The new CIP2k z-Vertex Trigger for the H1 Experiment at HERA

Dissertation

zur

Erlangung der naturwissenschaftlichen Doktorwürde
(Dr. sc. nat.)

vorgelegt der
Mathematisch-naturwissenschaftlichen Fakultät
der

Universität Zürich

von

Max Christoph Urban

aus
Deutschland

Promotionskomitee

Prof. Dr. Ulrich Straumann (Vorsitz)
Dr. rer.nat.habil. Eckhard Elsen
Prof. Dr. Peter Truöl
Dr. Stefan Schmitt

Zürich 2004

Übersicht

Von 2000-2002 wurde die **H**adron **E**lektron **R**ing **A**nlage (HERA) am DESY, Hamburg, umgebaut. Durch die Installation von supraleitenden Fokussierungsmagneten direkt in den Detektoren H1 und ZEUS konnte die spezifische Luminosität $\mathcal{L}$ signifikant gesteigert werden. Allerdings ist auch die Rate an unerwünschten Untergrundereignissen gestiegen. Protonen kollidieren mit den Restgasteilchen, induziert durch eine erhöhte Synchrotron Strahlung. Falls diese Kollision im rückwärtigen Detektorbereich stattfindet, erzeugt dies im zentralen Detektor unerwünschte Kollisionsvertices (Untergrundereignisse). Für den H1 Detektor wurde ein **neuer z-vertex Trigger** entwickelt, der zwischen Signaturen von erwünschten Elektron-Proton Kollisionen und unerwünschten Untergrundereignissen unterscheiden kann, indem er entlang der Strahlachse (z-Achse) in einem großen Akzeptanzbereich Teilchenspuren rekonstruiert. Das Trigger System verwendet Kammersignale (Pads) einer fünflagigen Vieldraht Proportional Kammer (MWPC), die eigens für diesen Zweck entwickelt und in den zentralen inneren Bereich des H1 Detektors eingebaut wurde (CIP2k). Insgesamt werden $\approx$8500 Padinformationen ausgelesen, um daraus Spursignaturen von geladenen Teilchen zu rekonstruieren. Dazu werden **F**ield **P**rogrammable **G**ate **A**rrays (FPGAs) verwendet (Altera Apex 20k400), die eine flexible Gestaltung des Triggers ermöglichen.
Anhand von ersten Messungen mit ep-Daten kann die Leistungsfähigkeit des Triggers gezeigt werden.

Abstract

The **H**adron **E**lektron **R**ing **A**nlage (HERA) at DESY, Hamburg, was upgraded from 2000-2002. The installation of superconducting focusing magnets at the detectors H1 and ZEUS led to an increase of the specific luminosity $\mathcal{L}$. But also the number of background events went up. In particular, if collisions of protons with rest gas particles induced by a high level of synchrotron radiation occur in the backward region of the detectors, many particles may reach the central detector parts. To cope with the background, a new z-vertex trigger was developed that is able to separate ep-events from background events by reconstructing signatures of particles along the beam axis (z-axis) in a wide acceptance range.
The trigger uses signals (pads) of a newly developed, five layer multiwire proportional chamber (MWPC), mounted in the central part of the H1 detector (CIP2k). A total of $\approx$8,500 pads are read out and used to reconstruct a z-vertex distribution for each event. **F**ield **P**rogrammable **G**ate **A**rrays (FPGAs, Altera Apex 20k400) are used, guaranteeing a flexible implementation of the trigger.
First measurements with ep-data show the capabilities of the new trigger system.

Contents

List of Figures

Chapter 1

Introduction

Research in particle physics requires high energies to resolve the structure of particles on a sub-atomic level. Particle beams are accelerated and collide with a fixed target or with another accelerated beam. Around the intersection zones large detectors are installed to measure the trajectories, momenta and energies of the particles generated in high-energy collisions. The design of these detectors strongly depends on the type of accelerator they belong to. To analyze the particle interactions (events), detector data are stored electronically. Out of several hundreds of thousands of events produced per second, only a few events are interesting for physics analysis. The selection of these *good* events is done by the trigger system. In most high energy physics detectors, the successful detection of interesting physics events heavily depends on an elaborate trigger system.

A trigger system is necessary for two major reasons:

1. The number of detected events exceeds the readout and data storage capabilities of the detectors involved.

2. Most of the events are not very interesting. A good trigger system is able to separate events with interesting signatures from other reactions.

One of the largest particle detectors currently in operation is the H1-detector installed at the *ep*-collider HERA at DESY, Germany. Electrons are brought in head-on collisions with protons at a center-of-mass energy of $\sqrt{s_{ep}} \approx 320\,\mathrm{GeV}$. Both the collider and the detector were upgraded in the years 2000-2002 (HERA II).

In the course of the upgrade, the high energy physics experiments H1 (and ZEUS) have been equipped with focusing magnets near the interaction point.

As a result of the beam focusing and higher beam currents, the luminosity is approximately five times higher than before the upgrade. However, the number of background events also increased dramatically. Due to the beam focusing near the interaction point, a high level of synchrotron radiation is produced close to the detector. The high amount of synchrotron radiation leads to an increased rest-gas pressure in the beam pipe. In consequence, the number of collisions of protons with rest-gas particles increases. Compared to the number of non *ep*-events at HERA I, the number of such events that reach the inner part of the detector, is ten times higher at HERA II.

To cope with the high rate of background events at HERA II, a trigger system was developed that is able to distinguish between beam background induced events and

true ep-interactions, without causing any interruption of data taking (dead time free). This trigger system is based on a set of five new multi-wire proportional chambers (CIP2k). A total of $\approx$8,500 channels of this chamber are available to find a trigger decision, providing sufficient granularity to resolve the event structure, even if a large number of tracks is present in an event. Moreover, sufficient redundancy is provided to ensure efficient operation in the presence of inevitable chamber defects.

The high number of channels and the necessity for a fast evaluation of a trigger decision (a maximum latency of 2.3μs) requires the use of the latest technologies in the field of electronics development. With the advent of highly integrated field programmable gate arrays (FPGAs) on the market, possibilities to realize a large range of complex algorithms have increased to some extent for high energy physics applications. In Summer 1999, first tests with large FPGAs (Altera Apex 20k) used for the CIP2k trigger system, were performed. Because the complete information of one φ-sector of the new CIP2k chamber can be fitted into only two FPGAs, the development of the trigger algorithm is to a large extent independent from the development of the hardware design, and very flexible. FPGAs are programmed in hardware description languages (HDL), meaning that the behavior of an hardware design can be described in a standardized programming language. Development software is used to fit the described hardware design into the FPGA. After that, the design is operational. If any changes on the design have to be done, this procedure can be repeated.

Another advantage of using FPGAs of the latest generation is the availability of an integrated memory, which is flexibly programmable via the HDL. With the integrated memory, nearly every digital design can be fitted into those FPGAs (e.g. look-up-tables, dual-ported-memory, huge adder and random-accessible-memory applications). In the CIP2k z-vertex trigger, the integrated memory is used to store the chamber signals (pad information) while the trigger algorithm derives a trigger decision. In the case of a positive trigger decision, the data can be read out.

A challenging part of the work presented in this thesis was the use of these FPGAs (as one of the first groups in high energy physics) for the development of the new CIP2k z-vertex trigger. Design requirements, development and maintenance of the trigger system are subjects of this thesis.

- **Chapter 2** gives an overview of the HERA accelerator and the H1 detector environment. Special attention is given to the upgraded components of the detector and the background situation at HERA II.

- In **Chapter 3**, the principle of the H1 trigger system is described in terms of the new z-vertex trigger, which is a central part of the H1 trigger.

- In **Chapter 4**, the fundamental concept of the trigger, building a z-vertex histogram to find a trigger decision, is described. Requirements for the new H1 z-vertex trigger are defined, taking into account the detailed situation after the upgrade of HERA.

- **Chapter 5** describes the new five-layer multiwire proportional chamber that is used by the z-vertex trigger system. The data flow, starting from the detection of charged particles up to the delivery of the digitized pad information, is described.

- In **Chapter 6**, the developed hardware components of the trigger and their connections amongst each other and to H1 components are specified. Hardware comprises all components that are not (re-)programmable.
- In **Chapter 7**, the trigger algorithm that is programmed into the FPGAs, is described. It calculates the z-vertex histogram of track origins along the beam axis. This histogram is analyzed to decide whether the interaction took place close to the nominal interaction point or not.
- In **Chapter 8**, the (re-)programmable part of the new CIP2k z-vertex trigger is described.
- **Chapter 9** gives an overview of the development process. Testing routines (simulations, tests with a pattern generator, etc.) are described.
- In **Chapter 10**, the system performance is analyzed. The development of the CIP2k z-vertex trigger has been successful and its trigger decision is used in most of the physics triggers at the H1 detector.

The thesis is concluded with a short summary.

Chapter 2

HERA and H1

2.1 The HERA Ring at DESY

The Hadronen Elektronen Ring Anlage (*HERA*) is the first electron proton collider worldwide. It was built at the *Deutsches Elektronen Syncrotron* (*DESY*), in Hamburg, Germany and is designed to measure electron proton scattering at center-of-mass energies of up to $\sqrt{s_{ep}} \approx 320\,\mathrm{GeV}$.

The construction of the HERA ring started in 1984. HERA came into operation in May 1992. Two independent machines are installed in a 6.3 km long tunnel, some 15 m below the surface. One is an electron storage ring, capable of accelerating electrons from 12 GeV to 27.6 GeV. The proton machine is loaded with 39 GeV protons, which are accelerated to 920 GeV before being brought into collision with the electrons. The former e^+e^- storage ring *PETRA (Positron Elektron Teilchen Ring Anlage)* serves as pre-accelerator. HERA can be operated with positrons as well as with electrons and in this work the term "electron" is used for both e^- and e^+.

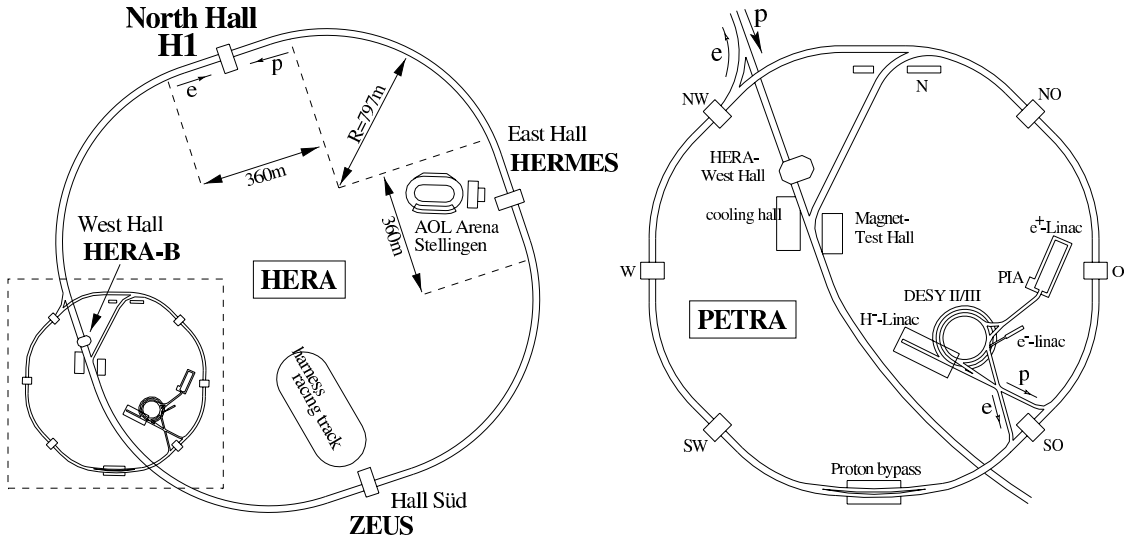


Fig. 2.1: The HERA storage ring with two collider experiments (H1, ZEUS [1]) and two fixed target experiments (Hera-b [2], HERMES [3]). On the right-hand side of the picture an enlarged view of the pre-accelerators DESY II, III, PETRA is shown.

Superconducting technology is used for the dipole bending magnets in the proton ring, because of the high momentum of the protons, while conventional dipole magnets are used in the electron ring. The particles are accelerated by high frequency (*HF*) resonators. For the electron ring, superconducting technology is chosen for some of the resonators. Figure 2.1 gives an overview of HERA and its injector chain. The design proton current for HERA is 100 mA (135 mA after 2002), the electron design current is 50 mA (55 mA after 2002). The particles are packed into a maximum of 220 *bunches*. Every *bunch* contains approximately 10^{10} - 10^{11} particles. The *bunch spacing* is 96 ns corresponding to a *bunch crossing* rate of 10.4 MHz.

Altogether, there are four experiments located around the ring, the two collider experiments H1 and ZEUS and the fixed target experiments HERA-b and HERMES. At H1 and ZEUS, protons and electrons are brought to head-on collisions around a defined interaction point (IP) in the center of the detector.

One of the main goals of the two collider experiments is the measurement of the proton structure at low Bjorken x and at high momentum transfer Q^2. At HERA-b, protons collide with a stationary wire target. Heavy quark decays are studied based on J/ψ triggered events. At HERMES the (polarized) electron beam is used to analyze the scattering behavior with polarized atomic nuclei in a gas target.

Compared to fixed target experiments, the accessible kinematic range of the collider experiments H1 and ZEUS is extended to high Q^2 and very small x. Topics being investigated at HERA are, e.g.:

- A precise measurement of the proton structure function $F_2(x, Q^2)$ [4], measurements of the strong coupling constants and a measurement of the gluon density of the proton [5], [6], [7].

- At very low momentum transfer $Q^2 \simeq 0$ the structure of quasi-real photons is measured [8].

- At very high Q^2 and at high transverse momentum transfer p_t with $Q^2 \approx 0$, searches for exotic particles are conducted. The asymmetric initial state makes HERA a unique place to look for particles having leptonic and hadronic quantum numbers (e. g. "Leptoquarks") [9], [10].

2.2 The Luminosity Upgrade Project for HERA

The specific luminosity of the HERA ring was $0.66 \times 10^{30} \mathrm{cm}^{-2} s^{-1} \mathrm{mA}^{-2}$ in the year 2000 (*HERA I*). To allow collecting significantly more data in the future, an upgrade project of the HERA ring has been carried out from 2000 to 2002 ($HERA\,I \rightarrow HERA\,II$) to increase the luminosity. An improved sensitivity for detecting non standard model physics and the extension of the range of physics experiments to higher Q^2 phenomena compared to HERA I were goals of the upgrade. The luminosity was increased by approximately a factor of five to about $7.4 \times 10^{31}\,\mathrm{cm}^{-2}\mathrm{s}^{-1}$, the expected integrated luminosity is $250\,\mathrm{pb}^{-1}$ per year, compared to an integrated luminosity of $\simeq 100\,\mathrm{pb}^{-1}$ collected by each of the colliding experiments from 1993 to 2000 (see Figure 2.2).

	HERA I		*HERA II*	
	e-beam	p-beam	e-beam	p-beam
beam energy	27.5 GeV	920 GeV	27.5 GeV	920 GeV
nr. of bunches	189/174	180/174	189/174	180/174
particles per bunch	3.5×10^{10}	7.3×10^{10}	4.0×10^{10}	10.3×10^{10}
max. beam current	50 mA	100 mA	55 mA	135 mA
beam dimensions $\sigma_x \times \sigma_y$	$192\,\mu m \times 50\,\mu m$	$189\,\mu m \times 50\,\mu m$	$112\,\mu m \times 30\,\mu m$	$112\,\mu m \times 30\,\mu m$
Luminosity	$1.69 \times 10^{31} cm^{-2} s^{-1}$		$7.57 \times 10^{31} cm^{-2} s^{-1}$	
specific Luminosity	$0.66 \times 10^{30} cm^{-2} s^{-1} mA^{-2}$		$1.82 \times 10^{30} cm^{-2} s^{-1} mA^{-2}$	

Tab. 2.1: Operational parameters of *HERA I* and *HERA II*

The instantaneous luminosity $\mathcal{L}$ is given by:

$$\mathcal{L} = \frac{N_p \times N_e \times \nu}{2\pi \times \sigma_x \times \sigma_y} \tag{2.1}$$

where ν is the frequency of the collisions of the *bunches*, N_e, N_p is the number of e^- and p^+ particles and $\sigma_{x,y}$ the beam cross section. A higher luminosity can be reached by increasing the number of particles N_e, N_p and / or by decreasing the beam cross section $\sigma_{x,y}$. The HERA accelerator was modified at two major points:

- It was equipped with four new super-conducting focusing magnets close to the experiments H1 and ZEUS. At H1, two magnets were installed for focusing the electron beam, one in the forward region (GO) and one in the backward region (GG). Forward and backward direction correspond to the proton direction. Each of them is equipped with an independent cooling system. To create space for these magnets, significant changes to the inner part of the H1 detector were necessary. The vacuum beam pipe in the detector had to be changed. The new beam pipe has an elliptic design. The innermost detectors had to be adapted to the beam pipe geometry.

- Operate HERA with the highest possible beam currents. The goal is to increase the proton current I_p up to 135 mA and the electron beam I_e to $\approx$ 55 mA (see Table 2.1).

Dipole magnets in the new super-conducting focusing magnets are used to steer the electron beam. As a result, a high amount of synchrotron radiation is produced near the experiments that involve the vacuum quality inside the beam pipe and hence leads to an increased number of beam-gas collisions. The new background situation is explained in more detail in Subsection 2.3.2. The *HERA upgrade project* is described in detail in [12].

2.3 The H1 Detector

In the course of the upgrade of HERA the H1 detector was also modified and improved [13]. A detailed description of the pre-upgrade H1 detector can be found in [15, 14]. In the following, the upgrade projects are explained in some detail.

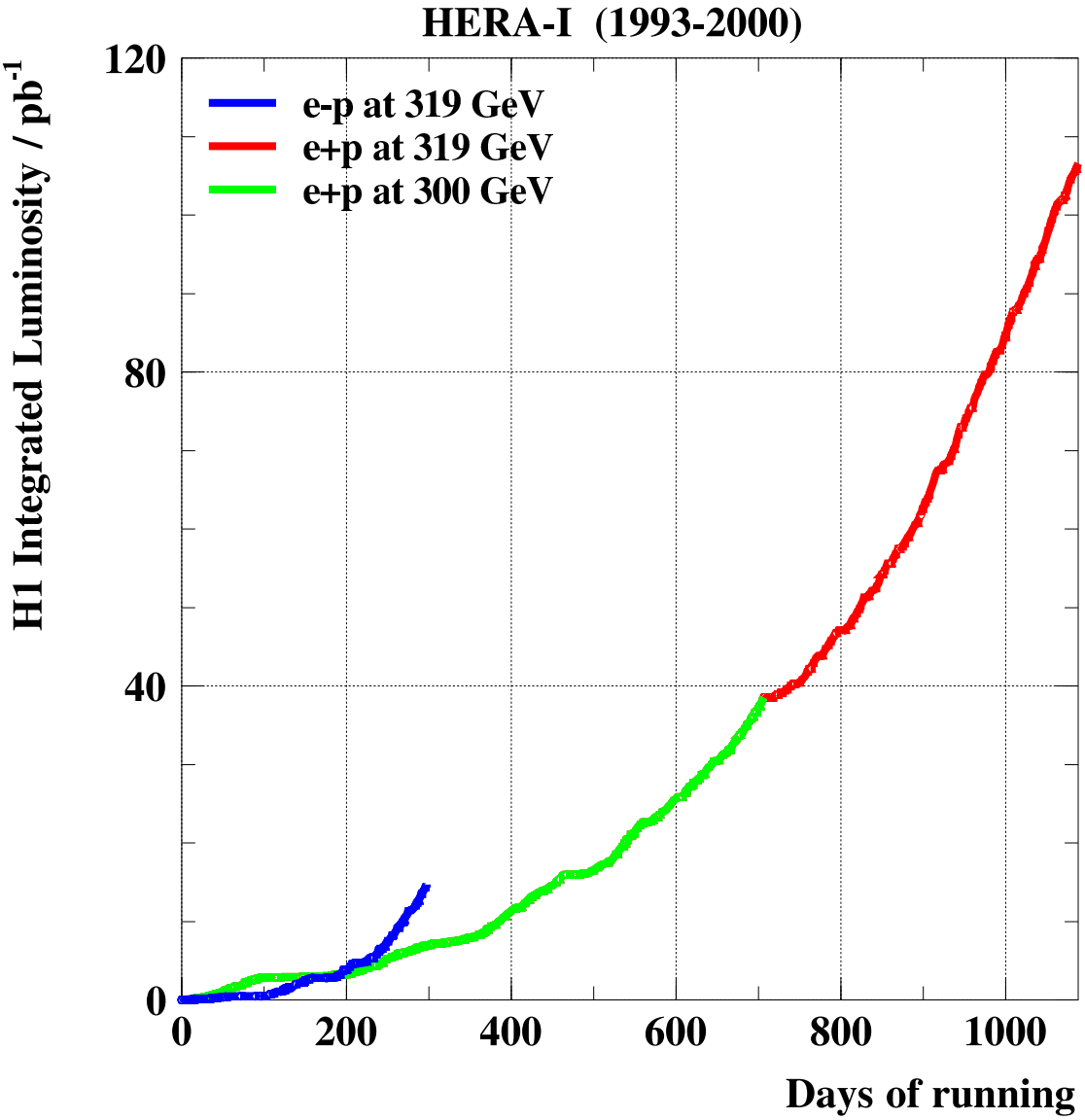


Fig. 2.2: The collected integrated luminosity in the years 1993 to 2000, the complete run period of HERA I [11]. The expected collected luminosity after the upgrade is $250\,\mathrm{pb}^{-1}$ per year.

The H1 coordinate system is defined relative to the HERA ring: The z-axis points along the proton flight direction, the x-axis points towards the center of the ring, the y-axis in the upward direction. The origin of this right handed Cartesian coordinate system is the nominal interaction point of the H1 experiment (see Figure 2.3). Sometimes it is useful to use spherical instead of Cartesian coordinates. The azimuthal angle in the xy-plane is φ. The polar angle is Θ (positive z, $\Theta = 0$). The parts of the detector with positive (negative) z coordinate are often referred to as the forward (backward) regions.

As mentioned in Section 2.2, the H1 detector needed to be changed significantly to create space for additional focusing magnets and the elliptic beam pipe. Parts of the H1 detector had to be removed or redesigned [13].

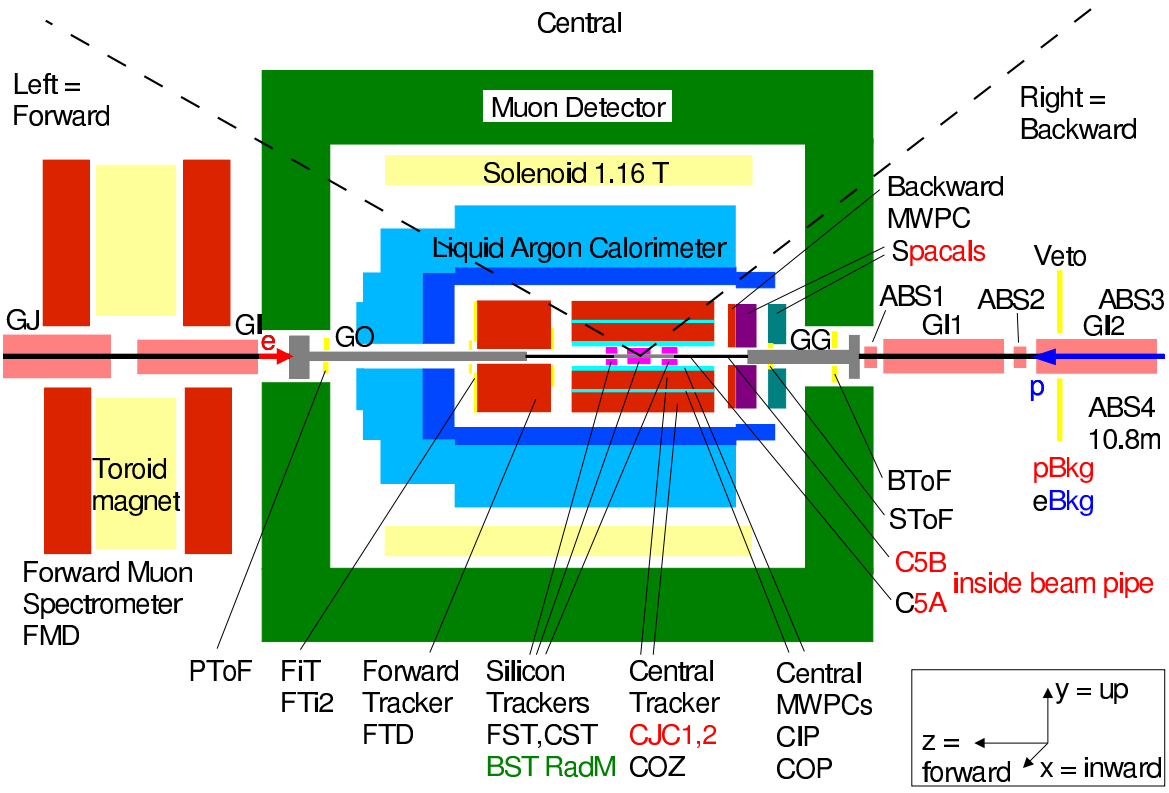


Fig. 2.3: A schematic side view of the H1 experiment, together with the final focusing magnets of the electron machine, which extend into the H1 detector [16].

In Figure 2.3, a schematic side view the upgraded H1 detector is shown. The newly installed superconducting magnets (forward: GO, backward GG) and other components of the beam transport system are visible (absorber ABS1-4, collimators C5A, C5B). The collimators are installed to shield the detector against synchrotron radiation.

In Table 2.4, the most important modifications to the H1 detector are summarized.

2.3.1 Components of the H1 Detector

2.3.1.1 The Silicon Detectors

CST: The innermost detector of H1 is the central silicon tracker (CST) [17]. It consists of two layers of silicon strip detectors, arranged parallel to the beam pipe. The polar angle acceptance range of the CST is $30° < \Theta < 150°$. The intrinsic resolution is $\sigma_z = 12\,\mu$m in z and $\sigma_{r\varphi} = 22\,\mu$m in the $r\varphi$-plane [18]. The CST is used for a precise reconstruction of the interaction vertex. It was adapted to the new beam pipe during the upgrade.

BST: The backward silicon tracker (BST) [19] has an acceptance coverage of $164° < \Theta < 176°$ and a resolution of $\sigma_r = 22\,\mu$m in r-direction. The BST has the silicon planes arranged vertically to the beam pipe. It is used to identify the scattered electron at

Component	Function of Subdetector	upgraded?
	Tracking System	
BST	Backward Silicon Tracker, track recognition	☐ no ■ mod. ☐ new
FST	Forward Silicon Tracker, track recognition	☐ no ☐ mod. ■ new
CST	Central Silicon Tracker, track recognition	☐ no ■ mod. ☐ new
CIP	Central Inner Proportional chamber	☐ no ☐ mod. ■ new
COP	Central Outer Proportional chamber	■ no ☐ mod. ☐ new
CJC 1,2	Central Jet Chambers, track recognition	■ no ☐ mod. ☐ new
FTD	Forward Tracking Device	☐ no ■ mod. ☐ new
BPC	Backward Proportional Chamber	☐ no ☐ mod. ■ new
	Calorimeter	
LAr elm.	LAr, electromagnetic calorimeter	■ no ☐ mod. ☐ new
LAr hadr.	LAr, hadronic calorimeter	■ no ☐ mod. ☐ new
Jet Trigger	triggers Jet Signatures in the Liquid Argon Calorimeter	☐ no ☐ mod. ■ new
SpaCal elm.	Spaghetti Calorimeter, electromagnetic	☐ no ■ mod. ☐ new
SpaCal hadr.	Spaghetti calorimeter, hadronic	☐ no ■ mod. ☐ new
	Magnets	
GO focus.	beam focussing	☐ no ☐ mod. ■ new
GG focus.	beam focussing	☐ no ☐ mod. ■ new
Solenoid	magnetic field	■ no ☐ mod. ☐ new
	Muon Detectors	
CMD	central muon device	■ no ☐ mod. ☐ new
FMD	forward muon device	■ no ☐ mod. ☐ new

Fig. 2.4: Overview of the upgraded, modified and not upgraded subdetector components.

small polar angles. Components of the BST are used in the trigger system (BST PAD trigger). The BST was modified during the upgrade.

FST: The forward silicon tracker (FST) [20] only came into operation after the upgrade. The FST consists of five planes of silicon strip detectors. It improves tracking in the forward direction from $\Theta = 8°$ to $\Theta = 16°$. It has a resolution in r-direction similar to that of the BST ($\sigma_r = 22\,\mu$m).

2.3.1.2 The Proportional Chambers

CIP2k system: The new central inner proportional chamber **CIP2k** is located between the silicon track detector and the inner drift chamber[1]. It replaces the old two layer central inner proportional chamber (CIP) and the central inner z-chamber (CIZ) [21]. Aim of the new trigger system is to suppress background events after the HERA upgrade (see Section 2.3.2 on page 16) and to deliver a precise information about the event timing (t_0-information).

[1]To differentiate between the **old** (replaced) CIP chamber and the **new** CIP chamber, the new chamber is called CIP2k in the following. This convention also applies to the trigger identification: the old trigger is referred to as *old z-vertex trigger* or *CIP z-vertex trigger*, the new one as the *new z-vertex trigger* or the *CIP2k z-vertex trigger*.

The new *multiwire proportional chamber (MWPC)* consists of five cylindrical detector layers with cathode pad readout [22]. The inner radius is $r_{inner} = 15\,\text{cm}$, the outer radius is $r_{outer} = 20\,\text{cm}$. It is segmented into 16 sectors in azimuth (φ). The total active length of the detector is 2.2 m. The size of the pads along the beam axis is about 2 cm. The total number of readout channels is $\approx$8.500, to be compared to 960 channels for the previous CIP chamber.

Multiwire proportional chambers are characterized by a very fast response to ionizing particles. The response time depends critically on the chamber gas and the field strength. A typical value for the intrinsic time resolution is 10 ns [23].

With the replacement of the old CIP, also a new trigger system has been developed [24]. The new trigger algorithm recognizes tracks and fills a histogram with 22 regions (bins) along the z axis. Each bin has a size of 16.4 cm. This allows the detection of particles in a wide range of $16.4\,\text{cm} \times 22 = 360.8\,\text{cm}$ along the z-axis . With this, a reconstruction of the event vertex position is possible. By the use of flexible programmable hardware, the size and position of the z-vertex histogram are adjustable according to the background situation.

COP: The central outer proportional chamber (COP) is a two layer multi wire proportional chamber located between the inner and the outer central jet chamber (CJC1, CJC2). It was used in the old z-vertex trigger to reconstruct the exact vertex in combination with the old CIP in a z-region of $\pm 43.9\,\text{cm}$ around the nominal interaction point (with a z-resolution of $\approx 5.5\,\text{cm}$). Due to the long lever arm, a good suppression of tracks with a low momentum transfer can be achieved (p_t-cut). However, the old z-vertex trigger is not able to actively suppress collisions from the backward region. Due to the high importance of an effective p_t-cut mechanism, it is still used in parallel to the new CIP2k trigger ($\Delta z \approx 20\,\text{cm}$). After the replacement of the old CIP, an interface from the new chamber (signals from layers 0 and 1) to the old z-vertex trigger hardware had to be developed (see Chapter 6, Section 6.5).

2.3.1.3 The Drift Chambers

The drift chambers measure the trajectories of charged particles. The superconducting magnet solenoid, surounding the tracking system, creates a homogeneous magnetic field of about 1.15 Tesla parallel to the z-axis. In this field, charged particles are moving on a helix trajectory. A projection of the trajectory in the xy-plane yields a circle with radius $r \approx \frac{1}{p_t}$, where p_t is the transverse momentum of the particle.

CJC1 and CJC2: The two central drift chambers (CJC1, CJC2, central jet chambers) [25] are the most important components of the H1 tracking system shown in Figure 2.5 (in a xy-view). The inner drift chamber (CJC1) contains 720 wires parallel to the z-axis. They are distributed in 30 azimuthal cells with 24 radial layers each.

The outer drift chamber (CJC2) has 32 radial layers in 60 cells. Each chamber is equipped with field- and potential-wires and cathode strips. The wire signal induced by a charged particle allows to measure the position of hits in the xy-plane with a precision of $\approx 170\,\mu\text{m}$. The z-coordinate of the particle is measured with a resolution

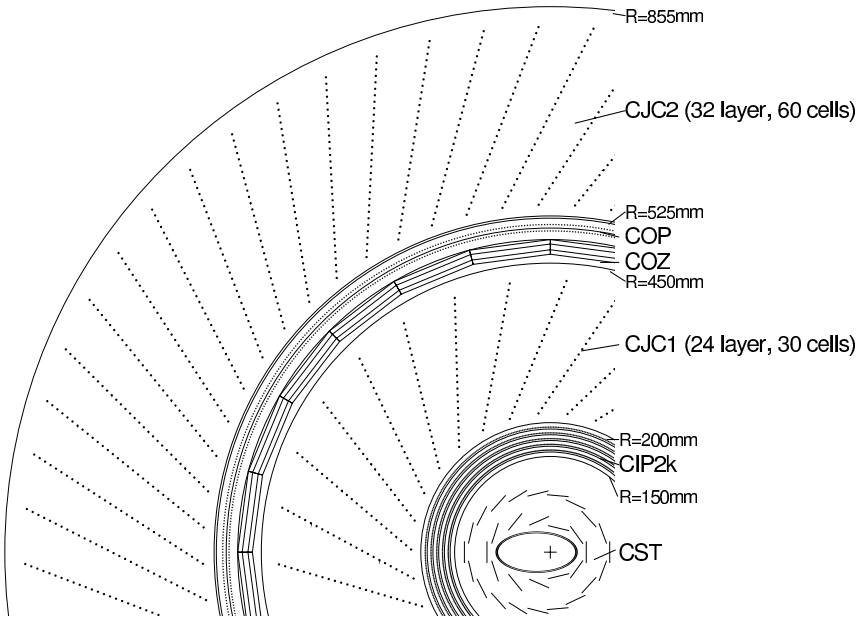


Fig. 2.5: The xy-view of the tracking system of the upgraded H1 detector.

of $\sigma_z \approx 2-3\,\mathrm{cm}$ (1-2% of the length in z) by comparing the collected charge at both ends of the wire [26].

COZ: Between the CJC1 and the CJC2, another drift chamber is located, the central outer z chamber (COZ). The COZ is used to precisely measure the z-coordinate ($\sigma_z \approx 350\,\mu\mathrm{m}$). The wires in the COZ are mounted tangential to the z-axis (in the xy-plane, see Figure 2.5) on wire supporting rods. Between these rods, planar cathodes were introduced. Charged particles drift in z-direction.

The drift chamber trigger: The $DCr\varphi$-trigger system [27] is realized using the drift chamber signals of CJC1 and CJC2. It uses 9 planes from the CJC1 and 3 planes of the CJC2. The drift times measured in these planes are compared to valid patterns for tracks originating from the vertex. Up to 10,000 masks are available. Tracks with a minimum transverse momentum being triggered by the $DCr\varphi$-trigger is $p_t > 400\,\mathrm{MeV}$. The new Fast Track Trigger (FTT) [28], [29] is based on the same input signals. However, making use of modern electronics, it is possible to increase the granularity and the number of masks available, and to include a z-measurement in the new trigger system.

2.3.1.4 The Calorimeters

The Liquid Argon Calorimeter: The Liquid Argon Calorimeter (LAr) [30] is the most important detector for measuring the energies of the final state particles from ep interactions. The main reasons for choosing the liquid argon technique were good stability and ease of calibration, fine granularity for e/π separation and energy flow

measurements as well as homogeneity of response. The Liquid Argon Calorimeter covers an angular range of $5° \leq \Theta \leq 156°$. The calorimeter is housed in a single cryostat inside the superconducting coil surrounding the tracking system.

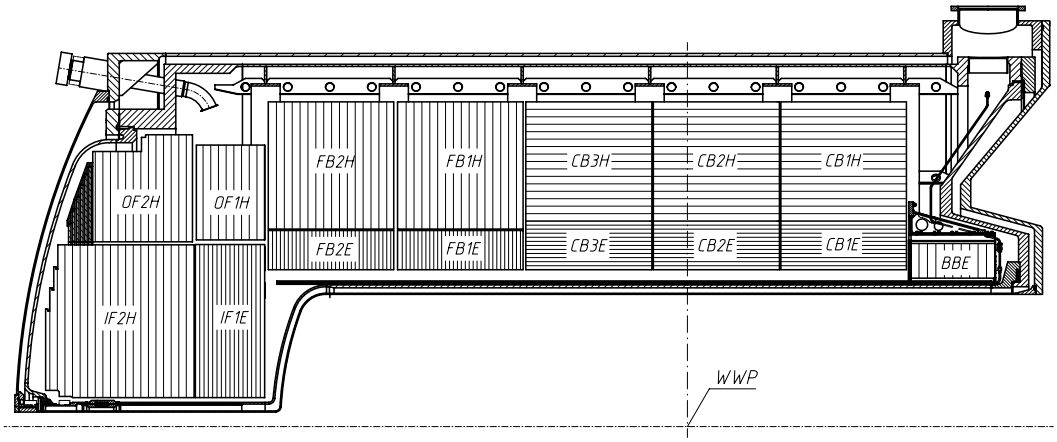


Fig. 2.6: Longitudinal cross-section of the Liquid Argon Calorimeter.

Figure 2.6 shows a longitudinal cross-section of the H1 LAr calorimeter. The calorimeter is segmented along the z-axis into 8 *"wheel"* self-supporting structures (starting from the backward direction: BBE, CB1, CB2, CB3, FB1, FB2, OF, IF). Each wheel is further divided along the azimuthal direction into 8 identical units (octants). From inside to outside, the LAr calorimeter consists of absorber plates interleaved with LAr interspaces. In the inner part (electromagnetic section) lead is used as absorber material, with a total thickness of about 20 (30) radiation lengths in the central (forward) region. In the outer part (hadronic section), steel serves as absorber material. The total hadronic absorption length is about $5\,\lambda$ in the central and $8\,\lambda$ in the forward area.

The energy resolution of the LAr calorimeter for electromagnetic showers is

$$\frac{\Delta E}{E} = 15\%/\sqrt{\frac{E}{GeV}} \oplus 1\%, \tag{2.2}$$

and for hadronic showers (for pions)

$$\frac{\Delta E}{E} = 70\%/\sqrt{\frac{E}{GeV}} \oplus 2\%, \tag{2.3}$$

determined in test beam measurements [31], [32].

The LAr calorimeter consists of 45,000 readout channels in total. For the LAr-trigger system, the channels of the calorimeter are grouped in segments (*Big Towers*). In total, for each octant 14 *Big Towers* are defined over the complete Θ range (see Figure 2.7). Together with the track information from the z-vertex trigger (*Big Rays*), those signals are merged in the big ray distribution box to form track calorimeter coincidences (*validated Big Towers*). This is the main motivation for keeping the old z-vertex trigger in parallel to the CIP2k trigger (see Chapter 4).

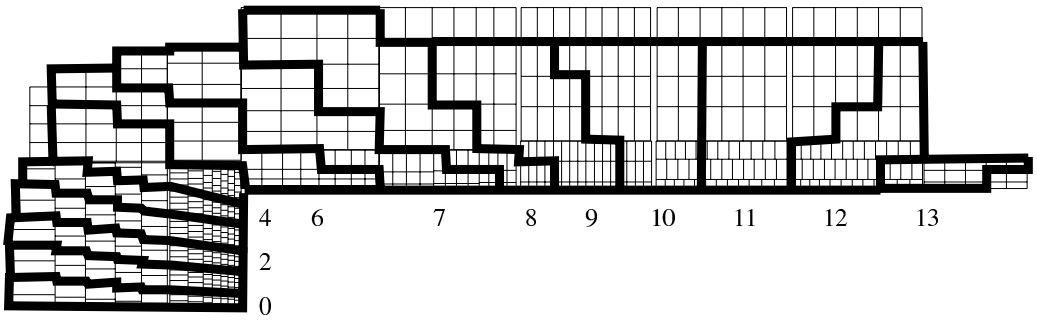


Fig. 2.7: Definition of the *Big Towers* in the Liquid Argon Calorimeter.

The Spaghetti Calorimeter: The Spaghetti Calorimeter (SpaCal) [33] detects the energy of particles that are scattered into the backward region. It covers a range from 155° to 177° in Θ. The SpaCal uses scintillating fibers embedded in a lead absorber. Incident particles develop a shower in the lead. The SpaCal consists of two parts, an inner electromagnetic section and an outer hadronic section. The electromagnetic section is 28 radiation lengths deep. It consists of cells of transverse dimensions $(40.5\,\mathrm{mm})^2$, which ensures a good position resolution. In total 1192 channels are read out. The electromagnetic energy resolution [34] is

$$\frac{\Delta_E}{E} = (7.1 \pm 0.2)\% \sqrt{\frac{E}{GeV}} \oplus (1.0 \pm 0.1)\%. \tag{2.4}$$

The SpaCal signals are read out with an excellent time resolution of 1 ns and a granularity of 3 mm. Events originating from outside the H1 detector are rejected by exploiting the SpaCal timing information at an early state of the trigger.

With the installation of the new GG-magnets, the SpaCal had to be modified. In particular, some modules of the hadronic section were removed. Details of the modifications are described in [35].

2.3.1.5 The Muon System

CMD: The muon system is mounted outside the superconducting magnets. The Central Muon Detector (CMD) is integrated into the iron return yoke. Ten iron layers, each 7.5 cm thick, are instrumented with limited streamer tubes. The detector covers the angular region $4° \leq \Theta \leq 171°$ and allows to detect muons with an energy greater than $\approx 1.5\,\mathrm{GeV}$. The CMD is subdivided into four subdetectors (forward end cap, forward and backward barrel and backward end cap (tail catcher), see Figure 2.8).

FMD: The Forward Muon Detector (FMD). The FMD is situated between 6.4 m and 9.4 m forward of the nominal *ep* interaction vertex and detects muons in a polar angle of $3° \leq \Theta \leq 17°$. It consists of six double layers of drift chambers, three on either side of a toroid magnet providing a field of roughly 1.15 Tesla. Only muons with momenta

of at least 5 GeV will reach and pass through this detector. The FMD is also used to detect highly energetic forward jets [36].

High energy muons from *ep*-interactions are not absorbed in the calorimeters but make tracks in the muon system. Due to the large amount of material in front of and inside the muon detectors, the muons suffer significant multiple scattering, which allows only a rough measurement of the muon momenta in the toroid field of the FMD.

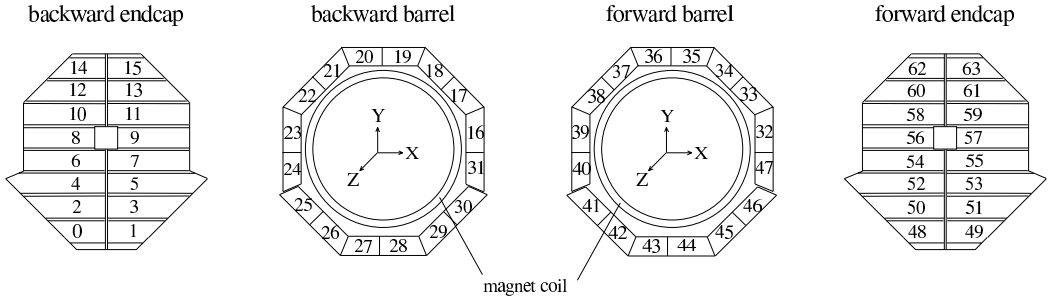


Fig. 2.8: Distribution of the 64 trigger modules for the Central Muon System.

The central muon trigger system groups all muon channels into 64 groups (as shown in Figure 2.8). The forward muon trigger differentiates between tracks passing through one layer of one of the double layer drift chambers and the toroid magnet and tracks that pass through both layers of one double layer drift chamber. Altogether, tracks can occur in 8 φ-sectors that are again segmented in 8 radial areas [37].

2.3.1.6 The Time-of-Flight Detectors

Time-of-Flight detectors are vital for the rejection of non-*ep* interactions. Major sources of such interactions are beam-gas collisions and beam-wall events (see below). Such background events are uncorrelated in time with the desired *ep* interactions.

The Time-of-Flight detectors consist of several large scintillator counters located at various locations around the H1 detector. Non-*ep* background is rejected by measuring the arrival time of particles in the scintillators outside a defined *time window*, defined for *ep*-events.

2.3.1.7 The Luminosity Measurement System

The luminosity determination is performed by detecting bremsstrahlung gammas emitted by electrons in the interaction region. The instantaneous luminosity is calculated by dividing the counting rate of the gammas over a certain energy threshold by the visible *Bethe-Heitler* [39] cross-section [14]:

$$e + p \rightarrow e + p + \gamma. \tag{2.5}$$

The accuracy of this measurement is determined by the precision of the energy measurement of the photon. The luminosity measurement system consists of a photon calorimeter at z=-105 m (photon detector, PD) and electron calorimeters at z=-6 m and z=-40 m (electron tagger, ET) positioned downstream in the direction of the proton beam (see Figure 2.9). The bremsstrahlung cross section is determined from

the count rate of photons in the photon calorimeter in coincidence with the electron taggers.

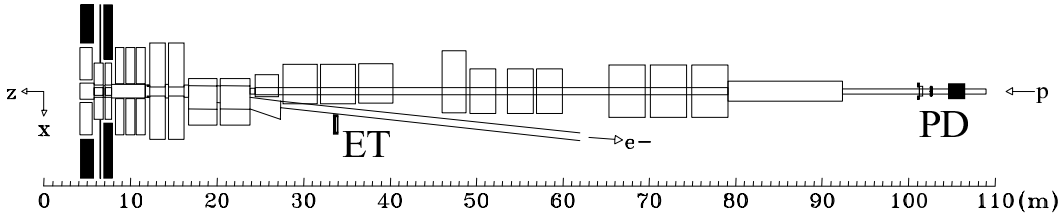


Fig. 2.9: Schematic view of the luminosity system.

In Figure 2.9, a schematic view of the luminosity system with the H1 detector on the left side is shown. During the upgrade, a new photon detector was installed at the H1 luminosity system to cope with the higher synchrotron radiation and the increased luminosity itself. The new photon detector is a Cerenkov sampling calorimeter with tungsten radiators and quartz-fibers as active medium read by photo-multiplier tubes. A detailed description can be found at [40].

2.3.2 Backgrounds after the HERA Upgrade

The HERA upgrade leads on the one hand to an about five times higher luminosity $\mathcal{L}$ ($1\,\mathrm{fb}^{-1}$ expected till 2007). On the other hand also backgrounds have increased significantly. The number of events where protons interact with the residual gas inside the beam pipe or with the beam pipe wall (beam-gas or beam-wall collisions) increased by a factor of about ten compared to HERA I. The high background rate causes the danger of permanent radiation damage of all components of the detector. For this reason, the collider experiments and HERA investigated ways to reduce the backgrounds.

Reasons for a higher background:

1. Synchrotron Radiation: One major improvement of the upgrade was the installation of stronger focusing magnets close to the interaction point. Additionally, these new superconducting magnets bend the electrons with dipole magnets into the proton beam direction and separate them after the nominal interaction point (see Figure 2.10, GO and GG magnets). This early separation is necessary to avoid deflecting the electrons in the proton focusing magnets (GM at $\pm 10.8\,\mathrm{m}$). A new electron beam pipe was installed that adapts the new beam guidance. As a consequence of bending the electrons in the strong focusing dipole magnets, synchrotron radiation is emitted by the (accelerated) electrons. Figure 2.10 shows the new beam pipe geometry and the resulting synchrotron radiation in the region close to the interaction point. The electron beam is steered in such a way that no collimator or beam pipe wall less than 10.8 m from the IP is hit by direct synchrotron radiation. At 10.8 m from the IP, the proton focusing magnet (GM) is installed. There parts of the synchrotron radiation is backscattered into the H1 detector. To avoid hitting the H1 detector with synchrotron radiation collimators are installed, visible in Figure 2.10 (C5A, C5B).

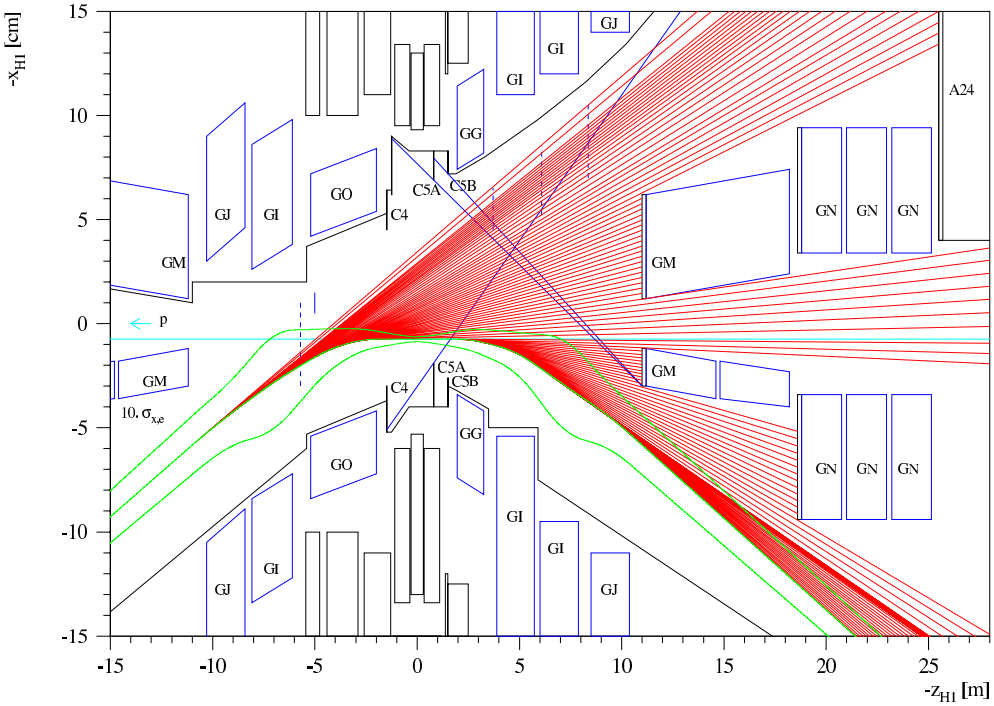


Fig. 2.10: Synchrotron radiation shown as tangential lines in the horizontal plane. Angles with respect to the H1 z-axis are enlarged [16]. The high amount of synchrotron radiation leads to a increased pressure near the absorber.

2. To reduce te amount of backscattered synchrotron radiation, before the GM magnet at 10,8 m, a copper absorber is placed. The high level of synchrotron radiation leads to an increased pressure close to this absorber (see Figure 2.3). High energy photons from the synchrotron radiation evaporate particles from the surface. The pressure of the residual gas in the beam pipe rises with an increasing electron current in the HERA accelerator.

3. As a consequence of the increased number of beam-gas particles (bad vacuum) in the backward region of the H1 detector, the number of collisions of protons with these gas molecules increases (beam-gas interactions).

4. In such collisions between protons and beam-gas, off-momentum particles are produced. These off-momentum particles are generated in the backward region and collide with the collimators C5A and C5B. There, the particles create secondary interaction vertices. These collisions have to be identified as background events.

The particle distribution along the z-axis: In Figure 2.11 the new background situation is visualized. It shows the distribution of the reconstructed z-position of the interaction vertex for a luminosity run. Peaks at the position of the collimators C5A (-80 cm) and C5B (-145 cm) are clearly visible. The relative height of the different peaks depend strongly on the beam conditions and on the trigger settings. In this case, the distribution shows that the number of background events exceeds the number of ep-collisions at the nominal interaction point. The new z-vertex trigger is able

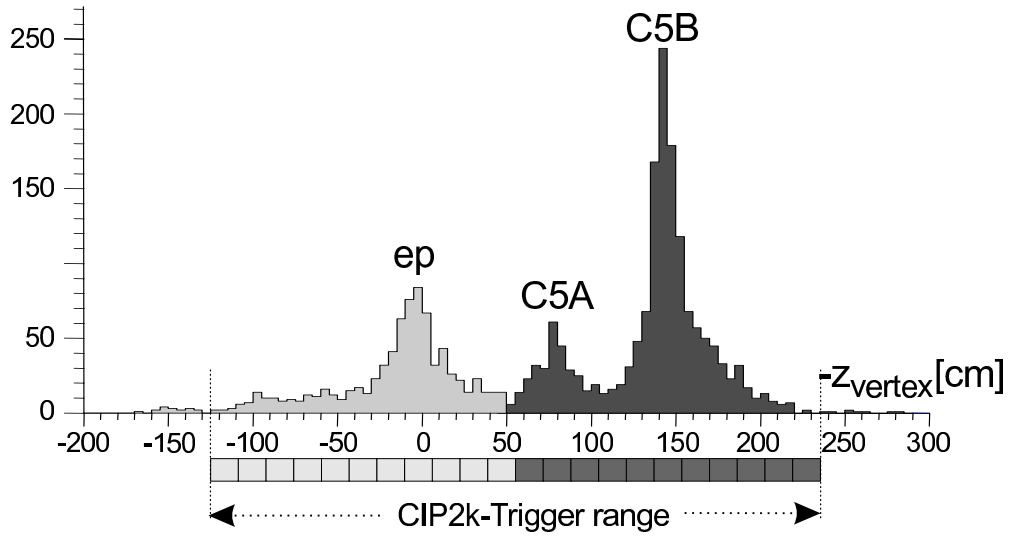


Fig. 2.11: Distribution of the reconstructed z position of the interaction vertex for a luminosity run. Peaks appear at the nominal vertex ($0\,cm$) and at the position of the collimators C5A ($-80\,cm$) and C5B ($-145\,cm$) [16]. The range of the new CIP2k z-vertex trigger is shown below the distribution. A possible background rejection area is shaded in dark grey (a more detailed description can be found in Figure 7.6 on page 78).

to separate background events from ep-events on the first trigger level. The possible detection range is shown in Figure 7.6, where events from the dark shaded region can be suppressed.

To reduce the pressure of the residual gas in the beam pipe, a powerful ion-getter vacuum pump was installed near the collimator C5B. With this pump, the vacuum can be improved by approximately a factor of two. Nevertheless, the HERA II background situation requires a powerful handle to separate background events from ep-events.

A more detailed description of the reasons for, and investigations of the background situation at HERA II can be found elsewhere [16], [41].

Chapter 3

The H1 Trigger System

The aim of the H1 trigger system is to identify good *ep*-events and reject background events [42]. Depending on the beam quality the number of background events is a factor 1000 higher than the rate of good *ep*-interactions. Therefore, carefully selected trigger signatures are defined, matching the requirements of various physics analysis. However, there are basic requirements, that every *ep*-candidate has to fulfill. The collision of electron and proton for example has to take place close to the nominal interaction point (z=0 cm in the H1 coordinate system). For this selection the z-vertex trigger has been developed. It has to be very fast to reduce backgrounds as soon as possible. Other, more elaborate triggers require more time to find a decision. For this reason, the trigger system is divided into different trigger levels. With every trigger level the trigger decision is based on a more precise event reconstruction.

The H1 trigger system is divided into four trigger levels. A schematic diagram of the trigger level (L1 to L4) is shown in Figure 3.1. The first and second level systems (L1

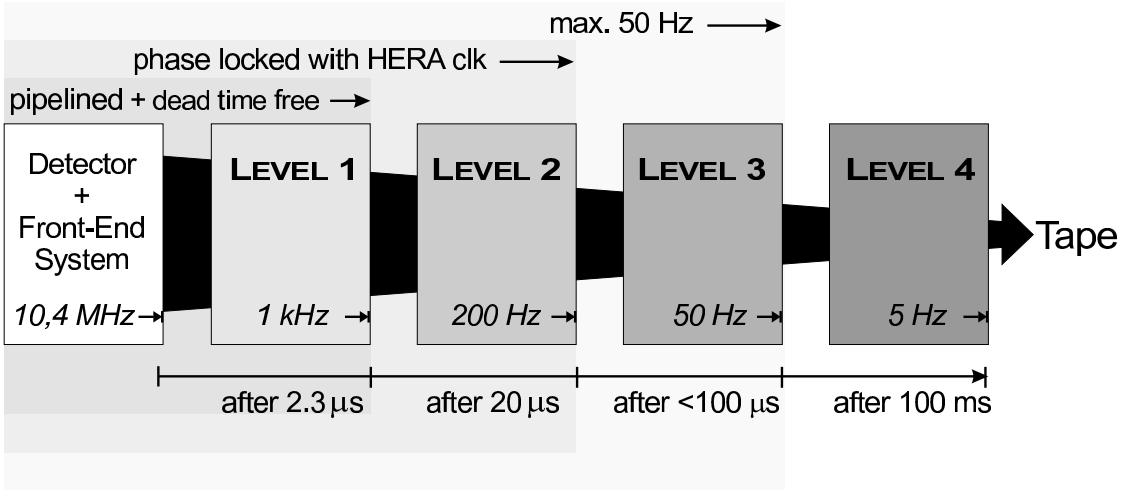


Fig. 3.1: A schematic view of the trigger pipeline. The input and output rates as well as the time limit of every trigger level are illustrated. The *first level trigger (L1, FLT)* must come to a trigger decision within 2.3 μs. It reduces the input rate of 10.4 Mhz to $\approx$1 kHz. Clear *non-ep-events* are rejected in an early stage of the trigger, the data taking continues without causing a dead time.

and L2) are phase-locked to the HERA accelerator clock of 10.4 MHz. The L1 system provides a trigger decision for each bunch crossing without causing a dead time. Dead time free triggers must not interrupt the data taking process and are able to reject most of the bad events. Most of the H1 detector components deliver information for a L1 trigger decision. The L2 trigger does not work dead time free. With a positive L2 trigger, the read out of the detector components begins. The readout can be stopped if L3 triggers reject the event. The first three trigger levels have to reduce the event rate to a maximum of ≈ 50 Hz, which is the maximum input rate for the L4 trigger [43].

3.1 The First Level Trigger (L1)

With a bunch crossing (BC) rate of 10.4 MHz (96 ns period) the *first level trigger (FLT)* is phase-locked to the HERA clock and provides information for a trigger decision for each bunch crossing. Because it is not possible to get a trigger decision within 96 ns, all subdetector data of one event are stored in a pipeline ring buffer. The limited length of this pipeline leads to a maximum latency of 2.3 μs for L1 trigger decisions [44].

Many subdetector systems generate L1 trigger information. The central trigger control (CTC) evaluates the L1 subtrigger decision and builds a H1-wide L1 trigger decision (see Section 3.5). Following a positive L1 decision all front-end pipelines are stopped and the calculation of the L2 trigger decision begins. In Figure 3.2, a schematic overview of a typical H1 first level trigger system is presented. While information from the detector is stored in a cyclic buffer for 32 BCs the subdetector trigger logic evaluates a trigger decision and sends it as a set of trigger elements to the L1 CTC. The CTC evaluates the decision of all subtriggers. In case of a positive trigger decision the pipeline is stopped, indicated by inverting the `!pipe_en`-signal. The `!pipe_en`-signal is distributed to the subsystems via the subsystem trigger control (STC).

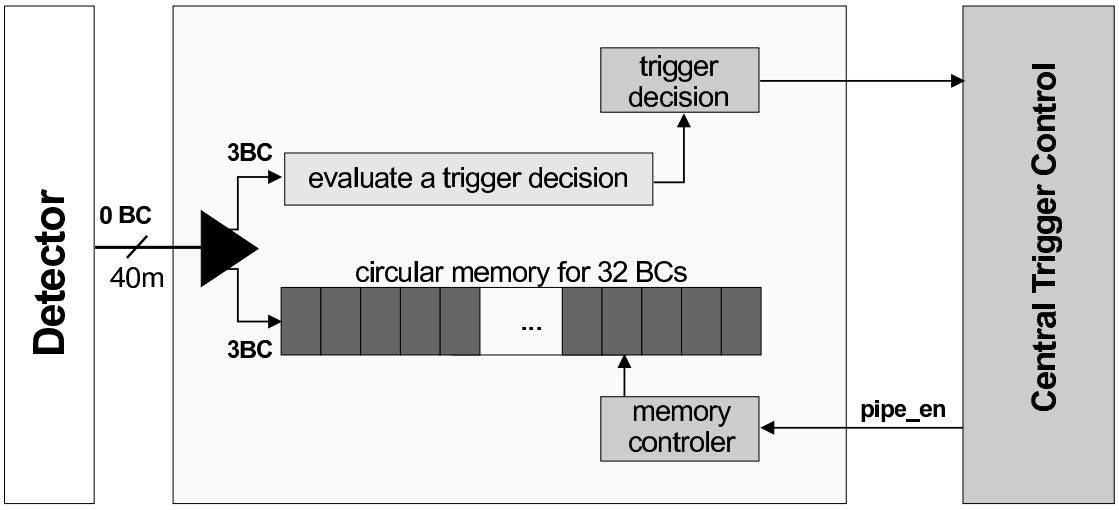


Fig. 3.2: Schematic overview of the CIP trigger system as a typical H1 first level trigger system. While the trigger logic calculates a trigger decision, the information of the detector is stored in a circular buffer for 32 BCs [44].

3.2 The Second Level Trigger

The L1 trigger decision is validated by the *second level trigger* (SLT,L2) [45]. The SLT consists of two independent trigger systems, the *neural net trigger* (L2NN) and the *topological trigger* (L2TT). L2NN and L2TT have to decide within 20μs whether the event should be accepted. Otherwise it will be rejected and the central trigger restarts the pipelines (*L2-reject* signal). The L2 trigger gets L2-information from the subdetectors. Some trigger systems generate dedicated information for the L2 trigger. This information is used to validate or reject the L1 trigger decision.

The results of the L2 trigger are given to the central trigger L2 decision logic (CTL2).

L2NN: This trigger is based on neural networks (NN). The networks have been trained with samples of *ep*- and background events. For the neural network trigger, parallel processors are used. The L2NN trigger is described in detail in [46].

L2TT: The topological trigger (TT) is based on a matrix which represents the geometry of the detector in the (θ, φ) space. The L2 input data are received and are preprocessed into a Boolean *projection* onto the (θ, φ)-matrix. A "distance to background" is calculated to find a trigger decision [47].

Following a L2 decision, the readout of the $\approx 270,000$ channels of the H1 detector components begins. A detailed description of the L2 trigger can be found at [45].

3.3 The Third Level Trigger

The L3 trigger system, based on software algorithms running on RISC processors, refines the L2 trigger decision, potentially causing an abort of the readout if the event is to be rejected (*l3_reject* signal). It calculates invariant masses and topological quantities. The L3 Trigger was not set up prior to the HERA upgrade in 2000. The L3 algorithms will benefit from the new *Fast Track Trigger (FTT)* by reconstructing tracks and looking for resonances and jet-like structures. It is able to detect different structures or topologies within 50 μs [48], [49].

3.4 The H1 Level 4 Filter Farm

For the forth trigger level the complete detector information is transferred to the filter farms at a rate of 50 Hz. Processors reconstruct the event. The online calculation of the trajectory, energy signature and momentum of all particles of the event is used for an event classification. If the signature matches the selection criteria of an *ep*-event all detector information is written to tape resulting in a rate of approximately 5 Hz [50].

3.5 Trigger Elements and Subtrigger

Each subtrigger delivers its trigger information as a set of trigger elements. The CIP2k trigger delivers 16 trigger elements at L1 to the central trigger. In total up to 256 trigger elements are received from the subdetectors at L1. A positive trigger decision is made on the basis of 128 logical combinations of these trigger elements, the subtriggers. Subtriggers are used to find different physics events with the H1 detector. If such an event signature is seen, the subtrigger is activated and the central trigger control stops the pipeline (data taking). This is a positive L1 trigger decision.

The list of the actually defined L1 subtriggers is stored in a file [51], defining the trigger strategy. Due to its importance in discussing the CIP2k trigger system performance, this strategy file is discussed in more detail below.

In the first part of the file, the trigger elements are defined as they are delivered from the subdetectors. Currently, 208 trigger elements of the total of 256 possible channels are used. The first 16 trigger elements are shown below. Trigger elements 8 - 15 in trigger group (TG) b are reserved for the CIP2k trigger system, corresponding to Table 7.1 on page 80, CIP2k TEs 1-8.

```
/*****************************************/
/**  Definitions of Trigger Elements     **/
/*****************************************/

/* ------  TG a -------- LAr T0 ------ TE   0-  7 ------*/

#DEF  LAr_T0           = a t0
#DEF  LAr_electron_1   = a t1
#DEF  LAr_electron_2   = a t2
#DEF  LAr_BT           = a t3
#DEF  LAr_BR           = a t4:2

/* ------  TG b -------- CIP T0 ------ TE   8- 15 ------*/

#DEF  CIP_T0           = b t0
#DEF  CIP_T0_nextbc    = b t1
#DEF  CIP_sig          = b t2:2
#DEF  CIP_mul          = b t4:3
#DEF  CIP_cosmic       = b t7

     ...
```

CIP_mul = b t4:3 means there is a 3-bit word standing at position bit 4 of trigger group b (TG b).

As an example, the subtriggers **s66,s67,s75,s77** are shown as defined in the strategy file, included the CIP2k trigger decision (since November 2003). These subtriggers are designed to trigger events with a high Q^2 or a large missing transverse energy E_t, using the LAr calorimeter. They were used to analyze the trigger decision and efficiency of the operational CIP2k trigger system (see Section 10.2.2):

```
/*****************************************/
/* ep Triggers for 2003 Lumi Run         */
/*****************************************/
```

```
...
s66  LAr_Etmiss>2 && LAr_IF>1              v:2 f:0 t:5 l:0 r:0
s67  LAr_electron_1                        v:3 f:0 t:5 l:0 r:0
...
s69  SPCLe_IET>2 && DCRPh_Ta && !LAr_IF    v:5 t:0 f:2
...
s75  LAr_electron_2&&DCRPh_THig            v:3 f:0 t:0     r:0
...
s77  LAr_Etmiss>1                          v:0 f:0 t:0 l:0 r:0
...
```

The CIP2k trigger is on the one hand used to suppress background events (r:0), on the other hand to deliver a precise timing information for the current subtrigger (t_0, t:0,1,2,3,5). The CIP2k trigger veto condition (r:0 !(CIP_mul==7 && CIP_sig==0)) is described in Chapter 10.2.3. The important definitions, containing the CIP2k trigger, are listed below:

```
/* CIP2k background veto: */
  #GLOBAL r:0 !(CIP_mul==7 && CIP_sig==0)
/* CIP2k T0: */
  #GLOBAL t:0 CIP_T0
  #GLOBAL t:1 CIP_T0||LAr_T0
  #GLOBAL t:2 CIP_T0||DCRPh_T0_VETO_nextbc
  #GLOBAL t:3 CIP_T0||DCRPh_T0_VETO_nextbc||LAr_T0
  #GLOBAL t:5 CIP_T0||(LAr_T0&&!CIP_T0_nextbc)
```

Due to its importance in separating background events from *ep* event candidates, the CIP2k trigger is used since November 2003 in the H1 trigger strategy file as defined above. Studies, which have been performed to test the CIP2k trigger system can be found in Chapter 9 and 10. A list of the actually used subtriggers is presented at [51].

3.6 Trigger Signals

To communicate with the subdetector on the different trigger levels, control signals steer the triggering process. Data taking is only possible during a **data taking run**, set up by the H1 central DAQ system. A run can be started if all subdetectors are initialized and prepared. This is checked before the run in the **run start** sequence. A run start signal is sent to all subdetector systems. The subdetectors now initialize and program all settings for the expected run. After initialization, an acknowledgement signal is sent to the central DAQ and the data taking run starts.

Trigger decisions are now accepted by the central trigger control. In case of a positive L1 trigger, the data taking is stopped (but not the run) and the event is analyzed at a higher trigger level. In Figure 3.3, the timing of the H1 trigger signals and their behavior for different trigger decisions is shown.

The event can be rejected at every trigger level. In case of keeping the event, the trigger decision is expected after a defined time (L1, L2, L3 only). If no reject signal is set within that time, a positive decision (keep) is enforced.

The trigger control signals are distributed via the subsystem trigger control (STC) system. Every sub system has an own STC interface (STC crate). It provides control

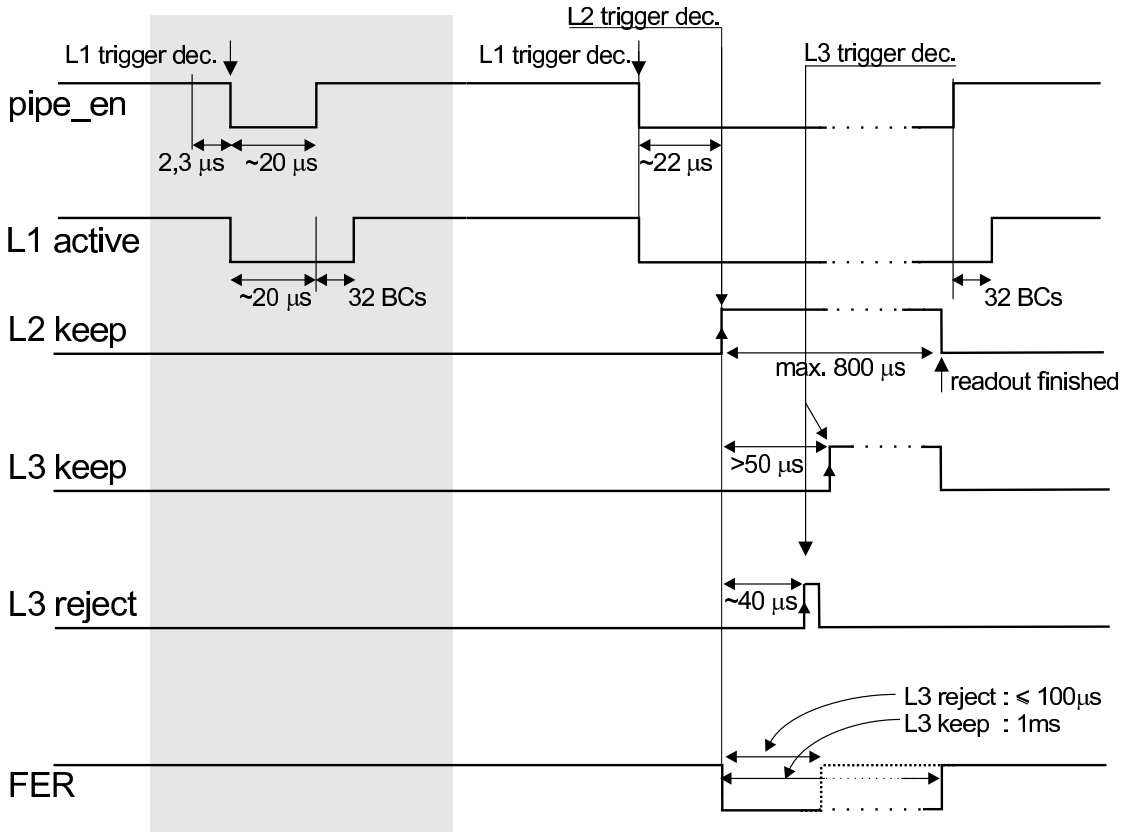


Fig. 3.3: Timing of the H1 trigger signals: In the gray shaded box, a L1 triggered event (2.3 μs) is shown that is not accepted by the L2 trigger. The `l1_active` continues for 32 BCs after the `pipe_en` to clean all pipelines. On the right side, an event that is accepted by the L2 trigger is shown. With the `l2_keep`, the readout of the detector begins. If a `l3_reject` signal turns on after $\approx$40 μs, the readout is stopped, the pipelines cleaned and the run continues. Otherwise, the complete event is read out. This is indicated by the `l3_keep`, coming after a maximum of 50 μs (L3 response time).

signals of the H1 trigger system to the subsystems and manages the communication between the subsystem and the CTC. In case of a positive L1 trigger decision, the STC system inverts the pipe enable signal (pipe_en = 0). All pipelines are stopped and the L2 trigger evaluates a trigger decision, which takes about 22 μs. If the L2 trigger decision is not negative, the readout of the subdetectors begins and the L3 trigger evaluates a trigger decision. In case of a rejection at L3, a L3 reject signal (`l3_reject`) is generated and distributed to the subsystems to avoid deadtime. Every subsystem has to abort its readout as fast as possible. After that, the front end ready signal (`FER`) is set to 1. The pipe enable signal is also set to 1 and the L1 trigger operation is enabled with a delay of 32 BC [52].

3.7 Trigger Strategies after the HERA Upgrade

With the upgrade of the HERA accelerator and the H1 detector, the necessity of improvements of the trigger strategy emerged. Sophisticated trigger strategies are

needed to trigger rare event signatures while keeping the overall trigger rates sufficiently low.

- The luminosity is significantly higher than in the HERA I run period. This was reached with better beam focusing close to the interaction point (see Section 2.2). Problems with the new beam steering and the induced background were discussed.

- To cope with the five fold increased luminosity the trigger system has to be improved. The new fast track trigger is able to reconstruct events at the second and the third level of the trigger system with good precision. Moreover, the trigger rate on an early stage of the trigger has to be reduced significantly. A powerful first level trigger system reduces the number of background events and keeps the dead time as small as possible.

The new CIP2k trigger system was developed to suppress background events very efficiently and deadtime free on the L1 trigger level. Concept, development, testing and performance of the new CIP2k z-vertex trigger are subject of this dissertation work.

Chapter 4

Requirements for a new z-Vertex Trigger

Trigger systems, based on the signals of multi wire proportional chambers (MWPCs) can be found in many high energy physics experiments. Ionization charge released by traversing particles are generating signals on pads/wires in the MWPCs. If the pads in multiple MWPCs are correlated to a predefined track pattern, a track is identified. During the years 1992 to 2000, a z-vertex trigger was used at H1, finding a trigger decision based on the data of six MWPCs. This z-vertex trigger was a powerful tool to determine the IP of events. With the upgrade of HERA and the H1 detector, requirements for a z-vertex trigger for the upgraded H1 detector were discussed taking in account the new background situation as discribed in Section 2.3.2. Before discussing these requirements, the concept of a z-vertex trigger in general is introduced.

4.1 How to Build a z-Vertex Trigger

Charged particles generate a typical signature in the MWPCs, arising from ep-collisions or proton-interactions with residual gas particles in the beam pipe. With the sandwich-like structure of multiple chambers, a reconstruction of the trajectory of the charged particle from single space-points is possible. Combining the pad information of all chambers, a track recognition through the activated pads is possible. This track represents the trajectory of the particle. The back extrapolation of the track onto the beam axis crosses the z-axis at the z-vertex, where the interaction took place. The z-vertex trigger algorithm builds a histogram along the z-axis, counting all tracks and sorting them into histogram bins according to their z-position. The number of bins in the histogram depends on the trigger algorithm.

The reconstruction of the z-vertex position of the interaction is a very powerful tool to separate ep-events from background events.

Concept of the old z-vertex trigger: The trigger algorithm of the z-vertex trigger, used in the first HERA run period (HERA I), is based on the cathode pad signals of three MWPCs: The old CIP chamber with 960 pads, the COP (Central Outer Proportional) chamber with 576 pads and the FPC (Forward Proportional Chamber) with

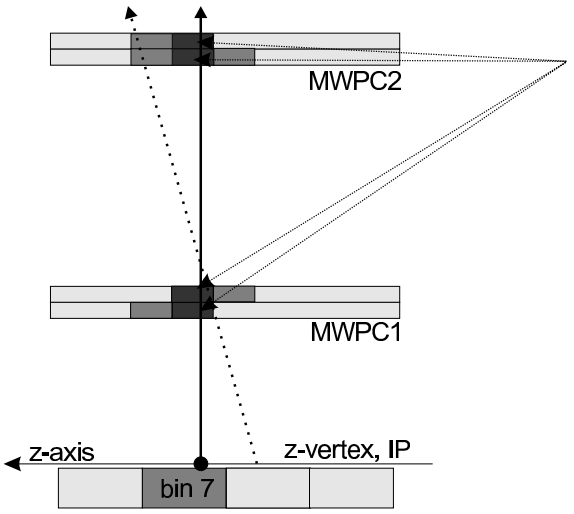


Fig. 4.1:
if marked pads = **on**
$\Rightarrow$ *add one track to bin 7*

In most cases more than one pad per layer is activated, if a charged particle crosses the chamber. Thus, more tracks are added to the same bin or neighboring bins. Nevertheless, the z-vertex of the particle is reconstructed at the right position.

384 pads. FPC and COP both have two independent layers with a 16-fold segmentation in ϕ, the old CIP has two layers with an 8-fold segmentation, but the second layer is arranged with a tilt of 22.5° in ϕ. In total, 1920 cathode pads are used in the trigger electronics for track recognition.

A track (pattern) is defined as the coincidence of four MWPC pads, interconnected by a straight line in the rz plane (see Figure 4.2) in the same ϕ-sector. Two pads (one per layer) have to be active in the CIP, and two pads have to be active in either the COP or the FPC. This is illustrated in Figure 4.2. The extrapolated origin of

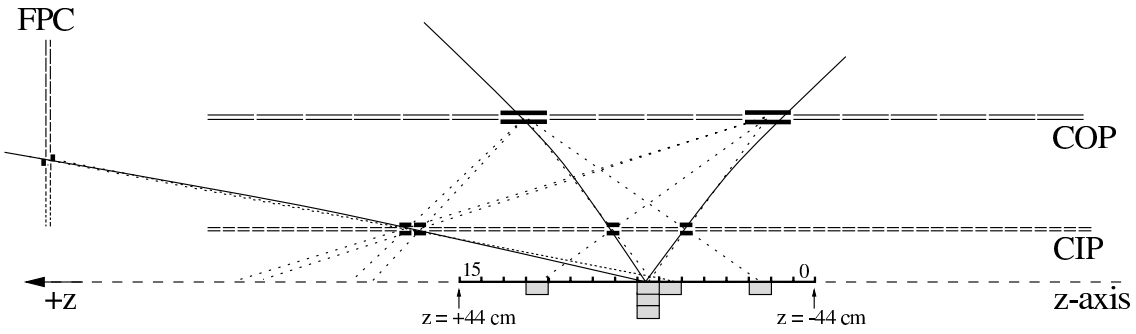


Fig. 4.2: Arrangement of three multi wire proportional chambers (MWPCs): CIP, COP and FPC: A track is defined as the coincidence of four MWPC pads, interconnected by a straight line in the r-z plane. Tracks originating from the same z-vertex position, are sorted into the same bin of a 16 fold z-vertex histogram [53].

the recognized track is the z-vertex position. It is written into a histogram along the z-axis. This *z-vertex histogram* is divided into 16 *bins*. Tracks, originating from the same z-vertex position, are sorted into the same bin. The size of each bin is 54.9 mm, determined by the chamber geometry. The z-vertex histogram is built in two steps for every event.

1. All tracks in one bin, reconstructed by the trigger system, are summed up. The

16-fold ϕ-segmentation of the chambers corresponds to 16 independently built histograms in the trigger, one for each ϕ-segment.

2. The bin information of the 16 histograms is summed up over ϕ, bin by bin. Every bin of the global histogram contains all identified tracks of the corresponding z-position of all ϕ-sectors.

The global histogram covers a range from -44 cm to +44 cm around the nominal interaction point (IP).

After identifying and sorting the tracks into the z-vertex histogram, the histogram is analyzed to evaluate a trigger decision. The peak height of bins pointing to the nominal interaction vertex is compared. Tracks originating from the nominal interaction point are identified as *ep*-tracks if they have a significant peak (higher than a threshold value) at the nominal vertex location in the bins 3 to 14, corresponding to a z-position from -27,5 cm to +33 cm. Events with a vertex outside of the area covered by the z-vertex histogram (background events) are not reconstructed in the pattern algorithm. They do not show this typical peak as they mostly originate from interactions outside the nominal interaction zone.

Besides this fact, pattern combinations are possible that do not point to the nominal interaction point although all tracks originate from this vertex. In figure 4.2, a so-called *fake track geometry* is shown, (dashed line that points to bin 2). This combinatorial background produces only a few entries in the histogram. In many cases, this does not lead to a *wrong* trigger decision. However, the number of those *fake* tracks clearly depends on the noise situation and the number of background tracks. For this reason, the trigger algorithm is susceptible to high multiplicity events and to chamber noise [53], [55].

The old z-vertex trigger was sufficient for the H1-HERA I setup and has delivered a precise event timing (t_0) and a good *ep*-event recognition with a sufficient background rejection in the years 1992 to 2000. However, with increased beam currents and a different background situation due to the HERA II upgrade, a trigger with better rejection power for non *ep*-events was needed.

Hardware Realization: Because the trigger hardware of the old z-vertex is still used (see section 4.4 and Figure 4.4) with the new chamber, a short description of its hardware realization is given here. A more detailed description can be found at [53] and [54]. The trigger hardware reflects the principle of the z-vertex trigger described above. It is implemented in five different types of printed circuit boards (see figure 4.3):

- The Receiver Card receives, shapes and digitizes the analog chamber data. It is stored in readout pipelines and simultaneously sent to the Ray Finder Cards. Every Receiver Card handles 15 analog inputs.

- The Ray Finder Cards search for possible track candidates and fill the 16 z-vertex histograms for the 16 ϕ-sectors. Every Ray Finder Card deals with one bin in one ϕ-sector (256 cards in total).

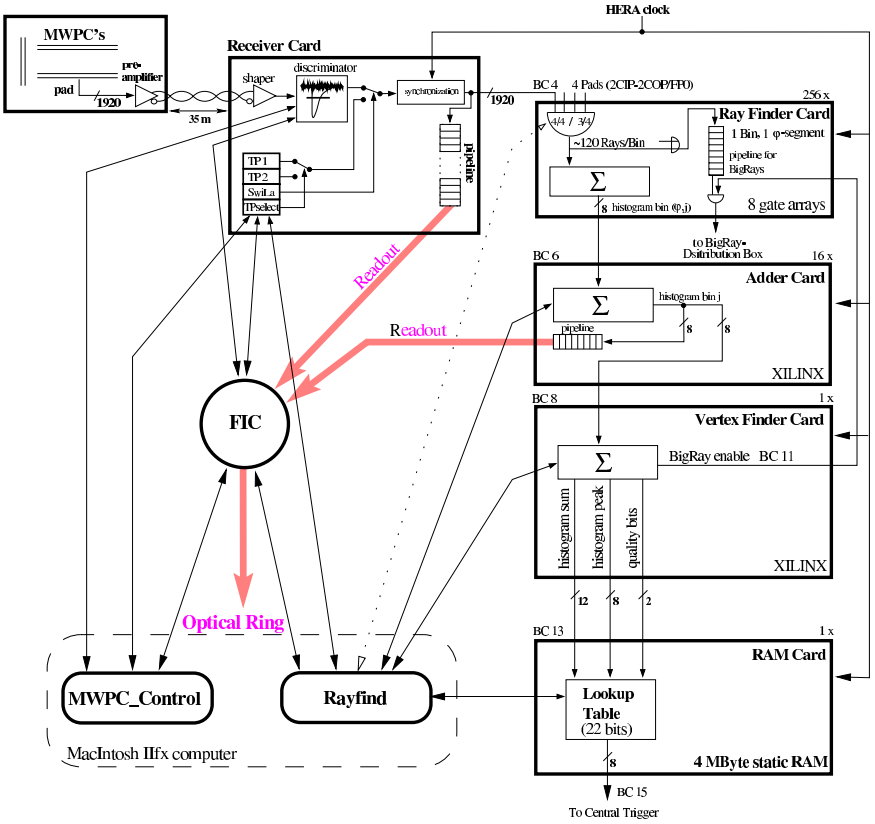


Fig. 4.3: Overview of the Hardware of the old z-vertex trigger [53].

- The Adder Cards (16) add the z-vertex bins of all 16 ϕ-sectors respectively. The result is the global z-vertex histogram.
- The Vertex Finder Card analyzes the histogram and passes this information to the RAM Card.
- The RAM Card evaluates the trigger decision. Possible trigger decisions are stored in look up tables.

A detailed description of the old CIP, COP and FPC based z-vertex trigger can be found in [53].

4.2 Requirements

The successful operation of the old z-vertex trigger in the HERA I phase implied that the trigger system in its general functionality could still be used. However, as discussed in the last section, the old trigger would not be able to reject background originating from outside the chamber region and to separate ep-events with a high multiplicity.

A list of requirements on the z-vertex trigger for the HERA II period is discussed below (as defined in [22]):

- **Background rejection:** Compared to the old z-vertex trigger, an improved background rejection capability is necessary. Induced by a higher amount of synchrotron radiation (new beam guidance) an increased rest-gas pressure in the beam pipe is observed. In consequence, the number of collisions of protons with rest-gas particles increases at the backward direction of the H1 detector. It is necessary to suppress the upstream proton beam background at the first level of the trigger (L1).

- **Bunch timing:** The CIP z-vertex trigger has to determine the correct bunch crossing of the triggered events. This is the minimum requirement for the new CIP2k trigger.

- **Solid angle acceptance** A new system has to cover a range from 15° to 160° around IP, without relying on the FPC, which had to be removed.

- **High radiation level:** The high radiation level was a problem for the central inner drift z-chamber (CIZ) and the central jet chambers (CJC1,2). Since the central silicon tracking device (CST) is operational, the CIZ became obsolete. It had also severely suffered and was therefore removed leaving more space for improvements of the CIP2k chamber.

- **Cut of low momentum particles (p_t-cut):** For topological coincidences track cluster correlations (so called *big rays*) between the LAr trigger and the old z-vertex trigger were built. On the basis of a long lever arm (CIP-COP) a good suppression of tracks with a low momentum is possible (p_t-cut). Thus the LAr trigger is able to set lower threshold limits by using only those trigger towers validated by a CIP-COP coincidence.

To illustrate which requirements were met with the former system, the advantages and disadvantages of the old z-vertex trigger system are summarized below:

$\oplus$ bunch timing t_0 is determined.

$\oplus$ very fast and *good* z-resolution (54,9 mm).

$\oplus$ high p_t-cut due to large lever arm with the COP (big rays).

$\ominus$ very small z-vertex histogram ($\pm 80\,cm$ around IP).

$\ominus$ poor background rejection capability

The capability of reconstructing background events was limited due to the long radial distance of the CIP and the COP. Thus the z-vertex histogram can not cover a range of more than 80 cm around the nominal interaction point in z. Events from outside this region can not be reconstructed accurately, but will produce interfering combinatorial background. Figure 4.4 shows the major differences between the old CIP

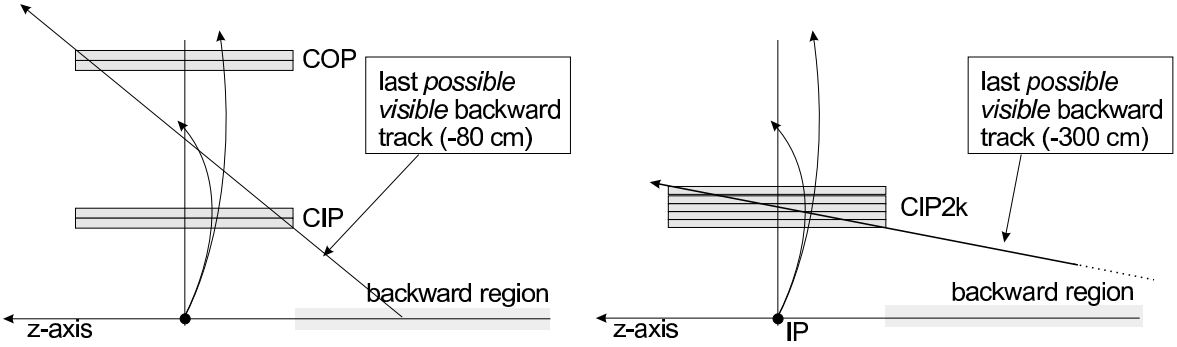


Fig. 4.4: On the left side the old chamber is shown. With the long lever arm of the CIP and the COP, the old z-vertex trigger is able to cut off efficiently low momentum particles having a strongly curved trajectory (p_t-cut). However, for the same reasons, the capability of reconstructing background events is very bad. The new CIP2k (right) is able to reconstruct tracks from particles arising from up to z=-300 cm from the backward region, but an estimation of the particles transverse momentum is not possible.

and the new CIP2k. Modifications of the trigger electronics to enlarge the z-vertex histogram are not possible with the given chamber geometry (CIP, COP and FPC). It was therefore proposed to replace the CIZ and the old CIP with five planes of cylindrical proportional chambers with a two times higher z-granularity ($\Delta z \approx$2 cm). In fact, with a five layer chamber, additional redundancy is given if parts of the chamber become defective by e.g. aging or defective electronics. The FPC was removed to make space for a new improved forward tracking system. A new vertex trigger, using only information from the new five layer chamber alone, would have the same vertex reconstruction efficiency as the old system but with a larger solid angle acceptance and a largely improved background rejection power.

Nevertheless it was desirable to keep the good z-resolution and the possibility to cut off low momentum particles (p_t-cut) for a coincidence with the calorimeter. Hence it was decided to re-use the old trigger electronics (*Big Rays*) with the COP signals and with the signals of the two innermost layers of the new CIP. A precise description of the proposed upgrade plans can be found in [22].

4.3 Studies of a new z-Vertex Trigger

Simulation studies of the expected background rejection and ep recognition power for possibly improved chamber designs were carried out before fixing the final design. The simulation was tuned according to data taken in proton only runs at the HERA machine in 1998. To simulate the performance of the new chamber in a realistic way, a so-called background sample was triggered: It consists of a track from the interaction region using the weakest condition of the old z-vertex trigger (zvtx-t_0) and a minimal energy deposited in the backward calorimeter. These events are compared to typical low p_t physics events with a scattered electron detected in the electron tagger and a jet in the H1 detector [56].

Simulating a five layer chamber: To get information of the capability of the background rejection of a five layer CIP, the distribution of active pads, measured with the old CIP, was used to simulate the response of a five layer chamber. Active pads measured in the old CIP were correlated with tracks measured in the tracking system of H1. A simulation was developed which uses the tracks to calculate the charge deposit of the CIP pads, thus producing the CIP pad information. The simulation reproduced very well the pad response of the chamber, both for physics events and the background sample. However, it was found that a fraction of active pads is not correlated to measured tracks in the tracker system, predominately in the forward and backward region. Looking at those activated pads, they show the same, typical signature as pads that can be assigned to tracks of the tracking system. Therefore, it is assumed that these pads belong to particle tracks with flat polar angles, which are badly measured in the central tracker. Hence, additional flat tracks were added in the simulation. In a last step, the simulation (calculation) of active pads was extrapolated to a chamber with active pads in five layers. In [56] a detailed description of the method and the results of the simulation are presented.

Vertex reconstruction and background suppression in the trigger: Typical proton induced background has a vertex position that is outside of the H1 detector (-80 to -200 cm), whereas the vertex position of physics collisions is at $z = \pm 30$ cm. The reconstruction of the vertex position on the first trigger level may be done with track segments as reconstructed with the new 5-layer CIP.

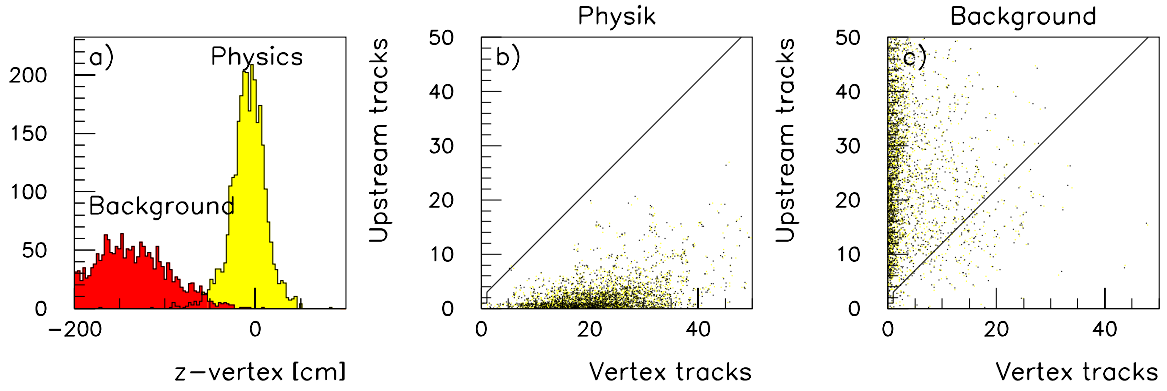


Fig. 4.5: The reconstruction of the z-vertex position (z-vertex histogram) with a five layer CIP is sufficient to separate backward events (dark gray) and physics events (light gray) (Figure a). The identification of background events is possible by comparing the number of upstream tracks ($z < 70\ cm$) to the number of tracks from the interaction region ($z \approx \pm\ 50\ cm$) in the event (Figure b and c), based on the simulated CIP response [56].

The extracted z-vertex distribution for physics and background events (see Figure 4.5a) shows a significant difference between the two event classes. Figure 4.5b and 4.5c clearly show that a determination of the z-vertex position with a five layer CIP not using the COP is possible. An identification of background events is possible by counting the number of upstream tracks ($z < 70\ cm$) and the number of tracks from the interaction region ($z \approx \pm\ 50\ cm$) in the event (Figure 4.5b and 4.5c). In Figure 4.5c a

simple cut illustrates how efficiently background events are rejected. The simulations prove that a five layer CIP chamber meets the requirements of a powerful background rejection in any case.

Optimal Chamber Design: Having shown the separation power of a five layer CIP, the best design has to be evaluated. Especially the *ep*-recognition and the rejection power dependence on noise, number of triggered layers and various trigger algorithms were analyzed with a set of possible designs:

- chamber with 8 φ sectors and 120 z-pads per φ-sector.
- chamber with 16 φ sectors and 120 z-pads per φ-sector.
- chamber with 8 φ sectors and 240 z-pads per φ-sector.

The response of these geometries was analyzed with different physics event classes (low Q^2 events, high Q^2 events) and events taken from the mentioned background sample.

Noise dependence: Defective chamber pads or front end electronics may lead to an increased level of electronic noise. Therefore, simulations have been done with randomly activated pads (simulating noise effects) superimposed to simulated pad information of *ep*- and background tracks to determine the behavior of the chamber at different noise levels in detail. As an example, the dependence of the *ep*-event recognition power (efficiency) and the rejection power for background events as a function of the chamber noise are shown in Figure 4.6(left, efficiency) and (center, background suppression).

Simulations of the electric configuration of the new chamber with the simulation tool SPICE showed that crosstalk and input capacitance of the signal amplifier have a strong effect on the noise behavior (*noise* $\approx$ *input capacitance*) of the trigger efficiency. It turned out that increasing the granularity in z beyond a number of 120 pads per layer leads to large crosstalk effects and hence makes no sense.

Dependence on trigger algorithm: The background detection and recognition depends heavily on the trigger algorithm used. Algorithms that detect clusters and combine them to a possible track compete against algorithms that identify tracks, if all pads belonging to predefined *track masks* are activated ($\simeq$*Bitmuster*). Figure 4.6(right) shows the noise dependence for both types of trigger algorithms. As seen, the importance of designing a well-suited trigger algorithm is evident.

Loss of pad information in several φ-sectors: Because in several φ-sectors single pads or even the complete information of one layer can become unavailable due to chamber defects or front end electronic problems, simulations with a reduced number of pads/layers have been done. In this simulations the efficiency of the trigger algorithm is analyzed, if pads of one or more layers are forced to 0 (off). Several trigger configurations are possible: In a trigger configuration with 4 out of 5 layers (4-out-of-5), a track is identified, if at least pads in 4 layers of a maximum of five layers of the predefined *pad mask* are active. In this case, *defect simulated* pads of one layer are

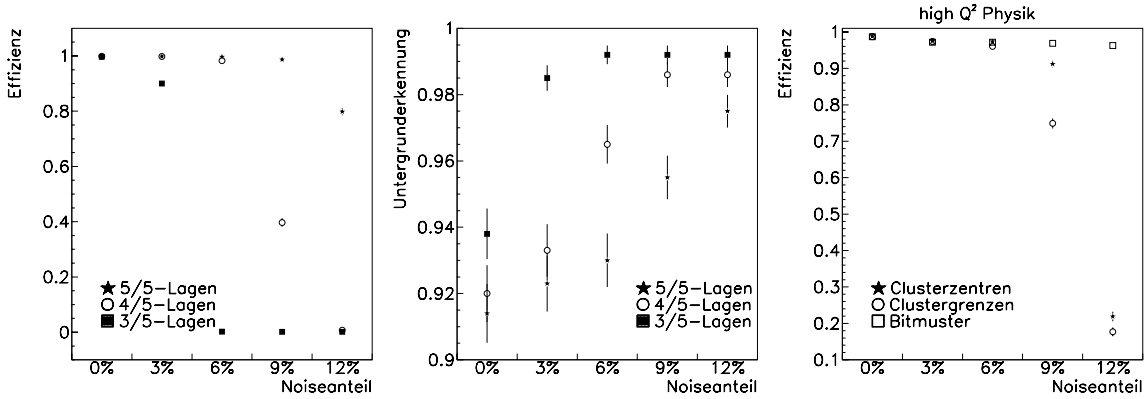


Fig. 4.6: left: noise dependence of the trigger efficiency for *ep* events; center: noise dependence of background suppression power and right: noise dependence of the trigger efficiency for different trigger algorithms [57].

forced to 0. In the same way, the efficiency of the trigger algorithm has been simulated in a 3-out-of-5 and a 2-out-of-3 layer configuration.

A detailed description of all studies can be found elsewhere [57].

4.4 Specification of the new CIP2k System

Following the investigations presented in section 4.3, a new five layer multi wire proportional chamber with 16-fold φ-segmentation and 120 pads in z direction was built. The chamber information of layer 0 and layer 1 is delivered to the old z-vertex trigger. As explained above, the old trigger system is still used for coincidences with the LAr trigger. The fourfold increased granularity of the new chamber ensures a precise track recognition and a powerful background rejection even in events with a high multiplicity. A five layer chamber has a high robustness against single noisy pads or clusters. Moreover, the simulations show, that even after loosing one or two layers of the chamber in some φ-sectors, a good rejection efficiency is still maintained. However, the trigger algorithm has to be flexible in order to be able to operate with less than five layers (4-out-of-5, 3-out-of-5).

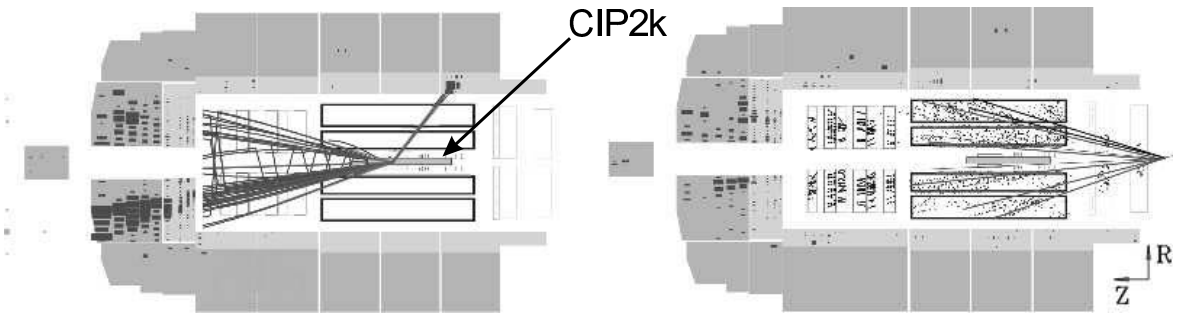


Fig. 4.7: Overview of the H1 detector (event display): (left) typical signatures of a physics event. (right) Proton induced beam gas collisions. The CIP2k trigger system should select physics events while excluding background related events.

The new chamber, readout and optical link system are described in Chapter 5.

The new trigger hardware is described in Chapter 6. The purpose of the new **CIP2k z-vertex trigger system** was proposed to the DESY PRC (Physics Research Committee) as follows [58]:

z-Vertex Trigger: The new CIP2k z-vertex trigger system is designed to determine a trigger decision at the first level of the H1 trigger system. It has to deliver a trigger decision to the central trigger logic within $2.3\,\mu s$ latency. The trigger compares the number of tracks originating close to the nominal interaction point (n_{CTR}) to the number of tracks from further upstream or downstream the nominal interaction point (n_{BWD}, n_{FWD}) in a large z-vertex histogram (over 2 m, compared to 80 cm at the former system). Events from an ep-collision have high n_{CTR}, while background events have high n_{BWD} or n_{FWD}.

Chapter 5

The new CIP2k Chamber

As described in Chapter 4, *Multi Wire Proportional Chambers (MWPC)* are used for a fast detection of charged particles in high energy physics experiments. The general concept of the proportional chamber has been implemented in many variants, such as the new *central inner proportional chamber* with a cathode pad readout.

5.1 Detection of Charged Particles

In high energy physics various detectors are used to record the position, arrival time and identity of *charged* particles. The precise evaluation of the position is required in order to determine the particle's trajectory. Precise timing is often used to associate one particle to others for the same interaction. The processes that lead to particle detection differ for neutral and charged particles. The electromagnetic interaction of charged particles allows a measurement of the particle trajectories. Photons interact in different ways depending on their energy (via photo effect, Compton scattering and pair production) producing charged particles which then can be measured directly. Neutrons produce charged secondary particles via strong interactions with nucleons, hence only their energy can be measured. Neutrinos can only be observed through their weak interaction with electrons and nucleons, therefore they usually escape detection.

The Proportional Counter: Charged particles were first detected with a proportional counter (1908, Geiger, Rutherford [59]). Single counters consist of a cylindrical metal or glass tube of radius r_2 at negative potential, with a thin central anode wire of radius r_1 at positive potential. The tube is filled with gas. The electric field in the gas for a potential difference V_0 is given by

$$E(r) = \frac{V_0}{r \cdot ln(r_2/r_1)}. \tag{5.1}$$

An electron freed by ionization at radius r_a drifts towards the anode and gains an energy

$$T = e \int_{r_b}^{r_a} E(r) dr \tag{5.2}$$

at radius r_b. If T exceeds the ionization energy of the gas, more gas molecules are ionized. A chain of such processes leads to an avalanche of electrons and positive

ions. For sufficiently high voltages the avalanche fully discharges the detector. This is used in the well known *Geiger-Mueller-counter*. For lower voltages the anode signal is *proportional* to the initial ionization with a gas amplification factor of typically 10^4 to 10^5. The signal only depends on the number of primary ions.

The Multi Wire Proportional Chamber: The most significant progress in this field was the introduction of the *multiwire proportional chamber (MWPC)* by Georges Charpak [60] in 1968, using the proportional range of the proportional counter, but with many (≈ 100) wires. With the multiwire chamber, it became possible to determine the tracks of charged particles with good precision. However, the main improvement of Charpak's multiwire proportional chamber was the enormous increase of the data-taking rate. Every single wire in the multiwire chamber acts as a detector, which can detect thousands of particles per second. This made it possible to study very rare processes in particle physics research.

In every MWPC, many parallel anode wires are stretched in a plane between two cathode planes (see Figure 5.1). A typical structure (here: the new CIP chamber) has

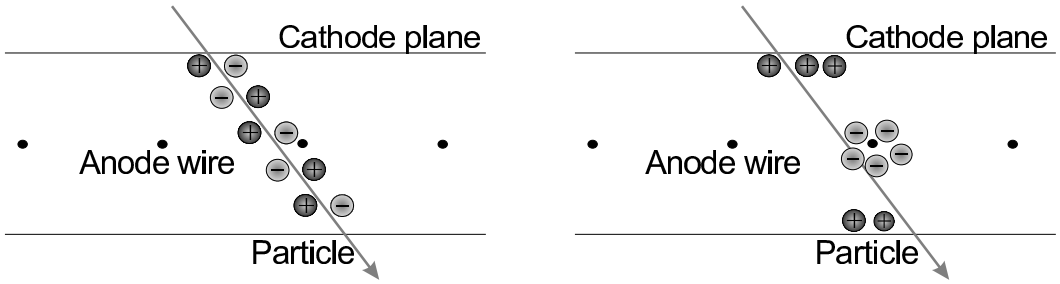


Fig. 5.1: Idea of a multiwire proportional chamber: Charged particles interact with the gas particles and produce positive ions and electrons. With a anode electrons are accelerated to the anode wire and positive ions drift to the cathode (with avalanches starting at 50 μm from the wire).

wires of 25 μm diameter with 2 mm spacing, between cathode planes 6 mm apart. It contains a *argon-isobutane* gas mixture and is operated at a potential difference of some kV. Charged particles interact with the gas particles and produce positive ions and electrons. In the electric field electrons are accelerated to the anode wires and positive ions are drifting to the cathode plane. By this, further electrons are freed that cause avalanches near the wire with a powerful gas amplification giving rise to negative pulses with a very fast rise time (≈ 0.1 ns). The positive ions have a much lower mobility and induce pulses on both the cathode and the neighboring anode wires of $20 - 30$ ns duration.

From the standard MWPC, a lot of other familiar chamber types are derived such as the drift chamber (e.g. the CJC and the COZ chambers at the H1 detector) or the central inner (outer) proportional chamber (CIP2k, COP).

5.2 The new CIP2k Chamber

The new CIP2k chamber is designed as a five layer multiwire proportional chamber with cathode pad readout.

Principle of the chamber: In Figure 5.2 a side view of the CIP2k chamber is shown in the rz-plane. A strong electric field of $\approx$2 kV is applied between wires and ground. On the inner side, the cathode plane is made of aluminum. On the outer side, a capton layer, coated with carbon, is used as cathode. The carbon has a finite resistivity (550-600 kΩm), thus the positive charge does not immediately discharge and an electric potential between ground and the location of the charge accumulates. On the lower right side of Figure 5.2, the equivalent circuit diagram of the cathode pad readout is shown. The signal of a charged particle is separated into negative charge at the anode wire (negative pulse form) and a positive induced charge on the carbon ground plane. A capacitor, consisting of the carbon layer and the readout pad with capton-foil as dielectric, transports the charge to the readout electronics. The induced charge ($\mathcal{O}(10^5)$ electrons) of every single pad is fed into a charge-sensitive amplifier and then to a discriminator for digitalization [61].

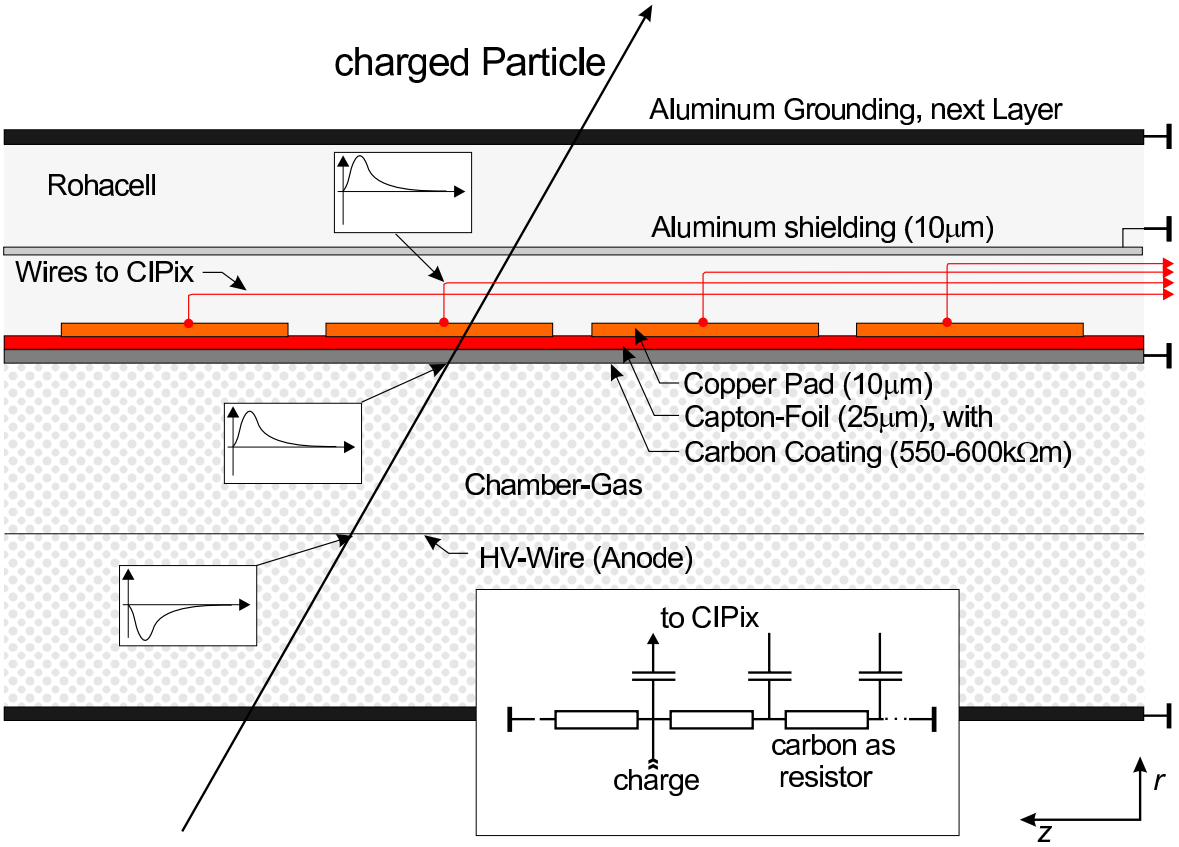


Fig. 5.2: Side view of the CIP2k chamber in the rz-plane: The charged particle deposits charge, that cannot discharge immediately due to the high resistance of the carbon coating. Thus, a current is induced on the cathode pad near the accumulation of the charge [61], [62].

The cathode pad readout leads to a precise and fast detection of the position of the

particle.

Development of a test system: Just like the former CIP chamber, the new chamber was developed and built at the mechanics workshop of the University of Zürich [63]. Before designing the final chamber, a test system was built. It included a test chamber with readout electronics and thus allowed to test the complete chain from the detection of the particle, the amplification in the new readout chip to the transportation of the information to the trigger electronics via the optical link system. Experience was collected on the cross talk of pads in other layers, the size of the deposited charge on a pad or the amplifier settings. Moreover, the connection of the chamber with the readout electronics was tested. A description of the test system can be found in [66], [67].

Design of the chamber: The final CIP2k chamber was built between 1999 and 2000. It has five radial layers with 480 high voltage wires each. The chamber is operated at a high voltage of ≈2250–2500 V[1], depending on the radius[2] of the layer. The cathode of each layer is segmented into cathode pads, 8480 pads in total. Each pad covers 22,5 degrees in azimuth. The pad length along the z axis is different for each of the five layers, as summarized in Tab. 5.1. As the pad size of the pads in every layer is increasing, the number of pads is decreasing. The innermost layer is defined as layer 0. It has 1904 pads, 119 per φ-sector. The second layer (layer 1) has 1792 pads, the third (layer 2) 1696 pads, the fourth (layer 3) 1584 pads and the fifth layer (layer 4) 1488 pads. The CIP2k is positioned between the central silicon detector (CST) and central

Layer	Radius[mm]	Pad length[mm]	Number of Pads
0	157	18.250	119
1	166	19.322	112
2	175	20.531	106
3	184	21.900	99
4	193	23.464	93

Tab. 5.1: Radial location and length along the z-axis of the anode pads. The last pad in layer 0 ($\hat{=}$ 119), layer 2 ($\hat{=}$ 106) and layer 4 ($\hat{=}$ 93) have only half the pad size.

drift chamber (CJC) and ranges from z = -1127 mm to z = +1043 mm along the z-axis symmetrical around the nominal interaction point. The total length of the active area is 2170 mm. The angular acceptance for particles from the nominal interaction region ranges from 11° to 169° in the θ direction. The pads of layer 0 are placed at a radius of 157 mm. Every layer has a thickness of 9 mm (see Table 5.1).

Thus a rough estimation of the geometric z-resolution R(z) of the chamber is possible:

$$R(z) = \frac{\text{padlength} \cdot \text{radius}_{L2}}{\text{distance}(\text{radius}_{L4} - \text{radius}_{L2})} \approx 199,6\,\text{mm} \tag{5.3}$$

[1] A determination of the efficiency dependence on the applied HV is presented in Chapter 10. The presented values are the result of these investigations.

[2] With increasing radius and equal number of wires, the electric field gets stronger. This is balanced by a lower HV in the outer layers.

The final z-resolution depends on the trigger algorithm, the overlap of the pads, the crosstalk effects and the number of active neighboring pads.

A technical drawing of the new five layer CIP2k chamber is shown in Figure 5.3 in the rz-plane of the H1 detector. Dashed lines indicate a cut in the drawing, but the scales are proportional to the dimensions of the real chamber. The five pad-layers were

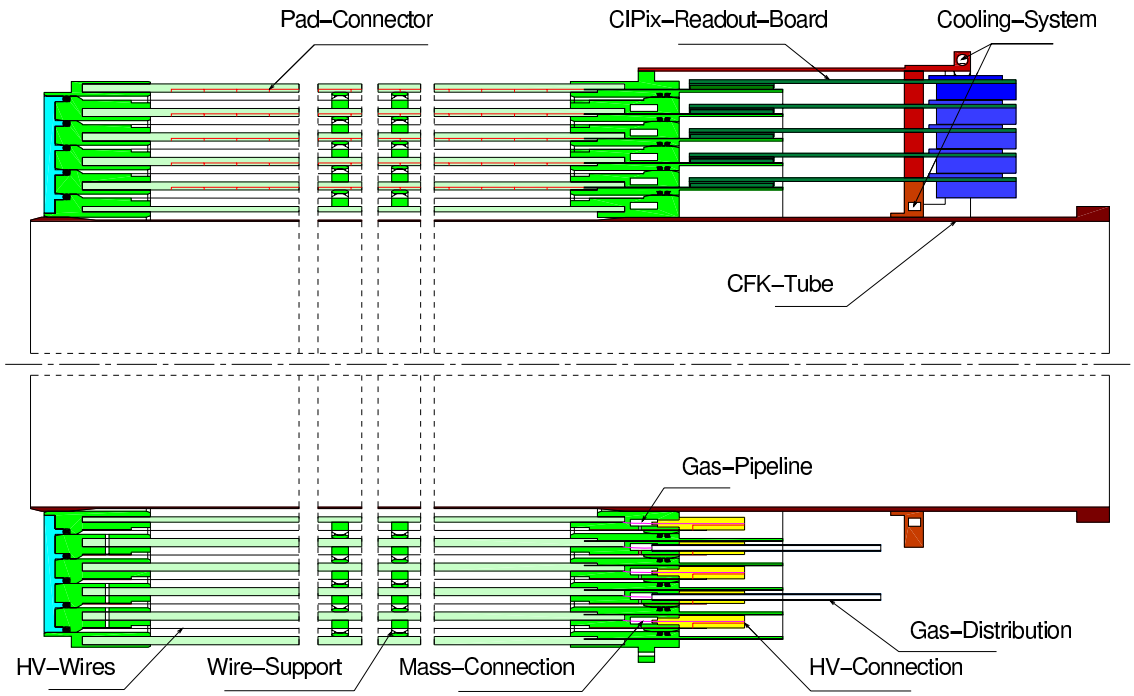


Fig. 5.3: Technical drawing of the new five layer CIP2k chamber. On the right side the front end readout boards are mounted. Every pad is directly connected to the readout board with a coaxial cable. The cooling system for the CIPix-chip and the optical link senders is also mounted on the right side [62].

constructed sandwich-like, each layer in the same way, as outlined in Figure 5.2. The innermost plane is an aluminum plane, followed by the gas volume, which contains the HV wires. The upper cathode consists of the carbon coating and the copper pads, covered with Rohacell. Rohacell has a very low radiation length X_0 but is nevertheless mechanically very stable. A coaxial cable is soldered directly onto every cathode pad and transfers the pad charge to the chamber connector. 16 chamber connectors per layer are mounted onto the chamber. Each connector contains 120 pad connections corresponding to up to 120 pads in z-direction. After the Rohacell plane, the aluminum plane of the next layer follows.

On the right side of the chamber ($-z$ position), the front end readout boards with the readout chips are mounted onto the chamber connector. 16 boards (8 double boards, see Section 5.4) are used per layer, 80 (40) in total. Each board covers 1 (2) *phi*-sector(s). The cooling system for the boards, containing readout chips and optical hybrid, is also mounted on the right side. Copper blocks with a direct contact to the readout boards are connected to an outer and an inner cooling ring above and below the readout electronics. The latter are water-cooled. The cooling of the boards is

critical, because without it the generated heat would destroy the bond-wires, which connect the readout ASIC to the board. Moreover the inner connection vias of the 12 layer printed circuit board (PCB) can suffer.

For reasons of mechanical stability, the chamber has an additional wire support construction after one third and two thirds of the chamber in z-direction to avoid oscillations of the wires (see wire-support in Figure 5.3).

A more comprehensive description of the CIP2k chamber and its development can be found in [64].

The gas system: The chamber is operated with a gas mixture of argon (49,9%), isobutane(49,9%) and freon (0,2%). It is controlled by the H1 wide gas control system [68]. Two gas circuits are used for the CIP2k chamber, CIP_inner for layers 0,1 and CIP_outer for layers 2,3,4.

The HV system: The high voltage is controlled by C.A.E.N. units (SY127, 40 Channel High Voltage System [69]) which automatically switch the chamber off if the current limit is exceeded. The 480 HV wires of each layer are interconnected with 1 MΩ resistors in a ring structure. HV supplies are connected to every 15th HV wire. A total of 32 channels are used per layer. Hence, a group of 15 wires has to be switched off with a short circuit of one wire. Neighboring groups have to be degraded. A detailed description of the gas- and HV-system can be found elsewhere [68].

5.3 Geometry of the Chamber Pads

The new chamber was built using a special arrangement of the pads. The relation between the distance to the beam axis and the pad size which increases with the layer number, leads to a special geometry of the new chamber, called *projective geometry*. This special choice of geometry allows to perform the track recognition independently of z and θ (in H1 coordinates) [65].

The pads are arranged such that tracks from the same vertex generate similar patterns independent of θ. By recognizing these patterns, the vertex of the particle is determined. Every 328.50 mm the pads of the five layers are exactly aligned above each other. This defines the smallest unit that represents the entire projective structure, the *standard block* of the projective geometry (SBPG), shown in Figure 5.4. A *standard block* consists of *18 pads* in layer 0, *17 pads* in layer 1, *16 pads* in layer 2, *15 pads* in layer 3 and *14 pads* in layer 4. In total $\approx$ *6,6 geometry standard blocks* are used for the whole chamber ($6.6_{SBPG} \cdot 18_{pads\ in\ layer0} = 118.8 \rightarrow 119$ pads). To illustrate the projective geometry, three tracks are shown in Figure 5.4. The intersection of these tracks is at z_0 of the beam axis. All tracks originating from z_0 have the same pad number in all layers plus an unique offset for each layer depending on z_0. The right track is identified if all pads with the pad number 0 in layers 0 to 4 are active, the track in the middle can be identified by pads 2 in layers 0 to 4 (offset 2) and the left track by all pads 5 in every layer (offset 5). Tracks from different z-positions always have the same relation of pads in each layer. This illustrates the idea of a projective chamber: **Every track from the z-position z_x is identified by an unique track pattern x plus a constant**

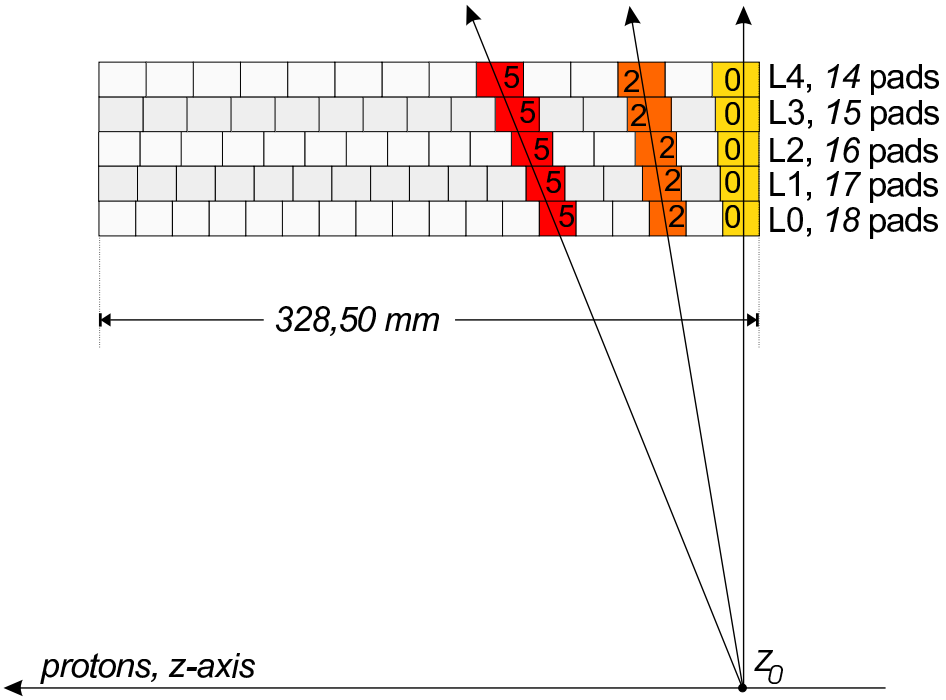


Fig. 5.4: Standard block of the *projective geometry*: Every 328.50 mm the pads of the five layers are aligned exactly above each other.

offset d, applied to all layers. The track pattern x may be characterized by the offsets of layer 0,1,2,3,4 relative to layer 2. The constant offset is the pad number in layer 2.

The development of the trigger algorithm with the projective chamber is described in Section 7.1 of Chapter 7.

5.4 Front End Electronics

The high number of pads of the new chamber and the limited space for support mechanics and the front end electronics implies sophisticated front end readout electronics. In total $16 \times 5 = 80$ front end readout boards with support structure and cooling, low and high voltage and gas support lines have to share the 130 mm of space between the CIP2k chamber and the backward end flange. In Figure 5.5 the backward region of the chamber is shown in a technical 3d-drawing.

Highly integrated readout boards were developed at the Paul Scherrer Institute (PSI), Switzerland [70]. These boards contain a specially developed readout ASIC (Application Specific Integrated Circuit, the CIPix[3]) and moreover 17-to-1 multiplexer and optical components to transfer the chamber information to the trigger system 40 m away in the H1 electronics trailer (see Chapter 6). All components are mounted on two $5 \times 13\,\mathrm{cm}^2$ eight-layer PCBs. Each PCB contains two CIPix and two multiplexer units. In addition one of the PCBs holds an optical transmitter unit. The two boards are

[3]Therefore the frontend readout boards are usually called CIPix-boards

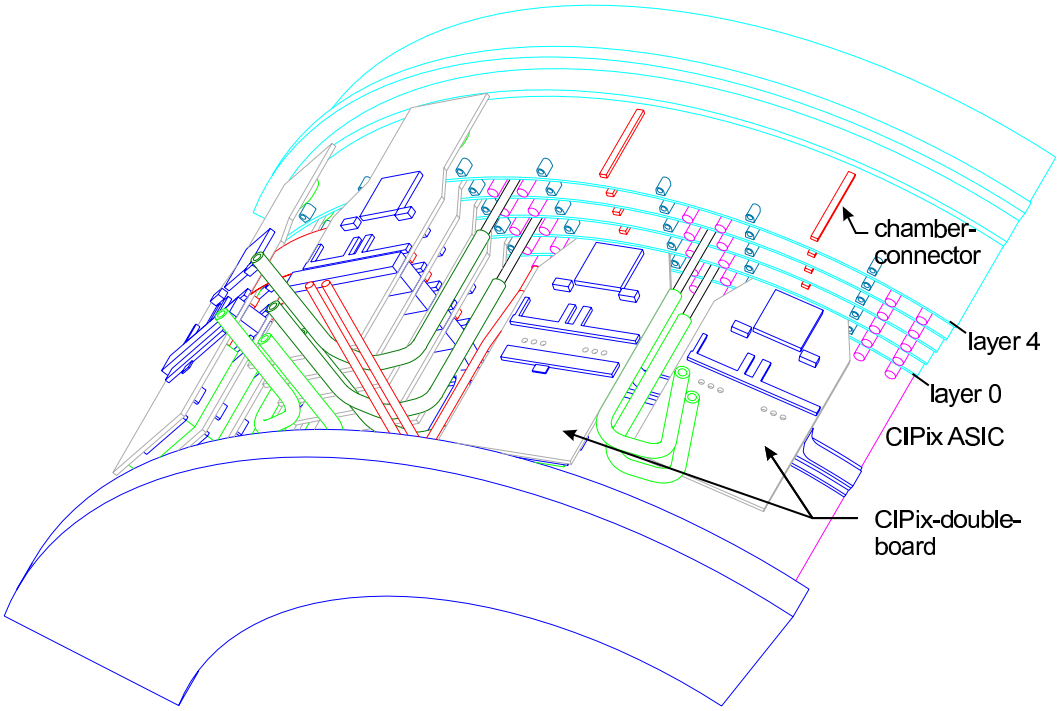


Fig. 5.5: Technical 3d-drawing of the backward end flange of the new CIP2k with the CIPix boards, mounted onto the CIP2k chamber at all five layers [62]. Clearly visible are the gas supply tubes (between layers 0-1 and layers 3-4), the HV coaxial connectors and the 120-pad chamber connector.

interconnected by a flexible strip-line on a capton foil carrier. One double-board reads out two φ-sectors of one layer.

With the 120-pin connector on the chamber, the pad information is directly transferred to the two CIPix chips on each PCB. Each CIPix has capacities for 64 analog inputs[4].

CIPix ASIC: The CIPix readout chip was developed at the ASIC laboratory [71] of the University of Heidelberg extending an earlier development, the HELIX-chip. The latter was used for the readout of multi strip gaseous chambers (MSGC strips) at the HERA-b experiment. MSGCs have different input characteristics than MWPCs [72]. Each of the 64 analog input channels consists of:

- A charge sensitive preamplifier with a gain of 20 mV per 10^5 electrons to minimize crosstalk effects of other pads and coupling of electric noise. The preamplifier characteristics are programmable.

- Analog signal shaping to a *CR-RC* semi-Gaussian pulse with programmable shaping times (peak time of $\approx$ 50-70 ns).

- Digitalization by an 1-bit comparator with programmable threshold and polarity.

[4]Only (up to) 60 channels are used. The additional channels are used for testing purposes only.

All programmable parameters of the CIPix are controlled by an I^2C bus. The I^2C system itself is controlled by the experiment control system PVSS II [73].

The discriminated signals are synchronized with the HERA clock signal (10.4 MHz) and multiplexed four-fold into 17 digital channels (15 outputs and two control channels) at 41.6 MHz. In parallel, an analog output is provided, which can be used to monitor any of the 64 input channels. The *first word* bit (fst_wd) tags the first of the four multiplexed pads and, hence, allows a synchronization with the (demultiplexing) electronics in the trigger system in the trailer, described in Section 6.2.4 on page 58. In Figure 5.6, a schematic diagram of the front end PCB is shown. The two boards are

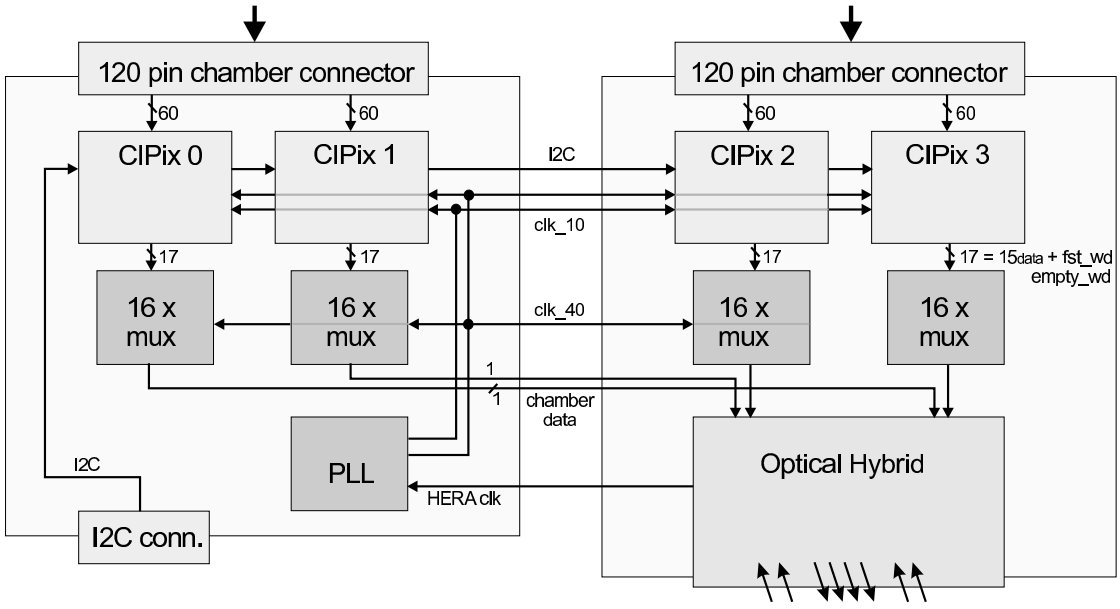


Fig. 5.6: A schematic overview of the CIPix double board (see text).

connected with a strip line. The HERA clock signal is given to a phase-locked loop (PLL). The 41.6 MHz clock is delivered to the CIPix chips and the 17 to 1 multiplexer. 15 four time multiplexed data and first word and empty word signals are handshaked from the CIPix to the 16 fold multiplexers and then to the optical hybrid (1 wire per CIPix) for transmission to the trigger system in the electronics trailer.

5.5 The Optical Link System

To transmit the ≈ 10.000 pad signals to the trailer, an optical link system was developed by the ETH Zürich. Every optical fiber carries the information of one CIPix double-board to the trailer. In total 40 fibers are used.

Multiplexer: To reduce the number of optical fibers, the 15 digital output signals (plus the fst_wd signal and the empty word (see Chapter 6)) of each CIPix are multiplexed into a single channel using a multiplexer unit (Hewlett Packard HDMP 1032). In total, every CIPix double board contains four CIPix chips, therefore four multiplexer are necessary (see Figure 5.6).

Optical hybrids: The multiplexed information is delivered to an optical hybrid. It has six transmitters and two receiver diodes. The six transmitters transfer the four digital data channels (one for each CIPix) and two analog channels (each shared between two CIPix units) to the trigger system. The receiver diodes transport the 10.4 MHz clock to synchronize it with the HERA beam clock. The individual fibers from one transmitter unit are bundled in a single cable. Special receiver cards (see Chapter 6) in the electronics trailer are connected on the other end of each cable. They provide the 10.4 MHz clock and contain four demultiplexers (Hewlett Packard HDMP 1034) to recover the digital outputs of each CIPix. The buffered analog outputs are available for monitoring. The CIPix and the optical link system are described in more detail elsewhere [74], [75], [76].

Chapter 6

The Trigger System Hardware

In this chapter the hardware structure of the CIP2k trigger system is presented.

The chamber data is transferred via 40 m long optical waveguides to the CIP2k trigger system in the electronics trailer next to the H1 detector. The CIP2k trigger hardware serves two purposes:

1. Providing a z-vertex trigger decision for every BC at trigger level 1.

2. Holding the chamber information in a pipeline and delivering it to the H1 storage system in case of a positive trigger.

It was decided to use large FPGAs[1], type Altera Apex 20k400, for the trigger system. The whole trigger algorithm can be hosted in a few FPGAs. Moreover, pipelines for the chamber data can be realized in the same FPGAs. FPGAs are programmable in a flexible way with specially developed hardware description languages (HDL, see Section 6.8.2 for an introduction and a detailed description). The development of the hardware can be reduced to the routing process of the chamber information to the FPGAs and to a development of a support structure of control signals (clock, data bus, reset ...). The complexity of the algorithm is shifted from the hardware development to the programming of the FPGAs. In Chapter 8 the programming of the presented FPGA-based hardware is described.

A test system was developed [67] to clarify how much of the chamber data can be handled in one FPGA and if this FPGA is able to store the necessary amount of chamber data. This system was realized on a VME-board with one Altera APEX 20k400 FPGA. It turned out that this type of FPGA fits all requirements.

A data flow plan of the system is presented in Section 6.1. The components of the trigger system visible on the data flow plan are described in Sections 6.2 to 6.5. In Section 6.6 the CIP2k slow control is discussed. Section 6.7 gives an overview of the present CIP2k trigger system hardware. In Section 6.8 a detailed description of the relevant logic chips is given.

[1] **F**ield **P**rogrammable **G**ate **A**rrays

6.1 Flow Plan of the Trigger System

A total of 40 optical waveguides transfer the information from the front end electronics over a distance of ≈ 40 m to the trigger-system. In Figure 6.1 a flow plan of the CIP2k trigger system is shown. Each optical fiber ends in a *receiver card (RC)*. 10 receiver

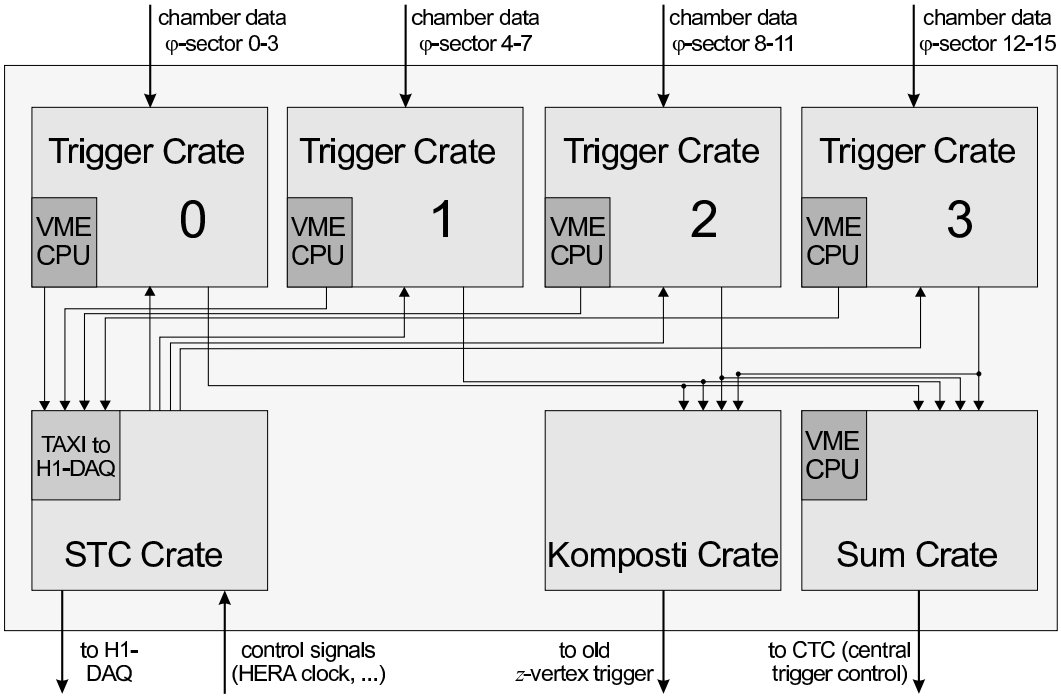


Fig. 6.1: An overview of the CIP2k data flow and hardware components. All crates are shown. On the top, the signals from the chamber are linked into the trigger system. Below the outputs to the old z-vertex trigger, to the central trigger control and to the H1 central data acquisition system (DAQ) are shown. Subsystem Trigger Control signals are linked into the STC crate.

cards are connected to the back of one trigger crate, providing the signals from all five layers of one quadrant of the CIP2k chamber. The complete chamber signals are thus received in four trigger crates.

The receiver cards are plugged from behind into a specially developed backplane, which distributes the information to four trigger cards on the front side. Each *trigger card* (TC) contains the logic of the trigger algorithm and the storage pipeline for one φ-sector of the chamber. In total 16 trigger cards deliver independent trigger information to the sum crate. In the sum crate the trigger information of all 16 φ sectors is added and the trigger decision is calculated. Five identical sum cards are used to perform this task. Four cards receive the information of four trigger crates each and build the trigger information of a quadrant. The fifth card, the main sum card, builds the final trigger information and provides a trigger decision to the H1 level 1 central trigger control.

Besides the trigger cards, each trigger crate contains two *control cards* and a CPU. The control cards handle the distribution of the control signals. In addition, they provide the information of the first two layers of the CIP2k (layer 0 and layer 1) for the old H1 z-vertex trigger as discussed in Section 4.2. These data are transferred to the

komposti[2] crate, containing eight komposti cards. The komposti card uses the same hardware as the sum cards. The pipeline data stored in the trigger cards are read out by a CPU (CES RIO 8062 [77]) via the VME (**V**ersa **M**odule **E**urocard Bus) bus. Each CPU reads the data from four φ sectors (one trigger crate). One of the CPUs collects the data from all φ sectors and writes it into a local event buffer. If the H1 trigger approves the event (at trigger-level 3), the data are sent via a VIC (VME interconnect) link to the H1 central data acquisition system. Control signals (HERA clock, etc.) from the subsystem trigger control (STC) are linked into the CIP2k STC crate. It distributes the signals to all crates of the system.

6.2 Trigger Crates

Each trigger crate hosts ten receiver cards, 4 trigger cards, two control cards and one VME CPU. The crates were built by the manufacturer Wiener [78]. They have a special backplane and additional power supply units for 3.3 V and -5 V. The P1 connector conforms to the VME-crate standard. In Figure 6.2, a schematic overview of the trigger crate is shown.

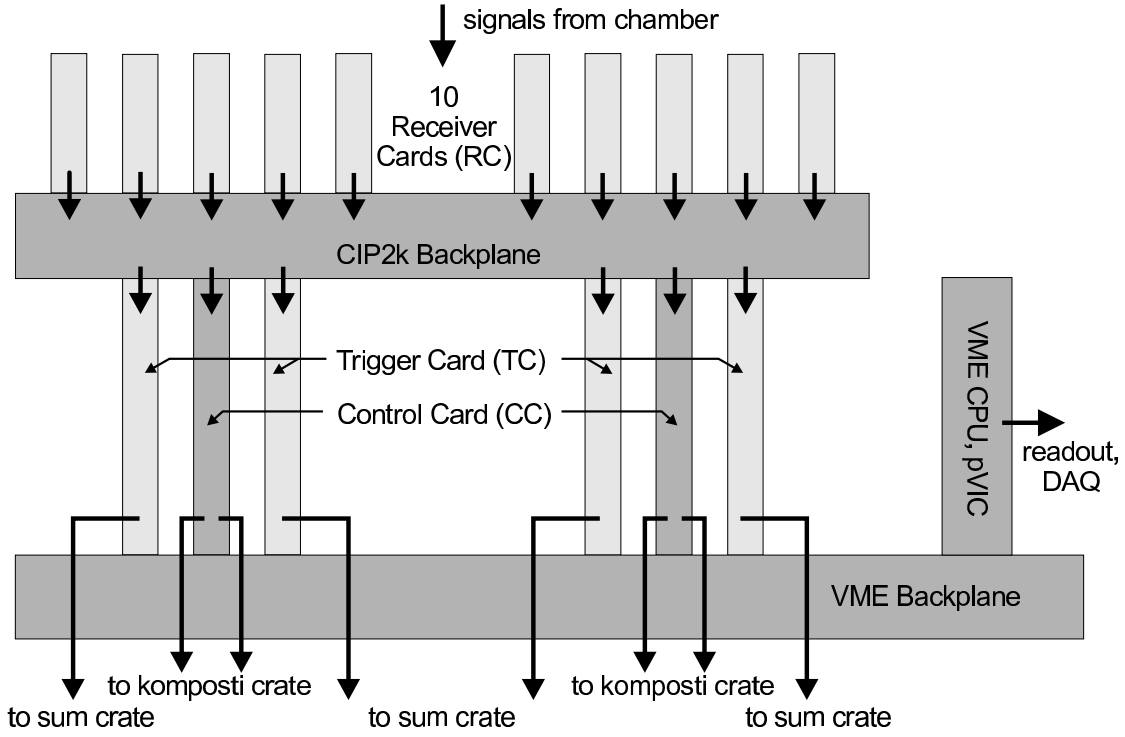


Fig. 6.2: An overview of the CIP2k trigger crate.

[2]The name **komposti** crate/card was formed in a discussion about how to shrink the information of the new chamber with a four times higher granularity to the old z-vertex trigger. This seemed to be something like a heap of compost: you end up with a useful concentrate of the input.

6.2.1 Backplane

A special 10-layer PCB-backplane is used in the trigger crate at the lower side (instead of a standard VME bus P2 connector). It was developed at the University of Heidelberg. At the backside of the backplane, 10 VME-like three row *male*-connectors are installed, used to connect the receiver cards. Two rows of this connector receive the chamber signals via the receiver card, the inner row contains all necessary clock lines to/from the receiver card and front-end-electronics. On the front side, four slots have VME-conform connectors, six slots have special 250-pin connectors, arranged in two groups. In every group, two trigger cards and one control card are connected to the backplane as shown in Figure 6.2. The following assignment function is used to map the data from the receiver card to the trigger card:

$$2 \times 5_{RC} \times (2\,\varphi\,\text{sectors for 1 layer}) \Longrightarrow 4_{TC} \times (1\,\varphi\,\text{sector for 5 layer}) \qquad (6.1)$$

A similar assignment function is used to map the chamber data from the receiver card to the control card (layer 0 and layer 1). The clock information (41,6 MHz clock (clk_40[3]) and the first word signal (fst_wd)) of the receiver cards of layer 2 is distributed to the trigger cards.

The (upper) P1-backplane of the trigger crate is a standard VME backplane.

6.2.2 Receiver Card

The receiver card was developed at the ETH in Zürich. Every receiver card deals with the information of four CIPix units corresponding to one CIPix-double-board at the chamber. The receiver card converts the optical signal to electrical and demultiplexes it. Moreover, every receiver card provides the fst_wd of four CIPix chips. The clk_40 is generated on the receiver card, phase locked to the HERA clock. The pad and clock signals are sent to the backplane. Additionally, on every receiver card LEMO and pin connectors are mounted, which can be used for measurements of the clk_40 and the fst_wd signals as well as for selected analog pad information from the chamber. Detailed information can be found elsewhere [76].

6.2.3 Trigger Cards

The trigger card, realized as an 8 layer PCB, was designed in close cooperation with the electronics workshop of the University of Heidelberg [79] (internal identifier: DL533-1). Considerations for the design, evaluation of the test system and the creation of the circuit diagrams of the trigger card are described in [67]. In total, 20 trigger cards were produced (4 fully equipped spare cards).

In the following, the trigger card hardware is described in its actual implementation.

Layout of the board: The trigger cards hold the whole trigger logic in a total of 32 FPGAs, two FPGAs per φ-sector. Thus, each trigger card contains two Altera APEX 20K400E FPGAs [80], mounted in a ball grid array (BGA) technique onto

[3] clk_40 means: 4 times multiplied HERA clock (10.4 Mhz) $\Rightarrow$ 41.6 MHz

the PCB, and one Lattice ISpL 1048E PLD (programmable logic device [81]). Every FPGA has 655 pins. 502 pins are free programmable I/O pins, the other pins are mainly power supply connections and pins to program the FPGA (Section 6.2.3). The board is equipped with a 5 V to 1.8 V power converter to deliver the 1,8 V FPGA kernel voltage. External 3.3 V and 5 V power supplies are needed for the FPGA I/O pins and the ISpL and VME bus connections. The FPGA needs 3,3 V for in- and output pins, thus all I/Os are driven with additional 5 V from/to 3,3 V converters (type: 74LVT16244B). To configure the FPGAs the PLD is used. Six EEPROMs (electronic erasable programmable read only memory, type: EPC2 LC20) keep the configuration code, three for every FPGA. Furthermore, the PLD is used as a VME bus controller. In Figure 6.3, a schematic view of the trigger card is shown.

Each trigger card has five connectors in total, shown in the Figure.

1. A 250-pin female connector to connect to the P2 backplane.
2. A standard VME connector for the upper P1 backplane.
3. Two SCSI (micro sub-D) connectors[4] are mounted on the front panel, one with 68 pins, one with 50 pins. They are used to transport φ-dependent information to the sum cards and for monitoring.
4. A 10-pin program connector is attached to the front panel to configure the FPGAs and EEPROMs.

Dataflow: 95 lines are used to transport the chamber information from the 250-pin input connector on the right side of the trigger card to each FPGA. The first part of the algorithm (see Chapter 7) is done separately for the first 60 pads of the chamber in FPGA I and for the remaining (46) pads of the chamber in FPGA II. 15×4 multiplexed channels keep the information of one half of the layer (60 pads $\hat{=}$ 1 CIPix). In total $5 \times 15 = 75$ pins transfer the information of five layers. Additional 4×5 pins are linked from the other half of each corresponding layer to the other FPGA to deliver overlap information, needed for the trigger algorithm (see Figure 6.4).

A 90 lines interconnect bus connects FPGA I with FPGA II. Information evaluated by FPGA I is transferred to FPGA II. FPGA II is connected to the two SCSI-connectors to deliver φ-based data to the sum cards via LVDS converters (Dallas DS90LV031) that form the 3.3 V FPGA output (LVTTL, low voltage TTL) to a LVDS (low voltage differential standard) signal.

VME bus: Every trigger card is equipped with a 32-bit VME bus controller. 32 bidirectional data lines are linked to both FPGAs and to the ISpL. The ISpL switches the data bus direction according to a read or write access.

A 24-bit standard non privileged address mode (address modifier: 0x39) is used. In Figure 6.5, the address bits are shown. 8 address bits are used to identify each trigger card at the VME bus address (base address). It can be selected by two *hexadecimal*

[4]SCSI = small computer system interface, note: the connectors are not used in a SCSI conform function

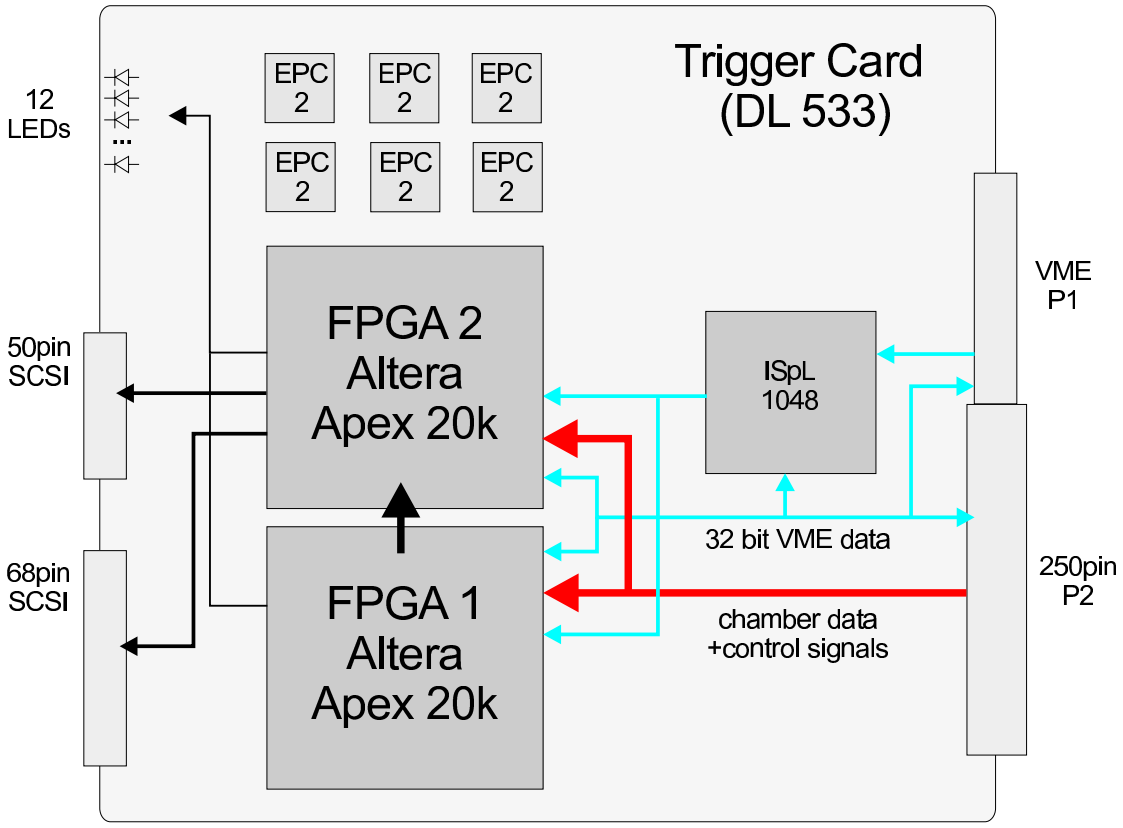


Fig. 6.3: Schematic view of the trigger card (TC). The chamber data are linked into each trigger card via the 250-pin P2 connector. A 90-lines interconnect bus connects FPGA I with FPGA II. Two SCSI connectors are connected to FPGA II, one with 50 pins and one with 68 pins. The VME bus is accessible via the ISpL 1048. Programming both FPGAs via six EPC2 devices, external PC and VME bus is described in detail in Figure 6.7 on page 55.

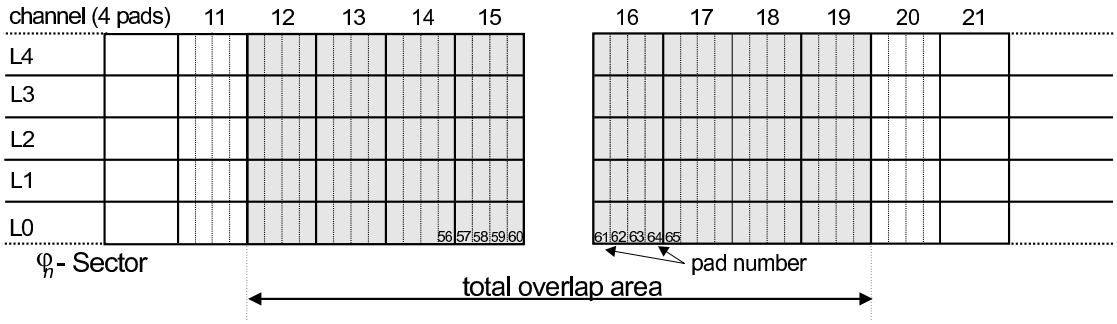


Fig. 6.4: Because one FPGA will not load the whole trigger algorithm for a complete φ-sector, the total amount of pads is divided into two FPGAs. Every FPGA deals with the information of 60 pads ($\hat{=}$ 1 CIPix) of five layers. Tracks that point to pads of both CIPixes define the overlap area. The information in the overlap area is given to both FPGAs.

switches on the upper left side of the trigger card (see Figure 6.3). Address bits 14 and 15 are used for the direct addressing of either the PLD (bit 15), or FPGA I (bit 14) or FPGA II ($\neg$bit 14). Bit 13 is used for the selection of either the pipeline readout or the setting of control registers (see Section 8.2.2.1: VME Interface). Bits 2 to 7 are used

for a direct addressing of 64 VME 32 bit words.

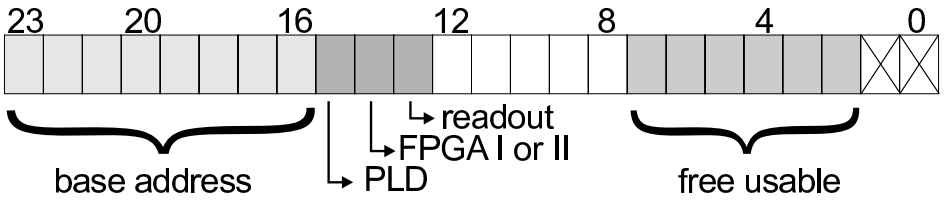


Fig. 6.5: 24 bit address space (AM 0x39) of the trigger card.

The ISpL is set up as a VME-bus slave unit according to the VME standard. It organizes the data transfer from the FPGAs to the VME bus master. A read, write, or block mode is supported. An VME interrupt handler is prepared.

Read cycle: In Figure 6.6, a VME bus read access is shown with its necessary VME control lines. Because the VME bus is an asynchronous bus, the ISpL converts

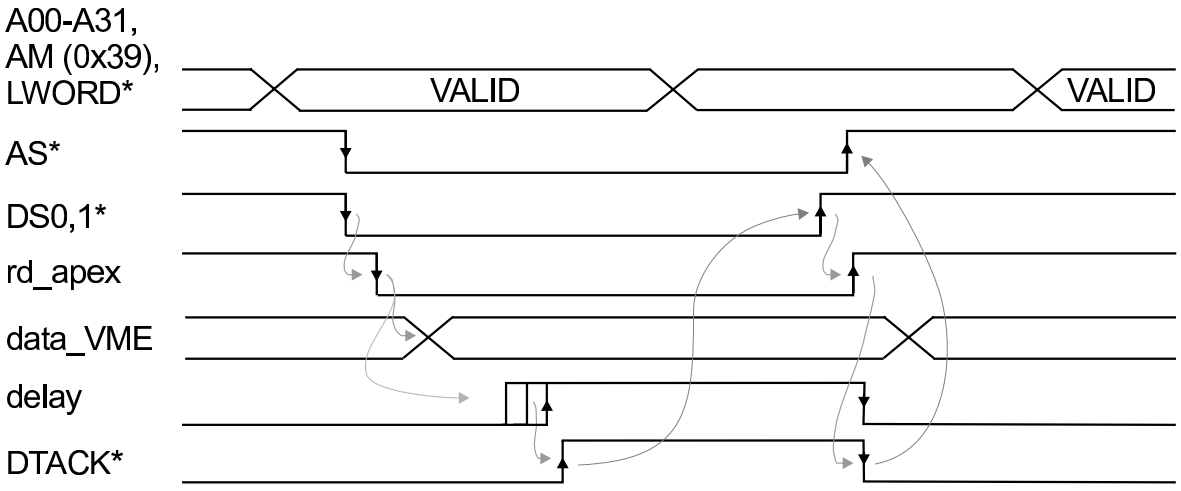


Fig. 6.6: VME read cycle from the trigger card FPGAs (unused VME signals are not shown).

the asynchronous protocol to a synchronous one for the FPGAs.

The VME bus master sets all address lines to a valid state (A00-A23, address modifier (=0x39) and $LWORD^*$[5]) and sets the address strobe signal (AS^*). Followed by this the data strobe signals $DS0^*$ are set. The $DS0^*$ signals communicates to the slave device that the master awaits a read or write cycle. With the $DS0,1^*$ signal the `rd_apex`* signal is set to low. Now, the FPGA receives data from the VME bus (`data_VME[31:0] = internal_info`). If the data have arrived safely at the bus, the data acknowledge signal ($DTACK^*$) is set by the slave device. Because there is no handshake of the data between ISpL and FPGAs, the setting of the $DTACK^*$ signal can be delayed by up to 6 internal clock cycles in the ISpL (64 MHz) to guarantee save information on the data bus. With the $DTACK^*$ signal, the VME bus master releases the data strobe signals $DS0,1^*$. The `rd_apex` signal is reset by the ISpL and the FPGA stops reading on the VME bus, then releases the $DTACK^*$. Following this, the master releases the AS^*. For every read cycle this procedure has to be repeated.

[5]Signals labeled with * are active low.

Write cycle: The VME bus write cycle operates similarly. Because the master is the receiver now, an acknowledgement signal has to be generated by the ISpL (slave) to access a VME write cycle with save data transport to the CPU (corresponding to the data strobe signals at the read cycle). With the `wr_pulse` the ISpL indicates to the FPGAs that it sets the $DTACK^*$ signal and the FPGA data have to be save at the VME bus with this cycle.

The code in the FPGA has to be written in a way that all data bits are set together with the read or write signals (`rd_apex`, `wr_apex` and `wr_pulse`) (see Section 8.2.2.1).

A precise description of the VME bus can be found at [82], the firmware code of the ISpL can be found at [83].

Configuration of the FPGAs: APEX 20k FPGAs have to be configured after each power cut. At startup each FPGA is configured with a configuration file. It contains the firmware to operate the FPGA as intended. It is generated by the development software and can be loaded into the FPGAs in several ways:

- Directly via programming connector with a personal computer.
- Via VME bus.
- From the EEPROMs, triggered by a VME signal. The EEPROMs can be programmed via VME-CPU or via programming connector.

Every APEX 20k FPGA provides a static random access memory (SRAM) area that is used to store the configuration information of the FPGA. After the configuration, this memory can be used. The configuration information is written into a binary sof-file (with the extension *.sof ,*SRAM object file*) and is transported into this memory in case of the configuration. The sof-file contains the data for configuring the FPGA. It is generated by the compiler's assembler module [84]. In addition, a binary pof-file (*programming object file*) is generated. It is used in EEPROMs (EPC2) that configures the FPGAs. FPGAs and configuration devices of every trigger card are linked together in a chain (JTAG-chain). Every device has a checksum for identification and is listed with its correct type in the correct order. In Figure 6.7 the configuration scheme for the EEPROMs and FPGAs is shown. The direction of the JTAG-chain is plotted. The chain in the trigger card contains six configuration devices and two FPGAs. To offer the same space like in the SRAM area of the FPGA as EEPROM space, three EPC2 devices are necessary. Therefore, a total of six pof-files and two sof-files are needed. The position of every EPC2 device and FPGA, the unique identifier and the information of every pof- and sof-file are written into one binary file, the jbc-file (**j**am-player **b**inary file for **c**onfiguration). It is used by the *jam-player* [86], reading the jbc-file. It creates a defined one bit data stream (TDI) together with a clock signal (TCK) and writes it into the EPC2 device or the SRAM area of the FPGA. To ensure the correct configuration, the data stream is read back (TDO). The configuration with the VME bus works in the same way as with the PC via connector. The *jam-player* is installed on the VME CPU and uses the VME bus for transportation of the data to the EPC2 and FPGA devices. The files for the *jam-player* are located on a local server [87]. Configuration with the EEPROMs takes only some 100 ms, configuration via VME bus or via programming connector takes roughly one minute per FPGA.

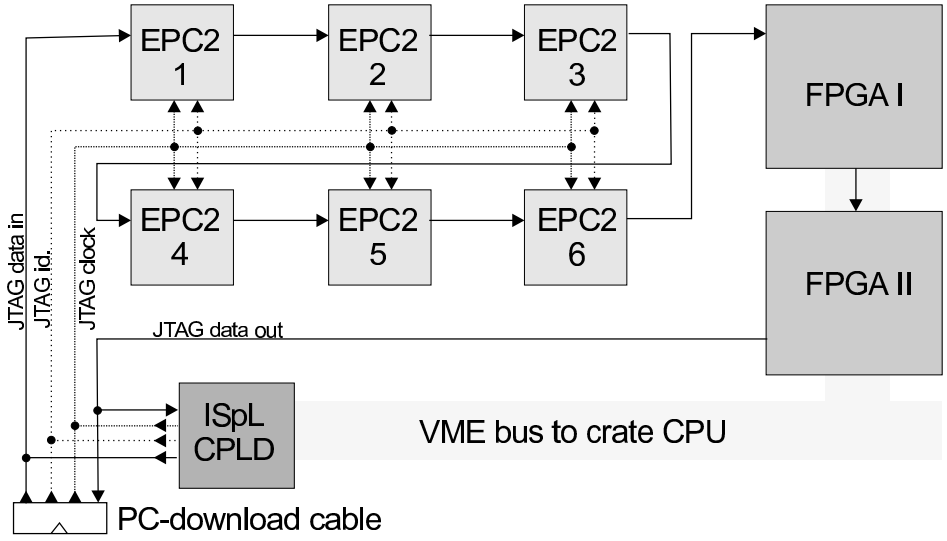


Fig. 6.7: Configuration scheme for the EEPROMs and FPGAs. The circuit diagram of this scheme can be found in [85], sheet 2.

The ISPL PLD is programmed with the help of a PC via an 8 bit connector on the trigger card.

Pin assignment: Every pin used in a dedicated design is assigned corresponding to a pin assignment file defined in the cdf-file. This file contains all I/O pins used in the design and the major settings to operate the FPGA like clock frequency, signal level of each pin (+3.3 V, tristate, ground, assigned) and the selected auto restart configuration.

Clock signals are assigned to dedicated *fast inputs* or *clock inputs*, provided in the FPGA structure (see Section 6.8.1). In Table 6.1 the assignments are summarized.

Control and monitoring facilities: To have a simple indicator of the correct operation of the FPGAs, in total 16 LEDs display important information, 8 for each FPGA. A correct VME transfer is indicated by a signal from the VME bus controller, showing every read or write access (sel-LED, according to the description above).

A successful configuration process of the FPGAs is indicated by a configuration done bit (*conf_done*), sent to the VME bus controller. It turns off the crate wide VME status bit (*VME_SYS_fail*) in case of a correct programming of all FPGAs.

In the improbable case of a FPGA failure, indicated by VME read or write errors on the corresponding VME base address (and *VME_SYS_fail*), the FPGAs have to be configured again. Fuses are installed on every trigger card. Two LEDs indicate power failures on the board, one for $3.3\,V$ and one for $5\,V$.

In Figure 6.8, a photo of the trigger card is shown. Both FPGAs, the 250 pin input connector (right), and the SCSI connectors (left) are visible.

Signal		FPGA I	FPGA II
fast-	clk_40, pipe_en,		
input	fst_wd, n_g_rst,	6	6
	l3_reject, n_rst		
input	data_CIP_E_x_in	5×19	5×19
input	jumper	2	2
input	histo2FPGA2 (from FPGA1)	-	90
input	VME-bus:		
	rd_apex, wr_apex,		
	wr_pulse, rd_blk,	5	5
	wr_blk		
	addr_VME[7:2]	8	8
	addr_VME13, addr_VME14		
I/O	data_VME	32	32
I/O	dasy_chain I/O	2	2
output	histo2FPGA2 (to FPGA2)	90	-
output	SCSI connectors		
	68 pin	-	32
	50 pin	-	24
output	LEDs	8	8
TOTAL		248	304

Tab. 6.1: Pin assignment of FPGA I and FPGA II.

6.2.4 Control Card

The control card (CC) was developed in addition to the trigger cards to fulfill the following duties:

- It distributes the HERA clock (h_clk) to the receiver cards and monitors the phase locked fst_wd signal from the CIPix chips. Thus, a correct distribution of clock signals to the PCBs in the trigger system is guaranteed.
- It controls the global reset signal.
- It delivers the chamber information of layer 0 and 1 together with all clock information to the komposti card.
- It provides a basic cosmics trigger with a logical combination of *not empty word signals* (!empty_wd). The !empty_wd is generated in every CIPix ASIC, if at least one of the 60 pads in on CIPix is *on*.

Every trigger crate hosts two control cards, corresponding to the complete chamber information of two φ-sectors ($\hat{=}$ 2 trigger cards, 5 receiver cards). The control card (internal identifier: DL534) was developed at the electronics workshop of the university of Heidelberg [79], [67], [85]. It consists of three types of PCBs: The main board, containing all active electronics components and the backplane connectors as well as the LEMO in and outputs, and two piggy back PCBs that contain two 68 pol SCSI connectors and LVTTL to LVDS converter (Dallas DS90LV031) each. Both piggy backs have the same layout design, but by mounting the connectors to the main board

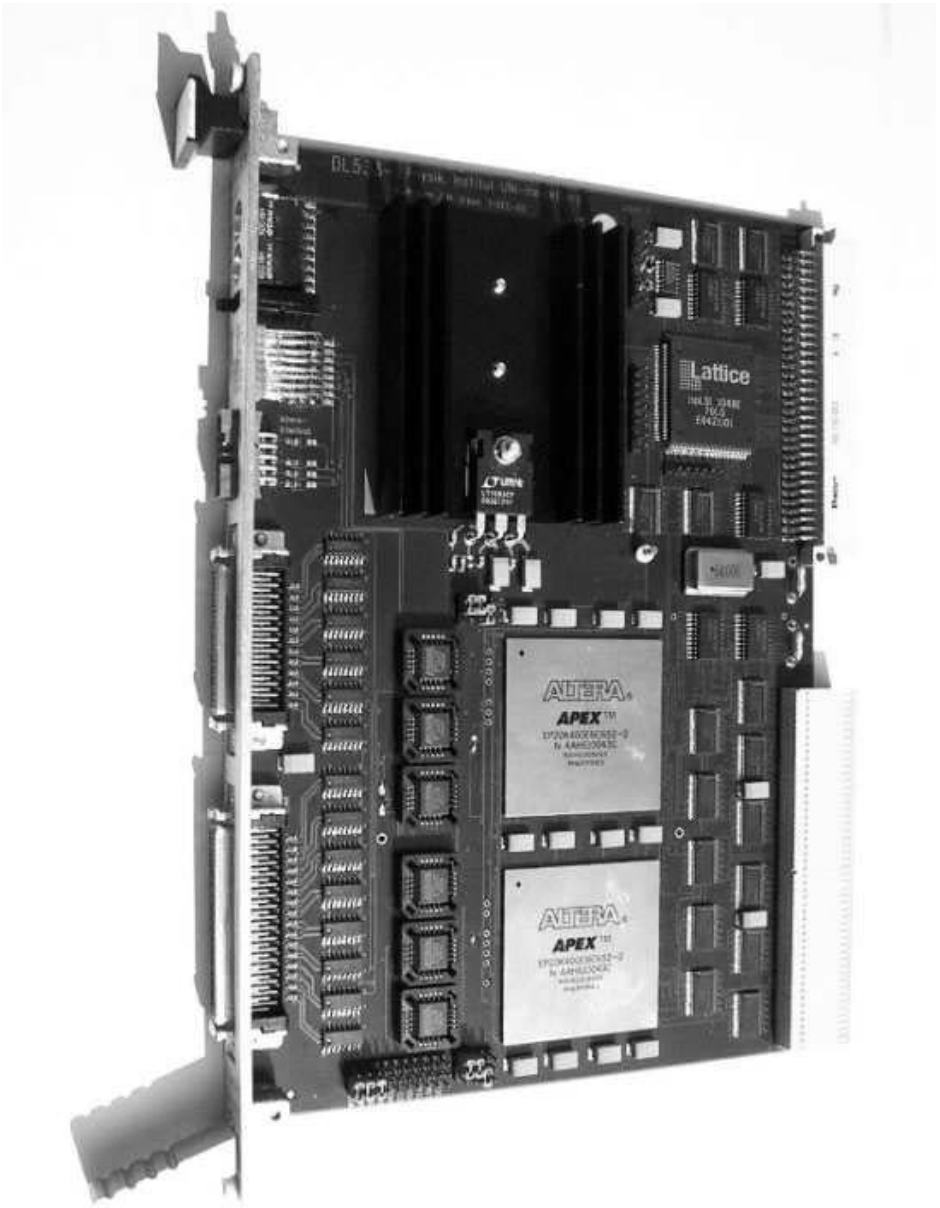


Fig. 6.8: Photo of the Trigger Card: Both FPGAs, the 250-pin input connector on the right side and the SCSI connectors on the left side can be seen.

on two different (provided) positions, the piggy back can either be used as the upper piggy back or the lower piggy back.

In the control card the same 250-pin connector as in the trigger card is used besides a standard P1 VME connector. On the front side of the board, 10 LEMO sockets are attached. Beside h_clk, pipe_en, g_rst and an additional l3_reject signal (see trigger signals timing in Section 3.6), another six LEMO sockets are used for free programmable extensions (with a jumper setting as in- or output). A PLD (Lattice ISpL 1048 [81]) is used for a flexible programming of all necessary signals. The same PLD as at the trigger card is used, configured in a similar way. A VME bus controller is programmed.

It has a 32-bit data bus and 6 free address bits in 16-bit address mode (AM 0x29). The PLD hardware description code can be found at [83]. VME programmable delay chips are used for the clock control circuit (see paragraph 6.2.4) together with a 10-bit analog digital converter (ADC, type: Fairchild SPT7855) that operates as a duty-cycle meter.

Distribution of the HERA clock: Every subdetector gets the same H1-wide clock signal (HERA clock, h_clk) to operate synchronously to H1. In the CIP2k system, the h_clk signal is linked to a global programmable delay line. It delays the signal in programmable steps of 0.5 ns. After that, the signal is given to five separate programmable delay lines corresponding to five receiver cards/CIPixboards. At the CIPix board, the h_clk is given to a phase locked loop (PLL). It generates a 41.6 MHz signal (clk_40). The fst_wd signal is generated in the CIPix chip. Both signals are phase locked to the h_clk. The control card receives the five fst_wd signals from each layer CIPix PCB and the ckl_40 signals from the receiver card PLL. The signals of (layer 2)[6] are delivered to both trigger cards.

To ensure that the signals of the other four layers are synchronous, the phase relation of the fst_wds of all layers is compared with the h_clk. The length of the fst_wd is extended from 25 ns to 50 ns for this measurement. The value of the modified fst_wd is combined with the h_clk with an logical AND. In case of the correct phase relation, the resulting duty cycle is maximal. It is integrated and afterwards digitized in the ADC. In Figure 6.9 this is shown. The digitized time difference value of each

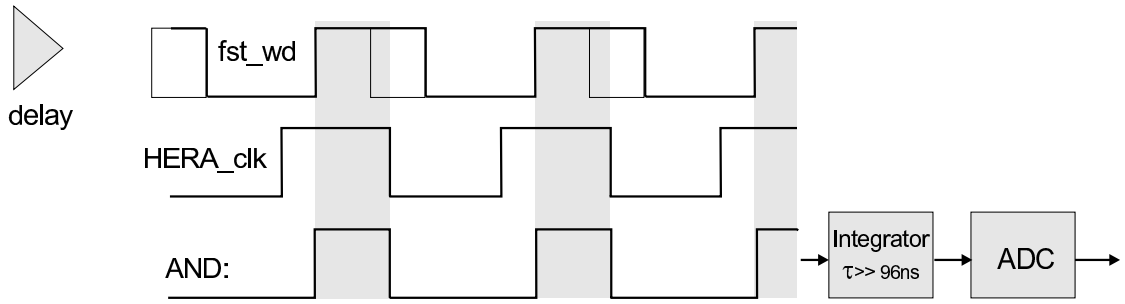


Fig. 6.9: Clock synchronization by comparing each fst_wd signal with the HERA clock. The resulting duty cycle is integrated. The integrated value is digitized in the ADC.

pair is accessible by the VME CPU on a unique VME address. The CPU sets every time difference to the same value by programming the delay lines on the control cards.

If rising edges of all fst_wds (all layers) are at the same time, chamber signals from all layers are demultiplexed in the same clk_40 cycle and the multiplexed information is demultiplexed correctly. Similarly, the clk_40 signals are tuned.

In Figure 6.10, the clock control circuits are shown.

[6] In case of a problem with the signals of layer 2, additional adapter cards are used that provide the fst_wd and clk_40 signals of another layer to the trigger cards. These adapter cards are put between the receiver card and the CIP2k backplane. Details can be found in [88].

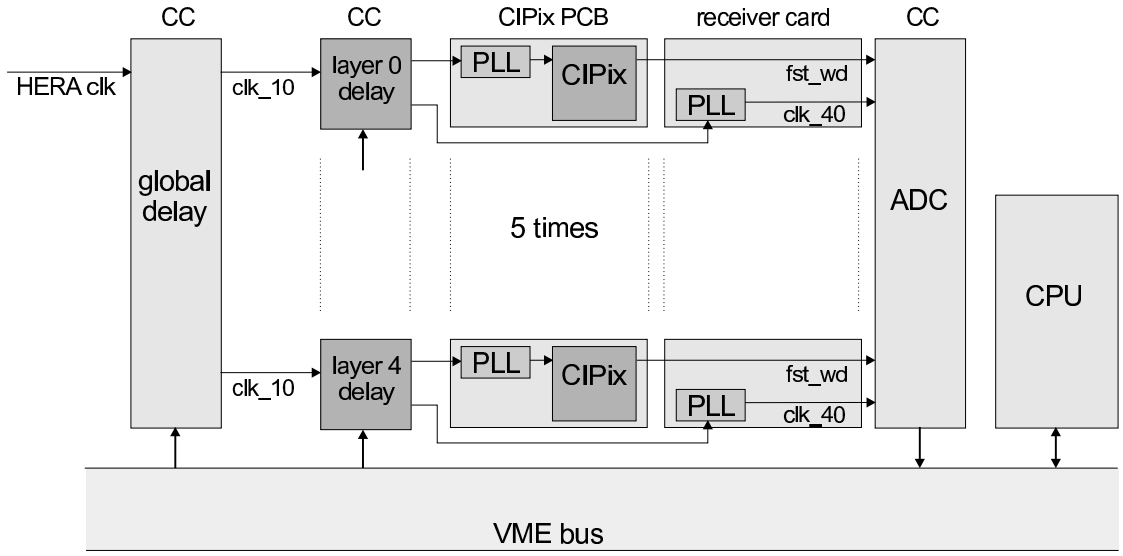


Fig. 6.10: HERA clock regulation circuit. The HERA clock is distributed to all five CIPix pcbs and can be delayed separately to ensure that chamber data from all layers are at the trigger card at the same time.

Distribution of the global reset signal: A system wide reset signal (*global reset*, g_rst) is generated at every run start. The g_rst signal is linked to the PLD. In the PLD the signal is formed to a defined length of one, two or four h_clk cycles ($\simeq$ 96, 192 or 384 ns), depending on the programming. A precise defined reset signal is important for the reset of the 16 fold demultiplexer in the receiver cards. The reset signal is also given to the trigger cards, used to set the FPGAs in a defined state (reset state) after a configuration procedure.

Distribution of the chamber information of layer 0 and 1: The chamber information of the first two layers is provided to the old z-vertex trigger. The four times multiplexed information of layer 0 and layer 1 together with necessary control signals is linked through the control cards to the komposti cards. Each control card has four 68-pol SCSI connectors and covers two φ-sectors, eight cards are used in total. Every 68-pol SCSI connector deals with 32 differential LVDS signals, transferred from the receiver card and converted. In Figure 6.11, the distribution of all channels is shown in a schematic diagram.

$$14\,\text{channels}_{CIPix} \times 2\,\text{CIPix}_{layer} \times 2\,\text{layers} \times 2\,\varphi\,\text{sectors} = 112\,\text{signals} \qquad (6.2)$$

are distributed per control card. Every SCSI connector transports four additional signals: h_clk, clk_40, fst_wd and g_rst. The first and the last multiplexed channel of the chamber in z-direction is cut off to fit the relevant chamber information onto four SCSI connectors (14 instead of 15 channels). Connector 1 hosts the information of the first 7 multiplexed channels in z for both layers and both φ sectors, in total 28 signals. The second connector hosts the next seven channels, and so on.

A simple cosmics trigger: To prove the functionality of the chamber independently of the trigger system, a cosmics trigger is programmed in the PLD. It uses the

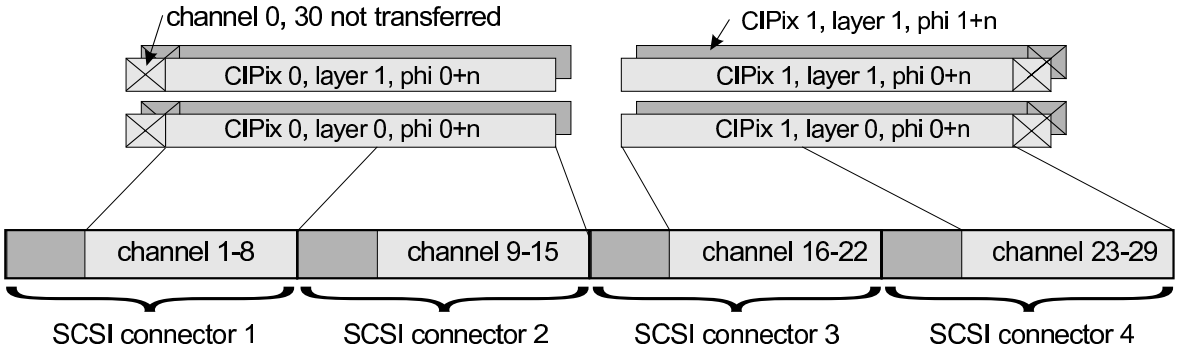


Fig. 6.11: In total four SCSI connectors per control card transfer the data of the new CIP2k to the old trigger system. Every connector contains 28 channels, 7 channels per layer and φ sector. Additionally, four serving signals are distributed: h_clk, clk_40, fst_wd and g_rst.

!empty_wd signals of every CIPix chip. The !empty_wd signal is set when at least one of the 60 pads on a CIPix is active (per event cycle). In total, ten !empty_wd signals are linked into every control card. To keep the number of logic units in the PLD at a reasonable amount, a rather simple algorithm was used:

$$\begin{aligned} \text{COSMIC} = &(\text{ew}_{00}\, \textit{or}\, \text{ew}_{10}) \times \text{enable\_0} \\ \textit{and}\ &(\text{ew}_{01}\, \textit{or}\, \text{ew}_{11}) \times \text{enable\_1} \\ \textit{and}\ &(\text{ew}_{02}\, \textit{or}\, \text{ew}_{12}) \times \text{enable\_2} \\ \textit{and}\ &(\text{ew}_{03}\, \textit{or}\, \text{ew}_{13}) \times \text{enable\_3} \\ \textit{and}\ &(\text{ew}_{04}\, \textit{or}\, \text{ew}_{14}) \times \text{enable\_4} \end{aligned} \tag{6.3}$$

ew_{00} means !empty_wd of CIPix 0 (at $-z$-position) in layer 0. With a programmable enable_{layer} bit, every layer can be checked. This provides a simple cosmics trigger without using the FPGA based trigger system. This simple cosmics trigger was used to test the CIP2k chamber in Summer 2001 (In this test run, only layers 3 and 4 were used. Layers 0,1 and 2 were not enabled).

6.3 Sum Crate

The sum crate, a VME 21-slot 64x crate manufactured by Wiener [78], contains five identical sum cards and one VME bus expander. The bus expander connects the sum crate to the CPU of trigger crate 0. The VME 64x backplane is used to deliver an additional 3,3 V supply to the sum cards.

In the sum crate the z-vertex histograms of all 16 φ sectors are added and the trigger decision based on the summed histogram is calculated. In Figure 6.12 a schematic view of the sum crate is shown.

Four sum cards are used as *pre*-sum cards. Each card adds four φ sector histograms (one quarter of the chamber). The fifth sum card, the *master*-sum card, calculates the total sum, using the pre-summed histograms. Every sum card, moreover, calculates a 16-bit trigger decision (H1 trigger elements), available on a 50-pin SCSI connector at the front panel of each sum card. The trigger decision of the master card is linked to the central trigger.

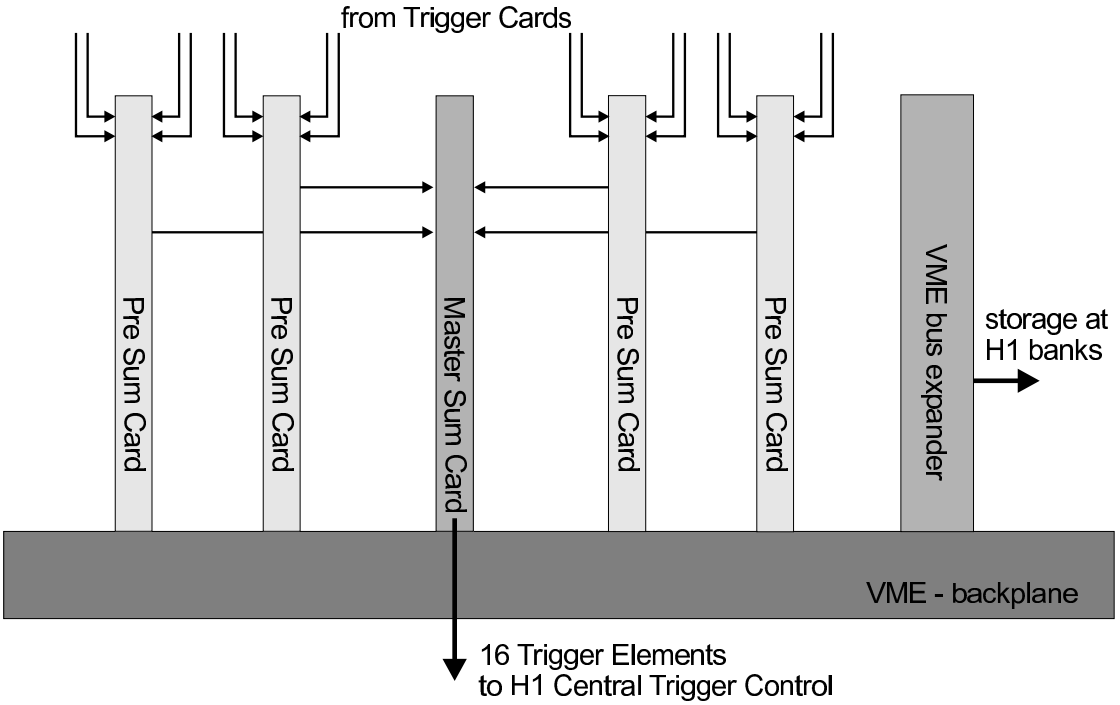


Fig. 6.12: Overview of the CIP2k sum crate. The information of each set of four trigger cards is processed in the pre sum card and given to the main sum card. The main sum card delivers 16 trigger elements to the central trigger control (CTC). A VME bus expander connects the sum crate to the trigger crate CPU in trigger crate 0.

6.3.1 Sum Card

The sum card was developed in cooperation with the electronics workshop of the university of Heidelberg (internal identifier: DL535). The sum card is built around a single FPGA (same type as in the trigger card, Altera APEX 20K400). Moreover, it uses the same PLD (Lattice ISpL 1048) for configuration and VME bus service. The sum card is equipped with a 32-bit VME bus (6 address bits in a 16-bit mode, AM 0x29). For the configuration of the FPGA, three EEPROMs (EPC2 LC20) are used. Unlike the trigger card, each sum card has a total of six SCSI connectors, five 68-pin micro sub-D (SCSI) connectors and one 50-pin micro sub-D connector. The signal levels meet the LVDS standard. Each 68-pin connector transfers 32 single channels; the 50-pin connector transports 24 channels. All important control signals are given to the sum card via the SCSI connectors (dedicated clock and fast input lines are reserved for the first four pins of the first SCSI connector). In Figure 6.13 the sum card design is shown in a schematic view. Each sum card is equipped with standard P1 and P2 VME 64x connectors. The VME 64x connector is used to support the sum card with 3,3 V power supply. An additional power supply for the FPGA kernel (2,5 V) is generated directly at the sum card with a power regulator from the 3,3 V power supply[7].

Each sum card has a piggy back board to cope with the large number of signals, which are processed in the sum card. Additional three 68-pin SCSI connectors are

[7] In fact, both types of FPGAs are used in the trigger system, the 1,8 V Apex 20k400E and the 2,5 V Apex 20K400.

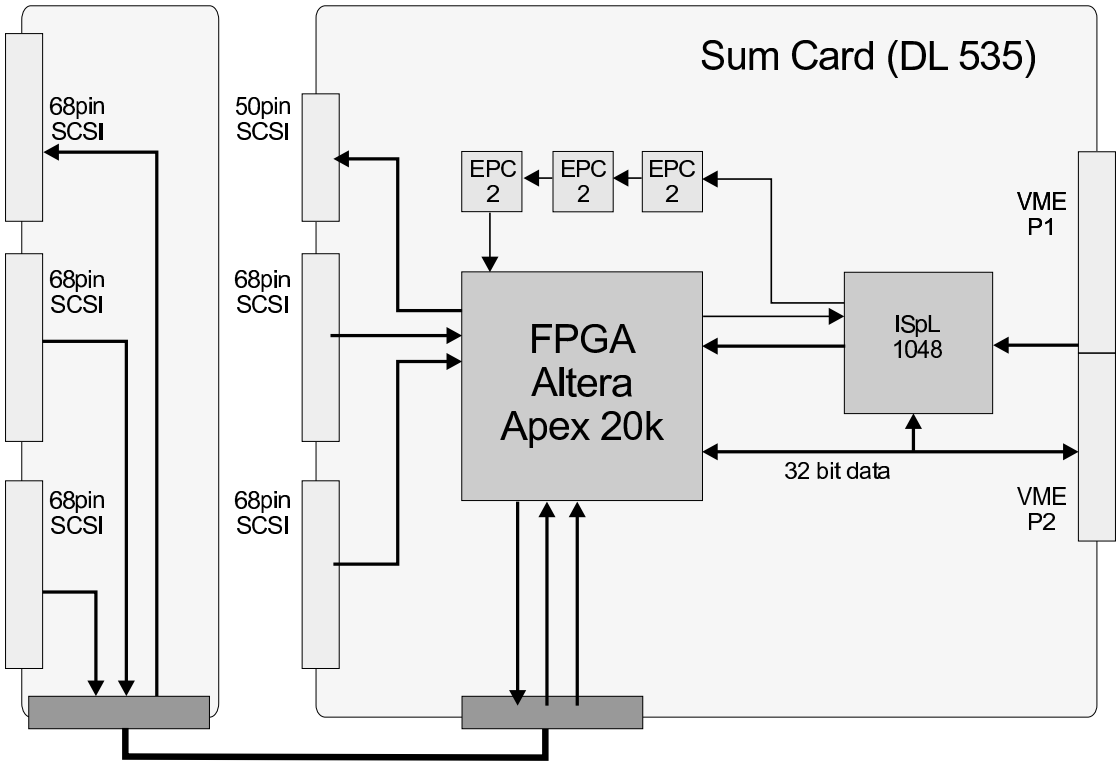


Fig. 6.13: Schematic view of the sum card. 6 SCSI connectors are connected to the FPGA. On the left side, the piggy back board is shown, providing three 68-pin SCSI connections. Three EPC2 devices (EEPROMs) hold the configured data in case of a power failure. A 32 bit VME bus supports readout and programming of the FPGA (6 address bits).

mounted on the piggy back board and are connected to the FPGA. The SCSI connectors can be used as input or output channels, depending on the LVDS converter/driver chip (input: Dallas DS90LV032, output: DS90LV031). Both converter chips can be used in a pin-compatible way. In its actual configuration, four 68-pin connectors are configured as inputs, the other two connectors as outputs. The signals from the trigger cards are received at the pre sum card by four 68-pin input connectors. The result is provided on the 68-pin output connector. At the master sum card, the signals from the pre sum cards are received by four 68-pin input connectors, the trigger decision (coded in 16 trigger bits) is provided on the 50-pin output connector.

It is worth noting that the sum cards are fully compatible to the VME standard. By downloading different FPGA codes these boards can be used as generic multi-purpose input-output or scaler boards with VME control (96 inputs and 56 outputs). The sum-card is already in use in other high energy physics experiments [89].

6.4 STC Crate

The STC crate provides control signals of the H1 trigger system to the subsystems (see Section 3.6). The STC crate consists of a STC fast card, a STC slow card, STC fanout cards and a CPU. In the CIP2k STC crate, a VME bus expander is used instead of the CPU. It expands the bus of trigger crate 0 (see Figure 6.1).

A trigger element card in the STC crate receives the trigger information, evaluated by the CIP2k trigger. This is a simple converter card to translate the LVDS signals of the master sum card (containing 16 trigger elements) into H1 conform input signal level (LV ECL).

The CIP2k STC is equipped with four specially developed fanout cards to distribute the HERA clock, reset, l3_keep and pipe_en signal to each control card.

6.5 Interface to the old Trigger System

Due to the high importance of an effective p_t-cut mechanism, the old z-vertex trigger is used in parallel to the new CIP2k trigger. After the replacement of the old CIP an adaption of the signals from the new chamber to the old z-vertex trigger hardware was developed. As mentioned in Section 6.2.4 the pad information of the first two layers are linked into the control card. 4 SCSI connectors are used to deliver the information of two φ-sectors, two layers and 120 pads in a four-fold multiplexed mode from each control card to one komposti card. In total eight komposti cards are hosted by a standard VME crate, including an additional VME CPU and the watch dog alarm system for the front end electronics of the CIP2k chamber.

6.5.1 Komposti Card

The komposti card (KC) uses the same hardware as the sum card, described in Section 6.3.1, but differs in the programming of the FPGA.

The komposti card demultiplexes the chamber data (120 pads of layer 0 and 1 of two φ-sectors) and merges it to 56 output signals provided at the two output SCSI connectors (1 x 68-pin (32 pads), 1 x 50-pin (24 pads)). Merging means that the four times higher granularity of the new chamber has to be resolved: 4 pads of the new chamber are merged to one input channel of the old system. The merged data are linked into 8 passive (without power supply and active electronic elements) converter cards that form the LVDS signal of the komposti card to a *pseudo*-analog differential input signal in two 50-pin connectors.

Together with the pad information of the two layers of the outer proportional chamber (COP), the old z-vertex trigger can be used in a 3-out-of-4 layer option [53]. The pad information of layer 0 and layer 1 is combined at the komposti card (logical AND) and the combined information is linked to the old trigger system. It contains the information of two layers. Therefore the second input (for the other layer input) is not used anymore.

In Figure 6.14, the interface of the CIP2k information to the old z-vertex trigger is shown in a schematic view.

6.6 CIP2k Slow Control

An I^2C-bus [90] based system was developed to program the registers of the CIPix [66]. The system is controlled by an 8-bit microprocessor (pic16F877) that is addressed via a RS232 serial port connection. The programming of the CIPix settings is done by

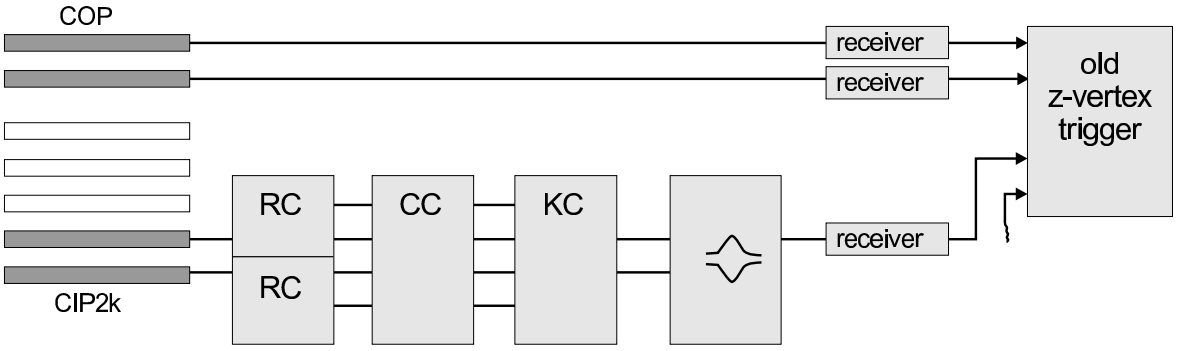


Fig. 6.14: Interface to the old z-vertex trigger. Two layers of the COP and a logical combination of two layers of the CIP2k are used in the old z-vertex trigger. The second input is not used anymore.

a Linux based PC, installed in the H1 control room with a 25 m long RS232 cable connection to the I^2C-bus CPU in the trailer (CIPix-trailer-box).

Moreover, a temperature control system was developed, using a (separate) 1-wire-bus, which is connected to a temperature measurement chip on each CIPix board at the detector. This chip sends the measured temperatures to the CPU in the komposti crate, being connected to a second serial port of the CIPix-trailer-box. The VME CPU receives the actual temperature and sends a continuous signal to a watch dog card. In case of a too high temperature or any other failure, this signal stops and the watchdog switches off all CIPix board power supplies.

6.7 Summary of the complete Trigger System Hardware

After describing all hardware components of the new CIP2k trigger system, an overview of the connections between all components is shown in Figure 6.15.

The following main tasks are fulfilled by the subsystems listed below:

CIP2k trigger: Four trigger cards deliver their φ-dependent info (z-vertex histogram, see Chapter 7) to one pre sum card with a 68-pin SCSI cable (28 channels + 4 control signals) at 10,4 MHz. In total, 16 trigger cards are connected to four pre sum cards. Each pre sum card delivers the information of one quarter of the chamber to the master sum card (also 28 channels + 4 control signals at 10,4 MHz). The main sum card produces a trigger decision (coded as 16 trigger elements + h_clk) and gives it via the 50-pin SCSI connector to a converter card, where the LVDS signals are converted to the H1 signal standard (LV ECL) and are linked to the central trigger (CTC).

CIP2k chamber readout - DAQ: The pad information of the CIP2k chamber is stored in a 32 bunch crossings deep ring buffer (pipeline), implemented in each FPGA of the trigger cards. Each trigger card provides the information of one φ-sector. In case of a successful L1 trigger decision, data taking stops and the readout of the triggered event is organized. This is done by the CIP2k data acquisition system (DAQ). The

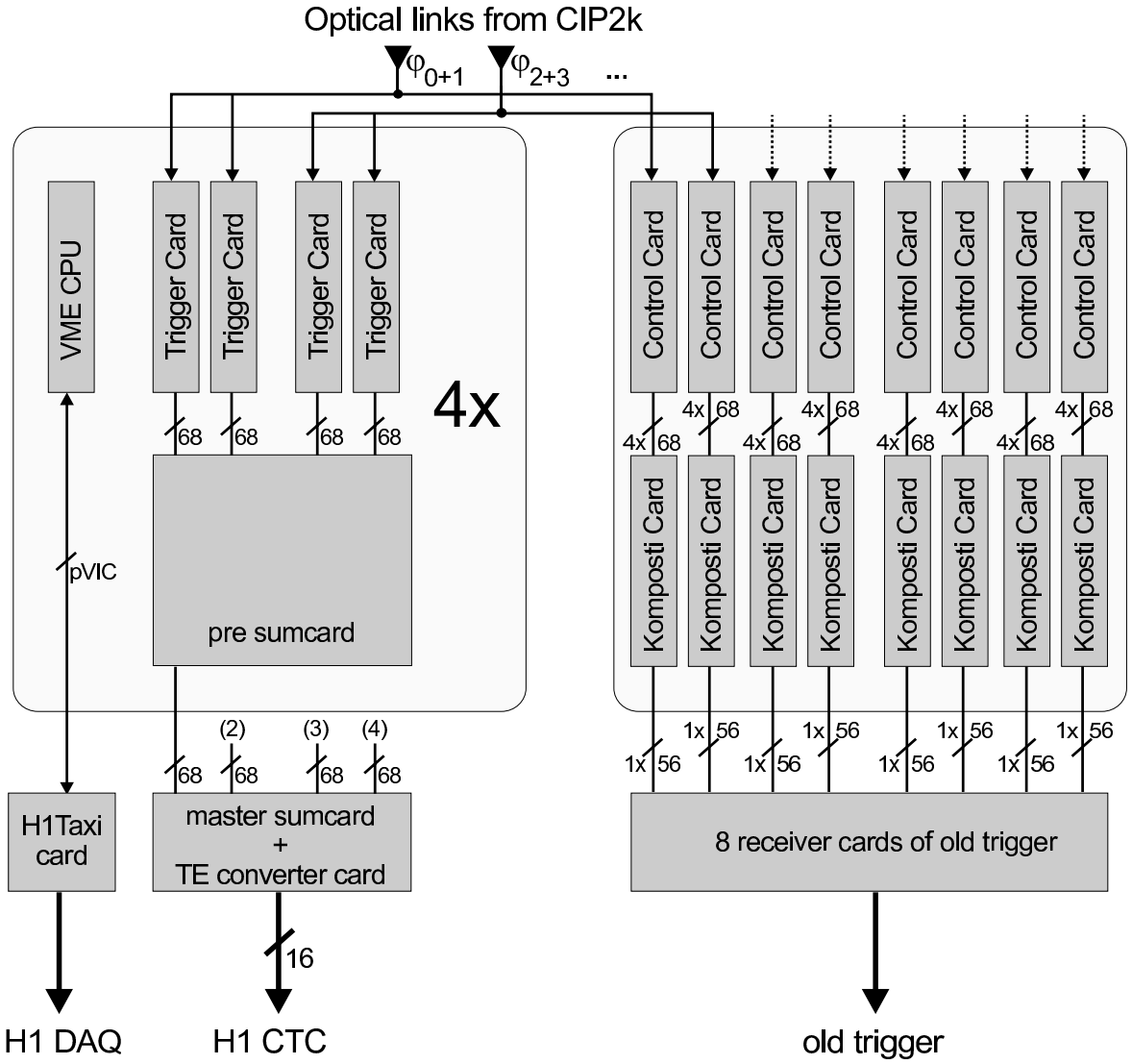


Fig. 6.15: Overview of the CIP2k trigger system hardware (see text for more details).

CIP2k DAQ system consists of four VME CPU processor boards (RIO2 8062 [77]), which communicate with the trigger cards, do the initial setup of the trigger-system and read out the stored chamber information from the ring buffer in the FPGA. Each CPU reads the data of four φ sectors. The communication between CPU and trigger card is done via the VME bus, using single cycle D32/A24 access, as described in Section 6.2.3. To ensure reading out the correct event information, a window of 5 bunch crossings is read out. The CPUs are interconnected by a pVIC (PCI Vertical Interconnect) bus system [77]. The CPU in trigger crate 0 collects the data from all φ sectors and writes it into a local event buffer. The BMA (block mover accelerator) mode of the pVIC bus is used for this data transfer. The four CPUs are synchronized, using a special protocol on the pVIC bus. With the arrival of all data packages at the main CPU, the synchronous part of the readout is finished and the pipelines are ready to be released again, typically 0.9 ms after receiving the trigger decision.

The asynchronous part of the readout starts with a data reduction process on the

main CPU. The chamber data stored as bit patterns is translated in 16 bit wide pad numbers, resulting in an effective zero suppression if the occupancy of the chamber is less than 10%. During this process, pads that are known to be bad are removed from the data. The data are then sent via a VIC (VME interconnect) link to the H1 central data acquisition system [93], [94].

Branch to old z-vertex trigger: Each control card delivers the pad information of layer 0 and layer 1 via 4 SCSI cables to a komposti card, in total eight control cards and komposti cards are used. The output of the komposti card is linked to a passive converter card, creating a differential analog signal and handshakes it to the receiver cards of the old z-vertex trigger.

6.8 FPGAs and their Programming Techniques

The new CIP2k trigger is realized in field programmable gate arrays (FPGAs). As in many high energy physics experiments, the progress in the development of large FPGAs (with up to 1 million gate equivalents) and easy-to-use development environments for the design development leads to fruitful use of this technology. In the CIP2k trigger system, Altera APEX 20k400E FPGAs are used, which just came to market when the development of the CIP2k trigger system began. A short introduction in FPGAs and programming techniques to FPGAs is given below.

Why using FPGAs: Field programmable gate arrays (FPGAs) are programmable digital logic chips that can hold almost any digital design. The structure in FPGAs is built up from a large number of small units, each containing a look-up table (LUT), a flip-flop and RAM. With these elements and special routing resources, an efficient implementation of digital circuits like binary counters and arithmetic functions (adders, comparators...) is possible. FPGAs compete with CPLDs, configurable programmable logic devices. Nevertheless, CPLDs are not RAM-based but they have re-programmable fuses to hold the configured design. Thus they can hold the configuration even at power cuts. However, it is not possible to produce CPLDs with such large structures as in FPGAs ($\approx$ some 50.000 gate equivalents [81]). In the Altera Apex 20k400 FPGA used here, 16.640 basic units, so called logic elements, LEs (see Figure 6.16 on the next page) are available (*fine-grained technology of FPGAs*). Additionally, the APEX 20k400 has 212.992 RAM units (bits) of free programmable (dual ported) memory for any kind of storage usage. Combined with 502 user I/O pins, there is enough space to host even large designs in one chip. Compared to a printed circuit in the APEX 20k400, a typical number of 600.000 gate equivalents (NAND, NOR) can be implemented.

6.8.1 Structure of FPGAs

APEX 20k devices incorporate LUT (lookup table)-based logic, product term-based logic and memory in one device. Signal interconnections within APEX 20k devices (as well as to and from device pins) are provided by the *FastTrack interconnect*, a series of

fast, continuous row and column channels that run the entire length and width of the device.

Each I/O pin is fed by an I/O element (IOE) located at the end of each row and column of the FastTrack interconnect. The IOE contains a bi-directional buffer and an I/O register. Dedicated clock lines are linked to every I/O register and to every logic element of the FPGA if a dedicated clock pin is used for the clock input (CIP2k clk_40 at pin W34 (dd clk)).

The LUT and product-term based logic is housed in a hierarchic structure. The largest block is the MegaLAB (Mega logic array block). The APEX 20k400 contains 104 MegaLABs with 104 ESB (embedded system block) cells. Signals are routed between MegaLAB structures and I/O pins via the FastTrack interconnects.

Every MegaLAB contains 16 logic array blocks (LABs), one ESB and a MegaLAB interconnect, which routes signals within the MegaLAB. The ESB can implement a variety of memory functions (RAM, dual ported RAM, ROM, FIFO, ...). Embedding the memory directly ensures a very flexible design of any kind of memory functionality. The 16 LABs per MegaLAB contain 10 logic elements (LE) each (104 MegaLABs $\times$ 16 LABs $\times$ 10 LEs = 16640 LEs). The logic element is the smallest unit of logic in the

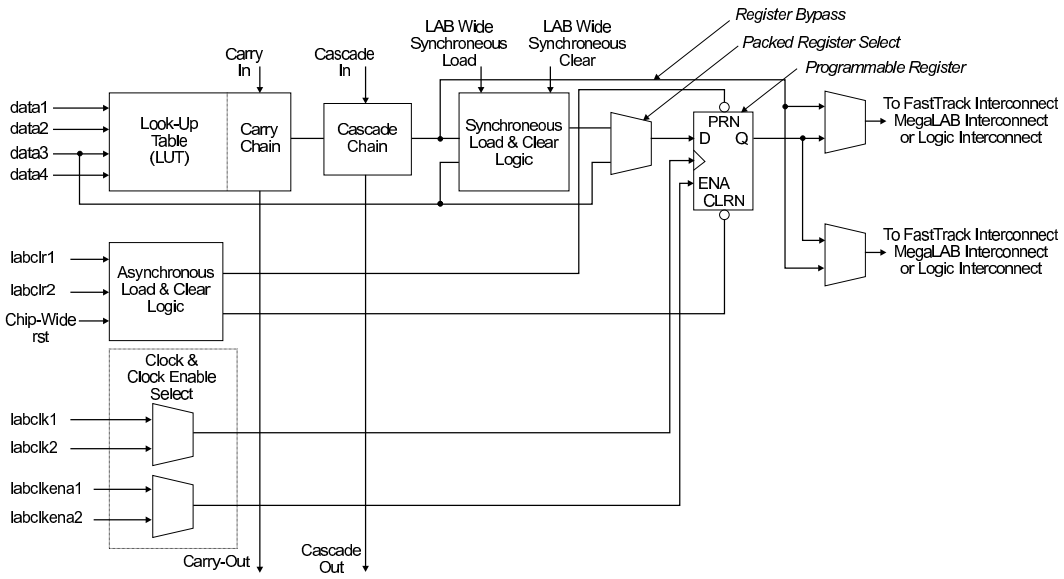


Fig. 6.16: Apex 20k logic element [80].

APEX FPGA shown in Figure 6.16. Each LE contains a four-input LUT which is a function generator on which any function of four variables can easily be implemented. In addition each LE contains a programmable register, a carry chain and a cascade chain. The LE is able to drive the local LE interconnect, Fast-Track interconnect and MegaLAB interconnect routing sources. Adjacent LEs are connected with very fast interconnects without using the local interconnect by carry chains. Therefore logical units across several LEs can be created. This is useful for large counters, adders or more complex digital designs. Equivalent to the carry chains, cascade chains can be used for very wide fan-in. For example this is used in the CIP2k trigger algorithm to combine bunches of chamber pad information at the trigger level to recognize track patterns (see Chapter 7.1).

A detailed description of the inner structure of the APEX 20k FPGA can be found in [95]. The layout of the structure inside a FPGA is performed optimally by the compiler, fitter and router provided by the FPGA manufacturer [84]. Manual optimization of the routing may be useful if nearly all hardware resources for a design are exhausted and a fitting of the complete design is not possible otherwise [80].

The configuration of the FPGA is loaded into the RAM. Then connections between the elements (gates) of the FPGA are programmed according to the pattern in the RAM (*programmable-link technique*). After finishing the configuration phase, the RAM can be used for storage duties in the hardware design. FPGAs can be reconfigured, but with a power failure, the configuration information has to be stored in the RAM again. For this purpose, EPROMs are used. With this functionality, mistakes in the design can be fixed easily.

6.8.2 Hardware Description Languages

A hardware description language (HDL) is a language used to describe a digital system that is programmed into FPGAs. The CIP2k trigger system is mainly programmed in Verilog. Verilog is one of the HDL languages available in the industry for designing ($\hat{=}$ describing) hardware. It became an IEEE standard in December 1995 (IEEE 1364-1995).

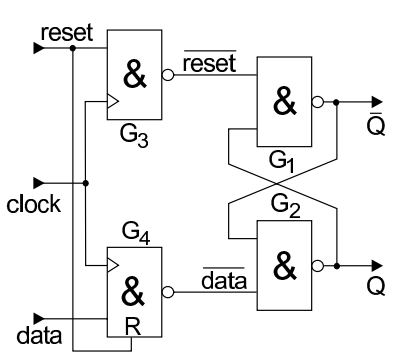
Hardware description languages look familiar to users that know other complex (high-level) programming languages like C/C^{++}, Fortran or Pascal. Bit operations, *for*-loops and *if else*-constructs are programmed in a similar way in Verilog and C. But there are some major differences in the conceptual idea, as explained below.

Description of complex hardware: With a HDL, the behavior of a digital circuit is described in a programming language like style. FPGAs work with a global clock. All gates (and LEs in LABs) are driven by this clock. This is described by the *always@(clock)*-construct in Verilog.

It can occur with the positive edge (*always @(***posedge** *clock)* or with the negative clock edge *always @(***negedge** *clock)*). Every line of HDL code inside of this always-construct is executed synchronous to the clock and every always-construct in the HDL code is done in parallel and at the same time.

As an example, in Figure 6.17 a RS-flipflop is shown. On the left side, the circuit diagram is shown and on the right side the equivalent Verilog-code. Data are stored in the register Q ($\bar{Q}$), only if the reset is not set and a positive clock edge appears. The code is embedded into a modular design, described in the next paragraph. The HDL code provides the same functionality as a real, soldered circuit.

Modular design in FPGAs: Like most other hardware description languages Verilog has a modular structure. Each module consists of a *module header* and a *module body*. The *module header* contains configuration settings and the interface definition where all inputs and outputs of the module are implemented. The *module body* contains the code or further recursive sub modules calls. The abstraction level in a module depends on the development technique. In the *bottom-up design*, the description of the design is on a gate level, equivalent to former development of circuit diagrams. With a




```
module rs_flipflop {
       clock,reset,data,Q,Q_bar};
  input clock,reset,data;
  output Q,Q_bar;
  reg Q, Q_bar;
    always @(posedge clock)
      begin
        if (reset)begin
          Q = 0; Q_bar = 1; end
        else begin
           Q = data; Q_bar = !data; end
      end
endmodule
```

Fig. 6.17: RS-flipflop: on the left side the circuit diagram, on the right side an equivalent Verilog HDL code.

description of every flip-flop and its connections, the design gets operational. But with increasing complexity of new designs the bottom-up approach is nearly impossible to maintain.

To develop large projects, hierarchical design methods are used (*top-down design*). Here an abstract model of the design is created and step by step defined more specifically. In Figure 6.18 the design process from an abstract model to the operational FPGA is shown.

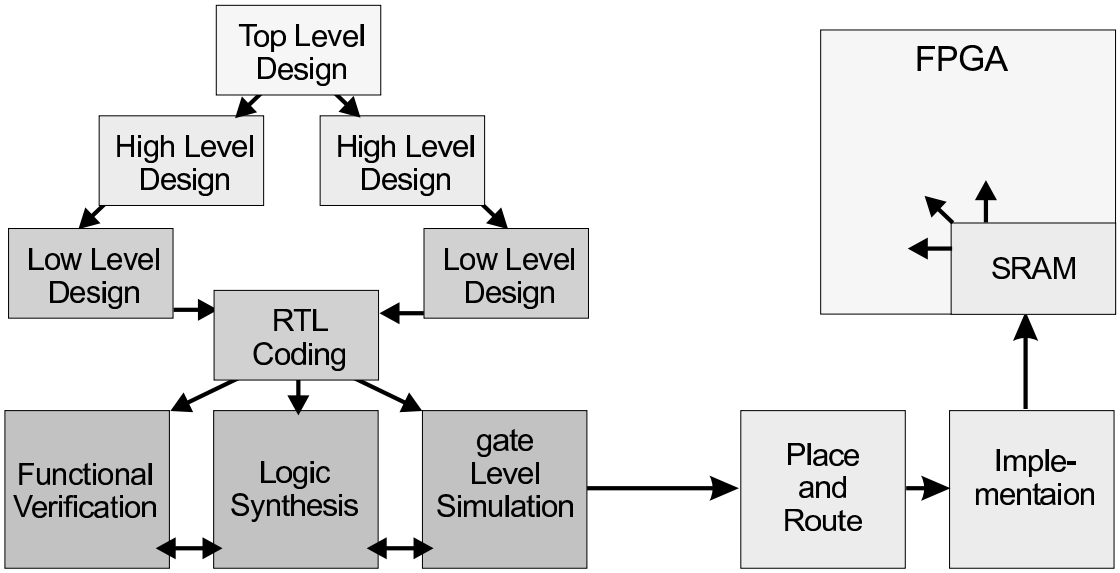


Fig. 6.18: Hierarchical design methods are used for large projects. The design process begins with a general model and gets more specific in every step. The adaption (routing, fitter) to the specific FPGA is done by the hardware development environment (see Section 6.8.3).

High level design means splitting the designs in blocks based on their function. For example, in the development of the trigger system, the design was split into a memory block and a trigger block. All necessary information for the memory was linked into the memory block, all necessary information for the trigger was linked into the trigger block. In this example, this is the top level design. Also with high level design, the

splitting of the trigger block into more specific blocks like adder, trigger algorithm block or demultiplexer (and so on) is performed. The low level design specifies how the block is implemented (state-machines, which type of memory ...). Sometimes predefined modules[8] can be used for the specified jobs.

In all levels the blocks are implemented as modules. In the top level block all inputs and outputs are directly connected to the corresponding pins of the FPGA. Blocks in the top level design are sub-modules. All information given to and taken from the block are the input and outputs of this submodule. This hierarchical design is divided, until the register-transfer level (RTL) is reached ($\hat{=}$ gate level). The register-transfer level specifies the characteristics of a circuit by operations and the transfer of data between the registers. An explicit clock is used and RTL design contains exact timing possibility. A good definition is: "Any code that is synthesizable is called RTL code". The RTL code is given to the compiler/router and can be used for simulations or can be transferred into the FPGA (see Section 6.8.3).

Parallel processes in FPGAs: A parallel pipelined algorithm can be realized with registers that store processed data in every step of the pipeline. With every clock, one step of the algorithm can be processed. With the (next) clock (edge), the data are linked to the next step of the algorithm, processed and written into the next register. The complete algorithm is processed step by step. The most important point is that in every step of the pipeline data are **processed in parallel**, synchronously to the global clock. In Figure 6.19, this is illustrated with an example of a pipeline with four steps in Verilog HDL. With a positive clock edge, a new input signal (**in 1**) is linked to the

```
module pipeline {in, out};
  input in;
  output out;
  reg A, B, C, out;
    always @(posedge clock)
      begin
        A = in;
        B = A;
        C = B;
        out = C;
      end
endmodule
```

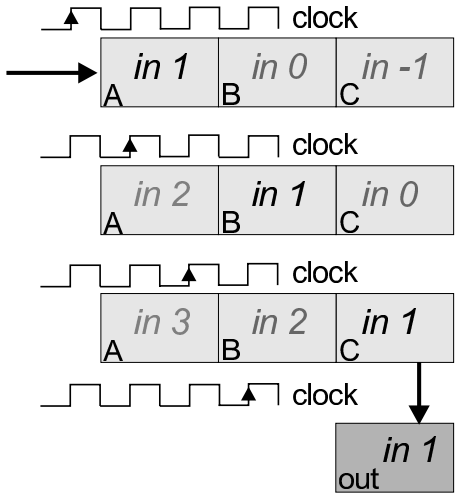


Fig. 6.19: Parallel processing with a pipeline concept can be realized with HDLs in FPGAs.

pipeline (*A = in;*). The former value of A is assigned to B (*in 0*), and so on (*in -1*). With the fourth clock, the signal at the output is *out*. A, B, C and out are register (**reg** *A, B, C, out;*). At the same time, three signals are processed in the pipeline.

[8]For nearly every function (problem), a predefined module can be found in the hardware development software, in the web or from hardware development companies, who offers complex modules for sale. Every third party module can be adapted to the design by editing configuration settings in the module header.

6.8.3 HDL Development Environment

To optimize the development process, development environments with all tools embedded are provided by the FPGA manufacturer. The user creates a project environment, containing all necessary files from the design to the operational FPGA. In this project environment, the user is guided through the following design steps:

- **Design entry:** The first step in the development of the top level design. It can be created in HDL or drawn in a schematic view (block diagram).
- **Hardware description:** By the use of the HDL, the design is described as shown in Section 6.8.2. A code checker verifies the syntax of the written code.
- **Linking and compilation:** The HDL code is linked and code on the gate level is created by the compiler, the so called netlist file. This file is independent of the FPGA type.
- **Simulation on logic level:** The logical behavior of the design can be simulated. In the logical simulations gate delays are not taken in account.
- **Pin assignment and FPGA settings:** All defined I/O pins, locked by the board design, are written into a pin assignment file. This file also contains necessary information for the operation of the FPGA (clock speed, unused pins, etc.).
- **Fitter and router:** Fitter and router translate the net list file into the FPGA, taking into account the pin assignment and configuration file. A binary file is created, which can be loaded into the FPGA. In case of a too large design or unreachable timing constraints, there is a compiler error (design will not operate).
- **Post layout simulation:** If the fitting is successful, the design is operational. To avoid testing the design with real FPGA hardware, simulations of the design can be done. These simulations take the real hardware situation into account and gate delays are known.
- **Programmer:** The programmer transfers the binary file into the FPGA or creates files that can be loaded into the FPGA.

The development of the trigger logic of the CIP2k trigger was carried out with the hardware development environment Quartus (Versions 0.99.6 to 2.0). This software is provided by Altera for the development of the APEX FPGAs. Quartus contains the complete development tools as listed above. It fulfills the development steps from the top level design entry to the operational FPGA as illustrated in Figure 6.18.

Chapter 7

The Trigger Algorithm

In this Chapter, the CIP2k z-vertex trigger algorithm is described. The algorithm is divided into steps which are processed in a pipeline as described in Chapter 6, Section 6.8.2. In Figure 7.1, the data-flow of the algorithm is shown.

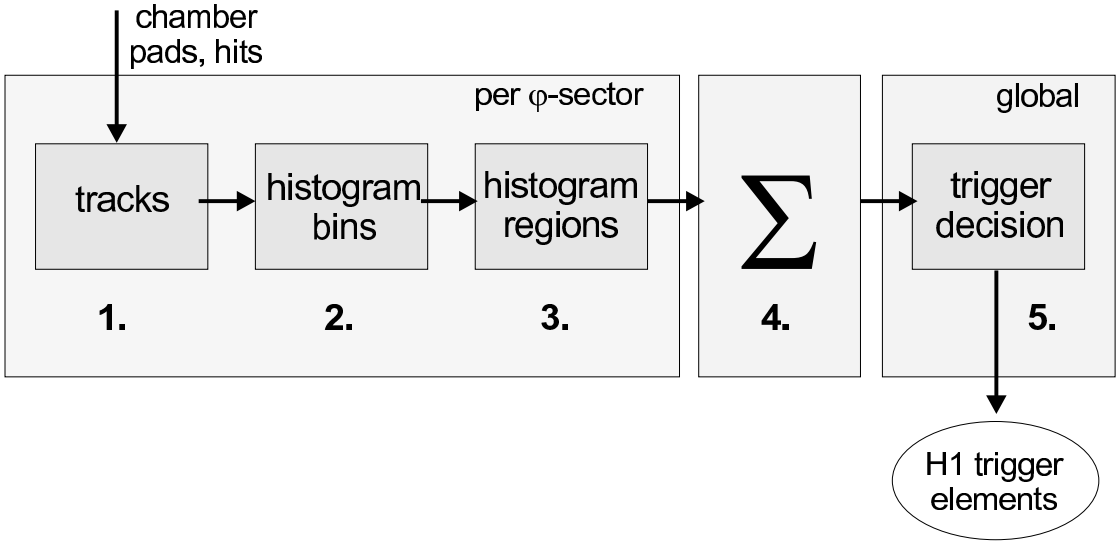


Fig. 7.1: Schematic view of the data-flow of the trigger algorithm.

The algorithm proceeds as follows:

1. Track recognition by the chamber pads.

2. Counting of the tracks in histogram bins according to their z-position.

3. Separation of histogram bins into three regions; forward, central and backward.

4. Summing of the histogram regions of all separate φ-sectors to global regions.

5. Evaluation of the trigger decision based on the track information of the three regions.

The first three steps are done in parallel for every φ-sector. The trigger decision (step 5) is evaluated based on global, φ independent track information[1].

7.1 Track Finding

In the first step, the trigger algorithm identifies tracks from the hit information of the chamber[2]. To reconstruct tracks, possible track patterns are defined. This is done by using the concept of the *local environment* (LE), as described below. If a charged particle switches on all the pads or at least a sufficient number of pads belonging to a track pattern, the particle is identified and its z-position is stored. Valid track patterns have hits in more than three of the five detector layers. The number of layers required is programmable to some extent, see Chapter 8.2.2.2 on page 96.

The local environment: All possible track patterns going through a unique pad of the middle layer (layer 2), called *cpad* (center pad), are considered. This defines the *local environment* of the cpad. Being the center of a valid track pattern, this pad is part of all possible track patterns as well as one or two pads of each other layer. In every track pattern, one pad of layer 0 and the mirror symmetric pad of layer 4 can be connected by a straight line trough the center pad in layer 2. In layer 1 and layer 3 either one or two pads are necessary to complete the track pattern (*symmetric* (one pad) or *asymmetric* (two pads) track patterns). In total, 22 possible track patterns define the complete LE, shown in Figure 7.2. A local environment consists of a total

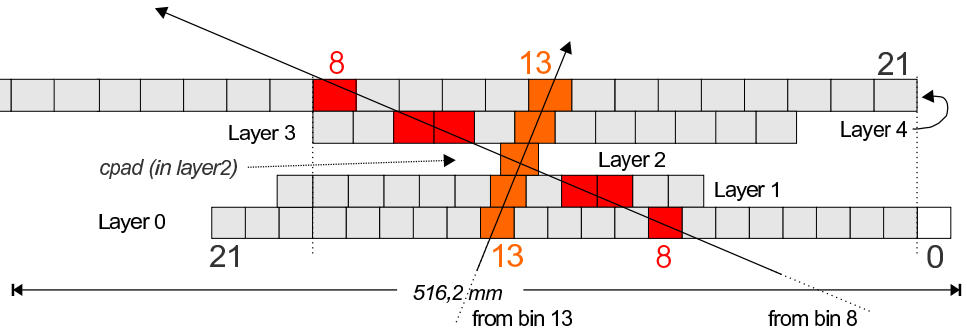


Fig. 7.2: The *local environment* (LE) of one medial pad. Due to the special *projective geometry*, pads in layer 0 are smaller than in layer,4. Two possible track patterns are shaded. One points to bin 8 and the other one to bin 13

of 69 pads.

Definition of the z-vertex histogram bins: The *projective geometry* of the chamber leads to a connection between a track pattern and z-bin. The pad size in each layer

[1] For special trigger conditions, the information of the four quarters of the chamber can also be used (cosmics trigger).

[2] In this Chapter, an ideal pad situation without noisy channels or defects is assumed. In the hardware realization, a filter algorithm is used to suppress bad channels (see Chapter 8.2.2.2 on page 96).

of the chamber is chosen such that each possible track pattern of the local environment points to a defined position at the z-axis. A total of 22 z-bins are therefore defined, each corresponding to a possible track pattern. In Figure 7.3 two tracks, originating from bin 8 and 13 are shown with the corresponding track patterns. Bin 13 is assigned to an asymmetric track pattern (two pads in layer 1 and 3 are combined), bin 8 to a symmetric track pattern. Each z-bin has a size of $z = 16.4\,\text{cm}$. With 22 bins, the z-vertex histogram is able to cover a range of $z_{total} = 22 \cdot 16.4\,\text{cm} = 360.8\,\text{cm}$ (also see Figure 7.6 on page 78).

The *local environment* is defined for every pad of layer 2, in total 106 times, corresponding to 106 pads (cpads) in layer 2. Again, the *projective geometry* of the new CIP2k chamber is useful. The same track pattern at the described *local environment* that points back to a certain bin of the z-vertex histogram, points at the same z-position in every other *local environment*. This can be easily shown, if the *local environment* for the neighbour medial pad is spread in the described way (i.e. shifted by a constant number of pads in each layer). As a result of the different pad sizes in each layer the *projection* of the same track pattern points back to the same position at the z-axis (this defines the name *projective geometry* as described in Section 5.3 in Chapter 5). This means that the same track pattern definition can be used for the *local environment* of each pad of the central layer, 106 times in parallel. In Figure 7.3 one half of one φ-sector is shown. Bin 7 is placed such that its middle position points vertical to pad 0 of every layer of the CIP. This is determined by a global bin-offset `mid`, which defines the relation between position of the bin and track pattern[3] (in steps of projective standard blocks (see Section 5.3 on page 42)). As described in Chapter 5, the

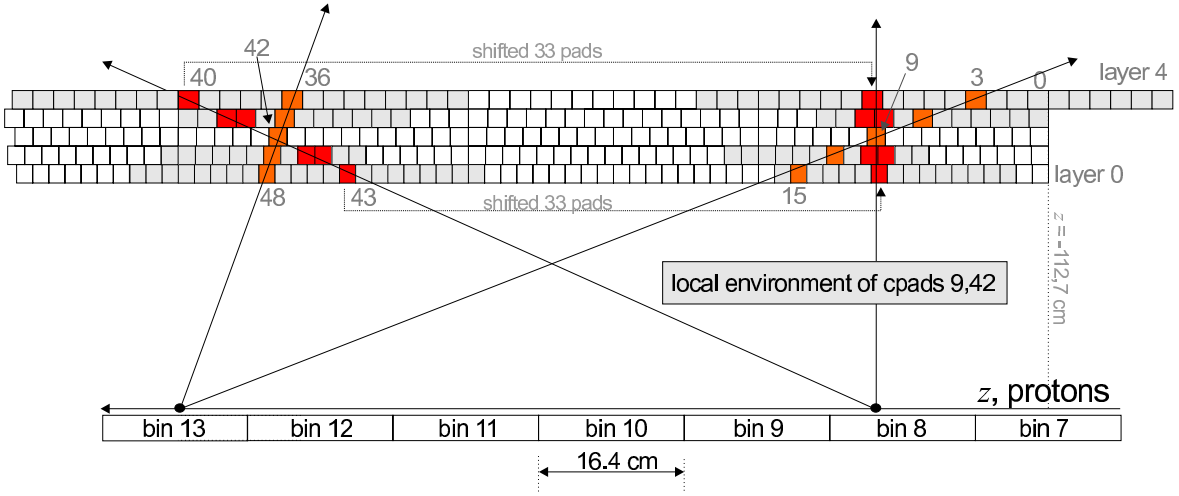


Fig. 7.3: The projective chamber simplifies the track finding: tracks from the same vertex cause similar track patterns in the local environment of each cpad. On the left side, the local environment of cpad 42 is shown. It is the same as in Figure 7.2 on the facing page. On the right, the LE of cpad 9 is shown. Exactly the same track pattern in the LE is used to identify tracks from bin 8 and 13, shifted by 33 pads to the right. (Note: this Figure is shown in a larger scale in appendix C.2.).

[3] With the variable `mid`, the center (middle) of the projection is determined.

counting of pads begins with pad 0 at z=-112.7 cm. Note that the readout electronics is mounted on the $-z$ side of the chamber. Two local environments, from cpad 9 and 42, are shaded. On the left side, the local environment of cpad 42 is shown. It is the same as in Figure 7.2 on page 74. On the right, the LE of cpad 9 is shown. Exactly the same track pattern in the LE is used to identify tracks from bin 8 and 13, shifted by $42 - 9 = 33$ pads to the right. This means that a constant offset of 33 pads in each layer is the only difference between local environment 42 and 9.

The hitlist: The logical equation

$$\text{hit flag}_{bin,cpad} = \text{L0}_{bin,cpad} \;\&\; \text{L1}_{bin,cpad} \;\&\; \text{L2}_{bin,cpad} \;\&\; \text{L3}_{bin,cpad} \;\&\; \text{L4}_{bin,cpad} \qquad (7.1)$$

is checked for each possible track pattern of each bin in all LEs simultaneously, $22_{bin} \cdot 106_{cpad} = 2332$ times per φ-sector. As described in Section 6.8.2, FPGAs are very well suited to check a large number of logical equations in parallel. The result of the hit-finding step is written into a *hitlist* containing 106×22 storage cells for each cpad (LE) and z-bin. The flag indicates whether a track was found or not. In Figure 7.4, one column (corresponding to bin number 8) of the hitlist is shown. Each column consists

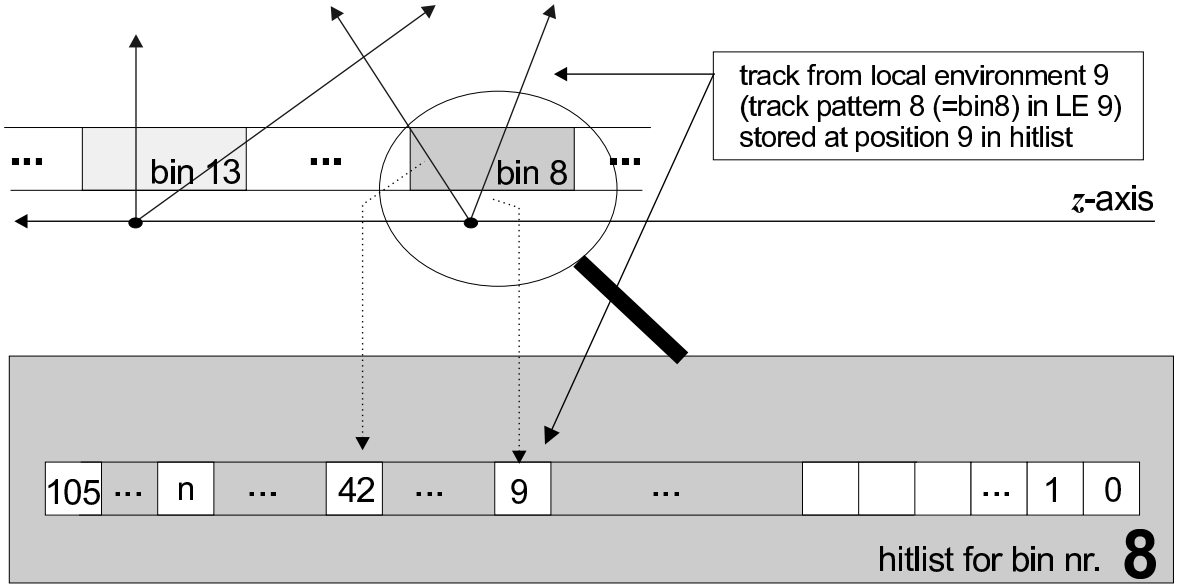


Fig. 7.4: Hitlist column 10: One flag for every *local environment* can be stored; reserved for tracks, arising from the z-position at bin 8.

of 106 hit flags. As illustrated, a track pointing to the z-position of bin number 10 identified in *local environment* 17 is stored in column 10 at position 17 in the hitlist. The track, identified in *local environment* 34, is stored at position 34.

7.2 Building the z-Histogram

The next step in the algorithm is to count all active tracks of the hitlist originating from the same z-bin. In total, the hitlist with 106 *1-bit* values has to be reduced to a

seven-bit number with values between 0 and 106 tracks. The adder uses a major part of the FPGA logic because $22 \cdot 106 = 1590$ tracks have to be summed up. The result has to be evaluated in less then 2 BCs ($\leq 2 \cdot 96\,\text{ns}$). A cascade of adders is used to meet these requirements.

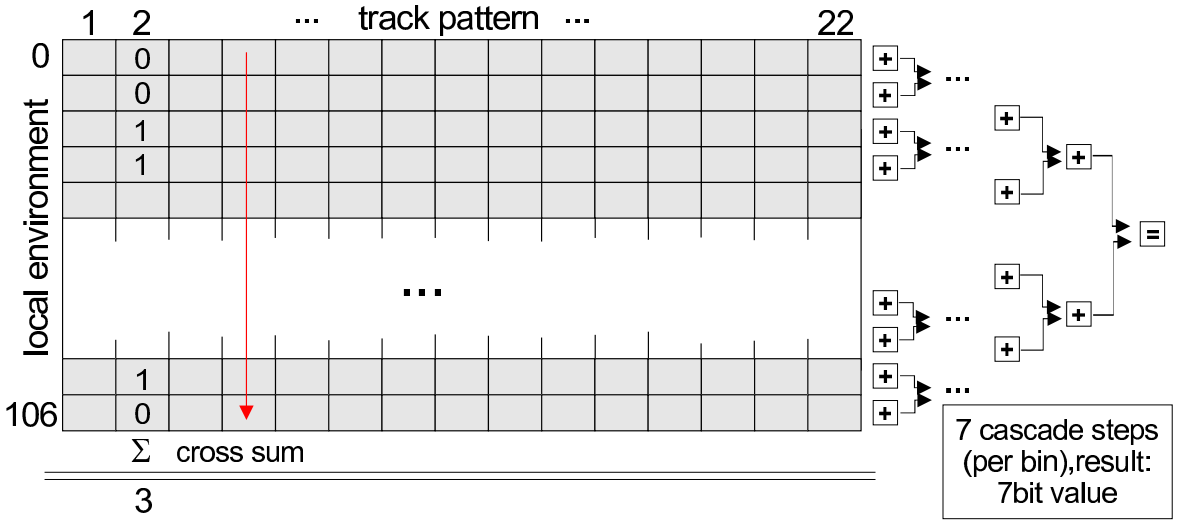


Fig. 7.5: The hitlist is evaluated by counting the tracks in the cross sum column. 106 1-bit values are summed in 7 cascade steps. This is done for each bin in parallel. The result are 22 (7-bit) values, corresponding to the number of tracks in each bin.

As illustrated in Figure 7.5, the cross sum of $22 \cdot 106$ tracks is calculated. $106/2 = 53$ two-bit adders sum three tracks each. The result are 53 2-bit numbers. In a cascade of seven steps, a 7 bit number is generated for every bin, 22 times in total.

Only 15 consecutive bins, starting at the bin-offset `mid`, are used for the trigger decision. Thus 15 of the 22 bins are selected and represent the ***z*-vertex histogram** for one φ-sector. The selection of the 15 out of 22 bins is programmable in a z-range of 360.8 cm from -235.4 cm to $+125.4$ cm.

For monitoring and optimization purposes, the selected histogram is stored in a pipeline for a total of 32 BCs ($15 \times 7 - bit = 105\,bits$) for each φ-sector. It can be read out via the VME bus together with the chamber information of the corresponding event. Every z-vertex histogram contains 15 programmable connective bins in a range of 22 bins.

7.3 Assignment of Bins to Regions

It is not necessary to deliver a z-vertex histogram with the granularity of 15 bins to evaluate the trigger decision[4]. Instead, the bins of the z-vertex histogram are grouped into a forward region, a central region and a backward region.

A measured z-vertex distribution of ep-tracks and tracks originating from the collimators C5A and C5B (collisions with off-momentum particles) is shown in Figure 7.6 to illustrate the separation of backward tracks from central tracks. More detailed results

[4]The trigger has to separate backward tracks (background) from good ep-events

are given in Section 10 on page 129. Two examples are given: In configuration (1),

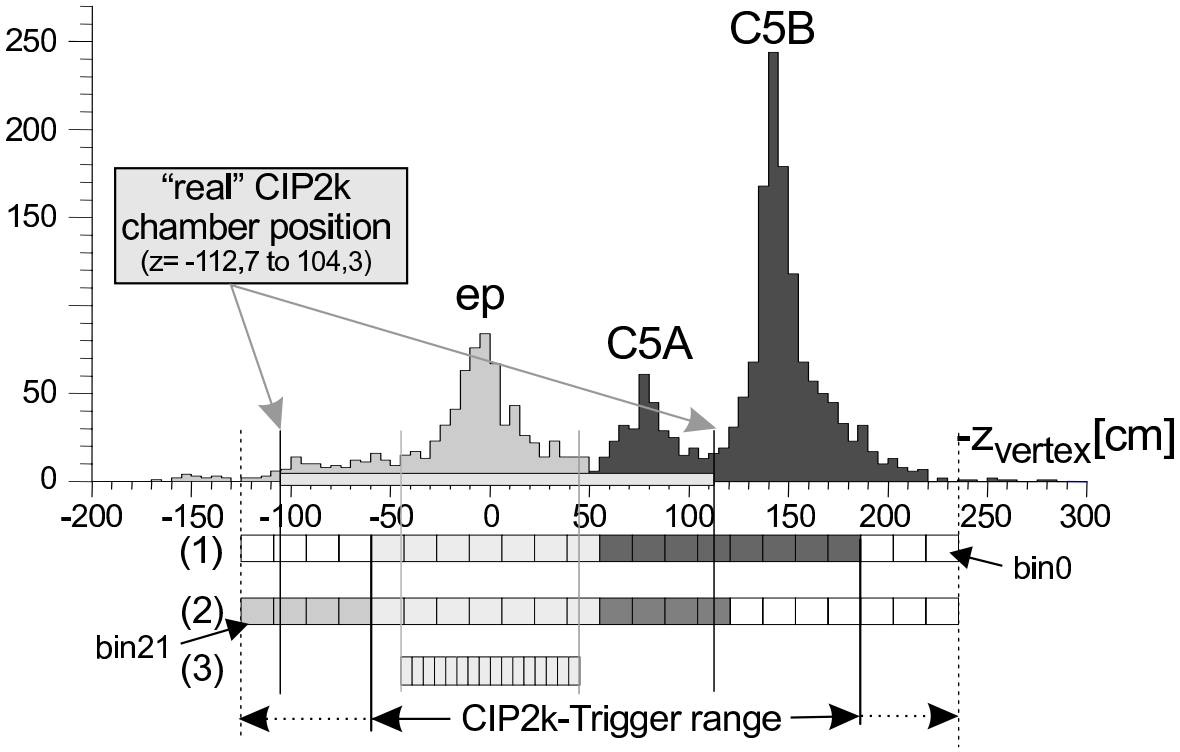


Fig. 7.6: An expected z-vertex distribution (*ep*, C5A, C5B) and the programmable regions of the CIP2k z-vertex trigger. Programmable configuration (1) is used to separate background from *ep* in luminosity runs, configuration (2) is used for cosmics detection.

there are two regions, a backward region and a central region. The z-range covered by the programmed regions is calculated by the following equations:

$$\mathrm{Lim}_{-z} = \mathrm{nrb}_{-z} \cdot \mathrm{bsize} - \mathrm{b}_0 - (\mathrm{bsize}/2) \tag{7.2}$$

$$\mathrm{Lim}_{+z} = \mathrm{nrb}_{+z} \cdot \mathrm{bsize} - \mathrm{b}_0 + (\mathrm{bsize}/2) \tag{7.3}$$

where b_0 is the z-position of the middle of bin 0=-227.2 cm, bsize is the size of a histogram bin=16.4 cm, $\mathrm{nrb}_{\pm z}$ are the numbers of the first and the last bin of the defined region and $\mathrm{Lim}_{\pm z}$ are the positions of the region from Lim_{-z} to Lim_{+z}. In example (1), the backward-region covers a range from

$$\mathrm{Lim}_{-z} = 3 \cdot 16.4\,cm - 227.2\,cm - 8.2 = -186.2\,cm \tag{7.4}$$

to

$$\mathrm{Lim}_{+z} = 10 \cdot 16.4\,cm - 227.2\,cm + 8.2 = -55\,cm. \tag{7.5}$$

The central region covers a range from -55 cm to 59.8 cm.

In configuration (2), the trigger is programmed to trigger on cosmic rays. It uses three regions around the nominal IP (fwd: 59.8 cm to +125.4 cm, ctr: -55 cm to 59.8 cm and bwd: -55 cm to -120.6 cm) and is able to cover all muons, which cross two opposite φ-sectors of the central drift chamber (see Section 7.5).

Example (3) shows the 16 bin z-vertex histogram of the old trigger system. It covers a hard wired range from -44 cm to +44 cm.

7.4 Summing Process

Once the number of tracks in the three regions of each φ-sector is calculated, they are summed up to give a global, φ independent track distribution. This is done in two steps. In the first step, the information of four φ-sectors is added. The result are z-vertex regions for the four quadrants of the CIP2k chamber. In the second step, the number of tracks of the quadrants is summed up to get the global z-regions. To keep the number of transfer cables to a reasonably low level, the number of tracks is limited to 255 in each region. If the number of tracks in one z-region exceeds 255, it is set to

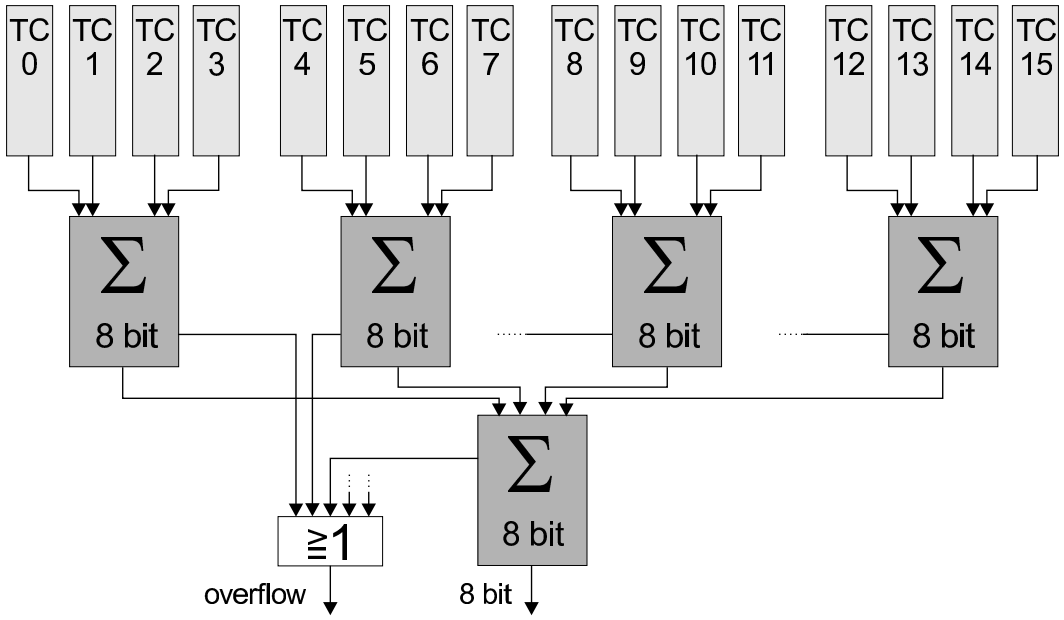


Fig. 7.7: 8 bit adder for summing tracks of φ-sectors to φ independent information for the trigger decision (shown for one z-region).

255. Additionally, an overflow bit is set. The information of the three overflow bits (ovflw_{fwd}, ovflw_{ctr}, ovflw_{bwd}) is also used for the trigger decision.

7.5 Algorithm for Trigger Decision

A trigger decision is made by analyzing the track information in the forward, central and backward region generated from the z-vertex histograms of each event in the pipeline. For analyzing the z-vertex distribution, several methods are currently under investigation. The most promising method seems to be to trigger on events with more tracks near the middle of the histogram compared to the outer regions of the histogram. This information is transferred to the central trigger system for every event. 16 trigger elements are defined in total.

7.5.1 Trigger Element Signal Definition

In Table 7.1, an overview of the trigger elements, evaluated by the CIP2k trigger, is given. All settings and values of the trigger element definition are hard coded into the

FPGA of the (master) sum card. Modifications are possible, depending on investigations of the H1 hardware situation and requests of the physics working groups of H1. The latest settings are written into the H1 data bank.

number	trigger element id	full name	meaning
1	CIP_T0	CIP2k t_0	event timing
2	CIP_T0_NEXT_BC	!CIP2k t_0	not event timing
3	CIP_SIG	significance	CIP_SIG · more
4	CIP_SIG[1]		tracks in ctr region
5	CIP_MUL	multiplicity	Multiplicity
6	CIP_MUL[1]		of all tracks
7	CIP_MUL[2]		per event
8	CIP_COS	!CIP Cosmics	Cosmic ray detected
9	CIP_SPARE[0]	spare	spare
10	CIP_SPARE[1]	spare	spare
11	CIP_SPARE[2]	spare	spare
12	CIP_SPARE[3]	spare	spare
13	CIP_SPARE[4]	spare	spare
14	CIP_SPARE[5]	spare	spare
15	CIP_SPARE[6]	spare	spare
16	CIP_SPARE[7]	spare	spare

Tab. 7.1: Overview of the CIP2k trigger elements: The studies presented in Chapter 10 have been done with trigger elements 1-8.

T0 information (CIP_T0, CIP_T0_NEXT_BC): An important function of the CIP2k trigger is to provide a t_0-signal. It is used in subsystems, especially the drift chambers, as a timing reference. The trigger system is well suited to decide, in which bunch crossing the event occured. `T0` is set to 1, if at least one track is seen in the central region. Additionally, the same `T0` signal is given exactly one bunch crossing earlier in the second bit of the 16-bit trigger element word. This signal is called `T0_next_BC`. If a trigger comes at the same time as the `T0_next_BC`, it has the wrong timing and can be rejected. Instead, it will be recorded in the following bunch crossing, as long as it is still active.

Significance of tracks from the central region (CIP_SIG): Two trigger elements (4 possible states) are used to give the ratio between the number of central tracks versus the number of background (bwd + fwd) tracks:

$$N(ctr) > S \cdot (N(bwd) + N(fwd)). \tag{7.6}$$

The value of S is given to the central trigger control. If $S = 0$, there are more or the same number of background tracks than central tracks. On the other hand if $S > 0$, there are S times more central tracks than background tracks, as shown in Table 7.2.

sig	1	2	3
S	1	2	4

Tab. 7.2: The significance (sig) is set to the given value if the number of tracks in the central region is S times higher than the number of backward tracks.

Multiplicity information (CIP_MUL): Three trigger elements are reserved for an event multiplicity information. This means, that the total number of tracks, counted in the CIP2k z-vertex trigger are added:

$$N(ctr) + N(fwd) + N(bwd) > M. \tag{7.7}$$

The value M is correlated with the number of tracks given to the central trigger control in the 3-bit decoded multiplicity (mul) as described in Table 7.3.

mul	1	2	3	4	5	6	7
M	0	2	6	10	20	30	100

Tab. 7.3: Multiplicity (mul) is set to the described values if the number of tracks (M) is higher than the presented threshold.

Cosmics trigger (CIP_COS): The CIP2k trigger is able to recognize cosmic rays by using the region information of the quarter sum cards. A cosmics trigger is defined as shown in Figure 7.8. This trigger looks for coincidences of tracks in separate quadrants

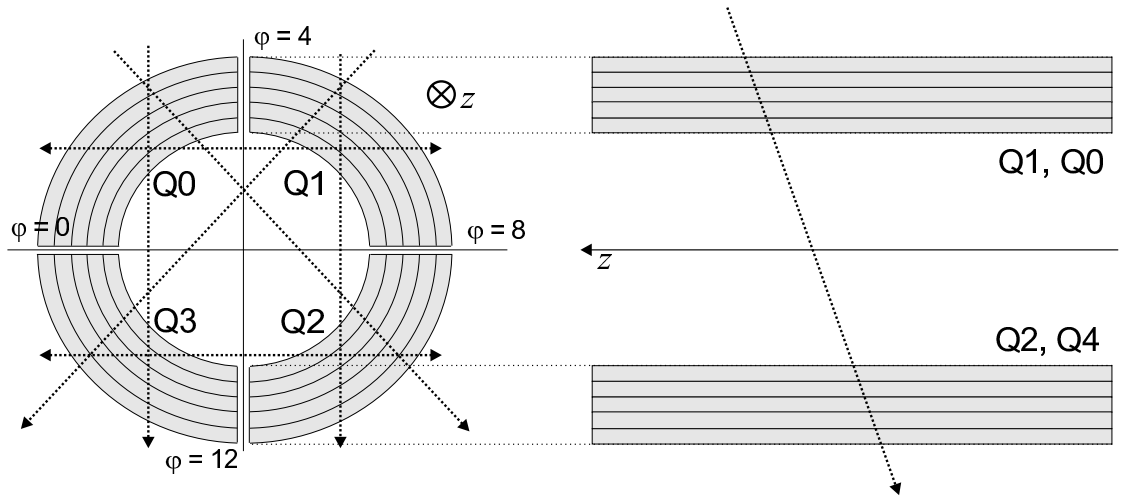


Fig. 7.8: Cosmics trigger: Tracks that cross at least two of the four quadrants (in any combination) are recognized as cosmics.

of the chamber. The trigger requires eight out of ten layers and is able to detect cosmics or other events with tracks in separate quadrants in each combination of the four quadrants. This is illustrated by the dashed arrows in Figure 7.8.

The studies presented in Chapter 10 have been performed with the presented trigger elements (TEs 1-8). The usage of trigger elements 9-16 is still subject of investigations. Possible applications are as follows:

- **Enhanced multiplicity definition:** It is planned to expand the multiplicity information to four trigger element bits. Thus a more detailed separation of track numbers in the range above 100 (tracks) is possible. Moreover, this classification should be programmable.

- **Significance of backward region tracks:** For the ratio between the number of backward and central tracks a significance definition is available:

$$N(bwd) > k \cdot N(ctr). \tag{7.8}$$

  Here a value $k > 0$ indicates that there are k times more backward tracks than central tracks. Useful settings for the value k are subject of the current studies, as well as investigations of values of the significance of the central region tracks and the multiplicity.

- **Overflow information:** The information of the three overflow bits ($ovflw_{fwd}$, $ovflw_{ctr}$, $ovflw_{bwd}$) can be used for a validation of the trigger information (i.e. $(CIP\_SIG > 2)$ && $(!ovflw_{ctr})$).

- **Confidence information:** Four bits are reserved (bit 13-16) for a possible evaluation of a confidence information of the presented trigger information. Values between 0 and 15 indicate, how sure the CIP2k trigger is of its trigger decision. The actual definition of the confidence information is subject of ongoing studies.

Signal delay: To simplify the tuning of the timing of the trigger element output of the CIP2k trigger system, two delay circuits were developed. The first delay circuit delays the input clock (hck), coming from the STC crate. This delay is described in Section 6.2.4. Additionally, a programmable delay is contained in every FPGA of the sum cards. With this programmable delay, the complete trigger element information can be shifted in 25 ns steps by up to 200 ns corresponding to 2 BCs without shifting the global CIP2k clock.

Chapter 8

The Trigger System Firmware

By using large FPGAs the trigger algorithm is implemented in a flexible way. A clear separation of hardware and software/firmware development is achieved. While the hardware development was finished in 2001, the software/firmware is improving continuously. This is one of the biggest advantages of the system-on-a-programmable-chip ((SOPC), [96]) based trigger system. In this Chapter the firmware[1] is described which is programmed into the FPGAs with hardware description languages (HDL, see Section 6.8.1).

8.1 The Firmware Concept

The complete trigger algorithm is distributed over four types of differently configured FPGAs. The data is shifted through the FPGAs according to the algorithm described in Chapter 7. In Figure 8.1, the firmware overview is shown.

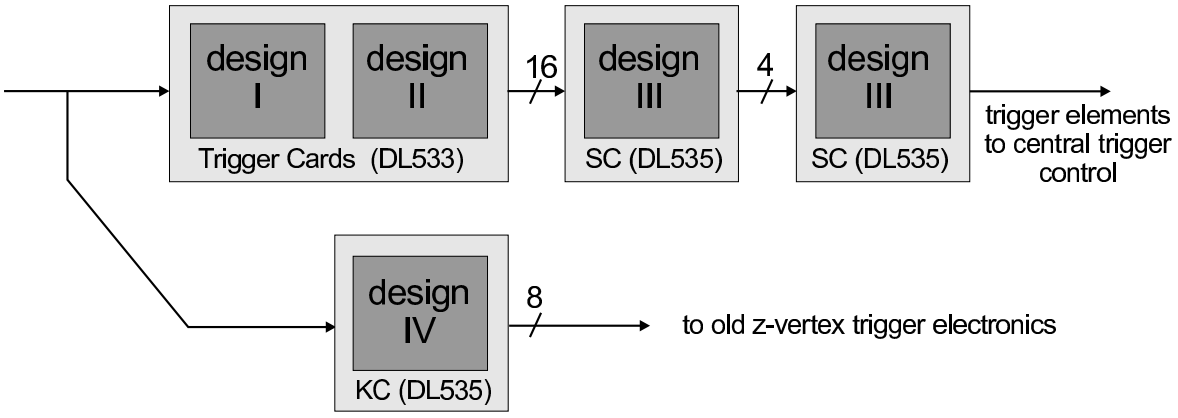


Fig. 8.1: Overview of the firmware code: Two FPGAs are programmed on the trigger card (design I and design II), with a slightly different design. Design III is programmed into the FPGA of the sum cards. The fourth design is used in the komposti card.

[1]Firmware is the middleware between hardware and software. The result of the hardware description design is the firmware.

- Design I in the FPGA I of the trigger card (type no. DL533-1), project name: `CIP2k_TC_FPGA1`. Both FPGAs on the trigger card together create the z-vertex histogram for one φ-sector. In FPGA I, the tracks corresponding to the first 60 cpads are evaluated and transferred to FPGA II.

- Design II in the FPGA II of the trigger card (type no. DL533-1), project name: `CIP2k_TC_FPGA2`. In FPGA II, tracks from the remaining cpads (46) are evaluated. In addition, the complete z-vertex histogram is extracted and grouped into regions as described in Chapter 7.

- Design III in the FPGA of the sum card (type no. DL535-1), project name: `CIP2k_SC`. A total of 16 trigger cards handshake their result to the four pre sum cards. In each of these, the data are added for a quarter of the CIP2k. The master sum card adds the histograms of each quarter to the complete histogram and generates the trigger decision. The FPGAs on the pre- and master-sum card have the same firmware design. They are programmed in different ways via the VME bus.

- Design IV in the FPGA of the komposti card (type no. DL535-1), project name: CIP2k_KC. In the fourth design, the branch to the old z-vertex trigger is realized with data from CIP2k layer 0 and 1. Pad data simulating the old CIP's geometry is generated.

The development was done with the development tool *Altera Quartus* [84]. The abstract model description (Section 6.8.2) at the top-level and high level design is realized in the block diagram file format (*.bdf), provided by Quartus. At the register transfer level (RTL), Verilog HDL is used. In the following, the four design projects are presented. All files of these projects are available at [97].

8.2 Design I: FPGA I of Trigger Card

This design implements the logic for the track finding and z-vertex histogram summing for 60 cpads of one full φ sector as well as pipelines to store the hits for readout by the data acquisition. The CIP2k_TC_FPGA1 design is organized in a hierarchic structure of modules. The top-level design is divided into two submodules according to their function (`trigger_module.bdf` and `readout_module.bdf`). Each module is divided into submodules itself. The hierarchical structure is described in Section 8.2.1. An overview of the signal data flow is given in Figure 8.2.

The chamber signals are linked to the integrated pipeline memory and to the trigger module in parallel after buffering.

In the trigger module (`trigger_module.bdf`), the first step is the demultiplexing of the pad-data into 10.4 MHz signals (`demux.v`, see Section 8.2.2.2). In the next step, they are piped into a filter module (`filter.v`, see Section 8.2.2.2). In this module every pad is compared with a database of inevitably noisy or defective pads, controlled by VME registers. After filtering, the data is piped into the track finding module (`trigger.v`, see Section 8.2.2.2), where the track finding algorithm is implemented. The resulting data is the hitlist, which still has to be counted (see Section 7.1) to obtain

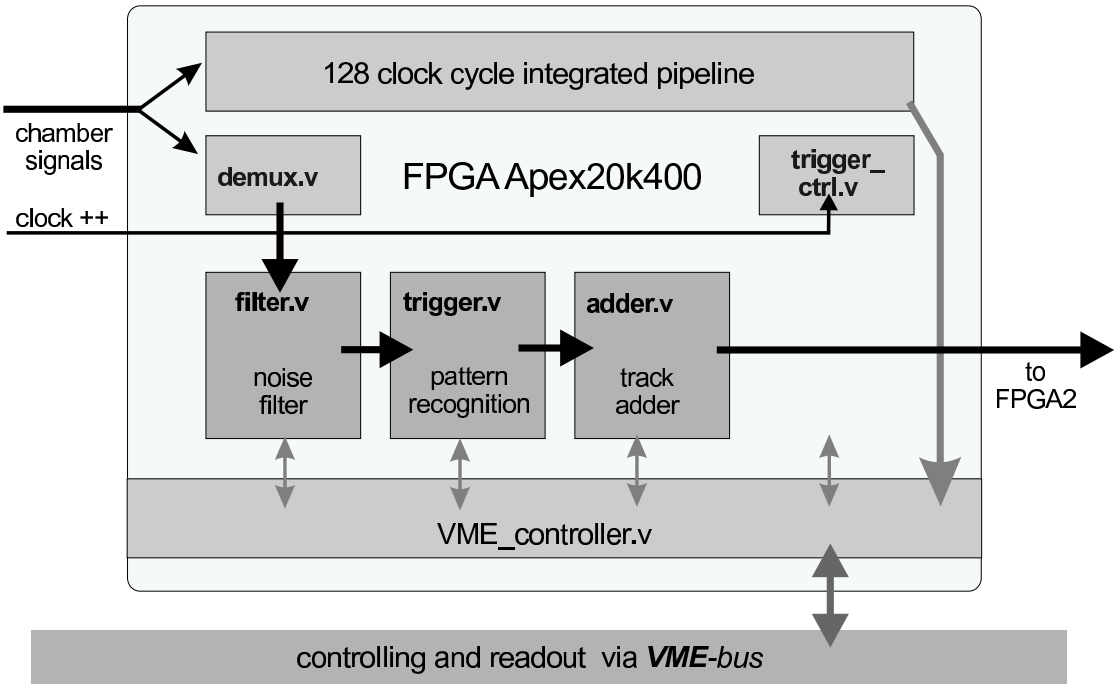


Fig. 8.2: Schematic view of the data-flow in FPGA1 of the trigger card.

the z-vertex histogram. For this a cascade adder is used (adder.v, see Section 8.2.2.2). The 60 bits are summed in 6 steps with the internal clock of 41.6 MHz (ckl_40).

Up to this point, both FPGAs are running similar algorithms. In the next step, the z-vertex histogram of FPGA I is transferred to FPGA II via a 90-bit wide bus (15 bins $\times$6 bit) as illustrated in Figure 8.2. FPGA II then adds both parts of the histogram (see Section 8.3).

In the readout module (`readout_module.bdf`), the chamber signals are stored in a 128 cycles deep ring buffer in parallel to the trigger module. Since the data at this point are still four-fold multiplexed, the 128 cycles correspond to 32 bunch crossings. The ring buffer is implemented using the integrated memory of the Altera FPGA (ESB, see Chapter 6, Section 6.8.1). The ring buffer can be read over the on-board VME interface.

If there is a positive trigger decision, a part of this ring buffer has to be read out, indicated by the pipe_en signal (see Section 6.4) from the H1 central trigger logic. The logic of starting and stopping the pipeline and preparing the data for VME readout is controlled by a finite state machine (VME_controller.v, see Section 8.2.2.1).

The implementation of the integrated readout pipeline memory and the finite state machine is the same in both FPGAs, except for the additional readout feature of the complete z-vertex histogram (see Chapter 7, Section 7.2), only possible in FPGA II.

All design parts that vary from Design I are described in Section 8.3 (Design II).

8.2.1 Hierarchical Hardware Description Level

To illustrate the hierarchic structure of modules in the CIP2k_TC_FPGA1 design (and also in the CIP2k_TC_FPGA2 design), the tree structure is shown in Figure 8.3 (shaded

in gray, the additional modules of design II are shown). According to this tree structure,

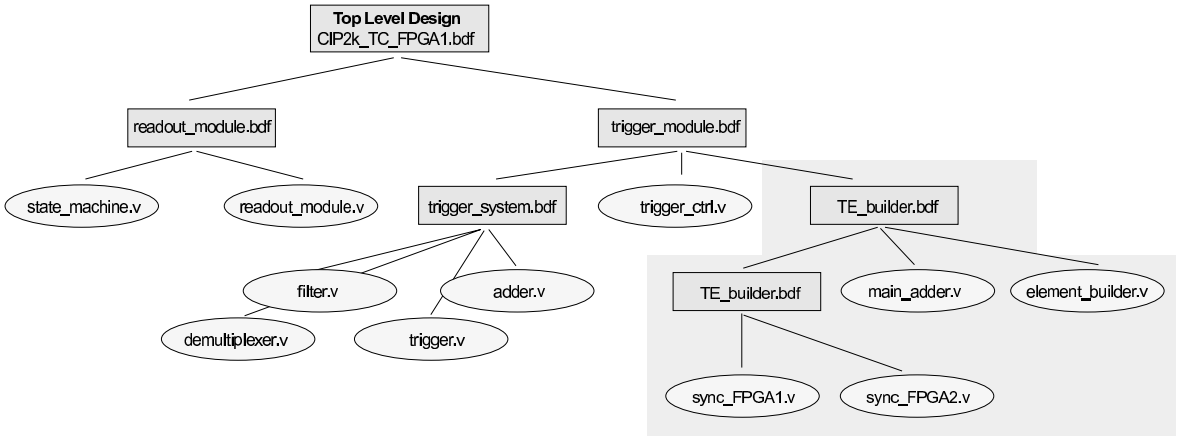


Fig. 8.3: Top-down tree structure of the design of FPGA I and FPGA II (additional modules in FPGA II are shaded). Boxes represent block diagram files, ovals Verilog modules.

the submodules are developed. Modules shown in boxes, are represented with block diagram files, modules in ovals represent Verilog HDL modules.

Top-level design: The top-level design represents the design entry. Here, all input and output signals are defined and assigned to I/O-pins. Furthermore, a first differentiation into submodules according to their function takes place. I/O-signals in inferior submodules are not linked to I/O-pins of the FPGA, unless they are linked via the top-level design to an I/O-pin. In Figure 8.4, the top-level block design is shown. Input pins are arranged at the left side of the module, output and bidirectional pins at the right side. Every I/O signal is assigned to an I/O-pin of the entire FPGA as described in paragraph **pin assignment** in Section 6.2.3 on page 50. Pins with the same functionality are grouped to bus structures, i.e., the bus `data_CIP_E0[18..0]`, where all signals of layer 0 for FPGA 1 are included. In the middle of the block diagram two modules are arranged, the trigger module and the readout module. All necessary information for the trigger algorithm is routed to the trigger module, all information for the storage of the chamber information and its readout via the VME bus is linked to the readout module. Both modules are supplied with all clock and STC signals, namely the 41,6 MHz clock (clk_40), the pipe enable signal (pipe_en), the first word signal (fst_wd) generated from the CIPix to ensure the correct demultiplexer sequence (see Chapter 6 in Section 6.2.4 and Chapter 5 in Section 5.4) and the global reset signal (n_g_rst). Additionally the L3-reject signal (l3_reject) from the central trigger is linked to the readout module to initiate an efficient stop of the readout in case of a rejection of the trigger decision at trigger level 3. The chamber signal input (`data_CIP_E0[18..0]` to `data_CIP_E4(18..0)`) is linked into a register synchronized with the 41,6 MHz clock (clk_40) to ensure a stable data to clock relation in each FPGA at each submodule across all 16 φ-sectors. After that, the information is distributed to each module. The FPGAs of the trigger card are equipped with a 32-bit VME bus (see Chapter 6, Section 6.2.3: VME bus). For the VME bus, one read (`rd_apex_VME`), two write clocks (`wr_apex_VME, wr_pulse_VME`) and 8 address bits (`addr_VME[7..2],`

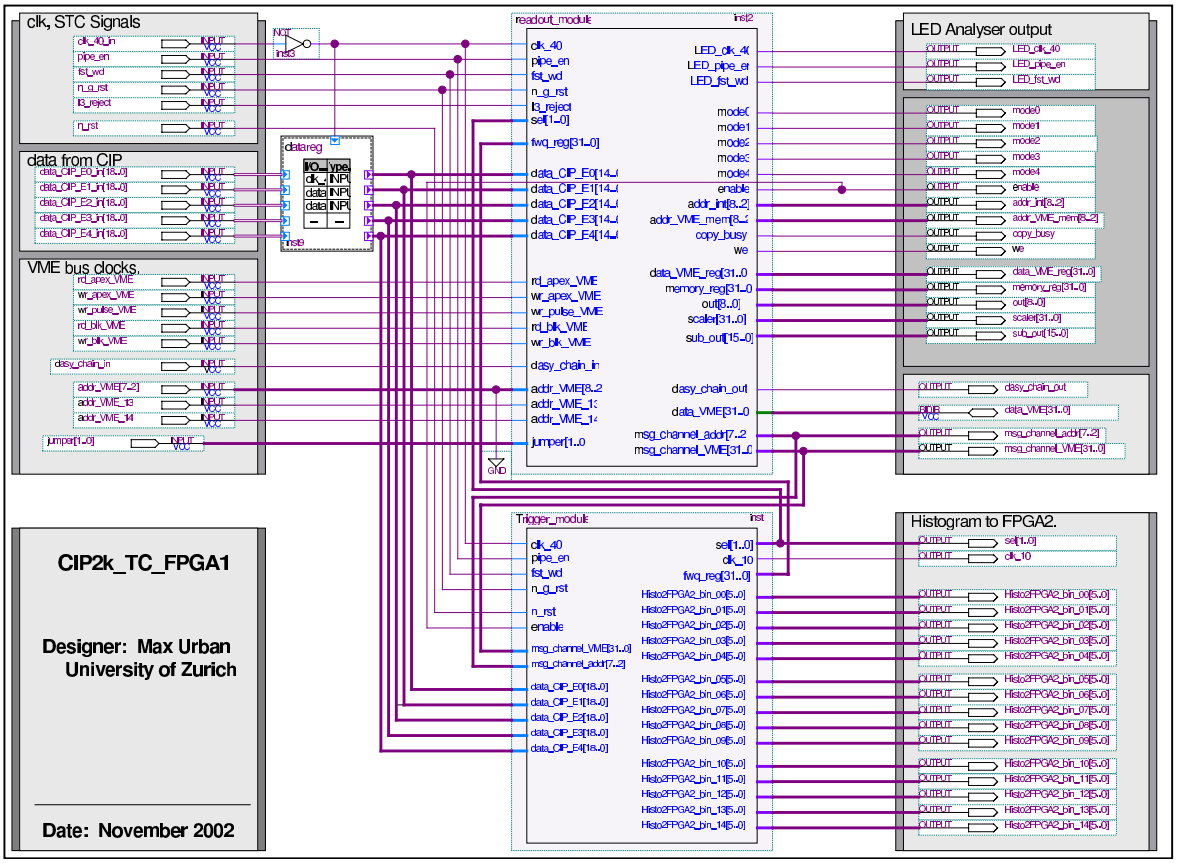


Fig. 8.4: top-level block design of the first FPGA (CIP2k_TC_FPGA1).

addr_VME_13, addr_VME_14) are delivered to the readout module besides the 32-bit bidirectional data bus (data_VME[31..0]). To enhance the readout speed, the VME block mode is prepared. For this, a block mode read and write clock is delivered and moreover a dasy chain signal [82] (dasy_chain_in, dasy_chain_out) to directly connect all FPGAs in the trigger crate (without using address modifier and base address information, according to the VME block mode convention). The block mode functionality is not implemented.

Signals for the LED indicator on the front panel of each trigger card are generated. On each trigger card (from each FPGA), an indicator of the 41,6 MHz clock (LED_clk_40), the pipe enable signal (LED_pipe_en) and the first word (LED_fst_wd) is displayed. Four additional LEDs (mode 1-4) are used to display the authors birthday. FPGA I displays the day, FPGA II the month (05.01).

FPGA I delivers the histogram information of 15 histogram bins to FPGA II, each 6 bit deep (Histo2FPGA2_bin_00[5..0] to Histo2FPGA2_bin_15[5..0]). Output signals in the dark gray shaded block are used for diagnostics in the simulation. They are not assigned to any output pins.

The readout module: The functionality of storing the chamber data and the VME bus interface is situated in the readout module. Its block diagram file is shown in Figure 8.5. The readout module is divided into two submodules. The module

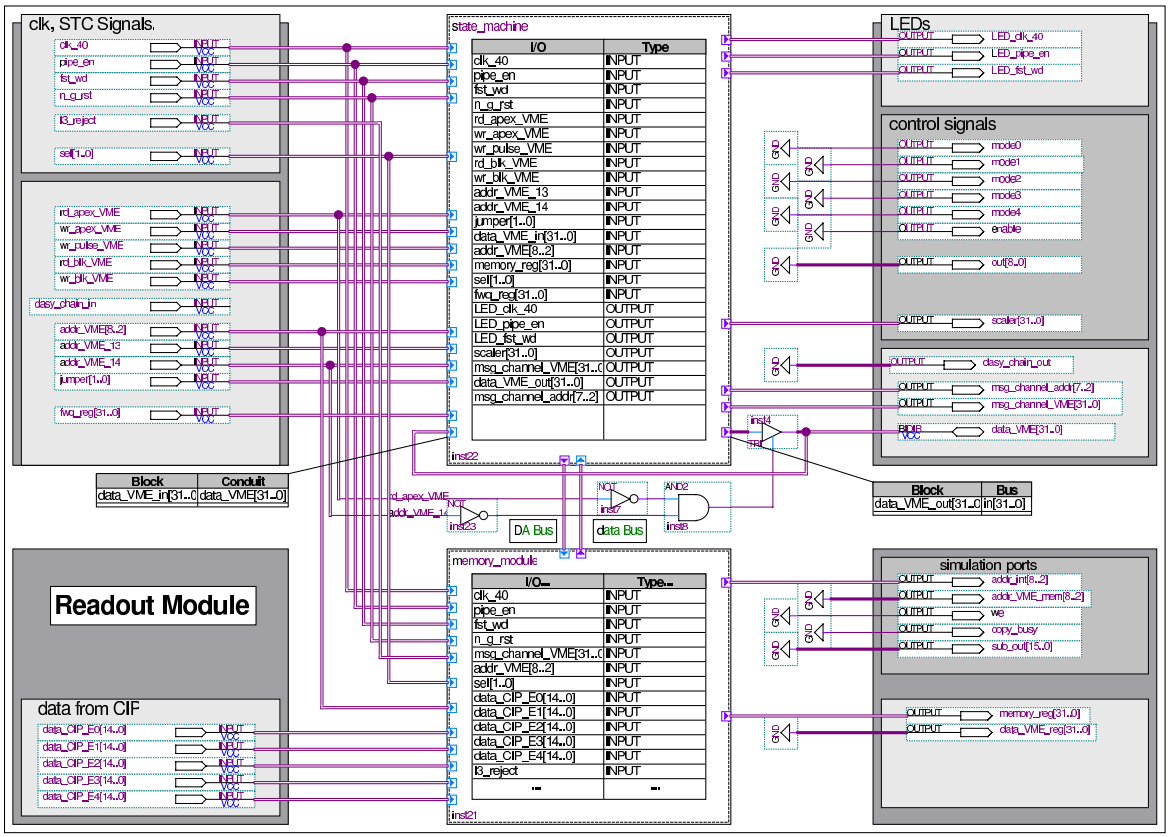


Fig. 8.5: Block design of the readout module of FPGA I (CIP2k_TC_FPGA1).

`state_machine.v` contains the VME interface controller that distributes the settings of all VME-programmable registers to the submodules. In addition, it includes a statemachine that organizes the correct readout and transportation of the chamber data from the FPGA memory to the VME CPU. The submodule `memory_module.v` contains all storage routines. Both submodules are written in Verilog HDL and are described in detail below (Section 8.2.2). To support bidirectional data transportation, the bidirectional bus `data_VME` is split to an input (`data_VME_in`) and an output bus (`data_VME_out`). With a tristate buffer (see Figure 8.5) it is only allowed to write onto the VME bus by the VME controller (`state_machine.v`) if there is no VME read access (rd_apex_VME) and the correct FPGA is addressed (!addr_VME_14).

In the block diagram, several outputs (on the right side) are set to ground level. These were used during the simulation only. To have a diagnostics functionality of the design, they were not removed completely.

The trigger module: The functionality for the complete trigger algorithm is situated in the trigger module. In FPGA I, the trigger module contains the submodule `trigger_system.bdf` and the submodule `trigger_control.v`. Representative for both designs in Figure 8.12 on page 101 the block diagram file of FPGA II is shown. In FPGA II, this module furthermore contains the histogram adder and the assignments of the z-vertex histogram bins to regions, not needed in FPGA I.

The trigger system (module): The trigger system `trigger_system.bdf` contains the core part of the trigger algorithm: The track finding (`trigger.v`) and the adding of the tracks in the hitlist (`adder.v`). Moreover, the demultiplexer (`demultiplexer.v`) and the filter (`filter.v`) are located there. The trigger system module is the same in both FPGAs except for the pad numbering and the total number of channels.

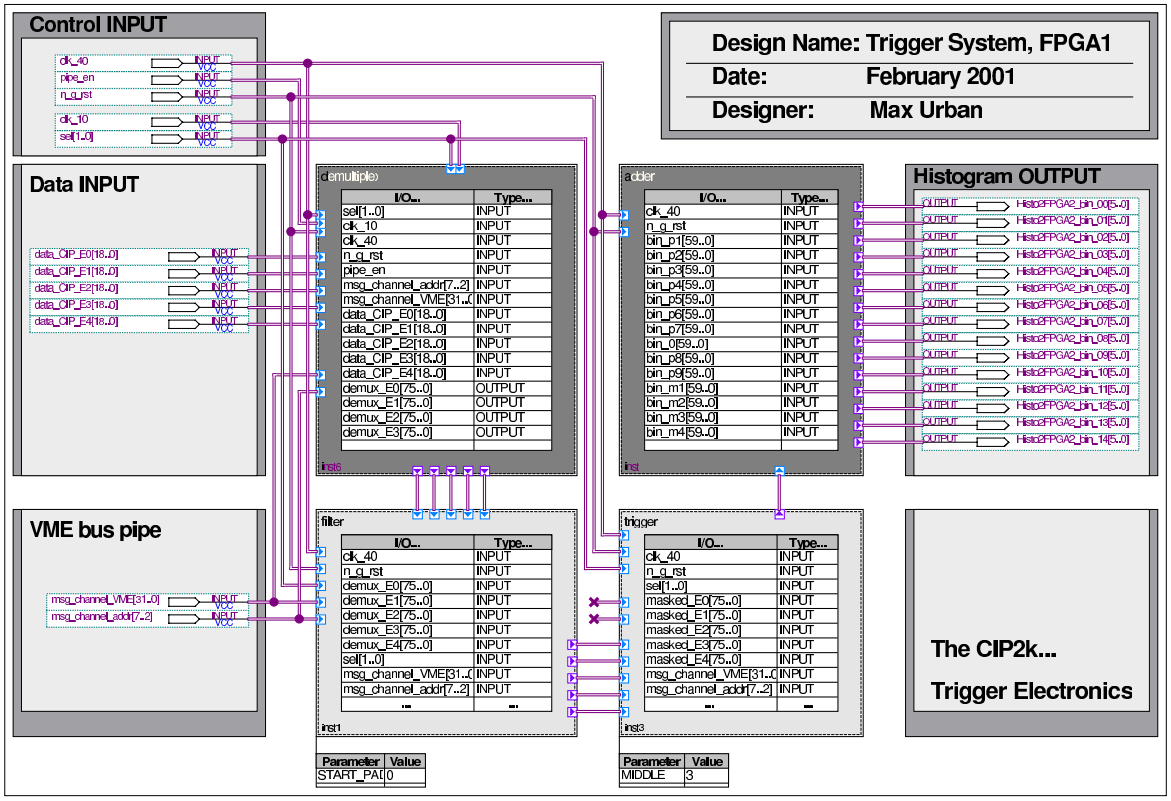


Fig. 8.6: Block design of the trigger system module of FPGA I (CIP2k_TC_FPGA1).

Figure 8.6 shows the block diagram file of the trigger system (module). The chamber signals are routed through the top-level design, trigger module and trigger system (module) directly into the demultiplexer (5× 19 bit). Demultiplexed data is transmitted into the filter (also 5× 19 bit) and after that into the trigger (again 5× 19 bit), where the track finding takes place. The hitlist information is transferred to the adder via 22×60 lines, grouped in one bus line (visible between trigger block and adder block). The adder selects 15 of the 22 bins and pipelines the result (15× 6 bit = 90 lines) to the output pins of FPGA I via the trigger module (block diagram) and the top-level design.

All submodules are connected to the message channel. It provides the settings of the internal registers, sent by the VME bus. The adder (`adder.v`) is not directly connected, as visible in the Figure 8.6. It gets all information via the trigger module (`trigger.v`, historical reasons).

8.2.2 Functional Description of Verilog HDL Submodules

8.2.2.1 Submodules for the Readout

In this Section, the submodules at the lower end of the tree structure are described. They contain the description of the hardware behavior of the trigger (and readout) system, implemented in Verilog HDL. To follow the hierarchical structure, first the submodules of the readout module are described, then the submodules of the trigger module.

VME interface: The VME interface is used to exchange information between the VME CPU and the trigger algorithm in the FPGA. For this purpose a total of 8 address lines are available in a 32-bit data mode. The accessible address range is shown in Figure 6.5 of Chapter 6, Section 6.2.3. A VME read access is possible if the `rd_apex` signal is set together with a valid address. A write access is possible if the `wr_apex` signal is set together with the `wr_pulse` signal (as described in Section 6.2.3 (Chapter 6).

The message channel: To provide the information from the VME bus to the submodules, an internal bus is used. It is called *message channel*. The message channel has the same address range as the VME bus but is synchronized to the 41,6 MHz clock. It consists of two busses, the address bus and the data bus, as shown below:

```
always @(negedge wr_pulse_VME_reg or negedge n_g_rst)
  begin
    if(addr_VME_13 & (addr_VME_14 == FPGA_FLAG) & !wr_apex_VME_reg)
      begin
        msg_channel_VME  = data_VME_in;
        msg_channel_addr[7:2] = addr_VME[7:2];
      end
  end
```

In the example shown, information of the VME bus is copied into the message channel. Since the message channel is connected to all submodules, the information still has to be copied to the registers in the submodules. All accessible registers and their VME address codes are presented in the address map shown in Appendix A.1 on page 145, A.2 on page 145 and A.4 on page 146.

Readout of the chamber data: In parallel to the evaluation of a trigger decision the chamber data are written to a cyclic memory[2], 128 steps deep (corresponding to 32 BCs×4 multiplexed channels). Thus, sufficient storage space is available to store the pad information long enough (while calculating the trigger decision) to be read out in case of a positive trigger decision. The hardware implementation of the ring memory and its VME bus readout is placed in the memory module (`memory_module.v`) and the VME module (`state_machine.v`). In total, six memory submodules are used, five submodules for five layers (with 15 channels (15 bit) each) and one submodule that

[2]Unlike in the trigger module, the information is not demultiplexed and not filtered before writing it into the memory.

stores the fst_wd information (1 bit). Each submodule requires an address pointer, a clock signal (clk_40), a read-or-write flag and the data in and out bus:

```
master_memory SUB0 (addr_int, clk_40, reg_pipe_en_neg, data_CIP_E0,
                    data\_reg\_E0);
```

Figure 8.7 gives an overview of the storage concept. The address pointer is used

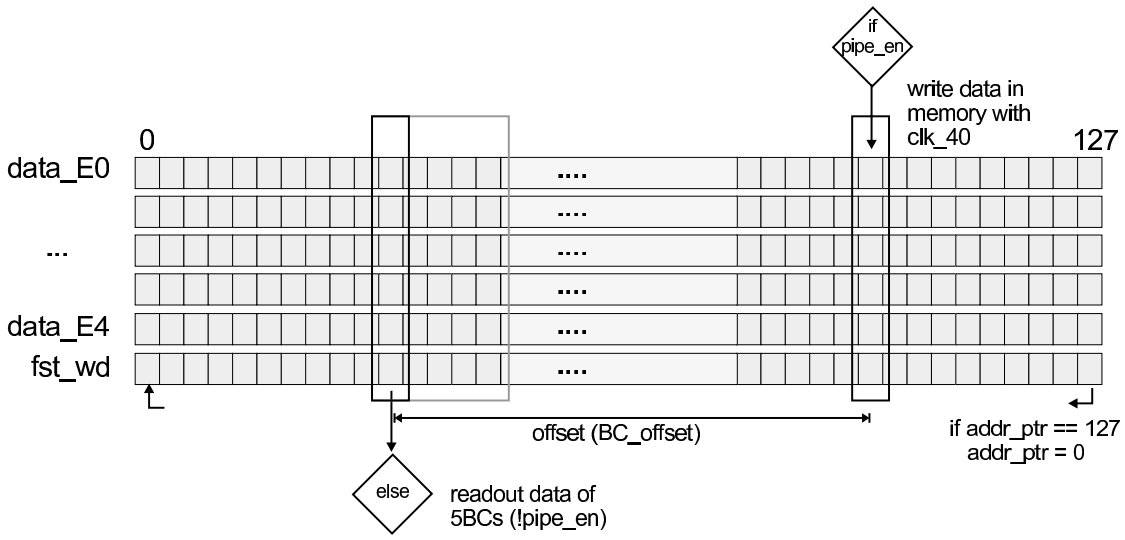


Fig. 8.7: The cyclic memory, 128 steps deep. Data is written into the memory at each clock cycle. In case of a readout (`pipe_en = 0`), the address pointer jumps to the position of the triggered event and reads out five events (±2 events + triggered event).

to create both the write address and the read address. In case of writing, the address pointer is assigned to a write pointer, otherwise it is assigned to a read pointer:

```
// devide into read or write pointer:
  always
    begin
      if(reg_pipe_en_neg)
        addr_int  = addr_int_wr; //write pointer
      else
        addr_int  = addr_int_rd; //read pointer
    end
```

The pipe_en signal from the STC crate is used to set the memory read/write access flag. Because the write process may not have written all four multiplexer steps into the memory, the pipe_en signal is synchronized to the fst_wd (sel == 0) signal:

```
// sync pipe_en with fst_wd (sel=0):
  always @(posedge clk_40)}
    if(sel == 0) reg_pipe_en = pipe_en;}
```

During the data taking period, indicated by pipe_en_reg = 1, the chamber data are written into the memory. The address pointer is incremented with every clk_40 cycle.

To avoid address changes while data is written into (read out of) the memory, all changes of the address (write and read) pointer and the read/write access flag are performed with the negative clock edge.

This is the code for the write pointer:

```
// internal write address counter:
  always @ (negedge clk_40)
    begin
      if (!n_g_rst)
        addr_int_wr = 0;
      else if (reg_pipe_en_neg)
        addr_int_wr = addr_int_wr + 1;
    end
```

This is the code to generate a pipe_en signal synchronized to the negative clock edge:

```
// register pipe_en with negedge clk_40
  always @(negedge clk_40)
    begin
      reg_pipe_en_neg = reg_pipe_en;
    end
```

In case of a positive trigger decision, the data taking is stopped (pipe_en = 0). The address pointer stops at its current position. With the next clock step, it jumps back exactly to the position of the triggered event in the pipeline (`BC_offset[8:2]`). The offset is defined as follows:
Bits [3:2]: timing within the BC (synchronous to the fst_wd),
Bits [8:4]: Pipeline position of the event (32 BCs).

The offset is terminated by the time that the CTC needs to evaluate the trigger decision of the subtrigger and the time of the CIP2k trigger to calculate the trigger decision. This offset is always the same and can be programmed into each FPGA (via VME access) once a precise measurement of the number of BCs to jump back has been done (see also Section 2, testing the DAQ).

This is the code for the read pointer:

```
// internal read address counter:
  always @(negedge clk_40)
    begin
      if (!n_g_rst)
        addr_int_rd = 0;
      else
        addr_int_rd[8:2] =  addr_int_wr[8:2] - BC_offset[8:2] +
                          ((addr_VME[8:2]*2)/5);
    end
```

In total, a maximum of 5 events (±2 events + triggered event) may be read out for a given BC offset. For each event, the information of every layer has to be read out. Every layer contains 15 channels per multiplexed step (60 pads in total[3]).

[3]In FPGA II, also 60 pads per layer are read out, independent of the projective geometry.

Because each VME word carries 32 bit, the information of two layers (one multiplexer step) can be read out in each readout cycle. The 16^{th} bit carries the fst_wd information, stored at the same time as the data. It should be on for the first multiplexer step. Otherwise, the VME readout has to be shifted or a timing error during the writing process occurs.

The 32^{st} bit carries the `l3_reject` signal. It is directly copied to the actual VME word during the readout of the pipelined chamber data. With the `l3_reject` signal, stopping the readout is indicated in case of the trigger rejection on trigger level 3 (see description of trigger signals in Chapter 3, Section 3.6).

The VME master starts the readout procedure by setting the first VME address. The data of the corresponding VME address (addr_int_rd[8:2] = addr_int_wr[8:2] - BC_offset[8:2] + ((addr_VME[8:2]*2)/5);) are copied into a 32-bit word (`memory_reg[31:0]`) with a sequence, shown in Figure 8.8.

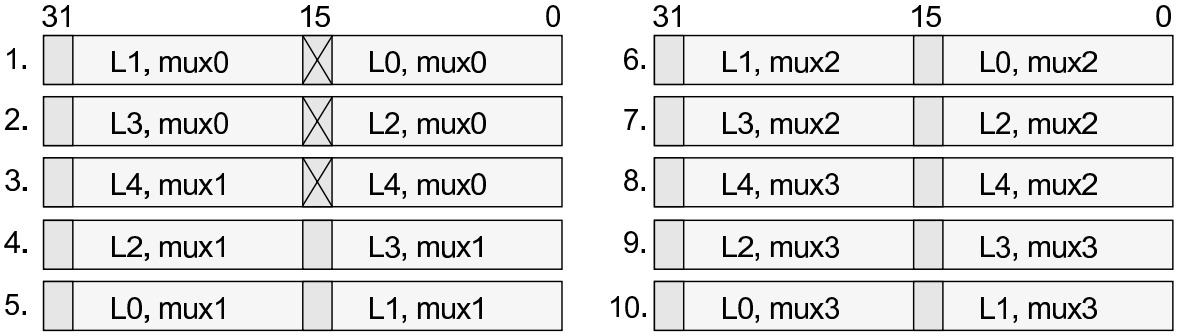


Fig. 8.8: 10 32-bit words are necessary to read out one event per FPGA. Each 15 channels of one layer are assigned to the 32 bit VME word in the shown sequence. The 15th and the 31^{st} bit carry the fst_wd information, stored at the same time as the data. It should be on for the first multiplexer step.

Then the 32-bit word is handshaked to the VME accessible register in the memory module (`state_machine.v`) and read out. The readout of chamber data (also) requires the setting of address bit 13 (see address map in Figure 6.5 in Chapter 6, Section 6.2.3):

```
always @(negedge clk_40) // assign outputs to VME bus
  begin
    if ((!addr_VME_13) & (addr_VME_14 == FPGA_FLAG) & (!rd_apex_VME))
      data_VME_out[31:0] = memory_reg[31:0];
    else if (addr_VME_13 & (addr_VME_14 == FPGA_FLAG) & (!rd_apex_VME))
      // readout of other registers ...
  ...
  end
```

Since the VME address is available at least two clock cycles (of the ISpL VME bus controller) earlier than the generated readout signal (rd_apex, see VME readout cycle, Chapter 6, Section 6.2.3), the corresponding data are stable at the 32-bit VME word `data_VME_out[31:0]` and can be fetched by the VME bus master.

To identify the right information in the readout sequence (5 steps for one half of the event), the following code is used to resolve each step with the modulo function:

```
// pipeline data to VME allocation
// (10 x 15bit -> 5 x 32bit for one half of event)
  always @ (posedge clk_40)
    begin
      if (!n_g_rst) memory_reg = 0;
      else
        begin
          case (addr_VME%5)
            0:
              begin
                memory_reg[14:0]  = data_reg_E0[END_PAD:START_PAD];
                memory_reg[15]    = fst_wd_reg;
                memory_reg[30:16] = data_reg_E1[END_PAD:START_PAD];
                memory_reg[31]    = l3_reject;
              end
            1:
              begin
                memory_reg[14:0]  = data_reg_E2[END_PAD:START_PAD];
                memory_reg[15]    = fst_wd_reg;
                memory_reg[30:16] = data_reg_E3[END_PAD:START_PAD];
                memory_reg[31]    = l3_reject;
              end
            ...
          endcase
        end
    end
```

In total, 20 readout cycles are necessary for one BC and FPGA, 100 cycles for 5 BCs. For one φ-sector (both FPGAs) 200 cycles are necessary, 800 cycles per trigger crate. All trigger crates are read out in parallel.

In the same way as the pad readout, the z-vertex histogram can be read out (only in FPGA II). The complete histogram covers a 105-bit word, 15×7 bit of 15 bins. It is placed in $105/32 = 4$ VME 32-bit words and is available at the VME addresses 28-31 (see VME address map). To access the correct histogram, corresponding to the readout of the pad information, an offset (zvtx_offset) is programmable in the same way but has to be 4 BCs smaller for the same event due to its evaluation time. The z-vertex pipeline is 32 steps deep only (10,4 MHz), compared to the memory of the chamber information, which stores the data for 128 steps in a four times multiplexed manner with 41,6 MHz. By varying the offset, all 32 z-vertex histograms can be read out.

For the other VME accessible registers a full description can be found in Appendix A on page 145.

8.2.2.2 Submodules for the Trigger

Trigger control: The submodule trigger control (`trigger_ctrl.v`) was developed to create a 2-bit counter (`sel[1..0]`) from the fst_wd signal synchronized with the 41.6 MHz clock. The counter is set to 0 after a reset and synchronizes with the fst_wd, if the pipe_en bit is low. After that, it works independently as long as it is synchronized.

```
  //statemachine for demultiplexer
    reg [1:0]  sel;          // define register
```

```
always @(posedge clk_40) // clock synchronization
  begin
    if (!n_g_rst) sel = 0;
      else if (fst_wd && (!pipe_en))
        begin
          if      (fst_wd_flag == 0) sel = 1;
          else if (fst_wd_flag == 3) sel = 0;
        end
          else sel = sel + 1;
  end
```

If there is a problem with the fst_wd signal, it synchronizes again, but only, if the pipe_en signal is low, indicating that the trigger is not working (during readout). In case a wrong phase relation is detected, an error counter is incremented. It can be read out from a VME accessible register (VME_addr = 5). The phase locked relation between the first word signal and the sel-counter can be shifted in two steps, according to its start position either the fst_wd indicates the first data bit with the next clock cycle (fst_wd_flag == 3), or the fst_wd indicates the first data bit with current clock cycle, (fst_wd_flag == 0)). The two bit `sel`-counter is used to resolve the four multiplexed steps at the demultiplexer and moreover to indicate the position in the actual BC in other modules (filter, adder, trigger, readout_module, state_machine).

Demultiplexer: The demultiplexer module (`demultiplexer.v`) is used to demultiplex the fourfold multiplexed chamber data, routed into FPGA I (`data_CIP_E0[18:0]`[4] to `data_CIP_E4[18:0]`). The result is four pads per channel. Together with the chamber data, the 2-bit counter (`sel[1:0]`) and a branch of the internal programming bus (`msg_channel_VME`) are linked into the module. For every layer, the demultiplexer sequence (0-1-2-3 or 3-2-1-0) (`dir[4:0]`) can be programmed externally via the VME-bus. The information of every pad is given to a submodule `sub_demux`, called for every channel ($19 \times 5 = 95$ times) in parallel.

```
always @(posedge clk_40)
  begin
    case (sel)
      2'b 00 : mem[0] = mux; // first step
      2'b 01 : mem[1] = mux; // second step
      2'b 10 : mem[2] = mux; // third step
      2'b 11 :               // fourth step
        begin
          demux[3]  = mux;
          demux[2:0] = mem[2:0];
        end
    endcase
  end
```

The demultiplexing process of all pads of one BC takes four steps with the 41,6 MHz clock ($\hat{=}$ 1 BC). The result of the first three steps is stored in a register (`mem[2:0]`) and is copied to the output register (`demux[3:0]`) in the fourth step. The demultiplexed pad information of all pads is given to the filter module with the next clk_40 step.

[4]There are different notations for the bus range in the bdf-file ([x..y]) and the Verilog HDL language definition ([x:y]).

Filter for noisy pads: To avoid a wrong track recognition due to noisy pads (see Chapter 4), all pads known to be defect are switched off in the filter module (`filter.v`). Every pad can be forced to **on** or **off** or it may be passed without any change. For this purpose, a so called *defect pad mask* is created. The pad mask determines the handling of defect pads of the CIP chamber. It is written into a H1-bank[5], called CRMD (CIP2k Trigger Set Up). At every startup of the trigger system, it is programmed into reserved registers in the filter module via the VME bus. For every layer, two registers contain the information of the pad mask (`F0_Ex_mask` and `F1_Ex_mask`). F0_Ex_mask is **1** except for the bits to be forced **off**. F1_Ex_mask is **0** except for the bits to be forced to **on**. The output data `masked_Ex` are calculated layer-wise as follows (for layer 0):

```
masked_E0 = ((demux_E0 & F0_E0_mask) | F1_E0_mask);
```

The "masked" data are handshaked to the track finding algorithm.

Track finding algorithm: In the track finder, track patterns, recognizable by the CIP2k chamber, are defined. To optimize the amount of logic elements used for the track finding, all possible track patterns were scanned, in total 22 per cpad. The grouping of the tracks into 15 bins, corresponding to the local environment, is done in the adder module according to the algorithm (Section 7.2 on page 76).

Tracks at the border of the chamber can not be identified if several pads of the local environment are beyond the chamber. Virtual pads are defined to be able to identify those tracks nevertheless. At the left and the right border of the chamber, as many virtual pads as needed are endowed to span a complete local environment for the first (0) and the last (106) cpad. The virtual pads can be enabled layer-wise via VME-programmable registers, five bits for each side of the chamber, to complete selected tracks at the chamber borders.

Before calling the track finder submodules the pads are grouped to a vector layer-wise, containing the virtual pads and the overlap pads. Because FPGA I deals with the first 60 cpads (and FPGA II with the remaining 106-60 = 46 pads), in both FPGAs different overlap pads are needed (see Chapter 7, Section 7.1).

```
//assign array (FPGA1)
  always @(posedge clk_40)
    begin
      // VME programmable border Pads
      if (LBP_E4)   E4[18:0] = 19'h7ffff;
      else  E4[18:0] = 19'h00000;
      if (LBP_E3)   E3[18:0] = 19'h7ffff;
      else  E3[18:0] = 19'h00000;
        ...

      // pad assignment camber pads
      E4[94:19] = masked_E4[75:0];  //overlap included
      E3[94:19] = masked_E3[75:0];  //overlap included
      E2[78:19] = masked_E2[59:0];  //no overlap for cpads
      E1[94:19] = masked_E1[75:0];  //overlap included
```

[5]The H1 bank system is used by every subdetector to store all required setup information, but also for the read out data in case of a positive trigger.

```
  E0[94:19] = masked_E0[75:0];  //overlap included
end
```

Once the vectors of the layers are defined, the track finding kernel is called. There are two track finding kernels, one for the even tracks and one for the odd tracks (see definition of symmetric and asymmetric track pattern in Chapter 7, Section 7.1). The same track finding kernel is called for each of the 11 symmetric track patterns and for each of the 11 asymmetric track patterns, but with vectors shifted against each other. This is illustrated in Figure 8.9.

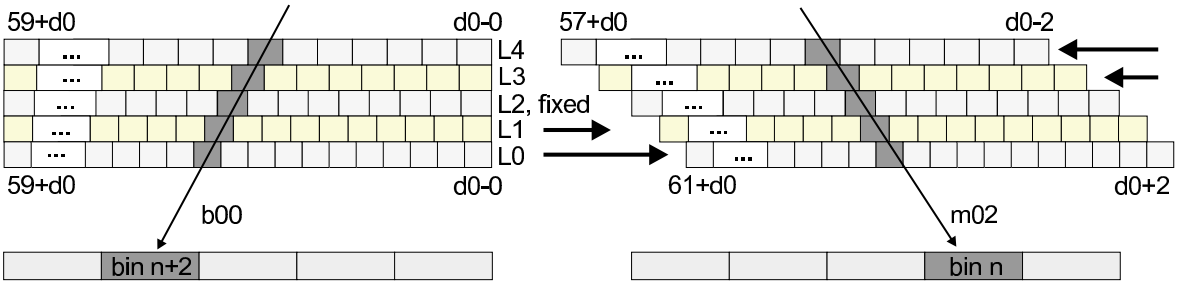


Fig. 8.9: Two patterns of symmetric tracks are shown. The same symmetric track finding kernel is used in both cases, but the submodule is called with different vectors (b00, m02). In the same way, all 11 symmetric tracks are generated.

The trigger kernel is called for the complete vector of 60 (46) cpads. For three tracks (two symmetric (as in Figure 8.9) and one asymmetric), this is shown below:

```
parameter d0 = 19; //number of virtual pads (no negative index allowed)
stracks b00(clk_40,option,E0[59+d0:d0-0],E1[59+d0:d0-0],E2[59+d0:d0-0],
                          E3[59+d0:d0-0],E4[59+d0:d0-0], bin_0[59:0]);
atracks m01(clk_40,option,E0[58+d0:d0-1],E1[59+d0:d0-1],E2[59+d0:d0-0],
                          E3[60+d0:d0+0],E4[60+d0:d0+1], bin_m1[59:0]);
stracks m02(clk_40,option,E0[57+d0:d0-2],E1[58+d0:d0-1],E2[59+d0:d0-0],
                          E3[60+d0:d0+1],E4[61+d0:d0+2], bin_m2[59:0]);
```

In the same way all 11 symmetric tracks are generated by shifting the vector ranges. Asymmetric tracks are generated in the same way, but here, in layer 1 and 3, two neighboring pads are logically combined with an **or** to be identified as a track.

The trigger kernel (for symmetric tracks) is shown below:

```
// define symmetric tracks
module stracks(clk_40, option, E0, E1, E2, E3, E4, strack);
  input          clk_40;
  input  [14:0] option;
  input  [59:0] E0;
  input  [59:0] E1;
  input  [59:0] E2;
  input  [59:0] E3;
  input  [59:0] E4;
  output [59:0] strack;
  reg    [59:0] strack;

  always @(posedge clk_40)
```

```
    begin
      strack =  ((      E3 & E2 & E1 & E0) * option[0])
              | (( E4 &      E2 & E1 & E0) * option[1])
              | (( E4 & E3 &      E1 & E0) * option[2])
              | (( E4 & E3 & E2      & E0) * option[3])
              | (( E4 & E3 & E2 & E1     ) * option[4])
              | ((           E2 & E1 & E0) * option[5]);
    end
endmodule
```

The track `strack` is defined if at least 4-out-of-5 layers show a hit. To switch off individual layers in single φ-sectors, a 6-bit vector is defined, called `option`. It is programmable via VME-bus. In case of a defect in layer 1 in the current φ-sector, the FPGA is programmed like `option = 8;`, only the combinations with layer $4, 3, 2$ and 0 are used for the trigger algorithm. Pads of layer 1 are not used, so a 4-out-of-4 option is selected. A standard setting is `option = 1F;`. In this case all 4-out-of-5 combinations are allowed. In conjunction with the force 1 (`F1_Ex_mask`, see Section 8.2.2.2 on page 96) option, other trigger strategies are possible, e.g. 3-out-of-4, 3-out-of-3 or 2-out-of-3.

Adder: Each of the 22 track pattern vectors ($\hat{=}$ hitlist with 22 rows of one FPGA) has a size of 60 (46) bit when transmitted to the adder (`adder.v`). The adder sums each of the 22 vectors in a cascade adder (as described in Chapter 7), called for each of the 22 vectors, shown below:

```
// This is the Summing-Kernel of the cascade adder
    always @(posedge clk) (cascade 1)
      begin
        sum_7A[2:0] = zhit[00]+zhit[01]+zhit[02]+zhit[03]+zhit[04]+zhit[05]+zhit[06];
        sum_7B[2:0] = zhit[07]+zhit[08]+zhit[09]+zhit[10]+zhit[11]+zhit[12]+zhit[13];
          ...
        sum_7J[2:0] = zhit[56]+zhit[57]+zhit[58]+zhit[59];
      end

    always @(posedge clk) (cascade 2)
      begin
        sum_21A[4:0] = sum_7A[2:0] + sum_7B[2:0] + sum_7C[2:0];
        sum_21B[4:0] = sum_7D[2:0] + sum_7E[2:0] + sum_7F[2:0];
        sum_21C[4:0] = sum_7G[2:0] + sum_7H[2:0] + sum_7J[2:0];
      end

    always @(posedge clk) (cascade 3)
      begin
        sum_60A[5:0] = sum_21A[4:0] + sum_21B[4:0] + sum_21C[4:0];
      end

    always @(posedge clk) (cascade 4)
      begin
        if(!reset)
          begin
            bin_sum = 0;
          end
```

```
        else
          begin
            bin_sum[5:0] = sum_60A[5:0];
          end
      end
// End Summing-Kernel
```

The result are 22 6-bit values, indicating the number of tracks of their corresponding bins. Now 15 sequential bins out of 22 are selected according to the z-vertex position of the currently programmed z-vertex histogram position. To keep the number of used logic as small as possible, the selection procedure is done as shown in Figure 8.10.

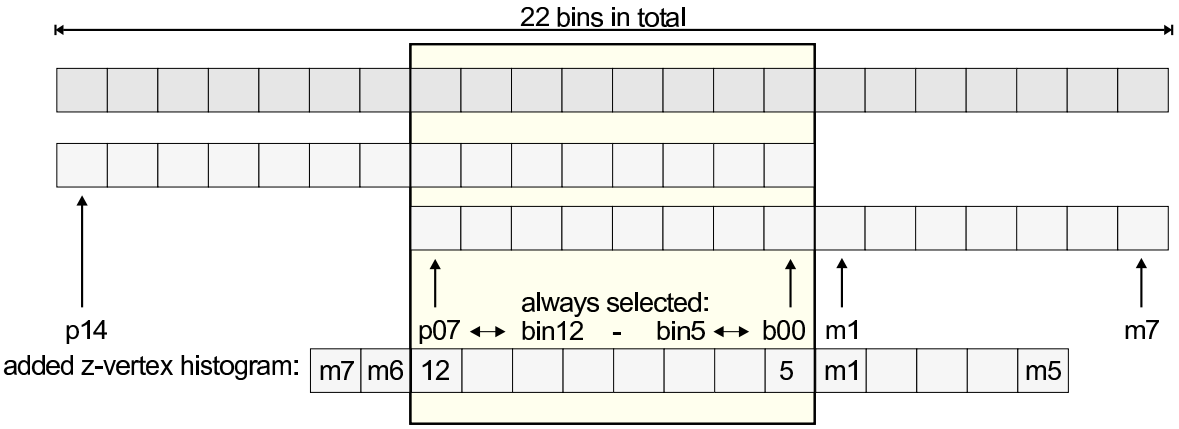


Fig. 8.10: Selection of 15 adjacent bins out of 22: In the middle, 8 bins are always selected. Additionally 7 bins are selected. In case of a backward optimized histogram, bins are distributed as shown.

The problem is the selection of 15 sequential bins out of 22: In the middle, 8 bins are always selected. These bins are assigned to bins 5 to 12 as shown. Additionally 7 bins are selected according to the required position of the z-vertex histogram. In case of a backward optimized histogram, bins are distributed as shown in Figure 8.10. In a case query the programmable bins are selected, as shown for one example (case = 2, as shown in Figure 8.10):

```
  ...
  case (mid)
    0:
...
    2:
      begin
        Histo2FPGA2_bin_00 = abin_m5;
        Histo2FPGA2_bin_01 = abin_m4;
        Histo2FPGA2_bin_02 = abin_m3;
        Histo2FPGA2_bin_03 = abin_m2;
        Histo2FPGA2_bin_04 = abin_m1;
        Histo2FPGA2_bin_13 = abin_p8;
        Histo2FPGA2_bin_14 = abin_p9;
      end
    ...
8:
...
```

In total, 8 possible assignments are programmable via the VME bus. In the histogram adder module in FPGA II, the histogram is sorted in the correct way (see Paragraph 8.3.2.2).

The result of shifting the z-vertex histogram can be best seen in Figure 7.6 in Chapter 7. In total 15×6 bits leave the adder module. In case of FPGA I, data are linked to FPGA II via the 90-pin interconnection and are routed into the module `sync_FPGA1.v` in FPGA II.

8.3 Design II: FPGA II of the Trigger Card

FPGA II has exactly the same functionality as FPGA I for the modules, described in the previous Section (Section 8.2: Design I). In addition FPGA II is used to add each 6-bit z-vertex histogram, from FPGA I cpads 0-59 and from FPGA II cpads 60-105. In Figure 8.11, a schematic overview of the data flow in FPGA II is given.

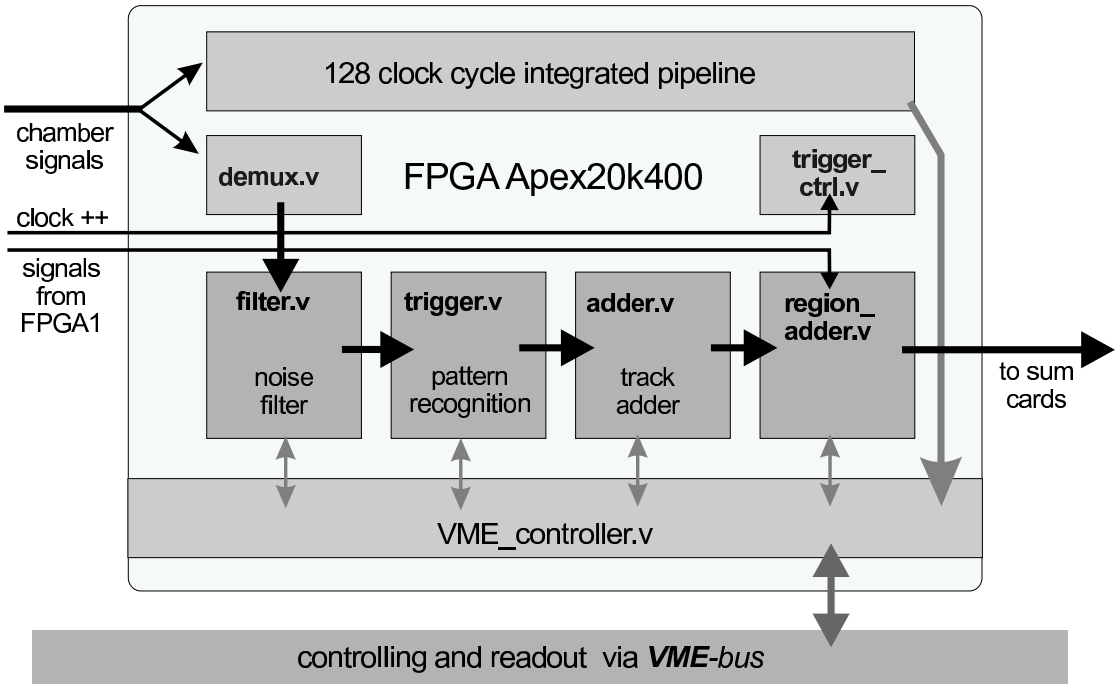


Fig. 8.11: Data flow in FPGA II of the trigger card. Compared to the data flow of FPGA I, in addition, a programmable definition of the regions (fwd, ctr and bwd) is implemented in FPGA II (region_adder.v).

The complete z-vertex histogram is then divided into three parts corresponding to three regions of the z-axis: a so called **forward** region, a **central** region and a **backward** region (as described in Chapter 7). The tracks found in each part are added, and the results are stored in three 8-bit words, transported to the sum crate.

8.3.1 Hierarchical Hardware Description Level

The CIP2k_TC_FPGA2 design is organized in a hierarchic structure of modules like the design of FPGA I. The top-level block design of the second FPGA (CIP2k_TC_FPGA2)

is shown in Figure 8.12. Additional outputs in the block diagram file of FPGA II, compared to FPGA I, are signals to the sum card. In total, four vectors, one of each region (fwd, ctr, bwd) and one spare line, and the corresponding control signals (clk_40, pipe_en, fst_wd, n_g_rst) are sent to the sum card. Note that the four vectors are inverted (active low logic) due to the fact that unconnected inputs at the sum card are set to high by the LVDS to LVTTL converter (Dallas DS90LV032). Without these measures, unconnected inputs would disturb the trigger algorithm in the sum card. In the FPGA of the sum card they are inverted to active high logic again.

The readout module is organized in the same way as in FPGA I. The trigger module of design II, shown in Figure 8.12, contains the submodule trigger_system.bdf as in FPGA I and furthermore the submodule TE_builder.bdf[6]. In this module, the z-vertex histograms of FPGA I and FPGA II are merged and the histogram is divided into the three regions along the z-axis.

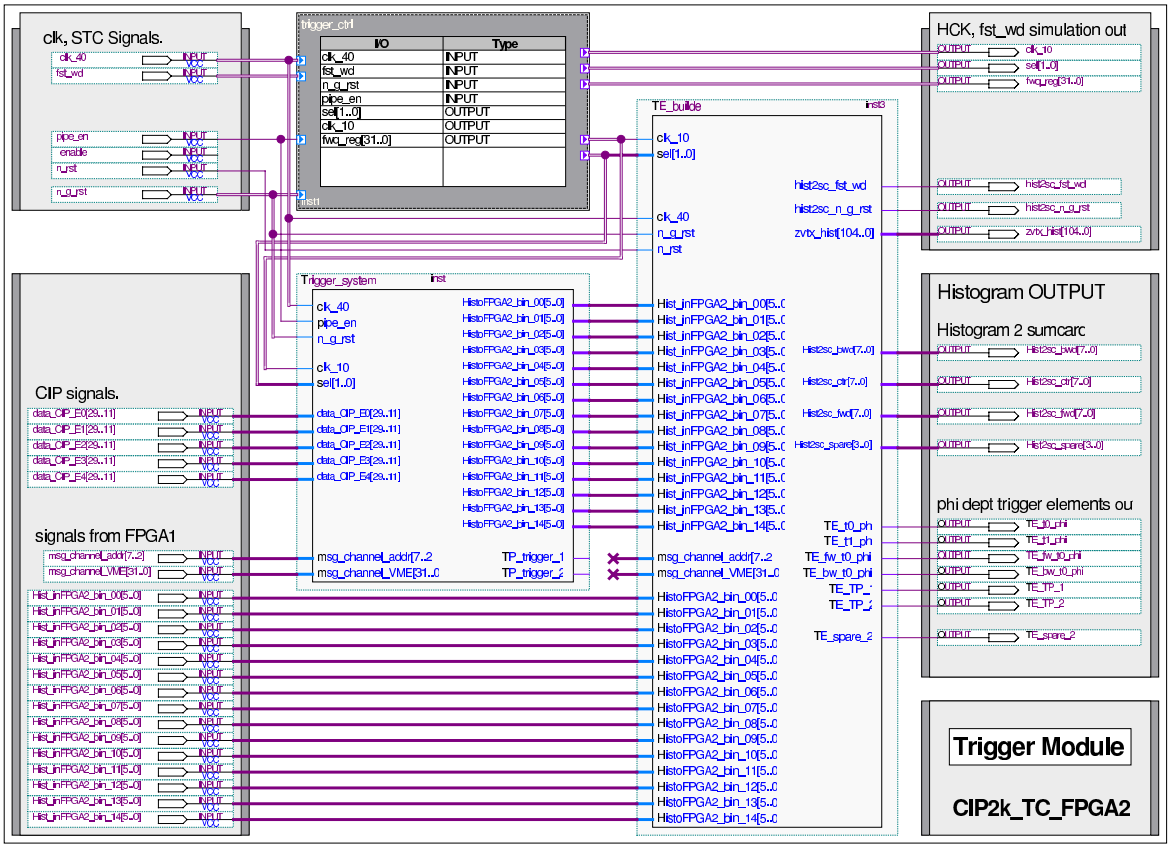


Fig. 8.12: The trigger module block design of the second FPGA (CIP2k_TC_FPGA2).

The histogram builder (module): This module contains the histogram adder and the partition of the z-vertex histogram bins to regions that are not needed in FPGA I. In Figure 8.13, the histogram builder module is shown. It contains the following submodules, described below: `synchronizer.bdf`, `main_adder.v` and `element_builder.v`.

[6]This historical name was selected before the sum cards were used to define the trigger decision (TE) and never changed.

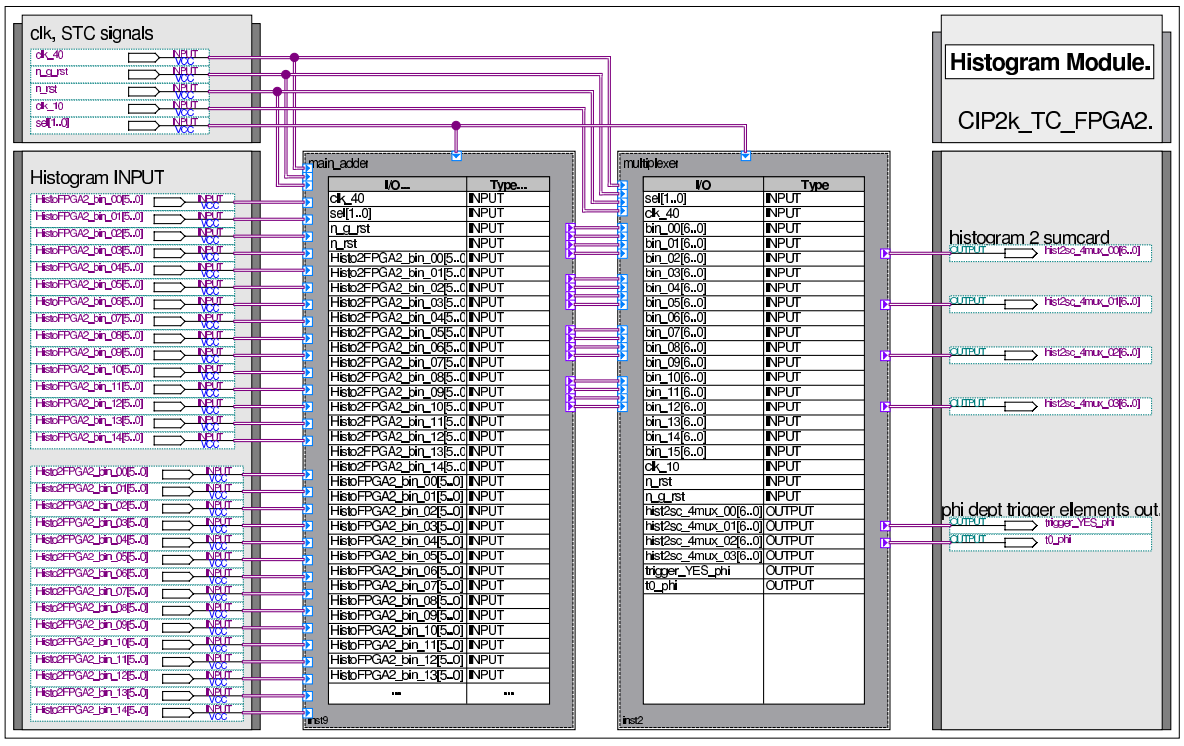


Fig. 8.13: The histogram module block design of the second FPGA (CIP2k_TC_FPGA2) is shown.

8.3.2 Functional Description of Verilog HDL Submodules

8.3.2.1 Submodules for the Readout

The readout module is organized in the same way as in FPGA I. The description of the submodules can be found in Section 8.2.2.1.

8.3.2.2 Submodules for the Trigger

Synchronization: In a first step, both 15-bin z-vertex histograms have to be synchronized. This is necessary due to delay effects during the transport of the z-vertex information from FPGA I to FPGA II. This is done in the modules sync_FPGA1.v and sync_FPGA2.v. Both submodules are hosted in the synchronizer.bdf module. The idea of the synchronization is rather simple. The z-vertex information is given to the output pins of FPGA I with the fst_wd state sel = 1. They have to be stable again 50 ns later with sel = 3 in FPGA II. The same synchronization is done for the information of FPGA II to ensure that both signals came from the same event (BC). The Verilog code of the synchronization is shown below:

```
always @(posedge clk_40)
  begin
    if (sel == 3)
      begin
        HistoFPGA1_sync_bin_00 = Hist_inFPGA2_bin_00;
        HistoFPGA1_sync_bin_01 = Hist_inFPGA2_bin_01;
           ....
```

```
        HistoFPGA1_sync_bin_14 = Hist_inFPGA2_bin_14;
      end
  end
```

Histogram adder: From the synchronizer both parts of the z-vertex histogram are transferred to the histogram adder (`main_adder.v`). The histogram adder adds each bin-pair of the corresponding bin number of FPGA I and FPGA II. The result is a 7-bit number:

```
always @(posedge clk_40)
  begin
    if (!n_g_rst) dataout = 0;
    else dataout[6:0] = dataa[5:0] + datab[5:0];
  end
```

In the next step, the z-vertex histogram bins have to be sorted. This is necessary in order to resolve the permutations in the adder (`adder.v`, see Figure 8.10 in Paragraph 8.2.2.2). Depending on the programmed position of the z-vertex histogram, the correct order of seven possibilities is selected (in a case query), programmed via the VME bus (message channel) in the same way as in the adder.v module.

The final z-vertex histogram is given to the element builder (`element_builder.v`)

Element builder: In the element builder module, the histogram is grouped into three regions. The size and position of the regions is programmable (via VME bus). One 32-bit VME word is used to decode the settings for the position and the number of bins of each region. With this, every bin can be assigned either to the forward, central or backward region or can be ignored. The information is coded by using the first 15 bits ([14:0]) and the last 15 bits ([30:16]) of the 32-bit VME word in combination. Every pair of two bits enables four possible settings (i.e. [16 & 0] $\hat{=}$ two bits $\rightarrow$ 3 states + off), shown in the following code:

```
// msg_channel communication:
  always @(posedge clk_40)
    begin
      if(msg_channel_addr == 3)
        begin
          selector_fwd[14:0] =  msg_channel_VME[30:16] &  msg_channel_VME[14:0];
          selector_ctr[14:0] =  msg_channel_VME[30:16] &(~msg_channel_VME[14:0]);
          selector_bwd[14:0] =(~msg_channel_VME[30:16]) &(msg_channel_VME[14:0]);
        end
    end
```

Each region is evaluated by adding all 15 histogram bins multiplied with the corresponding selector. If the selector-bit of a given bin is not set, this bin is not added:

```
// 1. cascade step  (8bit)
  always @(posedge clock)
    begin
      result_ab1 = ((dataa1)*(selector[0])) + ((datab1)*(selector[1]));
      result_cd1 = ((datac1)*(selector[2])) + ((datad1)*(selector[3]));
```

```
      result_ef1 = ((datae1)*(selector[4])) + ((dataf1)*(selector[5]));
      result_gh1 = ((datag1)*(selector[6])) + ((datah1)*(selector[7]));
      result_ab2 = ((dataa2)*(selector[8])) + ((datab2)*(selector[9]));
      result_cd2 = ((datac2)*(selector[10])) + ((datad2)*(selector[11]));
      result_ef2 = ((datae2)*(selector[12])) + ((dataf2)*(selector[13]));
      result_gh2 = ((datag2)*(selector[14]));
    end
```

The result of the complete adding process (4 cascade steps) are three regions. The number of tracks in each region is limited to 255 (8 bit). The information of the three regions are given to the output pins of the second FPGA. In case of counting more than 255 tracks, the overflow handler sets the result to 255 (8'hff), as shown:

```
// 4.step: result (10bit->8bit) and overflow handler
  always @(posedge clock)
    begin
      if (((result_abcdefgh1) + (result_abcdefgh2)) > 255)
        begin
          result[7:0] = 8'hff;
        end
      else
        begin
          result[7:0] = result_abcdefgh1[7:0] + result_abcdefgh2[7:0];
        end
    end
```

In addition, the control signals for the sum card are generated in the element builder module. The `fst_wd` signal for the sum card is created by using the sel-signal, and the `n_g_rst` is synchronized to the 41,6 MHz clock:

```
// fst_wd, n_g_rst to SCs:
  always @(posedge clk_40)
    begin
      if (sel == 0) hist2sc_fst_wd = 1;
      else hist2sc_fst_wd = 0;
    end

  always @(posedge clk_40)
    begin
      hist2sc_n_g_rst = n_g_rst;
    end
```

The pipe_en signal and the clk_40 signal are directly linked from the corresponding FPGA input pins to the output pins.

In total, FPGA II delivers a bunch of 32 signals to the pre sumcard:

- Number of tracks in each of the three regions (3×8 bit)
- Four of the control signals (clk_40, pipe_en, n_g_rst and fst_wd)
- Four spare lines

The final z-vertex histogram is grouped into a vector of 15×7 bits = 105 bits (`zvtx_hist[104:0]`) in the element builder module, too. Coded in this way, it is routed to the memory module, where it is stored in a pipeline, 32 steps deep. It can be read out as described in Paragraph 8.2.2.1.

8.4 Design III: The FPGA of the Sum Card

The FPGA of each sum card contains cascade adders to sum up the histograms (number of tracks in the three regions) either delivered from the four trigger cards or from four pre sum cards. Pre sum card and master sum card have the same HDL design: four input region histograms are added to one output histogram. The output format is the same as the input format in every region (8 bit), transferred/received to/from the sum card, independent of pre or master sum card.

In addition, the summed information is stored in a pipeline ring-buffer (32 BCs deep). The buffer can be read out by the VME bus for monitoring purposes.

In each sum card, a trigger decision module evaluates the summed histogram to create 16 trigger elements (see Section 7.5). The trigger elements of the master sum card are transmitted to the central trigger control of the H1 detector. The trigger element information of the pre sum card is not used for other than testing purposes. An overview of the data-flow in each sum card is depicted in Figure 8.14.

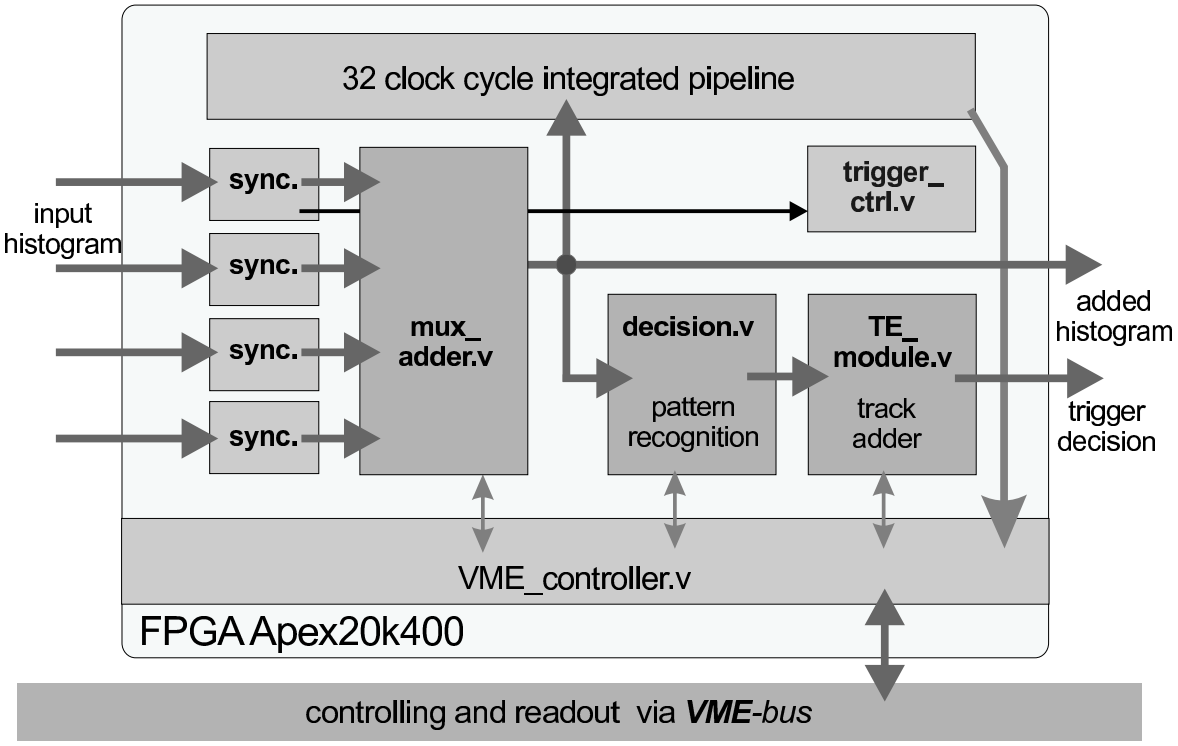


Fig. 8.14: Schematic view of the data flow in the sum cards.

8.4.1 Hierarchical Hardware Description Level

top-level design: The top-level design is shown in Figure 8.15. The control signals (clk_40, fst_wd, n_g_rst and pipe_en) are transferred via SCSI cable into the sum card. These signals are given to the output in the same way as they arrive at the sum card.

Furthermore, each sum card has a total of 12 input busses (8 bit each) for the four times three histogram regions and additional 4 spare busses (4 bit). In parallel to the

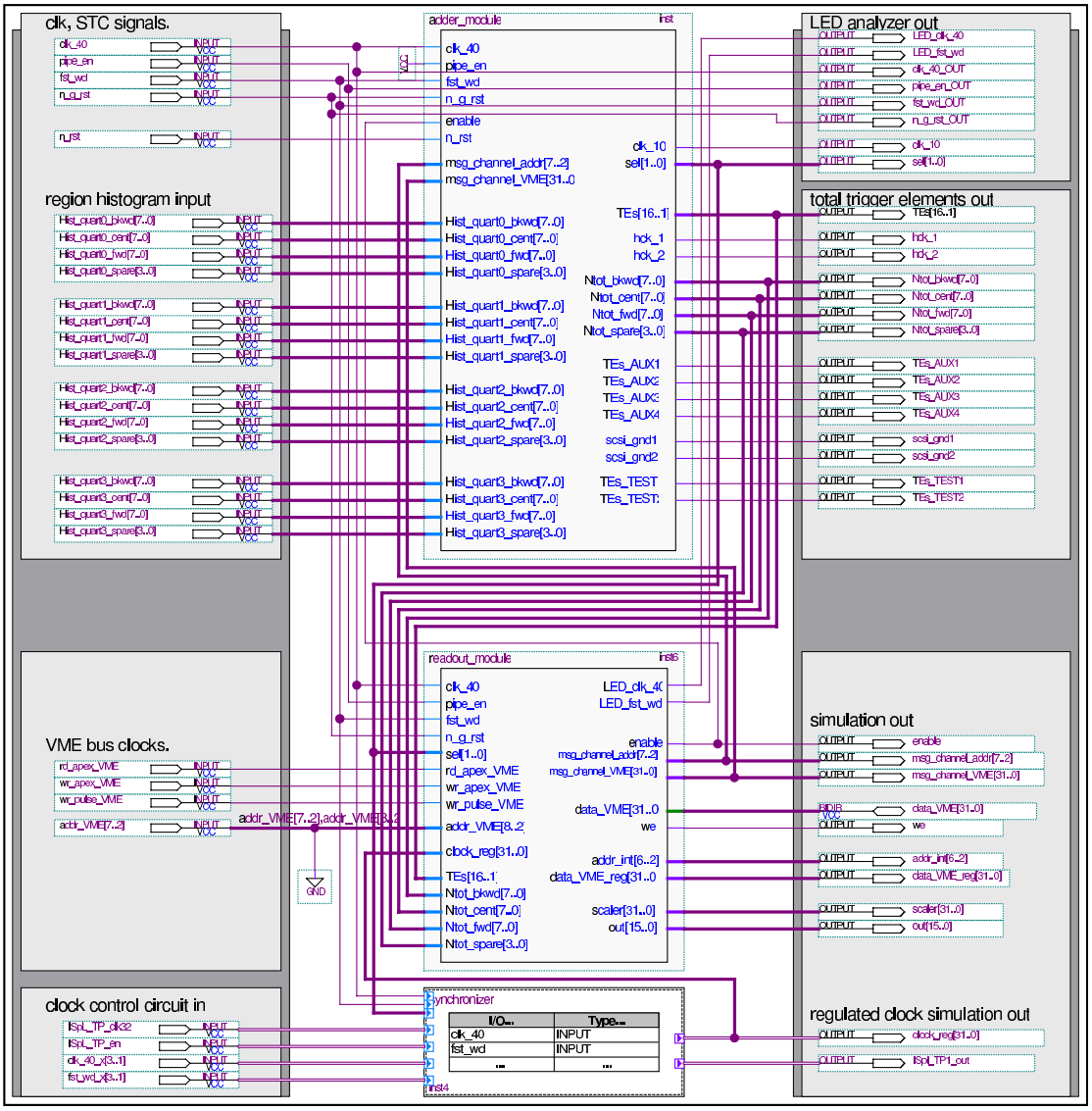


Fig. 8.15: top-level block design of the sum card design (CIP2k_SC_20k).

inputs, three 8-bit output busses (fwd, ctr and bwd region) and one 4-bit output bus (spare) are defined. The 16-bit trigger elements are liked to the 50-pin SCSI connector together with the necessary clock information.

Finally, all necessary connections for the VME bus are linked into the FPGA.

The top-level design is divided into three submodules, the readout module (readout_module.bdf), containing VME interface and memory module, the adder module (adder_module.bdf), containing the region adder, trigger decision module and the trigger element definition module and the synchronizer (`synchronizer.v`) used to synchronize the sum cards.

Synchronizer: This module is used to write the clock phases of every 41,6 MHz clock and first word signal from every input card (trigger card or pre sum card) into a VME register, in total four times (clk_40[3:0], fst_wd[3:0]). The signal of the first input card

(correlated hard wired with the lower left connector of the sum card) is used as the dedicated clock signal for the design (clk_40[0] $\hat{=}$ clk_40). The first word input of this card is used as the reference first word (fst_wd[0] $\hat{=}$ fst_wd). This is shown in the following Verilog code:

```
// write clk_40 + fst_wd of all inputs in VME reg
  always @(posedge clk_40)
    begin
      case(sel)
        0:
          begin
            clock_reg[0] = fst_wd;
            clock_reg[4] = fst_wd_x[1];
            clock_reg[8] = fst_wd_x[2];
            clock_reg[12] = fst_wd_x[3];
            clock_reg[16] = clk_40;
            clock_reg[20] = clk_40_x[1];
            clock_reg[24] = clk_40_x[2];
            clock_reg[28] = clk_40_x[3];
          end
        1:
          begin
            clock_reg[1] = fst_wd;
            clock_reg[5] = fst_wd_x[1];
            clock_reg[9] = fst_wd_x[2];
            clock_reg[13] = fst_wd_x[3];
            clock_reg[17] = clk_40;
            clock_reg[21] = clk_40_x[1];
            clock_reg[25] = clk_40_x[2];
            clock_reg[29] = clk_40_x[3];
          end
        2:
          ...
      endcase
    end
```

The select signal (sel[1:0]) is generated by the reference fst_wd (in the trigger_ctrl module). The 32-bit register can be read out any time via the VME bus (clock edge, see appendix A on page 145). If the first words of every input card are written at the same time as the reference fst_wd (in case 0), the timing is fine:

```
hex 0000 1111  --> OK
```

Otherwise, the input clock of the corresponding CIPix-board has to be shifted (see Section 6.2.4):

```
hex 000F 1114  --> timing error
```

In the example, the clock + first word phase of input 3 is wrong.

8.4.2 Functional Description of Verilog HDL Submodules

8.4.2.1 Submodules for the Readout

The VME module (`state_machine.v`) and the memory module (`memory_module.v`) are used in the same way as in the trigger card FPGAs (see Section 8.2).

8.4.2.2 Submodules for the Trigger

Region adder: The region adder (`mux_adder.v`) adds the number of tracks in each region (fwd, ctr and bwd). It adds the tracks in a two step cascade, shown in the Verilog code extraction below (simplified):

```
//four step cascade adder
// 1.step: 1.cascade adding  (2x8bit->9bit)
  always @(posedge clock)
    begin
       else
         begin
           result_ab = dataa + datab;
           result_cd = datac + datad;
         end
    end

// 2.step: 2.cascade adding  (10bit)
  always @(posedge clock)
    begin
      result_abcd = result_ab + result_cd;
    end

// 3.step: overflow definition  (cuts 10bit to 8bit)
  always @(posedge clock)
    begin
      if (result_abcd > 255)
        begin
          result_dly1[7:0]  =  8'hff;
  overflow = 1;
        end
      else
        begin
          result_dly1[7:0]  =  result_abcd[7:0];
  overflow = 0;
        end
    end

// 4.step: output
  always @(posedge clock)
    begin
      result[7:0] = result_dly1[7:0];
    end
```

The third step defines the behavior in case of an overflow. The total number of tracks is limited to 255 (8 bit) if the number of total tracks exceeds 255, it is set to 255.

All inputs (4×3 regions) can be switched off separately (via VME bus), before they are linked into the cascade adder. In this case the region information of noisy trigger cards can be switched off. Moreover, if the CIP2k chamber HV is off, the CIP2k trigger can be switched off to avoid delivering invalid information.

Trigger decision module: In the trigger decision module (`decision.v`), the trigger elements are evaluated as described in Chapter 7, Section 7.5.

The main trigger elements are:

CIP_T0 and CIP_T0_NEXTBC: If at least one track is identified in the central region (`Hist_full_cent > 0`), the CIP_T0 is set. It is shortened to exactly 1 BC (96 ns), because it is used as a reference timing signal for other sub detectors. The Verilog code is shown below.

```
// TE1 -> CIP_T0, limited to 100ns length
  always @(posedge clk_40)
    begin
      if(sel == SEL_TE)
        begin
          if (Hist_full_cent > 0)
            begin
              if(!TE_buf_1)
                begin
                  TE_reg_xx[1] = 1; //TE output signal
                  TE_buf_1 = 1; //register
                end
              else
                begin
                  TE_reg_xx[1] = 0;
                  TE_buf_1 = 1;
                end
            end
          else
            begin
              TE_reg_xx[1] = 0;
              TE_buf_1 = 0;
            end
        end
    end
```

To shorten the signal to 1 BC, it is written into an additional register (`TE_buf_1`). If there are more than zero tracks in the following BC, `TE_buf_1` is already on. This indicates that this track also was on in the last BC (and has to be shortened). In this case, the trigger element `TE_reg_xx[1]` is switched off. It stays off until `TE_buf_1` is off again. With the next triggered track (in the central region), both signals can be switched on again.

The CIP_T0_NEXTBC signal comes one BC prior to the other 15 TEs. This is done by delaying all TEs by 96 ns with the exception of CIP_T0_NEXTBC.

Significance and multiplicity of the event: Significance and multiplicity are calculated as defined in Section 7.5.1 on page 79. The following code is used to generate the significance and multiplicity information:

```
// TE3 -> CIP_SIG_0
// TE4 -> CIP_SIG_1
  always @(posedge clk_40)
    begin
      if(sel == SEL_TE)
        begin
          if (((Hist_full_cent+3)>>2) > (Hist_full_fwd + Hist_full_bkwd))
            TE_reg_xx[4:3] = 3;
```

```
          else if (((Hist_full_cent+1)>>1) > (Hist_full_fwd + Hist_full_bkwd))
            TE_reg_xx[4:3] = 2;
          else if (Hist_full_cent > (Hist_full_fwd + Hist_full_bkwd))
            TE_reg_xx[4:3] = 1;
          else
            TE_reg_xx[4:3] = 0;
        end
    end

// TE5 -> CIP_MUL_0
// TE6 -> CIP_MUL_1
// TE7 -> CIP_MUL_2
  always @(posedge clk_40)
    begin
      if(sel == SEL_TE)
        begin
  if ((Hist_full_fwd + Hist_full_bkwd + Hist_full_cent) > 100)
            TE_reg_xx[7:5] = 7;
          else if ((Hist_full_fwd + Hist_full_bkwd + Hist_full_cent) > 30)
            TE_reg_xx[7:5] = 6;
          else if ((Hist_full_fwd + Hist_full_bkwd + Hist_full_cent) > 20)
            TE_reg_xx[7:5] = 5;
          else if ((Hist_full_fwd + Hist_full_bkwd + Hist_full_cent) > 10)
            TE_reg_xx[7:5] = 4;
          else if ((Hist_full_fwd + Hist_full_bkwd + Hist_full_cent) > 6)
            TE_reg_xx[7:5] = 3;
          else if ((Hist_full_fwd + Hist_full_bkwd + Hist_full_cent) > 2)
            TE_reg_xx[7:5] = 2;
          else if ((Hist_full_fwd + Hist_full_bkwd + Hist_full_cent) > 0)
            TE_reg_xx[7:5] = 1;
          else
            TE_reg_xx[7:5] = 0;
        end
    end
```

The CIP2k cosmics trigger (TE8): The cosmics trigger uses the track information of all four pre sumcards. At the main sum card, the number of tracks in each region of each pre sum card is analyzed to generate four 3-bit vectors (each 1 bit for forward, central and backward region) indicating the number of tracks. This information is used to build a cosmics trigger decision, as shown (see also Section 7.5.1 on page 81):

```
// TE8 -> CIP_COS
  always @(posedge clk_40)
    begin
      if(sel == SEL_TE)
        begin
          if(SC_flag == MSC)
            begin
              if(((Hist_quartbit0 > 0) & (Hist_quartbit1 > 0)) |
                 ((Hist_quartbit0 > 0) & (Hist_quartbit2 > 0)) |
                 ((Hist_quartbit0 > 0) & (Hist_quartbit3 > 0)) |
                 ((Hist_quartbit1 > 0) & (Hist_quartbit2 > 0)) |
                 ((Hist_quartbit1 > 0) & (Hist_quartbit3 > 0)) |
```

```
              ((Hist_quartbit2 > 0) & (Hist_quartbit3 > 0)))
              TE_reg_xx[8] = 1;
            else
              TE_reg_xx[8] = 0;
          end
            else
              begin
                if ((Hist_full_fwd + Hist_full_bkwd + Hist_full_cent) > 0)
                  TE_reg_xx[8] = 1;
              end
        end
  end
```

Thus the cosmics trigger uses the same trigger algorithm, but unlike the z-vertex trigger, it has an additional fourfold Θ-resolution. This is useful to detect events with tracks opposite in φ in ep runs.

Trigger element output module: In the trigger output module, all trigger elements are pipelined into a programmable delay. The trigger elements can be shifted against the HERA clock and can be delayed in steps of 24 ns (corresponding to the sel signal), by a maximum of 192 ns (2 BCs).

Each trigger element is delayed by the programmed select state (sel_prg[1:0]) and a BC-wise delay, shown in the two code examples:

Programming of delay values:

```
always @(posedge clk_40)
  begin
    if (msg_channel_addr == 4)
      begin
        sel_prg[1:0] = msg_channel_VME[17:16];
        BC_delay_prg[1:0] = msg_channel_VME[19:18];
      end
  end
```

Assigning delays to trigger elements:

```
// create delay pipeline
  always @(posedge clk_40)
    begin
      if (sel == sel_prg) //24ns steps
        begin
  TE_reg_1 = TE_reg;
  TE_reg_2 = TE_reg_1;
  TE_reg_3 = TE_reg_2;
        end
    end

//pipeline for BC delay
  always @(posedge clk_40)
    begin
      if (sel == sel_prg)
        begin
          if(BC_delay_prg == 0 && sel == sel_prg) TEs = TE_reg;
```

```
  else if (BC_delay_prg == 1) TEs = TE_reg_1;
  else if (BC_delay_prg == 2) TEs = TE_reg_2;
  else if (BC_delay_prg == 3) TEs = TE_reg_3;
end
```

8.5 Design IV: The FPGA of the Komposti Card

The komposti card design is used to merge the information of the new CIP2k chamber to fit the requirements for the receiver card input of the old z-vertex trigger system (see Chapter 6, Section 6.5). In a first step, the information from the control cards is demultiplexed. The demultiplexed information is handshaked to a filter module in the same way as in the trigger cards. After that, the information of two layers and two φ-sectors (with 120 pads) is merged corresponding to one layer of the old, eight-fold segmented chamber with 60 pads in z-directions. The merging of the layer information is programmable. By default, a bitwise **AND** is selected. An **OR** connection can be selected or one of the layers can be turned on. The merged information is given to the output pins of the FPGA. Demultiplexer, filter and merger are each supported with VME bus information via the internal message channel. The merged information is furthermore written into a pipeline ring buffer, 32-bit deep. The buffer can be read out by the VME bus, also for monitoring purposes. In Figure 8.16, a data flow of the komposti card is shown.

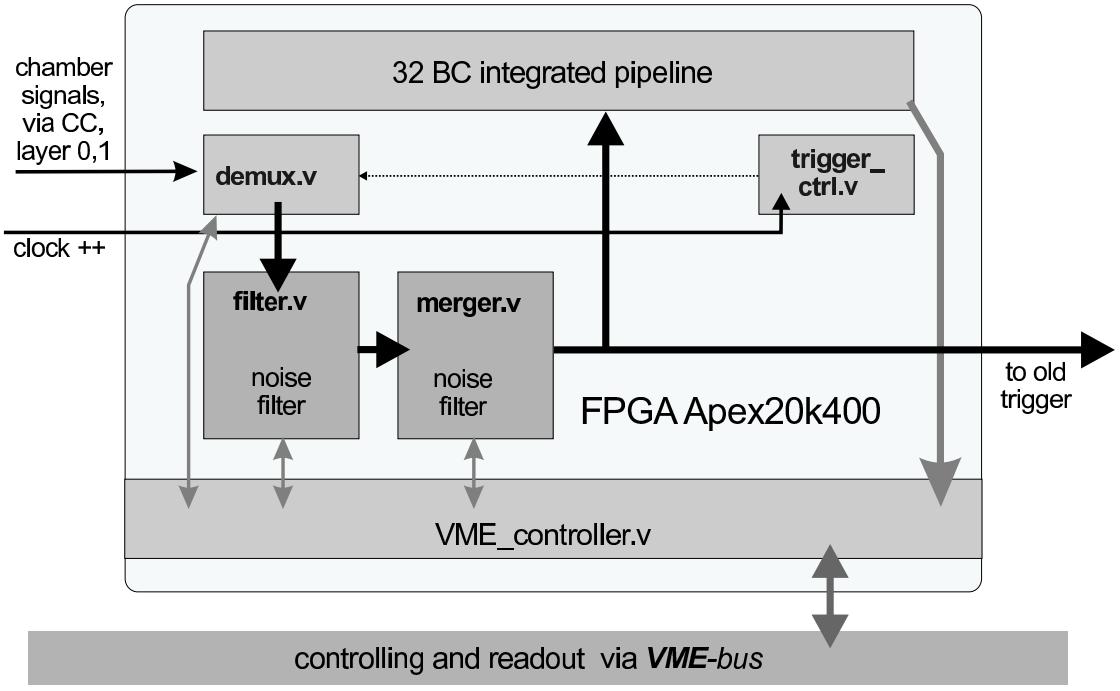


Fig. 8.16: Schematic view of the data flow in the komposti card.

8.5.1 Hierarchical Hardware Description Level

top-level design: The top-level design is shown in Figure 8.17. Just as for the sum

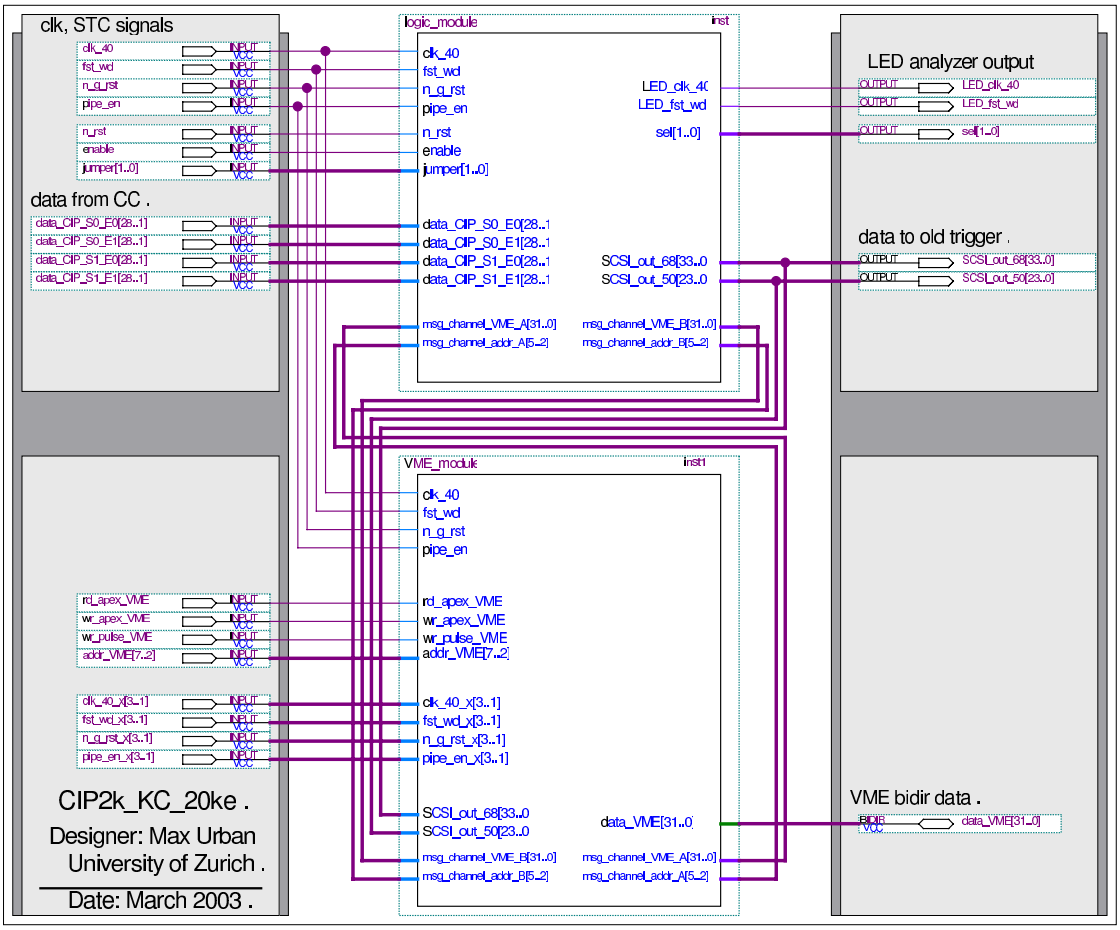


Fig. 8.17: top-level block design of the komposti card design (CIP2k_KC_20ke).

card, the control signals (clk_40, fst_wd, n_g_rst and pipe_en) are linked via SCSI cable into the komposti card, generated or forwarded by the control card. The chamber information of each control card is routed in the komposti card FPGA as described in Chapter 6 in Section 6.2.4. Unlike in the trigger card, it is not necessary to store this information in a pipeline. Additionally, all necessary connections for the VME bus are linked into the FPGA. The design only has two output channels, one for the 50-pin SCSI connector (24 channels) and one for the 68-pin connector (32 channels).

The top-level design is divided into two submodules, the VME_module.bdf, containing VME interface and memory, and the logic_module.bdf, containing the demultiplexer, filter and merger module.

8.5.2 Functional Description of Verilog HDL Submodules

8.5.2.1 Submodules for the Readout

For the VME readout of the merged pad information (see below), the same readout algorithm is used, as in the FPGA II for the readout of the z-vertex histogram (see paragraph "VME interface" in Section 8.2).

8.5.2.2 Submodules for the Trigger

The same modules for demultiplexing and filtering are used as in the trigger card FPGAs. A detailed description can be found in Section 8.2, Paragraph "demultiplexer" and "filter for noisy pads".

Merger: The merger module gets the demultiplexed and handshaked signals of layer 0 and 1 of two φ-sectors. As mentioned in Section 6.2.4, the first and the last four pads of each layer are not transferred. In case of layer 1, the last four pads do not carry any chamber signals anyway (projective geometry).

The demultiplexed and filtered information of layer 0 and layer 1 can be sent to the old trigger system, feeding the receiver card inputs of layer 0 and 1 of the correlative φ-sector. But to keep the number of necessary output channels (important limitation) as small as possible, the komposti card takes over the evaluation of the logical combination of both layers and handshakes the result to a single layer of the old trigger system.

Therefore, a merger algorithm was developed. The merger algorithm checks possible overlaps of pads of layer 0 and layer 1 and produces a result for each pad, transferred to the old trigger as shown in Table 8.1.

old	0	1	2	3	...	57
new, layer 0	-	-	4,5	6,7	...	114,115
new, layer 1	-	-	-,4,5	5,6,7	...	113,114,115

Tab. 8.1: Assignment of coincidences of layer 0 and 1 of the new chamber to corresponding pad geometry of the old chamber.

The following cutout of the code shows the merging for one output pin (SCSI_out_50[0]):

```
// combine pads of CIP2k to match old CIP
  always @(posedge clk_40)
    begin
      SCSI_out_50[0]=
        (and0&(demux_CIP_S0_E0[48]|demux_CIP_S0_E0[49])&
        (demux_CIP_S0_E1[45]|demux_CIP_S0_E1[46]|demux_CIP_S0_E1[47]))|
        (or00&(demux_CIP_S0_E0[48]|demux_CIP_S0_E0[49]))|
        (or10&(demux_CIP_S0_E1[45]|demux_CIP_S0_E1[46]|demux_CIP_S0_E1[47])) |
        (and1&(demux_CIP_S1_E0[48]|demux_CIP_S1_E0[49])&
        (demux_CIP_S1_E1[45]|demux_CIP_S1_E1[46]|demux_CIP_S1_E1[47]))|
        (or01&(demux_CIP_S1_E0[48]|demux_CIP_S1_E0[49]))|
        (or11&(demux_CIP_S1_E1[45]|demux_CIP_S1_E1[46]|demux_CIP_S1_E1[47])) |
        (lauflicht&((count==96)||(count==255))) |
        (testpuls&(count==0));
...
    end
```

The algorithm takes care of the projective alignment of the pads. Each pair of two pads in layer 0 (4+5, 6+7, ... 114+115) has exactly the same position as pads of the old chamber. Layer 1 has only 112 pads in z, distributed along the same length. This pad geometry is resolved before delivering pads to the old trigger.

8.6 Timing and Statistics

Timing of the algorithm: The timing path, beginning with the chamber signals linked into the trigger system and ending with the handshake of the trigger elements to the central trigger, is shown below. The timing of each Verilog module is displayed in units of the clk_40. Figure 8.18 shows all modules of the trigger pipeline and their latency times.

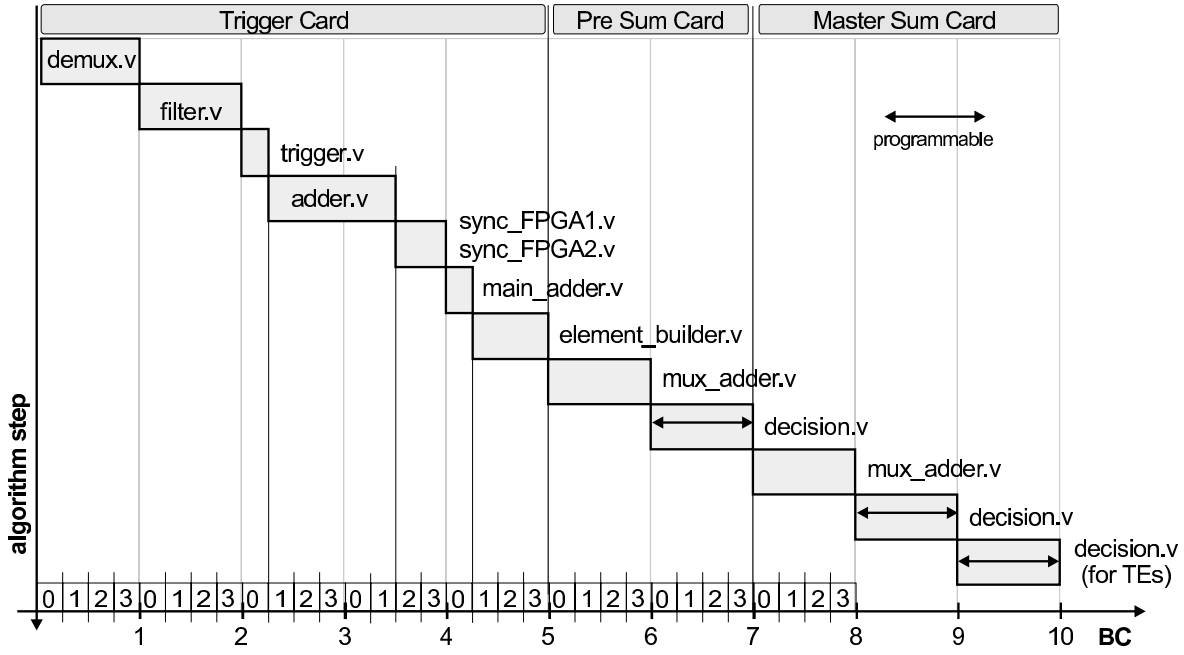


Fig. 8.18: The timing behavior of each step of the trigger algorithm is shown. The evaluation of the trigger elements takes 10 BCs ($\hat{=} 1\,\mu s$). Because the trigger decision has to be at the CTC after $2,3\,\mu s$ only, no further optimization of the timing is necessary.

Every Verilog module is fed with the 41,6 MHz clock (clk_40). Each result being processed, is registered with this clock. Thus, each step in the calculation takes 24 ns. By counting every step in the timing path, the total latency time can be calculated. To delay the delivery of the trigger elements to the central trigger control in steps of 24 ns, programmable delay-registers can be used synchronized to sel = 0,1,2 or 3 and the fst_wd signal. Note that the filter module is delayed by three clock cycles in any case. This is useful in order to have possible space for future changes in the filter algorithm without changing the timing behaviour of the complete trigger algorithm.

Statistic of used resources in the FPGA: In this Paragraph, statistics of the usage of the logic elements and memory blocks are given. In FPGA I, 98% of the logic cell resources are used for the trigger algorithm. Compared to FPGA I, FPGA II only analyzes 46 cpads, therefore only 88.3% of the logic cells are needed, even with the additional adder hardware, needed to evaluate the φ dependent z-vertex histogram for both FPGAs. This confirms that the track finding and adding modules need a major fraction of the total logic. In Figure 8.19, the logic cell and memory use of FPGA I and

```
FPGA1:
                            top-level   submodules
    trigger module:             15132
        histogram adder:          -
        trigger System:
              demultiplexer:               657
              filter:                     1596
              trackfinder:                8330
              adder:                      4444
              trigger control:             105
        readout module:            861
              statemachine:                478
              memory:                      383
    datareg:                    84
------------------------------------------------------------
total logic cells (16400):      16078   16078       = 98,0%

FPGA2:
                            top-level   submodule
    trigger module:             13381
        histogram adder:          1364
        trigger system:
              demultiplexer:               1007
              filter:                      1247
              trackfinder:                 6092
              adder:                       3566
              trigger control:              105
        readout module:           1019
              statemachine:                506
              memory:                      513
       datareg:                     84
------------------------------------------------------------
total logic cells (16400):      14485   14485       = 88,3%
```

Fig. 8.19: Usage of memory and logic elements in FPGAs of the trigger card.

FPGA II of the trigger card are shown. Compared to the logic used in the trigger card, the resources in the FPGAs of the sum card and komposti card are rather untouched.

Chapter 9

Testing the Trigger

The development of the trigger system was checked with a number of different diagnostic tools. The analysis can be divided into the following examination methods:

- Functional simulations were carried out to verify the functional behavior of each single module of the firmware in an early stage of the development.
- After having examined the functional simulations, the timing behavior of the trigger algorithm was analyzed. Timing simulations are provided by the hardware development environment.
- Tests with a pattern generator and real hardware: After installing the hardware and configuring the FPGAs, a pattern generator was used to provide typical chamber pad and control signals. This was done by removing the receiver cards and putting in adapter cards, carrying the signals from the pattern generator. The response of these well defined input patterns is used to check the track recognition and DAQ consistency.
- Tests with the DAQ: The results of the trigger algorithm are compared with a simulation, using the pad read out information. This is done on different levels of the trigger (φ-histogram, quarter-wise and global region information, trigger elements). This method provides the most detailed consistency analysis of the trigger electronics.

Each test is described in detail below.

9.1 Functional Simulations

The aim of a functional simulation is the understanding of each single hardware module in an early stage of the development. Functional simulations are provided at the RTL design level (see Section 6.8.2) without compiling the design. The development of the hardware and its functional simulation are closely related. All of the described modules were simulated in separate simulation projects, where only the behavior of a current module is tested. A detailed description of functional simulation techniques with the Quartus environment are presented it [67].

9.2 Timing Simulations

In both a functional and a timing simulation, a vector of all input waveforms (*.vwf, vector waveform file) is given to the simulation tool. The simulation generates all output files according to the current HDL design. The simulation tool is embedded into the Quartus development environment. Unlike the functional design, the HDL design is simulated after the compilation process for a timing simulation. Thus, not the hardware description on the RTL layer is tested but the fitted design in the FPGA. This type of simulation has to know the hardware structure (LEs, LABs, Megalabs, ESB..) of the hardware used. Nevertheless, these tests can be done without using any hardware.

In the timing simulation, the timing behavior of each output signal is analyzed. It is shown in a one dimensional diagram with a scalable time axis, as shown in Figure 9.1. The input signal stimuli are set according to the expected signals from the real hardware. After that, the behavior of the output signals is simulated, taking into account the signal propagation, gate delays and timing requirements like maximum clock frequency and *input to clock*[1] relations in the FPGAs. Timing simulations give a good view of the behavior of the real hardware.

Two examples of timing simulations are now presented. The complete simulation of the trigger algorithm (one simulation file for each design (I-IV) with all input stimuli and output signals) can be found in [97].

- **Sum card, trigger element generation:** Figure 9.1 shows the simulation file of the trigger element generation. The time is shown on the x-axis. On the vertical axis, input stimuli and generated (simulated) output signals are shown. In the first line of the simulation file, the clk_40 is shown, set to 40 MHz $\hat{=}$25 ns. 40 MHz is selected instead of 41.6 MHz to get a correlation with the 100 ns guide of the display. The pipe_en is on for the whole simulation because the result is processed during the data taking period (pipe_en = 1) only. The fst_wd is set according to the signal from the CIPix board. The duty cycle is 25 to 75 ns. It must not change its value near a positive clk_40 edge to meet the *setup and hold* requirements of the APEX FPGA. The select signal (sel[1:0]) is derived from the fst_wd signal as described in Chapter 8. The sum card expects track numbers ($\hat{=}$ input vector) from the three regions of the z-vertex histogram of each input card (three regions of each trigger card or pre sum card (as in this simulation)). In total 12 (8-bit) input vectors are linked into the system. Note that all input signals to the sum cards are transferred in an active low logic (see Section 8.3.1). Thus, these input stimuli are given to the simulation in an active low logic. The simulation was done with the default settings, hardwired accessible at the FPGA after a reset (!n_g_rst = 0). Thus, any simulation of programming the FPGA via VME-bus are not necessary and not shown. The trigger elements are defined as shown in Table 7.1. All signals, not used (like VME bus signals) in this simulation, are dropped from the Figure.

[1] To register any input signal (like chamber signals) with the current clock (clk_40), the input signal has to be linked into the FPGA in the correct time window (not at the **setup-and-hold** times)

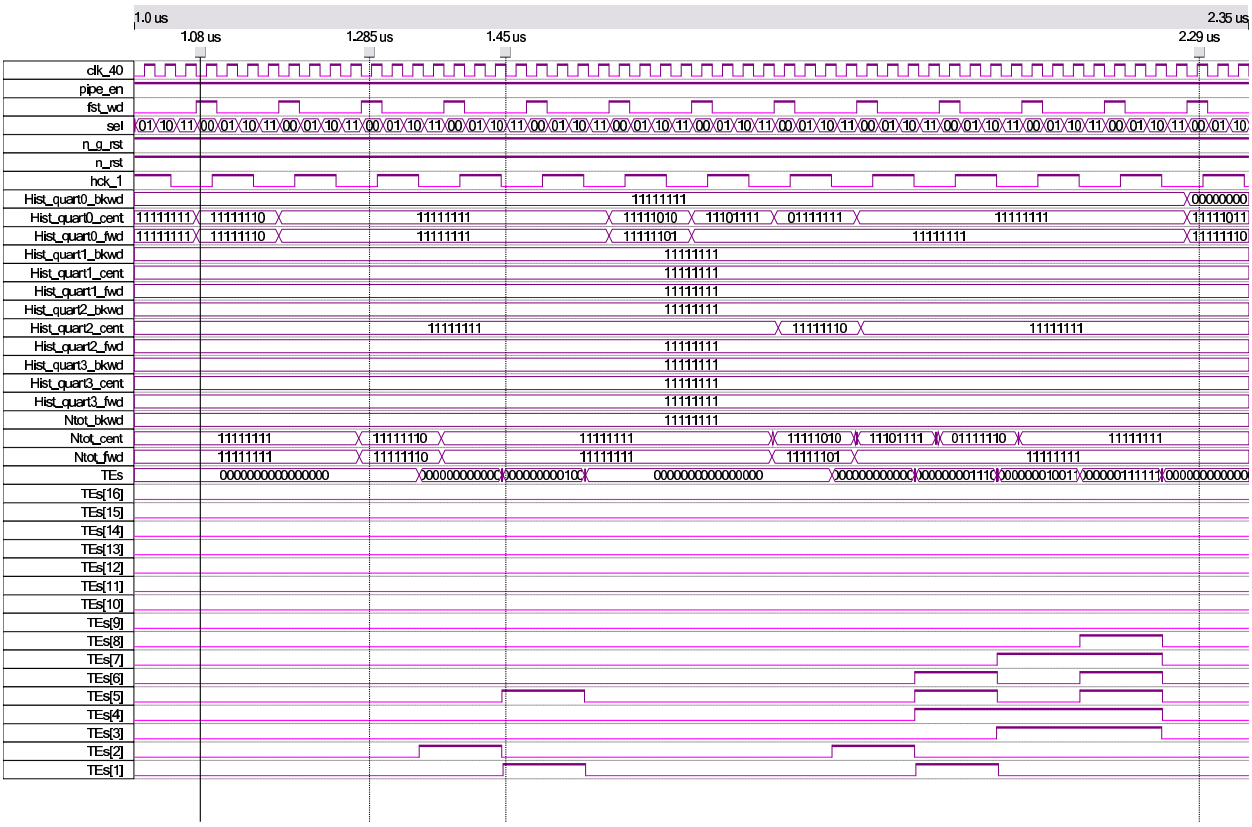


Fig. 9.1: Simulation of the sum card trigger element generation. The input signals are set according to the expected signals in the real system. All unused signals (i.e., VME bus signals) are removed in this sample simulation.

In the first test, one track was put into the simulation in the central and backward region of one quadrant (Hist_quart0_cent and Hist_quart0_fwd of one pre sum card, time bar at 1,08μs). The result of the `mux_adder.v` module (Ntot_cent, Ntot_fwd) is accessible after two BCs (200 ns), shown in Figure 9.1, corresponding to the description in Chapter 8, Section 8.4.2.2. The result of the `mux_adder.v` is linked to the trigger element builder, evaluating the trigger elements. The result is linked to the output if the condition `sel = 2` (programmable via VME, default = 2) is fulfilled. With one track in the central region, the CIP_T0 ($\hat{=}$TE[1]) has to be on. One BC earlier, the CIP_T0_NEXTBC ($\hat{=}$TE[2]) is on, as shown in the Figure. The multiplicity ($\hat{=}$TE[7:5]) is set to 1 (TE[5] = 1), according to the trigger element definition; at least one track, but not more than two tracks.

In a second test, more tracks were switched on to test the other TEs. Additionally, the track numbers were changed one after another with the next BC. Thus, it is shown that the algorithm also works fine if there is more than one event in the pipeline. Three BCs were filled as follows:

1. Five tracks in the central region, two tracks in the forward region, in both cases from the same quadrant.

Result: CIP_T0 and CIP_T0_NEXTBC are on, as expected. The significance ($\hat{=}$TE[4:3]) is set to 2, because there are more than twice as many tracks in the central region as in the forward or backward region (5:2). The multiplicity is set to 3, because there are more than 6 tracks in total (5+2=7).

2. 16 Tracks in the central region from the same quadrant, no tracks in other regions.

   Result: The CIP_T0 and CIP_T0_NEXTBC are **not** switched on because these signals are suppressed due to the CIP T0 signal one bunch crossing earlier (see Section 8.4.2.2). The significance is set to 3 (maximum), because there are more than four times as many tracks in the central region as in the forward or backward region (16:0). The multiplicity is set to 4, because there are more than 10 tracks in total (16+0=16).

3. 128 Tracks in the central region of quadrant 0, in addition one track in the central region of quadrant 2, no tracks in other regions.

   Result: Again the CIP_T0 and the CIP_T0_NEXTBC are **off**. The Significance is set to 3 (maximum), (129:0). The multiplicity is set to 7, because there are more than 100 tracks in total (128+1=129). In addition, the CIP_COS (TE8) is switched on, because there are tracks in different quadrants.

- **Trigger card, readout of chamber data via VME bus:** In the second example, a readout cycle of the chamber data is analyzed. A window of 8.4 μs to 10.5 μs is displayed in Figure 9.2. The total simulation spreads over 20 μs. Preceding programming of VME registers (like BC_offset) is not shown in the Figure. At $\approx$8,5μs the pipe_en signal is set to 0. The pipeline stops. The registered pipe_en_reg_neg signal changes with the next (sel = 0) state, since the chamber signals of all multiplexer steps of the current BC have to be written to the memory (as described in Section 8.2.2.1). This is finished at 8.6 μs, indicated by a vertical bar.

  Induced by the change of the pipe_en_reg_neg signal, the internal memory address pointer addr_int is now assigned to the memory address **read** pointer (addr_int_rd). This pointer now gets the information of the last write position of the memory address **write** pointer (addr_int_wr) with the BC offset subtracted. The value of the memory address pointer addr_int changes from 23 to 119, the programmed offset is 32, corresponding to 8 BCs (23-32 = (-9)+128 = 119).

  Moreover, the pipe_en_reg_neg signal serves as the read-write flag for the memory. The memory address pointer (addr_int) now jumps to the address 119, and the value at this address is read out into the register `memory_reg[31:0]`. memory_reg = h8000 8000. Bit 31 carries the information of l3_keep, bit 15 carries the information of the first word, written into the memory at the same time. Since the offset was programmed correctly, the data at VME word 0 does carry the fst_wd information. Otherwise, the offset has to be changed accordingly. This is shown in the Figure at 9.1 μs (time bar). The first VME readout address is 0, the corresponding memory address pointer (addr_int) is 119. With the read_apex_VME VME bus

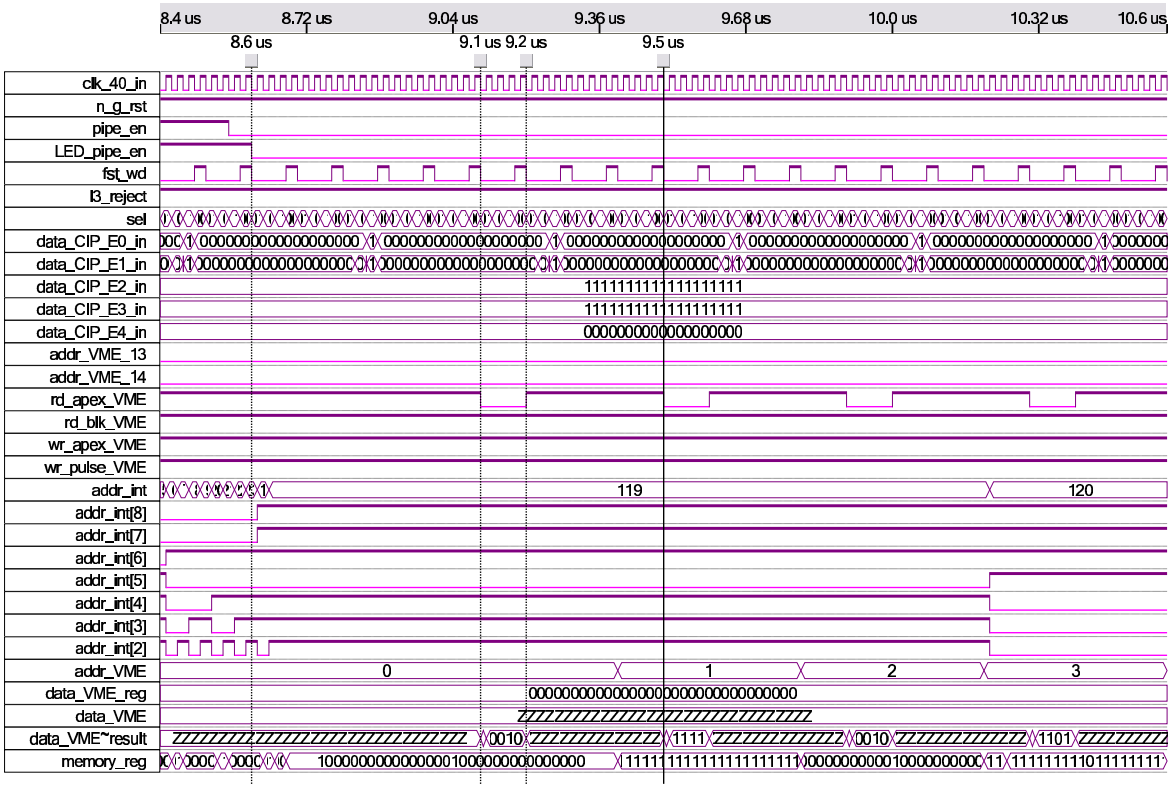


Fig. 9.2: Simulation of the trigger card readout of chamber data via VME bus. Again, all unused signals (i.e. VME bus signals) are removed in this extraction of the simulation.

read pointer (and addresses 13, 14 set to 0) the information from the memory_reg is handshaked to the VME register `data_VME-result`[2]. The read_apex_VME is set to 100 ns. With the next VME cycle, the information of layer 2 and 3 is read out (time bar at 9.5 μs). The inputs (`data_CIP_E2` and `data_CIP_E3`) were set to 1 for every channel. Therefore, in the read out VME word, this information is also set to 1. Furthermore, the read out fst_wd signal is set to 1, indicating the first multiplexer step (mux=0). With VME read out address 3 (at 10.3μs), the information of layer 3 and 2 of the second multiplexer step (mux=1) are read out. The address pointer goes to 120. Now, the fst_wd has to be off, indicated by the 15th bit.

9.3 Tests with a Pattern Generator

Nearly every process was simulated to evaluate the behavior of the design before programming (configuring) it into the FPGA. Due to the explicit setting of all input stimuli, the simulation is good for tests of the general functionality. However, it is impossible to simulate every possible behavior of the design. Thus, tests on the real hardware were done. Before testing/using the trigger system with real chamber data, a

[2]data_VME is a tristate bus signal. In case of writing something onto the bus, the simulation tool indicates this with the appended result.

pattern generator was connected to analyze the correct functionality of the developed firmware and, moreover, to understand inconsistencies in the algorithm. Furthermore, the hardware description was checked in detail. For this creating well defined input signals was very helpful. Figure 9.3 gives an overview of tests done with the pattern generator.

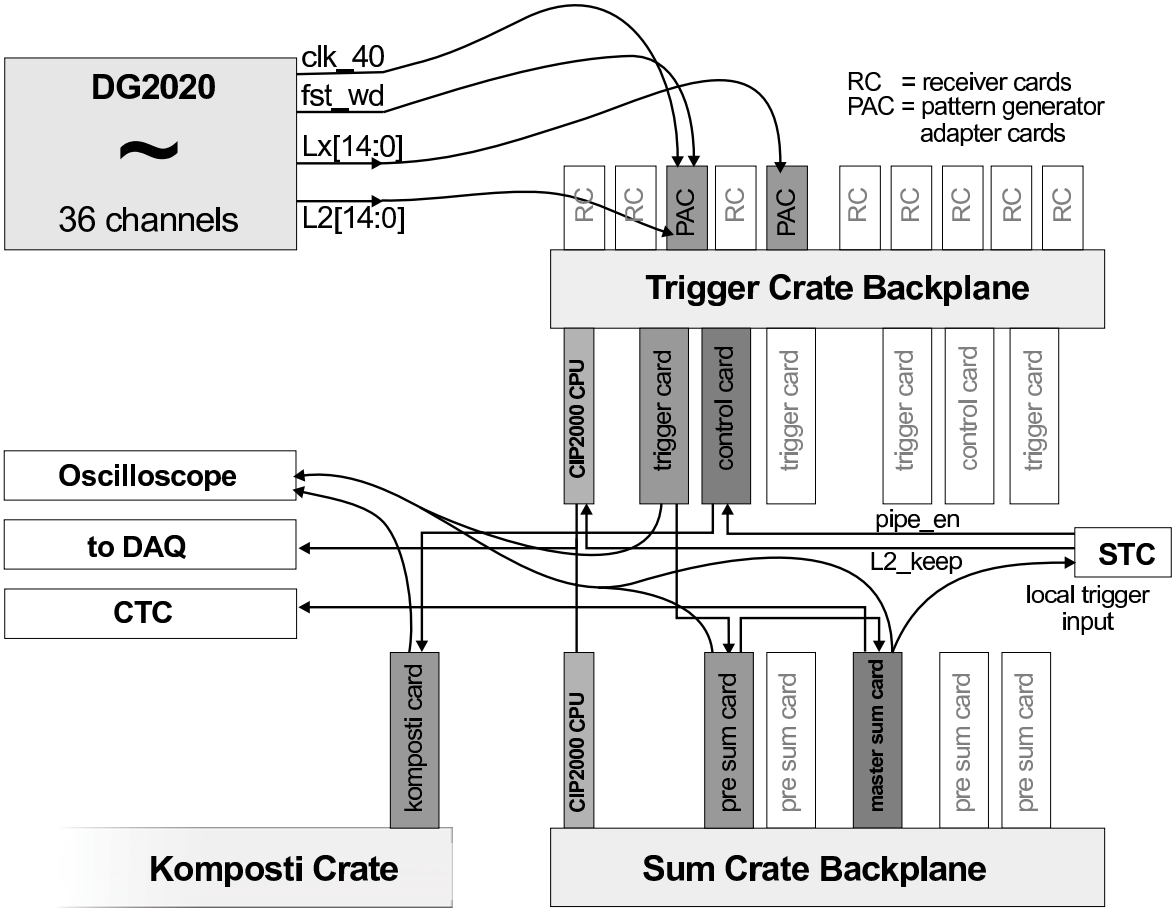


Fig. 9.3: Schematic view of the tests with the pattern generator. The receiver cards (RC) were removed and pattern generator adapter cards (PAC) were connected instead. The pattern generator generates the clk_40 and the fst_wd and 15 signals per PAC. Tests were done with the oscilloscope, DAQ and/or local triggers/CTC.

The pattern generator (type: Sony-Tektronix DG2020a [98]) generates a total of 36 outputs with a pattern memory of 64 kB and a maximum clock frequency of 50 MHz. To adapt these signals to the system, the receiver cards (RC) were removed and specially developed pattern generator adapter cards (PAC) were connected to the trigger crate backplane instead. The pattern generator generates the clk_40 and the fst_wd signal of layer 2 and, in addition, 30 signals for a total of two PACs with 15 channels (of a total of $4 \times 15 = 60$ channels $\hat{=}$ 4 CIPixes) each, feeding two layers of the trigger card simultaneously. The track finding algorithm can be tested with this setup because the other layers can be switched **on** or **off** in the trigger algorithm. The PAC signals, not connected to the pattern generator were set to ground. The information of the two PACs is sent to one trigger card, shaded in the Figure. Curved lines in the

Figure indicate variable lines for the measurement, the rest of the system is used in its design-functionality (see Chapter 6). One pre sum card is receiving the information of the trigger card and provides information to the master sum card. There the trigger elements are generated. At the trigger card, φ-wise trigger elements in each φ-sector and the number of tracks in the regions can be independently measured with an oscilloscope. At the pre and master sum card, the same information of a quarter or the full CIP2k is available (for trigger tests). Instead of using an oscilloscope, the information can be sent to the STC crate, to be used as a local trigger to stop the readout in case of a (local) trigger (for DAQ tests). Since the information of the generated pads is also written into the ring memory of the trigger card, it may be read out by the VME processor (for DAQ tests).

To test the functionality of the komposti cards, the signals from the PAC were used to identify the correct routing and assignment of the chamber pad information at the trigger crate backplane and through the control card and komposti card to the old z-vertex trigger receiver card input.

Two of the tests performed with this pattern generator are described below:

I. **Track finding consistency checks:** The signals of two PACs, acting as receiver cards, were given to the trigger card. Two of the three remaining layers were set to 1 (layer 3, 4), layer 0 was set to 0. Thus the trigger card operates in a 2-out-of-3 layer option. A test pattern was programmed into the DG2020 that produced a predetermined *track pattern*. In Figure 9.4, all patterns that cause a recognizable track for one pad of layer 2 are shown. Thus, according to the algorithm described

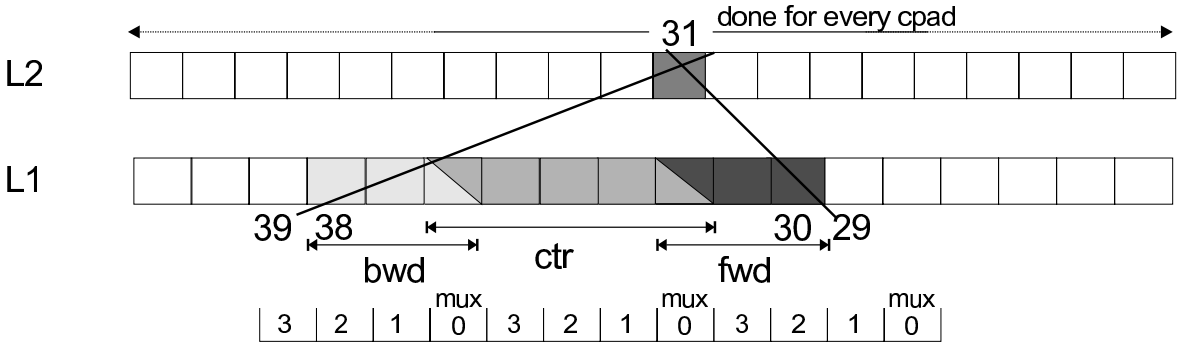


Fig. 9.4: Idea of the pattern generator tests: For one pad of layer 2 (here: cpad 31), all pads of layer 1 were switched on one after the other. The fwd-region is expected, if pads 30, 31 and 32 are switched on, the ctr-region is expected with pads 32-36 and the bwd-region is expected with pads 33, 37 and 38. Pad 29 of layer 1 is in mux step 1, pad 30 in mux step 2 and pad 32 again in mux step 0, corresponding to the fst wd position.

in Chapter 7, nine pads in layer 1 generate trigger signals corresponding to the nine pads of the local environment of one pad in layer 2 (cpad). Only tracks in the three regions can be measured at the output of the trigger card (but not the 15 bins of the z-vertex histogram). The distribution of the tracks into the three regions is shown in the Figure. With the oscilloscope it was checked that the generated tracks were found in the correct region and that other tracks (outside the local environment) did not generate tracks at the output. With these tests,

the functionality of the demultiplexer, filter, trigger, adder and mux_adder in the trigger cards and sum cards were checked:

- **Demultiplexer:** By shifting the pads in groups of four according to the fst_wd, each demultiplexer step was verified to produce the track pattern, as shown in Figure 9.4: Pad 39 has to be in mux step 3, Pad 38 in mux step 2 and so on. A wrong demultiplexing leads to wrong track results.
- **Filter:** By switching on and off layers, the 2-out-of-3 layer option was tested. By switching on single pads in the filter mask (see Chapter 8, Section 8.2.2.2), the same track patterns as delivered by the pattern generator can be observed (In this case the track pattern is activated permanently).
- **Trigger:** The track finding algorithm is tested. If the track is recognized correctly, it can be found in the correct region.
- **Adder and mux_adder:** By switching on more than one test pattern in one region, the adder modules can be tested.
- **Border and overlap pads:** By programming the test pattern in the overlap area (described in Chapter 6, Section 6.2.3) or near the borders of the chamber (described in Chapter 8, Section 8.2.2.2), the correct assignment of overlap pads and virtual (border) pads were checked.

II. **DAQ consistency checks:** With the pattern generator, a powerful handle is given to check the readout of the chamber signals via VME bus. Two major problems were observed during the maintenance of the FPGA DAQ/readout code (as described in Section 8.2.2.1):

- The number of BCs passed between the event occurrence and the arrival of the trigger (pipe_en) had to be measured, corresponding to BC_offset setting.
- The correct sorting of the pad information, distributed across five memories and four multiplexer steps in the correct order into VME words is verified.

The test described below was very useful to solve some problems:

A 32 steps counter (5 bit) was programmed into the pattern generator for 5 pads of layer 2. This way the mapping of pipeline steps, layers, first words to VME words (Figure 8.8) was verified.

In addition, one track pattern in the central region was programmed, active for one out of 32 BCs. Figure 9.5 shows the idea of this test.

1. At BC 12, the test pattern is set.
2. The track pattern is recognized as a track in the central region by the trigger algorithm. Therefore, the CIP_T0 signal is set. This signal is linked into the STC crate and can be used to generate a local trigger (visible in Figure 9.3).
3. Following this trigger, the pipeline is stopped. This is seen by the trigger card (pipe_en=0). The write cycle stops and the readout is initialized. Before starting this "test run" the VME readout offset was programmed into the FPGA (BC_offset[8:2], described in Chapter 8, Section 8.2.2.1).

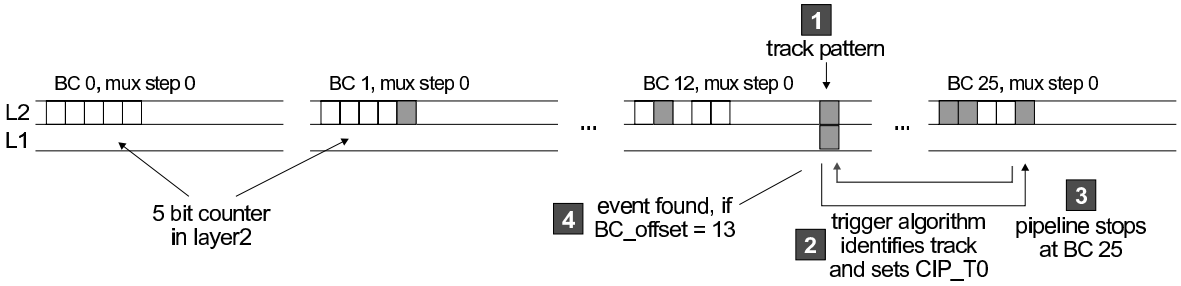


Fig. 9.5: The idea of the DAQ consistency checks with a pattern generator are illustrated. The four steps are described in the text.

4. If the correct offset was programmed, the track pattern is found in the read out data (in the middle of five pipeline steps). Otherwise, a different offset value has to be chosen. In this case, the 5-bit counter indicates the direction where the track pattern can be found.

By finding the track pattern (anywhere) in the read out BCs, the correct pipeline position offset (bits [8:4]) is found. By finding it, moreover, in the correct mux step (together with the fst_wd), the correct offset within the BC (bits [3:2]) is found. Thus, at the end of these tests, most of the inconsistencies in the VME interface and readout software were identified and removed.

The full electronic chain was tested by pattern generator tests. However, only a small fraction of the 10000 input channels may be tested simultaneously this way. The trigger algorithm alone is better tested by comparing the response of the trigger system to real chamber data, in conjunction with a software simulation of the trigger system. This is shown in the next Section. However, this requires a working DAQ system, fully integrated in the H1 environment.

9.4 DAQ-aided Tests

The idea of the DAQ aided tests is to compare the evaluated trigger decision (trigger elements) of the hardware (trigger system) with the results of a simulation, written in the programming language C. These tests can be done with cosmics data, *ep*-data or simply with chamber noise. The chamber signals are stored with the H1 data in the **CRM2**-data bank[3], then they are passed through the simulation to evaluate a trigger decision. It can be compared to the hardware trigger decision, as shown in Figure 9.6. The trigger decision, processed by the simulation, should exactly match the hardware trigger decision. In order to check the evaluation of the hardware trigger decision step by step, the z-vertex histogram and the number of tracks in the three z-vertex regions (both, pre and master sum cards) can be stored for every event as described in Chapter 8. The z-vertex (φ-)histogram of every trigger card is written into the **CRMZ**-bank. The number of tracks in the three regions of each pre sum

[3]The bank for the storage of the pad information of the former CIP trigger system is the CRME bank. The new bank is called CRM2-bank

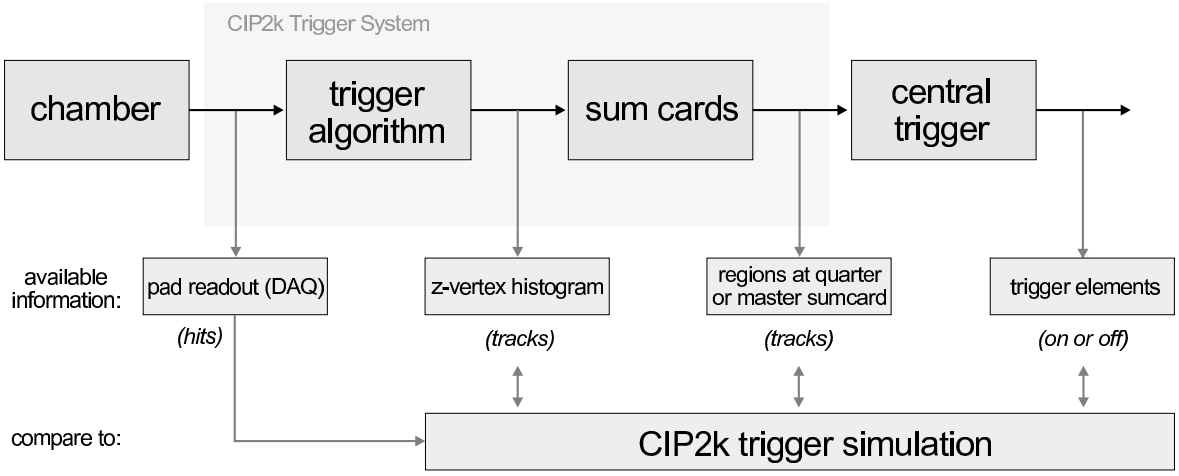


Fig. 9.6: Scheme of the DAQ-aided tests: Evaluated results of the trigger hardware are compared to the reconstructed information of the CIP2k trigger simulation.

card and the master sum card are written into the **CRMS-**bank. The trigger elements are not read out by the CIP trigger system, but this information is available in the **TEL1**-bank (Trigger Elements on L1).

To read out the CIP2k z-vertex histogram, four VME words per BC and trigger card are necessary, so in total $4_{VME} \times 5_{BCs} \times 16_{\varphi} = 320$ VME words have to be read out. This leads to an additional delay of the readout. For this reason, the z-vertex histogram is read out for special runs only.

Trigger inconsistencies: With the comparison of simulated trigger decision and hardware readout at many different positions of the trigger pipeline, the DAQ-aided tests were a powerful tool to find trigger inconsistencies. Inconsistencies were then traced back to an error in the firmware or the hardware (backplane, defect trigger card). Investigations to find this error began by looking at the trigger elements, as illustrated by an example: Figure 9.7a. shows a comparison of simulated trigger elements (using chamber data) and read out trigger elements (from sum cards) for five events of the trigger pipeline. If the simulation matches the hardware, the event is plotted at the diagonal. If the event is only seen in the simulation, it is plotted near the y-axis. If it is only seen in trigger pipeline, it is plotted near the x-axis. The small box at $x = 1$, $y = 0$ indicates a timing error. Inconsistencies appear with a low rate, seen by unexpected entries off the diagonal. However, the cause of the problem could not be found. Figure 9.7b shows the fraction of events with inconsistencies between simulated and hardware recognized events, now separately for each bin of the z-vertex histogram. Looking at this distribution of inconsistencies in each bin, it is clearly visible that problems only occur at odd bins with a rate of ≈3% (and in bin 12). This is a good hint that the error may be (and finally was) found in the definition of the asymmetric tracks in the trigger kernel (Section 8.2.2.2). Additionally, two pad assignments were swapped, when calling submodules for bin 12 (see Figure at bin 12). This example shows that detailed studies had to be done to find all sources of inconsistencies. Having done several of those studies, the actual trigger algorithm firmware does not show any more

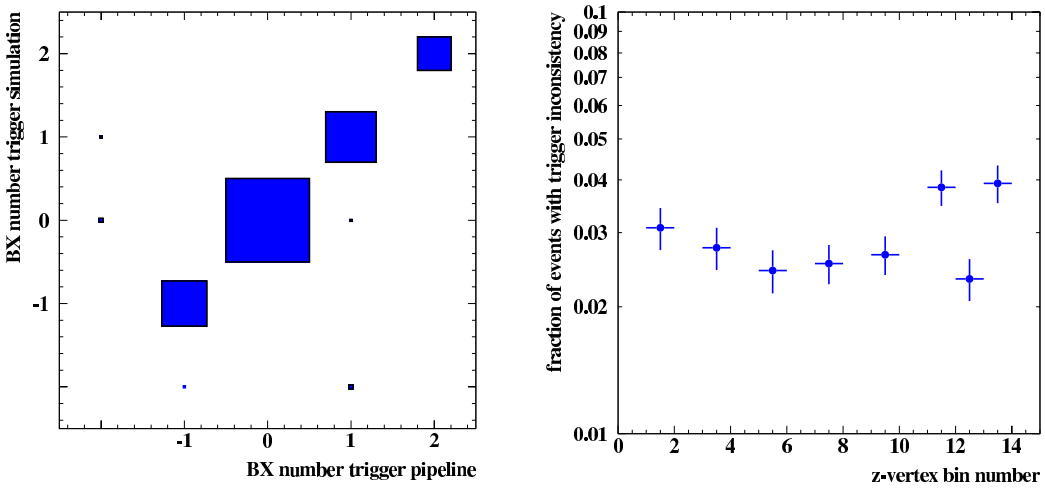


Fig. 9.7: Example of an inconsistency between hardware evaluated trigger and simulation: Figure a shows the inconsistencies by comparing the trigger element information. In this diagram, the cause of the inconsistency can not be located. Figure b shows the rate of inconsistent events (of the same run) distributed along the z bins. Here it is clearly visible that there is an error at the recognition of odd bins (both plots: run nr. 341378, cosmics data).

inconsistencies.

9.5 Results

The DAQ-aided testing of the trigger algorithm is a very powerful tool to maintain the CIP2k trigger system and to identify inconsistencies.

As a result of these tests and the ones introduced before, the decision of the FPGA trigger is found to be **100%** consistent with the simulation. This proves that the development of the CIP2k trigger system has come to a successful end. In Chapter 10, further optimization to the H1 conditions within the HERA II operation is described. Most of the optimization is possible by changing values in registers, programmable via the VME bus. Major hardware changes are no longer necessary.

With a permanent comparison of the hardware evaluated trigger decision and the corresponding simulation, the functionality of the CIP2k z-vertex trigger is monitored. This information is available online (see [88]).

Chapter 10

CIP2k System Performance

After proving the general functionality of the trigger system, the performance of the system has to be analyzed. For this purpose, readout and trigger of the CIP2k trigger system in its final configuration are compared event-by-event to the response of other detectors. The H1 central drift chambers (CJC1, 2) were used to perform this task.

The performance of the new trigger system is analyzed in two major steps:

- For the performance analysis of the CIP2k chamber (high voltage, pad efficiency, front end electronics, timing) and the track recognition efficiency of the trigger system, minimum ionizing particles from cosmic rays were used. Cosmic rays are very useful because the event rate is sufficiently high and they have a clear signature in the H1 detector. Thus precise single track efficiencies can be evaluated.

- To analyze the timing information and the separation power of the CIP2k z-vertex trigger, beam data were used. Events taken in ep-runs since October 2003 without using the CIP2k trigger are used to compare triggered results with offline evaluated results using the CIP2k trigger. Studies using subtriggers with an additional CIP trigger element (**or**) are used to determine the trigger efficiency.

10.1 CIP2k Trigger Performance with Cosmic Rays

The performance of the new trigger system is analyzed with cosmics data taken during Summer 2003.

The analysis of the CIP chamber and trigger performance is done by selecting events, where a single muon crosses the central jet chamber in two sectors opposite in azimuth with a maximum distance of $5\,cm$ from the beam axis. This way, the chamber response on two track events may be studied. Figure 10.1 shows the selection of such cosmics tracks. In Figure 10.1 (right), the z-range of the muon sample is shown. It is limited to $\approx \pm 110\,\mathrm{cm}$ due to the chamber borders of the CJC. It cannot be exceeded if the cosmics track are required to cross both halves of the chamber [100].

An analysis of the CIP2k trigger is done step by step as shown in Figure 10.2. Selected examples of these studies are discussed in the following sections.

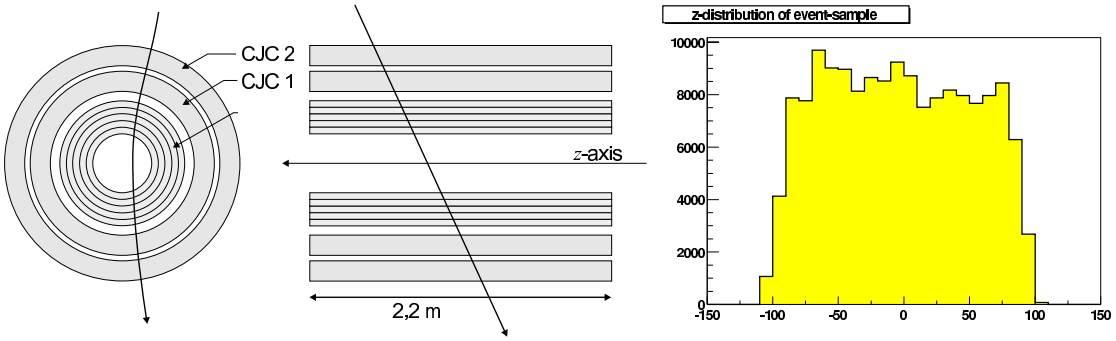


Fig. 10.1: Selection of particles in cosmics: Muons have to cross both CJC halves. The maximum distance from the beam axis is required to be below 5 cm.
On the right, the z distribution of the muon event sample is shown.

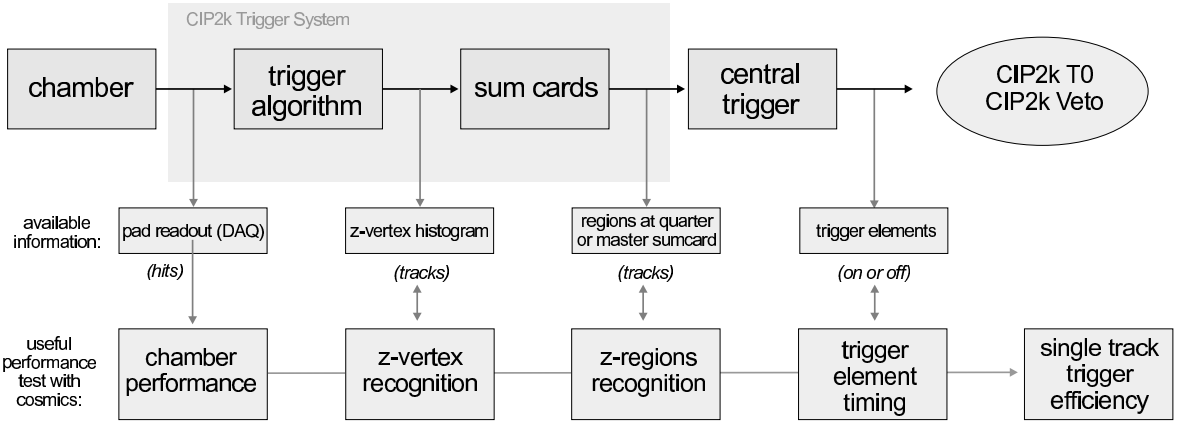


Fig. 10.2: Scheme of the test of the trigger system step by step with cosmics data: First, the chamber performance is checked. Then, the recognition and sorting of tracks in bins and regions (of the z-vertex histogram) is shown. Finally, timing tests of the delivered trigger decision are performed and the single track efficiency is evaluated.

10.1.1 Chamber Performance

In order to check how efficient tracks detected by the CJC are also detected by the CIP chamber, CJC tracks are extrapolated to the CIP2k chamber pad position of a selected layer. It is then checked whether CIP2k pads nearby are active. The z-vertex position of this CJC track is plotted versus the correlated pad position as shown in Figure 10.3 (small box).

To determine the z-resolution of the CIP chamber, the distance between each active CIP pad and the corresponding CJC track is plotted, shown in Figure 10.3. The z-resolution of the chamber is about $2\,cm$, estimated with a best fit Gaussian. Each plot represents the information of one layer in one φ-sector. In total $5{\times}16 = 80$ plots are available. Layers that do not show a clear correlation (i.e. due to high voltage or CIPix problems), are switched off in the `filter.v` (Section 8.2.2.2) module of the trigger algorithm.

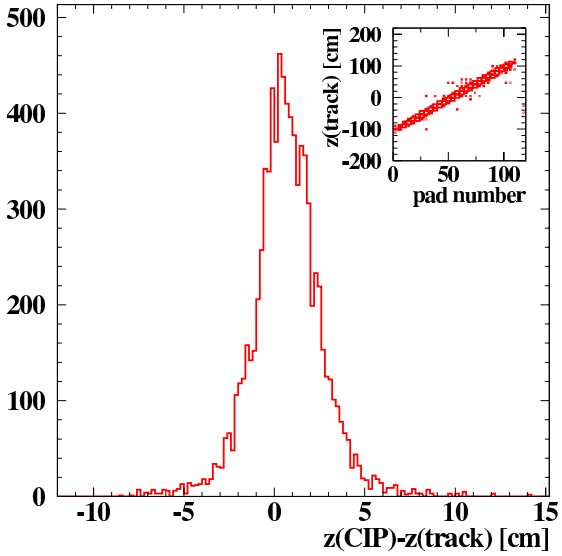


Fig. 10.3: The distance of the CJC track to the corresponding pad is shown for layer 0, φ-sector 3 (Run 346178).

10.1.2 z-Vertex Reconstruction

Correlation of the CIP2k z-vertex histogram with CJC tracks: Having determined the resolution of the chamber, the correct assignment of recognized tracks to bins (track finding) and the counting of tracks in the trigger is tested.

This is done by comparing the extrapolated z-position of the cosmics recognized by the CJC with the tracks seen in the CIP2k. Since the CIP2k trigger does not provide track information of each detected track, the tracks sorted into the bins of the z-vertex histogram have to be used instead. Each cosmic muon of the selected data sample generates two tracks in the CJC (see Figure 10.1a). For each event the z-position at the beam axis (z_0) of a combined fit of these CJC tracks is compared to the number of CIP2k tracks **in each bin of the z-vertex histogram**, weighted with the number of tracks in each bin (N(bin_i)) divided by the total number of tracks in all bins (N(all)). The weighting W for each bin of a single event looks as follows:

$$W_i = \frac{\mathrm{N}(bin_i)}{\mathrm{N}(all)} \tag{10.1}$$

This is done for all events. The result is shown in Figure 10.4. If in most of the events the tracks are in the bin nearest to the z_0 of the CJC track, a correlation is expected. At the ordinate, the z-position of the CJC track is shown, at the x-axis **all** bins of the CIP2k z-vertex histogram are displayed.

In order to check the VME programmable z-vertex histogram position, the histogram position was shifted by 4 bins to the backward region. In Figure 10.4b the shifted z-vertex histogram is shown. As expected, the complete distribution is shifted by four bins to the right. The plots, shown here, can be compared to the histogram ranges, presented in Section 7.3: Figure 10.4a shows the z-vertex histogram in configuration 2. It covers a z-range from -120.6 cm to +125.4 cm. Figure 10.4b shows the

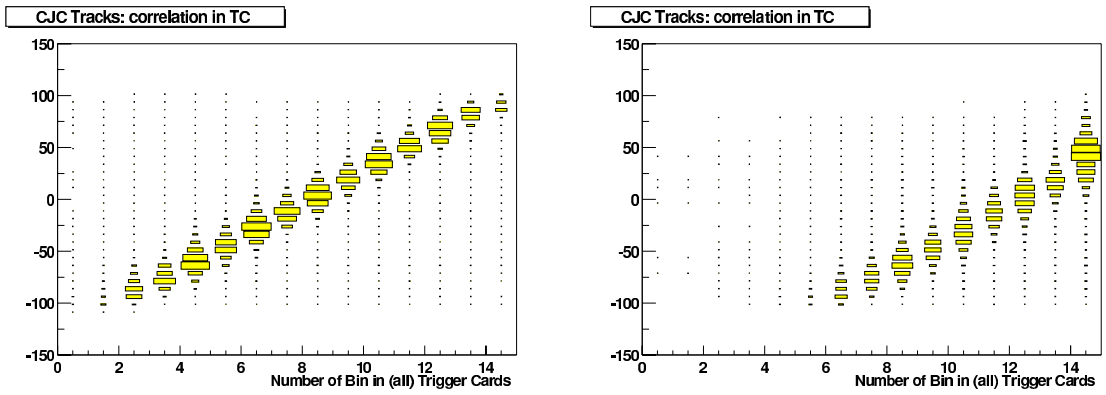


Fig. 10.4: Sorting CJC tracks in CIP2k z-vertex bins. The z-position of the CJC track is shown versus **all** bins of the CIP2k z-vertex histogram, having tracks at the corresponding CJC track. On the left, the histogram, corresponding to configuration 2 is shown. On the right, configuration 1 is shown.

z-vertex histogram in configuration 1. It covers a z-range from -186.2 cm to +59.8 cm. Note that the z-range is limited to $\approx \pm 110$ cm due to the CJC track selection.

z-resolution: To get an impression of the z-resolution of the CIP2k z-vertex trigger, the difference d between a CJC cosmic track z_0 and the CIP2k bin position z_{bin} of each bin containing tracks, is plotted. Each bin of the CIP2k z-vertex histogram is weighted as shown in equation 10.1,

$$d = \sum_{bin=0}^{14} \frac{(z_{bin} - z_0) \times \mathrm{N(bin)}}{\mathrm{N(all)}}. \tag{10.2}$$

The result is shown in the left plot of Figure 10.5. The resolution is fitted with a

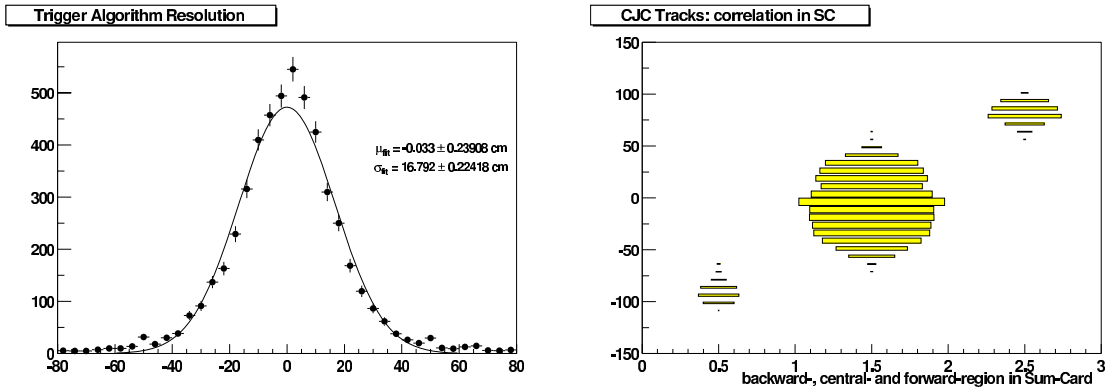


Fig. 10.5: In the left plot, the z-Resolution of the CIP2k Trigger is shown. The difference d between a CJC cosmic track z_0 and the CIP2k bin position z_{bin} of each bin containing tracks at the corresponding event is plotted. In the right plot, the sorting of CJC tracks in CIP2k z-vertex regions (forward, central and backward) is shown. The z position of the CJC track is shown versus each region.

gaussian of mean (0.03 ± 0.24)cm and width (16.79 ± 0.22)cm. This compares well to the single-track resolution estimated earlier (Section 5.2).

z-regions reconstruction An accurate sorting of tracks from bins into z-regions is tested analogous to the sorting of tracks into bins. For each cosmics event, the CJC track z_0 is plotted against the z-regions (fwd, ctr, bwd) of the CIP2k trigger, weighted by the number of tracks in each region divided by the number of all tracks seen in the CIP2k.

The plot is shown in Figure 10.5 (right). The borders between forward, central and backward region are visible (see also Figure 7.6 on page 78). As expected from the number of 2-bins contributing, most of the cosmics are in the central region.

For the plots in Figure 10.4 and 10.5 data from run 349244 were taken. The shifted histogram was plotted with data from run 352659.

10.1.3 Timing

The CIP2k trigger is used to deliver a precise information in which BC of the read out pipeline the *ep*-collision happens. This information is necessary to define a reference timing and to reject events triggered too early by other subdetectors. Once the CIP2k trigger is tuned to the correct timing, it may be used to deliver an H1 wide timing information for those events with charged particles in the CIP2k acceptance.

Before using the CIP2k trigger system as an H1 trigger, its timing has to be adjusted. All φ-dependent trigger information has to come at the same time. The timing of each sum card has to be checked. Programmable clock delays on the control card (see Section 6.2.4 on page 56) and programmable data delays in the FPGAs (see Section 8.4.2.2 on page 111) are available to adjust the timing in the system. Once the internal timing in the CIP2k trigger system is tuned, it has to be adjusted to be consistent with the timing of other H1 components[1].

To verify the correct timing, the CIP2k trigger element information is analyzed using runs, where no CIP2k trigger element was used in the H1 trigger configuration. In total 5 events around the expected event position in the pipeline are available in the H1 TEL-bank. In every event it is checked whether the CIP2k trigger element is on. Then the time reconstructed from CJC hits (corresponding to a H1 reference event time) is plotted versus the CIP2k pipeline position (BC). In Figure 10.6 the timing of trigger element CIP_COS[2] is shown. For each BC in the pipeline there is a distribution of CJC_T0 timing positions. At the expected position of the event in the pipeline (BC=0) this distribution is around CJC_T0 = 0 ns. One BC later, the CJC_T0 is distributed around 100 ns, one BC earlier, the CJC_T0 is distributed around -100 ns. Thus the correct timing of the CIP_COS trigger is validated.

10.1.4 Single Track Efficiency

Finally, the trigger efficiency is analyzed with cosmics. The number of tracks recognized (and sorted into the z-vertex histogram) along the z-axis for every φ-sector are compared to the number of tracks identified by a reference system. Again, the CJC, delivering precise track information, is used. The trigger efficiency is given by the

[1] Tuning the timing of the CIP2k trigger system was done prior to all tests presented in the preceding Section.

[2] In the cosmic run, CIP_COS, the CIP2k cosmic trigger element is used instead of the CIP_T0.

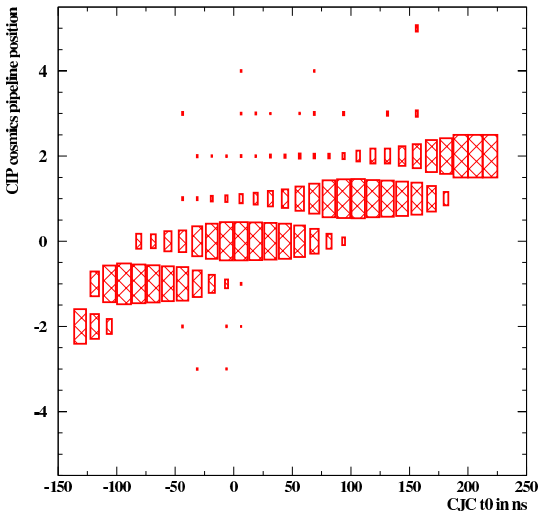


Fig. 10.6: Timing of the CIP2k Trigger: The pipeline position of CIP2k trigger elements (here: CIP_COS) is correlated to the time difference between the H1 reference timing and the CJC_T0 timing. If the CIP_COS trigger element is at pipeline position 0 (BC=0), the timing of CJC_T0 has to be distributed around 0 ns.

number of events with at least one CIP2k track, normalized to the number of CJC tracks. This is done as a function of φ and z_0 of the CJC track (see Figure 10.7).

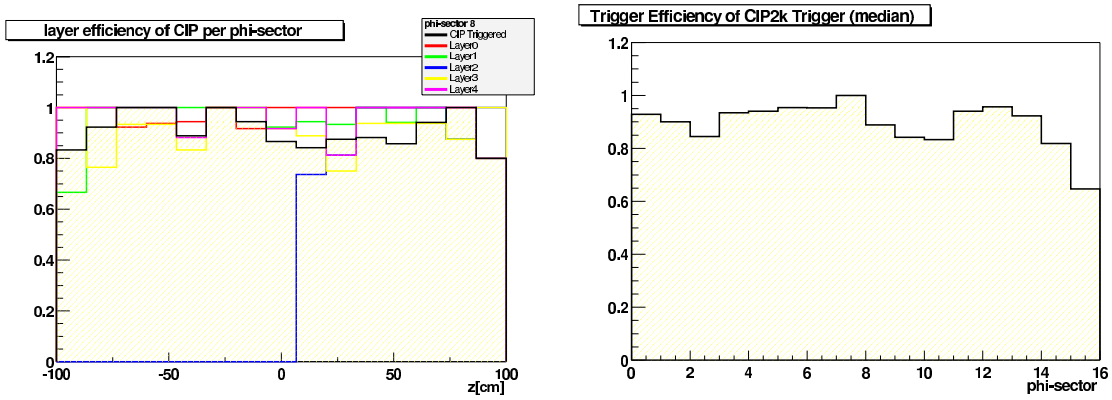


Fig. 10.7: Single track efficiency of the CIP2k Trigger: On the left plot, the efficiency is shown for a single φ-sector along the z-axis. In the same plot, the efficiencies of each single layer are visible. On the right side, the median efficiency of all φ-sector efficiencies is plotted along φ.

The single layer efficiencies of all five layers are calculated in the same way as described in Section 10.1.1 on page 130. In Figure 10.7 (left) the layer efficiency of φ-sector 8 is shown.

The median value of the trigger efficiencies of the 15 bins along the z-axis is calculated for each φ-sector. Using the median instead of the mean value of all bin efficiencies ensures that hardware problems not concerning the trigger system are bet-

ter suppressed. In Figure 10.7 right the single track trigger efficiency of the CIP2k trigger system is plotted versus φ. The overall efficiency is better than 95%, if 4-out-of-5 layers are fully functional. If more than one layer of a φ-sector is defect, the efficiency drops. In φ-sector 15 two layers are fully operational, one layer has a reduced efficiency. Therefore the trigger efficiency is below 75%. As seen in Figure 10.7 (left), the efficiency at the negative z-side of layer 2 is 0 due to a CIPix-defect, but the loss of pad information of one layer does not do any harm on the trigger performance if the trigger is set up correctly.

An overview of all plots and discussions of the results of the cosmics run in Summer 2003 can be found at [99]. Further improvements on the CIP2k system have been made since. Major topics are the optimization of high voltage settings, front end electronics settings (thresholds, amplification of chamber pulses) and trigger settings.

10.2 CIP2k Trigger Performance in ep-Collisions

The minimal requirement of the CIP2k trigger is identifying ep-events by detecting at least one particle per event in the central region of the z-vertex histogram near the interaction point (CIP_T0). Furthermore, the CIP2k trigger should be able to separate background events from ep-events (CIP_SIG, CIP_MUL) as suggested in Chapter 4. In measurements with cosmics, both features have been proven to work for single track events. Because cosmic rays with two signatures in opposite halves of the detector cover only a z-range from -110 cm to $+110$ cm and a limited range of the polar angle ϑ, ep-events are used to test the performance of the CIP2k trigger over the full phase space. The measurements presented here have been done with beam data of runs taken in October and November 2003.

10.2.1 Online Monitoring

The performance of the CIP2k trigger is monitored using special online histograms, provided for the CIP2k trigger system. Online histograms are used to get a fast response to the status of the H1 detector. Most subdetector components provide online histogram information. Online histograms contain vital information of the subdetector component, thus technical problems can be identified during data taking. They are calculated on the trigger level 4 processor farm.

CIP2k timing (CIP_T0): A basic online histogram, the event timing behavior, is shown in Figure 10.8 (displayed on the online histogram monitor Zubr [101]), where the pipeline position of the trigger element CIP_T0 versus the T0 of the CJC is plotted. Unlike in cosmics events that occur randomly across one BC, the ep-collision takes place at a fixed time in the BC. This leads to a sharp distribution in the timing diagram at t=0 ns. Compared to the timing measurement presented in Figure 10.6 for cosmics, the timing of the CJC is scaled in CJC FADC units on the x-axis. 500 FADC units correspond to one BC (96 ns). On the ordinate axis, the trigger pipeline is plotted (BC number). Note that the CIP_T0 trigger element is shortened to one BC length (96 ns), even if the detected event has CIP2k hits in more than one BC in the pipeline.

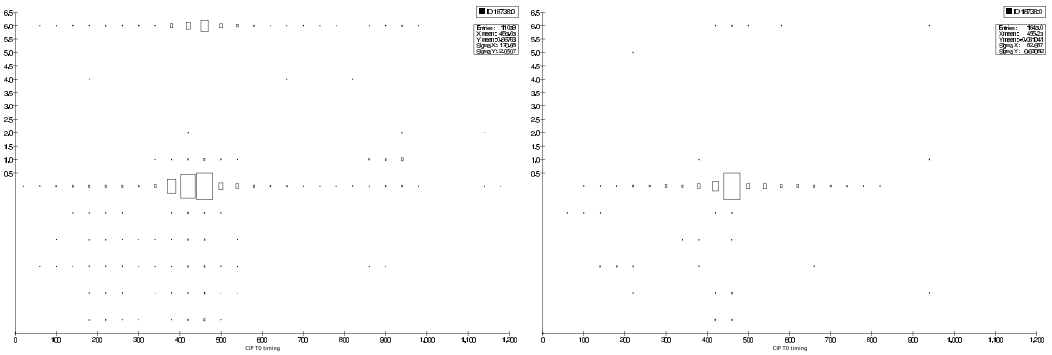


Fig. 10.8: CIP2k Trigger Online Histogram, displaying the timing of the CIP_T0 trigger element in the event pipeline versus the CJC T0 (ADC value). A correct timing is indicated, if the CIP_T0 is at the CJC ADC value 500. In the left plot, the timing is displayed for an early data run, not using the CIP2k trigger. In the right plot, the timing of the H1 trigger is displayed, using the CIP2k trigger in the actual H1 trigger settings. At pipeline position 6 are events, not seen by the CIP2k trigger.

The CIP_T0 has to be *set* at the correct pipeline position (BC=0) for each event. In Figure 10.8 (right) the CIP_T0 is displayed, using the CIP2k trigger in the actual H1 trigger settings. The entries at pipeline position 6 are events without CIP_T0 trigger element. These are events with

1. CJC tracks not passing the CIP2k or

2. huge pileups from background events more than 6 BC early. Due to the pulse shortening of the CIP_T0, these events are not triggered.

CIP2k significance: In Figure 10.9 another basic online (L4) histogram is displayed, pointing out the separation power of the CIP2k trigger. For this plot, H1 trigger settings are selected, where all H1 events with a CIP_T0 are sorted according to their CIP2k significance information as presented in Section 7.5.1 on page 79 (trigger element bits 3-5). The significance trigger element itself is not used for the trigger decision. On the x-axis, the z-position of CJC tracks is shown, plotted into either the upper left window (sig=0), the upper right window (sig=1), the lower left window (sig = 2) or the lower right window (sig=3), depending on the value of the significance (sig = 0,1,2,3) as defined in Section 7.5.1.

As shown in the plots, the CIP2k trigger is able to separate background events, arising from the backward region (z¡-80 cm), from ep-events around the IP ($\pm$ 80 cm). If the significance CIP_SIG is 0, a powerful suppression of background events should be possible. It has to be analyzed further in which way the CIP2k trigger elements can be used in the H1 trigger to support the trigger decision and how efficient this is.

A lot of investigations have been done by physics working groups. Some selected results are presented in the next Section.

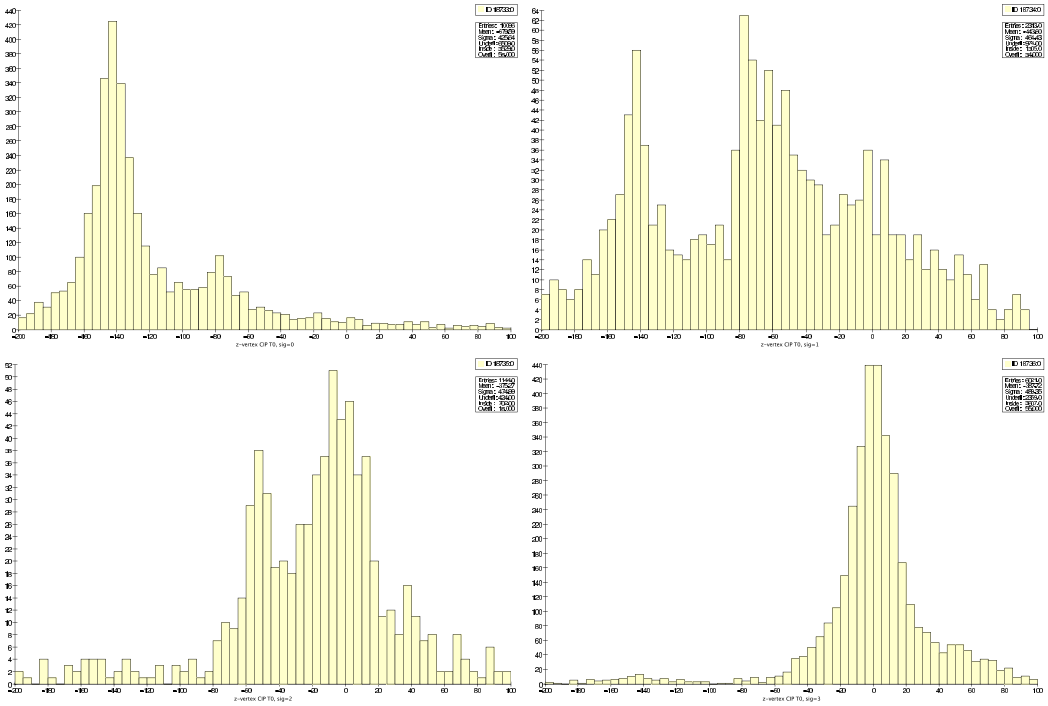


Fig. 10.9: CIP2k Trigger Online Histogram, displaying the z-vertex distribution of tracks, sorted according to the CIP2k significance information (sig = 0,1 on top, sig = 2,3 on bottom).

10.2.2 Trigger Performance Studies

The CIP2k trigger is not a stand-alone trigger. It is used in combination with other systems for several H1 subtriggers. The CIP2k z-vertex trigger may be used as:

1. **veto trigger:** Background events with a reconstructed z-vertex position in the backward region are rejected.

2. **positive trigger:** Events with a clear vertex in the correct ep-timing window are selected, depending on significance, multiplicity, etc.

Special care has to be taken about physics channels which do not have tracks in the CIP2k. They must not be rejected by the veto trigger, they can not be selected in the positive trigger either.

To find good **veto** and **positive trigger**-conditions, the effect of the CIP2k trigger elements on several subtriggers was analyzed [102]. The CIP2k trigger elements were not used, when these tests were performed. Subtriggers **s66,s67,s75,s77** are used to trigger events at high Q^2 or with large transverse energy E_t, using the LAr calorimeter:

- **s66:** Etmiss-Trigger, $ep \rightarrow \nu\mathbf{X}$. Used to trigger charged current events with a high transverse energy transfer detected in the LAr calorimeter ($LAr\_Etmiss > 2$), X is detected in the LAr calorimeter.

- **s77:** Etmiss-Trigger, $ep \rightarrow \nu\mathbf{X}$. Used to trigger charged current events in the same way as s66, but with a lower transverse energy transfer threshold ($LAr\_Etmiss > 1$), (gamma-p).

- **s67:** LAr-Electron, $ep \rightarrow \mathbf{e}'X$. Used to trigger neutral current events. The scattered electron is detected in the LAr calorimeter.
- **s75:** LAr-Electron, LAr-Electron, $ep \rightarrow \mathbf{e}'X$. Used in the same way as s67, but with a higher threshold for the triggered electron in the LAr calorimeter.

In all of these subtriggers, a T0 information is required.

Studies about the VETO condition: In Figure 10.10, the z-vertex distribution is shown, at the top as seen by subtrigger s67, below as seen by subtrigger s75.

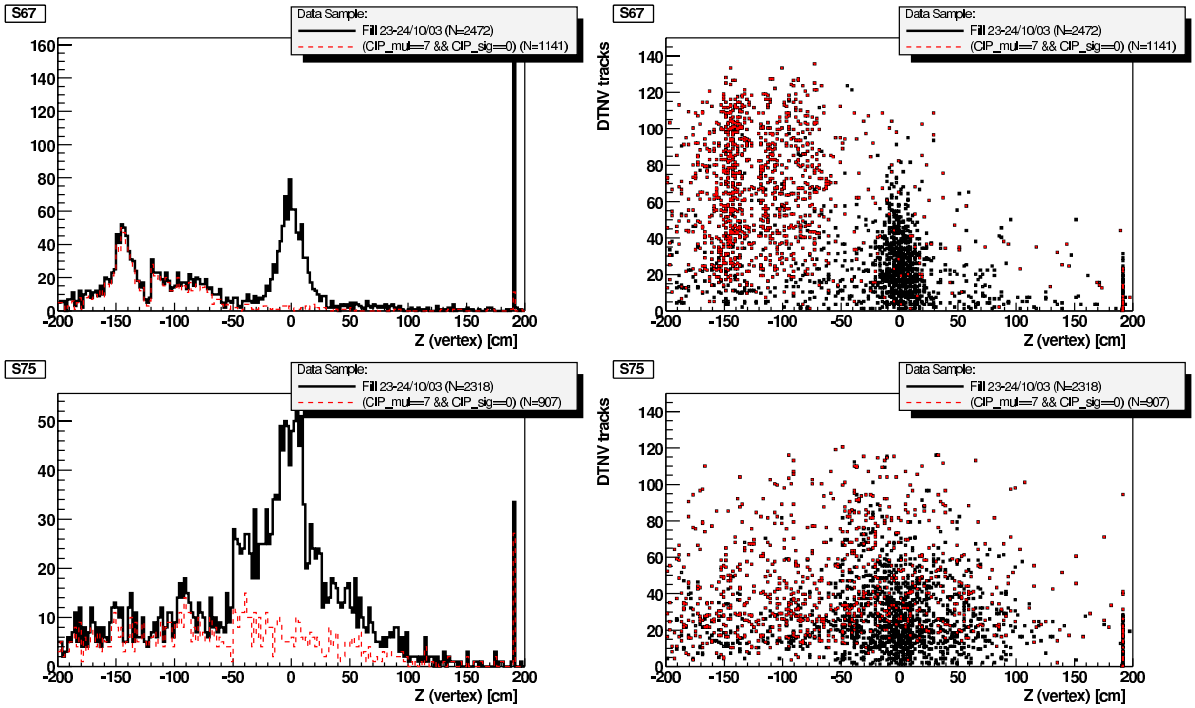


Fig. 10.10: z-vertex distribution triggered with s67 and s75. The solid line shows the total number of events. The dashed line shows the events (tracks), being rejected with the veto condition `CIP_mul==7 && CIP_sig==0` [102].

On the left side, the reconstructed vertex in all events is shown. On the right side, the number of tracks per event versus the z-vertex position is shown. The solid line shows the total number of events, the dashed line shows the events (tracks), being rejected with a proposed veto condition `CIP_mul==7 && CIP_sig==0`. This curve was reconstructed offline by cutting with the corresponding trigger element conditions. Events without reconstructed vertex are collected at $z = +190\,\text{cm}$. In the right plot, events accepted by the CIP veto are shown in black, those rejected by the veto are shown in gray.

Figure 10.11 shows the same plots for subtriggers s66 and s77.

These Figures prove that the CIP2k trigger is capable of suppressing background events arising from beam-gas or beam-pipe interactions, especially in events with a large number of tracks. Such high-multiplicity background events are otherwise limiting the performance of the L4 trigger. This emphasizes the importance of the CIP2k system for the H1 data taking efficiency.

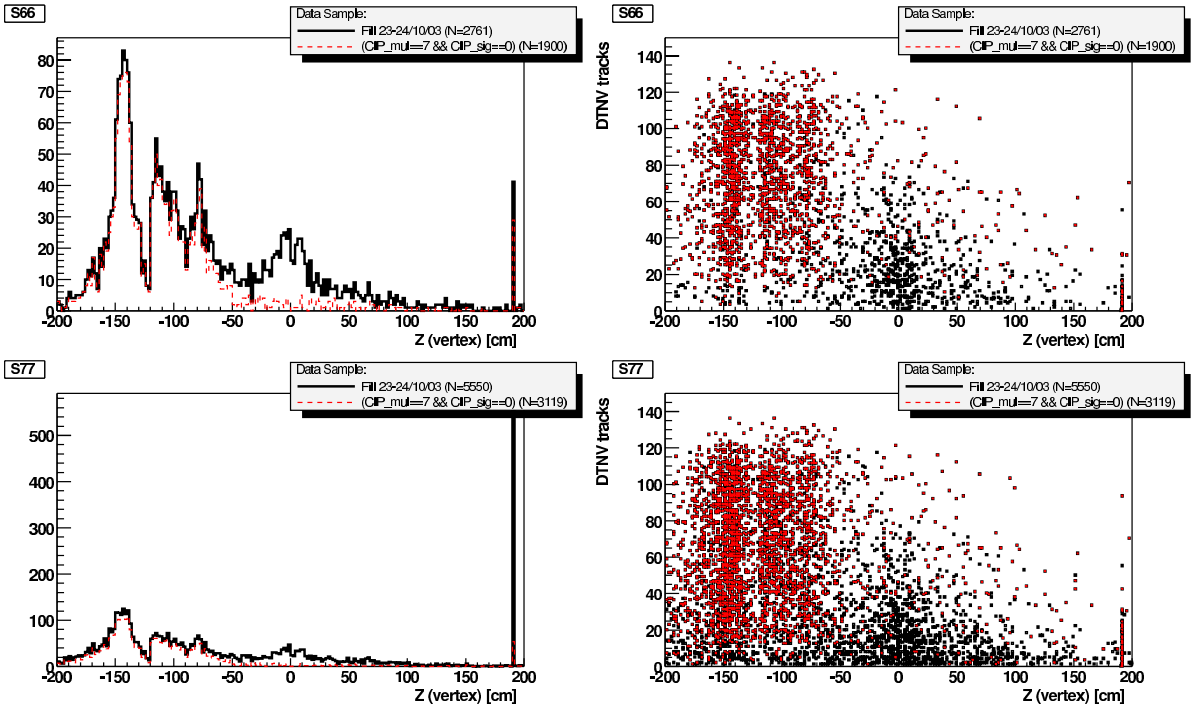


Fig. 10.11: z-vertex distribution triggered with s66, s77. As in Figure 10.10, the solid line shows the total number of events (tracks), and the dashed line shows the events (tracks), being rejected with the veto condition `CIP_mul==7 && CIP_sig==0`. On the right side, black events are accepted, gray events are rejected by the CIP2k veto condition [102].

Studies testing the positive CIP2k trigger decision:

s69 and s0 with a selection of ρ-events: Subtrigger s69 is a diffractive trigger on events with an electron in the SpaCal calorimeter and tracks in the central region of the detector. It has been tested with a selection of $\rho \to \pi^+\pi^-$-events. This subtrigger uses a veto condition of the ToF detector to cut off background events. Those background events are decorrelated in time with the desired ep interactions.

The CIP_T0 is used to determine the correct event timing. A different subtrigger is used to monitor the trigger decision of trigger elements used in **s69**. The subtrigger **s0** has the similar SpaCal trigger requirements and veto conditions, but the CIP_T0 is not required.

The performance (and efficiency) of the CIP_T0 can now be estimated by comparing the number of ρ-events with **s0** and with **s69**. The number of ρ-events collected with **s0** not fulfilling the **s69** conditions indicate that the CIP_T0 was not active. In Figure 10.12 the efficiency of the CIP_T0 trigger element is shown (run 362487 and following). The efficiency in detecting different variables of a physics analysis, studying the electro-production of ρ mesons at HERA, is displayed. A detailed description of the analysis and the measured variables can be found in [103]. The CIP_T0 efficiency measured is $\approx 96\%$.

s67 with high P_t electron-events: Subtrigger s67 uses the CIP veto condition. However, it also uses the CIP_T0 as a positive trigger. In the study presented here

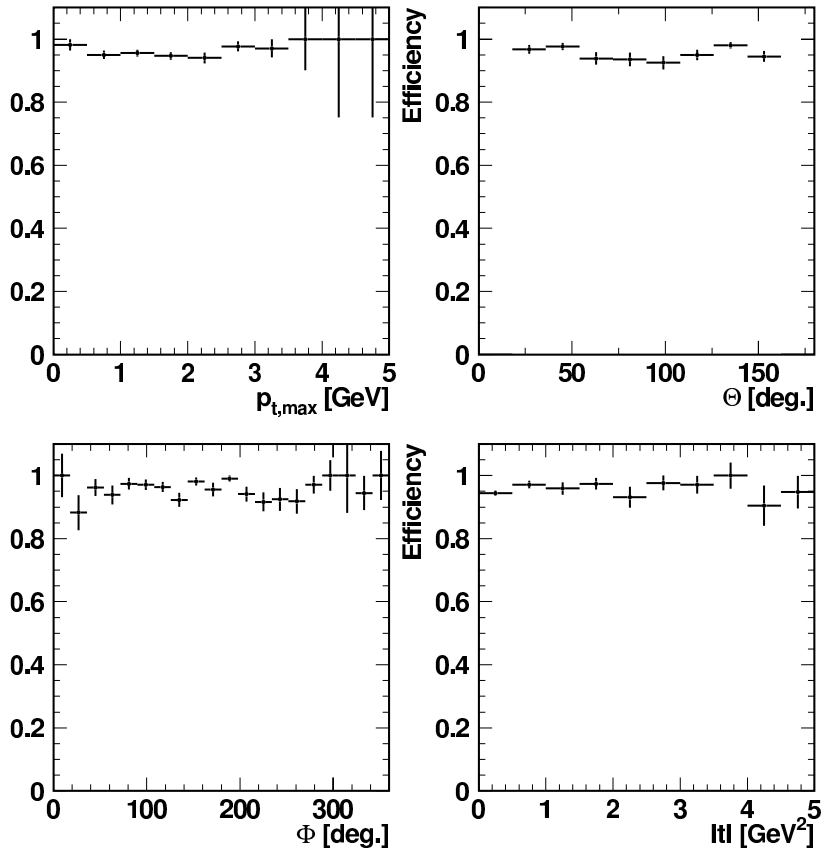


Fig. 10.12: Efficiency of the CIP_T0 trigger element measured in good ρ-events taken since mid of November. The provided efficiency is $\approx 96\%$.

[104], s67 is used to estimate the CIP2k efficiency for events with a high P_t electron, triggered in the LAr calorimeter ($ep \rightarrow \mathbf{e}'X$). In subtrigger s67, either the CIP_T0 **or** the LAr_T0 trigger condition is required. The efficiency can be determined using the LAr_T0 as an independent monitor trigger. A clean neutral current selection of events triggered only with the LAr_T0 is compared to the number of events, also triggered by the CIP_T0. In Figure 10.13 the efficiencies for some kinetic variables of the electron (**E,** φ_e, z_{impact}), are presented. The efficiencies are of order 95% without any indications of large efficiency leaks.

Example of a more elaborate use of the CIP2k trigger s111:

Since November 2003 (run 363449) a new subtrigger is defined, used to trigger elastic (diffractive) ρ events in photo-production, where the scattered proton is detected in the Forward Proton Spectrometer (FPS) and the ρ_0 meson is detected through its decay products (π^+, π^-) in the central tracker. The $\pi^+\pi^-$ pair can be identified by the CIP2k cosmics trigger element (CIP_COS), triggering charged particles in different φ-sectors. Furthermore, the significance and multiplicity trigger elements (CIP_SIG, CIP_MUL) are used to select low multiplicity events ($\rho \rightarrow 2\pi$). CIP_mul has to be between 2 and 4 (more than 2 CIP2k triggered tracks but less than 10) and CIP_sig has to be larger than 1, indicating that *all* tracks are in the central region (at least two

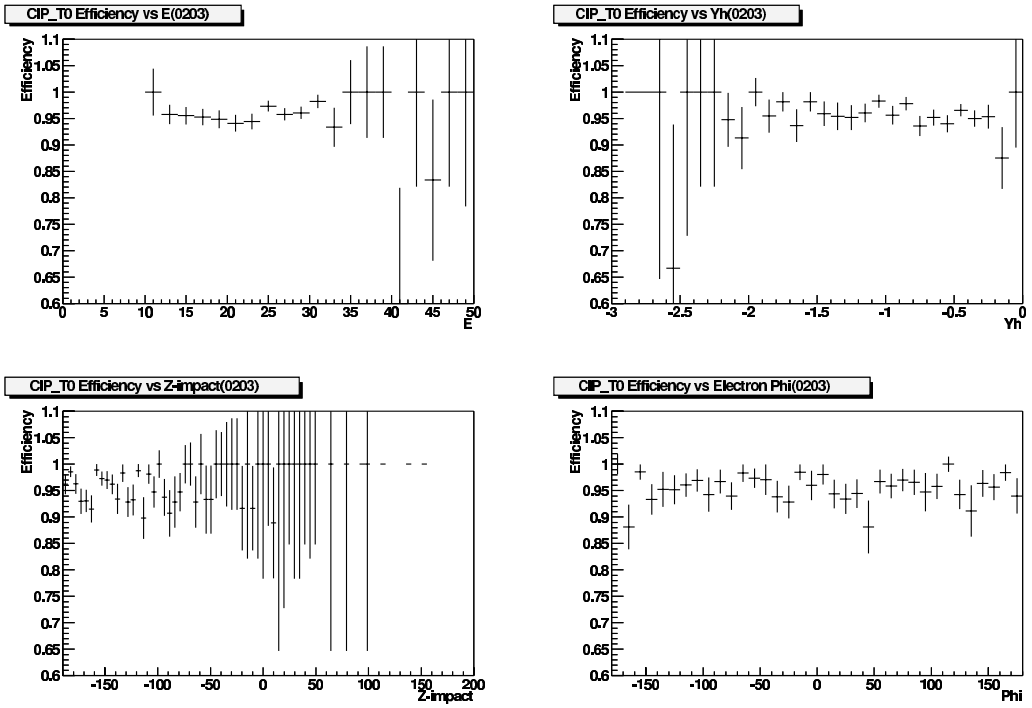


Fig. 10.13: Efficiency of the CIP_T0 trigger element measured in high P_t electron-events. The provided efficiency is of the order of 95% [104].

times more tracks in central region than in backward region).

Here the CIP2k trigger is used to find a good compromise between selecting good ρ-events with high efficiency and not selecting too many background events.

10.2.3 Integration into the L1 Trigger

The effect of the CIP2k trigger on all of the subtriggers presented here is considerable. It is used either as a veto trigger, rejecting background events, or as a positive trigger, contributing to the validation of ep events.

The CIP2k trigger elements have been integrated in the H1 L1 trigger strategy file, accessible as an additional global trigger in every subtrigger:

```
/* CIP2k background veto: */
  #GLOBAL r:0 !(CIP_mul==7 && CIP_sig==0)
/* CIP2k T0: */
  #GLOBAL t:0 CIP_T0
  #GLOBAL t:1 CIP_T0||LAr_T0
  #GLOBAL t:2 CIP_T0||DCRPh_T0_VETO_nextbc
  #GLOBAL t:3 CIP_T0||DCRPh_T0_VETO_nextbc||LAr_T0
  #GLOBAL t:5 CIP_T0||(LAr_T0&&!CIP_T0_nextbc)
```

The actual veto condition is:

```
CIP_VETO := !(CIP_mul==7 && CIP_sig==0)
```

The CIP2k trigger suggests to reject the event, if at least 100 track patterns are identified (CIP_mul==7) and more track are assigned to the backward region

(CIP_sig==0). For example in subtrigger s66, s67, s75 and s77, the veto condition (r:0) is used. In s75 and s77, the CIP_T0 is required as global trigger condition (t:0) in addition to the veto condition (r:0). In subtrigger s66 and s67 the global condition (t:5) is used, requiring a CIP_T0 or a LAr T0. The LAr T0 is only accepted, if the CIP_T0_nextbc (see Section 7.5.1) is not set, thereby improving the timing of the LAr T0. In s69, the CIP_T0 is used as a positive trigger.

Furthermore, the CIP2k trigger elements are used for many other subtriggers, e.g. for subtrigger s111 (see above).

The CIP2k background veto condition (r:0) is subject of a lot of studies of several physics working groups nowadays. Due to a flexible programmable trigger system, further improvements on the actual CIP2k trigger algorithm can be implemented easily.

Summary

Aim: This work describes the development and implementation of the new z-vertex trigger system for the H1 detector at HERA II. The main purpose of the trigger is to suppress a large fraction of background events originating from the backward part of the detector. After the luminosity upgrade of the HERA collider, the rate of these events has increased significantly. This emphasizes the need for an improved z-vertex trigger system.

Principle of operation: The trigger decision is derived by building a z-vertex histogram along the beam ($\hat{=}z$)-axis, making use of all tracks reconstructed in an event. Tracks with z-positions between $\approx$ -200 cm and -60 cm are sorted into the backward region of the histogram, tracks with a z-position of ± 60 cm around the interaction point are sorted into the central region. The number of tracks in both regions is analyzed. Typically, background events have more tracks in the backward region than in the central region. The important information of each trigger decision is transferred to the central trigger control of the H1 experiment, coded in a set of 16 trigger bits.

Techniques: The trigger system is implemented in large field programmable gate arrays (FPGAs), that hold the complete trigger algorithm. They are programmed in the hardware description language Verilog HDL. In the FPGAs, the trigger decision is derived in a pipelined process; each event is piped through every step of the algorithm. A trigger decision can be delivered for every bunch crossing, with a fixed latency of 10 bunch crossings (960 ns).

Two different types of printed circuit boards containing FPGAs were developed. They are called DL533 and DL535. DL535 is used for two different purposes.

1. The **trigger card** (DL533) contains two FPGAs (Altera APEX 20k400) holding the track finding and z-vertex histogram builder for one complete φ-sector of the chamber. In total, 16 trigger cards are used.

2. The purpose of the **sum cards** (DL535) is to add the z-vertex histograms from all 16 φ sectors and to calculate the trigger decision based on the summed histogram. Five identical PCBs are used to perform this task, four **pre sum cards** and one **master sum card**. Each sum card contains one FPGA, type Altera APEX 20k400.

3. The **komposti card** (DL535) is used to adapt the chamber signals from the new CIP2k chamber to the old z-vertex trigger system that is still used in addition to the new system.

Tests: The trigger was tested using a number of different diagnostic tools, like **functional simulations**, **timing simulations**, tests with a **pattern generator** and tests using the read out information (DAQ aided). In these tests, the trigger decision derived by the hardware (trigger system) was compared to the results of a software simulation.

The trigger decision of the CIP2k trigger system is found to be **100%** consistent with the simulation, proving that the development of the new z-vertex trigger has been completed successfully.

Results: To analyze the **performance** of the CIP2k trigger (recognition power, efficiency) minimum ionizing particles from cosmic rays were used that were accumulated during cosmics data taking runs (Summer 2003). A **good correlation** between tracks recognized in the drift chamber and CIP2k triggered tracks was observed. The measured CIP2k **trigger resolution of approximately 16.8 cm** matches the estimated expected resolution. The **single track efficiency (over φ) is of the order of 95%.**

First events taken in ep-collisions, provided by HERA since October 2003, were used to set up the CIP2k trigger. By analyzing the z-vertex distribution of selected events, an excellent rejection power of the CIP2k trigger system was established.

The CIP2k trigger performance was analyzed in more detail for selected physics channels. Diffractive ρ-events in DIS are triggered by the CIP2k with an efficiency of about 96%. Similar efficiencies are observed for high p_t electrons in neutral current DIS.

The CIP2k trigger is now used to reject background events in most H1 subtriggers. Furthermore a positive CIP2k trigger decision is used in most subtriggers to provide precise timing information.

The optimization of the CIP2k trigger decision algorithms (trigger elements) is the subject of ongoing studies. Due to the flexible and programmable trigger system, further improvements can be implemented easily. A stable and efficient **CIP2k z-vertex trigger decision** can be expected for the coming years.

Appendix A

VME Memory Map

PCB	IC	Address		used bits	name	meaning
		word	*hex*			
		1	0x2004	0-7	DAQ_settings	DAQ readout offset (25ns steps)
		1	0x2004	19-23	DAQ_settings	swap pad 100, 102
		1	0x2004	24-28	DAQ_settings	multiplexer order
		2	0x2008	0-31	version	version of FPGA code + date
		3	0x200C	0-30	zvtx_mapping	mapping of the z-vtx bins
		4	0x2010	0-31	scaler	nr. of pen since run start
	FPGA	5	0x2014	0-31	fstwd_err	nr. of fwd with wrong timing
TC	I,II	6-27	0x2018-0x206C	0-31	-	unused, not connected to FPGAs
		28-31	0x2070-0x207C	0-31	zvtx_readout	readout of z-vtx-histogram
		32-46	0x2080-0x20C8	0-31	force1	programmable padmask 1-bit
		47-61	0x20CC-0x2104	0-31	force0	programmable padmask 0-bit
		62	0x2108	12-14	strategy_mid	opsition of z-vtx histogram
		62	0x2108	0-4	strategy_bdr	borderpads per layer, -z side
		62	0x2108	5-9	strategy_bdr	borderpads per layer, +z side
		63	0x210C	0-4	zvtx_offset	z-vtx readout offset (BC steps)
		0	0x8000	0-4	-	JTAG chain, configure FPGAs
TC	ISpL	1	0x8004	0-31	VME_dly	delay of VME read cycle time
		3	0x800C	0-31	version	version number of ISPL code

Tab. A.1: The VME memory address map of the trigger card of the CIP2k z-vertex trigger.

PCB	IC	Address		used bits	name	meaning
		0-5	0x00-0x08	0-31	layerDelay	clock delay for 5 layers
		6	0x0A	0-31	globalDelay	global clock delay
		7	0x0C	0-31	ADC_diff	digital time difference (ADC)
CC	ISpL	8	0x0E	0-31	ADC_sync	digital time difference (ADC)
		10	0x20	0-31	cosmic_a	empty word cosmics trigger A
		11	0x22	0-31	cosmic_b	empty word cosmics trigger B

Tab. A.2: The VME memory address map of the control card of the CIP2k z-vertex trigger.

PCB	IC	Address		used bits	name	meaning
SC	FPGA	1	0x01	0-3	dir	demux direction (4 possibilities)
		2	0x02	0-31	version	version of FPGA code + date
		3	0x03	0-7	mode	operation mode of KC:
						bit 0: layer0 AND layer1 in φ_n
						bit 1: OR layer0 in φ_n
						bit 2: OR layer1 in φ_n
						bit 3: layer0 AND layer1 in φ_{n+1}
						bit 4: OR layer0 in φ_{n+1}
						bit 5: OR layer1 in φ_{n+1}
						bit 6: pulser, pulses each μs for 100ns
						bit 7: lauflicht ;-)
		5	0x05	0-31	scaler	nr. of pen since run start
		8	0x08	0-5	offset	DAQ readout offset
SC	ISpL	16	0x80	0-4	-	JTAG chain, configure FPGAs
		17	0x88	0-4	mux	
		18	0x8C	0-4	adc	readout of sum card adc (unused)
		19	0x90	0-4	n_config	signal 2 VME if FPGA configured
		30	0xFC	0-31	-	revision number of ISpL

Tab. A.3: The VME memory address map of the sum of the CIP2k z-vertex trigger.

PCB	IC	Address		used bits	name	meaning
SC	FPGA	0	0x00 - 0x3C	0-31	pipeline	readout pipeline of sum cards
		1	0x40	0	enable	enable trigger alg. (disabled)
		2	0x44	0-31	msg_chn	message_channel readout
		3	0x48	0-31	version	version of FPGA code + date
		6	0x54	0-31	scaler	nr. of pen since run start
		7	0x58	0-31	clockedge	fwd relation of 4 pre sum cards
		8-15	0x5C - 0x7F	0-31	-	reserved (addr 8-15)
SC	ISpL	16	0x80	0-4	-	JTAG chain, configure FPGAs
		17	0x88	0-4	mux	
		18	0x8C	0-4	adc	readout of sum card adc (unused)
		19	0x90	0-4	n_config	signal 2 VME if FPGA configured
		30	0xFC	0-31	-	revision number of ISpL

Tab. A.4: The VME memory address map of the komposti card of the CIP2k z-vertex trigger.

Appendix B

List of Files and Its Meaning

Below, a list of all circuit diagrams and layout diagrams of the developed PCBs (see Table B.1) and the a list of all HDL code (see Table B.2) is given.

File Name	meaning,used for...
Trigger Card	
TC_dl533_1.ps	Circuit diagram of Trigger Card (DL533), Sheet 1; power supply
TC_dl533_2.ps	Circuit diagram of Trigger Card (DL533), Sheet 2; configuration
TC_dl533_3.ps	Circuit diagram of Trigger Card (DL533), Sheet 3; CIP-data FPGA1
TC_dl533_4.ps	Circuit diagram of Trigger Card (DL533), Sheet 4; CIP-data FPGA2
TC_dl533_5.ps	Circuit diagram of Trigger Card (DL533), Sheet 5; VME bus
TC_dl533_6.ps	Circuit diagram of Trigger Card (DL533), Sheet 6; data to SC
TC_dl533_7.ps	Circuit diagram of Trigger Card (DL533), Sheet 7; LEDs, etc
TC_best1.ps	Layout diagram of Trigger Card (DL533)
Control Card	
CC_dl534_1.ps	Circuit diagram of Control Card (DL534), Sheet 1; clk, lemo I/O
CC_dl534_2.ps	Circuit diagram of Control Card (DL534), Sheet 2; ISpL, VME bus
CC_dl534_3.ps	Circuit diagram of Control Card (DL534), Sheet 3; LEDs, etc
CC_dl534A_1.ps	Circuit diagram of Control Card (DL534), Sheet A1; piggy back
CC_dl534B_2.ps	Circuit diagram of Control Card (DL534), Sheet A2; piggy back
CC_best1.ps	Layout diagram of Control Card (DL534)
Sum/Komposti Card	
SC_dl535_1.ps	Circuit diagram of Sum + Komposti Card (DL535), Sheet 1; clk
SC_dl535_2.ps	Circuit diagram of Sum + Komposti Card (DL535), Sheet 2; config.
SC_dl535_3.ps	Circuit diagram of Sum + Komposti Card (DL535), Sheet 3; TC data
SC_dl535_4.ps	Circuit diagram of Sum + Komposti Card (DL535), Sheet 4; TC data
SC_dl535_5.ps	Circuit diagram of Sum + Komposti Card (DL535), Sheet 5; VME bus
SC_dl535A_1.ps	Circuit diagram of Sum + Komposti Card (DL535), Sheet A1; piggy
SC_dl535A_2.ps	Circuit diagram of Sum + Komposti Card (DL535), Sheet A2; piggy
SC_best1.ps	Layout diagram of Sum + Komposti Card (DL535)

Tab. B.1: Circuit diagrams and layout diagrams of the developed PCBs of the CIP2k z-vertex trigger system. The files are stored at [85].

File Name	meaning,used for...
Trigger Card	
CIP2k_TC_FPGA1.bdf	Top level design file of FPGA I
CIP2k_TC_FPGA2.bdf	Top level design file of FPGA II
readout_module.bdf*†	Readout module design file of FPGA I
Trigger_module.bdf*	Trigger module design file of FPGA I
Trigger_system.bdf*	Trigger system design file of FPGA I
synchroniser.bdf†	Synchronizes FPGA I and FPGA II
TE_builder.bdf†	Trigger element design file FPGA II, SC
adder.v*	Track adder of FPGA I + II
demultiplex.v*†	Demultiplexes chamber pads of FPGA I + II
filter.v	Filter chamber data of FPGA I + II
state_machine.v*†	State machine of FPGA I + II
trigger.v*	Track finding of FPGA I + II
trigger_ctrl.v*†	Trigger control of FPGA I + II
memory_module.v	Stores chamber information of FPGA I + II
element_builder.v	Trigger element generation in FPGA II
main_adder.v	Adds histograms of both FPGAs in FPGA II
sync_FPGA1.v	Synchronizes FPGA I and FPGA II
sync_FPGA2.v	Synchronizes FPGA I and FPGA II
CIP2k_TC_FPGA1.csf	Pin assignment file of FPGA I
CIP2k_TC_FPGA2.csf	Pin assignment file of FPGA II
CIP2k_TC_FPGA1.pof	EPC2 configuration file 1 of FPGA I
CIP2k_TC_FPGA1_1.pof	EPC2 configuration file 2 of FPGA I
CIP2k_TC_FPGA1_2.pof	EPC2 configuration file 3 of FPGA I
CIP2k_TC_FPGA1.sof	FPGA configuration file of FPGA I
CIP2k_TC_FPGA2.pof	EPC2 configuration file 1 of FPGA II
CIP2k_TC_FPGA2_1.pof	EPC2 configuration file 2 of FPGA II
CIP2k_TC_FPGA2_2.pof	EPC2 configuration file 3 of FPGA II
CIP2k_TC_FPGA2.sof	FPGA configuration file of FPGA II
Sum Card	
CIP2k_SC.bdf	Top level design file of sum card
adder_module.bdf	Adding of histos + building TEs in sum card
histogram_sheet.bdf	Adding of histograms in sum card (submodule)
synchronizer.v	Synchronizes inputs of TCs or pre SCs
mux_adder.v	Adding of histograms in sum cards
decision.v	Building trigger decision in sum card
Komposti Card	
CIP2k_KC_20ke.bdf	Top level design file of komposti card
logic_module.bdf	Top level design file of komposti card
VME_module.bdf	Top level design file of komposti card
merger.v	merging pads from new CIP to old trigger
VME.v	State machine of komposti card

Tab. B.2: A list of the developed block file diagrams and Verilog HDL codes is given. A detailed description of the files can be found in Chapter 8. The files are stored at [97]. Note: (*)-marked files have the same name in trigger card FPGA I, and FPGA II. (†)-marked files have the same name in the trigger card, sum card and/or komposti card design. Please check files in the corresponding directories.

Appendix C

The Projective Chamber

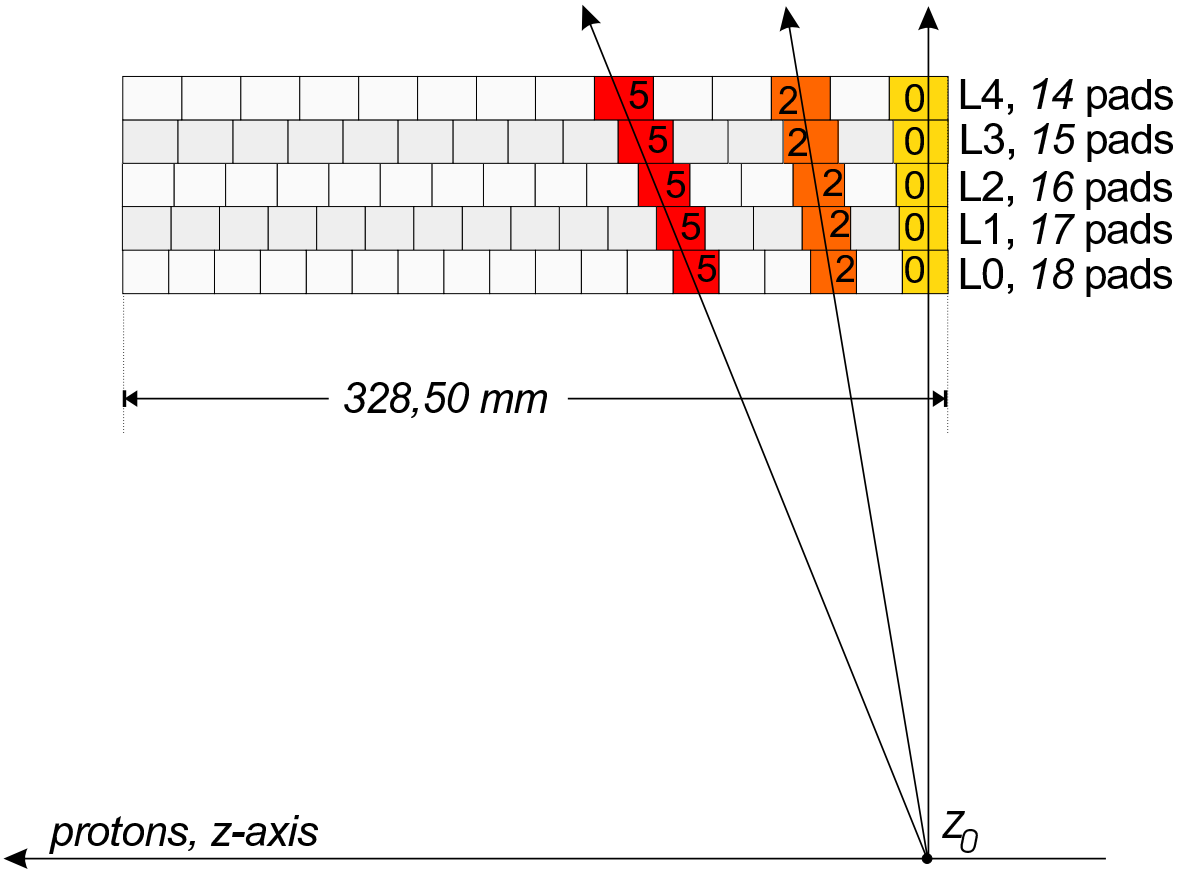


Fig. C.1: Standard block of the *projective geometry*. Every 328,50 mm the pads of the five layers are exactly aligned above each other [65].

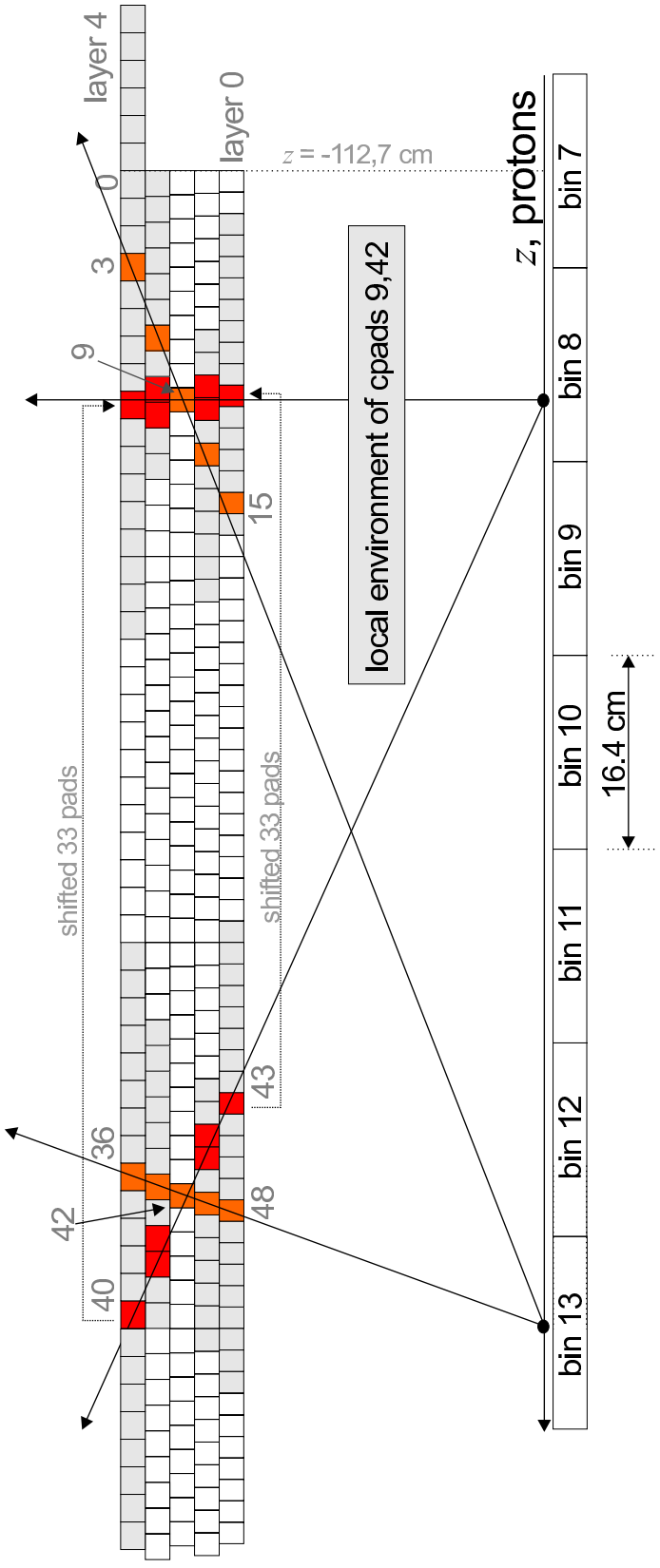


Fig. C.2: On the left side, the local environment of cpad 42 is shown. It is the same as in Figure 7.2 on page 74. On the right, the LE of cpad 9 is shown. Exactly the same track pattern in the LE is used to identify tracks from bin 8 and 13, shifted by 33 pads to the right.

Bibliography

[1] The ZEUS Collaboration, see http://www-zeus.desy.de/.

[2] The Hera-B Collaboration, see http://www-hera-b.desy.de/.

[3] The Hermes Collaboration, see http://www-hermes.desy.de/.

[4] C. Adloff *et al.* [H1 Collaboration], Nucl. Phys. B **497** (1997) 3, [arXiv:hep-ex/9703012].

[5] C. Adloff *et al.* [H1 Collaboration], Nucl. Phys. B **545** (1999) 21 [arXiv:hep-ex/9812023].

[6] S. Aid *et al.* [H1 Collaboration], Nucl. Phys. B **449** (1995) 3 [arXiv:hep-ex/9505014].

[7] S. Aid *et al.* [H1 Collaboration], Phys. Lett. B **354** (1995) 494 [arXiv:hep-ex/9506001].

[8] C. Adloff *et al.* [H1 Collaboration], Eur. Phys. J. C **13** (2000) 397 [arXiv:hep-ex/9812024].

[9] C. Adloff *et al.* [H1 Collaboration], Phys. Lett. B **523** (2001) 234 [arXiv:hep-ex/0107038].

[10] C. Adloff *et al.* [H1 Collaboration], Phys. Lett. B **479** (2000) 358 [arXiv:hep-ex/0003002].

[11] H1 Collaboration: Luminosity Summary plots, see: http://www-h1.desy.de/h1det/lumi/lplots/

[12] H1 Collaboration: "The Luminosity upgrade 2000-2002", see: http://www-h1.desy.de/h1det/.

[13] H1 Collaboration: "*ep* Physics beyond 1999", H1 internal note H1-IN-531 (10/1997)

[14] I. Abt *et al.* [H1 Collaboration], "The H1 detector at HERA," Nucl. Instrum. Meth. A **386** (1997) 310.

[15] The H1 Collaboration, see http://www-h1.desy.de/

[16] V. Andreev *et al.* [H1 background working group], "Technical Report on the Beam Induced Backgrounds in the H1 Detector", H1 note H1-IN-606 (10/2002).

[17] D. Pitzl *et al.*, "The H1 silicon vertex detector," Nucl. Instrum. Meth. A **454** (2000) 334 [arXiv:hep-ex/0002044].

[18] J. Gassner, "A Measurement of D-Meson Production at HERA by Decay Vertex Identification", Dissertation Nr. 14774, Appendix F, ETH Zuerich, 2002 (ETHZ-IPP Internal Report 2002-45).

[19] W. Eick *et al.*, Nucl. Instrum. Meth. A **386**, (1997) 81.

[20] W. Braunschweig *et al.*, "A Forward Silicon Tracker for H1", DESY-PRC-99/01, H1 note H1-IN-563 (02/1999).

[21] K. Mueller *et al.*, Nucl. Instr. Meth. A **312** (1992) 457.

[22] M. Cuje *et al.*, "H1 High Luminosity Upgrade 2000 CIP and Level 1 vertex Trigger", H1 internal report H1-IN-535 (01/1998)

[23] F. Sauli, "Principles of MWPC and Drift Chambers", CERN 77-03, in T.Ferbel, "Experimental Techniques in High Energy Physics".

[24] M. Urban et al., "The Cip2k First-Level Trigger System At The H1 Experiment At Hera," IEEE Trans. Nucl. Sci. **50** (2003) 903.

[25] The H1 central Jet Chamber (CJC), see: http://www-h1.desy.de/h1det/tracker/cjc/poster.html.

[26] I. Abt *et al.* [H1 Collaboration], Nucl. Instrum. Meth. A **386** (1997) 348.

[27] Th. Wolff *et al.*, Nucl. Instrum. Meth. A **323** (1992) 537.

[28] A. Baird *et al.* [H1 Collaboration], Nucl. Instrum. Meth. A **461** (2001) 461.

[29] Y. H. Fleming, "The H1 first level fast track trigger," DESY-THESIS-2003-045.

[30] B. Andrieu *et al.* [H1 Calorimeter Group Collaboration], Nucl. Instrum. Meth. A **336** (1993) 460.

[31] B. Andrieu *et al.* [H1 Calorimeter Group Collaboration], Nucl. Instrum. Meth. A **350** (1994) 57.

[32] B. Andrieu *et al.* [H1 Calorimeter Group Collaboration], Nucl. Instrum. Meth. A **336** (1993) 499.

[33] R. D. Appuhn *et al.* [H1 SPACAL Group Collaboration], Nucl. Instrum. Meth. A **386** (1997) 397.

[34] T. Nicholls *et al.* [H1 SPACAL Group Collaboration], Nucl. Instrum. Meth. A **374** (1996) 149.

[35] D. Reyna, "Modifications to SpaCal for H1 High Luminosity Operation and the Effect on Acceptance", H1 internal note H1-11/98-555.

[36] P. Biddulph *et al.*, Nucl. Instrum. Meth. A **340** (1994) 304.

[37] T. Ahmed *et al.*, Nucl. Instrum. Meth. A **364** (1995) 456.

[38] H1 Collaboration, Contributed paper to the 28th International Conference High Energy Physics, Warsaw, Poland, Paper pa17-026, 1996.

[39] H.Bethe, W.Heitler, Proc. Roy. Soc. **A146** (1934), 83

[40] B. Andrieu *et al.*, in "Proceedings of the 9th Conference on Calorimetry in High Energy Physics" (CALOR 2000)", Annecy, France, 9-14 Oct 2000, Frascati Physics Series, Vol. 21.

[41] V. Andreev *et al.*, "Further Report on the Beam-Induced Backgrounds in the H1 Detector", H1 internal note H1-IN-607 (01/2003).

[42] E. Elsen, "Aspects Of The H1 Trigger And Data Acquisition System", Prepared for 2nd Annual Conference on Electronics for Future Colliders, Chestnut Ridge, N.Y., 19-21 May 1992.

[43] H. C. Schultz-Coulon, E. Elsen, T. Nicholls, J. Coughlan and H. Rick, IEEE Trans. Nucl. Sci. **46** (1999) 915.

[44] F. Sefkow, E. Elsen, H. Krehbiel, U. Straumann and J. Coughlan, IEEE Trans. Nucl. Sci. **42** (1995) 900.

[45] T. Nicholls *et al.*, IEEE Trans. Nucl. Sci. **45** (1998) 810.

[46] J. K. Kohne *et al.*, Nucl. Instrum. Meth. A **389** (1997) 128.

[47] J.C. Bizot *et al.*, "Strategy Studies for the H1 Topological L2-Trigger(L2TT)", H1 internal note H1-IN-508 (01/1997).

[48] J. Naumann, "Development and Testing of the Third Level Trigger for H1", Dissertation, Univ. Dortmund, 2003.

[49] Ch. Wissing, "Entwicklung eines Simulationsprogramms und Implementierung schneller Spurfitalgorithmen für den neuen H1-Driftkammertrigger", Dissertation, Univ. Dortmund, 2003

[50] A. Campbell *et al.*, "The Fourth Level Trigger Scheme of the H1 Collaboration at HERA; Practical Performance and Future Prospects", at CHEP 1998, Chicago, note: only transparencies available; see http://www-h1.desy.de/trigger/publications.html.

[51] H1 Trigger Home Page, list of actual H1 subtrigger at L1, see: http://www-h1.desy.de/h1/iww/itrigger/TrigSetup/tdl.subtriggers.

[52] E. Elsen, "The H1 Trigger and Data Acquisition System", H1 internal note H1-01/93-262.

[53] S. Eichenberger, "A fast pipelined trigger for the H1 Experiment at HERA Based on Multiwire Proportional Chamber Signals", Dissertation, Univ. Zürich, 1993

[54] H1 Collaboration: "The H1 Detector at HERA", H1 internal report DESY-A1-96-01, Pages 19-30: The Trigger. see http://www-h1.desy.de/h1/www/h1det/detpaper/p1s5.ps

[55] H. Beck, "Principles and Operation of the z-Vertex Trigger", H1 internal note H1-05/96-479.

[56] D. Baumeister *et al.*, "Progress Report on CIP and Level 1 Vertex Trigger", for the DESY PRC, 1998.

[57] A. Schweizer, "Optimierung eines z-Vertex-Triggers für den H1-Detektor bei HERA", Diploma Thesis, Univ. Heidelberg, 1999.

[58] M. Cuje *et al.*: "H1 High Luminosity Upgrade 2000 CIP and Level 1 vertex Trigger", page 10: "Improving the Trigger", H1 internal report H1-IN-535(01/1998).

[59] H. Geiger, E. Rutherford, "Prototype of the Geiger-Zaehler", Proc. Roy. Soc. **A82**, 1909, 495.

[60] G. Charpac *et al.*, Nucl. Instr. Meth. **62**, 262(1968); **80**, 13(1970).

[61] M. Kollak, "Entwicklung einer Stripline-Auslese für Kathodensignale einer langen, zylindrischen Vieldrahtproportionalkammer", Diploma thesis, Univ. Heidelberg, 1998.

[62] S .Steiner, Development of the new CIP2k chamber, technical drawing of the chamber for this thesis, University of Zürich, 2003.

[63] The Mechanics Workshop of the Physics Institute of the University of Zurich, see: http://www.physik.unizh.ch/groups/werkstatt/.

[64] K. Bösiger, P. Robmann, S. Steiner, "H1 CIP-Upgrade: 5-lagige Proportionalkammer", BRS Verlag, Zürich, 1999.

[65] U. Straumann, private communications about the projective geometry, 2000, 2001, 2003.

[66] A. Vollhardt, "Entwurf und bau einer Frontend-Steuerung für das CIP-Upgrade Projekt für H1 bei HERA", Diploma thesis, Univ. Heidelberg, 2001.

[67] M. Urban, "Ein schneller Trigger Für H1 bei HERA", Diploma thesis, Univ. Heidelberg, 2000.

[68] N. Werner, "Measurement of the Charged Current Cross Section in Positron–Proton Collisions at HERA", Appendix on the CIP2k chamber gas and high voltage systems, Dissertation, Univ. Zuerich, 2004.

[69] CAEN S.p.A. company, italy, see: http://www.caen.it/nuclear/index.php.

[70] Paul Scherrer Institute, Switzerland, see: http://www.psi.ch.

[71] ASIC Laboratory Heidelberg, "The H1 CIP read-out: CIPix", see: http://wwwasic.kip.uni-heidelberg.de/Projects/finished.html.

[72] M. Ziegler, P. Sievers and U. Straumann, Nucl. Instrum. Meth. A **471** (2000) 260.

[73] ETM AG, "PVSS II", process visualization and control system, see: http://www.pvss.com/english/forschung.htm.

[74] D. Baumeister, "Entwicklung und Charakterisierung eines ASICs zur Kathodenauslese von MWPCs für das H1-Experiment bei HERA", Diploma thesis, Univ. Heidelberg, 1999.

[75] S. Loechner, "Charakterisierung und Entwicklung eines CIP-Auslese-ASIC für das H1-Upgrade-Projekt 2000", Diploma thesis, Univ. Heidelberg, 2001.

[76] S. Luders *et al.*, Nucl. Instrum. Meth. A **484** (2002) 515, [arXiv:hep-ex/0107064].

[77] Creative Electronic Systems, documentation of the RIO 8062 CPU and pVIC, see: http://www.ces.ch/documents/documents.html.

[78] W-IE-NE-R Plein & Baus GmbH, VME crates, see: http://www.wiener-d.com/fr-cr.htm.

[79] Electronics Workshop of the Faculty of Physics and Astronomy of the University of Heidelberg, see: http://pi1.physi.uni-heidelberg.de/physi/ew/Entwicklung.php.

[80] Altera Corporation, "Description of Altera APEX 20k device family", see: http://www.altera.com/literature/ds/apex.pdf.

[81] Lattice Semiconductor Corporation, see: http://www.latticesemi.com/lit/docs/datasheets/cpld/1048e.pdf and http://www.latticesemi.com/products/cpld/index.cfm.

[82] Motorola Semiconductor Products Inc., "VMEbus Specification Manual", Revision C, 1985

[83] "Documentation of the CIP2k trigger and DAQ system", firmware codes for Lattice ISpL CPLD, see: http://www.physik.unizh.ch/groups/grouptruoel/cipupgrade/cipsources.html

[84] Altera Corporation, "Quartus Hardware Design Software", see:
http://www.altera.com/support/software/quartus2/design_flow/des-index.html
and http://www.altera.com/literature/manual/intro_to_quartus2.pdf

[85] "Documentation of the CIP2k trigger and DAQ system", circuit diagrams for trigger card (DL533), control card (DL534), sum card (DL535) and backplane (AS16), see:
http://www.physik.unizh.ch/groups/grouptruoel/cipupgrade/cipsources.html

[86] Altera Corporation, Jam Player, see:
http://www.altera.com/literature/an/an122.pdf

[87] Jam-player binary files for the CIP2k Trigger System, located at
(sftp) enzian.desy.de/ c̃ip/jam_files/

[88] H. Klehr, "CIP2000 User Manual", see:
http://www.desy.de/ h1cip/documents/CIP2000_expertManual.ps

[89] The CASCADE neutron detector, see:
http://pi1.physi.uni-heidelberg.de/physi/cascade/index.php.html

[90] Phillips Semiconductor, "The I^2C-Bus Specification", Version 2.1, 2000, see:
http://www.semiconductors.philips.com/acrobat/literature/9398/39340011.pdf.

[91] Lynuxworks, "Embedded operating systems", see:
http://www.lynuxworks.com/.

[92] VMIC, Fanuc Company, "VME bus expander", see: http://www.vmic.com.

[93] J. Becker *et al.*, Proc. 8th Topical Seminar on Innovative Particle and Radiation Detectors, Siena, 2002.

[94] S. Schmitt, private communications about the CIP2k Data Acquisition System, 2002, 2003.

[95] Altera Corporation, Description of Altera APEX 20k device family, Functional Description, page 9 et seq., see: http://www.altera.com/literature/ds/apex.pdf.

[96] Altera Corporation, "APEX 20K Devices: System-on-a-Programmable-Chip Solutions", see: http://www.altera.com/products/devices/apex/apx-index.html.

[97] "Documentation of the CIP2k trigger and DAQ system", firmware and simulation files for trigger card, sum card and komposti card, see:
http://www.physik.unizh.ch/groups/grouptruoel/cipupgrade/cipsources.html

[98] Tektronix Inc., DG2020A Data Generator, see:
http://www.tek.com/site/ps/76-10799/pdfs/76W_10799.pdf.

[99] "Documentation of the CIP2k trigger and DAQ system", list of talks given about the CIP2k trigger system, see:
http://www.physik.unizh.ch/groups/grouptruoel/cipupgrade/cip.html

[100] S. Schmitt, private communications about the Cosmics Run Data Analysis, 2003.

[101] "The H1 Online L4/5 Histogram Display (Zubr)", see manual at: https://www-h1.desy.de/idet/ishift/zubr.html.

[102] F.-P. Schilling, "Trigger setup: CIP background veto applied", H1-Hypernews, Trigger Forum, 24. October 2003 and "Trigger Setup Overview" at the H1 Trigger Meeting, 17th November 2003.

[103] L. Favart, "Diffractive L1/2 Triggers" at the H1 Trigger Meeting, 17th December 2003.

[104] O. Henshaw, "ELAN hiQ2 Trigger Efficiencies" at the H1 Trigger Meeting, 17th November 2003 and private communications.

Curriculum Vitae

Personalien:

Name:	Urban
Vorname:	Max Christoph
Geboren:	5. Januar 1975 in Würzburg, Deutschland
Staatsangehörigkeit:	deutsch

Bildungsgang:

1981-1985	Grundschule in Fröndenberg-Dellwig, Deutschland
1985-1994	Pestalozzi Gymnasium in Unna, Deutschland
1994	**Abitur**
1994-2000	Studium der Physik an der Ruprecht-Karls-Universität, Heidelberg
Mai 1996	**Vordiplom in Physik**
1998-2000	Wissenschaftliche Mitarbeit an der Wirtschaftswissenschaftlichen Fakultät der Universität Heidelberg im Fachbereich Wirtschaftsinformatik, (Prof. Dr. R. Fahrion)
1999-2000	Diplomarbeit in der experimentellen Hochenergiephysik unter Leitung von Prof. Dr. U. Straumann, Titel: *"Ein schneller Trigger für H1 bei HERA"*
Juni 2000	**Diplom in Physik** (Nebenfach: Volkswirtschaftslehre, Wahlfach: Elektronik)
2000-2004	Wissenschaftliche Mitarbeit am Physik-Institut der Universität Zürich **Promotion** unter Leitung von Prof. Dr. U. Straumann,
2001-2004	Forschungsaufenthalt am *Deutschen Elektronen-Synchrotron* (DESY) in Hamburg am H1-Experiment
2004	**Dissertation**, Titel: *"The new CIP2k z-Vertex Trigger for the H1 Experiment at HERA"*