code number.

[1] The electron probability `P` of the segments is not processed in this prototype design since the particle identification is not yet stipulated. It is imaginable to multiply the probability of the contributing segments and to compare with a threshold value fixed before.

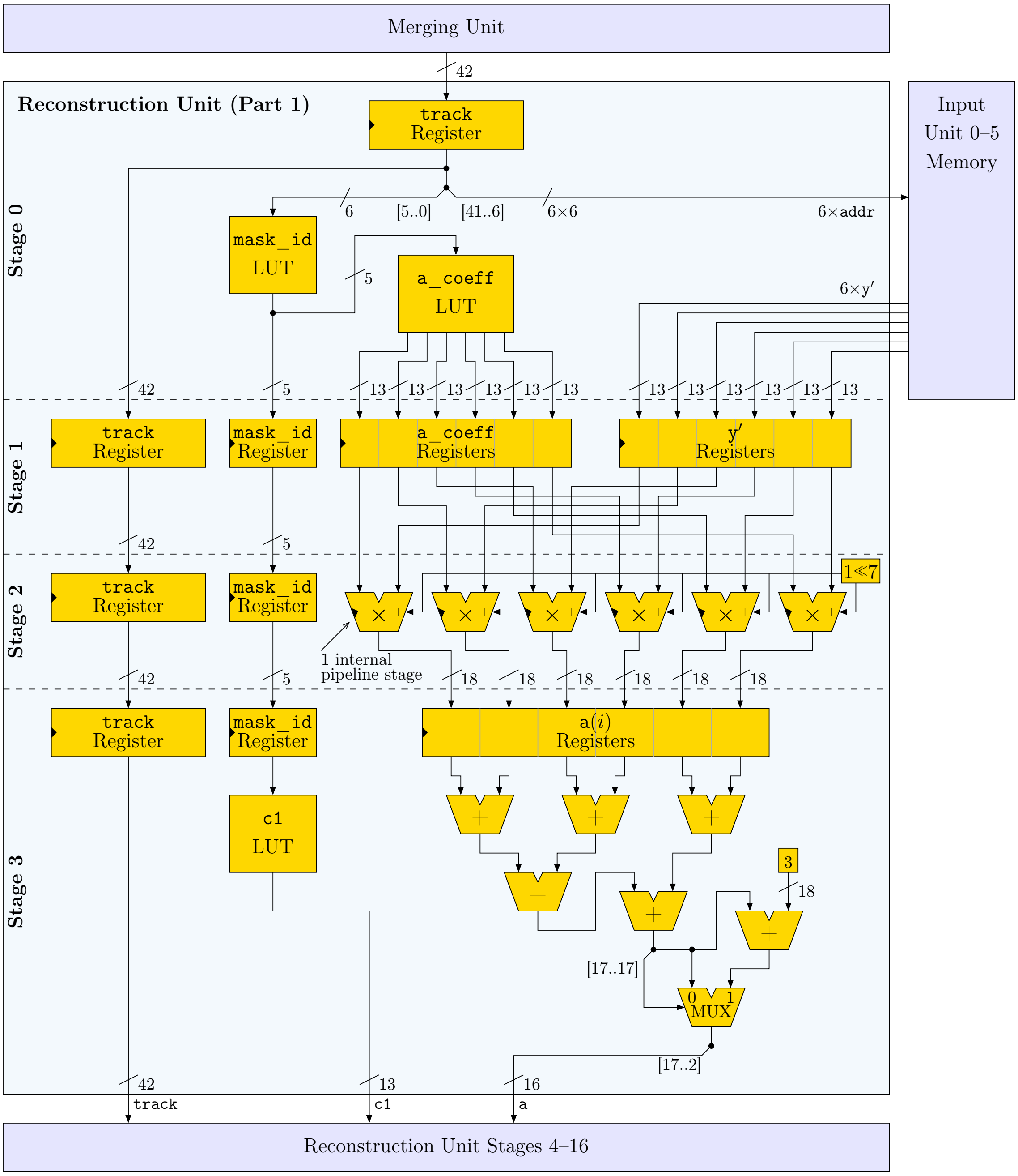

Figure 5.4: The set up of the first four pipeline levels of the reconstruction unit. For a track data set the relevant reconstruction parameters from the input units are read out. The parameter a is calculated which is equivalent to the intercept of the axis of a compensating even by the track segments. From the values c1 and a the transverse momentum can be calculated.

a track. The vector is first transferred by a constant values table (see Table 5.3) into a code number `mask_id` which only numbers the combinations which are really possible. The new signal `mask_id` is only 5 bit wide and is used a an index for the access to the greater look-up tables (LUTs). Already in the same clock cycle a LUT delivers the values `a_coeff`(i) for the six planes. These are coefficients for the linear regression according to Table 5.1 which scales to

$$\texttt{a\_coeff}(\texttt{mask\_id})(i) := 2^9 \cdot a_{ki}$$

and was changed into 13-bit integer numbers.

In the following pipeline levels 1 and 2 the products $\texttt{a\_coeff}(i) \cdot \texttt{y}'(i)$ are generated. As the 13-bit multiplication can not be calculated within one clock cycle with this technique multipliers with an internal pipeline level are used. From the result only the front 18 bit as $\texttt{a}(i)$ are used. Before the last 8 positions are cut off a one is added at position 7 which causes the after decimal points to become a round off.

At level 3 the products calculated right before are added. The sum is reduced only by the last two digits. The calculation is equivalent to equation (5.8, left). Thereafter the sum is reduce by the last two digits. Before the number is increased by three if it is negative. That prevents the sum of the number from being changed depending on the sign if the last two digits are cut off which would lead to a systematic shifting of the results. For the sum calculated `a` is

$$\texttt{a} = a \cdot \frac{2^9}{160\,\mu\text{m} \cdot 2^8 \cdot 2^2} = \frac{a}{320\,\mu\text{m}}$$

with the axis particle a of the compensation even.

Parallel to this on level 3 the 13-bit constant `c1` is read out from a LUT (see Table 5.4) which is calculated according to

$$\texttt{c1}(\texttt{mask\_id}) := -\frac{\text{e} \cdot B}{320\,\mu\text{m} \cdot \text{GeV/c}} \cdot c_1 = -\frac{375}{m^2} \cdot \frac{x_1 x_2}{2}.$$

From `a` and `c1` the transverse momentum p_t can be

$$p_\text{t} = \frac{\texttt{c1}}{\texttt{a}}\,\text{GeV/c}$$

id	c1	id	c1	id	c1	id	c1
0	-1890	6	-2034	12	-2034	18	-2202
1	-1962	7	-2034	13	-2118	19	-2036
2	-1962	8	-2034	14	-2036	20	-2119
3	-1962	9	-2118	15	-2034	21	-2037
4	-2043	10	-2036	16	-2118		
5	-1963	11	-2034	17	-2036		

Table 5.4: The values for the constant `c1` as they are used in the hardware draft. The value is exclusively depending on the contributing detector planes.

calculated as can be seen by comparison of equation (3.7) and (5.18).

The set up of the following pipeline level depends on the kind of the results desired. If only the amount of the tracks shall be registered of which the transverse momentum is above a certain threshold value then the division can be omitted. Then the value from `a` can be multiplied with the constant threshold value and the result can be compared with `c1`.

If however further calculations shall be carried out like the reconstruction of the invariant mass of the original particle from an electron-positron-track pair the division must be carried out and the transverse momentum ought to be calculated. This variant furthermore allows a direct verification of the result in the simulation and therefore is described here.

The set up of the pipeline levels 4–16 is shown in Figure 5.5. For the division the numerator `c1` at first is augmented by eight digits. At the actual technique and a clock rate of 40 MHz a dividend with 11 internal pipeline levels is necessary. Only the most significant 17 bit of the results are used.

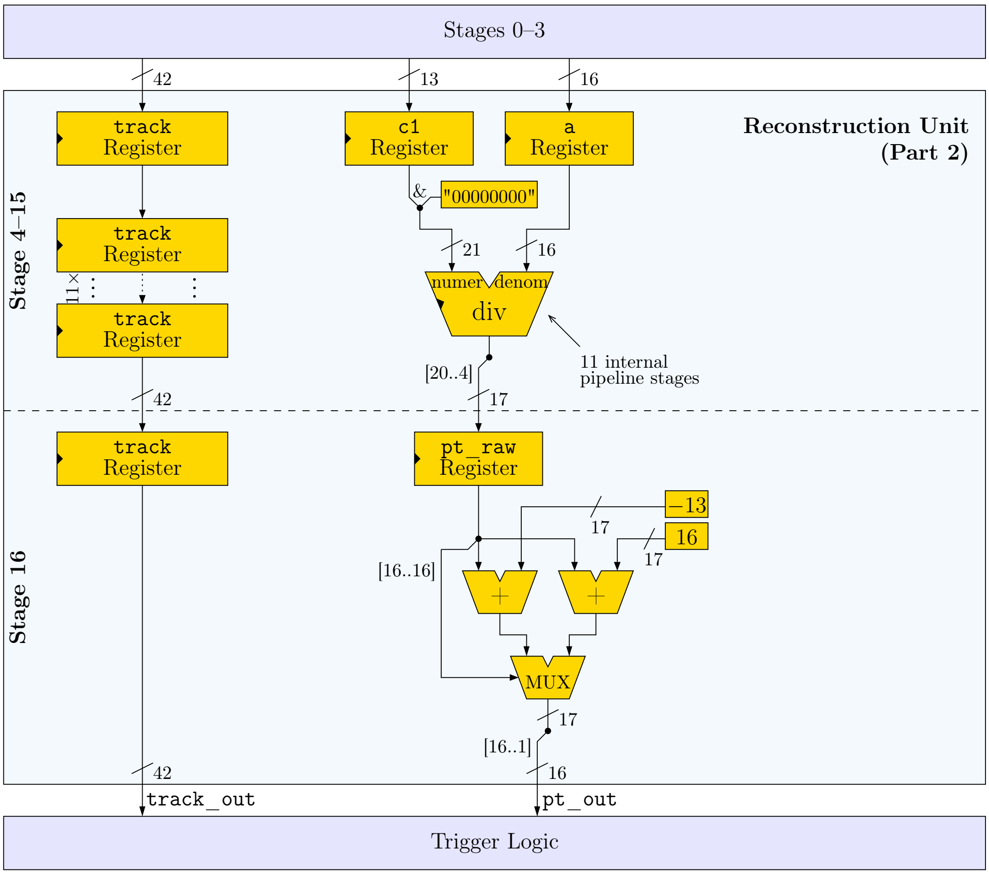

Figure 5.5: The set up of the pipeline levels 4–16 of the reconstruction unit. From the values `c1` and `a` of the former level the transverse momentum is calculated by division. If the exact value is not required the time consuming division can be omitted and this part of the reconstruction unit can be left apart.

At level 16 afterwards the calculated transverse momentum is corrected by a small additive constant depending on its sign and reduced to 16 binary digits. By the correction a systematic shifting is reduced which occurs because of the use of the large approximation formula (5.18) in the middle.[2] The result `pt_out` gives the reconstructed transverse momentum as a fixed point with seven binary fixed point digits in the unit GeV/c:

$$p_{\mathrm{t}} = \frac{\texttt{pt\_out}}{2^7}\,\mathrm{GeV/c}.$$

The description of the TMU set up unit is hence complete. In the following chapter the efficiency and the time behavior of the design as well as the required space after the mapping on a FPGA is examined.

[2] The two correction values are different from one another in the sum to even out the sign depending value shifting in the middle which occurs because of the cut off of the after decimal point digits.

6 Results

This chapter summarizes the results of various simulations and analysis. In the first paragraph it is demonstrated by the results of the root simulation how well the developed system can fulfil the requirements of efficiency and exactitude. In the second paragraph the time behavior and size of the hardware implementation is analyzed.

6.1 Results of the Simulation in Root

In the following some statistical results of the simulation are presented. For the complete trigger decision there is so little time available that large amounts of calculations and especially iterative procedures like the Kalman filtering can not be done. The detection efficiency for this reason is not measured by the efficiency of the off-line analysis.

It is of decisive importance how high the probability is that a searched particle which passes the detector is registered (efficiency). Important is also the probability of a background event. It quotes how many times another effect is falsely taken for a particle searched by the detector. If the background rate is to large the trigger would be constantly activated and hence be useless. Finally the quality of the reconstruction is important for the particles found. It shows the exactitude (resolution) of the determination of the character of the particles.

In order to test various procedures the simulation was implemented in several variants the results of which are only slightly different. This paragraph presents only the result of the variant of which the hardware implementation of chapters 4 and **??** is deducted. During the realization of the calculation procedure into a hardware architecture that is as fast as possible only the algorithm was slightly varied and optimized. The additional losses which are caused by the changes for example are however only in the range of round off errors so that the significant characteristics of the model in the root simulation are visible.

6.1.1 Evidence Efficiency

Table 6.1 gives a detailed overview of the simulation results for the recognition of electron tracks in the GTU. The table shows the amount of track segments of a track which are transmitted to the GTU [1] and has all together ten columns. By the first two columns (from

[1] tracks with 7 and 8 segments can occur if a particle generates ionization tracks in a detector module at two neighboring pad-rows. The GTU in such a cases uses always only one of the two track segments.

tracks with 4 segments										
$\mathrm{d}N_{\mathrm{ch}}/\mathrm{d}\eta =$			0		2000		4000		8000	
total			171		150		256		325	
fnd.	+0	−0	138	80,7 %	103	68,7 %	154	60,2 %	116	35,7 %
	+1	−0	8	5,3 %	3	2,0 %	6	3,1 %	8	3,4 %
		−1	1		0		2		3	
not found			24	14,0 %	44	29,3 %	94	36,7 %	198	60,9 %
tracks with 5 segments										
$\mathrm{d}N_{\mathrm{ch}}/\mathrm{d}\eta =$			0		2000		4000		8000	
total			621		371		393		328	
found	+0	−0	541	96,5 %	239	90,6 %	185	82,4 %	106	64,9 %
		−1	58		97		139		107	
	+1	−0	2	0,5 %	1	0,8 %	0	2,3 %	2	3,7 %
		−1	0		2		7		3	
		−2	1		0		1		7	
	+2	−2	0		0		1		0	
not found			19	3,1 %	32	8,6 %	60	15,3 %	103	31,4 %
track with 6 segments										
$\mathrm{d}N_{\mathrm{ch}}/\mathrm{d}\eta =$			0		2000		4000		8000	
total			1769		425		289		109	
found	+0	−0	1477	98,0 %	262	95,8 %	123	90,7 %	19	70,6 %
		−1	196		121		100		33	
		−2	61		24		39		25	
	+1	−1	1	0,1 %	4	1,6 %	1	2,0 %	1	2,8 %
		−2	0		1		3		1	
		−3	0		2		1		1	
	+2	−3	0		0		1		0	
not found			34	1,9 %	11	2,6 %	21	7,3 %	29	26,6 %
tracks with 7 segments										
$\mathrm{d}N_{\mathrm{ch}}/\mathrm{d}\eta =$			0		2000		4000		8000	
total			65		15		11		3	
fnd.	+0	−1	54	100,0 %	8	100,0 %	5	90,9 %	1	100,0 %
		−2	10		5		4		1	
		−3	1		2		1		1	
not found			0	0,0 %	0	0,0 %	1	9,1 %	0	0,0 %
tracks with 8 segments										
$\mathrm{d}N_{\mathrm{ch}}/\mathrm{d}\eta =$			0		2000		4000		8000	
total			4		0		0		0	
fnd.	+0	−2	2	75,0 %	0	–	0	–	0	–
		−3	0		0		0		0	
		−4	1		0		0		0	
not found			1	25,0 %	0	–	0	–	0	–

Table 6.1: Simulation results for the recognition of particle tracks in the GTU (see text).

left) one line of a constellation is given where the track can be found. The first column shows how many segments which do not belong to the track are falsely added to the track by the GTU (+0, +1, +2). In the second column one can see how many track segments of the track were not found (−0, −1, −2, . . .). The right part of the table shows for each of the constellations how often it occurs in the simulation of the various multiplicities. The combinations that are not shown in the table do not occur in the simulation.

During the evaluation all the electrons and positrons are recognized which are generated at a Υ-decay with a transverse momentum $p_t > 3.0\,\mathrm{GeV/c}$ *mostly uninfluenced* pass the detector and generate a track segment at least at four planes [2]. The criterion *mostly uninfluenced* is if a particle has at least 90 % of its original transverse momentum when entering the detector and furthermore does not loose more than 10 % of its original transverse momentum during the transition of the detector. The analysis is limited to these particles in order to show especially the influence of the GTU to the result. The particle numbers quoted in the table result from the event data sets used during the simulation (see Table 2.1) and of the efficiency up to the plane of the LTUs (see paragraph 6.1.4).

In the simulation there are also heterodyne track segments recognized as they occur often in the TRD at higher multiplicity and as the efficiency of the GTU is defined also by the fact how the heterodyne track segments are handled during the reconstruction. That means at one hand are there segments with a slight deviation correctly attributed and on the other hand how it can be avoided that strongly influenced track segments deteriorate the reconstruction of the other segments. The results do not show the behavior of the isolated GTU under optimal conditions but its way of processing considering the application in the TRD.

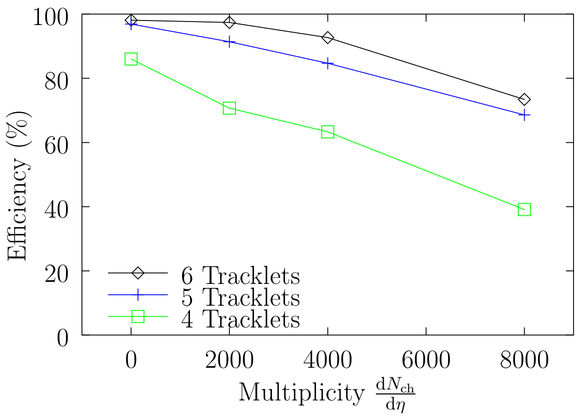

Figure 6.1: Evidence efficiency of the GTU for tracks with four, five, and six segments and various multiplicity densities.

In Figure 6.1 some of the values for the efficiency of the GTU are shown graphically. One can see that the evidence efficiency is declining with increasing multiplicity density $\frac{dN_{ch}}{d\eta}$. Tracks with lesser segments are more effected.

The decline of the evidence probability at higher multiplicity has two main reasons. At

[2] more exactly: "...where most of them contribute to a track segment." In the simulation a generating particle is attributed to each of the track segments. If the tracks are heterodyne the particle with the largest contribution is chosen.

first with the multiplicity the probability increases that there is an irrelevant track segment added to a searched track in the GTU because of falsely and accidentally meeting the window criterion. In this case it is possible that the additional segment falsifies the reconstructed transverse momentum $p_{\mathrm{t}}^{\mathrm{rec}}$ in a way that it is shifted underneath the limit of $p_{\mathrm{t,\,min}}^{\mathrm{GTU}} = 2.7\,\mathrm{GeV/c}$ and therefore the electron searched for can not be found.

At larger multiplicity on the other hand the probability increases that at the LTU plane already a single track segment exists from the heterodyning of two particle tracks. By the influence of the other particle it is quite possible that the slope of the track segment is falsified in a way that is not added to the track by the GTU. Tracks with only four segments are already omitted if one segment is missing which results in a very obvious effect on the efficiency.

At a multiplicity density of $\frac{\mathrm{d}N_{\mathrm{ch}}}{\mathrm{d}\eta} = 2000$ – which is a bit lower than the actual expected one – for example there are 97.4 % from the complete particle tracks found (6 segments). At the tracks with 5 or 4 segments it is still 91.4 % respectively. 70.7 %.

6.1.2 Background Rate

Background events i. e. “false” electron findings mainly base on two effects:

1. A pion with a high transverse momentum is recognized as an electron.
2. An electron with a low transverse momentum does not come from the primary vertex and is hence reconstructed with a transverse momentum which is too high.

The details of the statistic particle determination on the base of the charge measured are not yet defined and shall not be looked at within the frame of this work. The simulation presented here does not recognize the first source of error. The attribution is perfectly accepted.

In the following there is a short description of the influence of the GTU onto the second source or error. For a precise statistical evaluation however the amount of the event data used is not sufficient.

If the primary vertex presumption which is used at the reconstruction of the transverse momentum is not correct than in the procedure used a particle can appear randomly high energetic. The amount of background particles per event found is shown in Table 6.2 for the LTU and GTU plane. In the LTUs the track segments are merely read out on the base of their slope. The equivalent threshold value $p_{\mathrm{t,\,min}}^{\mathrm{LTU}} = 2.3\,\mathrm{GeV/c}$ here is lower than

$\frac{\mathrm{d}N_{\mathrm{ch}}}{\mathrm{d}\eta} =$	0	2000	4000	8000
LTU	0,00	17,80	33,95	68,33
GTU	0,00	1,50	2,85	3,30

Table 6.2: Background particles per event at LTU and GTU plane at various multiplicity densities.

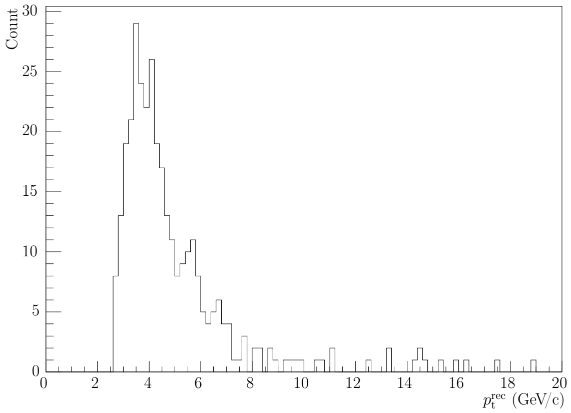

Figure 6.2: Reconstructed transverse momentum for background particles. For a demonstration the data of all multiplicity densities are used.

in the GTU. In the Table there are only these particles at the LTU plane which are at least falsely recognized on four planes and were transmitted to the GTU. On average the particles have a real transverse momentum of $\overline{p_\mathrm{t}} \approx 0.3\,\mathrm{GeV/c}$.

In the GTU the background rate is minimized by two effects. First of all the exactitude of the reconstruction is higher because of the combination of the track segments so that a higher threshold value $p_{\mathrm{t,\,min}}^{\mathrm{GTU}} = 2.7\,\mathrm{GeV/c}$ can be chosen. Therefore the probability is statistically lower that a particle might pass the detector with a falsely fitting slope. Figure 6.2 shows the distribution of the reconstructed transverse momentum $p_\mathrm{t}^{\mathrm{rec}}$ for the background particles. With this one can derive the minimization of the background rate by another increase of $p_{\mathrm{t,\,min}}^{\mathrm{GTU}}$.

Secondly the merging of the track segments leads to a rejection of particles with lower transverse momentum no matter what kind of angle as its orbit is curved to much to meat the window criterion. Therefore the background particles at GTU plane have a significantly higher real mean transverse momentum: $\overline{p_\mathrm{t}} \approx 1.2\,\mathrm{GeV/c}$. To lower the background rate one could lower the window size again (at the expense of the efficiency).

6.1.3 Resolution of the Reconstructed Transverse Momentum

The medium error is looked at as a criterion for the quality of the reconstruction. It makes sense to look at the relative error as this error becomes larger at big transverse momentums (follows at constant error of the axis section a from (5.18)). The distribution of the relative error with relation to the real transverse momentum is shown in Figure 6.3.

Many electrons loose a significant part of their energy before and in the detector for example by absorption radiation so that the original transverse momentum can not exactly be determined but systematically a lower value is derived. In order to examine the p_t-resolution as independent from this effect as possible only electrons and positrons are recognized with a transverse momentum of $p_\mathrm{t} > 3.0\,\mathrm{GeV/c}$ which loose before and in the detector not more than $1\,\%$ of its transverse momentum. Without this limitation one would see a clear asymmetry in Figure 6.3. In order to keep the reconstruction free from disturbances

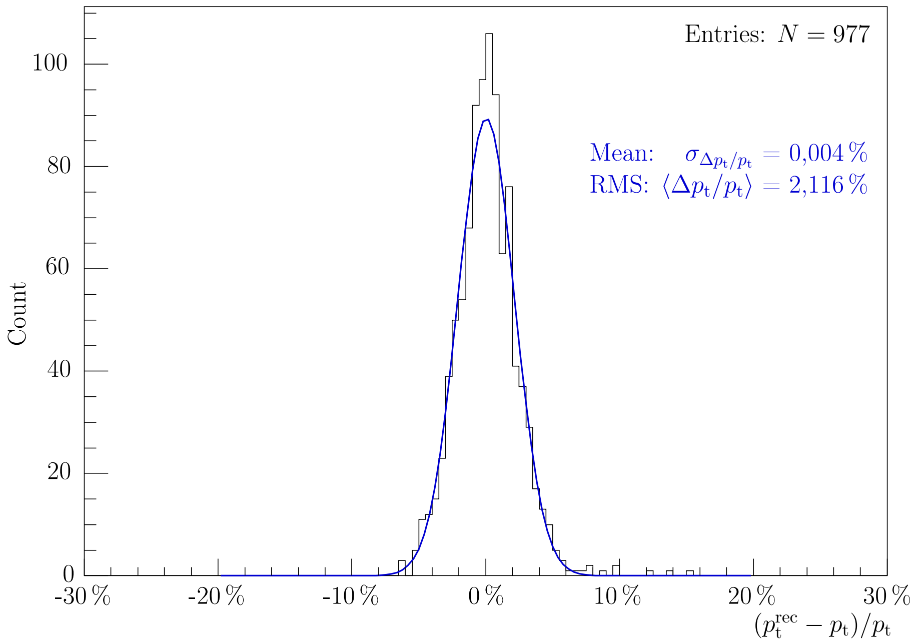

Figure 6.3: Resolution of the transverse momentum reconstruction. A standard distribution is adjusted to the error distribution (blue lines). The mean value and width of the error distribution are a measurement for the exactitude of the reconstruction procedures.

by the background particles we use the simulation data sets with multiplicity densities of $\frac{dN_{ch}}{d\eta} = 0$ (signal data) for the evaluation.

For the comparison a standard distribution is adjusted to the data. The width $\sigma_{\Delta p_t/p_t} = 2,12\,\%$ of the distribution is an estimation for the relative resolution of the reconstructed transverse momentum. The absolute reconstruction error in the simulation is at the mean $\sigma_{\Delta p_t} = 0.13\,\mathrm{GeV/c}$. The results are limited by the threshold of the detector in y-direction and therefore do not individually show the exactitude of the calculations in the GTU.

6.1.4 Analysis of the Complete Detector System

In order to test whether the results of the root simulation for the GTU meet the expectations the evidence efficiency shall be compared with the one of the abstract "theoretical" simulation of the detector which was developed by Dr. B. Vulpescu and which tries as a software simulation to produce the best results by limiting itself to simple procedures.

In contradiction to the simulation presented here which describes the behavior of the hardware-implementation of the GTU in the "theoretical" simulation it is calculated with unlimited exactitude. All sizes are floating point digits. The data transfer is not simulated. The track segments are projected to a cylinder surface instead of onto a plane. The merging of the track segments is implemented as a two levelled process. For the tracks found in the first step their track segments are only corrected by the influence of the "tilted Pads" on the y-coordinate. In the second step again all track segments are examined and the final

	$\mathrm{d}N_{\mathrm{ch}}/\mathrm{d}\eta$			
Segments	0	2000	4000	8000
0–3	31,0	34,7	37,4	49,6
4	6,6	11,7	18,3	22,4
5	18,5	26,2	25,9	20,5
6	42,0	26,2	17,6	7,3
7–8	1,9	1,2	0,9	0,1

	$\mathrm{d}N_{\mathrm{ch}}/\mathrm{d}\eta$			
segments	0	2000	4000	8000
0–3	3,6	8,4	12,3	28,4
4	6,3	14,3	23,7	30,4
5	22,8	35,4	36,3	30,7
6	64,8	40,5	26,7	10,2
7–8	2,5	1,4	1,0	0,3

Table 6.3: Efficiency of the total detector system of the GTU. The table lists the probabilities (in percent) of a certain amount of track segments being transmitted to the GTU. In the right table there are only searched electrons are with a transverse momentum which was not yet significantly reduced of up to the detector.

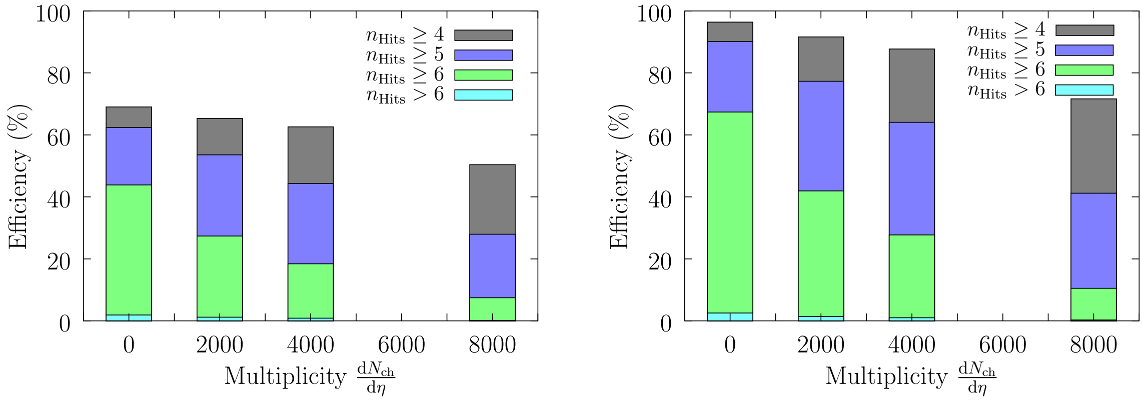

Figure 6.4: Graphical draft of the data from Table 6.3 for the efficiency without recognizing the GTU. In the right diagram only *mostly uninfluenced* electrons are recognized.

tracks are found. In difference to the present simulation the tracks are detected across the stack boundaries. Finally no round offs are used at the reconstruction of the transverse momentum any more.

For comparison with the results of this idealist simulation as they can be found in [ALI01, S. 100] the efficiency for the complete detector system must be examined with all the electrons searched being recognized.

Efficiency up to the plane of LTUs

In Table 6.3 (left) the probabilities are given (in percent) that per electron searched a certain amount of track segments is transmitted to the GTU. The combination of these values with the evidence probability for every track type in the GTU gives the total of the detector efficiency. Particles that are transmitted with 0–3 segments can under no circumstances be detected from the GTU. Its proportion is around $\frac{\mathrm{d}N_{\mathrm{ch}}}{\mathrm{d}\eta} = 0$ (signal data) already 31.0 %.

One of the main reasons for the limited efficiency here too is the loss of energy of the electrons. For comparison in the right table there are the values which are derived if only *mostly uninfluenced* electrons are recognized as is the case in the results for the GTU seen in paragraph 6.1.1. In Figure 6.4 the comparison is shown graphically.

Total Efficiency

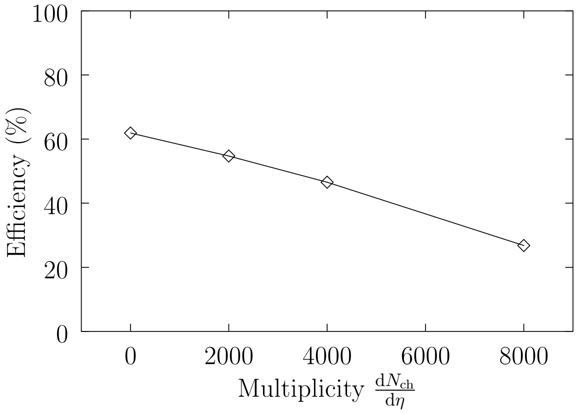

Figure 6.5: Evidence efficiency of the total detector system for primary electrons with $p_t \geq 3.0\,\mathrm{GeV/c}$ at various multiplicity densities.

If one combines the results with the efficiency of the GTU in form of a total simulation than one gets the efficiency process in dependence of the multiplicity density as is shown in Figure 6.5. During the evaluation all electrons and positrons are recognized which during a Υ-decay with a transverse momentum $p_t > 3.0\,\mathrm{GeV/c}$ are generated in the angle range of the detector. The deflection to the comparable "Cut A" in [ALI01, S. 100] is only a few percent whereas the difference is higher for larger multiplicities than for lower.

Obviously no big disadvantages occur because of the simplified procedures used in the hardware related simulation. The deviation can be explained by the differences mentioned above especially the use of round off procedures and the limited exactitude during the transmission and processing of the track segment parameters.

6.2 Analysis of the Hardware Implementation

6.2.1 Time of Processing

The time of processing of the GTU is depending on the clock rate with which it can be run. Furthermore there is no constant amount of computing steps until the result is generated.

The GTU is designed so that each unit forwards the data of the means with as little latency as possible. At this point the time delays can be evened out to a certain degree between the units. Furthermore it can for example be presumed that the maximum of 40 track segments per module at highest multiplicity occur distributed more or less across the chamber. As the GTU was developed for a low time delay under these presumptions it does not make sense to examine the theoretical maximum of the processing time.

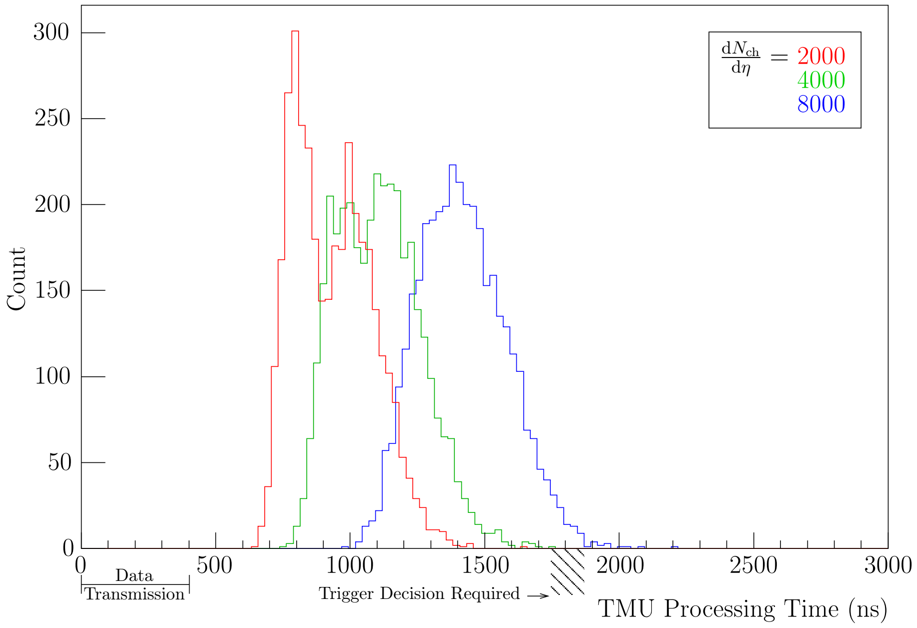

Figure 6.6: The processing time of the TMUs is not constant but depends strongly on the amount of transmitted track segments – and hence on the mean value of the multiplicity density –. The zero stands for the arrival time of the first track segment.The duration of transmission is hence included in the time given.

Instead of this the real processing time for simulated events is examined. Therefore the simulation data sets from the root simulation are taken in form of track segment data words in a functional simulation of the VHDL description. For this reason a clock frequency of the GTU of 40 MHz is presumed.

Figure 6.6 shows the distribution of the processing time of the single TMUs The computing time shown contains the time of the transmission. The balance point of the time scale hence shows the arrival of the first track segment. We presume in the simulation that the track segments are transmitted without interruption. The model simulated is the variant without division (see paragraph 5.3) as the number of clock cycles of the division – if it is really needed – is very much depending on the FPGA model chosen. The processing time does as expected depend very much on the multiplicity. The two peaks which can still be seen at the values for $\frac{\mathrm{d}N_{\mathrm{ch}}}{\mathrm{d}\eta} = 2000$ occur because of the TMUs with zero respectively of a track found.

Decisive for the use in the experiment is the total processing time of the GTU. In order to estimate it we put in Figure 6.7 for each of the 40 events the maximum processing time of all the TMUs. The transmission time is according to the former description included in the quotations. However the time must be added which is needed for the data transfer from the TMUs to the global trigger logic and for its computing.

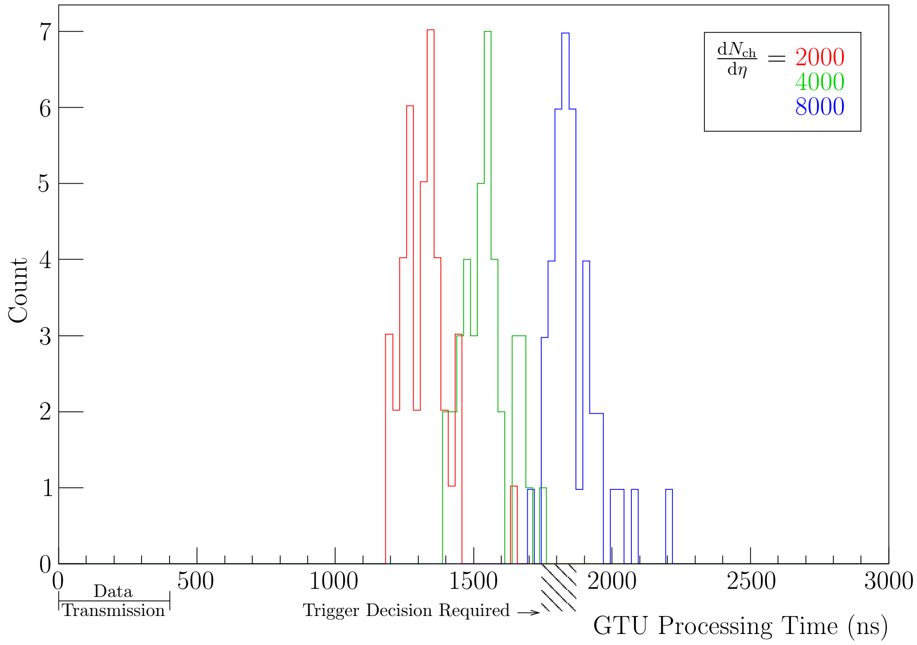

Figure 6.7: The processing time of the GTU results for each event from the maximal processing time from the single TMUs. The balance point of the time scale stands for the arrival of the first track segment.

Up to the multiplicity density of $\frac{dN_{ch}}{d\eta} = 4000$ the computing of all the simulated events can be finished within 1750 ns of the start of the data transfer and is hence within the time frame presumed. At $\frac{dN_{ch}}{d\eta} = 8000$ the computing is already finished in 36 from 40 cases in less than 2 µs.

6.3 Synthesis for FPGAs

During the mapping of the VHDL description onto an FPGA (*synthesis*, see section 2.2) the maximum clock rate with which the unit can be processed does occur besides others. Furthermore one gets a measurement for the size of the design through the amount of the logic cells used.

Portability. For the set up of the GTU the FPGA parts of several manufacturers can be used. Because of the deterioration of prices of the FPGAs the decision for a certain FPGA type shall be made as late as possible. Therefore during the development of the GTU design it is important to make the VHDL description portable so that it can be compiled for various FPGA models without big changes.

This principle is considered at the implementation of the architecture. Therefore most the time it is dispensed with the use of pre-manufactured modules of manufacturer specific design libraries in the VHDL description. Certain functions of the FPGAs however can only be used from the libraries of the individual manufacturer. In some parts therefore we used manufacturer specific components. Here we used the technology depending LPM design library [3] of Altera. The three components used are the Dual-Clock-FIFO-memory at the input levels (components, lpm_fifo_dc) and the arithmetical operation units with internal pipeline levels in the reconstruction unit (lpm_mult and lpm_divide) (see LPM-reference [Alt96]). In order to compile the design for the FPGAs of other manufacturers it is only necessary to exchange equivalent modules of other manufacturers on these three positions.

Manufacturer	Altera
Part	Stratix EP1S30
Part Number	EP1S30F780C5
Logic Cells	28 749 (of 32 470)
Memory (kBit)	210 (of 3 239)
DSP-Block Elements	22 (of 96)
maximum clock rate	45.2 MHz

Table 6.4: Results of the synthesis for an exemplarily chosen FPGA. The use of the part resources is named. The clock rate40 MHz required by the synthesis is achieved.

Results of the Synthesis. In Table 6.4 the results of the synthesis are exemplarily given for a Stratix-FPGA of the manufacturer Altera. At the synthesis a clock rate of 40 MHz is given as a goal. The DSP block elements are specifically for the Stratix series. They are automatically used for the multiplications in the input units and the reconstruction unit.

In selected FPGA the design reaches a maximum clock rate of 45.2 MHz. The planed clock rate of 40 MHz can be achieved. The maximum achievable frequency however depends very much on the parts selected.

The critical path is at the track finding units. It runs in a slope through the memory, the combination logic, the booster logic, and the address register. The register-to-register-delay is maximally 22.125 ns. The capacity at this point could be raised by reading out the memories for the two following values into registers one clock cycle before already so that the memory access disappears from the critical path.

The size of the individual design units is shown in Table 6.5. The total number of logic cells used is the most important size measurement. The demand of memory cells is less critical in this case. Because of the various architectures the amount of the logic cells however is not directly comparable with the information of other manufacturers.[4]

The track finding units all together need as was expected the highest amount of space. The 18 Z-channel units need the second most space but here the size is mainly determined

[3] "Library of Parametric Modules"

[4] The manufacturer Altera offers at present (September 2003) parts with up to 79 040 logic cells.

		logic cells		memory (Bit)	
design part	amount	per unit	all together	per unit	all together
input unit	6	567	3 403	3 392	20 352
Z-channel-unit	18	466	8 383	448	8 064
track finding unit	9	1 548	13 932	19 968	179 712
merging unit	1	1 319	1 319	6 584	6 584
reconstruction unit	1	1 712	1 712	588	588
total sum			28 749		215 300

Table 6.5: The Size of the Individual Design Units of the TMU after the Synthesis.

by the sorters. The size is not that much influenced from the units that are present only ones.

The results show that the complete TMU design at the clock rate planed can be implemented into one (actually available) FPGA.

7 Summary and Outlook

The transition radiation detector of the forthcoming experiment ALICE serves for the spatial resolved evidence of particle tracks, the measuring of momenta, and the identification of particles and especially as a trigger for slower, higher resolving detectors of the experiment. The hierarchical trigger concept of the transition radiation detector uses a separated, massive parallel data processing and reduction directly on the detector chambers which are set up in several cylindrical layers in connection with a central final processing and evaluation in a global track reconstruction unit (GTU). The track segments observed in the chambers are locally parametrisized as segments of an even. It is the duty of the GTU to re-merge the segments from various detector planes that belong together into tracks again and from the flow of these tracks to reconstruct the transverse momentum of the generating particles which is an important criterion for the initiation of the trigger.

The simulation of the GTU function was done in this work and specified and a feasible architecture was presented and implemented as a VHDL-description. The design presented here was verified by simulation and in tests synthesized for FPGA-parts.

A significant criterion at the development of the architecture was that there are less than 40 MHz available at the GTU computing until the trigger decision is made for the up to 20 000 track segments. Considering the realisation into FPGA a resource saving implementation was used. The design is laid out for an easy adjustment for example by putting the data of the detector configuration centrally into a VHDL-file and generating all value tables automatically from few configuration data.

The workload of the GTU is split up to several track finding units that work parallel with various spatial angle ranges. The data are received in input units from the readout network. The even sections from the six detector planes are projected to a common middle plane and for each segment the angle is adjusted to the vertex direction and one y-coordinate is calculated that was adjusted for the track reconstruction. The Z-channel unit controls the track segments implicitly for their accordance to their z-coordinate projected to the middle plane and thereafter assorts to its projected y-coordinate. The nine parallel track finding units are the most complex part of the design and the critical part of the architecture because of its maximum clock rate. Here the projected track segments of the various planes are executed in order to find tracks that belong together starting from a reference plane. It is presumed that track segments belong to the same track if their y-coordinates and angle to the vertex direction are only different by a fixed value. Because of the paralleling during the processing it is possible that the same tracks are found several times. In the merging unit the tracks found from several parallel processing units are compared and double tracks are eliminated. The reconstruction unit finally calculates by a three dimensional linear regression a balancing even by the segments of a track of which the axis particles define the transverse momentum of the generating particle.

The design presented in this work describes the main part of the GTU regarding the complexity. For a complete hardware description of the GTU it is necessary to add the particle identification, control units for the raw data selection, the trigger logic, and the communication with the central trigger processor.

The analysis of the simulation shows that the behaviour of the design meets the requirements with respect to the evidence efficiency and the reconstruction exactitude. During the procedure of this work minor changes were made at the plane for the geometry of the detector which were not noted in this script any more. With the new geometry a slight decrease of the efficiency is expected since the tracks in the GTU are not detected across the stack limits and as in the new (less projective) geometry the angle ranges increase seen from the vertex not all six of the modules of a stack are traversed.

The time limits for the total computing time of the GTU are fulfilled from the design presented here up to the medium multiplicity. At maximum multiplicity the processing time in some cases is to long. In the architecture presented however it is possible to further reduce the number of clock cycles required. For example the merging unit is laid out for dealing with a high amount of tracks found. For the application as an electron/positron-trigger with a few tracks only the computing time can still be reduced by a simple and less efficient unit.

The results of the synthesis at the example of a FPGa of the manufacturer Altera finally show that the design presented can be implemented effectively at a clock rate planned of 40 MHz into FPGAs available.

Bibliography

[ALI01] ALICE Collaboration: ALICE Technical Design Report of the Transition Radiation Detector / CERN. Genf : CERN/LHCC, 2001 (CERN/LHCC 2001-021). – Forschungsbericht. – 246 S. – ISBN 92–9083–184–7

[Alt96] Altera Corporation. *LPM Quick Reference Guide.* URL `http://www.altera.com/literature/catalogs/lpm.pdf`. Dezember 1996

[BR03] Brun, Rene ; Rademakers, Fons. *ROOT Reference Guide.* URL `http://root.cern.ch/root/Reference.html`. 2003

[BV98] Bock, Rudolf K. ; Vasilescu, Angela ; Bonaudi, F. (Hrsg.) ; Fabjan, C. W. (Hrsg.): *The particle detector BriefBook.* Berlin ; Heidelberg ; [u.a.] : Springer, 1998. – 132 S. – ISBN 3–540–64120–3

[FWSW03] Frühwirth, R. ; Waltenberger, W. ; Strandlie, A. ; Wroldsen, J.: A review of fast circle and helix fitting. In: *Nucl. Instrum. Meth.* A502 (2003), S. 705–707

[Gre00] Green, Dan: *The physics of particle detectors.* Cambridge : Cambridge University Press, 2000. – 361 S. – ISBN 0–521–66226–5

[Hri00] Hrivnacova, Ivana. *AliRoot Coding Conventions.* URL `http://alisoft.cern.ch/offline/codingconv.html`. 2000

[Kle92] Kleinknecht, Konrad: *Detektoren für Teilchenstrahlung.* 3., durchges. und erw. Aufl. Stuttgart : Teubner, 1992. – 295 S. – ISBN 3–519–23058–5

[LWS94] Lehmann, Gunther ; Wunder, Bernhard ; Selz, Manfred: *Schaltungsdesign mit VHDL.* Poing : Franzis, 1994. – 317 S. – ISBN 3–7723–6163–3

[Vul03] Vulpescu, Bogdan. *Online tracking: first step to LTU and GTU algorithms.* URL `http://www-aix.gsi.de/~vulpescu/AliPw/Drawings.ps`. 2003