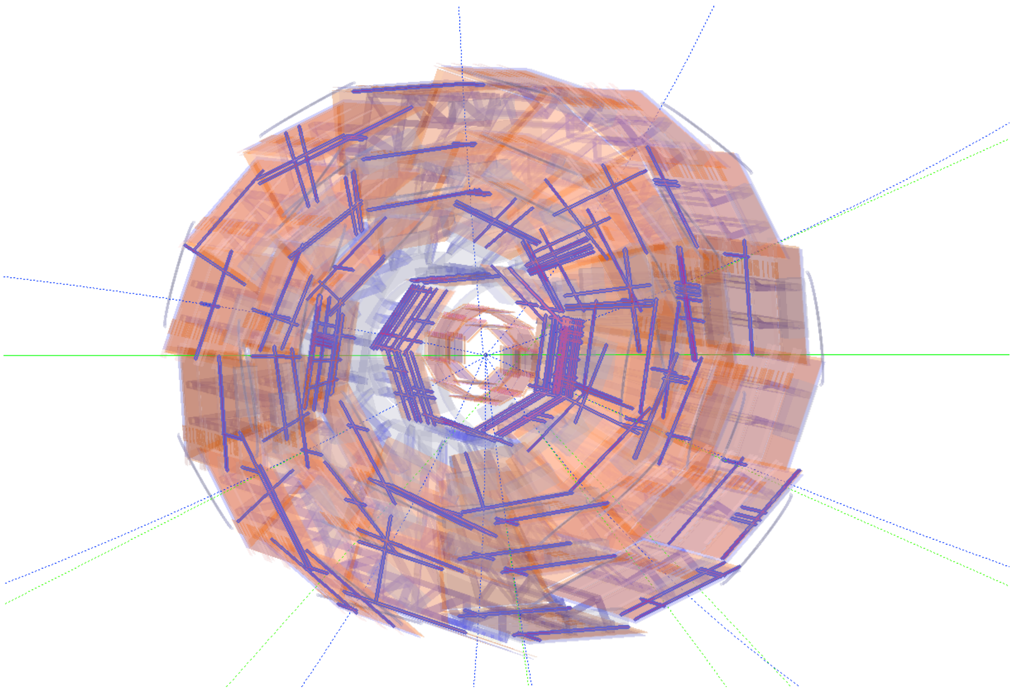
Development of an FPGA-based Data Reduction System for the Belle II DEPFET Pixel Detector



Michael Schnell

Development of an FPGA-based Data Reduction System for the Belle II DEPFET Pixel Detector

Dissertation
zur
Erlangung des Doktorgrades (Dr. rer. nat.)
der
Mathematisch-Naturwissenschaftlichen Fakultät
der
Rheinischen Friedrich-Wilhelms-Universität Bonn

vorgelegt von
Michael Schnell
aus
Bergisch Gladbach

Bonn 2015

Dieser Forschungsbericht wurde als Dissertation von der Mathematisch-Naturwissenschaftlichen Fakultät der Universität Bonn angenommen und ist auf dem Hochschulschriftenserver der ULB Bonn `http://hss.ulb.uni-bonn.de/diss_online` elektronisch publiziert.

1. Gutachter:	Prof. Dr. Jochen Christian Dingfelder
2. Gutachter:	Prof. Dr. Klaus Desch

Tag der Promotion:	15.06.2015
Erscheinungsjahr:	2015

Abstract

The innermost two layers of the Belle II detector at the KEKB collider in Tsukuba, Japan, will be covered by highly granular DEPFET pixel sensors. The large number of pixels leads to a maximum data rate of 256 Gbps, which has to be significantly reduced by the Data Acquisition System. For data reduction, the hit information of the silicon-strip vertex detector surrounding the pixel detector is used to define so-called Regions of Interest (ROI) in the pixel detector. Only hit information of the pixels located inside these ROIs are saved. The ROIs for the pixel detector are computed by reconstructing track segments from strip data and extrapolation to the pixel detector. The goal is to achieve a reduction factor of up to 10 with this ROI selection. All the necessary processing stages, the receiving, decoding and multiplexing of SVD data on 48 optical fibers, the track reconstruction and the definition of the ROIs, will be performed by the DATCON system, developed in the scope of this thesis. The planned hardware design is based on a distributed set of Advanced Mezzanine Cards (AMC), each equipped with a Field Programmable Gate Array (FPGA) and four optical transceivers. An algorithm is developed based on a Hough Transformation, a commonly used pattern recognition method in image processing to identify the track segments in the strip detector and calculation of the track parameters. Using simulations, the performance of the developed algorithms are evaluated. For use in the DATCON system the Hough track reconstruction is implemented on FPGAs. Several tests of the modules required to create the ROIs are performed

in a simulation environment and tested on the AMC hardware. After a line of successful tests, the DATCON prototype was used in two test beam campaigns to verify the concept and practice the integration with the other detector systems.
The developed track reconstruction algorithm shows a high reconstruction efficiency down to low track momenta. A higher data reduction than originally intended was achieved within the limits of the available processing time. The FPGA track reconstruction algorithm is found to be even three times faster than demanded by the trigger rate of the experiment.
The used concepts and developed algorithms are not specifically designed for the Belle II vertex detector only, but can be used in different experiments. It was successfully tested on the low-level trigger for Belle II, using drift chamber information and showed a comparably good track reconstruction performance.

Contents

Introduction

Particle physics experiments show a constant increase in number of readout channels at growing data rates over the past decades. New experiment detectors like Gigatracker [1], a fixed target experiment using pixel sensors with 324 million readout channels at 1 GHz readout rate, the upgrade of the ATLAS experiment with the recently installed Insertable B-Layer (IBL) with 90 million channels [2] or the Star pixel detector [3] with 400 million are only a few example experiments requiring fast readout systems and an efficient data storage to process the large amount of data. The currently upgraded Belle II detector at the High Energy Accelerator Research Organization (KEK) in Japan will be equipped with a highly granular pixel detector with 8 million readout channels. Its first layer is located only 14 mm away from the interaction point and thus will be exposed to high radiation, causing a great quantity of unwanted data. The expected maximum data rate is 256 Gbps at an occupancy of three percent. Because the storage system cannot cope with this amount of data, an online reduction is required.

Usually only a small fraction of the collected data is of interest for physics analysis, therefore every experiment has special constraints to select relevant data and reject unwanted (background) data. In every collision of beam particles, final state particles are produced that create hits in the subdetectors along their trajectories. In addition to the products created in beam collisions, unwanted hits in the detector are created by beam backgrounds, such as photon radiation from the beam, particle collisions with the beam pipe and electronic noise. With algorithms finding patterns in the detector hits, the particle tracks can be reconstructed and by applying constraints on the parameters of these tracks, many background hits can be removed. To perform such a selection online, a system is required which analyses the hit patterns in real-time. Since the number of readout channels of the mentioned pixel detectors is very high, many resources would be required to directly process the data. Another method is using the hit information in the surrounding detectors that typically contain fewer readout channels than pixel detectors, to reconstruct the track information. Based on the calculated properties, the track is extrapolated to the pixel detector. Around this extrapolated track a so-called Region of Interest (ROI) is constructed and only hits inside this ROI are saved, while the rest is discarded. An illustration of this method is depicted in Figure 1.1 demonstrating the concept with a single track.

In this thesis a system for data reduction in the Belle II Pixel Detector (PXD) will be presented by using information from the surrounding Silicon strip Vertex Detector (SVD). To keep the hardware requirements to a minimum the data of the SVD is tapped as close as possible at the sensors electronics. The data are processed by so-called Field Programmable Gate Arrays (FPGA). They are fast, freely programmable, high-density logic chips that can cope with sophisticated algorithms required for pre-processing the detector data, the track reconstruction and the logic necessary to provide the ROI. Hence, a light-weight track reconstruction algorithm is required that matches the parallel processing nature of FPGAs. The system to provide the ROI for the Belle II pixel detector is called Data Acquisition Tracking and Concentrator Online Node (DATCON), build up from scratch in this thesis. In principle, DATCON can also be used for other detectors, such as ATLAS or Compact Muon Solenoid (CMS) at the Large Hadron Collider (LHC). Parts of the FPGA track reconstruction implementation may also be refurbished in other projects, for instance to implement a fast low-level trigger system, also in discussion for the Belle II experiment.
This thesis starts with a description of the SuperKEKB facility and the Belle II detector with a special focus on the vertex

detector. The concept of the track reconstruction, based on a Hough Transformation, and the software implementation as well as performance studies of the DATCON system is described. The remaining chapters are divided into two parts. First, the detector-independent part, the track reconstruction algorithm and the ROI calculation code will be discussed along with a suitability analysis for use on an FPGA. Afterwards a detailed description of the FPGA implementation is provided. The second part describes the detector-dependent pre-processing of the SVD data, so that it can be used by the track reconstruction module. Finally, the first results of a small-scale DATCON prototype tested at two test beam campaigns at the Deutsches Elektronen SYnchroton (DESY) and the European Organization for Nuclear Research (CERN) are presented.

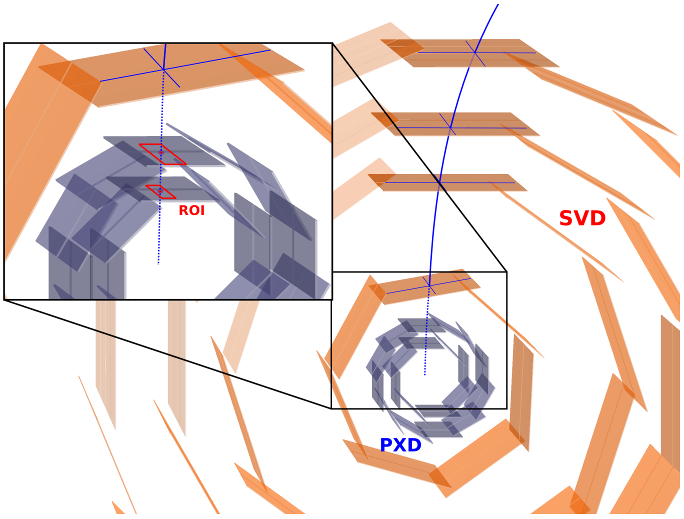


Figure 1.1.: ROI data reduction scheme in the Belle II Vertex Detector.

Upgrade of the KEKB Accelerator

The KEKB [4] accelerator, located at the KEK facility in Tsukuba, Japan, is dedicated to study flavour physics in order to advance the understanding of the weak interaction and the matter-anti-matter asymmetry in the universe. The Belle detector [5] at the asymmetric-energy collider KEKB collected data from e^+e^- collisions at a center-of-mass energy of 10.58 GeV over the past decade and ended its operation in 2010. Large samples of B mesons were collected and are used for precision studies of the weak interaction, in particular the violation of CP symmetry [9], the study of CKM [10] parameters and the search for new physics in rare decays. The KEKB collider currently holds the instantaneous luminosity record of $2.1 \cdot 10^{34}$ cm^{-2}s^{-1} and during the course of about 11 years a total integrated luminosity of about 1 ab^{-1} was delivered.

Deviations from the Standard Model (SM) [11] are searched for by particle physics experiments at two frontiers: the high-energy and the high-intensity (high-luminosity) sector. At the LHC [12] located in Geneva, searches for new particles produced in proton-proton collisions at a center-of-mass energy of 14 TeV are carried out. If a new particle at the LHC should be found, the underlying new physics theory, including its flavour sector, needs to be investigated. For this a new high luminosity machine is required. The KEKB collider, which is currently upgraded to SuperKEKB will provide large samples of B mesons to allow for precision tests of the flavour sector of the SM with high statistics and to look for rare decays. The SuperKEKB collider with its upgraded Belle II detector are described in the next section.

2.1. SuperKEKB

One of the key properties of a high-energy accelerator is the luminosity L, which describes the interaction rate of the colliding particles. It can be derived from basic machine parameters as show in Equation 2.1:

$$L = f \cdot n \cdot \frac{N_1 N_2}{\sigma_x \sigma_y}. \tag{2.1}$$

Here, f is the collision frequency of the bunches in the collider ring, n the number of bunches, N_i the number of particles in bunch i and $\sigma_x \sigma_y$ is the geometrical cross section of the beam. To increase the luminosity one can either increase the frequency, reduce the cross section of the beam, or increase the number of particles within each bunch. The upgrade from KEKB to SuperKEKB is based on tuning several of these parameters, where the most significant increase in luminosity comes from the so-called "Nano-Beam" scheme [13], corresponding to a significant decrease of σ_x. In total the geometrical bunch cross section is after the upgrade 10 μm $\times$ 60 nm. The beam current compared to KEKB is increased to 3.6 A/2.6 A (2.6 A/1.1 A) in the e^+ and e^- ring. Both rings intersect inside the Belle II detector at an crossing angle of 83 mrad. The energy of the High Energy Ring (HER) is reduced to 7 GeV. For the Low Energy Ring (LER) it is increased to 4 GeV, respectively, which results in a reduced Lorentz boost factor of 0.28 compared to KEKB (0.48). This demands higher vertex resolution of the detector, because the spatial separation between particles decreases. But this also reduces unwanted beam background in the inner detectors. SuperKEKB ran at a center-of-mass energy of 10.58 GeV, corresponding to the mass of the $\Upsilon(4S)$ resonance, which decays almost exclusively into $B\bar{B}$ mesons. Thus, SuperKEKB is a B-meson factory. An overview of the main accelerator components is depicted in Figure 2.1.

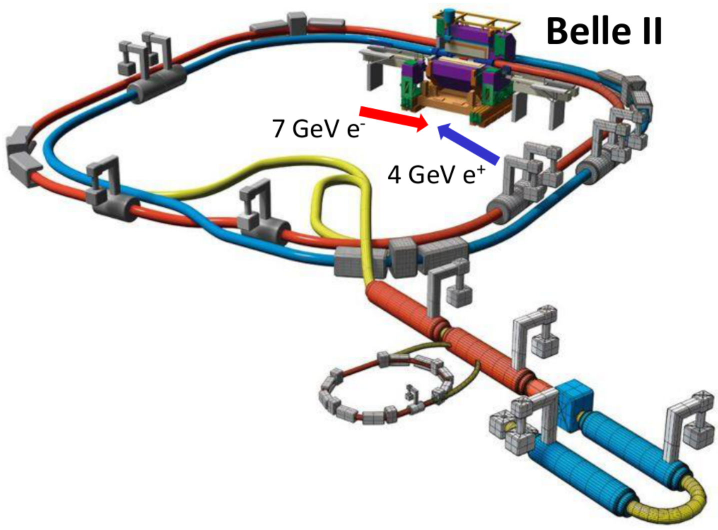


Figure 2.1.: SuperKEKB is an asymmetric double-ring e^+e^- accelerator with 7 respectively 4 GeV per ring. Special RF cavities accelerate the positron and electrons in a linear accelerator (LINAC) and are then injected into the main collider rings, where they reach their designated energies. In the only interaction region the two beams will be brought to collision in very dense bunches. The Belle II detector, located at the interaction region records the reaction products of these collisions.

With the aforementioned improvements, SuperKEKB aims to reach an instantaneous luminosity of $8 \cdot 10^{35}\ cm^{-2}s^{-1}$, which is about 40 times higher than in the old machine. In total it will accumulate an integrated luminosity of 50 ab^{-1}. This is 50 times higher compared to KEKB (1 ab^{-1}). To benefit from the changes made to the machine parameters a smaller beampipe and a new magnet design around the interaction region is required. Subsequently, a redesign of the Belle detector is mandatory and described in the next section.

2.2. Belle II Detector

The Belle detector components are currently undergoing an upgrade to cope with the increased demands of the higher collision rates of SuperKEKB. Figure 2.2 shows a cross section of the old Belle and the new Belle II detectors. A 3D model of Belle II is depicted in Figure 2.3. The most significant changes are located in the vertex detector. The four layers of a Double-Sided Strip Detector (DSSD) will be complemented by a highly granular ultra-transparent pixel detector to provide the required vertex resolution. It is radiation hard against the high particle density close to the beampipe. A new beampipe with 10 mm radius made up of a beryllium alloy with a thickness of 1 mm builds the core of the new detector. It allows one to place the first pixel layer at 14 mm, while the first layer of the strip detector is located at 38 mm. The two subdetectors are described in more detail in Chapter 3.

A new Central Drift Chamber (CDC) with more wires and a larger volume is integrated around the vertex detector. It also offers fast trigger capabilities and in conjunction with the 1.5 T magnet field allows momentum measurements. Behind the CDC the particle identification system is located, built up from an array of quartz bars. Exploiting the Cherenkov effect, charged particles with velocities higher than the speed of light in the detector medium emit light of a certain angle corresponding to its speed, which can be measured and used for particle identification. The Cherenkov detector is surrounded by an electromagnetic calorimeter system to detect particle showers produced by photons and electrons. The outer muon and K_0^L chambers complete Belle II and detect any remaining charged particles. Since this thesis deals only with the data reduction of the innermost pixel detector, thereby using information from the surrounding strip sensors, a closer look at the Vertex Detector (VXD) is provided in the next chapter.

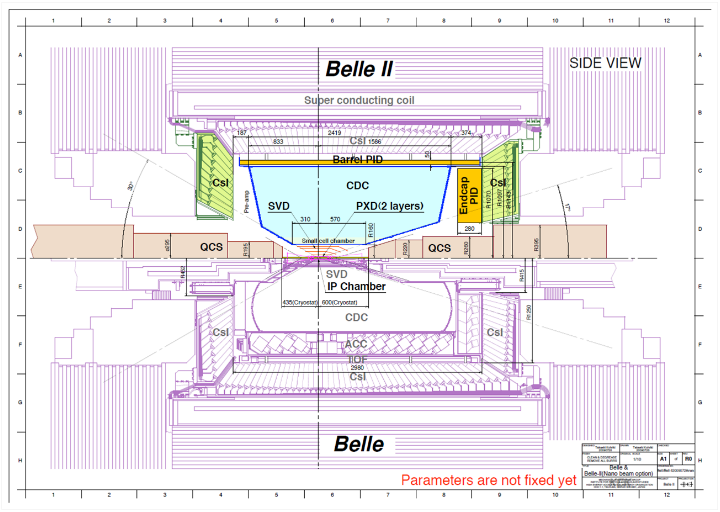


Figure 2.2.: Comparison between Belle (bottom) and Belle II (top). One highlight of the upgraded detector is the additional pixel detector (PXD) close to the beam pipe.

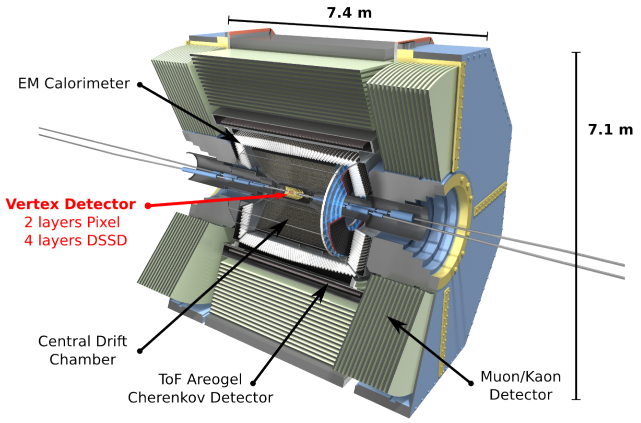


Figure 2.3.: The Belle II detector consists of the Pixel Detector (PXD), a Silicon strip Vertex Detector (SVD), a Central Drift Chamber (CDC), a particle identification system (Barrel and Endcap PID), an electromagnetic crystal calorimeter (ECL) and an outer muon and K_L system (KLM). Between the calorimeter and the muon chambers, the superconducting coil providing a 1.5 T magnet field is installed.

2.3. Background Sources in Belle II

Due to the increased luminosity of SuperKEKB, the background levels will dramatically increase compared to KEKB, especially in the inner detectors of Belle II. In several simulation campaigns it was tried to estimate the background in the PXD and SVD, which is important for certain design decisions, e.g. the Front End Electronics (FEE), buffer sizes or the geometry and size of the sensors. In addition, these simulations can be used to create a realistic model of events in the detector, which is needed for this thesis to estimate the performance of the track reconstruction algorithm. The idea for the data reduction scheme is based on a high background-to-physics-hit ratio. To understand the types of beam backgrounds present in the Belle II detector, their origin and the kind of impact they have on the occupancy on the PXD, the major background sources are summarized below [14]:

- **Touschek background:** Because of intra-bunch scattering, beam particles can drift out of the bunch and hit the outer vacuum and magnet walls. The showers created by these particles can reach the detectors and produce background hits. A high contribution of Touschek background is expected from the LER, since the rate is dependent on the inverse beam size, the bunch crossing rate and it is proportional to the third power of the inverse beam energy. The contribution from the HER is small, because of its higher energy.

- **Radiative Bhabha Scattering:** Scattered electrons or positrons can produce neutrons at the far end of the detector over an emitted photon via the giant photo-nuclear resonance mechanism. This is the major source of neutron background in the VXD, and the background scales with the luminosity of the machine.

- **Two-photon process:** This QED background, corresponds to the production of very low momentum e^+e^- pairs, in two photon processes:
  $e^+e^- \to e^+e^-e^+e^-$. This is the overall largest background contribution in the VXD.

- **Synchrotron Radiation:** is caused by bending of electrons and positrons mainly at the focusing magnets in the ring plane. This background scales with $I \cdot B^2 \cdot E^2$, where I is the beam current, B the strength of the magnetic field and E the energy of the beam. Thus, Synchrotron Radiation (SR) is dominated by the electrons in the HER. Special shielding structures, e.g. gold foils are put in place to protect the VXD, but not all of the background can be eliminated.

- **Beam-Gas Scattering:** Scattering of beam particles with residual gas atoms in the beampipe can cause energy loss through Bremsstrahlung or a direction change of the beam particles (Coulomb scattering). The scattered particles create showers when hitting the beampipe and induce backgrounds in the surrounding detectors. This effect is proportional to the beam current, the vacuum pressure in the ring and the strength of the magnetic field.

Simulation of these background sources were carried out and the results for the PXD and SVD are shown in Figure 2.4, Figure 2.5a and Figure 2.5b separated for different layers. Table 2.1 summarize the different background levels for the PXD. The PXD suffers the most from the high background levels, because of its close location to the Interaction Point (IP). The SR background is so far missing, but a first initial simulation shows an additional occupancy of 0.2 percent in the inner layer and 0.1 percent in the second layer. So far, the occupancy of the PXD with all accumulated backgrounds are below 3 percent, an important design constraint explained in Section 3.2.1.

Background	Layer 1	Layer 2
QED	0.8 %	0.3 %
Touschek	< 0.03 %	< 0.03 %
Radiative Bhabha	< 0.13 %	< 0.13 %
Beam-Gas	< 0.01 %	< 0.01 %
Total	< 1.0 %	< 0.5 %

Table 2.1.: Contribution of the different backgrounds to the pixel detector's occupancy for layer 1 and layer 2.

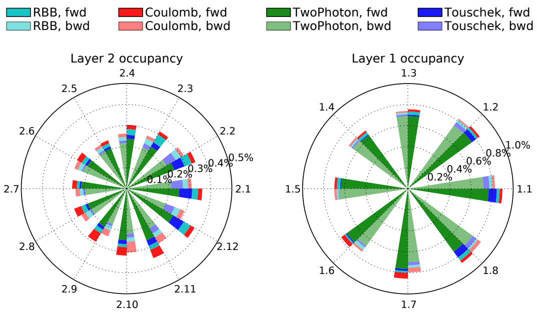
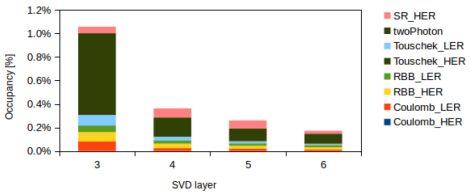
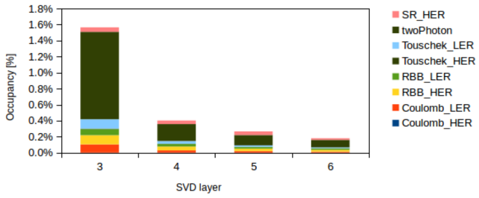


Figure 2.4.: Background contributions in the PXD, separated by layer and for the forward (fwd) and backward (bwd) regions.



(a) Background sources for the long strips.



(b) Background sources for the short strips.

Figure 2.5.: Occupancy contribution of the different background sources in the SVD separated by the long (perpendicular to the beam) and short strip side (parallel to the beam).

3 The Belle II Vertex Detector

Two different technologies are used to build the Belle II vertex detector. Both detector technologies need to be fast in readout to handle the increased bunch crossing rate and possess a better vertex resolution to handle the high particle densities. It also needs to be resistant against the harsh radiation environment close to the IP.
A common performance measure for a vertex detector is the impact parameter resolution. The impact parameter is defined as the point of closest approach of the particle track and its reconstructed vertex point. To compensate the smaller boost at Belle II compared to Belle, a better impact parameter resolution is mandatory. The resolution depends on the granularity of the sensor, e.g. the strip/pixel-pitch, and the distance of the closest and farthest detector layer to the IP. Because of the small and thin beam pipe, the first detector layer of the Belle II PXD can be placed very close to the IP. Due to the proximity to the place of collisions, the first layer suffers heavily from beam backgrounds. The, from the background induced high occupancy on the detector is too high to be handled by the Belle strip detector, so a completely new technology based on a pixel detector is required in the inner layer.
When a charged particle traverses matter, it is deflected due to multiple Coulomb scattering. While a single scattering process is purely random, multiple scattering is an effect which can be stochastically described. Equation 3.1 shows an approximation of

the scattered exit angle compared to the initial flight path of the particle.

$$\theta_0 = \frac{13.6\text{ MeV}}{\beta p} Z \sqrt{\frac{x}{X_0}} [1 + 0.038 \ln \frac{x}{X_0}] \ , \tag{3.1}$$

where Z is the atomic number of the material, p the momentum of the particle in MeV, γ the Lorentz factor, x the thickness of the material and X_0 the radiation length. The radiation length is the mean distance a particle can fly before it loses all but $^1/_e$ of its energy. It is a characteristic number for the material. While designing a tracking detector, the amount of material should be kept to a minimum to keep the particle tracks as unaffected as possible for precise trajectory measurements. The design of the vertex detector of Belle II aims for a vertex resolution of 10 μm.

In the following two sections a detailed description of the two vertex detectors will be provided.

3.1. DEPFET Pixel Detector

The DEpleted p-channel Field Effect Transistor (DEPFET) technology was first proposed in 1987 by Josef Kemmer and Gerhard Lutz at the Max Planck Institute of Physics. [15] After several decades of investigations and proof of principle studies the technology significantly advanced and has been chosen for the first two layers of the Belle II pixel detector. In this section a description of the concept and all involved readout components will be given with an additional focus on the structure of the data.

3.1.1. DEPFET Sensors

The principle building block of a DEPFET sensor is a Metal Oxide Field Effect Transistor (MOSFET) located on a fully depleted Silicon bulk. At a distance of 1 μm below the transistor channel an additional n-implant causes a potential minimum for traversing electrons. Electrons stored in this potential minimum, so-called internal gate, induce mirror charges in the gate, which in turn modulate the p-channel current. The change in this current is measured over the drain contact. A particle crossing through the active volume creates, according to the Bethe-Bloch equation[1], a specific number of electron hole pairs, depending on its energy. The momentum of the particle can be estimated by the number of electrons stored in the internal gate, determined by the modulated drain current. Because a transistor is already included in the pixel cells, the sensor has an internal amplification g_q, which was measured in a prototype structure to be around 500 pA/e^- [16]. One DEPFET pixel cell is depicted in Figure 3.1a.
The capacity of the internal gate and its parasitic coupling to the (MOSFET) gate can be controlled by the size and doping concentration. To achieve low noise the parasitic capacity to the gate should be low, while the internal capacity should be sufficiently high to store all electrons. A compromise between storage and parasitic capacity was accomplished with a noise level of $\sim$ 50 nA [16], mostly dominated by the readout electronics.
With higher occupancy of the internal gate the attraction to free electrons of the potential minimum starts to decrease. To prevent a saturated internal gate a clear process is required as depicted in Figure 3.1b. In an equivalent circuit the clear is realized over a second transistor turned by 90° forming an n-channel between internal gate and clear contact, once the clear-gate is activated. The

[1] Actually, the Bethe-Bloch equation describes the energy loss of particle ($^{dE}/_{dx}$) per distance, which can be translated into released electron hole pairs with the work function of the material.

electrons are then removed over this n-channel. During charge collection in the internal gate (clear-gate off), the clear contact is protected from accidental removal of electrons by a deep p-well implant.
For depletion of the sensor a method called "sidewards depletion", proposed and developed by E. Gatti and P. Rehak, is used [25]. A bias voltage is applied at the p+ doped back and front contacts of the n substrate of the sensor. The depletion regions extend from both sides until they merge in the middle of the sensor to a fully depleted bulk. This method has several advantages compared to traditional bias schemes. The voltage needed for depletion is a factor of four less and by applying different voltages to the back and front side, the potential minimum in the detector can be moved. In the Belle II design, the thickness of the sensor was chosen as a compromise between space for charge distribution, depletion length and material budget to be 75 μm.

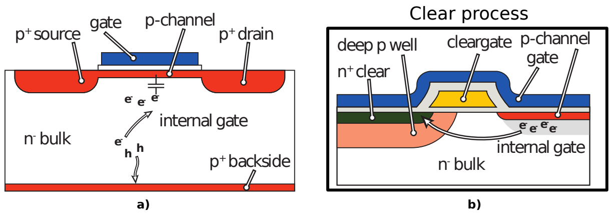


Figure 3.1.: Simplified illustration of one DEPFET pixel cell (a) and its clear process (b).

3.1.2. DEPFET Half-Module

The DEPFET pixel cells are arranged in a matrix containing 250 × 768 pixels, which is called a half-module, depicted in Figure 3.2. All clear and gate lines of each row and all drain lines of

one column are connected. In order to speed up the readout of the entire sensor, in the Belle II design four gates and clear lines are steered at the same time, while the drain current is read out. This method is called “rolling shutter” readout. Only active rows consume power, while the other pixels are still sensitive to charge. The sensor contributes only 1 W to the total 9 W power dissipated by one half-module. In order to place the required steering and readout Application Specific Circuit (ASIC)s outside of the sensitive area, long drain, gate and clear lines are used as depicted in Figure 3.2.
Three different types of ASICs are required to connect the driving lines and operate a DEPFET half-module. They are all mounted directly on the silicon die, so no additional support structure is present to further reduce the material budget. [26]

- The **Switchers** [27] are responsible to drive the gate and clear lines and are placed on a balcony on the side of the module. Each Switcher has 32 channels and each channel steers four gates and clear lines simultaneously, thus six of them are needed for one half-module. It is produced in a High Voltage (HV) Complementary Metal Oxide Semiconductor (CMOS) technology, since it needs to deliver up to 30 V for the clear process.

- The **Drain Current Digitizer (DCD)** [27] digitizes in 256 channels the incoming drain currents with an upstream connected transimpedance amplifier. To read all 1000 channels ($4 \cdot 250$), four DCDs are required, placed at the bottom of the module. Six channels per DCD are kept unconnected, or used as spares.

- **Data Handling Processor (DHP)** [28]: For further signal processing each DCD has a corresponding Data Handling Processor (DHP) to perform pedestal subtraction, zero suppression and data transmission over a 50 cm long and thin Kapton flex cable with four 1.6 Gbps high-speed serial Low Voltage Differential Signal (LVDS) lines. This Kapton cable also provides all power lines and slow control signals.

A four-rows readout cycle was measured to take at least 92 ns. Using a 100 ns read/clear-cycle a full frame readout would take 19.2 μs, which fulfills the Belle II requirements of 20 μs. A first stage of data reduction is already done in the DHP and is more closely described in the next section.

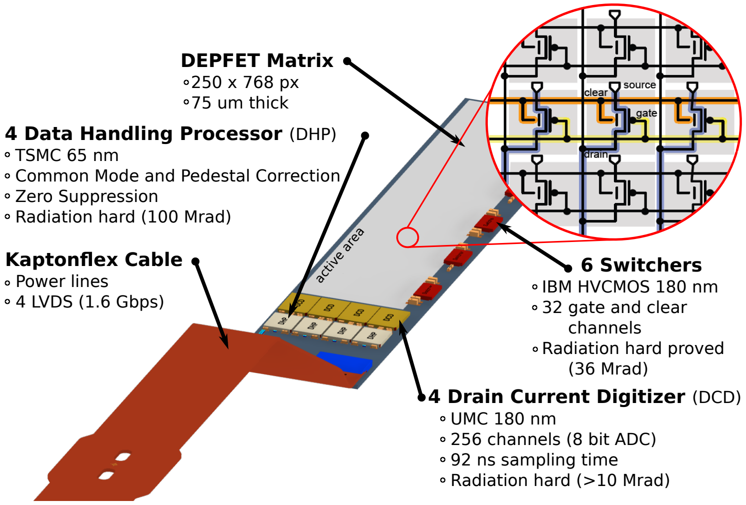

DEPFET Matrix
∘250 x 768 px
∘75 um thick
4 Data Handling Processor (DHP)
∘ TSMC 65 nm
∘ Common Mode and Pedestal Correction
∘ Zero Suppression
∘ Radiation hard (100 Mrad)
Kaptonflex Cable
∘ Power lines
∘ 4 LVDS (1.6 Gbps)
active area
clear
source
gate
drain
6 Switchers
∘ IBM HVCMOS 180 nm
∘ 32 gate and clear channels
∘ Radiation hard proved (36 Mrad)
4 Drain Current Digitizer (DCD)
∘ UMC 180 nm
∘ 256 channels (8 bit ADC)
∘ 92 ns sampling time
∘ Radiation hard (>10 Mrad)

Figure 3.2.: DEPFET half module for the Belle II Pixel Detector with all readout ASICs, transmission and power cabling.

	Layer 1	Layer 2
# Ladders	8	12
Radius	14 mm	22 mm
Ladder Size	$15 \times 136\ mm^2$	$15 \times 170\ mm^2$
Pixel size	$50 \times 55\ \mu m^2$ $50 \times 60\ \mu m^2$	$50 \times 70\ \mu m^2$ $50 \times 85\ \mu m^2$
# Pixels	250×1536	250×1536
Thickness	75 μm	75 μm

Table 3.1.: Properties of the pixel detector and key features of the sensors.

3.2. The Belle II Pixel Detector

Two half-modules combined, glued at their transition, which is further reinforced with ceramic rods, form one full ladder. To build the pixel detector for Belle II, 8 ladders for layer 1 and 12 for layer 2 are needed. The ladders are arranged in a so-called windmill structure to create an overlap between sensitive areas and thus avoid gaps. Table 3.1 summarizes the key features and properties of each layer and its sensors. The pixel sizes of the PXD sensors are divided into two groups between the central and outer regions. The smaller sizes are located at the central region to improve the resolution for perpendicular tracks, while in the forward region the incident angle is smaller, so the probability of creating clusters (neighbouring firing pixels) with two pixels increases. The larger size also prevents the charge cloud from spreading over too many pixels, which would decrease the resolution.

The occupancy of the PXD sensors is crucial for an optimal operation and design of the readout ASICs, thus extensive simulations were done and are still ongoing to get an idea of the expected levels. Figure 2.4 shows the latest simulation results of the different contribution and distribution of the major background sources described in Section 2.3. The inner layers have the highest background level, causing an occupancy of ~ 1 percent, the second is a factor of two less. The asymmetry in the occupancy of the different ladders is later exploited to load balance the data. In an average $B\bar{B}$ event about 10 tracks are created. Considering the current estimation of the background level, the PXD is completely dominated by background hits. Thus, the outgoing detector bandwidth can be reduced significantly by removing as many background hits as possible.

3.2.1. Data Handling Processor

The DHP is the first ASIC whose operation is sensitive to the expected background level as later explained in this section. It receives the data from the DCD and performs several signal processing tasks. Since every pixel has its own pedestal value (a constant offset level, dominated by the pixels noise and dark current). This needs to be subtracted in order to obtain comparable information about the deposited charge. In a second step, the so-called Common Mode (CM) is subtracted. This is a constant offset on the sampled data which is equal for all pixels at the same time (mostly caused by fluctuations in the electronics). After digitization, each DHP needs to handle an incoming data rate of 20 Gbps, stored event-by-event in a ring buffer[2]. For the whole detector this adds up to more than 3 Tbps. Thus, the first stage of data reduction is required to be on-module.
After pedestal subtraction and CM correction, the Analog-to-Digital Converter (ADC) values are written into a First-In First-Out (FIFO) memory and are processed by the zero-suppression logic block. Only pixels containing a charged particle hit are of

[2]A limited memory which is constantly overwritten when the write pointer reaches the end.

interest to be read out. These pixels stand out with a specific factor above their usual noise level. The threshold, which triggers the readout is pre-calculated offline and stored in the DHP and then compared with the incoming signal.
Once passed the hit-finding module the data is written into a second FIFO. The maximum supported occupancy of the sensor is limited by the depth of this FIFO, which is designed to hold 3 percent of the total number of unsuppressed pixel data per event. Beyond this value data will be lost. A high-speed serial LVDS link provides the connection to the next Data Acquisition (DAQ) stage. All communication between the backend electronics and the module is done over the DHP and a 50 cm long, thin Flex-Kapton cable, connected to a patch panel. Two additional 15 m long Infiniband cables connect the patch panel to an FPGA board, called Data Handling Hybrid (DHH) [32]. To ensure a stable 1.6 Gbps connection, a special link driver with pre-emphasis was developed for the high-speed data connection. At link level, the Aurora protocol is used[3]. All other timing and control signals (e.g. Joint Test Action Group (JTAG)) are transmitted at much lower speed. The DHP distributes the signal over separate JTAG chains to the Switcher and DCD.

3.2.2. Data Handling Hybrid

Each DHH has four unidirectional high-speed connections to one PXD half-module, connected over the patch panel. The DHH is based on the Advanced Telecommunications Computing Architecture (ATCA) standard and is equipped with an FPGA[4] design using the already included high-speed transceivers. One of the main tasks of the system is to perform a data load balancing and

[3] Aurora is developed as a lightweight, ready-to-use protocol for Xilinx chips. The source code is freely available. More information can be found in Section 6.1.1.

[4] For more details about FPGAs and the ATCA standard see Chapter 6.

the aggregation and conversion from LVDS lines to the more electrical noise resistive optical fibres. The load balancing is realised over a two-phase system: 5 DHHs are connected to one so-called Data Handling Hybrid Controller (DHHC). As described in Section 2.3 the asymmetry in the background can be exploited to combine sensors with low occupancy with sensors suffering from high background. Two half-modules from layer 1 and three from layer 2 (on the opposite side) are connected to one DHHC. In the second step, the DHHC is connected over four 6.25 Gbps links with the DAQ system, called ONSEN [33]. The data is distributed between these four links in a programmable round-robin[5] like sequence. The connection topology is depicted in Figure 3.3. In addition, the DHH has an FPGA-based pixel clustering engine. It helps to further reduce the data rate and save relevant physics tracks by an analysis of the cluster shape and the total deposited charge. A large cluster seed indicates a low-momentum particle, which is difficult to reconstruct without hits in the outer strip detector.

3.2.3. Online Selector Node (ONSEN)

The ONSEN system uses the Micro Telecommunications Computing Architecture (mTCA)[6] standard with custom-made Advanced Mezzanine Card (AMC) of the same type as the cards used by the DATCON system. The hardware will be discussed in Chapter 6. The ONSEN system performs the final data reduction in two steps. But first it acquires the data from the DHHC and stores them in an internal 4 GB large memory, which can keep the non-reduced, zero-suppressed data for about 5 s. A connection to the High Level Trigger (HLT) is used to achieve a first data reduc-

[5]Round-robin is a simple method of load balancing data between two or more links using a cyclic write-sequence.

[6]Deviation of the ATCA standard with a custom connection layout and smaller form-factor.

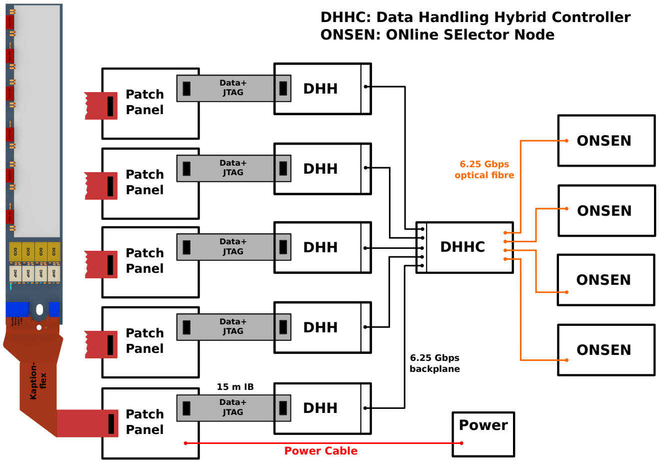


Figure 3.3.: Overview of the Belle II PXD DAQ system. One DHHC is connected to five DHHs reading each one half-module, thus to read out the whole PXD 8 such subsystems are required.

tion of a factor of 3 by only storing events approved by the HLT. The next phase uses the ROIs to select relevant pixels associated with a particle track. These ROIs are provided by two complementary systems. One is already integrated in the HLT and uses a cellular automaton and a Hopfield network[7] to reconstruct the tracks in the SVD while the other is the topic of this thesis. An additional node functions as input module to receive the ROIs from both systems and distributes them to all processing modules. Only one optical link with all aggregated ROIs is needed to be provided by DATCON. With this scheme an additional reduction of a factor of ~ 10 is aimed for. The triggered and selected data are transmitted from each ONSEN AMC over Gbit Ethernet

[7] A form of an artificial neuronal network.

to the Event Builder 2 (EVB2). There, the subevents[8] are merged and written to disc.

3.3. The Silicon Strip Vertex Detector

Compared to the old Belle SVD, the Belle II SVD is a completely new development with new sensors and it uses a new FEE ASIC, which allows for a faster readout and a better timing resolution. The sensors are based on DSSDs covering a larger acceptance area with fewer readout channels compared to a pixel sensor. A detailed description of the components of the SVD data chain are presented in the next sections.

3.3.1. Double Sided Strip Detector and Geometry

A pixel detector provides two-dimensional information for every hit, while in a strip detector only one-dimensional information per channel can be obtained. To create 2D space points at least a second strip oriented in the perpendicular direction is required. With a DSSD both spatial information are available on the same sensor with doping implantation in perpendicular (or shifted with a stereo angle) orientations on both sides. For Belle II the p-type implantation refers to the long strips facing the beam axis and providing the coordinates in r-φ. The n-type is for the short strips perpendicular to the beam line. Three different shapes of sensors are used to cover an acceptance range of 17 to 150 degrees:

- **Layer 3**: These sensors have 768×768 strips with a pitch of 50×75 μm^2 and are used for the inner SVD layer to cope

[8]A set of data from one event taken by five DHHs.

with the higher background. The active area has a size of 122.90×57.71 mm^2.

- **Barrel**: All barrel sensors from layer 4 to layer 6 have the same layout and strip numbers 768×512 with a pitch of 75×240 μm^2. The sensors are constructed using the so-called Origami chip-on-sensor concept, which allows for a placement of all readout ASICs for both sides in a row on top of the sensor to simplify the cooling structure and thus reducing the amount of inactive material. The electrical connectivity to the strips is ensured over thin flex cables bent towards the sensor.

- **Wedged**: In the forward region of the SVD the sensor shape differs from the rectangular modules. The sensors are cut in a trapezoidal shape and slanted towards the beam pipe. With this mechanical construction the sensors can cover a larger acceptance angle while using less space.

The size of the sensors is basically driven by the largest commercially available wafer technology size. Two different companies build the barrel and the trapezoidal sensors. A higher vertex resolution in the r-φ plane is required for a better vertex resolution and a more precise momentum estimation of charged particles. Therefore, the number of strips on the p-side is higher in the barrel. In the origami concept the readout APV25 chips are placed as close as possible to the sensors to keep the connection length to a minimum, reducing the noise. The sensors are mounted on large support ribs and fixed on end-rings forming a windmill structure, which avoids gaps between layers by creating some overlap. On each support rib, up to five sensors can be installed head to head, creating only small insensitive gaps in between. The sensors are arranged in such a way that a track originating from the IP can hit at most one of these gaps in a single layer. A cross section through the SVD, showing the windmill structure and the side

view is depicted in Figure 3.4.
The basic properties of each layer of the SVD are summarized in Table 3.2. A simple extrapolation from the information gathered at Belle, scaled up with the higher luminosity of Belle II and a $1/r^2$ estimation for the new radii of the layers was used to create a rough estimation of the occupancy to be expected in the SVD.

Layer	Radius (mm)	Occupancy	APV25	Ladders	Sensors	Strips
6	140	0.9 %	850	17	85	108800
5	115	1.3 %	560	14	56	71680
4	80	2.7 %	300	10	30	38400
3	38	6.7 %	192	8	16	24576

Table 3.2.: Number of strips, readout chips and the expected occupancy for the four different SVD layers.

3.3.2. APV25

One APV25 chip, originally developed for the CMS experiment at the LHC, is connected to a sensor and can read out 128 strips. The chip shows a good signal-to-noise performance of 270 electrons per pF load capacitance [17]. It is produced in 0.25 μm CMOS technology and with its thin gate oxide it is radiation hard, way beyond the levels expected at Belle II. Although the chip was designed to operate at the LHC collision frequency of 40 MHz, it can be used at lower or even higher operating frequencies of $f_r/16 = 31.8$ MHz or at $f_r/12 = 42.4$ MHz, respectively with the bunch crossing rate $f_r = 508.8$ MHz of Belle II.
A block diagram of one analog cell of the APV25 is depicted in Figure 3.5. Each strip of the sensor is connected to a preamplifier to pre-condition the input current for the shaper. The shaping constant of the circuit can be adjusted to fit the needs, if the

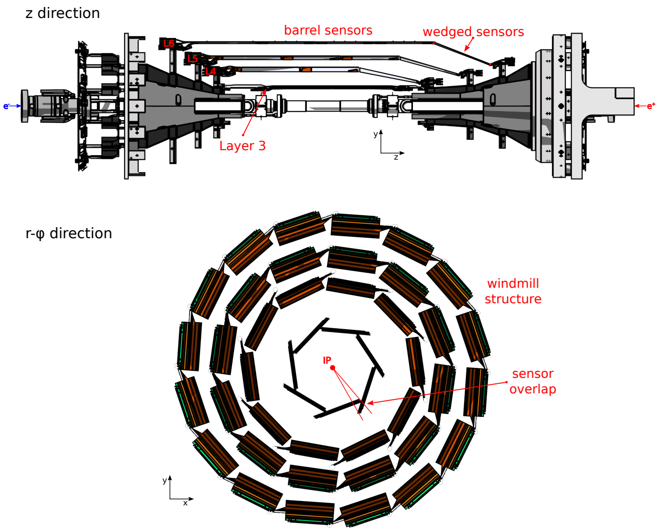


Figure 3.4.: A cross section, showing the windmill structure and a side view of the SVD detector.

standard 50 ns is not suitable. The shaped signal is then stored in a 192 deep cell ring buffer waiting for readout. On each trigger signal the ring buffer is, with a fixed offset (determined by the trigger delay), written to a multiplexer. Then a differential output driver sends the analog data to an ADC.

The number of samples (one, three, or six) to be read out of the ring buffer can be configured over the I^2C bus interface. In addition, the APV25 also provides the possibility to inject a calibration pulse and read an internal error register. An example frame of one sample of the LVDS signal from the APV25 is depicted in Figure 3.6. Every 35 clock cycles the APV25 sends a “tick” mark, when no trigger is processed to keep the backend electronics synchronised. Once a trigger arrived, the APV25 sends three

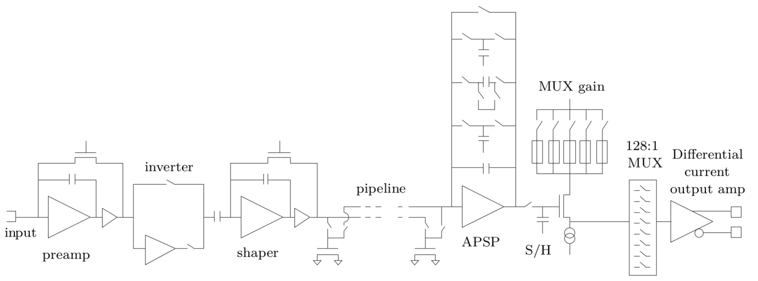


Figure 3.5.: A block diagram of the APV25 chip, depicting one channel.

consecutive digital high ticks as header indicator, followed by the 8 bit address of the currently read sample from the ring buffer and one error bit in active low logic[9]. If the error bit is asserted, the error register of the APV25 needs to be read over I^2C to determine its nature. After the error bit the analog data of the 128 input channels follows. The level is shifted with a configurable offset to distinguish the channel data from the header, address data and ticks. A tick mark indicates the end of one sample. Since the APV25 is mostly an analog chip, the interpretation and decoding of the signals is the responsibility of a different system described in the next section.

3.3.3. Fast ADC

The FADC [18] samples the signals coming from up to 48 APV25 chips. Each LVDS line of the APV25 is connected to a 10 bit wide ADC, which translates the incoming current to a digital signal. This signal is further processed by an Altera Stratix 4 FPGA.

[9]The signal is high when there is no error.

The kind of processing is dependent on the readout mode of the FADC. Four different readout modes are currently planned:

- **Raw:** The signal from the ADC is kept untouched and a configurable number of up to 1024 samples are stored in a memory waiting for readout. This mode is mainly for debugging and calibration purposes, e.g. synchronizing the sampling clock with the APV25 signal.

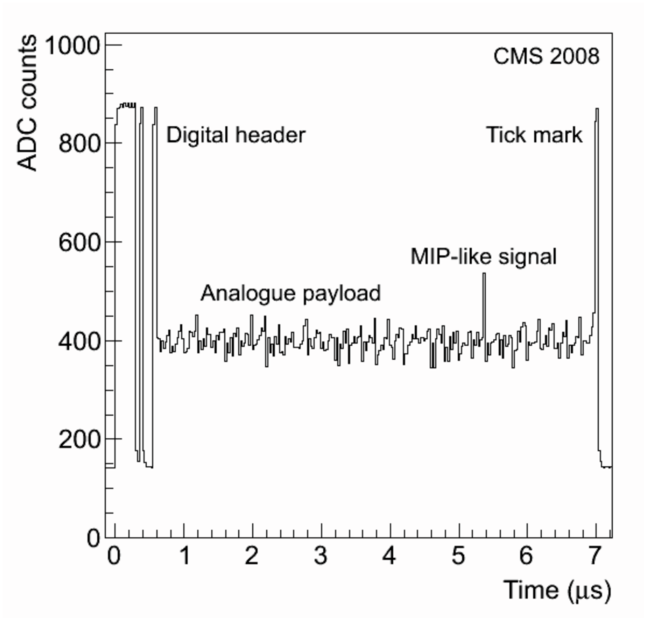


Figure 3.6.: An example frame of an APV25 chip from CMS, showing the digital encoded header and the analog payload containing the signal of 128 channel with a single MIP particle hit. [53]

- **Transparent mode:** In this mode, the FADC decodes the analog data, including header and error bit detection. Further it reorders the 128 strip signals in an ascending order. The 10 bit samples of the strips and the readout address of the APV25 ring buffer are stored.

- **Zero-suppressed mode:** In addition to the processing steps in Transparent Mode, a two-phase Common Mode Correction (CMC) is applied with pedestals, that are individually pre-calculated for each strip and loaded to a memory during the configuration phase. After CMC a second memory for each strip is read with the zero-suppression value, which is compared to the signal. Only when at least one sample is above the threshold, all samples are written to the readout FIFO. This modes supports the readout of 3 or 6 samples.

- **Hit-time mode:** The same steps as in Zero-suppression mode are applied, but instead of 3 or 6 samples only the deconvoluted waveform information (hit time and size of the peak) is stored. The deconvolution of the signal can be realized over either a waveform fit or a neural network. Both methods are currently under investigations.

In the Hit-time mode the occupancy of the detector can be further reduced by rejecting offtime triggered hits, when the reconstructed peak time is not in a specific window around the expected trigger time. In total the occupancy of the detector can be reduced by a factor of 100. In addition, the FADC controls the I^2C interface and distributes the clock and trigger signals to the APV25s. For configuration and data access a second smaller Altera Cyclone FPGA is present on the FADC which handles the communication of the Versa Module Europa (VME)-Bus and the Stratix FPGA. In every FADC crate exists a VME bridge, which acts as master and can read and write data to the bus controlled

by a Personal Computer (PC) connected over an optical fibre. For debugging purposes a so-called “Spy”-FIFO is implemented in the FADC to “spy” on the processed data in order to cross check and receive a copy of the processed data.
The normal readout process of the data is handled by a second FPGA board, which is connected over an VME backplane connector. It contains several control signals to distribute clock, busy and trigger signals in both directions and a 32 bit unidirectional wide bus interface (see Section 3.3.4 for more information about the Finesse Transmitter Board (FTB)).
Because of different voltage levels between *n*- and *p*-side, each FADC is only connected to one implantation type of strips. With its 48 input channels, for instance only four FADCs (two each for the *n*- and *p*-side) are needed to read all sensors in layer 3. To connect all 1902 APV25s in the SVD 48 FADCs are required. The synchronisation of the FADCs, housed in three different VME crates is realized over the FADC Controller, which distributes a common clock and trigger signal to one buffer board in each crate. Over the VME bus the buffer board sends the signals to all FADCs. A functional overview of the FADC and FTB is depicted in Figure 3.7.

3.3.4. Finesse Transmitter Board

The main purpose of the FTB is to translate the electrical signals to optical ones, which has several advantages: robustness against electrical interferences, support of larger transmission distances at high-speed and electrical decoupling of the systems. It is connected to the FADC over a 32 bit wide bus. The data received from the FADC is encapsulated with an additional FTB header and trailer. In addition, it receives trigger and timing information from the Fast Timing Switch (FTSW) [34] which is added to the event header. An overview of the data protocol is shown in Figure 3.8.

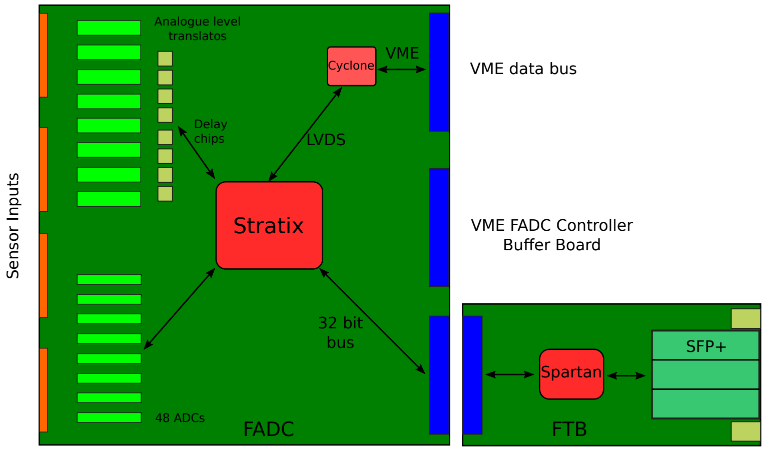


Figure 3.7.: The main components and connection topology of the FADC and FTB.

The data is propagated using the high-speed gigabit transceivers of the Spartan 6 FPGA on the FTB. One is connected over an optical link to the Common Pipeline Platform for Electronics Readout (COPPER) [35] [36] system. A copy of the data is transfered to a second optical transceiver which will be used for the DATCON system. For details of the implementation see Section 8.1.

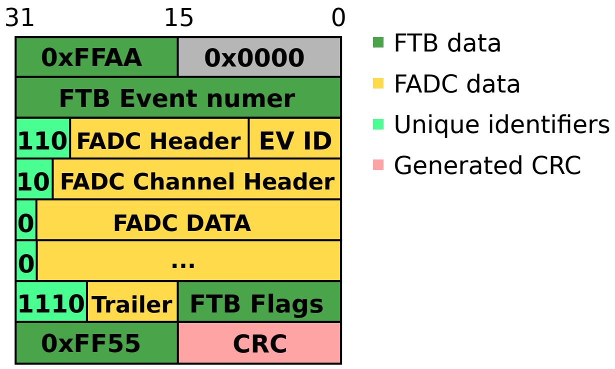


Figure 3.8.: Main parts of the protocol used for communication between the FADC and the FTB. The "unique identifiers" are fixed bits that help to assign the correct information.

4

Basics of Data Preprocessing and Track Reconstruction

Several concepts and algorithms were investigated before a decision for an implementation of a data reduction algorithm for the DATCON system was made. In this chapter the most important results of this evaluation and a detailed description of the chosen algorithm are presented.

The first idea of a lossless binary compression algorithm[1] was rejected because the suitable widely used algorithms cannot achieve the required reduction rate of a factor of 10. A customized algorithm specifically designed for data reduction in the PXD would be necessary to increase the compression factor. But this has a large uncertainty since the exact protocol format and data structure are currently unknown, or the occupancy of the PXD is higher than expected.
The only obvious option left is to select a subset of the pixel data in each event and remove the rest. This bears the potential risk of losing physics data. A method is now required to separate the physically interesting hits from non-relevant physics background hits. As seen in Section 2.3, the occupancy of the PXD is completely dominated by background hits, which can be rejected without information loss in the later analysis process. To select the physics-relevant pixels in the PXD the surrounding SVD detector is used to find track segments and their parametrization

[1]An algorithm reducing the data bandwidth while keeping the original data completely recoverable without information loss.

sourced from the interaction region. Using this parameterization, the track can be extrapolated onto the pixel sensor planes. The intersection point between extrapolated track and sensor is from now on referred to as Most Probable Hit (MPH). Since there are several effects that let the particles deviate from their ideal trajectory, mostly caused by multiple scattering and misalignment of the sensor planes in the fit, a ROI around the MPH is defined. Only pixels inside the ROI are further processed and stored.
The first task is to find an algorithm which provides a high efficiency for track finding and fitting. Two major types of algorithm were evaluated and their basic principles are presented in the next two sections. Along with the tracking algorithm, many other steps are required to transform the strip data of the SVD into coordinates, so it can be used for a tracking unit. A list of pre- and post-processing steps are provided below:

1. Acquiring and decoding of the data from the SVD.

2. Clustering of strip hits. This combines several neighbouring strips into a cluster, which helps to reduce the amount of data to be processed and, as shown later in Chapter 5, this also increases the hit point resolution in the sensor.

3. The two more closely investigated tracking algorithms rely on absolute coordinates. These are obtained through a coordinate translation of the clustered strip data.

4. The track reconstruction algorithm needs to find hits in the detector that belong to the same particle track. Afterwards the track parameters are obtained through track fitting.

5. With the parameters, the track is extrapolated onto the PXD planes. For this, the crossing point of the extrapolated track with the sensor planes needs to be determined first. Two different methods can be used to calculate the MPH: A

sampling of the particle track in small steps with a distance calculation to find the point of closest approach with the PXD sensor planes or an exact algebraic solution for the intersection point.

6. The MPH needs to be translated to a column and row pixel ID and sensor ID.

7. With the pixel ID of the MPH and the size of the ROI the pixel IDs of the outer edges, which defines the ROI can be calculated. The size of the ROI is flexible to compensate for multiple scattering and other effects e.g. the ROI should get larger for lower reconstructed track momentum.

All algorithms of the processing steps must be suited to run on the FPGA. Ideally there should be no cross-dependencies between different data words which are perfectly suited for parallel data processing. In the next section two track finding and fitting algorithms will be presented, which were implemented as a C++ model and on the FPGA.

4.1. Sector-Neighbour-Finder

In imaging applications, a large variety of algorithms are used to find regularities or patterns in unstructured data. A subset of them can also be used in particle physics. The patterns are in this case the hits in detector planes originating from particle tracks. Finding hits corresponding to the same track is the task of a "track finder". The parametrization of the track is done with a different set of algorithms. Usually these two steps are treated in different modules, but not all of them as seen in the next section. The first evaluated algorithm uses the assumption that each hit - related to a track - in the detector is in close proximity to a

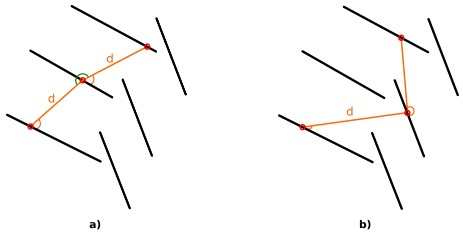


Figure 4.1.: Illustration of the sector-neighbour finder showing a) an accepted and b) rejected combination of hits.

hit in a neighbouring layer. By calculating different properties between two hits in neighbouring layers, like distance, angles and curvature, a set of parameters is created to be compared with the hits in the next layer. The compatibility of the parameters is checked for acceptable deviations between them. Once accepted the hits are marked as track candidate. Figure 4.1 shows an illustration of this neighbour finding algorithm.

The combinatorics (all combinations between detector hits) are directly proportional to the number of hits n with the run-time scaling with $\mathrm{O}(n^2)$. A reduction is possible by pre-sorting the hits by layer, so hits in each layer are only compared with hits in the next layer. The selection of the initial hit, so-called "seed hit" also helps in reducing the combinatorics, when choosing the hits in the outermost layer and then propagating inside. As seen in the background simulation Section 2.3, layer 6 of the SVD has a much lower occupancy than the inner layers. A further reduction is possible by dividing the detector into different, overlapping sectors to reduce the number of calculations. Only hits in compatible sectors in relation to the seed strip are processed. This requires a pre-calculated sector map and further logic to check for compatible sectors.

In principle this method is suited to run on an FPGA, because every calculation path starting from the seed strip can be treated separately. On the other hand it has some clear disadvantages: concurrent memory access to all hits is required, thus the hits in each layer need to be placed in different directly accessible registers, which increases the resource consumption. The tracks are bent in the r-φ-plane due to the presence of the magnetic field in Belle II. For slow particles ($\mathrm{p_T} \leq 100$ MeV), e.g. slow pions in $D^\star \rightarrow D\pi_s$ decays, the bending radius of the slow pion π_s is in the order of the detector radius and thus the sectors and the acceptable distances between hits needs to be extremely large. With increasing occupancy the number of mis-identified associated hits increases accordingly and not only the closest neighbouring hit in the next layer needs to be taken into account.

With this method only the detector hits belonging to a track can be identified, but no track parameterization is performed. For this a second algorithm is needed, which in the case of straight tracks only uses the basic equation for describing a straight line with an averaging method for the parameters:

$$\begin{aligned} y &= m \cdot x + a, \text{ with} \\ m &= \sum_{i=0}^{N} \frac{y_i - y_{i-1}}{x_i - x_{i-1}} \cdot \frac{1}{N}, \\ a &= \sum_{i=0}^{N} (y_i - m_i \cdot x_i) \frac{1}{N}, \end{aligned} \tag{4.1}$$

with the slope m and the intercept of the y-axis a of N hits associated with the same track. However, this simplification only works for high-p_T tracks, where the hits are in first order on a straight line.

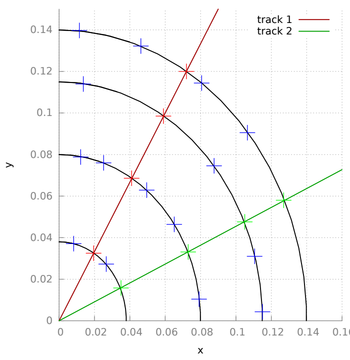
For bent tracks a different parametrization must be defined. There are several options, of which two are commonly used. The most direct approach would be to reconstruct the initial momentum vector of the particle at the vertex point. With the knowledge of the magnetic field, the track can be extrapolated. A different approach is to fit the particle trajectory with a helix parameterization. This involves solving a linear equation system. Since this isn't feasible to do on an FPGA, this algorithm was used only as an optional filter to find candidates of hits with the potential to be related to tracks. The fitting of the track is done in another way described in the next section.

4.2. Hough Transformation

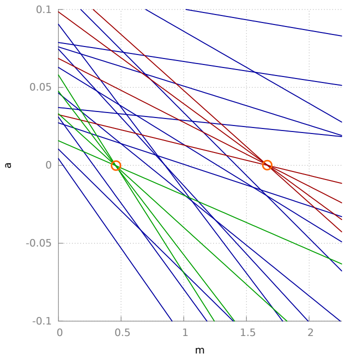
The Hough Transformation (HT) was originally proposed by P.V.C. Hough to automate the pattern recognition in bubble chambers [30], specifically to find straight and bent tracks. Later it was adapted to perform general pattern recognition by adapting the template matching paradigm. Several experiments and concept studies used the HT as their main tracking algorithm with great success e.g. the Compressed Baryonic Matter (CBM) trigger [37], or the CDC detector. One outstanding feature of the HT is the capability to perform track finding and fitting in a single step. As illustration of the reconstruction algorithm of two straight tracks in the r-φ-plane a random distribution of noise hits has been added to the event. In the beginning the simple parametrization of a straight track as show in Equation 4.2 is used. When solving this equation for the intercept a, the variables of the original formula become the parameters and the other way around.

$$y_i = m \cdot x_i + a \xrightarrow[trafo]{Hough} a = -m \cdot x_i + y_i \tag{4.2}$$

When filling the newly created space with detector hits in absolute coordinates (x_i, y_i), the points in normal space become lines in the Hough Space (HS). An intersection of two or more lines means there is a common parameter set of m and a for these particular detector hits. This not only allows for the detection of a pattern in these space points but also allows a direct track fitting. The



(a) x-y space with randomly distributed hits and two tracks.



(b) Corresponding Hough space with in orange marked intersections.

Figure 4.2.: Illustration of the HT.

task of pattern recognition is now reduced to find intersections in the HS.

However, the representation in this form has certain problems. For once, the whole space needs to be searched for intersections to find all possible tracks. Due to the limited resources it is not feasible to do in an FPGA. The second problem concerns the range of the track's slope m. If the particle track is parallel to the y-axis m is infinite. To solve this problem, R. Duda and P. Hart introduced a new parametric description of lines based on the work of L. Hesse. [31] A straight line is now described by the closest distance between the line and the origin, d, and the angle of the distance vector, φ, as depicted in Figure 4.3. The new parametrization is given by:

$$d = x_i \ \cos\varphi + y_i \ \sin\varphi. \tag{4.3}$$

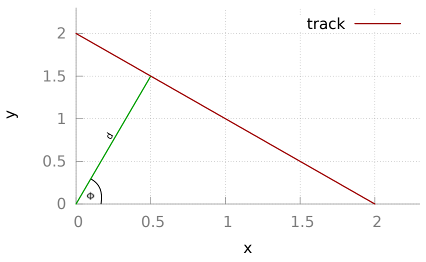


Figure 4.3.: Hesse parametrization of a straight line described by distance d from the origin and angle φ.

This limits the search in φ to the finite interval of $[-\pi/2, \pi/2]$. The maximum distance of the track is limited by the maximum deviation around the vertex region. The back transformation to get the original parametrization is:

$$\begin{aligned} m &= \tan\left(\varphi - \frac{\pi}{2}\right) \\ a &= r(\sin\varphi - m\cos\varphi) \end{aligned} \tag{4.4}$$

With the HT the task of finding tracks is reduced to detecting intersections in HS, but has so far only been used for a two-dimensional problem. For the application in the Belle II detector a three dimensional track reconstruction is mandatory. The HT also works in three dimensions but the free variables increase to four, so a four dimension HS needs to be searched. This 4D space would be inefficient to check, so the actual track finding is limited to a search twice in two different dimensions. The two parameter sets are later merged into a single 3D track parametrization. This also makes the system more flexible in terms of parallelization, since the search in each HS can be executed independently.

4.2.1. Conformal Mapping

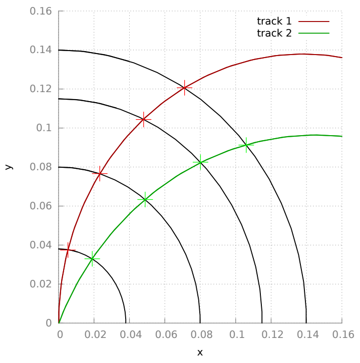
In the $r - \varphi$-plane the magnetic field of Belle II will cause the charged particle trajectories to be bent. In first order, for high-momentum particles the trajectories can be approximated by a straight line. Since the reconstruction of low-momentum particles is important in B-physics analysis of Belle II, a new method must be found, preferably reusing the HT. The conformal mapping is a way to transform points on an arc into points on a straight line. The transformation equation is as follows:

$$\begin{aligned} x_i' &= \frac{x_i}{(x_i - x_n)^2 + (y_i - y_n)^2}, \\ y_i' &= \frac{y_i}{(x_i - x_n)^2 + (y_i - y_n)^2}, \end{aligned} \tag{4.5}$$

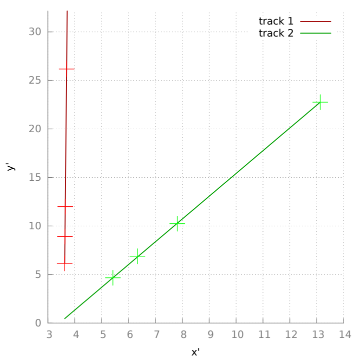
where $(x_n,\ y_n)$ is a point on a circle. In case of an hermetic detector around the interaction region it is set to the coordinates of the IP. To simplify the equation, the coordinate system is transformed so it lies on the origin. With this conformal mapping, the HT can be applied again. Figure 4.4 shows an example of two low-momentum particle tracks and the conformal transformed detector hits. The radius r of the circle can be calculated with

$$r = \frac{1}{2d}. \tag{4.6}$$

Since there is no precise vertex point in Belle II, a variance of one millimeter is assumed (the typical flight length of a B meson at Belle II is in the order of 150 μm), the algorithm used for detecting the intersections must be tolerant against fluctuations for compensation of the not exact intersection of several lines. Equipped with this knowledge the HT can be used to reconstruct the z component as an angle of θ and in $r - \varphi$, the track radius and its initial angle φ. Appendix B shows the helix coordinate system used to reconstruct the mentioned parameters φ, θ and r. More details about the way these two different HS are merged and



(a) x-y space with two bent tracks.



(b) Corresponding conformal space.

Figure 4.4.: Illustration of the conformal mapping with a simplified detector layout.

how they are actually searched will be presented in Section 5.3.2.

4.3. Most Probable Hit Extrapolation

Once the helix parameters are extracted with the HT an efficient method must be found to extrapolate the hits onto the PXD. In other words the intersection between the PXD planes and the helix trajectory must be found. Two different approaches were investigated, one sampling method and an algebraic solution of the problem.
The sampling method samples the helix trajectory along the x-

axis. From each sampling point x_{si} the coordinates of the points on the helix are calculated as follows:

$$\begin{aligned} x'_i &= x_{si}\cos(\varphi) - \sqrt{1-(x_{si})^2+2x_{si}r} sin(\varphi), \\ y'_i &= x_{si}\cos(\theta)\sin\varphi + \sqrt{1-(x_{si})^2+2x_{si}r}\,\sin(\varphi), \\ z'_i &= x_{si}\sin(\theta)\sin\varphi + \sqrt{1-(x_{si})^2+2x_{si}r}\,\sin(\varphi), \end{aligned} \tag{4.7}$$

Afterwards, the method of the point of closest approach will be used. The distance between each sampling point of the helix curve is compared with each of the detector layers. The one with the closest distance between the sampling point and the detector planes is taken as MPH. This method also requires a good filtering of tracks, so that only helix parameters which are able to cross the PXD planes will be used, or a maximum distance between the closest point and one detector plane must be defined. The coordinate of the closest sampling point is translated to a pixel ID.
An advantage of this method is that intersections between every track and detector shape with a known parametrization can be calculated. The penalty of this approach is the high calculation effort. For every sampling point on each track, the distance between the point and all detector planes must be computed and minimized. To reach a sufficient precision the number of samples must be close to the size of one pixel (50 μm - 75 μm), which means 4400 sampling points for the outer PXD radius. Considering that this algorithm has to be executed for every track a different solution must be found.

Since the PXD has a cylindrical structure with straight detector planes, which can be in first order estimated as straight thin lines with a fixed length, the direct solution of the analytic function may prove to be more efficient. The task of solving the equation to find the intersection between a helix and a straight line becomes more effective when using a normalized detector plane orientation. Each detector plane is back-rotated by their rotation angle (used to create the “windmill structure”), so that they are parallel to the y-axis. The helix track is also rotated by the same angle. This simplifies the intersection calculation between helix track and the detector plane, because the x coordinate is already known as the radius of the particular layer. Figure 4.5 illustrates the rotation concept. The calculation of the x-component of the sensor is, with this method, the radius of the particular detector plane. Insertion of the x coordinate into the rotated helix parametrization, the y-component can be computed and compared to

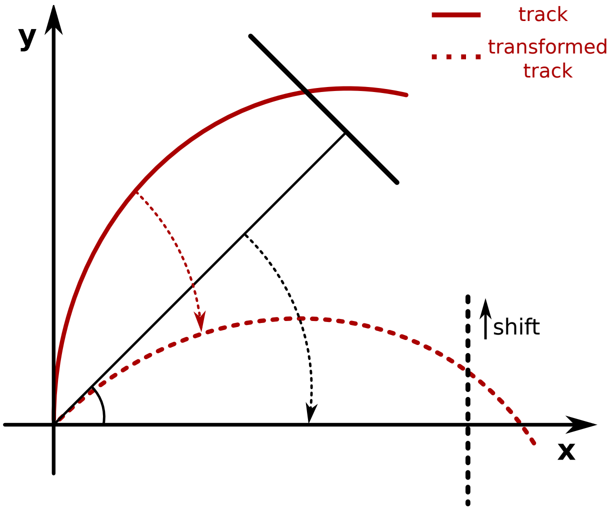


Figure 4.5.: Concept for the MPH extrapolation on the PXD planes, using an analytics method to precisely calculate the intersection.

the position of the sensor. The calculation is shown in the following equation:

$$\begin{aligned}
&\varphi = \varphi_t - \varphi_s \\
&e := r_s \cos\varphi- \mid r_t \mid \sin\varphi^2 \\
&x_t := e \pm \sqrt{e^2 - r_s^2} \ \ \forall\ e^2 - r_2^2 \geq 0 \\
&b := -x^2 - 2xr_t \\
&y = x_t \sin\varphi + \sqrt{b}\cos\varphi.
\end{aligned} \tag{4.8}$$

Here, φ_t and φ_s are the angle of the track and sensor, respectively, and r_t and r_s are the radius of the track and the sensor. Once the y coordinate is calculated it is compared with the sensor boundaries with respect to it's shift s to create the windmill structure. The position of the sensors' local coordinates can be expressed with the width w of the active area and the pixel pitch p to a pixel column ID via $id_{col} = \frac{y+s}{p}\frac{w}{2}$.
The z-component can be determined with the track's polar angle θ in the following way:

$$z = \frac{y}{\tan\theta} \tag{4.9}$$

and the corresponding pixel ID is $id_{col} = \frac{y+s}{p}\frac{l}{2}$ with the length l of the detector and the shift in the z direction s_z.

Simulation Model of DATCON

Before the implementation on an FPGA, the concepts presented in the last chapter were extensively tested in a simulation environment. This environment is written in C++ [19] so that the suggested algorithm could be implemented much faster and tested more comfortably. The used framework is introduced in more detail later in the thesis (see Section 5.1). First all the required pre-processing stages to prepare the SVD data for the track reconstruction algorithm are described, then the HT and the ROI creation code are presented.
The simulation environment is based on a structure that is typically found in almost every high energy physics experiments. The incoming data from the detector are divided into so-called runs, which last usually in the order of a few hours up to several days. Every trigger accepted by the DAQ system increments a global counter, the event number. All sub-detectors receive the same trigger information containing the current run and event numbers and embed them in their data structure, so that the data can later be merged. Thus, one event corresponds to the minimal independent incident time interval. This structure is also used in the simulation system to mimic all necessary steps involved in the ROI creation.

A simplified illustration of all processing stages is depicted in Figure 5.1 and described step by step in the following sections. Tracks traversing the SVD detector usually hit more than one strip on the same side. Neighbouring strip hits are first combined ("clustered") and translated to a single absolute coordinate, sep-

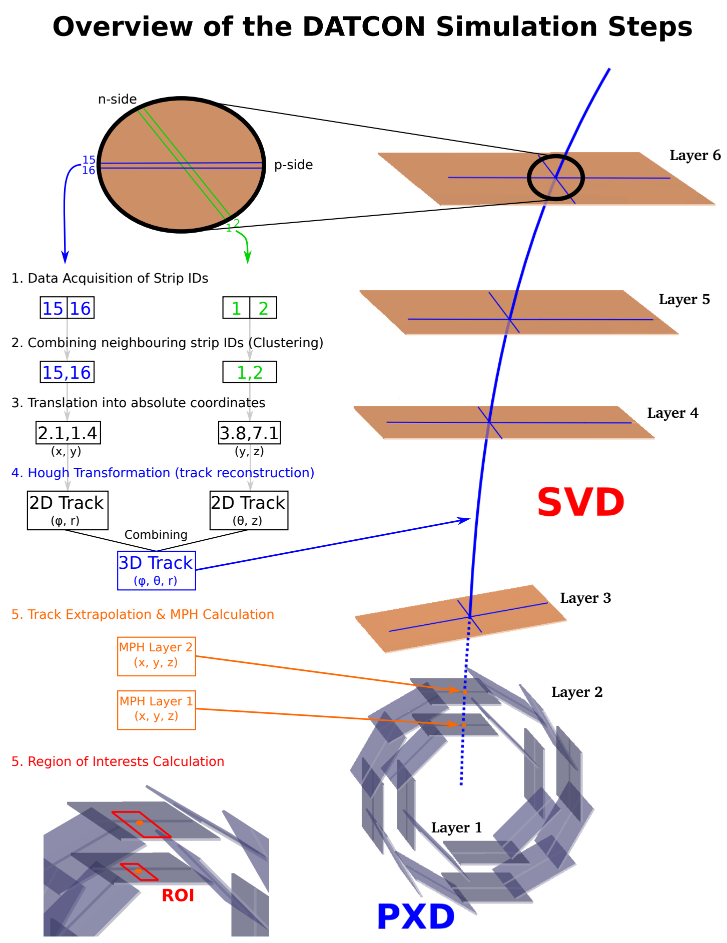

Overview of the DATCON Simulation Steps
n-side
p-side
15
16
1
2
1. Data Acquisition of Strip IDs
15 16
1 2
2. Combining neighbouring strip IDs (Clustering)
15,16
1,2
3. Translation into absolute coordinates
2.1,1.4
(x, y)
3.8,7.1
(y, z)
4. Hough Transformation (track reconstruction)
2D Track
(φ, r)
2D Track
(θ, z)
Combining
3D Track
(φ, θ, r)
5. Track Extrapolation & MPH Calculation
MPH Layer 2
(x, y, z)
MPH Layer 1
(x, y, z)
5. Region of Interests Calculation
ROI
Layer 6
Layer 5
Layer 4
SVD
Layer 3
Layer 2
Layer 1
PXD

Figure 5.1.: Simplified illustration of the different steps involved in the ROI creation process. Exemplarily, the clustering is demonstrated for layer 6. Every fired strip is identified by a unique strip ID. After combining neighbouring strips, they are translated into absolute coordinates and independently searched for 2D track candidates. Two compatible track candidates are merged into a single track parameter set. With the track parameters the MPHs on each PXD layer are calculated and around them the ROIs are created.

arated into two independent pipelines: the strips on the p-side provide the (x, y) coordinate and (y, z) on the n-implantation side. The 2D coordinates from both sides are independently searched for track candidates. Later the candidates from both sides corresponding to the same track are merged into a single 3D track. Using the determined track parameters, the intersection with the PXD planes (MPH) are calculated and around the MPH the ROI placed.
In the first section, the software framework will be described more closely and then the implementation of each pre-processing step. Subsequently, a detailed realization of the HT with certain improvements to increase the reconstruction efficiency is discussed. The chapter closes with a performance evaluation of all critical components.

5.1. Introduction to the Belle II Analysis Framework

The task of a software framework is to assist the user in creating own algorithms and abstract certain complex components, such as data handling, simulation of interactions between particles and matter or system IO. In the Belle experiment, the Belle AnalysiS Framework (BASF) has been used for over 10 years to analyse large amounts of physics data. One of its strengths is that it is not limited to data analysis, but that it can also be used for online data processing. However, certain aspects known from modern frameworks are missing, e.g. object oriented persistency, which is now incorporated in an update of the framework called Belle Analysis Software Framework 2 (BASF2) [20]. Every software-related task in the Belle II experiment, from simulation of physics events with real detector geometry over analysis of test beam data to online processing tasks in the HLT and Data Quality Monit-

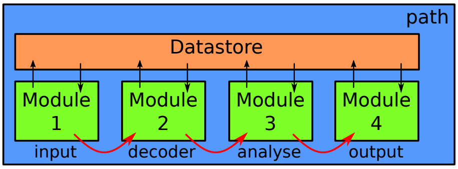


Figure 5.2.: A simple example of a BASF2 module path, illustrating a typical use case of reading data from an input source, decoding, analysing the data and writing the results to a file.

oring (DQM) with visualization is supposed to be written in this framework.
For this reason a modularized approach with a fixed set of virtual functions available to all modules is used. Even the basic input and output system has its own module to read or write to file. Several modules can be arbitrarily ordered and sequentially executed, which is called a "path" configured over a so-called "steering" file. The file is executed with a Python script language interpreter to set up the BASF2 parameters and configure the system paths. In addition, it provides the full feature set of Python to directly branch data into different paths or directly process results from the simulation. Figure 5.2 shows a typical example of such a path with the corresponding steering file. Although the modules are executed in sequence each module may use many different threads[1].

A common data store is used to exchange information between different modules. Once initialized and marked as persistent each data store can be accessed by every module in the path. Each module can in turn create as many data stores as needed for a structured access with a variable scope of persistence, e.g. valid only for one event or run.
On top of the modular design of BASF2 many third-party lib-

[1]A thread is a light-weight process with a minimal structure with little overhead to run on a one Central Processing Unit (CPU) core.

raries are used to include the necessary functionality for various tasks. Among them are the following major ones needed for a realistic detector simulation, performance evaluation and later data analysis:

- **GEANT4:** Simulation framework developed by CERN as a toolkit to simulate matter-particle interactions. Particularly interesting for this thesis is the energy deposited in a silicon bulk by a charged particle track, which is needed to create a charge distribution and estimation of the propagation inside the sensor and thus to simulate a realistic detector response. In addition, GEANT4 includes the basic geometries to model all active and passive components of the detector. [21]

- **ROOT:** A framework to analyse large amounts of data with visualization capabilities. It also provides means to create a full 3D visualization of the detector ("EVE"). [22]

- **libxml:** To abstract the access to basic parameters, e.g. sensor and geometry information, the XML parser and toolkit "libxml" is included. With structured XML files, various aspects of these parameters can easily be changed without modifying the C++ code. [23]

- **BOOST:** A library that builds on the standard library of C++ to provide more comfortable paradigms. It also includes many features, that would otherwise involve third-party libraries like a flexible Cyclic Redundancy Check (CRC) calculation code. [24]

Although GEANT4 offers many different models for charge creation by traversing particles, it cannot directly create a signal as expected by the digital logic in a detector. Because of the large variety of the different sensor concepts, the digitization needs to

be handled by the software framework. In the vertex detector the PXD and SVD sensors are based on different technologies and have their own digitization code, which work in a similar way. When a particle enters the sensitive material of a sensor, the flight path is sampled (saved in the "SimHit" structure) with a fine grade step size, which should be one to two order of magnitudes smaller than the thickness of the active material. In each step the energy loss is calculated and translated into an equivalent number of electron hole pairs. With the knowledge of the diffusion constant in silicon, the applied depletion voltage and the magnetic field (to include the Lorentz shift) the movement and distribution of the charge cloud can be simulated. The electrical signal response in the strip or pixel is then calculated with respect to the properties of the readout electronics and stored for public access by other modules in the data store. A reduced hit information with an averaged position for each track in each sensor plane is additionally provided by BASF2 and stored in the data store for "TrueHits".
Each of these pieces of information, in most cases it is a sample of the signal from the ADC including a simulation of its noise behaviour, is called a "Digit". This is the information the backend electronics needs to handle. In the following section, a module with the complete chain of pre-processing the incoming SVD Digits, track reconstruction and extrapolation, ROI calculation and pixel selection is constructed.
The BASF2 environment also provides the possibility to create tracks for different particle types and different track properties like track angle, energy distribution and vertex origin coordinates. An event generator, which can be used to simulate $B\bar{B}$ events with state-of-the-art knowledge about the decay products expected for B decays can be used to steer the detector simulation.

5.2. Cluster Algorithms

When a particle traverses a sensor and the electron-hole pairs that it creates are distributed over more than one strip, the strips are combined into one cluster. The sensor pitch is usually designed to have an average cluster size of two strips to increase the position resolution, as later seen in Section 5.2.2. For lower incident angles the probability of creating larger clusters increases, so not only the sensor shape and pitch, but also the geometry of the whole detector is important. The design of the detector has an influence on the incident angle. In the next sections an overview of the different clustering techniques is given, which are used for implementation in the FPGA and offer further improvements in coordinate precision.
In the overview in Figure 5.1 the clustering is the second step after the acquisition of the sensor data. Usually the coordinate translation (third step) is directly included in the clustering module.

5.2.1. Next-Neighbour Search

Since a strip sensor provides only one-dimensional information, the simplest clustering algorithm checks consecutively for neighbouring strips above a threshold. The BASF2 simulation provides information about the strip ID on either n-, or p-implantation side[2] of the sensor is given and, depending on the readout mode, a list of one, three, or six samples of the signal. When a strip k is above the threshold, strip $k+1$ is checked for a signal. The last strip ID of a consecutive row is saved as seed strip and so is the number of fired strips per cluster. With this information the

[2]The n-implantation provides the z information, while the p-implantation the r-φ information.

relative position p on the sensor in Cartesian coordinates can be estimated as:

$$p = (s_s - \frac{n_s - 1}{2}) \cdot p_s, \tag{5.1}$$

where n_s is the size of the cluster, s_s the seed strip ID and p_s the pitch of the sensor. To reduce the effect of pile-up[3] events, clusters with a size larger than three strips are split up in chunks. The coordinate resolution σ_p is dominated by the strip pitch and the size of the cluster described by:

$$\sigma_p = \frac{p_s}{\sqrt{12} n_s}. \tag{5.2}$$

5.2.2. Center of Gravity

The resolution of the hit position can be further improved compared to the next-neighbour search by considering the signal of each individual strip. The Center of Gravity (CoG) of the neighbouring strip signal charges are used to weight each position on the strip. Using the CoG, the new position of a cluster is determined by

$$p = \frac{\sum_{i=1}^{n} S_i s_i p_s}{\sum_{i=1}^{n} s_i p_s} \ , \tag{5.3}$$

where S_i is the signal of the strip with ID s_i. The position resolution is now proportional to the Signal to Noise Ratio (SNR)[4] of each strip:

$$\sigma_p = p_s \frac{\sum_{i=1}^{n} ENC_i}{\sum_{i=1}^{n} s_i} \ , \tag{5.4}$$

[3]Two tracks creating overlapping hits resulting in a single cluster are called pile-up.

[4]The number of electrons needed to be collected to create the same level of noise in one strip or pixel.

with the Equivalent Noise Charge (ENC) of each strip contributing to the cluster. However, a systematic effect is still missing and will be corrected in the next section. Figure 5.3 shows an illustration of the CoG algorithm for two different cluster sizes.

5.2.3. Eta Correction

The charge cloud created by a traversing particle spreads out only a few 10 μm around. At a vertical incident angle the particle deposits charge in only one strip unless it hits the gap between two strips. The charge sharing at high incident angles is not normal-distributed and depends on the sensor. It can be corrected with the η-correction method. The CoG and the highest neighbouring peak sample is taken and η is calculated as follows:

$$\eta = \frac{S_l}{S_l + S_r} \ , \tag{5.5}$$

where S_l and S_r are the left and right signal, respectively. The η-corrected position, p_c, of the track can now be calculated with the CoG position p as:

$$p_c = p + p_s \eta. \tag{5.6}$$

5.2.4. Simulation of an FPGA Cluster Algorithm

In view of resource consumption in the FPGA, the decision to develop a simplified cluster algorithm was made very early. While multiplication and division units are very expensive in term of resources, simple subtraction and addition operations can be realized easily in FPGAs. An adaption of the CoG and the next-neighbour search is used. The incoming Digits are sequentially

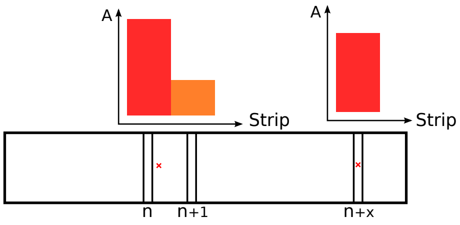
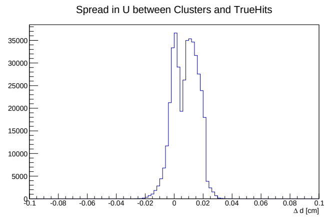


Figure 5.3.: Two incident tracks cause a single-hit $(n + x)$ and a 2-hit n, $n + 1$ cluster. The Amplitude A of each hit and how they affect the position resolution is shown at the top.

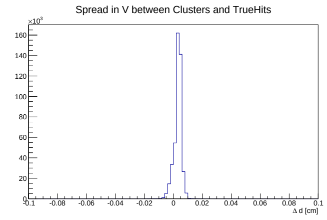
scanned for neighbouring strips (up to four in a row, more are split up) and searched for the peak strip. The CoG of the neighbouring strips with the highest signal is calculated as described in Section 5.2.2. To avoid the devision only the difference of the peak S_p and neighbouring signal S_n is calculated: $\delta_S = S_p - S_n$. An 8-value Lookup Table (LUT) is used to obtain a correction value c for the position. The absolute coordinate of the track is then calculated with:

$$p = s_s \cdot p_s + c \ . \tag{5.7}$$

The performance of the cluster algorithm was extensively tested with BASF2, the full detector geometry and realistic settings for the digitizer taken from latest noise measurements of the APV25 and ADCs. This includes electronics effects in the ADC and a noise level of 2000 e^- in the APV25 with a shaping time of 50 ns. With the so-called particle gun electrons and positrons are generated in a momentum range from 50 MeV to 2 GeV and with uniformly generated θ and φ angles within the acceptance range of the SVD. The created clusters and their computed positions are compared with the from GEANT4 calculated hit position (so-



(a) Difference between TrueHit position and reconstructed cluster in U.



(b) Difference between TrueHit position and reconstructed cluster in V.

Figure 5.4.: Performance of the cluster algorithm for both implantation sides from tracks with energies between 50 MeV and 3 GeV in a sample of $50,000$ events.

called TrueHit) of the tracks and plotted separately for U [5] in Figure 5.4a and V [6] in Figure 5.4b. While the position resolution is better than the strip pitch, one sees a systematic shift of the curve to the left for electrons and positrons. This can be explained by the not-corrected Lorentz angle on the p-side and the lack of an η-correction. It should be noted as well that there is of course a deviation between the averaged TrueHit position of the tracks flight path through the sensor, caused by the non-vertical incident angle of the track. The smaller deviation in V is caused by the smaller strip pitch.
Despite the non-Gaussian form of the distributions, the position resolution is better than the individual strip pitch with a minimal effort in calculation. The algorithm can be further improved by correcting the systematic shift and assigning a finer LUT with η-correction.

[5] Local coordinate on the sensor for n-implantation side. The naming convention is used in BASF2.
[6] And p-side implantation.

5.3. Iterative Fast Hough Transformation

The coordinates reconstructed by the clustering algorithm are searched for track patterns using the Hough Transformation as described in Section 4.2. This section focuses on the implementation in the simulation environment to mimic the FPGA algorithm for a comfortable measurement of its efficiency and speed performance as reference value for a later comparison with the FPGA implementation. In the overview in Figure 5.1, this is the fourth step.
The SVD provides only one-dimensional local information, which can be translated into two-dimensional absolute space points using the detector's x-position. To obtain a 3D track, two 2D HSs need to be searched and later combined to a 3D track. The first step is to search independently for track candidates in two different HS, so intersections of transformed detector hit coordinates need to be found. Since an algebraic solution would mean to diagonalize a matrix with $n \times n$ entries, a different solution to the intersection finding problem will be presented in the next sections.

5.3.1. Implementation

Incoming coordinates from the cluster engine will be transformed into the HS according to Equation 4.3. The magnetic field affects only charged particle tracks in the r-φ plane, so these coordinates will be mapped into the conformal space with the vertex point set to $(0, 0)$ before the HT. Figure 5.5a shows an example of the HS of two tracks with different track parameters. Because of the new proposed parametrization, straight lines become linear combinations of sine and cosine curves. To find the intersections of these lines, the HS is divided into small sectors and each sector is checked for the number of traversing lines. If a minimum number of lines is found, this sector is divided into smaller sectors until

a specific number of iterations is reached. The sequence of this is depicted in Figures 5.5b to 5.5g. Each sector which by then still contain the minimum number of lines is from now on called an “active sector”. A “track candidate” is a set of active sectors corresponding to the same track.
For the simulation in BASF2 the HS search is implemented as an iterative function. Once the required numbers of lines are found - four in this case, one hit in each of the four SVD layers - the sector is divided by two in the horizontal and vertical axis. This results in four new sectors, each called by the same iterative function. A few advantages of this “divide and conquer” algorithms are listed below:

- Computing time is reduced, since only active sectors with the minimum number of hits are further processed. Empty sectors, or sectors with not enough hits are ignored.

- The parameters of the track candidates are extracted from the active sector(s), which are identified by the coordinates of their outer edges.

- Error tolerance, because no intersection of four lines in a single point is required, but only a region that depends on the sector size.

As seen in Figure 5.5i, the lines do not exactly intersect in a single point, because of several reasons:

- Multiple scattering caused by the active and passive sensor material and support structures affects the tracks trajectory and let it deviates from its ideal flight path.

- As seen in Section 5.2.4, the cluster position and translation to absolute coordinates have an uncertainty.

- Misalignment of the detector: due to external mechanical influences, thermal expansion or sagging of the sensors under their own weight, the calculated coordinates differ from the real hit coordinate.

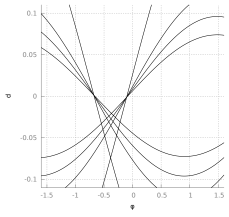
- In the r-φ plane, the active magnetic field shows inhomogeneities, which lead to slightly different curvature radii of the tracks.

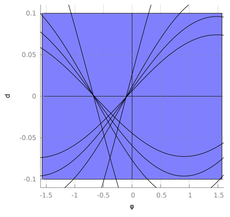
- The conformal mapping assumes the vertex point at (0, 0), but the average flight length of particles in a B-decay is around 150 μm. The z-axis is unaffected, since no conformal mapping is required there.

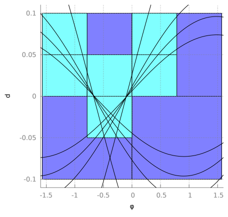
The size of the sectors and - directly connected with this - the number of iterations has a large impact on the track parameter resolutions. A compromise is required between tolerance for the track finding and precision of the track fitting. The track parameter's resolution is improved by averaging over all active sectors outer edge coordinates.

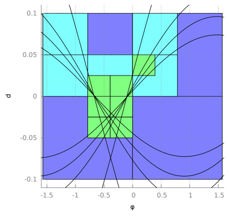
The most time critical path in the iterative function is the determination if a HT hit crosses a sector. The calculation is illustrated in Figure 5.6 with the coordinates of the sectors' edges v_1 to v_4. The y-component for every hit at x_v and x_v for each sector is calculated and compared with the y-values of the edges. The condition of "*no* sector hit" with the HT function $\mathcal{H}$ is: $(\mathcal{H}(x_v) > y_v \wedge \mathcal{H}(x_v) > y_v)$ or $(\mathcal{H}(x_v) < y_v \wedge \mathcal{H}(x_v) < y_v)$. This is valid only for a strictly monotonic function which is true, at least in the region focused on the search for track candidates and smaller sectors. The negation of this condition is used to determine if a sector is active.

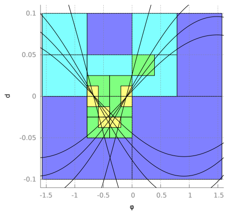
An optional feature to provide a more precise track parameter reconstruction is to allow for more iterations than the critical limit. If in the even smaller sectors no tracks are found anymore, the values from the last iteration are taken. The number of iterations

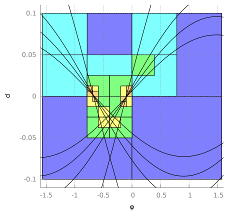

(a) HS of 2 tracks

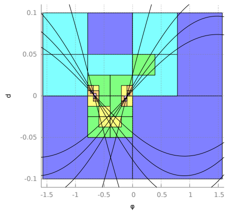

(b) Iteration 1

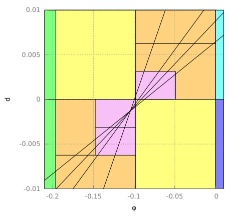

(c) Iteration 2


(d) Iteration 3


(e) Iteration 4


(f) Iteration 5


(g) Iteration 6


(h) Zoom to one intersection

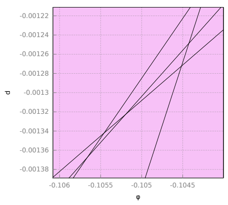

(i) Magnification of one sector at the intersection

Figure 5.5.: Illustration of the Hough track finding algorithm for different iteration steps, visualising the iterative process.

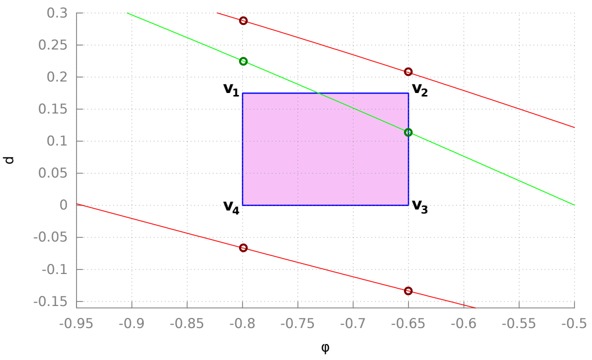


Figure 5.6.: Detection illustration whether a line crosses a sector determined by its outer edges v_1 to v_4. The green line is accepted, but not the red ones.

can be directly used as a measure for the quality of the track, which is later important to calculate the size of the ROI. This extension was evaluated, but is not used, since it yielded only in a very small increase in precision.

As seen in Figure 5.5h, each track activates more than one sector in the HS. However, in the end only one single parameter set per track is desired. A few methods to detect which sectors to merge and how this can increase the parameter resolution is described in the next section.

5.3.2. Sector Merging

After the HS is searched and the active sectors are found, the parameters containing the sectors coordinates are stored in an array. Several, up to a few dozens of sectors can be related to the

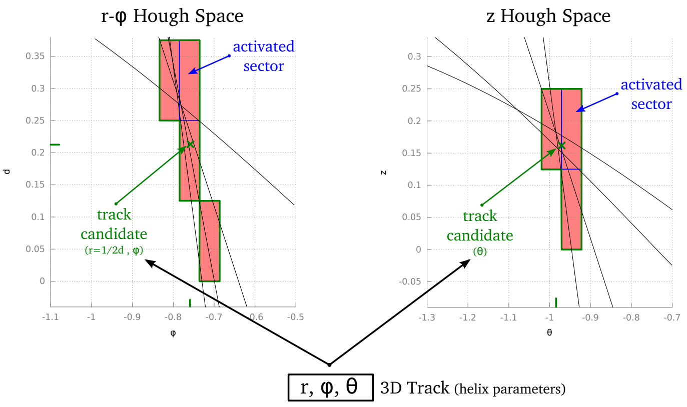


Figure 5.7.: HS sector merging process to create track candidates in the $r - \varphi$ and z-plane. Merging of two compatible track candidates into a single 3D track.

same track, depending on their position and transformation in the HS. To find compatible related sectors corresponding to the same particle track, three methods were tested and are described in the next section. Figure 5.7 depicts the process of finding clusters in each HS to creating track candidates. The merging of two track candidates from the $r - \varphi$ and z-plane in one track is also shown, described later in Section 5.3.3.

Hash Merging

To save resources and to speed up the comparison process between sectors, a hash algorithm was defined to measure the "similarity" of hits activating a specific sector. When a line is located inside a sector, the internally generated hit ID (an incremented counter for every particles hit within one event) is added to the hash sum of the sector. Hits corresponding to the same sectors have the same

hash sum and thus can be easily merged by a simple comparison of the hash sums. Coordinates of sectors with equal hash sums are averaged and stored in an 2D track list.
However, the efficiency of this hash merging algorithm proved to be poor during testing when dealing with many background hits in the detector. If a sector contains an additional hit, e.g. a background hit the hash will be different. The hash sum is ambiguous in rare cases, where different hit constellations create the same sum. This results in a very high track fake rate and loss in precision when dominating seed sectors are missing during the averaging process.

Hit ID Merging

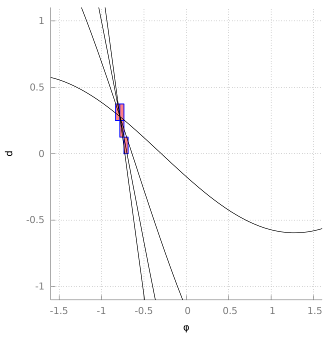
When storing a list with the hit IDs for every sector the ambiguity of line combinations in sectors can be circumvented. A majority function decides which sectors are merged by counting the number of equal hits. If the number of similar hits is equal to or larger than that of different ones and a minimum of at least three equal hits are included, the sectors are merged. Compared with the hash merging algorithm this provides a unique identification of hits inside a sector.
On the other hand, the resource consumption is higher, because each hit ID needs to be stored in conjunction with every sector. In addition, all hit IDs from every sector are compared with every other sectors' hit list, which is a very time consuming task. Another complication is that sectors are merged which are separated by a large distance. Tracks with an φ or θ angle of $90°$ and $-90°$ suffer from these effect with activating sectors at both ends of the HS. With a $30°$ cutoff value, clusters farther apart than these $30°$ create different track candidates. This increases the number of fake tracks, but also prevents the track from being reconstructed with completely wrong angles.

Cluster-Based Merging

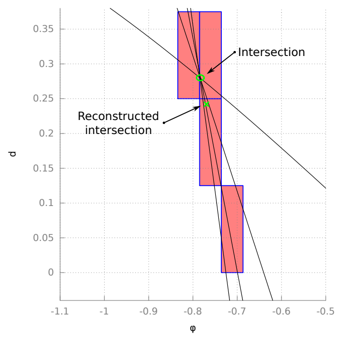
Since the active sectors of a track are clustered around the intersection point, a cluster-based merging procedure was evaluated. Partly the code for the strip clustering can be reused here which also simplifies the FPGA development. The obstacles of the hit ID merging process described in Section 5.3.2 can be completely avoided. In contrast, pile-up events creating large clusters cannot be separated or a maximum cluster size needs to be defined.
With the CoG the precision of the hit position can be improved by calculating the center of the cluster. Although the cluster shapes are different from what can be found e.g. in the pixel hit map, the principle works as well. A close concentration of many directly neighbouring sectors indicates the intersection of HS lines. Figure 5.8 shows a one-track example in the HS. In Figure 5.8a the active sectors are marked in red with a zoomed view in Figure 5.8b. As illustrated, the reconstruction of the vertex point is more precise than taking each sector's central point alone.

5.3.3. 3D Track Candidate Merger

So far the track candidates have only two-dimensional information. To obtain the full 3D helix parameters a second merging step is required to identify compatible partners in the HS of $r - \varphi$ and z (as illustrated in Figure 5.1 step four, second part). Finding these compatible partners is simple when using either the hash or hit ID merging information, since all the identification information is already available. Although this method would require additional computing time, because every hit ID list or hash sum of each track candidate in $r - \varphi$ must be compared to every track candidate in z. This results in the same advantages and disadvantages as for the 2D merging process.
With the cluster merging algorithm no information about how to



(a) One track with active sectors (red). Four hits represented as black curves in the HS.



(b) Zoom of one Hough cluster, showing the calculated and real intersection of the lines.

Figure 5.8.: Illustration of the CoG algorithm in HS for an one-track example comparing reconstructed and real intersection of lines.

merge track candidates from two different HS is available. One can of course merge every track in r-φ with the one in z, but this would result in a huge number of tracks. A more efficient way is to merge tracks that are originated from the same sensor to reduce combinatorics. This will also produces fake tracks, but in a much diminished form.

5.3.4. Track Reconstruction Definitions

The same system with similar settings as for the cluster performance testing are used to evaluate the performance of the Hough tracking system. First, a few useful definitions are shown, which help to measure the performance of the system. The track efficiency ϵ is defined as the number of correctly reconstructed tracks, n_{cor}, divided by the total number of generated tracks n_{tot}:

$$\epsilon = \frac{n_{cor}}{n_{tot}} \; . \tag{5.8}$$

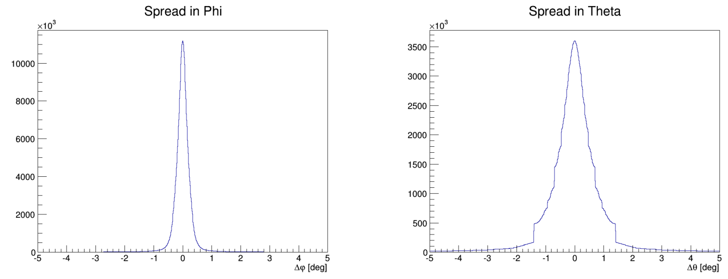
A second measure for the performance of the track reconstruction is the fake rate, f_r, which is defined as the average number of additional (fake) tracks per event:

$$f_r = \frac{n_{reco}}{n_{cor}\; n_{events}} \; , \tag{5.9}$$

with the total number of reconstructed tracks, n_{reco}, and the number of simulated events, n_{events}. The goal is to reach a track efficiency close to one, while keeping the fake rate at minimal level. In order to determine if a track is correctly reconstructed, the parameters (in this study the angles φ and θ) of the generated tracks are compared with the reconstructed angles of the track. In Figure 5.9a and Figure 5.9b, the spread[7] of φ and θ, respectively is plotted for different energies. The lower the momentum, the higher is the impact of multiple scattering, which results in a larger spread. Based on this distribution, an arbitrary cut is defined which defines a successful reconstructed track, once it is below the cutoff value. For φ this cut is set to $1.0°$ while θ has a larger cut value of $4.0°$, which is later in this section justified. It needs to be noted that this track reconstruction algorithm is only used to define ROIs. Even a discrepancy of one degree between the real and the reconstructed track causes only a deviation of 0.38 mm in the outer PXD layer. This is, assuming 75 μm pixel pitch, 5 pixels. But usually the reconstructed track parameter spread as seen in Figure 5.9a is lower than the chosen $1.0°$.

The θ angle is reconstructed by approximating the particle's trajectory in z with a straight line as depicted in Figure 5.10. This

[7]The difference between reconstructed and generated track parameter plotted accumulated for many events.



(a) Spread of φ between reconstructed and generated track.

(b) Spread of θ between reconstructed and generated track.

Figure 5.9.: Spread of φ and θ to motivate the definition of a reconstructed track and as illustration of the precision of the algorithm.

is accurate for a high-momentum track. For lower momentum the particle starts to curl in the detector, resulting in a sine shape trajectory, which can only be extrapolated accurately for a forth period of the cycle. This is the reason for choosing a larger θ tolerance to accept a track, because this will later be compensated by creating larger ROIs in the z direction.

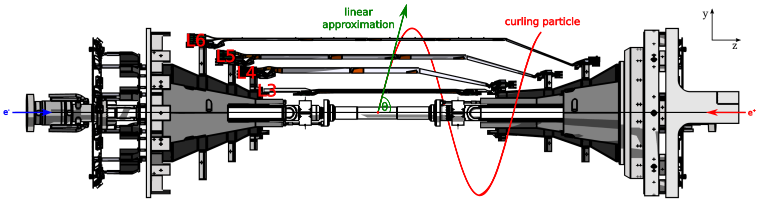


Figure 5.10.: Illustration of the linear approximation of a curling track in θ.

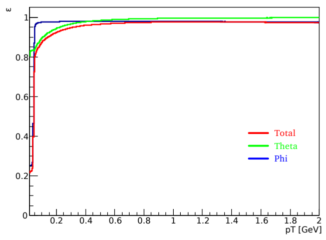
5.3.5. Track Reconstruction Results

The track reconstruction performance is important to reliably provide precise ROIs to the PXD. In order to create a realistic situation for the track reconstruction module to quantify the performance, the following settings for the BASF2 simulation are chosen:

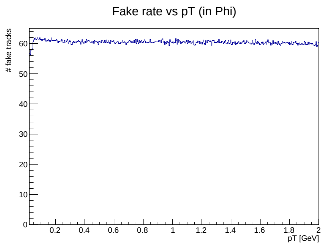
- **Particles:** e^+, e^-
- **Energy:** 50 MeV...3 GeV
- **Angle θ:** 17...150°
- **Angle φ:** −180...180°
- **Tracks/event:** 10
- **Events:** 100 million
- **Background:** Two-photons (QED), Touschek, Radiative Bhabha (RBB) and Coulomb

The latest backgrounds as discussed in Section 2.3 are used in the simulation since it will induce a lot of unwanted tracks and thus influence the fake rate. To speed up the simulation process and to mimic the off-time trigger occupancy reduction of the SVD, only background hits in the interval from −10 to 10 ns around the event are included (in the end it is planned to have a time resolution of around 3 ns [54], so a further reduction is possible). During the course of this thesis no simulations were available for the synchroton radiation, but for the SVD this is not a dominant background.
Figure 5.11a shows the track reconstruction efficiency versus p_T for φ, θ and in total, including the full 3D track information. The reconstruction efficiency in φ is constantly high until it reaches a breakdown at about 40 MeV, when the particle tracks cannot reach the outer layers (layer 5 and 6) anymore. The angle θ shows a constant decrease of efficiency starting above 300 MeV, caused by the linear approximation of the particle's trajectory in z direction as mentioned in Section 5.3.1 and depicted in Figure 5.10.

The overall 3D track reconstruction efficiency is 93.2 percent. Most of the not correctly reconstructed tracks are due to missing hits in the detector or pile-up hits. The number of fake tracks of around 60 tracks per event is almost constant as seen in Figure 5.11b. A slight increase towards lower momentum can be observed, caused by multiple scattering. Although 60 additional tracks per event seems quite high, its effect on the data reduction is limited, because of two different reasons. First, most events have much lower fake rates, but only in high occupancy events the fake rate is very large, contributing most of the fake tracks. The second effect is, that most of the fake tracks are in close proximity to the original track, which would create overlapping ROIs and thus will not greatly affect the data reduction as seen in Section 5.4.
The same simulation settings were used with different particles (π^+, π^- instead of e^+, e^-) and different numbers of generated tracks going from 1 to 15 per event, showing similar results. A slight increase in reconstruction efficiency towards higher numbers of generated particles were noted.[8]



(a) Track efficiency as a function of p_T.



(b) Track fake rates as a function of p_T.

Figure 5.11.: Track reconstruction efficiency and fake rate.

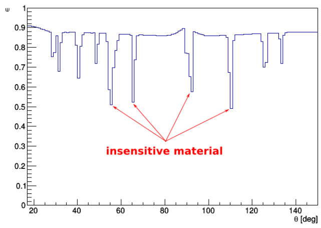
[8]Due to the increasing numbers of tracks the probability is higher that a reconstructed fake track matches with a generated particle.

For a later comparison with the FPGA implantation, the speed of the algorithm was measured by adding a time measurement environment around the critical path of the tracking pipeline, including all relevant steps from the HT over the sector finding to the calculation of the helix parameters of the tracks. No setup time of the simulation environment is included. The time is of course very dependent on the number of hits to process and their location in the HS. A track reconstruction with 250 detector hits takes 120 ms on a Intel(R) Xeon(R) CPU X5650 with 2.67 GHz, dominated by the search of the HS and on second position the clustering of the sectors.[9]
The next section provides some more details on how the track reconstruction algorithm was improved until it provided the shown sufficient track reconstruction efficiencies, fulfilling the Belle II requirements.

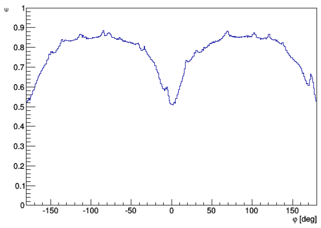
5.3.6. A Selection of Performance Enhancements

In this section an incomplete list of improvements to the Hough track reconstruction algorithm is provided that made the performance results shown in Section 5.3.5 possible. In three simulation campaigns the algorithm was optimized. The track reconstruction efficiencies of the first campaign are shown in Figure 5.12a for θ and in Figure 5.12b for φ. In comparison Figure 5.12c and Figure 5.12d depicts the same plots with the included improvements from the third and final simulation campaign. To achieve these performance enhancements the following incomplete list of modifications were introduced in the original track reconstruction algorithm from the first campaign:

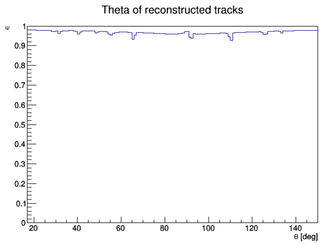
[9]Still some improvement ideas were available to increase the speed in the HT code path. For instance the hits can be sorted by layer and in case two layers show no hit in a single sector the remaining hits to check can be skipped. But this was beyond the scope of this thesis.



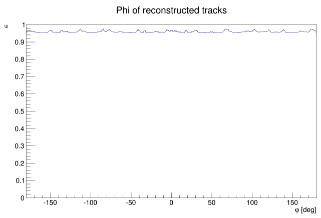
(a) Track efficiency on n-side (first simulation campaign).



(b) Track efficiency on p-side (first simulation campaign).



(c) Track efficiency on n-side (third simulation campaign).



(d) Track efficiency on p-side (third simulation campaign).

Figure 5.12.: Track reconstruction efficiency as a function of θ and φ compared with the first and second simulation campaign.

- Reducing the minimum number of lines required to activate a sector from four to three. That removed the dips in θ caused by insensitive material, where the SVD sensors are glued together. It also helps to increase the track reconstruction performance, when a low-momentum track only passes three instead of four layers in the SVD.

- Implementation of the cutoff value in the HS clustering module as described in Section 5.3.2, removed the double peak structure with the dips at -180, 0, and 180° in φ.

- A "layer filter" greatly improved the fake rate situation by a factor of 10 and increased the overall tracking performance. Only hits coming from at least three different SVD layers can activate a sector.

- Special online configurable "cut values" are used in the r-φ-plane to remove track candidates with a radius below 10 cm, which can not be originated from the IP or when the intersection with the z-axis is outside of the default Belle II z-vertex constraint of ± 7 mm. This reduces the fake rate by another factor of 2 to 3.

A last optimization was performed with a new steering program which automatically sweeps through different HS parameters (e.g. sector size, number of iterations), in order to find the optimal settings of parameters for the n-side and the p-side. A delicate balance has to be found between high tracking efficiency and a low fake rate. The detailed description of the program that was developed with a bachelor student, Christian Wessel during the course of his thesis, can be found in [38].
With the new optimization settings found, events with high occupancy are more efficiently processed with lower fake rates. An HS example of such an event is depicted in Figure 5.13, which shows the high density of lines in the HS with over one thousand transformed detector hits. When more than $10,000$ hits in the detector are found in a single event, the event is marked as "high-occupancy" and is skipped to keep within the limits of the processing time per event.

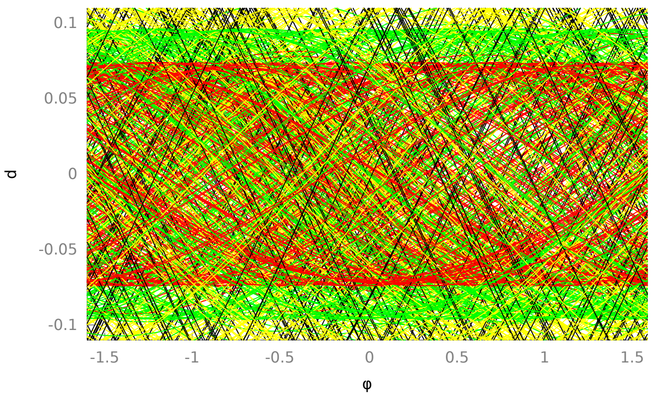


Figure 5.13.: HS of a high-occupancy example event. The different colors of the lines are assigned to the different SVD layers. The event contains over $1,000$ transformed SVD hits.

5.4. Region of Interests Calculation

Once the helix parameters of the reconstructed tracks are obtained, the ROIs are calculated. Equation 4.8 is used, as described in Section 4.3 to extrapolate the MPH (see step 5 and 6 in Figure 5.1) and calculate the pixel and sensor ID for the particular PXD half-module. In the first version a fixed size of the ROI was used with 16×16 pixels. The size of the ROI is divided by half and subtracted respectively added to the pixel ID of the MPH, which defines the outer boundaries of the ROI. This is illustrated in Figure 5.14. Thereby it is important to check the boundary of the sensor to prevent an ID overflow, when the MPH is close to the border of the sensor. Since there are several indicators of the track quality and momentum information is available from the track radius, the ROI size was changed to be variable. In

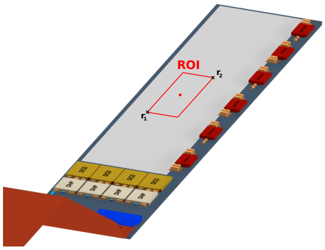


Figure 5.14.: One ROI on the PXD, which is defined by the lower left pixel row and column ID r_1 and upper right r_2.

addition, to prevent and to take curling tracks in the PXD into account, the ROIs are extended in the z direction. The following three different ROI sizes are used:

- **8×16 px:** high-momentum with reconstructed radius down to 100 cm.

- **10×64 px:** medium-momentum with reconstructed radius from 100 to 50 cm.

- **12×160 px:** low-momentum with reconstructed radius below 50 cm.

The smaller size in the r-φ direction also reflects the higher track reconstruction efficiency in the p-side of the sensor, due to the smaller pitch and higher strip density. Figure 5.15 shows the difference between the MPH and the TrueHit in the PXD. Most of the hits are within a few pixels distance, but there is a constant continuum of hits at higher values, caused by low-momentum tracks. To still include these in the ROI, the ROIs are chosen to be narrow in $r - \varphi$ but long in z direction at low-momentum.

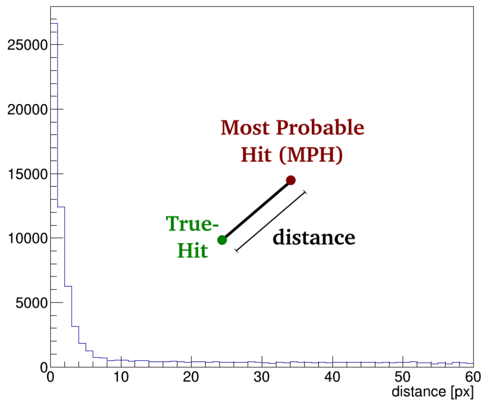


Figure 5.15.: Difference between PXD TrueHit and MPH measured in pixels (with a pitch of 50 μm.

The Data Reduction Factor (DRF) is defined as follows with the total number of PXD hits, n_{tot}, and the number of selected pixel inside an ROI, n_{sel}:

$$DRF = \frac{n_{tot}}{n_{sel}} \ . \tag{5.10}$$

To fulfill the Belle II requirements the DRF should be at least 10. The ROI efficiency is defined as the relation between the physics relevant hits inside the ROI and the total number of physics hits in the PXD. With the same settings for the simulation and the Hough tracking as described in Section 5.3.5 and the variable ROI calculation a ROI efficiency of **95.2 percent** with a DRF of **45** is reached more than 4 times higher as required.

In future, further improvements can be made by eliminating the cause of missing hits in the PXD:

- Track inefficiency: as seen in Section 5.3.5 not all tracks can be recovered, e.g. because of missing hits in different detector layers.

- Low-momentum tracks: the θ angle has a systematic error caused by the linear approximation of the particle's trajectory in z as described in Section 5.3.4.

- Very low-momentum tracks ($p_T \leq 35$ MeV), which do not reach the outer SVD layers, cannot be reconstructed at all.

- Missing curler interpolation: Only one MPH is calculated per PXD layer and track, but low-momentum tracks can curl several times through the detector. This is partly but not completely compensated by using narrow but long ROI in the z direction.

The efficiency may further improve when applying the parameter sweep of the HT to optimize the ROI efficiency and parametrizing the accurate sine shape of the tracks' trajectory in the z direction.

DATCON Hardware

The hardware of the DATCON data reduction system is based on FPGA chips from Xilinx. A few basic properties of an FPGA and the additional features it can provide are explained in the next section. Three different FPGA boards are directly involved in this project. The FTB is equipped with a Xilinx Spartan 6 chip, designed as rear transition module to the FADC VME board. Two different boards are used to realize the DATCON system: The first type, called “concentrator AMC”, handles the incoming data stream from up to four FTBs and is based on the same hardware as the ONSEN system, but with a lower resource Xilinx Virtex 5 FPGA. The details about the pre-processing and its main responsibilities are described in Section 8.
For the track reconstruction and ROI calculation (as stated in Section 7.1) an FPGA with more resources is required. This board will be equipped with a Xilinx Virtex 6 and are called “tracking AMC”. An overview of the interconnection of the different boards are shown in Figure 6.1.
All DATCON FPGA boards are designed in the Advanced Mezzanine Cards (AMC) standard, carrying four optical high-speed serial link transceivers with a maximum bandwidth of 6.25 Gbps to obtain the data from four FTBs in parallel. In addition, these AMCs are equipped with a 6-port backplane, each port connected to another high-speed transceiver with similar bandwidth. Over the backplane connection the AMCs can communicate control information, distribute several clock signals, and exchange data.
The properties and the design of a backplane to fulfill the requirements of the DATCON system to aggregate all data in a single AMC is described in the last section. A more detailed discussion

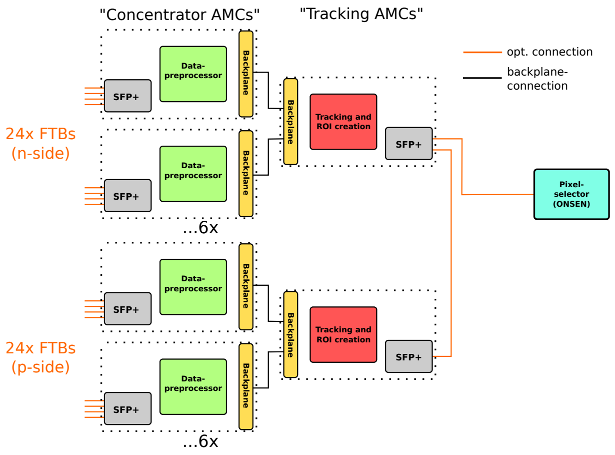


Figure 6.1.: Connection topology of the DATCON system: 12 “concentrator” AMCs collect the data from the SVD FEE separated into two groups for n- and p-side. Each group is connected over a backplane to the “tracking” AMCs. The track parameters between the two groups are exchanged over one optical connection.

of the different standards mentioned and the tools to configure and monitor the FPGAs are provided.

6.1. Introduction to Field Programmable Gate Arrays

Simply speaking an FPGA is a chip where the internal logic can be freely configured by loading a configuration, so-called firmware (also sometimes bitfile or bitstream) into the internal memory,

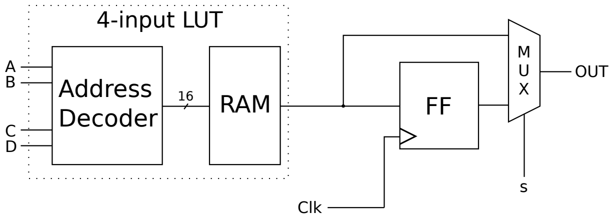


Figure 6.2.: Simplified layout of the Configurable Logic Block. The address decoder drives with respect to the input values the corresponding bit in the Lookup Table (LUT) to the output.

which defines the behaviour of the logic cells and their interconnection. Every FPGA consists of many of these logic cells, so-called Configurable Logic Block (CLB). Each is constructed as a 16 bit wide storage, known as LUT (also sometimes called distributed Random Access Memeory (RAM)), which includes an address decoder. The address decoder has four inputs to drive the storage memory. For each input combination a different bit of the storage is addressed and forwarded to the output, where it can either be directly routed inside the FPGA or stored in a clock steered Flip Flop (FF). A simplified illustration of this CLB is shown in Figure 6.2.

All these CLBs are arranged in a matrix and can be freely connected to each other over programmable routing resources. By combining different CLBs, every combinatorial, or state driven logic can be created. In addition, every FPGA has further dedicated memory, so-called Block Random Access Memory (BRAM), which offers the possibility to route the output to different CLBs, or pads. The main purpose of the Block RAM is to hold and save large amount of data, either provided by some combinatorial logic, or directly programmed during configuration.

Furthermore, there are so-called Digital Clock Management (DCM)

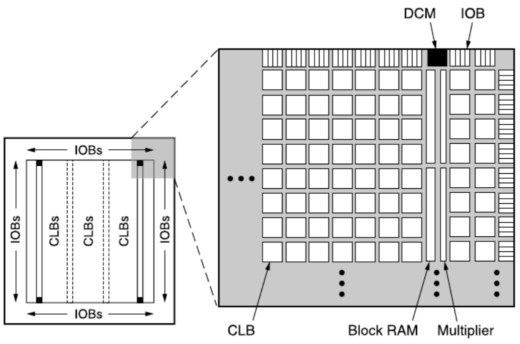


Figure 6.3.: The layout of a Xilinx Spartan 3 FPGA with CLBs arranged as matrices enclosed by by the Block RAM, Multiplexer and Input/Output pads. [43]

units installed with their own low-skew dedicated clock routes, that ensure a good quality of the sensitive clock signal through the whole FPGA. The DCM provides special configuration registers for additional properties, e.g. multipliers and dividers to set up a specific frequency or phase shift. Special global buffers and user constraints ensure the correct distribution of the clock signal.

All these structures are arranged in a special pattern as depicted in Figure 6.3, showing the basic layout of the Xilinx Spartan 3 FPGA. The CLBs are arranged at the center in matrices, enclosed by Block RAM and multipliers (Digitial Signal Processor (DSP)s), used for digital signal processing. The matrix layout provides an efficient way to interconnect different CLBs. On the edges, the Input/Output Buffers (IOB) are placed, as the connection to the outside world, which can also be configured. They have special registers to set up an optional pull-up/down resistor for inputs and configuration of the IO standard, which determines the digital output voltage levels. All these different FPGA structures

can be programmed with a bitstream that contains the values for all configuration memories and defines the functionality of the FPGA. There are special tools like the Xilinx ISE, assisting in the creation of these configuration data in an abstract way. So it is possible to construct the logic in an HDL [46] code and the tool chain will translate this into a configuration file that can be directly loaded onto the FPGA.
For the following project the Verilog Hardware Description Language [46] is used. In order to create a configuration file from the Verilog code, three major tasks have to be executed. In the first stage, the Xilinx tool chain will translate the Verilog code into a native generic netlist, independent of the used HDL. Not all of the functional possibilities in the Verilog language can be directly mapped to CLBs, so first a translation of the not directly synthesizable logic has to be done. For instance the use of DSPs are recommended when dealing with multiplications, which is taken care of at this stage. Afterwards all the used CLBs have to be selected, placed and connected to each other. Further, the clock has to be distributed to the clock driven CLBs. During all these different placement and routing stages, several optimization steps are executed, e.g. a Quine McCluseky algorithm [47] to strip off redundant logic in the design. In addition, the timing behaviour of the system is simulated and cross checked with user-defined constraints.
If the design has no obvious connection errors and all logic could be translated into CLBs and the routing and placement meets all user constraints, a configuration bitfile is created. This file can now be loaded onto the FPGA. The industry standard protocol for this transfer is defined by JTAG and is quite common to transfer the configuration file directly into the target. In standalone mode, the FPGA can obtain its configuration over a permanent Electrically Erasable Programmable Read-Only Memory (EEPROM), also programmable over JTAG.
Finally it's possible, sufficient FPGA resources assumed, to build any combinatorial and state logic inside an FPGA without design-

ing and lithographing real wafers. In conclusion, FPGAs are very useful, especially when connecting and controlling different devices with strict timing behavior. However, most FPGAs are not running at very high frequencies like custom ASICs, e.g. a computer processor with a clock frequency of a few GHz, but they can outrun any PC when it comes to massive parallel computation.
In addition, most FPGA provide several additional features, depending on the model. All FPGAs used in this project are equipped with RocketIOs, high-speed LVDS link transceivers used for communication over electrical, or optical cables. Double Data Rate 2 (DDR2) RAM interfaces, Ethernet, Universal Serial Bus (USB) to Universal Asynchronous Receiver/Transmitter (UART) and even a complete processor using the PowerPC, or ARM architecture can be found inside. Since the high-speed data transmission is of special concern in this project, a short description about the RocketIO is given in the next section.

6.1.1. RocketIO and Aurora Protocol

Many FPGAs from Xilinx are equipped with high-speed serial link transceivers, called RocketIOs. They are primitives inside the FPGA containing several functional blocks to ensure a stable multi-gigabit communication, including de-/serializer, comma-detection, transceiver and receiver, a programmable transmitter pre-emphasis and receiver equalizer to boost and enhance the signal, respectively. Xilinx distinguish between several types of RocketIOs, depending on the FPGA model:

- **GTP:** Dual transceivers for RX/TX communication with up to 3.25 Gbps. These are used in FPGAs of the FTB and evaluation boards.

- **GTX:** The basic construction is similar to GTP, but with higher link rate of up to 6.25 Gbps. They are used in the DATCON system. In addition they provide an encoding of 64b/66b and 64b/67b.

- **GTH:** Quad transceivers with even higher bandwidth up to 12.5 Gbps, but not used in this project.

There are also several other models with different properties and even higher bandwidth available, but they cannot be used here. The GTX/GTP tiles in the FPGA are connected over I/O pads and lines routed on a Printed Circuit Board (PCB) to Small Form Plugable (SFP) cages. In these cages different modules can be placed, for instance optical drivers with different speeds, or Gbit Ethernet modules. A direct routed electrical connection to a backplane connector is also possible and used for this project.

In order to establish a duplex communication with two participating partners, a layer-one[1] protocol is required to ensure a reliable connection on bit transmission level. One among many of these protocols is called Aurora [45], developed by Xilinx under a free license model. It provides a low-level handshake to establish a connection, indicating if the connection is up and operational, e.g. if the cable is correctly connected and able to transmit data. After the channel negotiation, Aurora sends a test idle pattern over the line to determine if the connection is reliable. If this is successful, the data transmission can begin. In order to transmit user data, a line encoding has to be selected to ensure the integrity of the waveform signal by balancing the voltage level. This encoding is also handled by the Aurora protocol. Therefore, in

[1]The Open Systems Interconnection model (OSI) standardizes and describes the typical characteristics of high-speed communication and divides them into seven groups going from low-level (1) bit transmission to high-level (7) application protocol, e.g. HyperText Transfer Protocol (HTTP).

the case of 8b/10b encoding, each 8 bit user data packet will be assigned to a corresponding 10 bit symbol. This symbol will be en- and decoded through a lookup table defined by a common standard. The encoding offers special control symbols, which are not considered as user data. A few of these control symbols are discussed now.
To compensate for frequency variances between different clock sources, special clock correction symbols to align an elastic buffer to ensure the correct bit alignment, are used. The receiver side has no knowledge about the beginning or the end of a data word in the bit stream. To align the data stream some special alignment symbols are sent and shift registers used for deserialization are adjusted to it, so the fully aligned data stream can be acquired.
In addition, it offers the possibility to frame the data into packets, therefore two wires are provided to the user logic for "Start Of Frame" (SOF) and "End Of Frame" (EOF). When these wires are asserted[2], the data word, which is about to be sent, is combined with a special character to indicate if it is the start or end of this frame. A constant data stream without framing is also possible, but not used in this project.

Aurora also provides some functionality to increase the bandwidth by combining several, so-called lanes. In terms of Aurora, a lane is a data line duplex connection, i.e. one optical fibre line each for data transmission and reception. Up to four of these lanes can be combined by Aurora to build one channel. During this project the Aurora protocol was modified to support independent data transmission on lanes from the same GTX transceiver.[3]
The Aurora protocol is well designed and has proven to be reliable

[2]In this context the word assert means activated, so a logical one is applied on a wire or register.

[3]Each GTX tile has two transceivers, which can in principle run independently of each other. However, Aurora automatically combines them into one channel with two lanes. Thus a modification was required.

with a considerably low overhead. Because of these properties it was chosen as foundation of the communication between FTB and DATCON and between DATCON and ONSEN. Working on the GTX transceiver level, all the mentioned characteristics during data transmission have to be handled in self-written logic and a user-defined protocol on application level has to be designed, which is described in Chapter 8.

6.2. xTCA System

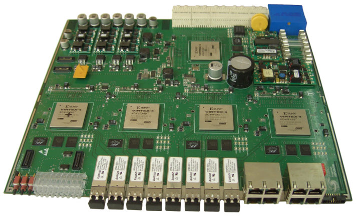
A carrier card to host the FPGA and provide the power and connectivity is required to operate the system. There are several establish standards offering such a system, but before, a list with requirements for the data reduction system is presented, which must be met by the standard:

- Support structure for optical transceivers (SFP cages), operating at least up to 3.25 Gbps at a wavelength of 850 nm. A minimum of four SFP slots are required to efficiently utilize the FPGA resources in the concentrator AMCs.
- A high-speed backplane connection to transmit the pre-processed data to the track reconstruction cards with at least 6.5 Gbps.
- FPGA with enough resources for the pre-processing code.
- Configuration and debugging capabilities over JTAG.
- A platform to manage power, startup and stop procedures and monitoring power participation, e.g. over the Intelligent Platform Management Interface (IPMI).

- A larger (1 GB or more) fast volatile memory in case a longer-term storage is required, e.g. DDR2 RAM.

A VME based solution was rejected, because the backplane bus cannot satisfy the bandwidth requirements. Since several projects within the Belle II collaboration use the ATCA standard and a wide set of systems are commercially available, it was decided to go for the same standard. Therefore a small overview of the main features of ATCA is provided here.
ATCA defines a modular standard which is widely used in backbone communication networks. Modules, containing the processing logic, with a wide spread of mounted chips for different applications e.g. CPU, Switches, or FPGAs are located in a standardized crate. The crate has a backplane with ports that are designed and standardized by the ATCA collaboration [48]. It provides power and slow control signals to manage the boards over I^2C and user defined signals. These are with 100 Ω terminated LVDS lines to provide a high-speed point-to-point communication between the modules. Each module is based on a 280 mm deep and 322 mm high PCB board, which can contain a great variety of chips. Figure 6.4a shows one of these boards used as a prototype structure for the DATCON and ONSEN system, carrying five FPGAs, one to manage communication and four for signal processing. A shelf that can carry up to 14 of these ATCA modules is shown in Figure 6.4b.

The design was later changed to the so-called mTCA standard, which provides several improvements to ATCA. Each of the front FPGAs is placed on a much smaller and more flexible PCB, called AMC, with a backplane connector linked to a “carrier board” with the same size as the ATCA boards. Now a broken circuit in one FPGA does not require a replacement of the whole unit. In addition, when the communication FPGA is not needed, the AMC can be operated stand-alone without the ATCA shelf and communicate directly over a smaller backplane using the same connectors as the carrier board. The layout of the 170 pin connector is di-


(a) ATCA module.

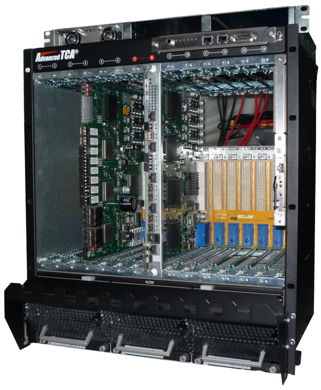

(b) ATCA shelf.

Figure 6.4.: Examples for an ATCA shelf and module.

vided in a similar manner as the ATCA backplane. There are signals to distribute the power, slow control for IPMI and LVDS clock lines. For high-speed communication 21 ports, so-called fat- or extended fat-pipes, are available, although most commercially available backplanes connect only the first 12 to 16.
To control both systems a so-called shelf manager is used, which controls power, fan speeds and performs other management tasks. It can also monitor the critical parameters and the “health status” of each AMC or carrier board. The shelf manager can be controlled over IPMI, usually they have an Transmission Control Porotocol/Internet Protocol (TCP/IP) interface to ease access.

Because certain synergies exists between ONSEN and the DATCON project, a joint-venture cooperation was founded to modify the AMCs originally designed for ONSEN to fit the needs of both systems. With a shared effort between Gießen (ONSEN), DATCON and the Institute of High Energy Physics (IHEP) Beijing (responsible for the production and design of the AMC), the layout of the AMCs were changed. A more detailed description of the features and testing of the prototype cards is provided in the next section.

6.3. Advanced Mezzanine Cards for DATCON

As seen in Chapter 4, the task of the DATCON system can be divided into two major independent subtasks: pre-processing of the SVD data and track reconstruction/ROI calculation. This is also reflected in the hardware design. Several smaller FPGAs with four SFP cages are used to receive and pre-process the data from the SVD FEE, while only two larger FPGAs reconstruct the tracks and calculate the ROIs. A first prototype of the new AMCs, called version 1 (AMCv1), is shown in Figure 6.5 with a Xilinx Virtex 5. It is equipped with two SFP cages and two Dual Inline Memory Module (DIMM) slots to put up to 4 GB of DDR2 memory. However, it does not fulfill the requirement of four SFP cages, to achieve a more cost efficient design. In a second iteration (called AMCv2) the connection density is increased by exchanging each SFP with a dual-SFP cage. Since this AMC is custom-made and was originally designed to work only with an also customized carrier board, the - for this project mandatory - high-speed backplane ports 6, 9, 12 and 18-20 are only connected to the FPGA.
In Figure 6.6 a version 3 AMC (AMCv3) is shown. Compared to AMCv2, AMCv3 contains only minor invisible fixes as described later. The major task of these AMCs is communication, so the optical transceiver as well as the backplane connection are carefully checked. Therefore, a test development crate as seen in Figure 6.7

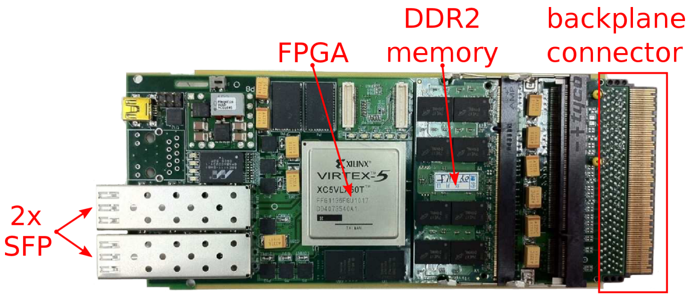


Figure 6.5.: Top view of the AMC with one Virtex 5 FPGA, two DDR2 memory modules with 4 GB and two SFP connectors.

with four full-size AMC slots is used. With a bit error test using a simple hashing algorithm and CRC to verify the data, the connection was tested. More details on the operation of the bit error tests can be found in [52]. All four SFP cages showed no bit errors at 2.54 Gbps over a period of 60 hours. In the development crate only 6 fat-pipes are connected on the PCB of the backplane, so only port 6 could be directly tested. With AMCv2 it showed 5 bit errors per second. Several equalization and pre-emphasis settings were tested to improve the signal quality, but it did not entirely solved the problem. Thus, a third revision of AMCs is required and carried out by the Beijing group to fix a routing problem to the backplane connector. Also the permanent storage for the firmware of the FPGA was to small, so it was exchanged with a larger model. Several other minor bug fixes went into this iteration as well. A new run of the bit error test confirmed that the problem is solved for fat-pipe port 6.
The standard Belle II DAQ system is operated with oscillators at a frequency of 127.21 MHz, but the special demands of the PXD system required the use of a 156.25 MHz clock for high-speed data transmission. All AMC versions have these oscillators

installed, but to fit the SVD requirements, this was exchanged with the 127.21 MHz oscillator. A second 156.25 MHz oscillator which provides the clock for the backplane RocketIOs is kept and used for internal communication as well as for the communication to ONSEN.

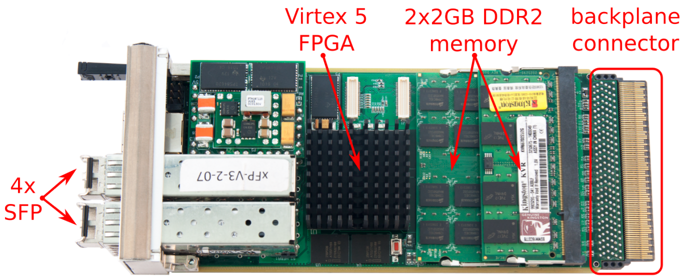


Figure 6.6.: Top view of the AMCv3 with one Virtex 5 FPGA, two DDR2 memory modules with 4 GB and four SFP cages.

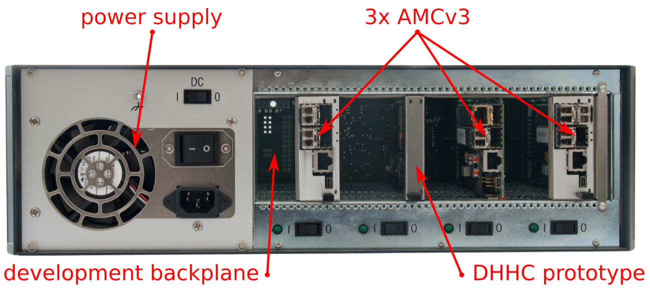


Figure 6.7.: Development mTCA crate with three AMCv3 and one DHHC prototype installed.

As shown later, the largest currently supported FPGA on the AMCv3 cannot host the firmware required for the track reconstruction code. An alternative, compatible solution was found in the AMCs used by the DHHC (introduced in Section 3.2.2). They are compliant with the mTCA standard and connect 16 backplane fat-pipes. A Xilinx Virtex 6 FPGA with roughly 8 times more resources is able to support the track reconstruction firmware for the full detector layout. A company is instructed to change the layout of the DHHC AMCs to add two SFP cages in the front for the ROI transmission to ONSEN. This card, however, lacks the DDR2 memory and Ethernet. A preliminary prototype of the DHHC that was tested in Bonn is depicted in Figure 6.8.

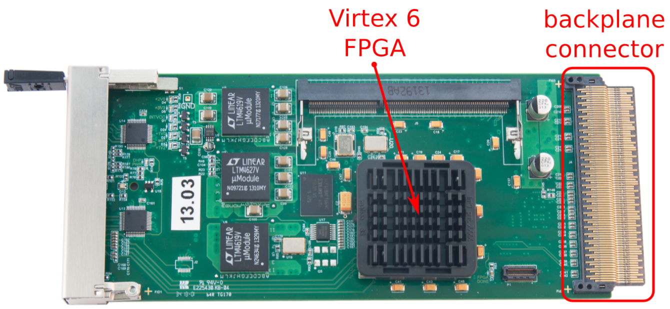


Figure 6.8.: Prototype of the DHHC AMC with a Virtex 6 FPGA and 16 connected backplane ports.

6.3.1. Backplane Design

In order to accumulate the pre-processed data in two track reconstruction AMCs, a custom backplane must be designed. Figure 6.9 shows a proposal of the connection topology. The system is hosted in two 9-slot mTCA crates, one for each implantation side in the SVD. Drawn in blue is the track reconstruction slot and in green the proposed backplane connection. A mapping must be done to route all AMCv3 ports 6 to three of the free DHHC AMC ports. This configuration is a called ”star-backplane”, which will be produced by the same company providing the crates. The remaining slots will serve as backup in case more AMCs are required. This can happen when the occupancy of the SVD sensors in the inner layer turns out to be higher than expected. In that case two additional FADCs might be added, with two more FTBs and optical connections.

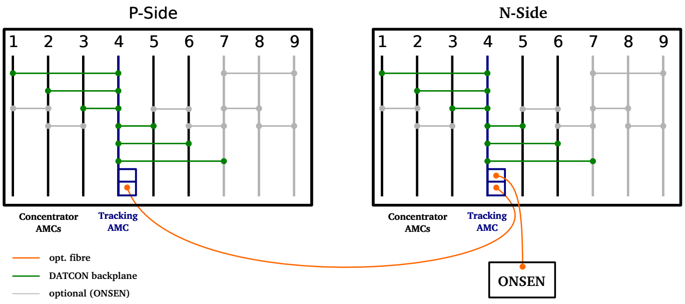


Figure 6.9.: Backplane topology proposed for DATCON indicated by the green lines, while they grey lines represent the Daisy-chain for ONSEN.

FPGA Implementation

The implementation of the track reconstruction code presented in Section 5.3.1 is handled differently in the FPGA to exploit its characteristics to gain a performance advantage. The necessary modifications of the algorithm are discussed in this chapter. Also the difference between the C++ simulation and the FPGA code is explained. C++ executes the code in a sequential order on one processor core, one instruction after the other. In contrast to FPGAs where many logic units and DSPs can run in parallel. This is also reflected in the programming language of Verilog, where several paradigms exists to ease the use of parallel programming and to express the desired kind of logic (e.g. a flip-flop or latch). This chapter focuses on the detector independent part[1], absolute coordinates of the detector hits are taken as input to the track reconstruction module. Two different tracking modules were created in the scope of this thesis. The first was very similar to the Iterative Fast Hough Transformation described in Chapter 5, but it consumed a lot of resources to map the iterative process, which was realized in the FPGA over a Finite State Machine (FSM)[2]. It was used in the first test beam campaign (see Chapter 9). A different approach was later implemented to solve the resource issue and meet the required trigger rate of Belle II. The new track reconstruction module also made changes required to the MPH and ROI creation code path. Moreover, the transmission protocol

[1]The absolute coordinates of the detector hit IDs are already calculated (which depends on the detector type, see Chapter 8 for details on the SVD pre-processing).

[2]An automaton with a finite number of states, where only one state is allowed at the same time. Transitions between the states are controlled by logic conditions. [42]

for the ROIs are shown and the test environment to compare the performance with the C++ simulation.

A further obstacle in the programming of FPGAs in Verilog is the lack of support for floating point[3] numbers in Verilog and of a Floating Point Unit (FPU) in the FPGA as it is usually available in every modern CPU. Thus, another representation is selected for the calculated absolute coordinate hits. A natural and easy way, also supported by Verilog, is to use signed fixed point numbers. One coordinate is represented by a 32 bit signed integer number (in two's complement [39]), where this number is multiplied by 1024 (2^{10}, or in other words shifted by 10) and the decimal number is taken. This helps later calculation, since shifts are simple logic operations. One drawback of the fix point calculation are larger rounding errors during calculations. Floating points have much smaller errors since more digits are involved and a consequent rounding model is used. A representation as a simple "float" is also 32 bit wide, so in principle possible, but even simple calculations like adding or multiplication requires complex logic according to the Institute of Electrical and Electronics Engineers (IEEE) 754 standard. [40] Since as many sectors as possible should be checked at the same time, the calculation should be kept simple to achieve a higher module density.

7.1. Fast Hough Transformation

The 48 bit input format of the track reconstruction module is depicted in Figure 7.1, where the first 32 bits are the already mentioned decimal coordinate number and the last 16 bits con-

[3] A floating point number is a binary representation or approximation of a real number which is written in the form of $-1^{sign} \cdot significant \cdot 10^{exponent}$. [41]

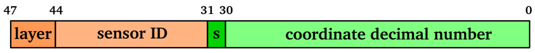


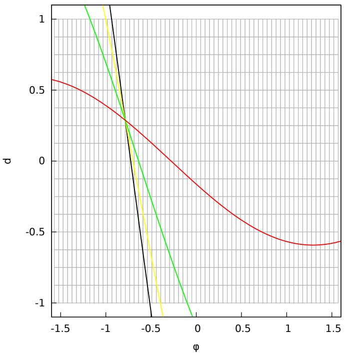
Figure 7.1.: Coordinate format for the track reconstruction module. The 3 bit wide layer is denoted to the layer id, followed by an 13 bit wide unique sensor ID for each layer, a sign bit s and the rounded to decimal coordinate number.

tain the sensor ID, which is divided in two parts. The last 3 bits enumerate the layer and the first 13 bits are an identification of the sensor in each layer. Only the first two bits of the layer identification is used, because the ROIs are calculated from SVD data. On n-side the absolute z coordinate and on p-side the y coordinate is transmitted. With the sensor ID the x coordinate can be reconstructed for the barrel sensors, while the x coordinate of the wedged sensors must be transmitted as well. This saves a lot of bandwidth in coordinate transmission and simplifies the transmission protocol as later shown in the test beam campaigns.

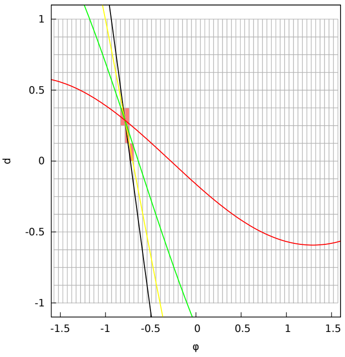
The incoming coordinates are transformed in the HS in a different way as in the simulation. Each coordinate, respectively line in the HS is simultaneously checked for crossing a sector as described in Section 5.3.1. Therefore the space is divided into the smallest sectors corresponding to the highest allowed iteration, illustrated in Figure 7.2a. Since the sector check provides the greatest quantity of modules and determines the speed, its resource consumption should be kept as low as possible to fit the design in the FPGA.

The (x, y) respectively (x, z) coordinates of the detector hits are transformed according to Equation 4.3. Xilinx provides a core generator[4] for trigonometric functions, which were also used in the first implementation to calculate sine and cosine values. However, calculating the values for a 32 bit decimal number takes 36 clock

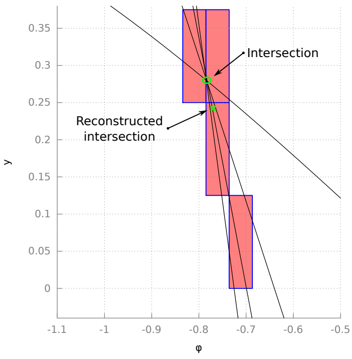
[4]A core generator enables the user to generate complex logic structures in an easy way for several thousand different applications, e.g. Aurora or GTX code.



(a) Pre-sectorized HS (grey grid) with four detector hits.



(b) Active sectors marked as red. Four sectors are found.



(c) Zoom of (b) with marked active sectors. The real and the reconstructed intersection with cluster-based method are depicted.

Figure 7.2.: Illustration of the Hough track reconstruction algorithm implemented on the FPGA for the different steps of processing. The color of the lines represents Layer 3 (black), Layer 4 (yellow), Layer 5 (green) and Layer 6 (red).

cycles and consumes two times 3200 FF and 1151 LUTs [44] of the FPGA's resources in a pipelined design for every value. Too much to fit the required 256 $\times$ 128 sectors in one FPGA. Since not all of the values are needed, a different approach is used to exploit the fixed sectors' location. A *C* program creates for a given input parameter set (the size of the HS and their range) a custom LUT which contains only the values required by the sectors in question. Every column of sectors shares the same values. For the multiplication the DSPs of the FPGA are used. To determine if a sector is crossed by a line the same condition as described in Section 5.3.1 is implemented. A simple light-weight comparator logic fulfills the task. In addition, each sector check module is equipped with 4 registers to realize the layer filter. The sensor information, coming with the transformed coordinate, is used to mask from which layer the sector was crossed. A customizable majority logic marks the sector as active when it was crossed by hits from three or four different layers. The output of the majority function is used to tell the superior module if the sector needs to be evaluated in the further processing chain. Figure 7.3 shows a simplified data flow diagram of the whole track reconstruction and ROI creation unit. In the next step, the active sectors (see Figure 7.2b) are clustered. Simultaneously every sector is evaluated separated in vertical and horizontal coordinates. Thereby the same next-neighbouring cluster algorithm is used to cluster the SVD hits. Instead of a center-of-gravity calculation, a simple averaging function with a weighting of the cluster size is used as shown in Figure 7.2c to create track candidates. Afterwards two track candidates are merged together to a single (3D) track.
Since every sector corresponds to a single track parameter description, which is extrapolated to a MPH, the extrapolation can be pre-calculated and stored in a LUT to save the complex computation shown in Equation 4.8. A further improvement in resource consumption can be achieved when directly translating the output of the clustering to a pixel row and column ID. Several neighbouring sectors produce the same pixel hit on the PXD. This method

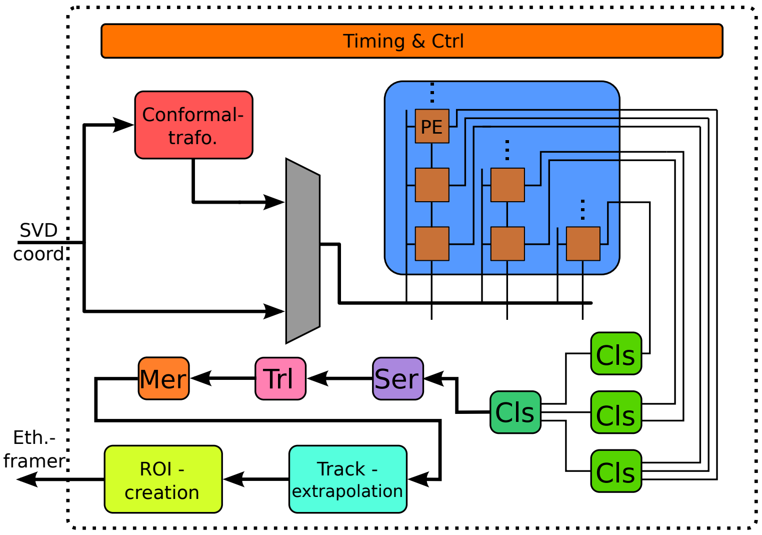


Figure 7.3.: Data flow diagram of the track reconstruction and ROI calculation unit. Each Processing Element (PE) checks one sector for traversing lines. Active sectors are clustered using a two-step next-neighbour-search clustering modules (Cls). The parallel data stream is serialized (Ser), translated (Tls) back into coordinates and merged into helix track parameter sets. The MPH is extrapolated from the track parameters and the ROI is created.

needs further investigations, but seems to be a promising approach to save resources and speed up the calculation.
The coordinates originated from arc tracks, coming from the r-φ-plane needs a conformal transformation, before they can be used with the HT. Over a multiplexer in the track reconstruction module, it can be selected. Equation 4.5 describes the transformation formula used to transform the coordinates. With the core generator of Xilinx a pipelined divider unit is created to realize the division. This adds an additional delay of 36 clock cycles to the execution time.

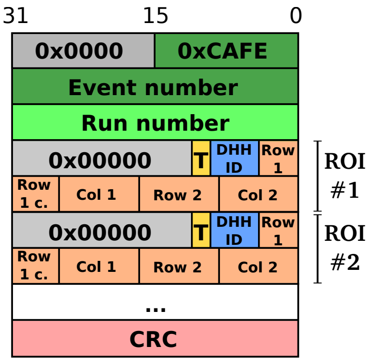
7.2. Regions of Interest

In the traditional track reconstruction module, the helix parameters are extrapolated to MPH and then translated to pixel column and row IDs. Therefore, the same method is used as in the simulation described in Section 4.3. The coordinates of the MPH are transformed to local coordinates on the sensor and subsequently divided by the pixel pitch. A division in the FPGA is very resource consuming, so it is postponed to the latest possible step, where the incoming local MPH coordinates are already serialized. The division unit is created with the Xilinx core generator in a pipelined design to speed up the operation, since a single division takes 36 clock cycles. The result of the division module directly provides a column or row ID depending on the detector side. The sensor ID is then translated into a DHH ID, which is used in the ONSEN system to match the PXD data with the ROIs. It is mandatory for the protocol as later described in Section 7.2.1.
The column and row IDs are calculated separately and are later merged based on a pairwise sensor/DHH ID comparison. In the FPGA test system the calculation and merging process is done on the same FPGA, because of lack of required hardware. The combined row and column IDs are sent to the ROI creation mod-

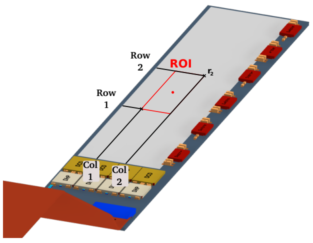
ule. With a combined row and column ID set, a 3 bit quality indicator is sent, which defines the ROI size over a programmable LUT. Thereby the boundaries of the sensor need to be respected to avoid an overflow of the registers. The details about the format and how the ROIs are transmitted to the ONSEN system is explained in the next section.

7.2.1. ROI Format and Transmission Protocol to ONSEN

In order to successfully perform the pixel selection, the ONSEN system needs to know the exact time to match ROIs with pixel data, which is identified by a globally valid event number, and the PXD sensor ID encoded in the DHH ID. The event number for each particle collision is distributed over the Trigger Timing Distribution (TTD) and encapsulated in the SVD data. During pre-processing and track reconstruction the event ID needs to be preserved for a later assignment to the ROI data. This is handled by the so-called event management unit, explained later in Section 8.3.
Once the ROIs are calculated they are formatted to be sent over one Aurora link to the ONSEN system. The protocol is depicted in Figure 7.4a. A magic word marks the start of the frame, followed by the event and run number. One ROI is described by a 64 wide data word, containing a source information T (for DATCON it is 1, for HLT 0), the DHH ID, and the position given in row (10 bit: 0...767) and column (8 bit: 0...191) IDs of the lower left and upper right corner of the ROI. The packet can also contain no ROIs, in fact it is supposed to send empty ROI packets when no tracks are found for consistency reasons and to indicate that the DATCON is still alive and finished processing the event. The integrity of the sent data packet is ensured by a CRC. To save resources the GTX's internal CRC module is used, interfaced



(a) ROI format.



(b) Description of one ROI.

Figure 7.4.: ROI format with one example.

with the Xilinx core generator.

Transmitting the pixel IDs of two edges of the ROI also means that during the calculation process the alignment constants for both detector systems needs to be stored and updated when necessary. An offset register for every sensor is foreseen to hold the alignment information. These registers are designed to be updated over a slow control interface, e.g. Ethernet.

7.3. Test Environment

The optimization process and synthesis of the FPGA firmware, with the track reconstruction module taking more than 90 percent of the resources, takes over 6 hours to complete. This is not feasible for an efficient debugging of the system and to monitor up to several thousands of internal signals simultaneously in the system. A general concept in FPGA programming includes a logic and time behaviour check and verification of the HDL code in a simulation environment, before the implementation and hardware tests. In the next section the software to build the simulation and the setup is described and some first results are given.

7.3.1. HDL Simulation

Questa's Advanced Simulator [49] (also-called Questasim) is used to set up a simulation environment for the track reconstruction HDL code. For a realistic model of the system, the complete pre-processing chain and all top modules of the FPGAs are included. A special, so-called testbench file is required to steer the input pins of the FPGA or emulate the hardware mounted on the PCB. For instance the different clocks are usually provided by an external source (oscillator). The clocks and a simple global reset for the FPGA are implemented in the testbench.
In addition, the complete data chain from the SVD needs to be emulated to provide the coordinates for the track reconstruction module. A *C* program is used to convert the Digits from BASF2 into the hardware data format and creates a LUT for the testbench. The block diagram of the testbench and the analysis module of the data is shown in Figure 7.5. A verification and debugging module is included which can write the complete HS into a file for plotting. This module is later partly reused in a test beam campaign for hardware debugging. An example waveform plot from the simulation recorded with the testbench including the most important signals, although simplified, is shown in Figure 7.6. The different signals are vertically arranged, while the time line goes in horizontal direction. Each event can be divided into three sub-tasks:

1. **DAQ**: The data is sent from two FTBs, one for each implementation side and are being processed by the concentrator

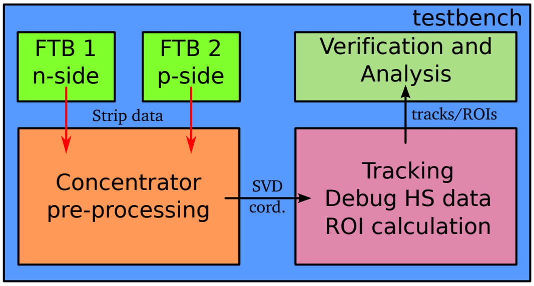


Figure 7.5.: Testbench block diagram to verify the full pre-processing and track reconstruction code in an HDL simulation.

AMC(s). More details on the implementation of the pre-processing are shown in Chapter 8.

2. **Communication:** Once the coordinates are calculated they are sent over the backplane to the track reconstruction cards.

3. **Track Reconstruction:** The track reconstruction and the serialization of the HS sectors takes most of the time, which is only needed for the analysis module. The HS is later clustered and merged.

Only a few thousand detector events can be simulated, since one event can take several minutes to simulate, depending on the number of hits. For a long-term stability check a hardware test is mandatory.

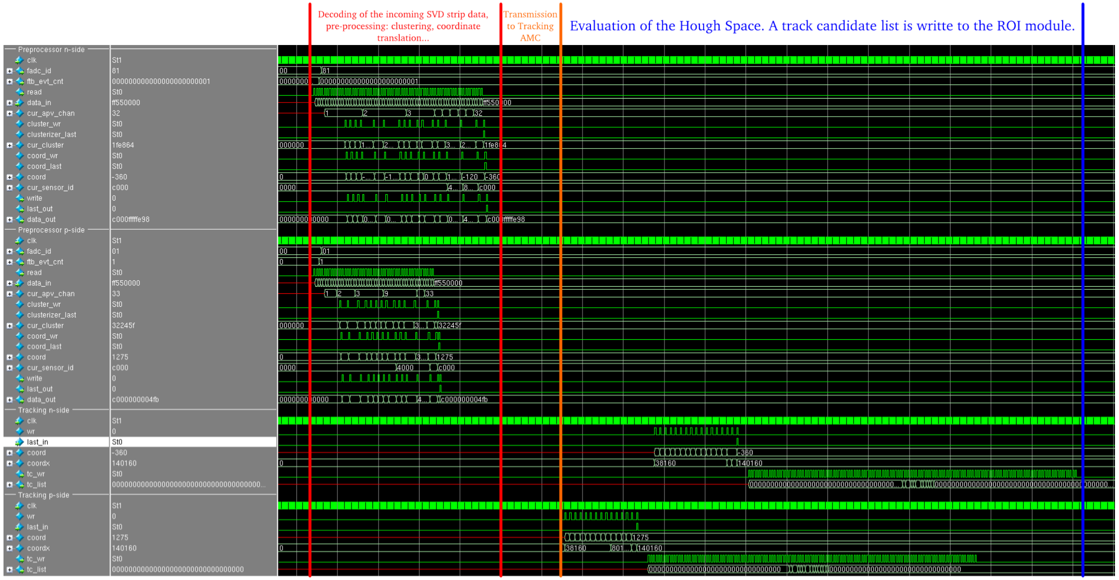


Figure 7.6.: An example of a waveform plot showing the ratio between the pre-processing steps (first quarter), the transmission (second quarter), HS evaluation of the detector coordinates (third quarter) and miscellaneous tasks (fourth quarter) with a selection of the most important involved signals.

The time required to search the HS in the FPGA implementation code scales linearly with the number of detector hits c_n. Equation 7.1 describes the time t required to search the HS, verified in simulation and later in a hardware measurement:

$$t = (s_{HS} + 4 + c_n) \cdot \frac{1}{f_c} \tag{7.1}$$

The processing frequency of the tracking module, f_c, is equal to the standard Belle II 127.21 MHz oscillator. However, a higher frequency is possible to further speed up the calculation. Four clock cycles are required for the clustering and the number of HS sectors, s_{HS}, in the φ-axis for the serialization. With a division of 256 sectors, a HS search with 1000 detector hits takes 10 μs, compared to 120 ms on the computer (see Section 5.3.5). Enough FPGA resources assumed to hold the complete HS. A solution for FPGAs with lower resources is described in the next section.
Another difference that needs to be evaluated between the C++ simulation and the FPGA implementation is the representation of numbers for coordinates and angles. In C++ floating point numbers are used, which gives a much higher precision than the fixed point integers in the FPGA implementation.
To determine the difference between the two representations and to study the influence on the results, the HS between the generated events in BASF2 and the C++ simulation of the track reconstruction code and the FPGA implementation are compared. During this comparison the intermediate data after each calculation step are evaluated. Most rounding errors originate from the pre-calculated rounded sine and cosine LUT values and during calculation. It accumulates to a worst case error of ± 5 μm for the coordinate position and 0.3° in angle resolution. The error can also be calculated with Gaussian propagation of uncertainty. Considering all required calculation steps in the FPGA from the Hough track reconstruction to the ROI creation, the difference between the ROIs is only one pixel in a rare worst-case scenario. Since a large error margin is already assumed in the ROI calculation process no increase of the size is required.

To verify the working conditions after changes to the track reconstruction module, a simple regression test was written, always using the same coordinate set. It is executed after every major change to the system to ensure that no new flaws, or miscalculation was introduced into the system.

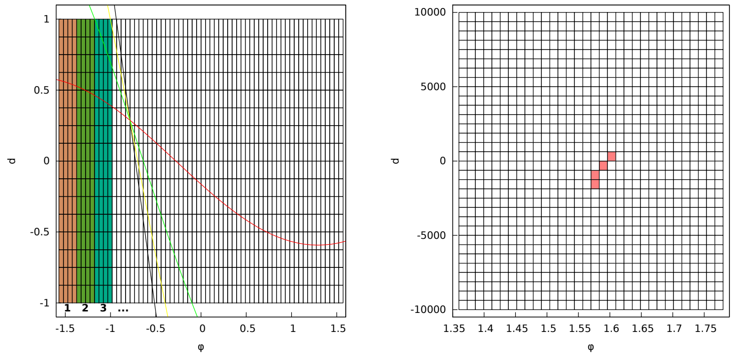
7.3.2. Hardware Implementation

For the hardware implementation, the HS track reconstruction algorithm needs to be modified to meet the available hardware resources, since the final revision of the DATCON AMCs was not available at that time. A version with reduced resource consumption of the track reconstruction algorithm is also useful to fit the module into low budget FPGAs. The *C* program that creates the Verilog code was adapted with several new input parameters to define the so-called "fragmentation" of the HS. Figure 7.7a illustrates the HS search in the modified code with a fragmentation size of four. Only four HS columns are evaluated at the same time. This increases the run time by:

$$t = \frac{s_{HS}}{c_f} \cdot ((\frac{s_{HS}}{c_f} + c_n) + 4) \cdot \frac{1}{f_c} \ , \tag{7.2}$$

where c_f is the fragmentation size, but reduces the resource consumption by a factor of s_{HS}/c_f. Additional BRAM is required to temporarily store the coordinates for one event, because in every fragmentation step, the coordinates need to be transformed and checked again. With this method, the track reconstruction code fits in the AMCv3 boards, as later shown in Chapter 9. Moreover, the synthesis time of the firmware is significantly reduced from over 6 hours down to 45 minutes, depending on the fragmentation level. This also helps the debugging process since testing of an updated firmware is much faster.

For a first hardware test and because of the lack of the final



(a) Fragmented HS search. (b) HS obtained with test system.

Figure 7.7.: Illustration of the fragmented HS search algorithm, only searches in a given time a specified number of sectors. Then goes to the next fragment.

hardware the ML505 evaluation board equipped with a Virtex 5 FPGA is used. The evaluation board is depicted in Figure 7.8 and provides a Peripheral Component Interconnect express (PCIe) interface to transmit coordinate data from the PC to the evaluation board and receives the results back. In addition, the RJ45 and the connected Ethernet chip are later used to build an ONSEN emulator to send the ROIs to a PC. The multi-pin connector was used to connect a mixed-signal oscilloscope which can show digital as well as analog signals. This is a convenient tool to check the time behaviour of the system.

To determine the resource consumption, the track reconstruction firmware was compiled several times with different fragmentation settings. Since event management, communication and other modules are still included in the code (but unchanged for every synthesis run), the difference between each run gives an indication of the resource consumption. Table 7.1 shows the results with the

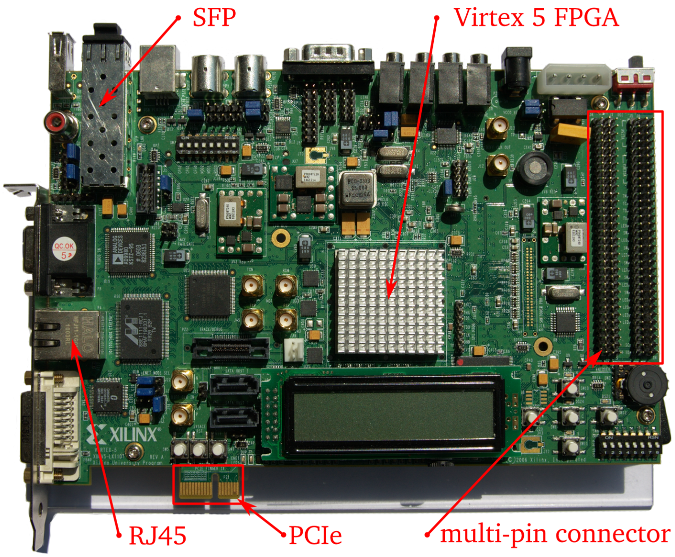


Figure 7.8.: Evaluation Board ML505 with a Xilinx Virtex 5 FPGA, an RJ45 connection for Ethernet, a PCIe connector to interface a PC and one SFP cage.

important resource properties of the FPGA. The utilization table distinguishes between slice registers, LUTs and logic LUTs, where logic LUTs are the most important indicator, since the comparators for each sector consumes a lot of logic resources. As there is only one FPGA available the two track reconstruction units for the two different implantation types of the SVD are included and the size of the HS was limited in this case to the test beam acceptance area as later described in Chapter 9. The last column of the table shows an optimized version of the same Verilog code with several tuned settings for a more aggressive optimization during synthesis in the Xilinx ISE. The saving of 8 percent in resources comes with an increased synthesis time. In case of shortage in lo-

Fragmentation size	HS size	Used registers	LUTs	Logic LUTs
2	$2 \times 32 \times 32$	5990	8321	7784
4	$2 \times 32 \times 32$	6771	12539	11415
8	$2 \times 32 \times 32$	8380	28249	27645
8 (opt)	$2 \times 32 \times 32$	8426	26473	25998
	$\Delta 8 - 8$ opt	+46	-1776	-1647

Table 7.1.: Resource consumption (separated by number of required LUTs, logic LUTs and registers) for different fragmentation sizes of the HS.

gic resources the remaining BRAM can be also used as logic cells.

Many different analysis and debugging modules were developed during the scope of this thesis to compare the HS with the C++ simulation and performance analysis and optimization of the Verilog code. One of the debugging modules allows one to transmit the status of the whole HS over one optical link to a second board. The second board can either be another AMC or the aforementioned evaluation board. A simple example HS obtained with this chain is depicted in Figure 7.7b, where the intersection is clearly visible as active (red) sectors and is identical to the one obtained with the C++ simulation describing the generated track with a precision in φ of less than $0.15°$.

8

Data Acquisition and Preprocessing

Before the track reconstruction can be performed, the strip data of the SVD need to be processed and transmitted to the track reconstruction hardware. The first step is to acquire the data within the FTB, adding the event information (event and run number) from the FTSW, embedding these values in a data structure and implementing a protocol that ensures the correct transmission between FTB and the concentrator AMCs of the DATCON. As seen in Section 3.3.3, the FADC can provide the data in four different modes. At least two of these SVD modes are used during the run of the experiment and in test beam campaigns and must be decoded by the system. Afterwards, the pre-processing tasks, as described in the simulation Chapter 5, are executed. This includes the clustering of the strip data and their translation into absolute coordinates.

During the first test beam campaign a (noise) filter was required to determine if the signal of the strip corresponds to a particle hit or is just a single peak over zero-suppression threshold. All these steps can run simultaneously and independently on the 12 concentrator FPGAs. The pre-processing steps depend on the detector technology, which can greatly differ in its format of the incoming data and the information they can provide.

In addition to the pre-processing tasks, the data management will be discussed in this chapter. Since all the data must later be assigned a unique combination of event and run number, these data need to be preserved throughout all pre-processing steps. This is crucial for the later matching of ROIs with PXD event

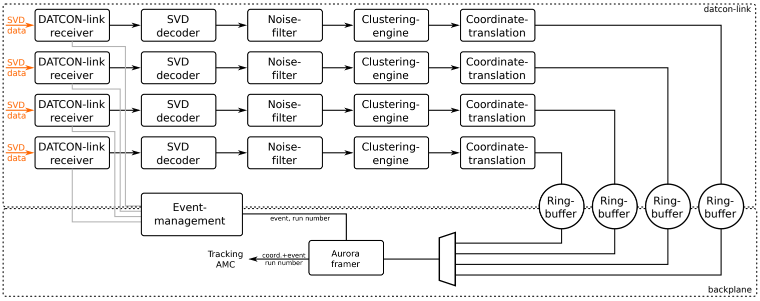


Figure 8.1.: Functional block diagram of the concentrator AMC. The dotted lines represent the different clock domains of the system, while the orange lines are the four optical links coming from four FTBs.

data. Because of the different time dependent requirements of all the pre-processing stages, this part takes a major portion of time during testing and development. All cases of discarded events in each pre-processing step and the correct propagation of a skipped event throughout the whole DATCON system must be handled in the event management.
For communication between the concentrator AMCs and two track reconstruction AMCs, an internal protocol for communication on the high-speed backplane is developed. This protocol needs to transmit control and alignment information as well as the co-ordinate data of the system.

All major components involved in the DAQ chain that are related to the DATCON system are discussed in the next sections. Figure 8.1 shows the pre-processing steps of the concentrator AMCs, which are explained step-by-step in the following sections. The FTB firmware is described first, even though not shown in the figure.

8.1. FTB Firmware

The firmware development for the FTB is a collaboration between the hardware developer of the FTB, the head developers of the COPPER and the DATCON system. Incoming data from the FADC are checked for compliance with the protocol shown in Figure 3.8 in Section 3.3.4. The most important checks include the validation of the CRC and the presence of the protocol header and trailer. Afterwards the trigger information (event and run number) obtained from the FTSW connection is added to the FTB protocol header. Flags about the state of the received data stream and errors (wrong CRC number, oversized event, header detection error) are included in the FTB protocol. At the end a new CRC over the whole recently constructed frame is calculated and appended. The processed data stream is written into a sequence of FIFOs. Figure 8.2 shows the last processing stages and the interface to the transmission modules. The DATCON-related FIFO is marked in red and provides a 34 bit wide data line, which includes two wires to indicate the start and end of event. In addition, an event ready signal is provided and asserted every time a complete event was written to the FIFO. Interface signals of the FIFO, e.g. read, full, or empty indicates the FIFO status or perform operations.

The FIFO is read by the so-called DATCON-link framer, which is responsible for transmitting the data to the concentrator AMCs. A layout of the protocol developed and used to frame the FTB data is depicted in Figure 8.3. For consistency, an internal event number is integrated here as cross-reference value. An appended CRC ensures the integrity of the data and marks the end of frame. On link layer the Aurora protocol, as described in Section 6.1.1, is utilized with a bandwidth configurable of up to 2.54 Gbps, although for the test and prototype implementation only 1.27 Gbps

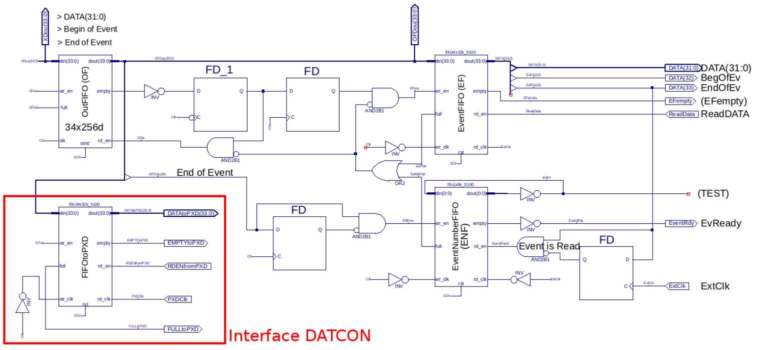


Figure 8.2.: Interface schematics of the FTB. Marked in red is the DATCON-related FIFO and the available pins.

was used, fulfilling all the bandwidth requirements of the tests.[1]

The connection between the FTB and the concentrator AMCs were tested with a long-term bit error test. A 60 hours long run as well as several hardware tests and test beam campaigns showed no errors.

8.2. Concentrator Firmware

The detector dependent processing is performed in the concentrator AMCs. 12 concentrator AMCs will be required to translate the strip IDs to absolute coordinates, so that they can be used by the track reconstruction AMCs. The data is acquired and processed on four optical links in parallel on each card. The processing module, one for each link, is designed as a fully pipelined processing chain. The four major stages of this module are dis-

[1] In the test beam campaigns only 24, or 18 APV25 chips are read per FADC board, respectively at a lower rate than in SuperKEKB (see Chapter 9).

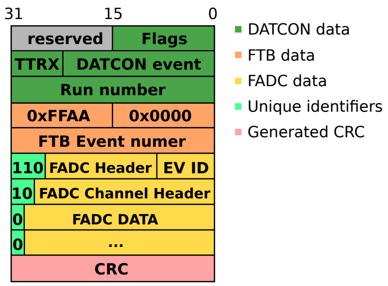


Figure 8.3.: Format description of the DATCON-link protocol. “TTRX” is the FTB board number, “Flags” contains certain status information about the system. The FTB protocol (see Section 3.8) is encapsulated in the DATCON-link.

cussed in this section. Most were developed for the test beam campaign and do not yet support all the features required for the final detector, since no detailed specifications have been released and no reference implementation is available.

8.2.1. SVD Decoder

The extracted FTB packets are decoded over an FSM. So far only two of the four modes are implemented in the DATCON system: zero-suppressed with six or three samples or hit-time mode. The remaining modes are for debugging and calibration purposes and are not used during data taking of the experiment, since the data rate would be too high to be managed by the COPPER system at the full 30 kHz trigger rate. So far only the six sample modes are implemented in the FADC. In the first state, the FSM decodes the protocol header of the FTB and FADC, containing the event information (run and event number), the unique ID and the

run mode of the FADC. This information is propagated to the event management unit. Afterwards the APV header (specifically important here is the APV channel number), strip IDs and samples, or the peak sample are extracted and the corresponding processing pipeline is started to process the event, depending on the run mode. A more detailed description of the FADC protocol and the two used run modes are described in Appendix A.
In zero-suppression mode with six samples (the only implemented in the FADC during the scope of this thesis) the noise filter is required to reduce off-time trigger and noise hits (see next section). Therefore the noise filter is optionally included in the decoder, since it can run independently on each strips' samples.

8.2.2. Noise Filter

During the first test beam campaign some sensors showed an extreme erratic noise behaviour. An increase of the zero-suppression threshold could not entirely solve the problem, so many noisy strips were above the threshold. To suppress the effect a filter in the sensors is required to reduce the occupancy. The filter must be able to distinguish between a particle hit and noise by analysing the shape of the samples. Figure 8.4a shows an example of a recorded charged particle hit (during the DESY test beam campaign) in one strip in six-sample mode. The shape of this waveform can be approximated with the rise and fall of CRRC shaper:

$$f(t) = A\frac{t}{\tau}e^{-} \tag{8.1}$$

where τ is the time constant of the shaper, proportional to the load capacitance of the sensor. With the detector connected to the APV25, the shaping time is around the designed 50 ns of the chip. A fit of the waveform with cubic splines or an exponential fit results in a precise peak time that can be determined with a

resolution of down to 3 ns and an accurate peak value. [54] With this information offtime trigger hits can be removed, where the peak is not in an acceptable time window. In addition, noisy strips usually have a completely different shape with single spiking samples. The fit of these shapes will fail and the firing strip is completely rejected. The size of the peak caused by a particle track is also within a certain limit and not higher than 180 ADUs. However, an implementation of a cubic spline or exponential fit in the FPGA would be beyond the scope of this thesis and the resources of the FPGA. A basic approach with a heuristic detection of a valid waveform signal with a limited offtime trigger filter is defined.
The decoding of one strip FADC signal in six-sample mode is distributed into two data words, thus two clock cycles are available for decoding. In each clock cycle the data of three samples are processed. A combinatorial net searches for the maximum peak value within these samples and saves its position and peak sample in a register. In the second clock cycle the other three are searched and compared with the peak of the first samples. With another combinatorial net the neighbouring sample sizes are extracted and provided to the noise filter module. The value of the peak sample is shifted by a to be specified number of bits (in other words divided by multiples of 2). A standard value is one (a division by two) and the values of the left and right neighbouring samples must be above this value. Meanwhile the size of the peak is compared to a configurable minimum and maximum allowed value and the position of the peak is evaluated. During measurements of the latest two test beam campaigns the peak must be located at the second, third or fourth sample with a signal between 4 and 180 ADUs to accept the strip. The procedure is depicted in Figure 8.4.

With this scheme a reduction of noisy strips by up to a factor of 10 can be achieved. An example of this reduction with a simple strip hit map recorded in the DESY test beam 2014 is shown in Figure 8.5. The noisy APV25 is almost completely suppressed,

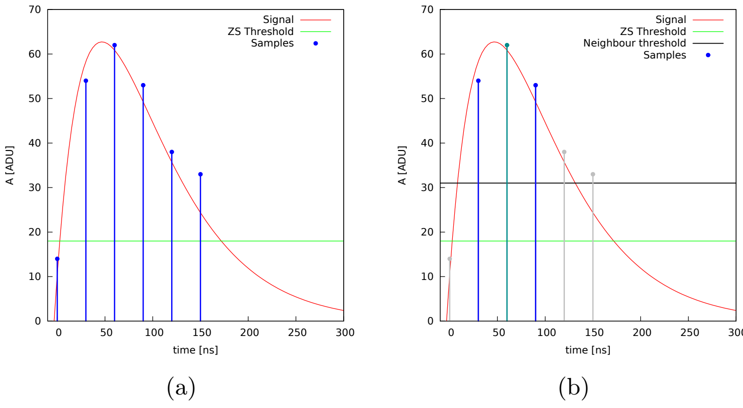
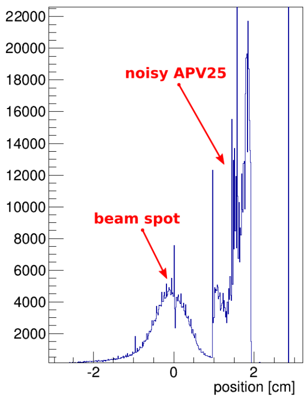


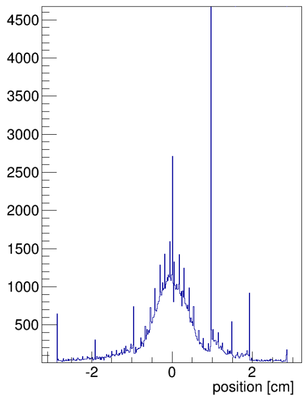
Figure 8.4.: (a) Six APV25 samples from the DESY test beam campaign, showing the fitted waveform of a particle track. (b) shows the found peak sample and the left and right neighbouring samples that have to be over the neighbouring peak threshold, depending on the peak sample.

while the beam spot is still clearly visible. Some events are removed with an overlap between noise and signal, or when the peak amplitude is below the minimum threshold. This can happen at the outer edges of clusters, when the peak signal is just above the zero-suppression threshold. An improvement of this behaviour would be to also consider the use of the neighbouring strips over threshold without the noise filter and always accepting the strip signal. However, the resource requirements would be higher and the implementation more complex.
The noise filter is not required for the final Belle II detector since the standard design will include a waveform fitting in the FADC, which was not developed during the test beam, while the sensors show excessive noise. The implemented noise filter was designed

to improve the signal-to-noise ratio. As shown in the simulation, events with high background must be skipped to avoid too many fake tracks.



(a) Without noise filter



(b) With noise filter.

Figure 8.5.: Strip data from the n-implantation side of one sensor during the DESY test beam campaign without (a) and with (b) the noise filter.

8.2.3. Cluster Engine

Once the strip data shows a valid waveform signal and pass the noise filter they are processed by the cluster engine. In the first step with the incoming APV (0...47) and channel ID (0...127), the sensor ID is reconstructed and then the full strip number (0...767). A programmable LUT contains the mapping of APV, channel and sensor IDs.
Afterwards, the strips with their peak sample are grouped into

clusters using the same next-neighbour search algorithm as described in Section 5.2.4. The implementation differs from the C++ code to take advantage of the nature of FPGAs, which is realized over a pipelined process controlled by an FSM. The 8-value center-of-gravity LUT provides a correction constant for the later coordinate translation with the last strip ID and the size of the cluster. This information is later provided to the coordinate translation. Clusters larger than 4 samples are split up, without taking care of their shapes that reduces the position resolution in pileup events. However, in a usual test beam event, not more than one track is present.
A regression test containing a list of different cluster shapes was used to test the logic of the FSM and verify that the cluster engine is able to handle all different strip combinations in a correct way. Wrongly reconstructed clusters may result in lost tracks.

8.2.4. Coordinate Translation

In a programmable LUT the absolute coordinate of the first strip in each sensor is stored. The strip pitch is stored in fixed registers with identification of layer and implantation side of the sensors. For future use the linear function to calculate the coordinates on slanted parts and the windmill structure angles are needed to be stored as well. They can all be pre-calculated in the PC and the results are stored in another LUT.
With the strip pitch, the size and seed strip ID, the hit position can be reconstructed as seen in Equation 5.3. With the added correction value the position resolution is increased. The coordinates are stored in 32 bit wide fixed point numbers, that can be directly used by the track reconstruction unit. Before the track reconstruction, the coordinates are stored in a ring buffer until the track reconstruction AMC requests the next set of coordinates, then they are sent over an internal protocol. The management

and aggregation of all coordinates in the track reconstruction card are done in the event management modules describe in the next section.
As an improvement and to save resources in the track reconstruction AMC it was considered to put the conformal mapping of the p-side strips already here. On the other hand, this also means that the x coordinate of each detector hit has to be transmitted as well.

8.3. Event Management

The event management modules are present on both the concentrator and track reconstruction AMCs. They handle the internal communication and ensure that all relevant information such as event and run numbers are synchronously transmitted with the corresponding coordinates.
Coordinates obtained from the coordinate translation are stored in a ring buffer, implemented with a true dual-ported Block memory of the FPGA. In another memory the start position of every event in the ring buffer with the event size is stored. The event ID, which is provided by the SVD decoder is loaded into a hash table, which supports up to 8 entries, implemented as registers. The event ID is connected with the address of the memory in the hash table to store the data start address of the ring buffer and its size. A running address pointer always points to the first entry of the hash table providing all information required for the further processing steps. Once an event is processed and transmitted, the hash table pointer is incremented. The hash table and the address memory are cyclically overwritten like a ring buffer. Figure 8.6 shows an example of the different memories and tables of one event.
The hash table with the event ID can in principle be larger than the memory for the start address and sizes of the ring buffer. This

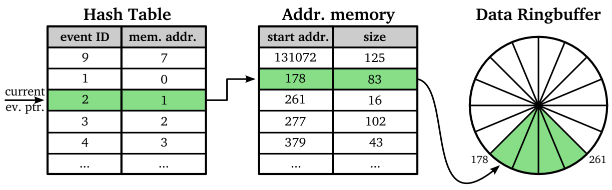


Figure 8.6.: Event management memory illustration. Highlighted in green are the information and coordinate data in the ring buffer of the current event.

way more event metadata are able to be stored than events can actually be processed by the system. Skipped events are detected when the detector bandwidth is too high and events are about to be lost. For the final Belle II experiment an indicator is required to monitor when an event does not fit into the ring buffer. With the current implementation this is not possible since a complete event of one FADC in test beam design with four sensors fit into the memory. Future studies of the occupancy need to be done to check if the memory of the FPGA is large enough or the 4 GB onboard DDR2 RAM needs to be used. First tests with the memory and a memory management unit can be found in [52].

In order to transmit the coordinate data to the track reconstruction AMC an extendible protocol was developed based on Aurora. The layout of the protocol is depicted in Figure 8.7. With the 2 bit *type* the kind of data can be assigned. So far two different modes are supported: *command* to send a 32 bit instruction and *data*. The 14 bit *flag* determines more specifically the kind of data. *0x00* indicates coordinate data, while *0x01* is reserved for an empty event. The raw SVD can also be transmitted to the track reconstruction AMC for debugging purposes (using flag *0x2*). The size of the complete data packet is written into *size* and *TTRX* (an FTB board identification as source of the transmitted data) and the event number into the next 32 bits.

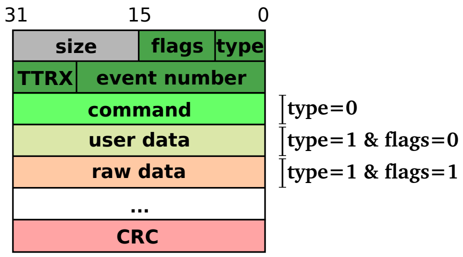


Figure 8.7.: Internal protocol with several modes. The nature of the data is determined by *type* and *flags*.

The four ring buffers in each concentrator AMC are read out consecutively over a multiplexer, once the complete event was recorded. The internal protocol framer starts to transmit the content of each ring buffer in separated packets, when the event is requested by the track reconstruction AMC. To improve the performance and avoid idle time, a coordinate buffer is used in the track reconstruction AMC to cache a few events locally. Up to five events can be stored. In case less than five events are in the local buffer, the memory management unit sends a command with the request to transmit the next event to all concentrator AMCs over the internal protocol. When there are no events ready or still in the pre-processing phase, the requests are queued in a 3 bit register in the concentrator AMC until the data are ready to be sent. The track reconstruction unit waits until all data of one event are received and stored.

9

Test Beam Results

The DATCON system was tested with a minimal setup, using the at the time available hardware in two test beam campaigns. Two AMCv2 boards were present at the first test beam campaign while an updated and most likely final version (AMCv3) was used in the second test beam. The purpose of these tests were to evaluate the feasibility of the track reconstruction algorithm under real conditions and to perform a first integration test into the Belle II DAQ. Several tests were scheduled before the first test beam campaign to check the communication and processing chains. The goal was to arrive as well prepared as possible to the test beam area and to discover and solve certain problems, e.g. protocol and format issues between the different subsystems, beforehand.
In the following chapter a detailed description of the two test beam setups, including the problems that occurred during the operation, is described. The solutions to these problems and the results are discussed in detail.

9.1. DESY Test Beam

The first test beam campaign took place at DESY, Hamburg from the 1st to the 31st of January, 2014. A short description of the test facility and its features are provided. Then the setup is described in detail. DESY offers several test beam areas providing an electron beam with selectable energies between 2 and 6 GeV with a rate of up to a few kHz. In addition, a superconducting solenoid magnet with a field of up to 1 T was available. The sensors were installed in a dry box in the magnet's coil. The beam is directed at the sensors in the dry box and therefore goes through the coil. This causes a broad momentum spectrum of the particles by Bremsstrahlung processes inside the coil. Figure 9.1 shows the test beam site and the solenoid magnet. The magnet is placed on a motor stage which allows one to move the magnet in three dimensions and to rotate it in the horizontal plane. Most of the DAQ components and power supplies were installed outside of the test beam area in a control hut to ease access and made debugging possible.

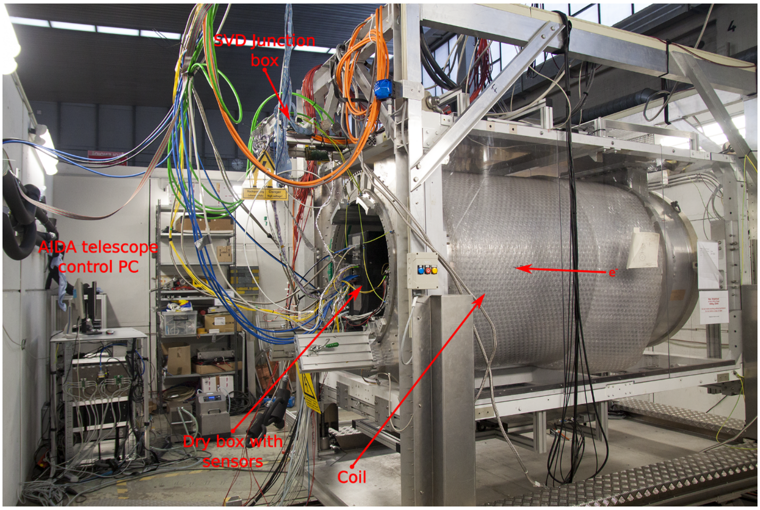


Figure 9.1.: Test beam hall at DESY. Shown are the solenoid providing a magnet field of up to 1 T, the dry box with the sensors and the direction of the beam.

9.1.1. Description of the Test Beam Setup

In order to practice the installation and operation of all components in the VXD system for the real experiment and to test the DAQ and computing concepts, a realistic small-scale model of the detector is required to provide the necessary sensor data. Especially to validate the data reduction scheme, at least four SVD sensors are required to reconstruct track segments and to create ROI. One PXD layer is necessary to match the in the SVD sensors found ROIs with pixel data. Figure 9.2 shows a CAD model of the sensors and the support structure located inside the dry box with the beam direction. The relative distances between the sensors match with the Belle II VXD specifications. The following components were installed in the solenoid magnet:

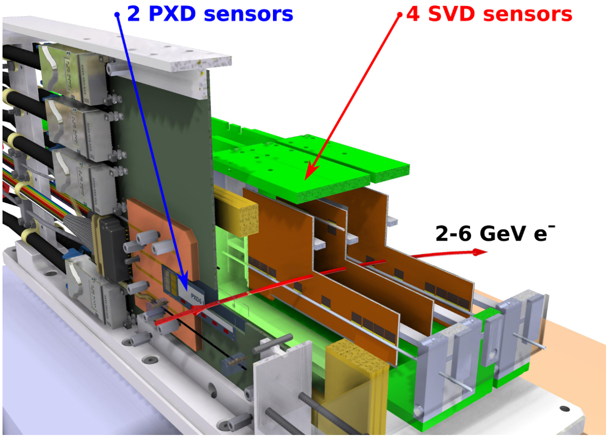


Figure 9.2.: CAD model of the test beam setup at DESY, showing the support structure as well as the PXD and SVD sensors. The whole assembly was located in a dry box inside the magnet.

- **DEPFET pixel sensor:** Only one large PXD matrix was available during the test beam campaign with 192×480 px and a pixel size of 50×75 μm^2. It was one of the first tested and operated big matrices with 50 μm thickness. The other DEPFET matrix arrived damaged and couldn't be used for the test. However, the data reduction scheme can also be tested with one sensor.

- **SVD sensors:** Four sensors of DSSD are installed in a row with the PXD. The first sensor representing layer 3 of the VXD consists of 768×768 strips with a pitch of 160×75 μm^2, while the other three sensors have 768×512 strips with 240×160 μm^2 pitch. The sensor in layer 5 was shifted down by 4.8 mm to move the strips connected to a broken APV25 chip out of the beam spot area.

- **AIDA telescope:** The six layers of the telescope system were used as reference planes to align the system and for efficiency studies. They are distributed over two arms with three sensors each, one before and one after the VXD sensor stack.

- **Scintillators:** Four scintillators, two crossed before and two after the telescope provide the trigger for the system. They are connected to a Trigger Logic Unit (TLU), which allows one to program different coincidence combinations between the scintillators in order to reduce the dark rate and trigger the VXD DAQ.

In addition to the sensors, prototypes of every DAQ component present in the Belle II backend system were involved in the readout of the sensors and data processing. The trigger system was also tested with several FTSWs controlled by one TTD. A simplified data flow chart of the different systems are shown in Figure 9.3. The data reduction was one of the major test subjects during the test beam campaign. Therefore, the complete HLT and a prototype system of the DATCON were present with two AMCv2 boards and one development crate. One (concentrator) AMC was responsible for acquiring the data from two FTBs (and also two FADCs, one connected to each implantation type of the SVD), while the other AMC received the pre-processed data and performed the track reconstruction, MPH extrapolation and ROI creation. Connected to the track reconstruction AMC was the merger node of ONSEN, receiving ROIs from both the HLT and the DATCON. ONSEN received the pixel data from one DHHC connected to one DHH and the PXD sensor. For testing purposes ONSEN can transmit the selected pixel data as well as the received ROI from both systems for a later analysis.

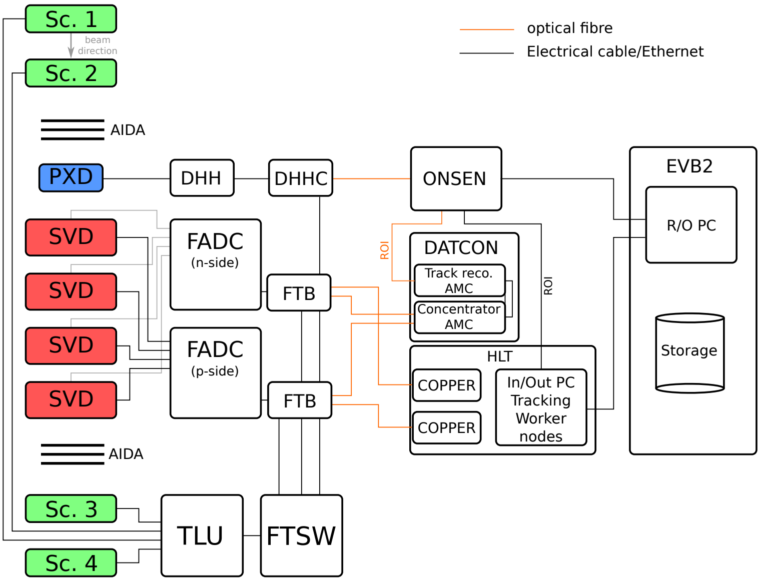


Figure 9.3.: Simplified illustration of the major DAQ components used in the DESY test beam campaign.

9.1.2. Preparations and Results

Before the begin of the DESY test beam campaign, the communication between the different subsystems was checked. The first tests took place in Vienna to check the protocol and link quality between the FTB and the DATCON. Random generated events were stored in a LUT inside the FADC and sent through the FTB to COPPER and a prototype DATCON AMCv2, where they were received and transmitted over a modified concentrator firmware to a readout PC. There the data were compared with the generated events. Several gigabytes of data were successfully transmitted over 12 hours without bit errors at a trigger rate of 60 kHz.
The second communication test dealt with the link to send ROIs from the track reconstruction AMCv2 to the ONSEN system. A few simulated events in BASF2 were loaded into a LUT inside a prototype of the FTB, which is connected to the DESY test beam DATCON system, consisting of one concentrator and one track reconstruction AMC. The calculated ROIs are sent to the ONSEN system and are compared with the expected ROI from the simulated events. Due to the lower speedgrade[1] of 1 of the FPGA in the AMCv2, a stable link could be established between both systems, but the test showed ~ 5 bit errors per second. This problem is fixed with the new AMCv3 and the higher speedgrade of 2, but the cards were not available for this particular test beam campaign. An analysis of the data showed, that always a single bit flip occurred every few events.

Figure 9.4 shows the installation of the DAQ crate with DHHC, ONSEN and DATCON. In the first two weeks, several subsystem tests were performed to prepare for real data runs, including the

[1]FPGAs from the same series are divided into different groups based on their performance. A higher speedgrade number means, the device can perform faster than one with a lower number.

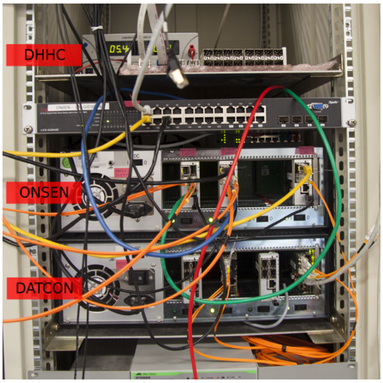


Figure 9.4.: Rack at DESY with DHHC, ONSEN and DATCON.

complete data chain, especially the combined connection between the SVD and PXD DAQ, data reduction scheme, the HLT data merging and recording. Afterwards, with beam energies of 6 GeV and trigger rates above 500 Hz the first combined data were recorded, including the ROIs from the DATCON.

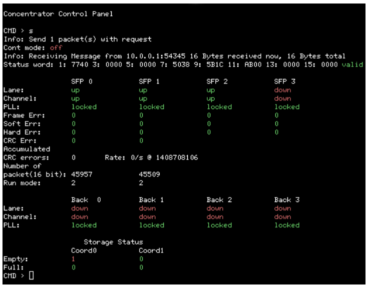
As mentioned in Section 7, the first implementation of the track reconstruction module worked similarly as expected from the simulation, but consumed much more resources. However, it provides the same results as the reimplementation. The HS was limited in φ (in HS coordinate system) between 1.3 to 1.7 which is in the acceptance area of all SVD sensors. That covers all tracks originating within 23° around the virtual interaction point. For the DESY test beam the "virtual" interaction point was fixed to the coil wall of the solenoid magnet, the entry point to the VXD dry box and source of secondary vertices. In addition, a prototype FPGA implementation of the sector-neighbour-finding algorithm was used in standalone runs and as pre-filter to find hits related to possible tracks for the HT module.

During the different runs, which took minutes to many hours the

DATCON modules were monitored over Chipscope to check the major modules of the track reconstruction process for a valid behaviour. In addition, a monitoring system was developed using Ethernet with the User Datagram Protocol (UDP) on each AMC board to transmit information about the system. A custom payload of the protocol contains the values of a 128 bit debug register inside each FPGA. The data are decoded and prepared by a *C* program and screenshots of the concentrator and track reconstruction monitor are shown in Figure 9.5. In the future, the system needs to be replaced with the Experimental Physics and Industrial Control System (EPICS)[2], which is integrated into the common Belle II slow control and DAQ monitoring system. However, for the test beam it provides valuable information about the status of the system during data taking and assists in the debugging of the system. The transmitted debug register include counters to indicate the number of CRC errors occurred in the communication between FTB and DATCON and between concentrator and track reconstruction AMC. During all tests no errors were detected with partial continuous day-long runs of data taking.

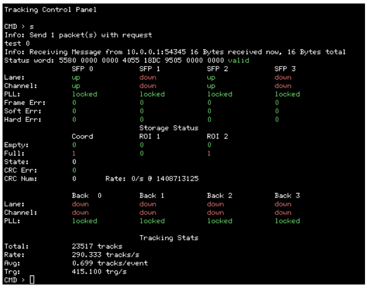
With the help of the AIDA telescope the alignment constants were calculated and included in the LUT of the concentrator and track reconstruction AMC. [51] Especially the relative positions between the PXD and SVD layers are important to create precise ROIs. For a more long-term verification, instead of evaluating event by event via Chipscope, a simple transmission module was written using the same UDP slow control interface. It is able to transfer the reconstructed 2D track parameters to a readout PC, but without any event information. The tracks are plotted from $20,000$ events in Figure 9.6 with the position of the PXD and SVD sensors. As one can see, most of the tracks are located in the center of the sensor, where the beam spot is located. The few outliers can be caused by fake tracks or bit flips in the Ethernet

[2]A slow control framework to manage and monitor scientific instruments [55].



```
Concentrator Control Panel

CMD > s
Info: Send 1 packet(s) with request
Cont mode: off
Info: Receiving Message from 10.0.0.1:54345 16 Bytes received now, 16 Bytes total
Status word: 1: 7740 3: 0000 5: 0000 7: 5038 9: 5B1C 11: AB00 13: 0000 15: 0000 valid

                SFP 0           SFP 1           SFP 2           SFP 3
Lane:           up              up              up              down
Channel:        up              up              up              down
PLL:            locked          locked          locked          locked
Frame Err:      0               0               0               0
Soft Err:       0               0               0               0
Hard Err:       0               0               0               0
CRC Err:        0               0
Accumulated
CRC errors:     0       Rate: 0/s @ 1408708106
Number of
packet(16 bit): 45957           45509
Run mode:       2               2

                Back  0         Back 1          Back 2          Back 3
Lane:           down            down            down            down
Channel:        down            down            down            down
PLL:            locked          locked          locked          locked

                   Storage Status
                Coord0          Coord1
Empty:          1               0
Full:           0               0
CMD > ▯
```

(a) Statistics for concentrator AMC.



```
Tracking Control Panel

CMD > s
Info: Send 1 packet(s) with request
test 0
Info: Receiving Message from 10.0.0.1:54345 16 Bytes received now, 16 Bytes total
Status word: 5580 0000 0000 4055 18DC 9505 0000 0000 valid
                SFP 0           SFP 1           SFP 2           SFP 3
Lane:           up              down            up              down
Channel:        up              down            up              down
PLL:            locked          locked          locked          locked
Frame Err:      0               0               0               0
Soft Err:       0               0               0               0
Hard Err:       0               0               0               0
                                Storage Status
                Coord           ROI 1           ROI 2
Empty:          0               0               0
Full:           1               0               1
State:          0
CRC Err:        0
CRC Num:        0       Rate: 0/s @ 1408713125

                Back  0         Back 1          Back 2          Back 3
Lane:           down            down            down            down
Channel:        down            down            down            down
PLL:            locked          locked          locked          locked

                                Tracking Stats
Total:          23517 tracks
Rate:           290.333 tracks/s
Avg:            0.699 tracks/event
Trg:            415.100 trg/s
CMD > ▯
```

(b) Statistics for tracking AMC.

Figure 9.5.: Slow control panel for the concentrator and track reconstruction AMC at the DESY test beam campaign.

packet, which were not protected by a CRC. This communication issue was fixed in the next hardware iteration. Fake reconstructed tracks (randomly intersecting lines in the HS) also contribute to the outliers.

A fully working complete DAQ chain was established in the last week of the test beam campaign. A beam with straight tracks with energies between 4 and 6 GeV at trigger rates of up to 1 kHz was reached. DATCON and HLT provided the ROIs to the ONSEN system which performs the pixel selection and embedded the received ROIs in their data. This data is sent over Ethernet to the storage PC and are written to disc. Each ROI has a fixed size of 16×16 pixels, or less when they are located near a boundary of a sensor. Later the ROI could be analysed with the DQM module for BASF2. The ROI for the DATCON system can be identified by a single bit in the flags of the ROI format. In the first test, a buffer overflow was fixed in the ROI calculation, creating very large ROI out of the boundary of the PXD sensor. During analysis of the plots, the aforementioned communication problem was

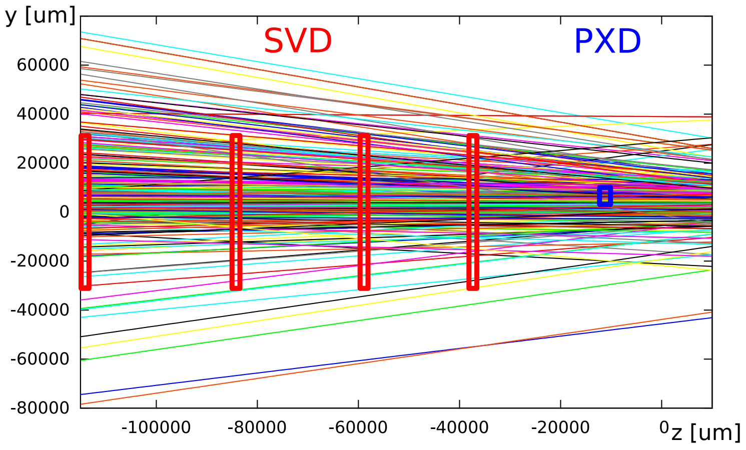


Figure 9.6.: HS plot showing 20.000 tracks taken during the DESY test beam campaign over a debug side channel of the track reconstruction AMC. Most of the tracks are located in the area of the beam spot. The outliers, not intersecting with the sensor, are caused by fake tracks or bit flips during transmission.

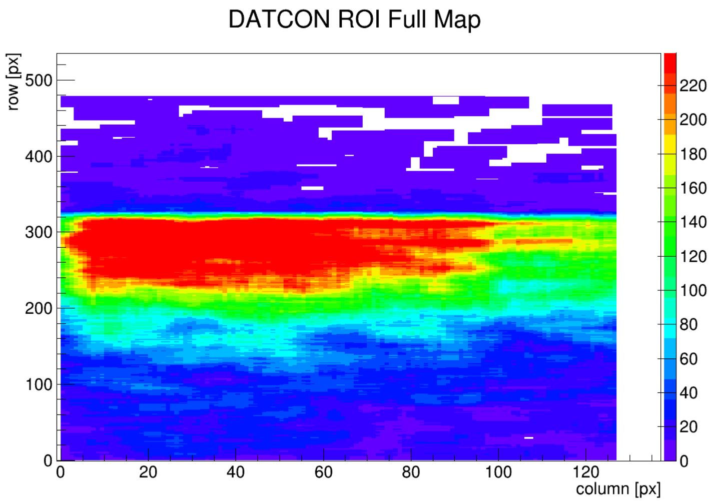


Figure 9.7.: Accumulated ROI map taken from run 623 at DESY with 6 GeV electron beam, no magnet field at 500 Hz trigger rate.

observed between DATCON and ONSEN. Flipping bits during transmission caused the ROI to be out of bound as well. Since the size of the sensor is known and the ROI size is fixed most of the ROI could be corrected, or removed if they are not unrecoverable. Figure 9.7 shows the corrected accumulated ROI map. It shows that most of the ROIs are centered at the beam spot area, with a few outliers. The sharp cut at 320 pixel is caused by a deactivated, broken APV25 chip in layer 5 in that area. Since this track reconstruction module requires four hits to create a track candidate almost no track and therefore no ROI could be calculated. The beam spot recorded with the PXD is located between row pixel 200 and 350 matching with the calculated ROIs.

9.1.3. Conclusion from the DESY Test Beam

The communication between the FTB and DATCON worked during the whole test beam without errors while millions of events were recorded. However, several issues were discovered and partially fixed during the test beam. The remaining points will be shortly discussed here.
In case no track is reconstructed and subsequently no ROI created, no data is sent to the ONSEN system. However, it is required to at least propagate that the event was processed by the DATCON system. An event, although empty needs to be sent. This required intense work and modifications at the event management unit on concentrator as well as on the track reconstruction side. There are several processing steps, where an event can be filtered out, e.g. no hits in the detector, then the SVD decoder needs to signal the event management that the event is empty or the noise filter removes noise signals without usable strips to process. An empty internal packet needs to be sent to the track reconstruction AMC, so it can sent an empty ROI packet to ONSEN. The "empty event flag" was introduced for this purpose in the internal communication protocol.
Two related problems occurred in the communication between DATCON and ONSEN to transmit the ROIs. The first problem about the bit errors in the transmission line was later fixed with a new AMC revision. A second problem concerns the different CRC implementations between the two systems. Without a common CRC algorithm no bit errors can be detected, so it was deactivated in ONSEN. In a later test, it was decided to use the CRC module already included in the FPGA and successfully tested it with a one day long run without CRC errors. This confirmed that the hardware problems were solved and the AMCv3 is ready for final production.
In some rare cases, the DATCON system suddenly stopped working when broken FADC data packets arrived at the concentrator

AMC. It was fixed with an implementation of a watchdog in all critical components to reset the module after remaining in a certain non-idle state for too long.

9.2. CERN Test Beam Campaign

The goal of the second test beam campaign was to evaluate the re-design of the track reconstruction module to obtain a faster and more parallel design (as described in Section 7.3.2). It took place from 16th to 23rd of November, 2014 at CERN. In contrast to the DESY test beam, only the four sensor SVD stack was installed at the Super Proton Synchroton (SPS) collider to debug the new track reconstruction unit. Also most of the Belle II DAQ, especially the backend (e.g. HLT, or COPPER) were not present in this test beam. This required certain modifications to some parts of the system to run in stand-alone mode as described in the next section.

9.2.1. Description of the Test Beam Setup

At the SPS test beam line H6A in building 887, a 120 GeV pion beam is used to irradiate the test sensors. The DESY SVD sensor stack is installed and chilled with a liquid CO_2 cooling system inside a movable motor stage. Figure 9.8 shows the setup. The junction box is placed outside with connection to the FADC-crate and power supply. On the back of the FADC, the FTB is installed with two 50 m long optical cables going to the control room, where the DATCON prototype crate is located.
The firmware of the FTB relies on the presence of the FTSW to obtain the 127.21 MHz clock and the trigger information. A modification of the firmware was required to change the input

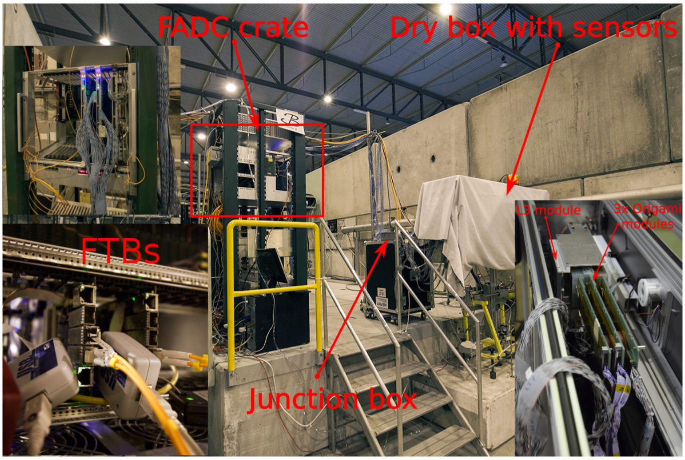


Figure 9.8.: CERN test beam setup in the experimental area showing the sensor stack installed in a dry box with the connected junction box and the FADC/FTB crate.

clock to the local 127.21 MHz oscillator and a simple trigger logic was implemented that increases a counter in the FTB on every "start of event" received from the FADC. Several other signals were routed differently or fixed to a special value to get the system operational.

The DATCON system used three AMCs, two with the known configuration from the DESY test beam, namely the concentrator and the track reconstruction AMC. A third AMC was required as ONSEN emulator to receive the track parameters from the track reconstruction AMC. The focus on this test beam was to test the new track reconstruction module, so no ROIs were calculated, since the PXD sensor and the rest of the DAQ were not present. The ONSEN emulator is a simple firmware which frames the incoming data on the optical links into Ethernet packets and sends

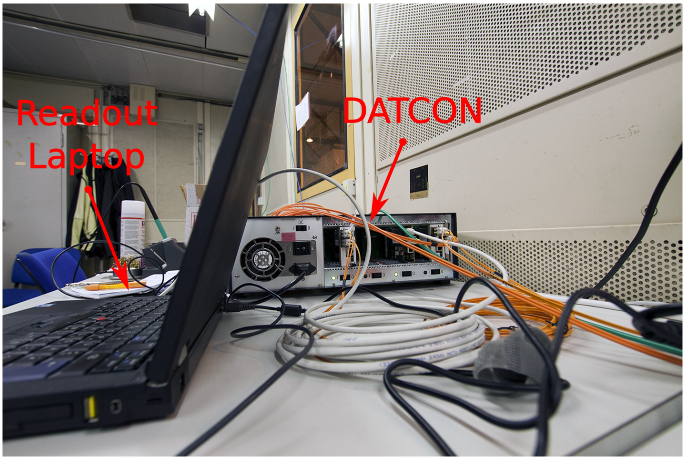


Figure 9.9.: DATCON setup with the readout PC in the control hut.

them out to a PC/laptop for data recording and analysis. The system is depicted in Figure 9.9.

Due to the absence of the Belle II EVB2 no raw data from the SVD were available to evaluate if the HS were correctly sectorized. Therefore the debug and calibration interface of the SVD was used, called “TuxDAQ”. It accesses the FADC data over the slower VME bus. The data is stored on another readout PC in a special TuxDAQ format, which can be later converted into BASF2 Digits for analysis.

9.2.2. Preparations and Results

Compared to the DESY test beam many subsystems like the FADC, the FTB and the DATCON hardware were present in a newer hardware revision. First basic tests were performed to check the communication between the components and cross-check the operation of the FTB firmware in “DATCON-only” mode without the Belle II DAQ. The conditions of the beam (high dense spills) provided by SPS are also different from DESY (continuous beam) making new developments in the firmware of the FADC and FTB necessary. The development of new features to cope with the different situation consumed most of the time. A configurable dead time was implemented in the firmware of the FADC Controller that receives the trigger from the scintillators over a Nuclear Instrumentation Standard (NIM) input. Several issues with the software and the new hardware revision have been discovered and solved. Still a significant noise problem similar to the one observed at the DESY test beam campaign was present in the system. Due to software and configuration issues at first no correct zero-suppression threshold could be loaded inside the FADC, thus this mode couldn’t be used. With a workaround a fixed threshold of 10 ADUs, which corresponds to the average calculated threshold of each strip, were used and data could be recorded.

At a trigger rate of 60 Hz DATCON processed data and sends the HS over the debug framer described in Section 7.3.2 to the readout PC and they are saved in a file. Meanwhile TuxDAQ recorded the zero-suppressed data. The SVD unpacker module for the strip data was modified to decode the TuxDAQ format. In addition, a module to analyse the sent HS data from DATCON was constructed to extract the track parameters. An example of an HS obtained with the FPGA hardware is shown in Figure 9.10. The visualization with the geometry of SVD is shown in Figure 9.11. Because of the fixed zero-suppression threshold the noise was very high, so many events were rejected by the high-occupancy filter. A visual inspection showed that the reconstructed tracks fit quite well, although only a short time was available for an online analysis of the data to obtain basic alignment constants.

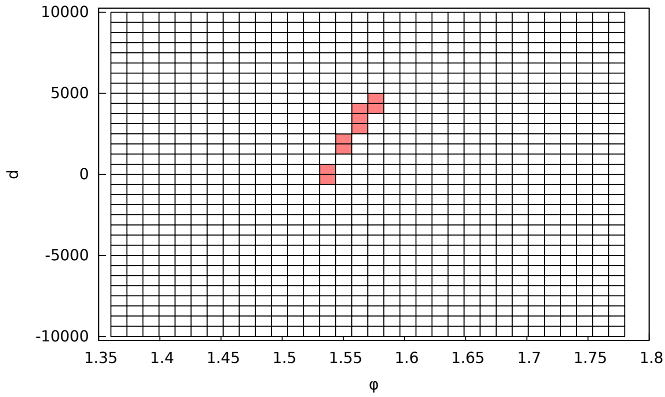


Figure 9.10.: HS of the n-side, processed and obtained with the full pre-processing and track reconstruction chain. The active sectors of the grid are marked in red corresponding to one successfully reconstructed track.

In the last day several issues with the FADC, trigger system and zero-suppression mode was fixed, but there was no more time to start up the DATCON system. With this fix to the dead time implementation in the FADC the trigger rate increased to 270 Hz and in the last hour $100,000$ events were recorded and later used for analysis with correct individually uploaded pedestal and zero-suppression thresholds.

The TuxDAQ unpacker was later used to read the data into BASF2 and with the simulation code of the FPGA an estimation of the track efficiency was calculated. The first one thousand events were visually checked event by event for the reconstruction efficiency. Therefore the visualization module of BASF2 was adapted accordingly to display the reconstructed tracks. With the settings also used for the FPGA the fake rate was very low (almost zero) and every reconstructed track matches the hits in the SVD.

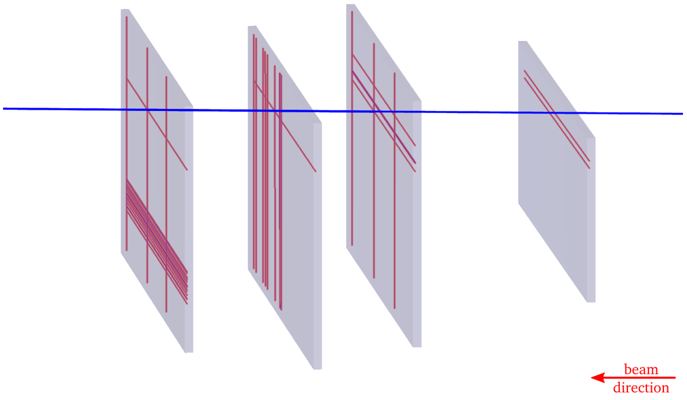


Figure 9.11.: Visualization of an event taken with TuxDAQ and track reconstruction performed with the DATCON hardware system at the CERN test beam campaign. The data is not processed by the noise filter and no cooling and custom zero-suppression cuts could be applied for this recording.

An example of such a visualized event is shown in Figure 9.12. It shows a much better noise behaviour compared to the previous run, caused by the correctly uploaded threshold values. Assuming this is also correct for the $99,000$ other events and there is only one track per event, the number of reconstructed tracks divided by the total number of events provides an estimation of the track efficiency. About 50 thousand tracks were recovered in this run, corresponding to an efficiency of 50 percent.
An investigation of the other events, where no tracks were reconstructed, reveals several issues listed below:

- Layer 3 n-side was disabled, due to a configuration error in the APV25s. This means that on the n-side in all remaining

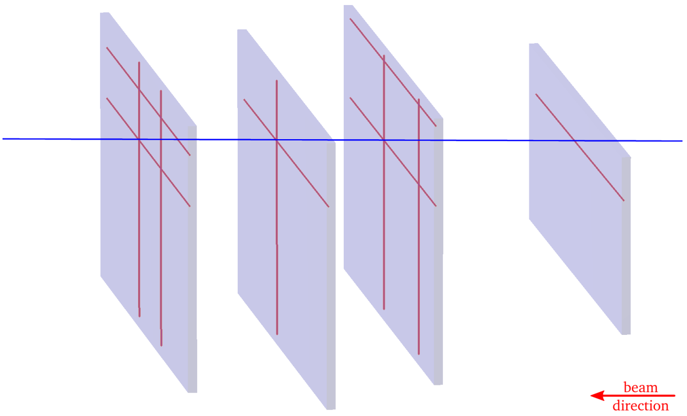


Figure 9.12.: Visualization of one track at the CERN test beam, reconstructed with the BASF2 FPGA simulation code. Individual zero-suppression values were uploaded and the noise filter was used to obtain the strip data.

three sensors a particle hit must be present for reconstruction.

- Noise caused a lot of high occupancy events with sometimes hundred of hits in the sensors. These events were rejected by the high occupancy filter since no useful tracking is possible with this large number of hits and ghost hits.

- No tuning of the HS sectors were done during the run and missing precise alignment constants, especially for the layer 3 sensors. This sensor was accidentally shifted during installation.

- Empty events with only a few hits, not enough to create a track, are counted as well.

In conclusion, despite the problems during the CERN test beam campaign and the very short hardware DATCON run, the new HT implementation was successfully tested. The simulation of the algorithm shows that even under difficult circumstances, the track efficiency is high to offer precise track parameters at low fake rates.

10

Summary and Outlook

The flavour factory SuperKEKB is designed to run at an instantaneous luminosity a factor of 40 times higher than its predecessor KEKB, which will allow for more precise measurements of Standard Model parameters, CP violation and rare decays. A new vertex detector is required to cope with the higher backgrounds at the inner layers. While the DEPFET pixel technology fulfills all the requirements it generates too much data to be handled by the backend with its 8 million readout channels and 20 μs integration time. Based on Field Programmable Gate Arrays (FPGA) a data reduction system, called DATCON, has been developed using information from the surrounding Silicon strip Vertex Detector (SVD). A suitable algorithm for track reconstruction was found in the Hough Transformation. This algorithm was in this thesis developed and adapted to process the SVD data. It was first constructed as a module in a simulation environment (BASF2) and extensively tested before implemented in the FPGA. In addition, it was shown that it can cope with the expected rates in the SVD while keeping a high track reconstruction efficiency. The Region of Interests (ROI) efficiency with the ROIs calculated from track parameters provided by the DATCON system is over 95 percent with a data reduction factor of 45. That's more than four times higher than the required factor of 10.
In this thesis, the track reconstruction algorithm was ported to FPGAs. The complexity of the implementation is much higher than in conventional computer systems, caused by the architectural concept. However, it provides certain advantages, especially regarding speed, by its capability of parallel data processing. Additionally, the data is directly available to the FPGA and the

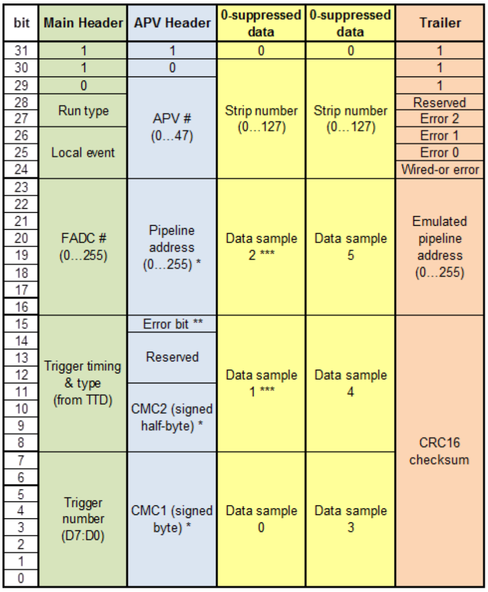
complex transmitting to a CPU and its overhead during calculation can be avoided. On the other hand, the debugging of the system is much more time consuming than with commonly integrated tools like a C++ debugger. The Chipscope tool provides only a limited view of all the signals and operations going on inside the FPGA. Therefore, the algorithm has been tested in a HDL simulation model before the firmware was synthesized and loaded in hardware.
In two test beam campaigns at DESY and CERN, the small scale DATCON system with two Advanced Mezzanine Cards (AMC) equipped with a Xilinx Virtex 5 FPGA, were successfully tested. This includes the firmware programming of the FPGAs of the complete processing chain from acquiring the data of the SVD front-end electronics over the track reconstruction to the ROI calculation and transmission. For the first time, all components were running together in preparation for the final detector installation. Several issues have been solved and many lessons learned for the operation of the Belle II experiment that will start data taking in 2017.

As the 12 final AMCs and the custom backplane required for the communication between the AMCs were not yet available at the time of writing this thesis, the scaling to the full system to read out over 200 strip sensors could not be tested. However, the design of the backplane was finalized in this thesis and the boards will be produced by IHEP Beijing. A challenge in scaling to the full system lies in the event management. It is required to scale with the number of connected AMCs and to provide the coordinates to the tracking module fast enough, so it can transmit the ROIs to ONSEN in the specific time window. Moreover, in the last test beam campaign the ROI algorithm couldn't be tested. A follow-up test beam at DESY combining the finalized hardware of the pixel detector and SVD system is scheduled in spring 2016.
Several ideas for future improvements of the track reconstruction

performance were presented, e.g. a better description of the sine shape trajectory in z-direction compared with the so far used linear approximation. To increase the ROI efficiency in the overlapping sectors of the pixel sensors, ROIs for both sensors are required. Curling tracks passing several times through the pixel detector should also create one ROI per passage.
While the possible gain in performance is still uncertain, even without these improvements, the DATCON system in its current form already provides sufficiently precise ROIs for suitable Belle II operation. This has been proved by several BASF2 simulations during the writing of this thesis. Combined with the other data rescue mechanism of the high level trigger and the cluster analysis for very low-momentum tracks, ~ 99 percent of ROI efficiency at the designed data reduction factor of 10 can be reached. In fact, the DATCON system provides already more than 95 percent with a four times higher data reduction factor.

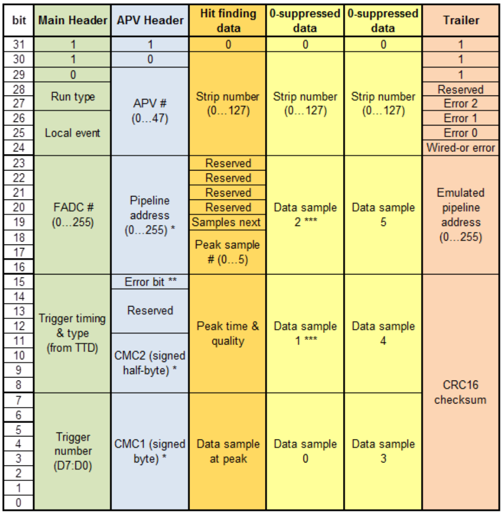
The FADC Protocol

Two layouts of the FADC protocol are described in this appendix. Figure A.1 shows the layout in case of run mode 2 (zero-suppressed) while Figure A.2 describes the hit-time mode. The hit-time mode can also write six, respectively three consecutive samples, when the hit-time cannot be precisely determined, for instance in pile-up events.



bit	Main Header	APV Header	0-suppressed data	0-suppressed data	Trailer
31	1	1	0	0	1
30	1	0	Strip number (0…127)	Strip number (0…127)	1
29	0	APV # (0…47)			1
28	Run type				Reserved
27					Error 2
26	Local event				Error 1
25					Error 0
24					Wired-or error
23	FADC # (0…255)	Pipeline address (0…255) *	Data sample 2 ***	Data sample 5	Emulated pipeline address (0…255)
22					
21					
20					
19					
18					
17					
16					
15	Trigger timing & type (from TTD)	Error bit **	Data sample 1 ***	Data sample 4	CRC16 checksum
14		Reserved			
13					
12					
11		CMC2 (signed half-byte) *			
10					
9					
8					
7	Trigger number (D7:D0)	CMC1 (signed byte) *	Data sample 0	Data sample 3	
6					
5					
4					
3					
2					
1					
0					

Figure A.1.: Zero-suppression data format of the FADC.



bit	Main Header	APV Header	Hit finding data	0-suppressed data	0-suppressed data	Trailer
31	1	1	0	0	0	1
30	1	0	Strip number (0…127)	Strip number (0…127)	Strip number (0…127)	1
29	0	APV # (0…47)				1
28	Run type					Reserved
27						Error 2
26	Local event					Error 1
25						Error 0
24						Wired-or error
23	FADC # (0…255)	Pipeline address (0…255) *	Reserved	Data sample 2 ***	Data sample 5	Emulated pipeline address (0…255)
22			Reserved			
21			Reserved			
20			Reserved			
19			Samples next			
18			Peak sample # (0…5)			
17						
16						
15	Trigger timing & type (from TTD)	Error bit **	Peak time & quality	Data sample 1 ***	Data sample 4	CRC16 checksum
14		Reserved				
13						
12						
11		CMC2 (signed half-byte) *				
10						
9						
8						
7	Trigger number (D7:D0)	CMC1 (signed byte) *	Data sample at peak	Data sample 0	Data sample 3	
6						
5						
4						
3						
2						
1						
0						

Figure A.2.: Zero-suppression and hit time data format of the FADC.

The Belle II Coordinate System

Several coordinate systems are well established in the different tracking systems and in use. Figure B.1 shows the absolute, the BASF2 and the DATCON coordinates. The BASF2 coordinate system is based on spherical coordinates, but always rotated so that the unit vector of φ and θ are perpendicular to each other, in contrast to the DATCON coordinates. The scope of φ is -180 to $180°$ and the acceptance region of the VXD in θ is from 17 to $150°$.

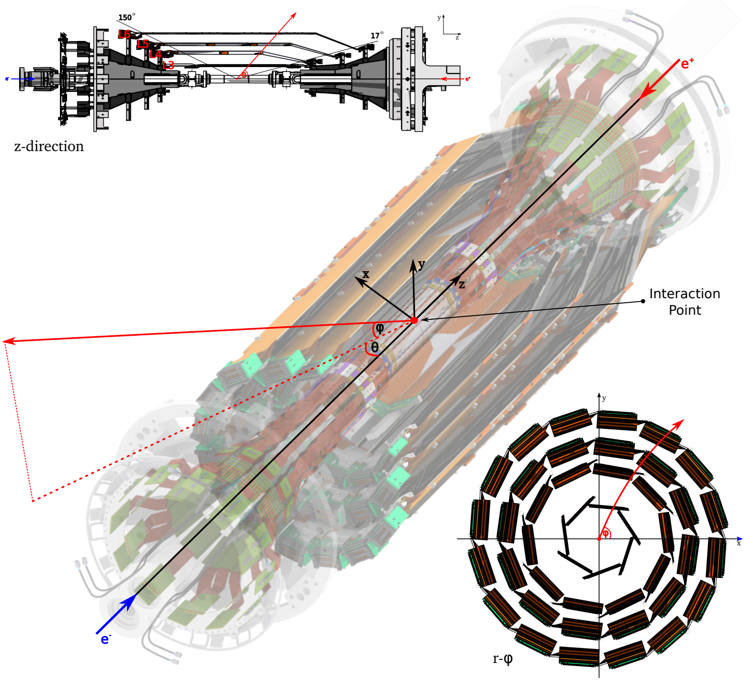
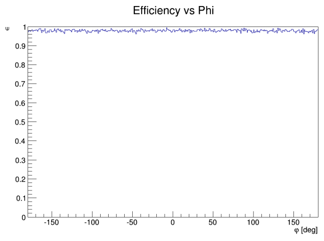


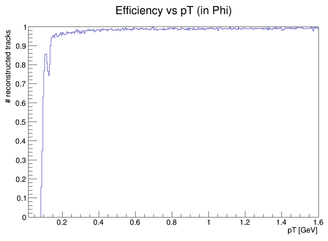
Figure B.1.: Different coordinate systems used in Belle II and DATCON: Cartesian and spherical coordinates.

CDC Simulation with HT

The HT algorithm was also tested with CDC data to prove the applicability to run on other tracking detectors. As described, only the pre-processing stages and some HS parameters needed to be changed. The efficiencies of the algorithm with 5 generated tracks (e^+, e^-) per event and with an initial track momentum ranging from 250 MeV to 2 GeV are shown in Figure C.1.



(a)



(b)

Figure C.1.: Efficiencies of the HT algorithm in r-φ-plane of the CDC with 5 tracks and $10,000$ events. It shows in φ a constant high track reconstruction efficiency until not enough wires were reached by a low-momentum track.

Acronyms

ADC	Analog-to-Digital Converter
ADU	Analog Digitcal Unit
AMC	Advanced Mezzanine Card
ASIC	Application Specific Circuit
ATCA	Advanced Telecommunications Computing Architecture
BASF	Belle AnalysiS Framework
BASF2	Belle Analysis Software Framework 2
BRAM	Block Random Access Memory
CBM	Compressed Baryonic Matter
CDC	Central Drift Chamber
CERN	European Organization for Nuclear Research
CLB	Configurable Logic Block
CM	Common Mode
CMC	Common Mode Correction
CMOS	Complementary Metal Oxide Semiconductor
CMS	Compact Muon Solenoid
CoG	Center of Gravity

COPPER	Common Pipeline Platform for Electronics Readout
CPP	C++
CPU	Central Processing Unit
CRC	Cyclic Redundancy Check
DATCON	Data Acquisition Tracking and Concentrator Online Node
DAQ	Data Acquisition
DCD	Drain Current Digitizer
DCM	Digital Clock Management
DDR2	Double Data Rate 2
DESY	Deutsches Elektronen SYnchroton
DEPFET	DEpleted p-channel Field Effect Transistor
DHP	Data Handling Processor
DHH	Data Handling Hybrid
DHHC	Data Handling Hybrid Controller
DIMM	Dual Inline Memory Module
DRF	Data Reduction Factor
DSP	Digitial Signal Processor

DSSD	Double-Sided Strip Detector
DQM	Data Quality Monitoring
EEPROM	Electrically Erasable Programmable Read-Only Memory
EPICS	Experimental Physics and Industrial Control System
EVB	Event Builder
EVB2	Event Builder 2
ENC	Equivalent Noise Charge
FADC	Fast Analog-to-Digitcal Converter
FEE	Front End Electronics
FF	Flip Flop
FIFO	First-In First-Out
FPGA	Field Programmable Gate Arrays
FPU	Floating Point Unit
FSM	Finite State Machine
FTB	Finesse Transmitter Board
FTSW	Fast Timing Switch
HDL	Hardware Description Language

HER	High Energy Ring
HLT	High Level Trigger
HS	Hough Space
HT	Hough Transformation
HTTP	HyperText Transfer Protocol
HV	High Voltage
IBL	Insertable B-Layer
IEEE	Institute of Electrical and Electronics Engineers
IHEP	Institute of High Energy Physics
IOB	Input/Output Buffers
IP	Interaction Point
IPMI	Intelligent Platform Management Interface
JTAG	Joint Test Action Group
KEK	High Energy Accelerator Research Organization
LHC	Large Hadron Collider
LER	Low Energy Ring
LVDS	Low Voltage Differential Signal
LUT	Lookup Table

MIP	Minimum Ionizing Particle
MOSFET	Metal Oxide Field Effect Transistor
MPH	Most Probable Hit
MSSM	Minimal Supersymmetric Standard Model
mTCA	Micro Telecommunications Computing Architecture
NIM	Nuclear Instrumentation Standard
RAM	Random Access Memeory
RBB	Radiative Bhabha
ROI	Region of Interest
ONSEN	ONline SElector Node
OSI	Open Systems Interconnection model
PC	Personal Computer
PCB	Printed Circuit Board
PCIe	Peripheral Component Interconnect express
PICMG	PCI Industrial Computer Manufacturers Group
PXD	Pixel Detector
SFP	Small Form Plugable
SVD	Silicon strip Vertex Detector

SM	Standard Model
SNR	Signal to Noise Ratio
SR	Synchrotron Radiation
SPS	Super Proton Synchroton
SUSY	Supersymmetry
TC	Track Candidate
TCP/IP	Transmission Control Porotocol/Internet Protocol
TLU	Trigger Logic Unit
TTD	Trigger Timing Distribution
UART	Universal Asynchronous Receiver/Transmitter
UDP	User Datagram Protocol
USB	Universal Serial Bus
VME	Versa Module Europa
VXD	Vertex Detector

Bibliography

[1] M. Fiorini, V. Carassiti, A. Ceccucci, E. Cortina, A. Cotta Ramusino, G. Dellacasa, P. Jarron and J. Kaplon *et al.*, "The NA62 gigatracker: Detector properties and pixel read-out architectures," Nucl. Instrum. Meth. A **624** (2010) 314.

[2] M. A. Kagan [ATLAS Collaboration], "Overview of the ATLAS Insertable B-Layer (IBL) Project".

[3] H. Qiu [STAR Collaboration], "STAR heavy flavor tracker," Nucl. Phys. A **31** (2014) 1141.

[4] S. Kurokawa, "Overview of the KEKB accelerators," Nucl. Instrum. Meth. A **499** (2003) 1.

[5] A. Abashian, K. Gotow, N. Morgan, L. Piilonen, S. Schrenk, K. Abe, I. Adachi and J. P. Alexander *et al.*, "The Belle Detector," Nucl. Instrum. Meth. A 479, 117 (2002).

[6] H. Baer, "Supersymmetry: Theory overview," AIP Conf. Proc. **792** (2005) 595.

[7] A. M. Teixeira, "The Next-to-Minimal Supersymmetric Standard Model: An Overview," arXiv:1106.2103 [hep-ph].

[8] J. Louis, T. Mohaupt and S. Theisen, "String theory: An overview," Lect. Notes Phys. **721** (2007) 289.

[9] M. Kobayashi and T. Maskawa, "CP Violation in the Renormalizable Theory of Weak Interaction," Prog. Theor. Phys. **49**, 652 (1973).

[10] A. Ceccucci, Z. Ligeti and Y. Sakai, "The CKM quark-mixing matrix,"

[11] F. J. Gilman, "Overview Of The Standard Model," Annals N. Y. Acad. Sci. **518** (1987) 159.

[12] D. d'Enterria, "Physics at the LHC: A Short overview," J. Phys. Conf. Ser. **270** (2011) 012001 [arXiv:1010.1491 [hep-ex]].

[13] Y. Cai, S. Ecklund, A. Novokhatski, J. Seeman, A. Seryi, M. Sullivan, U. Wienands and M. Biagini *et al.*, "Design of an Asymmetric Super-B Factory," Conf. Proc. C **060626** (2006) 646.

[14] I. Adachi *et al.* [sBelle Design Group Collaboration], "sBelle Design Study Report," arXiv:0810.4084 [hep-ex].

[15] J. Kemmer and G. Lutz, "New detector concepts", Nucl. Instrum. Meth. A 253 (1987) 365.

[16] M. Koch, "Development of a Test Environment for the Characterization of the Current Digitizer Chip DCD2 and the DEPFET Pixel System for the Belle II Experiment at SuperKEKB," CERN-THESIS-2011-084.

[17] M. Raymond, G. Cervelli, M. French, J. Fulcher, G. Hall, L. Jones, L. K. Lim and G. Marseguerra it et al., "The CMS tracker APV25 0.25-mu-m CMOS readout chip," Conf. Proc. C 00091111, 130 (2000).

[18] M. Friedl, K. Ackermann, H. Aihara, T. Aziz, T. Bergauer, A. Bozek, A. Campbell and J. Dingfelder *et al.*, "The Belle II Silicon Vertex Detector," Nucl. Instrum. Meth. A **732** (2013) 83.

[19] B. Stroustrup, "The C++ Programming Language", 1985.

[20] A. Moll, “The software framework of the Belle II experiment,” J. Phys. Conf. Ser. 331, 032024 (2011).

[21] S. Agostinelli *et al.* [GEANT4 Collaboration], “GEANT4: A Simulation toolkit,” Nucl. Instrum. Meth. A **506** (2003) 250.

[22] R. Brun and F. Rademakers, “ROOT: An object oriented data analysis framework,” Nucl. Instrum. Meth. A **389** (1997) 81.

[23] Libxml Library: `http://www.xmlsoft.org/`.

[24] Boost C++ Libraries: `http://www.boost.org/`.

[25] E. Gatti and P. Rehak, “Semiconductor Drift Chamber - An Application Of A Novel Charge Transport Scheme,” Nucl. Instrum. Meth. A 225 (1984) 608.

[26] H. Kruger [DEPFET Collaboration], “Front-end electronics for DEPFET pixel detectors at SuperBelle (BELLE II),” Nucl. Instrum. Meth. A 617 (2010) 337.

[27] I. Peric, P. Fischer and T. H. H. Nguyen, “DCDB and SWITCHERB, the readout ASICS for belle II DEPFET pixel detector,” IEEE Nuclear Science Symposium and Medical Imaging Conference (2011) p.1536-1539.

[28] M. Lemarenko, M. Havranek, T. Hemperek, T. Kishishita, H. Kruger and N. Wermes, “The data handling processor for the Belle II pixel vertex detector: Efficiency optimization,” JINST **7** (2012) C01069.

[29] C. Irmler, M. Friedl, J. van Hoorne and H. Steininger, “Efficient signal conditioning by a FIR filter for analog signal transmission over long lines,” JINST **7** (2012) C01082.

[30] P. V. C. Hough, "Machine Analysis Of Bubble Chamber Pictures," Conf. Proc. C **590914** (1959) 554.

[31] R. Duda and P. Hart, "Use of the Hough transformation to detect lines and curves in pictures" Commun. ACM 15, 1 (January 1972), 11-15.

[32] D. Levit, I. Konorov, D. Greenwald and S. Paul, "FPGA Based Data Read-Out System of the Belle 2 Pixel Detector," arXiv:1406.3864 [physics.ins-det].

[33] T. Geßler, W. Kühn, J. S. Lange, Z. Liu, D. Münchow, B. Spruck and J. Zhao, "The ONSEN Data Reduction System for the Belle II Pixel Detector," arXiv:1406.4028 [physics.ins-det].

[34] M. Nakao, T. Higuchi, R. Itoh and S. Y. Suzuki, "Data acquisition system for Belle II," JINST **5** (2010) C12004.

[35] R. Itoh, T. Higuchi, M. Nakao, S. Y. Suzuki and S. Lee, "Data flow and high level trigger of Belle II DAQ system," Real-Time Conference, 2012.

[36] S. Y. Suzuki, T. Higuchi, M. Nakao, R. Itoh and Y. Igarashi, "Upgrading the backend of the pipeline readout system for Belle II," Real-Time Conference, 2012.

[37] C. Steinle, "A first level trigger approach for the CBM experiment" Disserta Verlag; Hamburg (Germany) 2012.

[38] C. Wessel, "Studies of the Track Reconstruction Algorithm in the Vertex Detector of the Belle II Experiment", Bachelor thesis, Bonn 2014.

[39] D. J. Lilhja, S. S. Sapatnekar, “Designing Digital Computer Systems with Verilog”, 2005.

[40] IEEE, “IEEE 754-1985”: `http://754r.ucbtest.org/standards/754.pdf`.

[41] D. Knuth, “The Art of Computer Programming”, 1997.

[42] C. E. Cummings, “The Fundamentals of Efficient Synthesizable Finite State Machine Design using NC-Verilog and BuildGates”, International Cadence Usergroup Conference, 2002.

[43] Xilinx Spartan 3 User Guide: `http://www.xilinx.com/support/documentation/user_guides_ug332.pdf`.

[44] Xilinx LogiCORE IP Divider Generator v3: `http://www.xilinx.com/support/documentation/ip_documentation/div_gen_ds530.pdf`.

[45] Xilinx LogiCORE IP Aurora 8b/10b Generator: `http://www.xilinx.com/support/documentation/ip_documentation/aurora_8b10b_ug353.pdf`.

[46] Thomas & Moorby, “The Verilog Hardware Description Language”, 1998.

[47] McCluskey E.J., “Minimization of Boolean function”, Bell System Tech. Journal, vol. 35, No. 5, pp. 1417-1444, 1956.

[48] A. Janwary, G. Leavey, K. So, “ATCA Design Consideration for Telecommunication Platforms”, PMC-Sierra, White-Paper, 2008.

[49] Questa Advanced Simulator, Mentor Graphics: `http://www.mentor.com/products/fv/questa/`.

[50] M. Friedl, T. Bergauer, F. Buchsteiner, G. Casarosa, F. Forti, K. Hara, T. Higuchi and C. Irmler *et al.*, "First results of the Belle II Silicon Vertex Detector readout system," JINST **9** (2014) 12, C12005.

[51] T. Bilka, G. Casarosa, R. Frühwirth, C. Kleinwort, P. Kodys, P. Kvasnicka, J. Lettenbichler and E. Paoloni *et al.*, "Demonstrator of the Belle II Online Tracking and Pixel Data Reduction on the High Level Trigger System," arXiv:1406.4955 [physics.ins-det].

[52] M.Schnell, "Data Concentrator for the Belle II Pixel Detector", 2011.

[53] S. Chatrchyan *et al.* [CMS Collaboration], "Commissioning and Performance of the CMS Silicon Strip Tracker with Cosmic Ray Muons," JINST 5 (2010) T03008.

[54] M. Friedl, C. Irmler and M. Pernicka, "Time resolution of a few nanoseconds in silicon strip detectors using the APV25 chip,".

[55] "Experimental Physics and Industrial Control System": `http://www.aps.anl.gov/epics/about.php`.

List of Figures

ACKNOWLEDGMENT

First, I would like to thank my supervisors Prof. Dr. Jochen Dingfelder and Dr. Carlos Marinas. Without their guidance and encouragement, despite their full schedules, this thesis would not have been possible. It was both a pleasure and a great privilege to work in their group on this topic for more than 3 years now. Their knowledge and advise have helped me through all the problems that I encountered during my thesis. Special thanks go to my colleagues in Bonn who also have helped me with technical knowledge and assistance.

Finally, I want to express my deepest gratitude to my family and friends for their continuous support and the emotional stability they have provided throughout the whole time period.

The innermost two layers of the Belle II detector at the KEKB collider in Tsukuba, Japan, will be covered by highly granular DEPFET pixel sensors. The large number of pixels leads to a maximum data rate of 256 Gbps which has to be significantly reduced by the Data Acquisition System. For data reduction, the hit information of the silicon-strip vertex detector surrounding the pixel detector is used to define so-called Regions of Interest in the pixel detector.
In this thesis, the following topics are discussed:

- Pre-processing of silicon-strip data in FPGAs, including:
  - Data acquisition over high-speed Xilinx RocketIOs
  - Heuristic noise filter
  - Cluster engine and coordinate translation
- Online FPGA-based track reconstruction based on Fast Hough Transformation
- Extrapolation of track parameters on pixel detector planes
- Region of Interests creation and transmission protocols

Simulations to study the performance of the developed algorithms are presented as well as results obtained with an FPGA hardware prototype from different test beams.