FACULTY OF PHYSICS AND ASTRONOMY
RUPERTUS CAROLA UNIVERSITY OF HEIDELBERG

Diploma Thesis

in Physics

submitted by

Jan Becker

born in Hamburg (Germany)

2000

The Data Aquisition and Control System for a Fast Trigger at H1

This diploma thesis has been carried out by Jan Becker at the

Institute of Physics

under the supervision of

Prof. Dr. Franz Eisele

The Data Aquisition and Control System for a Fast Trigger at H1

The HERA (**H**adron-**E**lektron-**R**ing-**A**nlage) particle collider at DESY in Hamburg (Germany) is operating since May 1992. An upgrade for HERA is planned for 2000/2001 to gain higher luminosity. A bigger number of background events is therefore also expected. A new fast trigger called CIP2000 trigger is developed to examine and suppress the background events. The trigger system uses the pad information of five cylindrical multiwire chambers to calculate a histogramm of track origins along the beam axis. From the structure of this histogram background events can be identified. The aim of this diploma thesis is to develop and test the control system and the readout system for the new trigger. The communication between the different CPUs (pVIC) and the communication between the CPUs and the newly designed trigger cards (VME) are tested in a test system. Different data reduction algorithms to reduce the size of the read out events are simulated. The connection and communication of the STC (**S**ubsystem **T**rigger **C**ontrol) system and the CIP2000 trigger system is defined and some parts of the communication are tested. The control system for the whole CIP2000 trigger system is specified.

Das Datennahme- und Kontrollsystem für einen schnellen Trigger bei H1

Der HERA (**H**adron-**E**lektron-**R**ing-**A**nlage) Elektron-Proton-Speicherring am DESY in Hamburg (Deutschland) ist seit Mai 1992 in Betrieb. Eine Erhöhung der Luminosität von HERA ist für die Jahre 2000/2001 geplant. Deshalb wird auch eine größere Anzahl von Untergrundereignissen erwartet. Ein neuer schneller Trigger (CIP2000 Trigger) wurde entwickelt, um die Untergrundereignisse zu untersuchen und zu unterdrücken. Das Triggersystem benutzt die Kammerinformationen von fünf zylindrischen Kammern, um ein Histogramm der rekonstruierten Spurursprünge entlang der Strahlachse zu berechnen. An der Struktur dieses Histogramms können Untergrundereignisse erkannt werden. Das Ziel dieser Diplomarbeit ist die Entwicklung und der Test eines Kontroll- und Datennahmesystems für den neuen Trigger. Die Kommunikation der verschiedenen CPUs (pVIC) und die Kommunikation zwischen den CPUs und den neu entwickelten Triggerkarten (VME) wurde an Hand eines Testsystems untersucht. Verschiedene Datenreduktionsalgorithmen wurden simuliert, um die ausgelesene Datenmenge zu verkleinern. Die Verbindung des STC-Systems (**S**ubsystem **T**rigger **C**ontrol) mit dem CIP2000 Triggersystem wurde definiert und Teile dieser Verbindung getestet. Ein Kontrollsystem für das gesamte CIP2000 Triggersystem wurde entwickelt.

Contents

1 Introduction 1

2 HERA and the H1 detector 3

2.1 The HERA machine . . . 3

2.2 The H1 detector . . . 4

2.2.1 H1 trigger system . . . 5

2.2.2 H1 readout system . . . 7

2.3 The H1 upgrade for HERA 2000 . . . 8

3 The CIP2000 trigger system 10

3.1 Motivation . . . 10

3.2 Trigger algorithm . . . 11

3.3 Projective geometry . . . 13

3.4 Trigger system implementation . . . 14

3.4.1 Chamber geometry . . . 15

3.4.2 Chamber readout . . . 17

3.4.3 Trigger system hardware . . . 18

3.4.4 Timing . . . 24

4 The CIP2000 trigger control system 25

4.1 CIP2000 trigger system control structure . . . 25

4.2 STC system . . . 25

4.2.1 Trigger signals . . . 28

4.2.2 Trigger signal timing . . . 30

4.2.3 STC cards . . . 33

4.3 CIP2000 trigger STC and readout control system . . . 36

5 The CIP2000 trigger readout system 38

5.1 Readout system structure 38

5.2 Readout system timing 40

5.3 Readout system hardware 42

5.3.1 Connection of CPU and trigger card: VME 42

5.3.2 Trigger card 44

5.3.3 CPU board 45

5.3.4 Connection of CPUs: pVIC 47

5.3.5 Connection of main CPU and STC system: VIC 48

5.3.6 Connection of main CPU and VMEtaxi card: VSB, VIC 50

5.4 Readout system software 50

6 Tests with the CIP2000 trigger readout test system 52

6.1 Test of the VME bus 52

6.2 Test of the pVIC bus 55

7 Data reduction for the CIP2000 trigger readout 58

7.1 Raw data of the CIP2000 trigger 58

7.2 Data reduction algorithms 60

7.2.1 Zero supression 60

7.2.2 Run length encoding (RLE) 60

7.3 Simulation of data reduction algorithms 61

7.3.1 Activated pads 61

7.3.2 Data reduction algorithms 63

7.3.3 Conclusion 67

A Dictionary of abbreviations 70

Bibliography 72

Acknowledgements 74

List of Figures

2.1 The HERA storage ring 3
2.2 Overview of the H1 detector 4
2.3 H1 trigger levels 5
2.4 Principle of pipelining in H1 subsystems 6
2.5 Principle of the H1 detector readout 7

3.1 Definition of a local environment 12
3.2 Overview of the CIP2000 trigger algorithm 12
3.3 Projective geometry 13
3.4 Overview of the CIP2000 trigger system 14
3.5 Side view of the CIP2000 chamber 15
3.6 Front view of the CIP2000 chamber 16
3.7 Signal path of the CIP2000 chamber readout 17
3.8 Overview of the CIP2000 chamber readout for 4 sectors in ϕ 18
3.9 CIP2000 trigger system structure 19
3.10 CIP2000 trigger card 21
3.11 CIP2000 sum card 22
3.12 CIP2000 control card 23
3.13 Timing of the CIP2000 trigger 24

4.1 CIP2000 trigger system control structure 26
4.2 Structure of a standard STC system 27
4.3 Trigger signal timing – L1 Keep, L2 Reject 31
4.4 Trigger signal timing – L1 Keep, L2 Keep, L3 Reject 32
4.5 Trigger signal timing – L1 Keep, L2 Keep, L3 Keep 33

5.1 CIP2000 trigger readout structure 39
5.2 CIP2000 trigger readout timing 40

5.3 Structure of VME functional modules and buses 42
5.4 Address mapping to the VME resources on the trigger cards 44
5.5 Structure of the RIO2 8062 PC board 45
5.6 VIC 8250 module structure 48

6.1 Signal timing of a VME bus single cycle data transfer 52
6.2 Signal timing of a VME bus block data transfer 53
6.3 Test system to examine the VME bus performance 53
6.4 Test system to examine the pVIC connection performance 56
6.5 Transfer rate of the pVIC connection in the DMA mode 57

7.1 Mapping of the chamber data on the CPU memory 59
7.2 Data reduction algorithms 60
7.3 Activated pads: different event types, no noise 62
7.4 Activated pads: noise dependence for mean value of activated pads 63
7.5 Reduction ratio for different algorithms on high q^2 events, no noise 64
7.6 Reduction ratio for different algorithms on low p_T events, no noise 64
7.7 Reduction ratio for different algorithms on background events, no noise . . 65
7.8 Reduction ratio for algorithm 2 on different event types, no noise 65
7.9 Reduction ratio for algorithm 4 on different event types, no noise 66
7.10 Noise dependence of the reduction ratio for algorithm 2 on different event types 66
7.11 Noise dependence of the reduction ratio for algorithm 4 on different event types 67
7.12 Models for event data in the pipeline 68

List of Tables

3.1 Pad geometry of the new CIP2000 chamber 16

4.1 Trigger signals 28

5.1 Assumed duration of the horizontal and vertical readout 41

6.1 Measured timing parameters of VME bus data transfers: single cycle mode 54
6.2 Measured timing parameters of VME bus data transfers: block mode . . . 54
6.3 Measured VME bus data transfer performance 55
6.4 Transfer rate of the pVIC connection in the DMA mode 57

7.1 Run length encoding (RLE) algorithms 61
7.2 Mean value and width of distribution of activated pads: different event types, no noise 62
7.3 Reduction ratio and width of the distribution for different algorithms on different event types, no noise 63
7.4 Reduction ratio for different models of event distribution in the pipeline for algorithm 2 68
7.5 Reduction ratio for different models of event distribution in the pipeline for algorithm 4 69

Chapter 1

Introduction

The HERA (**H**adron-**E**lektron-**R**ing-**A**nlage) particle collider at DESY in Hamburg (Germany) is operating since May 1992. An upgrade for HERA is planned for 2000/2001. In the upgrade the size of the focus of the beams is reduced leading to a higher luminosity. The detectors have to be adapted to the new situation. A bigger number of background events is expected. A new fast trigger is developed to examine and suppress the background events.

The new trigger system is called CIP2000 trigger system. The trigger system uses the pad information of five cylindrical chambers to find and count tracks originating from the beam axis. The number of tracks originating from different parts of the beam axis are summed up to a z-vertex histogram. The structure of the z-vertex histogram makes it possible to detect background events. The trigger system uses FPGAs (**F**ield **P**rogrammable **G**ate **A**rray) mounted on trigger cards to implement the trigger algorithm. The chamber data is stored in pipelines situated in the FPGAs. CPUs read out the data via a VME bus and a pVIC connection.

The aim of this diploma thesis is to develop the control system and the readout system for the new trigger. The communication of the different CPUs and the new designed trigger cards is tested in a test system. The pVIC bus between the different CPUs is examined. Different data reduction algorithms to reduce the size of the events read out from the pipelines are simulated. The communication of the different parts of the whole CIP2000 trigger system is designed. The connection and communication of the STC (**S**ubsystem **T**rigger **C**ontrol) system and the CIP2000 trigger system is specified and some parts of the communication are tested.

- **Chapter 2** gives an introduction to the HERA particle collider, the H1 detector and the upgrade for HERA 2000.
- **Chapter 3** gives an overview of the CIP2000 trigger system and the hardware components used in the system.
- **Chapter 4** explains the structure of the trigger control system and the system to control the whole CIP2000 system.

- **Chapter 5** gives an overview of the planned readout system of the CIP2000 trigger and its hardware components.
- **Chapter 6** gives an overview of tests made with the readout test system.
- **Chapter 7** shows the results of simulations of the data reduction on the raw data of the CIP2000 trigger readout system.

Chapter 2

HERA and the H1 detector

2.1 The HERA machine

Two independent machines reside in the 6.3 km long tunnel some 15 m below the surface. One is an electron (alternatively positron) storage ring, which accelerates the particles from an initial 14 GeV to 30 GeV. The proton machine starts with initial 40 GeV accelerating the protons to 920 GeV before a collision with the electrons or a target is forced. The energy of the proton beam is limited by the strength of the magnets needed to bend the beam on its orbit. The HERA proton ring consists of superconducting magnets which produce a field of 4.6 T. Figure 2.1 gives an overview of HERA and its injector chains. The particles are packed into a maximum of 210 bunches, with a bunch crossing distance of only 96 ns. Not all of the bunches are filled with particles. The bunch crossings with no interactions are used to determine the background situation. The currents of the beams have been successfully increased in the last years. At the moment four experiments use the HERA storage rings. H1 and ZEUS are built around two interaction regions of the storage rings. The HERMES detector records the scattering of the polarized electron

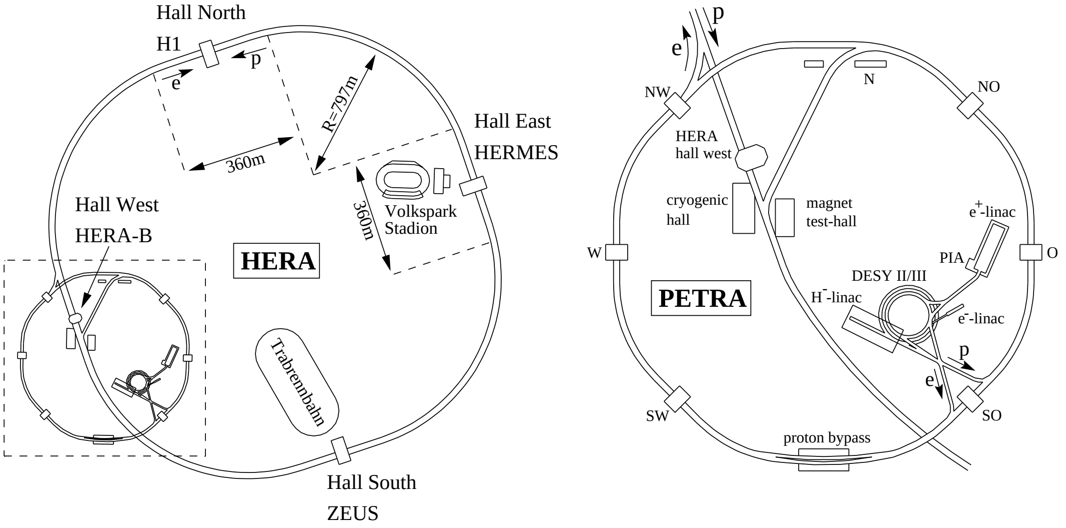


Figure 2.1: *The HERA storage ring with its pre-accelerators PETRA, DESY and Linacs*

beam on polarized gas targets. The HERA-B experiment is built to measure *CP* violation in $B^0 - \bar{B}^0$ systems generated by collisions of beam protons with a wire target.

2.2 The H1 detector

The H1 detector, located at the north interaction region of HERA is shown in Figure 2.2 also including the standard coordinate system. The detector is described in detail in [1].

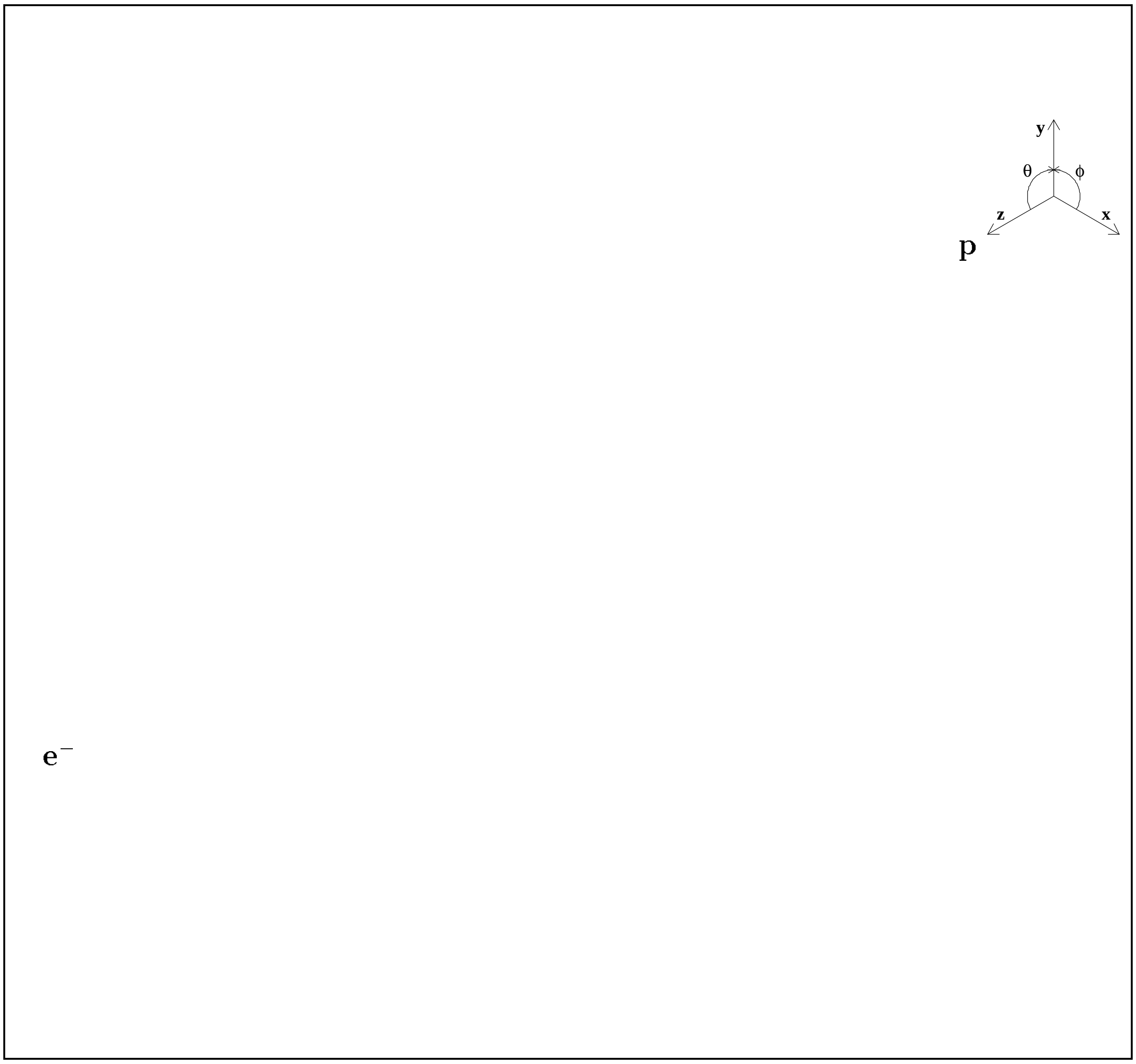
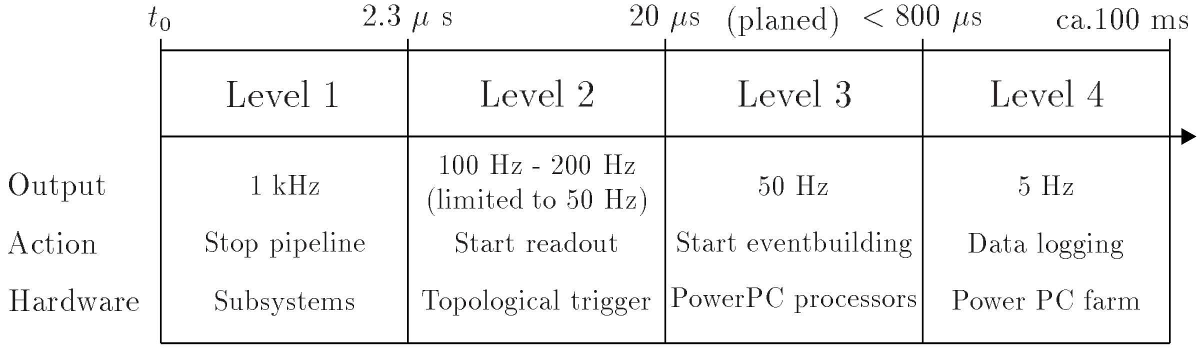


Figure 2.2: *Overview of the H1 detector.* $\boxed{1}$ *Beam pipe and beam magnets,* $\boxed{2}$ *Central tracker,* $\boxed{3}$ *Forward tracker,* $\boxed{4}$ *Electromagnetic LAr calorimeter,* $\boxed{5}$ *Hadronic LAr calorimeter,* $\boxed{6}$ *Superconducting coil,* $\boxed{7}$ *Compensating magnet,* $\boxed{8}$ *Helium supply for* $\boxed{7}$, $\boxed{9}$ *Muon chambers,* $\boxed{10}$ *Instrumented iron yoke,* $\boxed{11}$ *Forward muon toroid,* $\boxed{12}$ *Backward calorimeter,* $\boxed{13}$ *PLUG calorimeter,* $\boxed{14}$ *Concrete shielding,* $\boxed{15}$ *LAr cryostat*

The proton direction (positive z direction) is called the forward direction. The electron direction (negative z direction) is referred as the backward direction. Starting from the interaction vertex the detector consists of a central and a forward tracking system, each containing different layers of drift chambers and trigger proportional chambers. A liquid argon calorimeter surrounds the trackers. It is subdivided into an electromagnetic and a hadronic section. A superconducting cylindrical coil with a diameter of 6 m provides the analysing magnetic field of 1.15 T. The iron return yoke of the magnet is laminated and filled with limited streamer tubes. Hadronic energy leaking out of the back of the calorimeter is registered there. Muon identification benefits from additional chambers inside and outside of the iron. Muon tracks in the forward direction are analysed in a toroidal magnet sandwiched between drift chambers. Remaining holes in the liquid argon calorimeter are closed with warm calorimeters, a silicon-copper plug at very forward angles, a lead-scintillator calorimeter and lastly electron tagger in very backward regions. Two scintillator walls in the backward direction are installed to recognize background produced by the proton beam upstream of the H1 detector. Different parts of the H1 detector are grouped in so called subsystems. The subsystems are responsible for the readout of the subsystems detectors to the central DAQ. Several subsystems provide trigger systems for the different levels of the H1 trigger system.

2.2.1 H1 trigger system

Due to the low cross sections of ep interactions, the rate of genuine events is much smaller than the rates for background processes. Typical background sources are collisions between beam protons and rest gas in the vacuum pipes, scattering of off-momentum protons or electrons on beam apertures, collisions of protons between bunches (satellites) due to incorrect pre-acceleration or scattered synchrotron radiation from the electron beam. The different background sources are described in detail in Section 3.1. The HERA experiments need to rely on sophisticated triggering systems, selecting the good events very quickly. The trigger of the H1 detector is divided into four levels. Figure 2.3 shows an



	Level 1	Level 2	Level 3	Level 4
Output	1 kHz	100 Hz - 200 Hz (limited to 50 Hz)	50 Hz	5 Hz
Action	Stop pipeline	Start readout	Start eventbuilding	Data logging
Hardware	Subsystems	Topological trigger	PowerPC processors	Power PC farm

Figure 2.3: *H1 trigger levels. The trigger of the H1 detector is divided into four levels. The first and second level systems are phase-locked to the HERA accelerator clock. The further trigger levels run asynchronously to the HERA clock. The L2 trigger output rate is limited to 50 Hz because no L3 trigger is implemented at the moment.*

overview of the different trigger levels. A description of the H1 trigger system can be found in [1, 2].
The first and second level systems (L1 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 after 2.3 μs without causing dead time. Pipelines in the different subsystems store the data for this period. Every L1 subsytem generates trigger information coded in bit pattern of a multiple of 8 bits. These bit patterns are called trigger elements and contain special information of the event. With Boolean decisions subtriggers are formed from the different subsystem trigger elements in the central trigger system. The central trigger system merges all the trigger informations of the different subsystems and the different levels.
If one subtrigger comes to a positive decision the L1 Keep signal is sent to the subsystems and stops the pipelines. Figure 2.4 shows the principle of pipelining and the calculation of the trigger decision. All signals used until this point are synchronous to the HERA Clock. These signals are called fast signals. After the L1 trigger decision the asynchronous part of the trigger and the readout system begins. Signals used in the asynchronous communication are called slow signals.
The L2 trigger system decision is presently derived from a combination of two independent hardware systems within 20 μs of the preceeding L1 Keep signal, allowing sufficient time for transmission of trigger information and computation of the L2 trigger response. The L2 trigger output rate is at the moment limited to 50 Hz because no L3 trigger is implemented

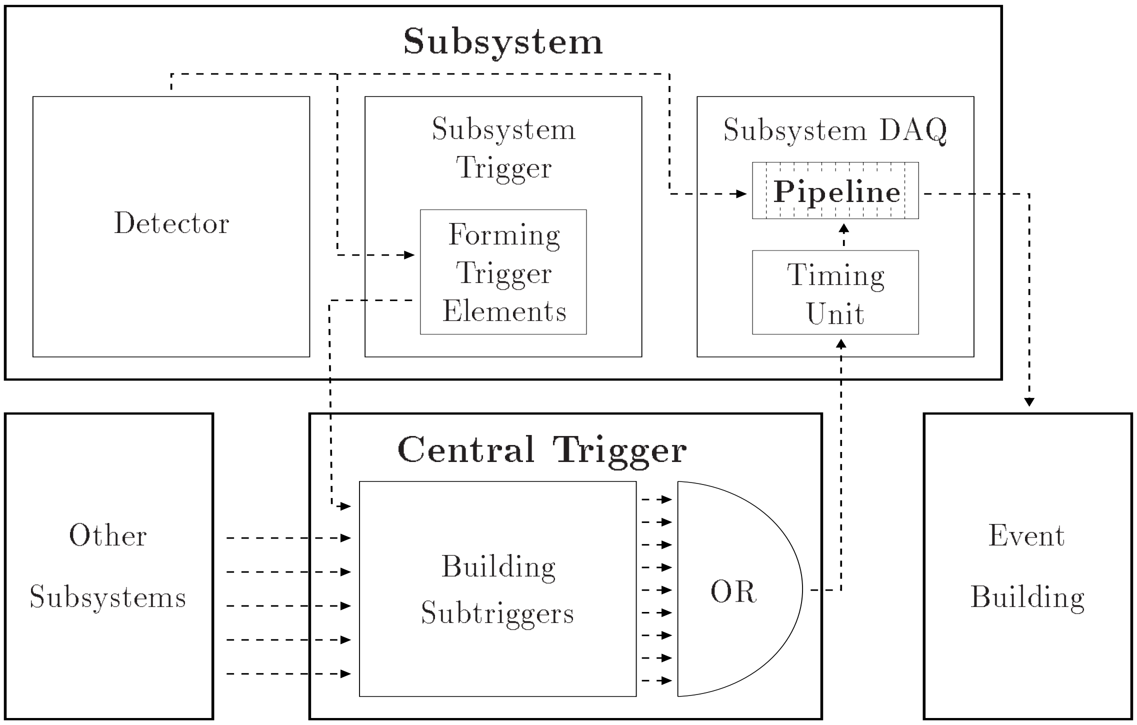


Figure 2.4: *Principle of pipelining in H1 subsystems. The chamber information is stored in a pipeline to avoid dead time in the first level of the trigger system. After a positive trigger decision the corresponding event information is read out from the pipelines.*

yet. The future L3 system will be a software trigger. The L4 stage of the trigger system cuts on full event quantities and manages the data logging and a preprocessing of the event reconstruction. Only after the data logging a full reconstruction of the events is performed.

2.2.2 H1 readout system

An overview of the H1 DAQ system can be found in [1, 2]. A total of over a quarter of a million analogue channels are read out and digitized, resulting in some 3 Mbyte of raw digitized information for a triggered event. Various levels of hardware triggering, software filtering and digital compression reduce the event size to between 50 Kbyte and 100 Kbyte. Figure 2.5 shows an overview of the H1 detector readout system. The whole data is passed from the subsystems via an optical readout ring to the event building unit where the data is examined in different stages and written on tape. For the readout the individual subsystems use VME bus crates each containing a readout controller, a memory buffer and a fibre optic link to a coordinating event manager. The event manager coordinates the data flow between the optical readout ring, systems observing the read

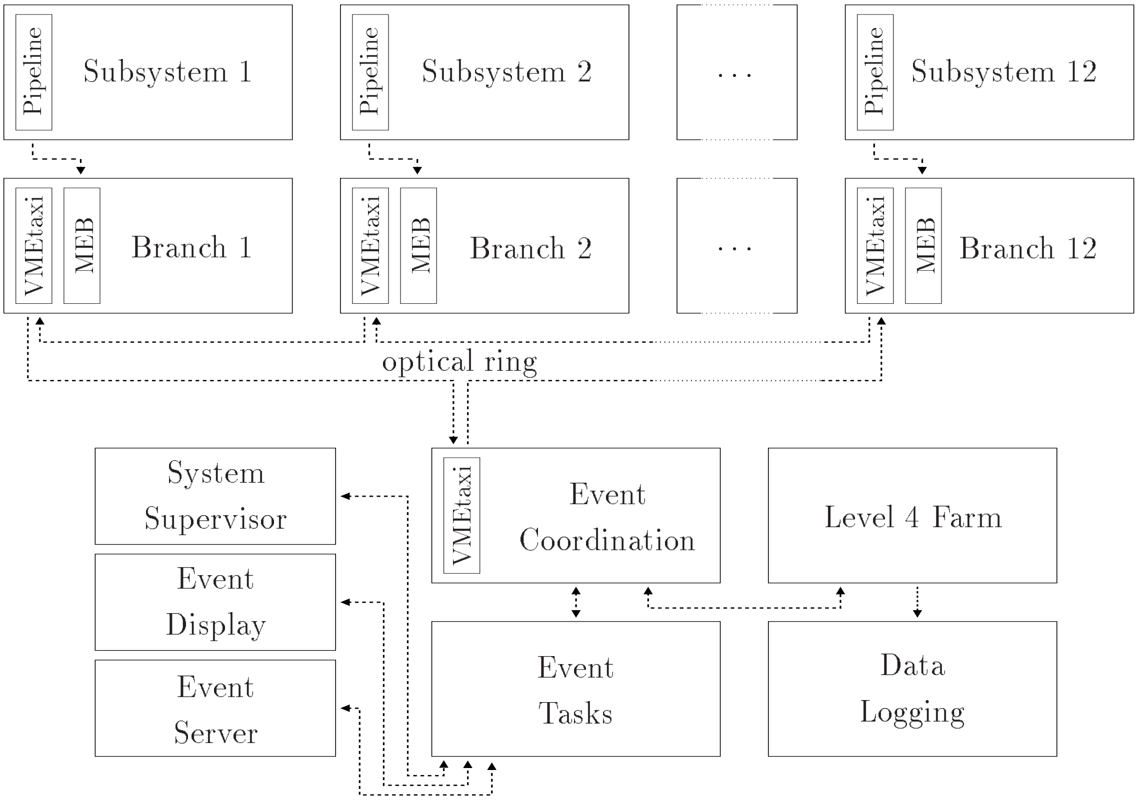


Figure 2.5: *Principle of the H1 detector readout. The different subsystems are connected to the main event building via an optical ring. The event coordination collects the full event data. It is processed in the event task unit. After processing the data it is send to the data logging. The Level 4 farm does first online reconstructions on the read out data.*

out data and the process of data logging. The different parts of the optical link are called branches. Some subsystems share the same optical branch. The VME bus and VSB bus are used as the main standards in the H1 detector readout. VMEtaxi [3] cards are used to drive the optical ring connecting the subdetector VME crates to the event building.

2.3 The H1 upgrade for HERA 2000

The investigations of rare processes at the limits of the kinematically accessible range at HERA requires high luminosity. For instance in the very high Q^2 region *ep* collision experiments allow to give the best limits compared to other accelerators on hypothetic particle states, which have both leptonic and quark quantum numbers ("Leptoquarks"). The goal is to increase the luminosity of HERA by a factor of 5 to about 7.4×10^{31} $\mathrm{cm}^{-2}\mathrm{s}^{-1}$, which will lead to an annual integrated lumionsity of 150 pb^{-1}. For further information on the physics goals of the upgrade see [4].

The higher luminosity will mainly be achieved by a stronger focusing of both beams in the experimental areas. This requires the installation of three new superconducting magnets inside the H1 detector, one in the forward and two in the backward hemisphere. Each of them is equipped with an independant cooling system. To gain enough space for these magnets and to adapt to the new beam geometry, significant changes to the inner part of the H1 detector are necessary. First of all the vacuum beam pipe (made from Aluminium - Beryllium alloy) will have an elliptical shape corresponding to the synchrotron radiation envelope and allowing to install at least part of the CST as close as possible to the interaction region. The inner hole of the SPACAL needs to be enlarged to make room for the new magnets.

We have to assume, that also the beam related background will increase with higher luminosity, although it is very difficult to predict quantitatively how much. For this reason the CIP2000 upgrade project replaces the CIP and CIZ chambers by a new 5 layer MWPC. A new fast trigger system based on the cathode pad signals of this new chamber will be able to identify uniquely upstream proton background and provide a strongly improved rejection capability for those events. Electron beam induced photon background will be suppresed due to the new five layer chambers instead of only two in the old system.

Although the main goal of the upgrade is the study of the very high Q^2 region with its low physics event rates, the access to low p_t physics, mainly for measurements using heavy quarks is still interesting and has not yet been fully exploited with the present system. *ep* events with only light quarks in the final states are the main background to these events. Using the present trigger system heavy quark events could only be accessed with good efficiency by increasing the overall bandwidth of the data taking system, which would be prohibitive. Therefore a Fast Drift Chamber Track Trigger (FTT) will be built to find tracks in the CJC detectors. Event signatures indicating heavy quark or other interesting final states will be calculated with the FTT in trigger level 2 and 3, thus allowing to selectively acquire these events.

In addition the calorimeter jet trigger will be adjusted to a higher granularity. The luminosity system will have to be adapted to the new luminosity situation. Detectors like the Forward Silicon Tracker (FST), Backward Silicon Tracker (BST), Forward Track Detector (FTD), Forward Neutron Calorimeter (FNC), Forward Proton Spectrometer (FPS) or Backward Proportional Chamber (BPC) will substitute or complement old detectors or are totally new developments. A new high performance processor farm will be able to fully reconstruct the event data online, allowing more specific physics cuts and monitoring tasks to run on the data before storage.

Chapter 3

The CIP2000 trigger system

3.1 Motivation

Due to the higher luminosity of the HERA 2000 upgrade also more background events are expected. Both the electron and the proton beam can be the source of background events. Protons can collide with residual gas particles. In addition they can leave the nominal orbit and collide with the beam pipe or e.g. the magnets. Bad pre-acceleration can lead to protons outside the correct bunch timing (satellite bunches). The satellite bunches can too collide with residual gas particles or with e.g. the beam pipe. The proton generated background results in event signatures with the interaction vertex on the beam axis but at a different position in the z direction than the nominal vertex region. The electron beam creates synchrotron radiation. The photons from this radiation can produce pairs of particles by photon conversion. These particles are absorbed in the chambers very fast due to their low energy. Mostly no tracks can be reconstructed in this event signature. These events are rejected automatically because of the five layers of the CIP2000 chamber. The CIP2000 trigger builds coincidences of pads in the five layered CIP2000 chamber and defines tracks of particles going through the detector. Tracks originating from the beam axis of the H1 detector are counted and summed up in different regions (size: 20 cm) of the x-axis separatly. The resulting numbers form a histogram called z-vertex histogram. It reaches from the forward direction nearly through the whole main detector up to the SPACAL. From this information proton-upstream background events can be easily identified. The events can be rejected or specially examined.
Simulations have been done to examine the background rejection of the CIP2000 trigger [5]. The geometrical constraints for the CIP2000 chamber geometry have been investigated [6] and a hardware trigger algorithm has been developed and simulated [7]. The loss for physics events is less than 1% whereas background from outside the nominal interaction region with lots of activity in the detector is rejected with an efficiency of more than 95%.
The old CIP chambers and the CIZ chamber will be moved out of the experiment to provide space for the new five layered CIP2000 chamber. The forward MWPC system will be taken out but the old MWPC trigger will still exist as it is, besides not providing the forward region. It uses the two inner layers of the new CIP2000 chamber and the

COP chamber to build a z-vertex histogram similar to the histogram provided by the CIP2000 trigger. The zVtx trigger z-vertex histogram covers a smaller region in z but has a better resolution (bin size: 5.49 cm). The zVtx trigger examines the region of the nominal vertex but does not provide informations of other regions in the z direction. The binning of the z-vertex histogram for the zVtx and the CIP2000 trigger can be found in Figure 3.5.

3.2 Trigger algorithm

The trigger algorithm of the CIP2000 trigger calculates a z-vertex histogram to determine the ratio of tracks originating from the *ep* interaction point to tracks originating from very forward or backward regions of the detector. This information is coded in the trigger elements of the CIP2000 trigger and sent to the central trigger unit.

To make the implementation of the trigger algorithm most efficient and fast the main tasks are separated into small steps which can be calculated in parallel. The possibility for a separation results mostly from the geometry of the chambers. With assuming that all tracks measured in the chambers are coming from the beam axis of the H1 detector the algorithm can be separated in ϕ. The local z-vertex histograms calculated in every sector in ϕ can be summed up to a total z-vertex histogram for the whole chamber. A "projective geometry" was chosen for the chamber geometry in order to allow to separate the algorithm in the z direction. This method is explained in detail in Section 3.3. These possibilities for the separation of the trigger algorithm leads to an implementation concept with the following three main parts, visualized in Figure 3.2:

Tracking: To find all the tracks in a given sector in ϕ the algorithm is subdivided into smaller parts. For every pad in the middle layer called central pad a local environment is defined. An example of a local environment is shown in Figure 3.1. The required binning of the z-vertex histogram defines how the algorithm searches for tracks in the local environment. The binning can be shifted in parts of one bin width by defining an offset in the definition of the local environment. Figure 3.1 shows how the tracks are defined. Because of the projective geometry of the chambers the definition of the tracks in the local environment is independent of the z position of the central pad. For the local environment of every pad in the middle layer it is memorized how many tracks originate from each bin. This is achieved by simple comparison of the bit pattern. The information is called the hit list. The algorithm is executed in parallel for every central pad and every sector in ϕ.

Adding the main histogram: The hit lists of all central pads in one sector in ϕ are added up. The result is a local z-vertex histogram for every sector in ϕ. After all local histograms are added up the main z-vertex histogram for the whole chamber is available.

Building trigger elements: Trigger elements (16 bit) are built from the main z-vertex histogram. The detailed definition of the trigger elements is not yet defined.

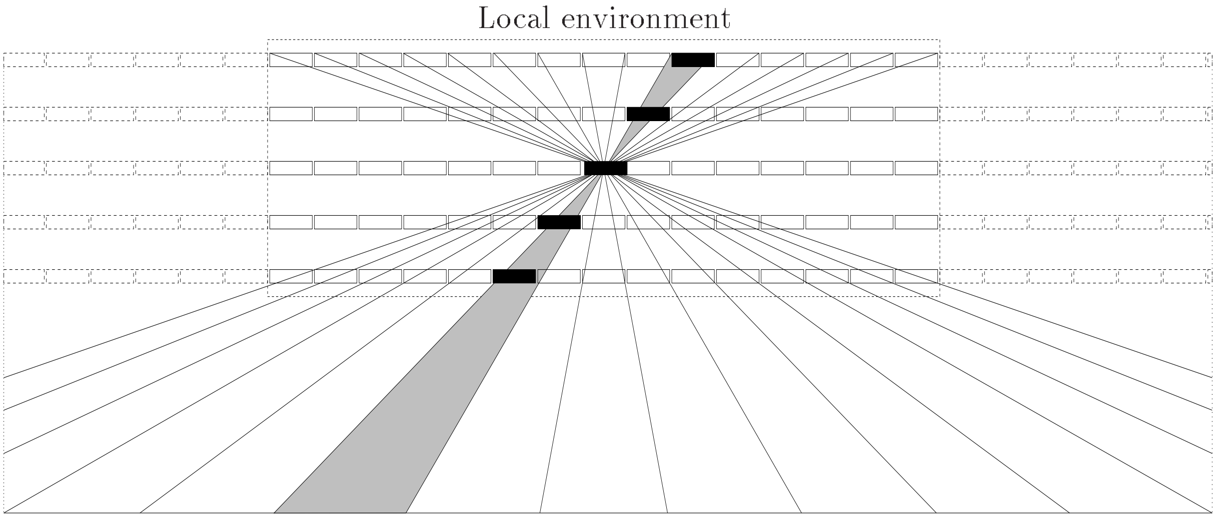


Figure 3.1: *Definition of a local environment. The possible tracks are defined in the local environment. Because of the projective geometry of the chambers the definition of the tracks in the local environment is independent of the z position of the central pad. The tracks are recognized by simple comparison of the bit pattern.*

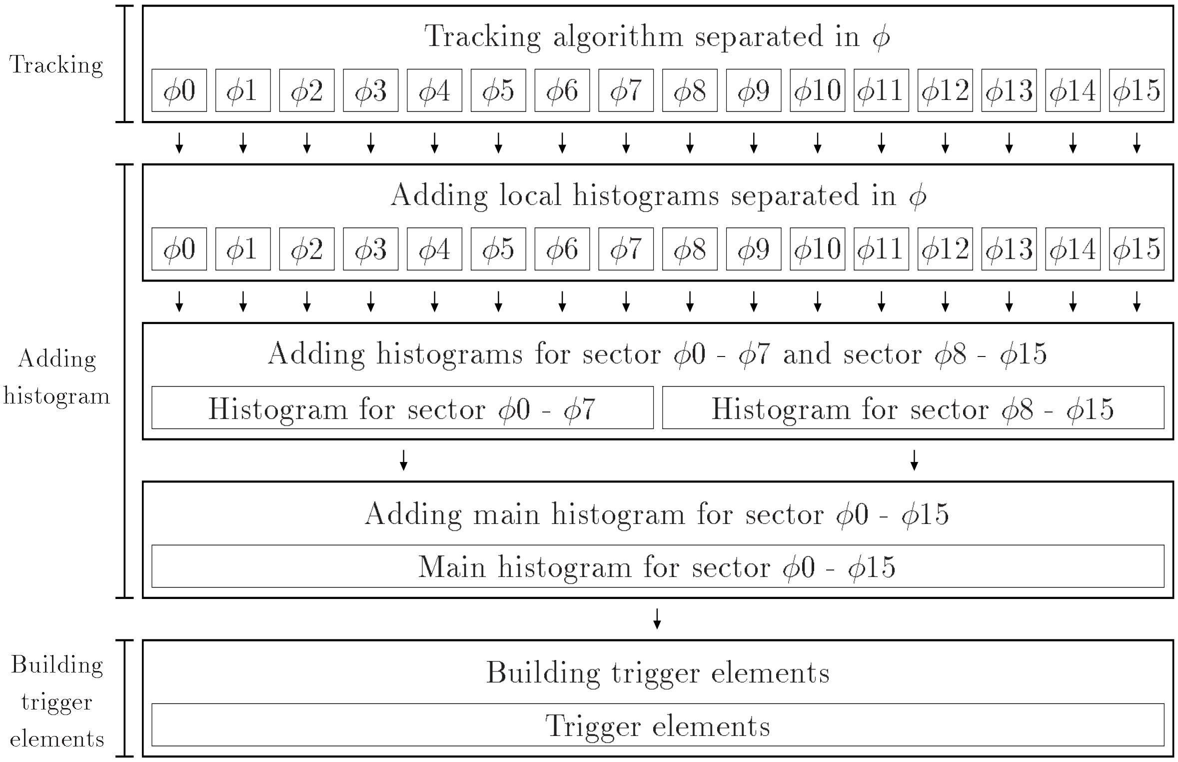


Figure 3.2: *Overview of the CIP2000 trigger algorithm. The main tasks of the trigger algorithm can be separated in small steps which can be calculated in parallel. The possibility for a separation results mostly from the geometry of the chambers.*

3.3 Projective geometry

To make the tracking algorithm easier a projective geometry was chosen. Figure 3.3 demonstrates what a projective geometry means and defines a coordinate system. The projective geometry means that the pad size in z of one layer is proportional to the radius of the layer. The tracking algorithm searches for tracks in a local environment like it is shown in Section 3.2. Every pad in a layer gets its pad number by counting the pads from the -z end of the chamber on. Fifteen consecutive pads of each layer with the same pad number in each layer form a local environment. The different local environments are tilted against each other. If one searches for a track in the environment originating from a given point on the z-axis one has to look for a special bit pattern. The projective geometry has the consequence that all possible tracks originating from the given point on the z-axis have the same bit pattern in the different local environments. The algorithm does not have to know the z position of the local environment to find a track originating from a special point on the z-axis. This means that all units running the tracking algorithm can run with the same setup.

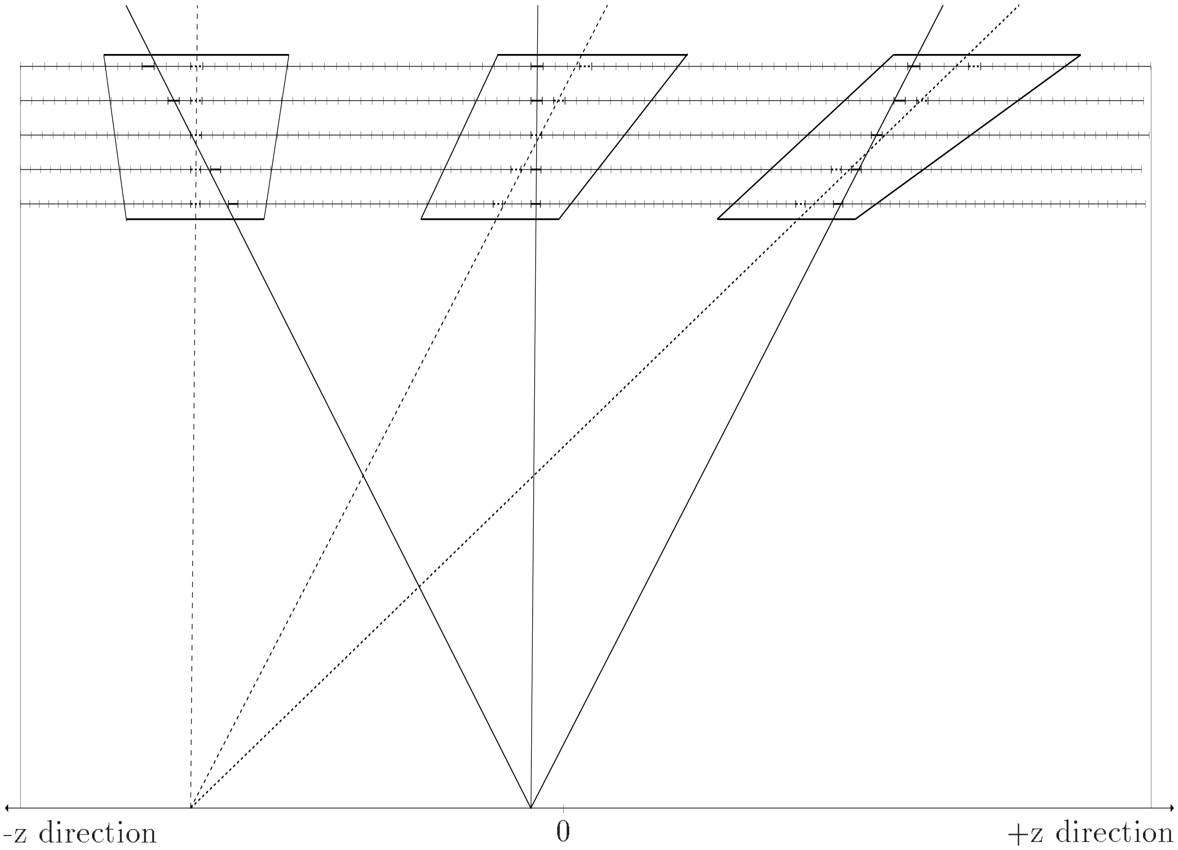


Figure 3.3: *Projective geometry. The pad size of one layer is proportional to the radius of the layer. The local environment of a pad in the central layer is tilted along the z direction the way that every track originating from one point on the z axis makes the same bit pattern in the local environment of any central layer pad.*

3.4 Trigger system implementation

The CIP2000 trigger system hardware implementation consists of three main parts:

- CIP2000 chamber (5 layers) and chamber readout
- Trigger system and STC (Subsystem Trigger Control) system
- Trigger readout system

Figure 3.4 shows an overview of the whole system. The CIP2000 chamber consists of 5 cylindrical layers. Mounted directly on the chamber there is electronics, which amplifies, discriminates and multiplexes the signals before they are sent via an optical fibre to the trigger crates in the electronic trailor. FPGAs (**F**ield **P**rogrammable **G**ate **A**rrays) on trigger cards in the trigger crates run the trigger algorithm in several stages and store the chamber information. The trigger elements are calculated in FPGAs on sum cards and

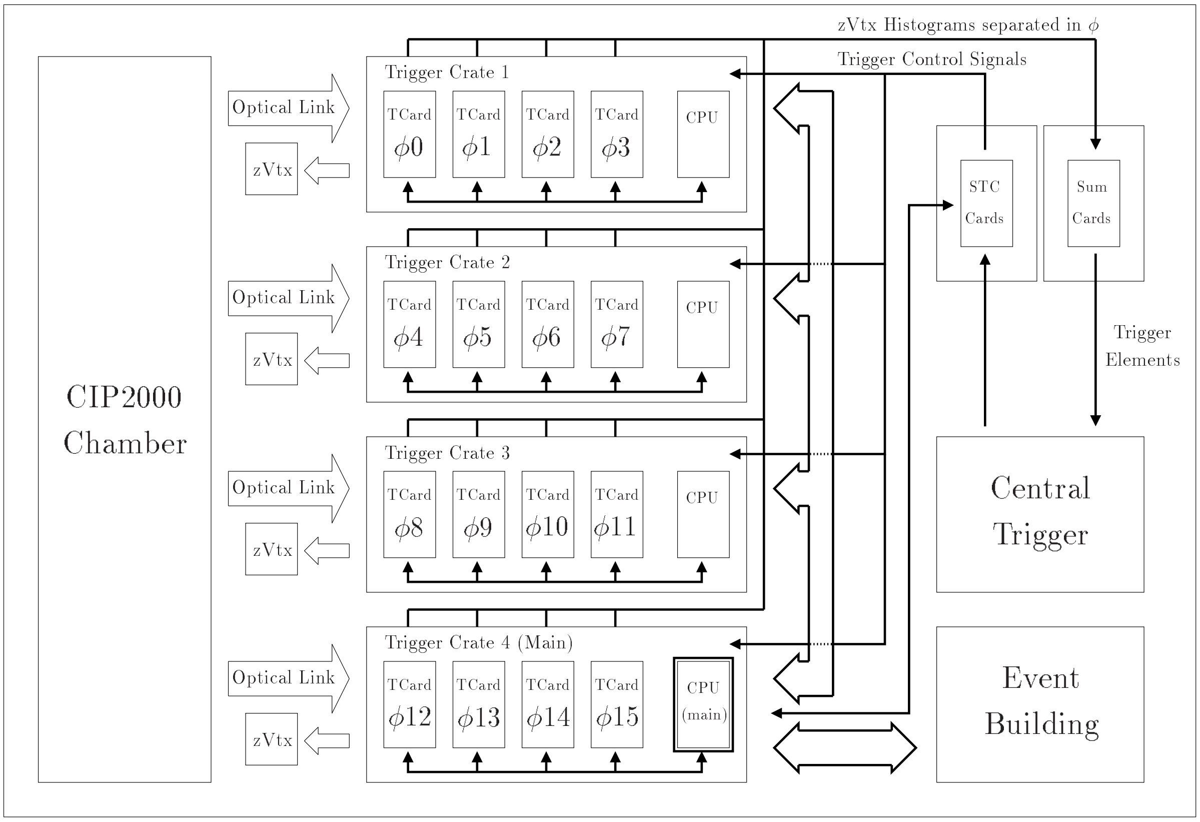


Figure 3.4: *The CIP2000 chamber consists of 5 cylindrical layers. On chamber chips amplify, discriminate and multiplex the signals before they are sent via optical fibres to the trigger crates. FPGAs on trigger cards run the trigger algorithm and store the chamber data. The trigger elements are built on sum cards. Trigger CPUs read out the chamber data, compress the information and send it to the main event building. The STC system manages the communication between the central trigger unit and the CIP2000 system.*

are sent to the central trigger unit. CPUs read out the chamber informations, compress the data and send it to the main event building. The STC system manages the communication between the central trigger unit and the CIP2000 trigger system.
FPGAs are an essential part of the CIP2000 trigger system. Investigations have been made if ASICs, DSPs or FPGAs are best suited for managing the tasks of running the trigger algorithm and storing the chamber data. The FPGAs contain freely programmable logical units thus combine a performance in speed and parallelism like a digital ASIC with the flexibility of a multiprocessor like a DSP. The FPGAs used in the system have a random access memory built into the chip to store the chamber data. It turned out that FPGAs are best suited for the requirements of the CIP2000 trigger system.

3.4.1 Chamber geometry

The chamber of the CIP2000 trigger consists of 5 cylindrical layers. Each layer is divided in 16 Sectors in ϕ and about 120 equidistant pads in the z direction. A projective geometry was chosen to make the tracking algorithm easier. The characteristic of this geometry is illustrated in Section 3.3. Table 3.1 shows the detailed sizes of the pads in the different layers. The chamber has an acceptance in Θ of 10° - 170°. Figure 3.5 shows a side view of one sector in ϕ of the new CIP2000 chambers.

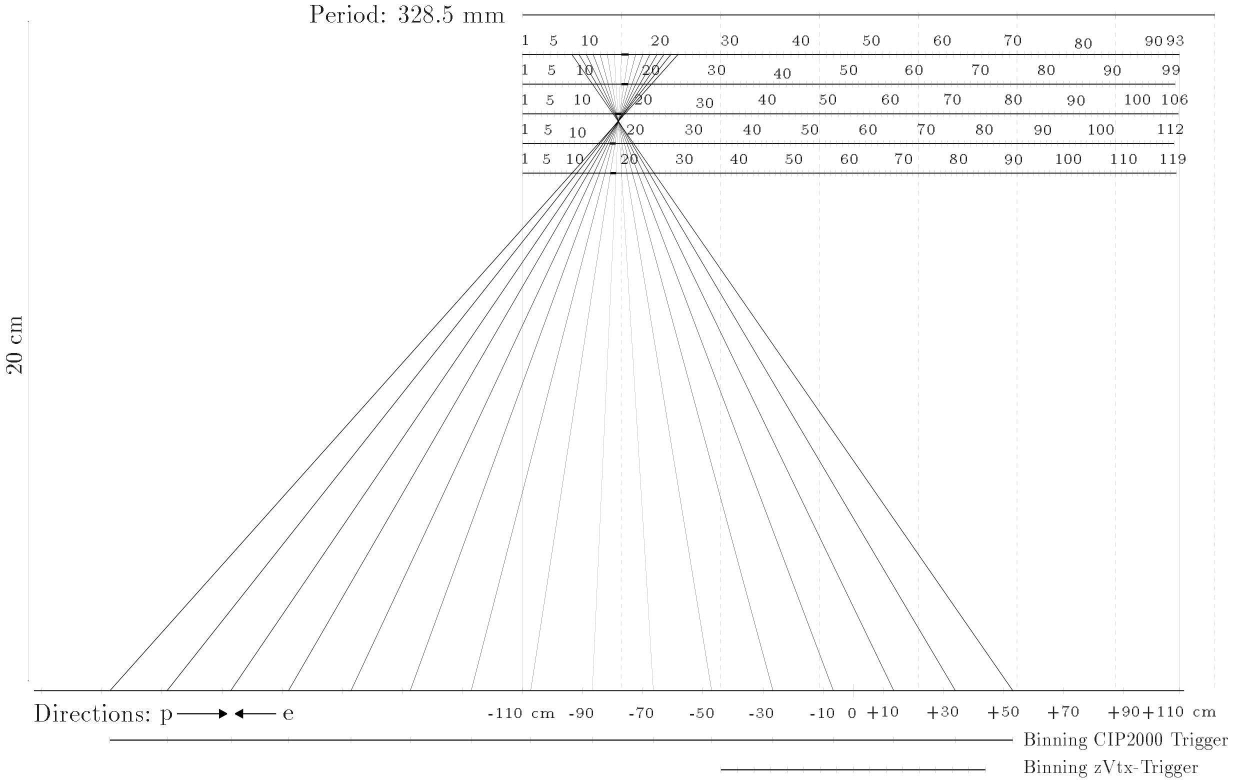


Figure 3.5: *Side view of one sector in ϕ of the new CIP2000 chamber. The binning of the CIP2000 trigger histogram in comparison to the histogram of the zVtx trigger is shown. By changing the program of the trigger algorithm on the FPGAs on the trigger cards the bins can be shifted in forward and backward direction in steps of ~20 cm.*

The figure shows the binning of the CIP2000 trigger histogram in comparison to the histogram of the zVtx trigger. By changing the program of the trigger algorithm on the FPGAs on the trigger cards the bins can be shifted in forward and backward direction in steps of ~20 cm. Figure 3.6 shows a front view of the new CIP2000 chambers with a part of the surrounding detector.

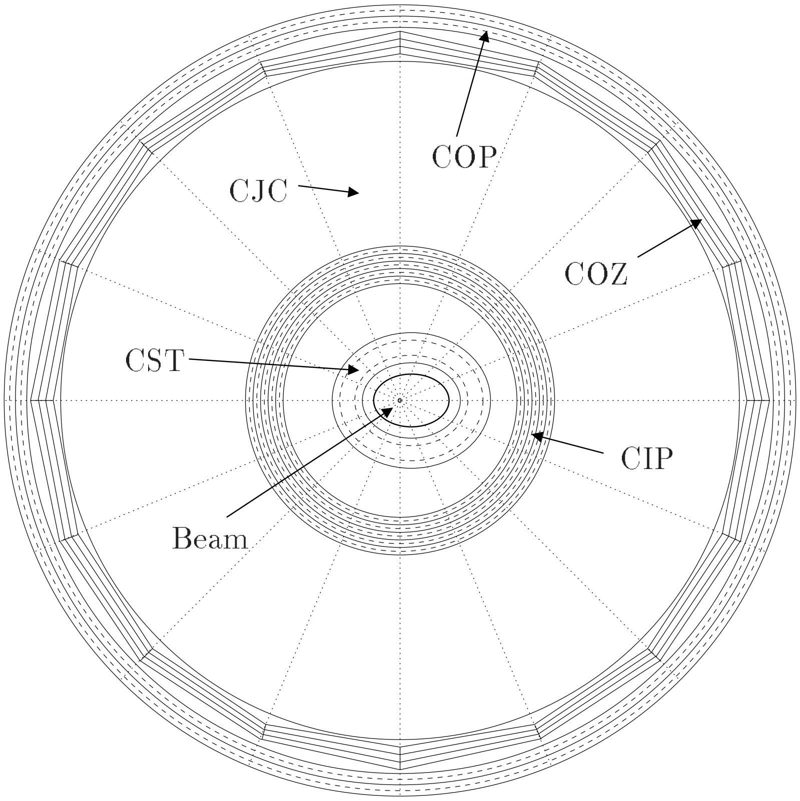


Figure 3.6: *Front view of the CIP2000 chamber with the surrounding detector.*

Layer	Radius[mm]	Pad length[mm]	Number of Pads
0	157	18.250	119
1	166	19.323	112
2	175	20.531	106
3	184	21.900	99
4	193	23.464	93

Table 3.1: *Pad geometry of the new CIP2000 chamber*

3.4.2 Chamber readout

The signals of the chamber pads are amplified, discriminated and synchronized to the accelerator bunch crossing clock (Hera Clock) by an ASIC called CIPiX. Every CIPiX has 64 channels and is connected to 60 pads in one layer and one sector in ϕ. The whole system uses 160 CIPiX. The digitized data is multiplexed 4 times on the CIPiX chip. The chips can be controlled by a processor via an optical link using the I^{2}C standard [8]. The modes of the preamplifier, shaper, discriminator and the multiplexing can be set. A test pulse generator can be used to test the readout chain. More information on the CIPiX chip can be found in [9].
The digitized data of four CIPiX chips is collected in one optical hybrid. Here the data is again multiplexed 16 times and sent via an optical fibre to the optical hybrids in the electronic trailor. Control signals like the HERA clock are sent from the electronic trailor to the chamber via the same link. The optical link operates at a rate of about 800 MHz. The optical hybrids in the electronic trailor are mounted on the back of the trigger crate backplanes. 10 of these optical hybrids provide the data of 4 sectors in ϕ. The data is demultiplexed 16 times. The backplane sorts the signal lines and distributes the data to the trigger cards. Every trigger card needs the data of all layers in one sector of ϕ. The FPGAs on the trigger cards demultiplex the data 4 times and store the data in pipelines. The same FPGAs run the trigger algorithm. Figure 3.7 shows the readout chain of the CIP2000 chamber. Figure 3.8 shows how the data is distributed in the CIP2000 chamber readout for 4 sectors in ϕ.
Every layer has its own I^{2}C bus ring connected to all CIPiX chips in a layer. The five busses are driven by a processor in the electronic trailor. This processor receives its comands from the main CIP2000 CPU in serial form.

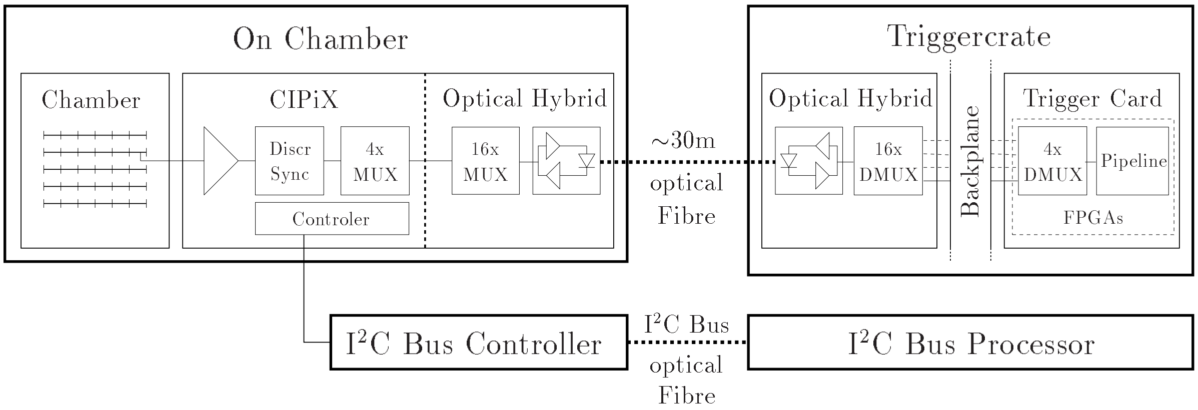


Figure 3.7: *Signal path of the CIP2000 chamber readout. The CIPiX chip amplifies and multiplexes the data 4 times. Again the data is multiplexed 16 times and converted in optical signals in the optical hybrid. The optical link operates at a rate of about 800 MHz. The optical hybrid on the back of the trigger crate backplane demultiplexes the data 16 times. The backplane sorts the signals and distributes the data to the trigger cards. The FPGAs on the trigger cards demultiplex the data 4 times and store the data in pipelines.*

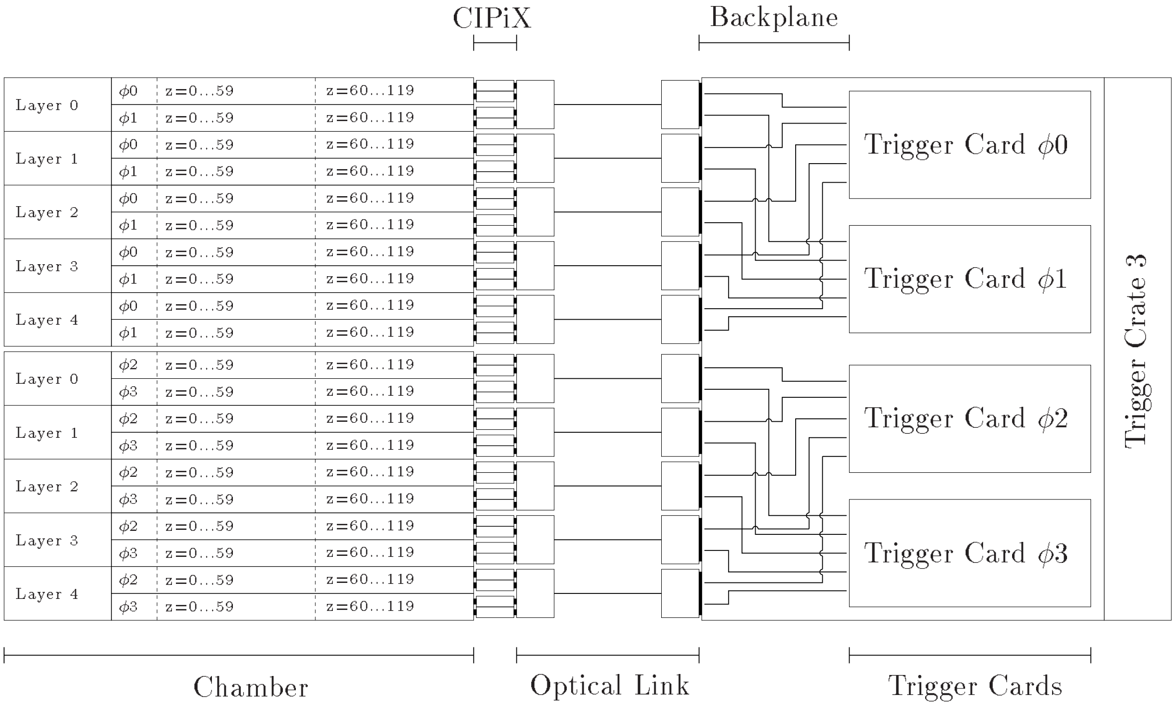


Figure 3.8: *Overview of the CIP2000 chamber readout for 4 sectors in* ϕ. *Two CIPiX chips amplify and multiplex the data of one layer and one sector in* ϕ. *The data of four CIPiX chips is multiplexed and converted in optical signals in one optical hybrid. The optical hybrid on the back of the trigger crate backplane demultiplexes the data. The backplane sorts the signals and distributes them to the trigger cards. Every trigger card needs the data of all layers in one sector of* ϕ.

3.4.3 Trigger system hardware

Information about the CIP2000 trigger system and the hardware components can be found in [10]. The CIP2000 trigger system is built following the structure of the trigger algorithm with a maximum of flexibility in adapting to possible new modifications of the algorithm. Figure 3.9 shows an overview of the CIP2000 trigger system structure. The trigger system consists of four trigger crates and additional crates housing the STC system and the sum cards to add the histograms of the different sectors in ϕ. In every trigger crate there are four trigger cards, two control cards and a CPU board mounted on the front side of the backplane. The CPU manages the chamber readout via the VME bus integrated in the J1 and J2 backplane of the trigger crates. Ten optical receiver cards are mounted on the back of the J2 backplane. The backplane sorts the signals of the chamber and distributes the signals to the different trigger cards. The control cards receive control signals from the STC system and deliver the signals to the trigger cards via the backplane.

The parallelism of the trigger algorithm determines the structure of the trigger system. Every trigger card is assigned to one sector in ϕ. The trigger card receives the 600 bit (120 pads × 5 layers) wide pattern of one sector in ϕ four times multiplexed from the backplane. The main part of the trigger card are the two FPGAs running the tracking algorithm. Each FPGA builds the histogram for one half of one sector in ϕ and stores the chamber data in an integrated pipeline. The histogram is transfered to one of the FPGAs to build the local histogram of one sector in ϕ. After the histograms are calculated they are sent to the pre-sum cards. The pre-sum cards add the histograms of 8 sectors in ϕ and sends the result to the main-sum card. The main-sum card builds the main z-vertex histogram and calculates the trigger elements before they are sent to the central trigger unit.

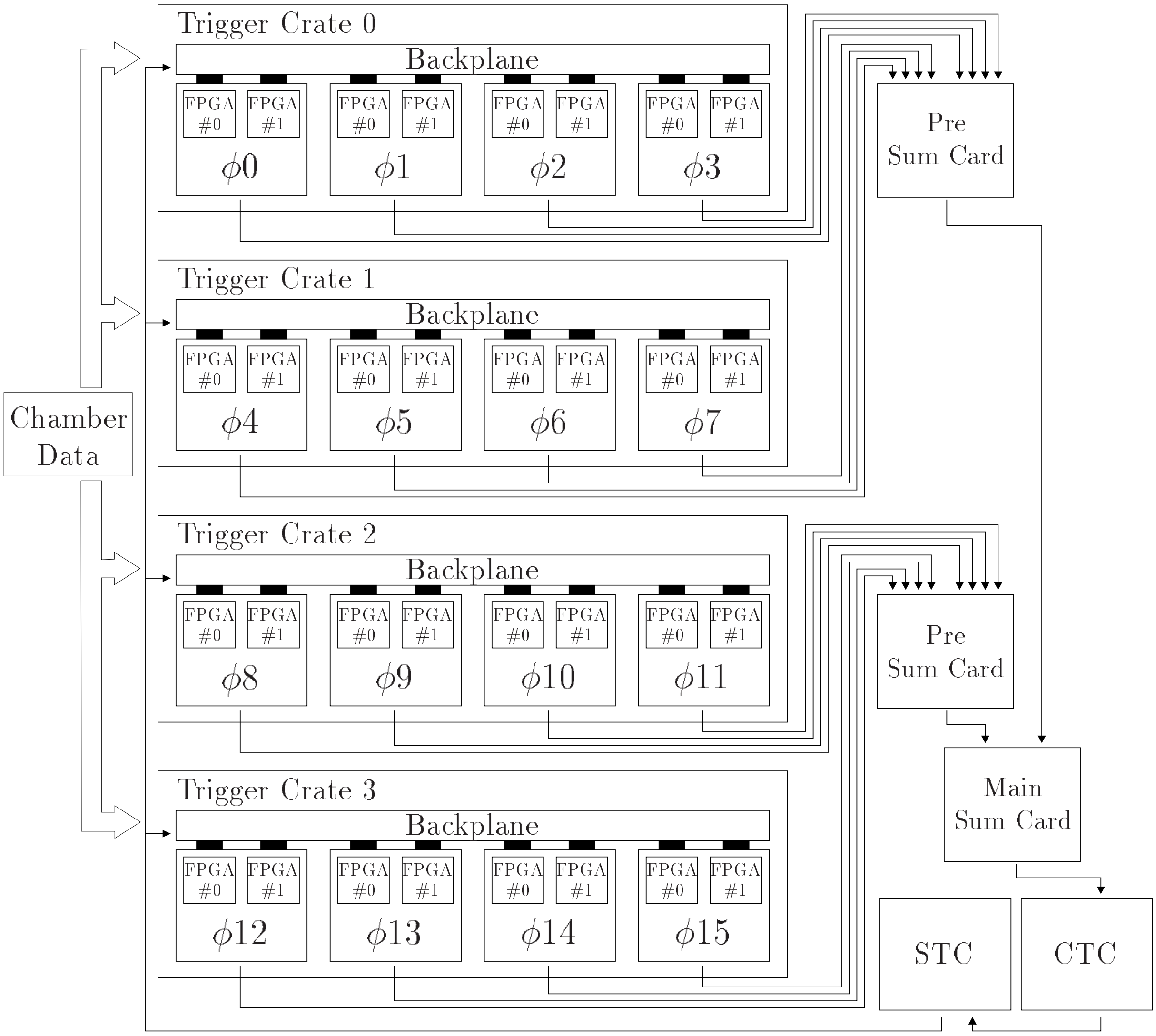


Figure 3.9: *CIP2000 trigger system structure. The trigger system consists of four trigger crates and crates housing the STC system and the sum cards to add the histograms of the different sectors in* ϕ*. Every trigger card is assigned to one sector in* ϕ*. The trigger card receives the chamber data of one sector in* ϕ *four times multiplexed from the backplane. The local z-vertex histograms are calculated in the trigger cards and sent to the pre-sum cards. The pre-sum cards add the histograms and deliver each the histogram of the half chamber in* ϕ *to the main-sum card. The main-sum card builds the main histogram and calculates the trigger elements to send them to the central trigger unit.*

Figure 3.10 shows an overview of the trigger card. The main devices on the trigger cards are the two FPGAs APEX 20k400GCC/1 [11] with 652 pins from the company Altera. The FPGAs have an internal memory space to implement a pipeline for the chamber data and sufficient room to run the trigger algorithm. They are running at a frequency of 41.6 MHz. First the chamber data is four times demultiplexed. The hit list is generated in each FPGA containing the histogram information of the local environment of every central pad in the sector in ϕ for the pads assigned to the FPGA. The local histogram is built in each FPGA. FPGA 1 sends its local histogram consisting of 15 bins with numbers of the width of 6 bit (90 bit transfer between FPGAs) to the FPGA 0. The FPGA 0 adds both local histograms resulting in a histogram of 15 bins with 7 bit wide numbers. The data of the histogram is multiplexed four times and converted into differential LVDS (ANSI/TIA/EIA-644 LVDS, IEEE 1596.3)[12, 13] standard resulting in 32 bit data that is four times multiplexed.

The trigger card is connected to the pre-sum card via cables with SCSI connectors (68 pins). An additional SCSI connector (50 pins) can be used to transfer additional data calculated (e.g. integrated histograms) in the FPGA 0 to other parts of the CIP2000 system or even to other H1 subsystems. A VME [14] controller built in a Lattice ispL-1048 PLD [15, 16] manages the VME access to the trigger card. The pipelines and special registers in the FPGAs, control registers of the VME controller and EEPROMs containing the FPGA program can be read and written via the VME bus. VME D32/A24 access in single cycle and block mode are supported. Section 5.3.1 describes the VME bus in detail. The FPGAs can be programmed via an external connector, via the EEPROMS on the trigger card or via the VME bus. A JTAG (Joint Test Action Group) [17, 18] chain is implemented on the trigger card connecting the external connector, the EEPROMs and the FPGAs. The JTAG chain can test the connections between the devices and manages the programming of the FPGAs.

Figure 3.11 shows an overview of the sum cards. Two pre-sum cards and one main-sum card are planned. Both types of sum cards use one FPGA of the same type like the trigger cards. The pre-sum cards are connected to the trigger cards via eight cables with SCSI connectors. After the signals from the trigger cards are received on the pre-sum card they are converted back from LVDS standard. The FPGA demultiplexes the signals and adds up the histogram for eight sectors in ϕ. The pre-sum cards send the histogram consisting of 15 bins with 10 bit wide numbers via the 150 bit wide link to the main-sum card. The main-sum card adds the main histogram resulting in a 15 bin wide histogram with numbers of the width of 11 bit and builds the trigger elements to send them to the central trigger unit. The sum cards have the same VME controller like the trigger cards to access registers on the FPGA and to programm the EEPROMS or the FPGA. The FPGA can be programmed via a JTAG chain like on the trigger cards.

Figure 3.12 shows an overview of the control cards. Its main task is to distribute the Hera clock signal, Pipeline Enable signal and a Global Reset signal to the trigger cards via the backplane and to distribute the Hera clock signal to the chamber electronics via the optical link. The Hera Clock signal can be delayed individually for every layer of the CIP2000 chamber. This is necessary to adjust the timing in the optical chamber readout. The delays can be set via the VME bus. The optical readout cards deliver a 40 MHz clock generated in the chamber electronics from the Hera Clock. These signals are received by

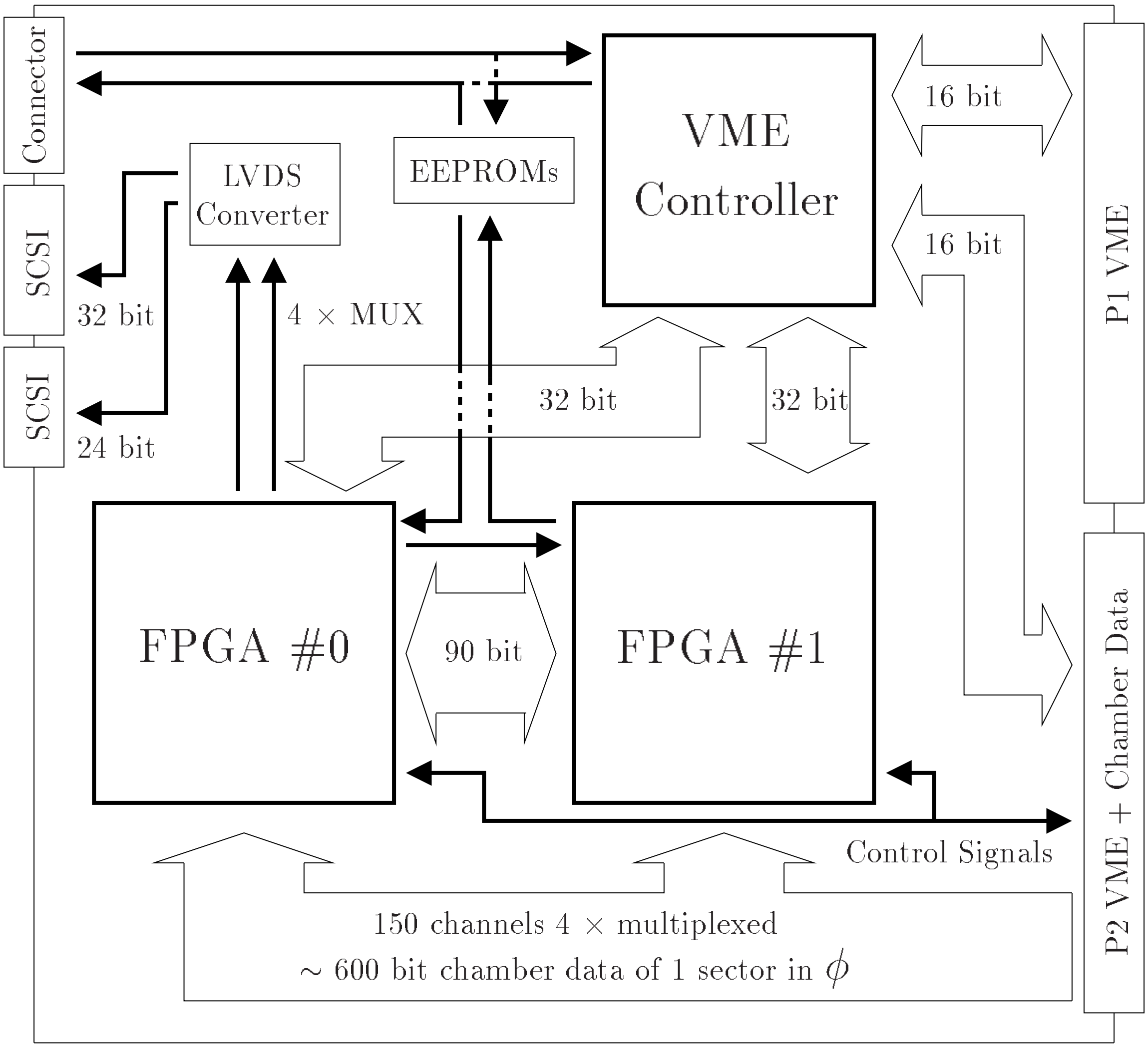


Figure 3.10: *CIP2000 trigger card. The two FPGAs APEX 20k400GCC/1 from the company Altera have an internal memory space to implement a pipeline for the chamber data and sufficient room to run the trigger algorithm. First the chamber data is four times demultiplexed and stored in the pipeline. The hit list is generated in each FPGA. The local histogram is build in each FPGA and the FPGA 1 sends its local histogram (90 bit transfer between FPGAs) to the FPGA 0. The FPGA 0 adds both local histograms resulting in a histogram of 15 bins with numbers of the width of 7 bit. The data of the histogram is multiplexed four times and converted into differential LVDS standard. The trigger card is connected to the pre-sum card via cables with SCSI connectors. An additional SCSI connector can be used to transfer additional data calculated in the FPGA 0 to other parts of the CIP2000 system or even to other H1 subsystems.*

the control card and the phase of the signals in different layers is examined and certain values are made available in registers. These registers can be read via the VME bus. The delays for the Hera clock can be adjusted depending on the values in the register.
Certain chamber signals from the inner layers are also needed for the zVtx trigger of the old MWPC trigger system. These chamber data is made available in LVDS standard on

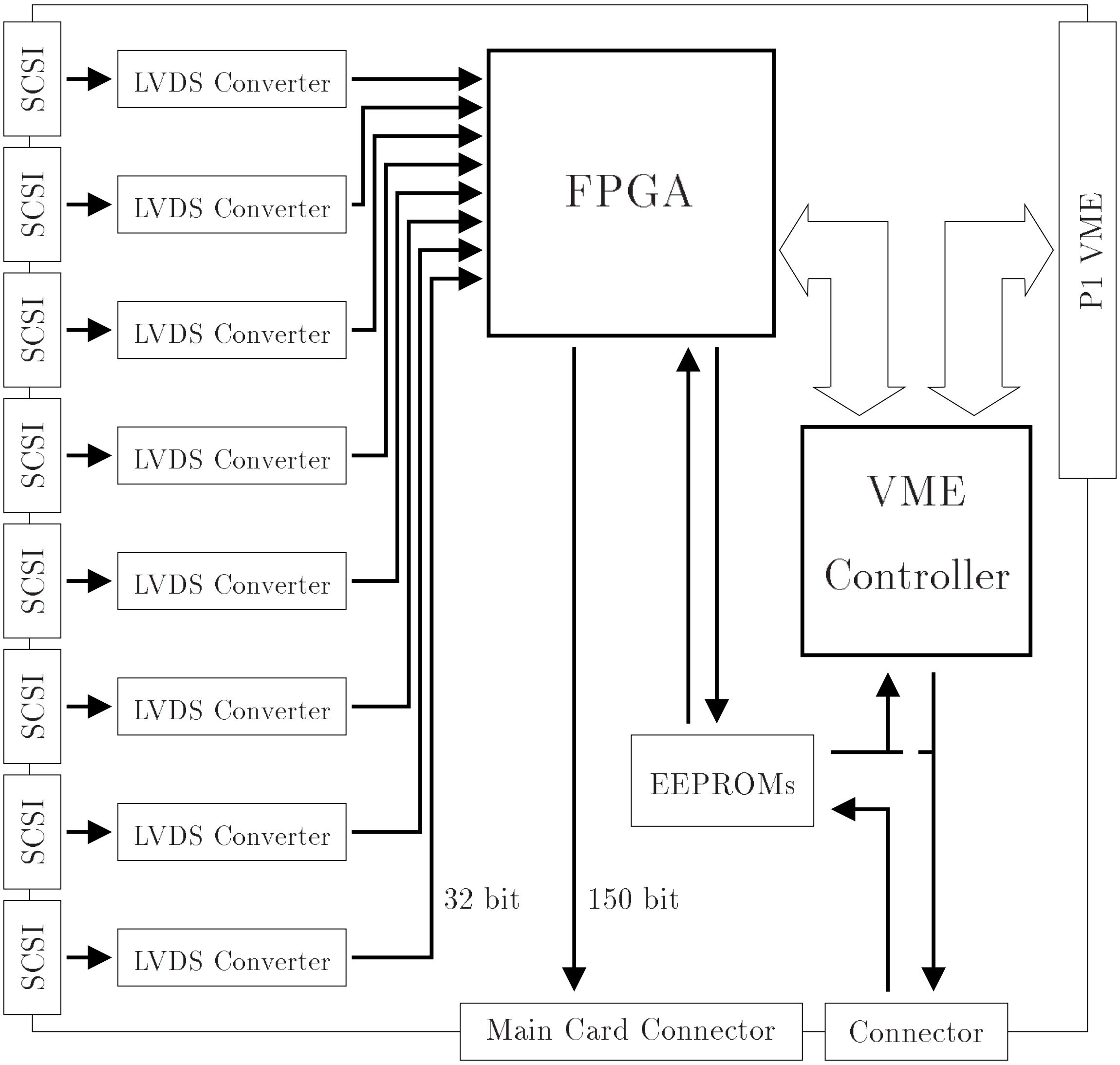


Figure 3.11: *CIP2000 sum-card. There are two pre-sum cards and one main-sum card planned. Both types of sum cards have one FPGA of the same type like the trigger cards. The pre-sum cards are connected to the trigger cards via eight SCSI connectors and differential cables. After the signals from the trigger cards are received on the pre-sum card they are converted back from LVDS standard. The FPGA demultiplexes the signals and adds up the histogram for eight sectors in ϕ. The pre-sum cards send the histogram consisting of 15 bins with numbers of the width of 10 bit via the 150 bit wide link to the main-sum card. The main-sum card adds the main histogram resulting in a 15 bin wide histogram with numbers of the width of 11 bit and builds the trigger elements to send them to the central trigger unit.*

a piggy-back card (piggy-back A) mounted on the control card. Another piggy-back (piggy-back B) card mounted on the control card makes signals used on the control card available on LED displays. The same signals are made available on lemo connectors on the front side of the control card. A Lattice ispL-1048 PLD is used as a VME controller and for the implementation of a cosmic trigger. This cosmic trigger combines the pad

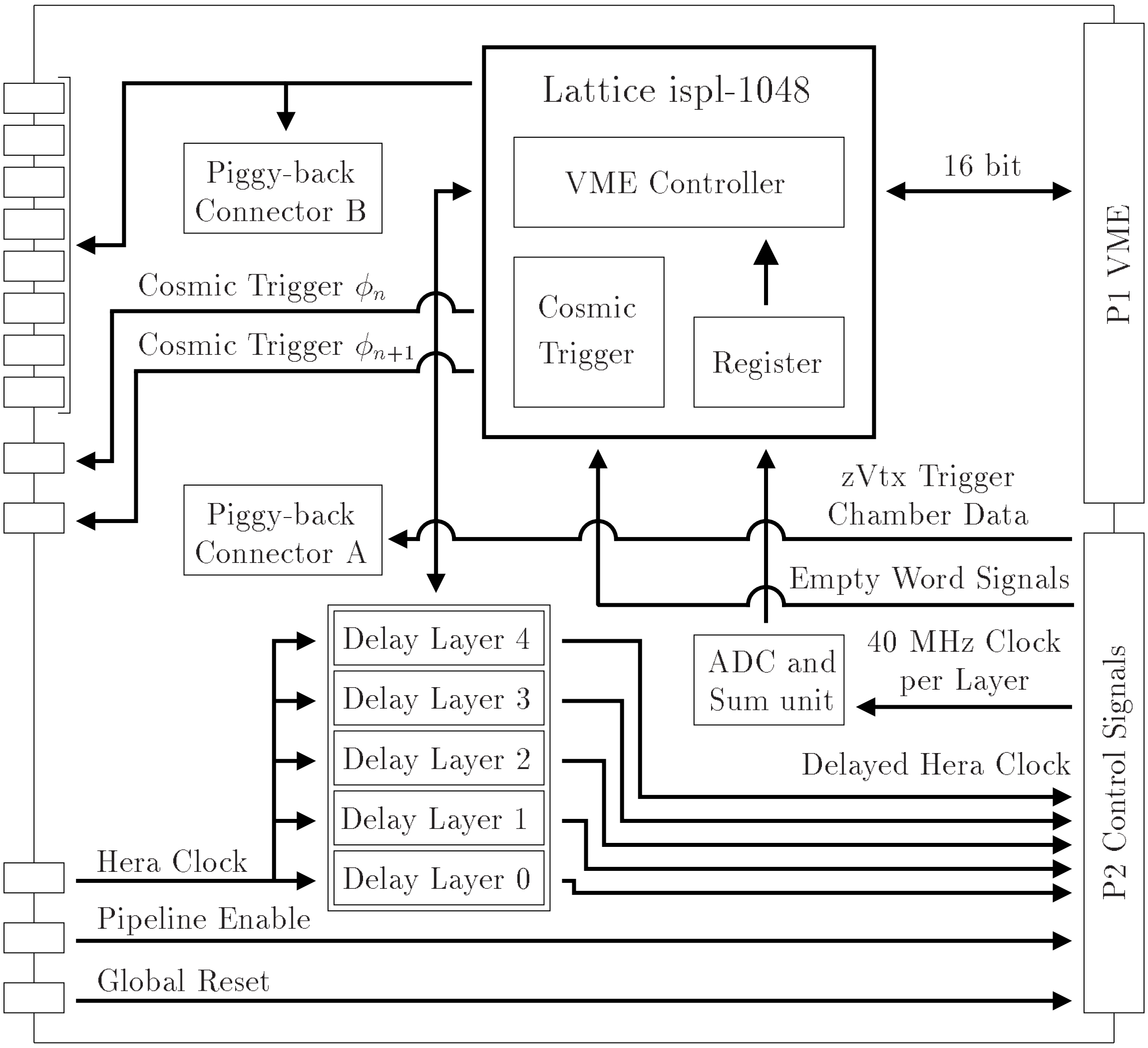
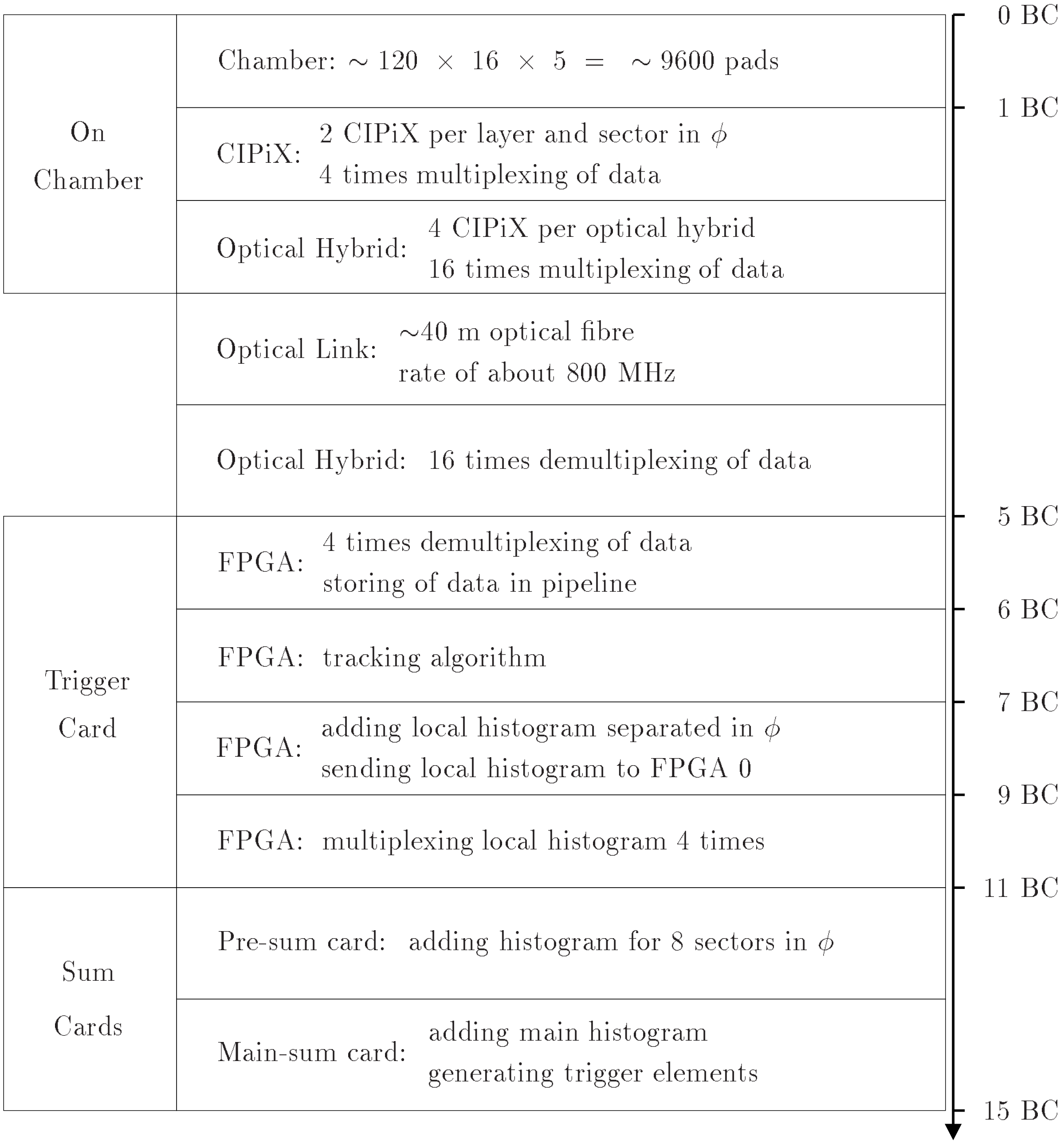


Figure 3.12: *CIP2000 control card. The Hera Clock signal, the Pipeline Enable signal and the Global Reset signal are distributed by the control card via the backplane to the trigger card. The Hera Clock can be delayed individually for every layer of the CIP2000 chamber. The delays can be set via the VME bus. The optical chamber readout cards use a 40 MHz clock generated from the Hera clock. This clock signal from every chamber layer is received on the control card. The phase of the signals is compared and made available in a register. This register can be read via the VME bus. The delays of the Hera clock can be adjusted depending on the value of the register. Chamber signals from the inner layers are needed for the zVtx trigger of the MWPC trigger system. These chamber data is made available in LVDS standard on a piggy-back card (piggy-back A) mounted on the control card. A cosmic trigger is implemented on the control card providing information if one pad of every layer in one sector in* ϕ *has been activated.*

signals (Empty Word Signal) of different layers and generates a cosmic trigger signal for both sectors in ϕ the control card is assigned to. The signal means that at least one pad of every layer has been activated.

3.4.4 Timing

Figure 3.13 shows the timing behavior of the CIP2000 trigger as it is derived from simulations. The time scale is plotted in units of bunch crossings (1 BC = 96 ns). In comparison to other detectors the chamber readout is relatively slow because of the multiplexing of the data. By running the trigger algorithm in parallel on multiple devices the time for the tracking algorithm can be kept short. The summing of the histograms how ever ia a time consuming procedure.



On Chamber	Chamber:	$\sim 120 \times 16 \times 5 = \sim 9600$ pads
	CIPiX:	2 CIPiX per layer and sector in ϕ 4 times multiplexing of data
	Optical Hybrid:	4 CIPiX per optical hybrid 16 times multiplexing of data
	Optical Link:	~40 m optical fibre rate of about 800 MHz
	Optical Hybrid:	16 times demultiplexing of data
Trigger Card	FPGA:	4 times demultiplexing of data storing of data in pipeline
	FPGA:	tracking algorithm
	FPGA:	adding local histogram separated in ϕ sending local histogram to FPGA 0
	FPGA:	multiplexing local histogram 4 times
Sum Cards	Pre-sum card:	adding histogram for 8 sectors in ϕ
	Main-sum card:	adding main histogram generating trigger elements

Figure 3.13: *Timing of the CIP2000 trigger. The time scale is ploted in units of bunch crossings (1 BC = 96 ns).*

Chapter 4

The CIP2000 trigger control system

4.1 CIP2000 trigger system control structure

The control system for the CIP2000 trigger has to connect and manage the communication between the different parts of the system. Three main parts have to be considered. The trigger and readout system, the high voltage system and the system to control the CIPiX chips. Figure 4.1 shows an overview of the planned system. The CIP2000 main CPU will run the main control system software that can be accessed via a firewall from the internet. A program displaying the status of the system can be run from the whole internet. A program to control the system can be run just on special machines or on a local computer in the control room. The CIP2000 main CPU is connected via a VIC (**V**ertical **I**nter**c**onnect) link to the Main Trigger CPU to control the trigger and readout system. A serial RS232 link to a processor communicating via the I^2C protocol with the CIPiX chips manages the control and setup of the CIPiX chips. A connection to the high voltage system of the CIP2000 chamber will be established from the CIP2000 main CPU. An ethernet connection to all Trigger CPUs manages the booting of the CPUs. The trigger CPUs are connected via a pVIC (**PCI** **V**ertical **I**nter**c**onnect) link. The trigger CPUs communicate with the trigger cards via the VME bus. The STC system is connected to the main trigger CPU via a VIC link to manage interrupts from the STC system and to set and read special registers in the STC cards. All connections in the trigger and readout system will be explained in detail in Chapter 5. Fast signals like the Hera clock signal are transfered directly from the STC system to the control cards in the trigger crates. The control cards are connected to the trigger cards via the backplane to provide trigger control signals.

4.2 STC system

The STC (**S**ubsytem **T**rigger **C**ontrol) system manages the communication between the subsytems and the CTC (**C**entral **T**rigger **C**ontrol). It provides all necessary signals from the CTC and sends all needed signals back. A STC system consist of a STC Fast Card, a STC Slow Card, one or more STC Fanout Cards and a device that handles the VME

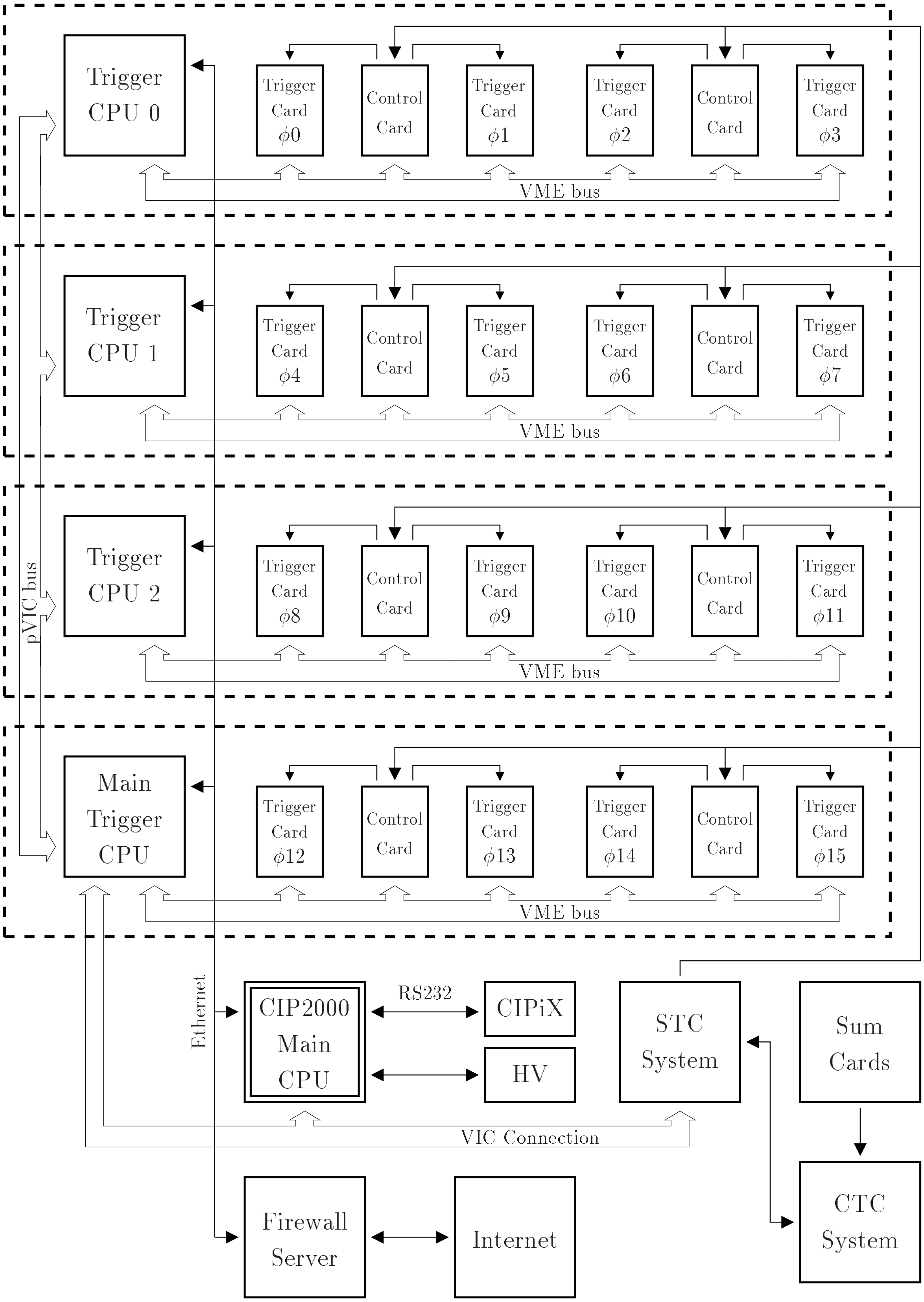


Figure 4.1: *CIP2000 trigger system control structure*

interrupts of the STC Slow Card. This device is normally a CPU that is housed in the STC crate. A Trigger Bit Card can be added to the STC sytem. The STC crate has a special VME/STC backplane. The backplane provides A24/D16 VME transfer. Figure 4.2 shows an overview of the connections between the different parts of the STC system and the CTC system. The subsystem trigger unit builds trigger elements from the detector information. The trigger elements are sent to the CTC L1 trigger logic. The CTC forms subtriggers from the information of various subsystems and makes a L1 decision. This decision is sent to the STC system in form of the L1 Keep signal. The STC system sends the L1 Keep signal to the parts of the subsystem where the pipelines are stopped. L2 information is sent from the subsystems to the L2 trigger and if the L2 decision is positive the readout of the detectors begins. Events that are not triggered are overwritten in the pipelines. The STC system can operate in different modes. It does not have to be connected to the CTC system as it is able to generate the clock signal and trigger decisions also autonomously. These modes are needed for standalone tests of a system during debugging or development phases.

The different parts of the STC system have different tasks. The Fast Card simply receives the fast output signals of the CTC and sends them on the STC backplane. The Fanout Card transfers the signals to the rest of the subsystem where the signals are needed and can transfer informations back from the different parts of the subsystem. The Fast Card can run in different autonomous modes where the clock and trigger decisions are generated in the card. It provides scalers to count information like e.g. L1 Keeps and bunch crossings. The Slow card receives interrupt requests and passes them through to the STC CPU via VME interrupt cycles. The Trigger Bit Card receives information from the CTC about which triggers have detected a physics event. The information can be used to calculate informations for the L2 trigger and for cross checks of the subsystem. An overviw of the STC system can be found in [19].

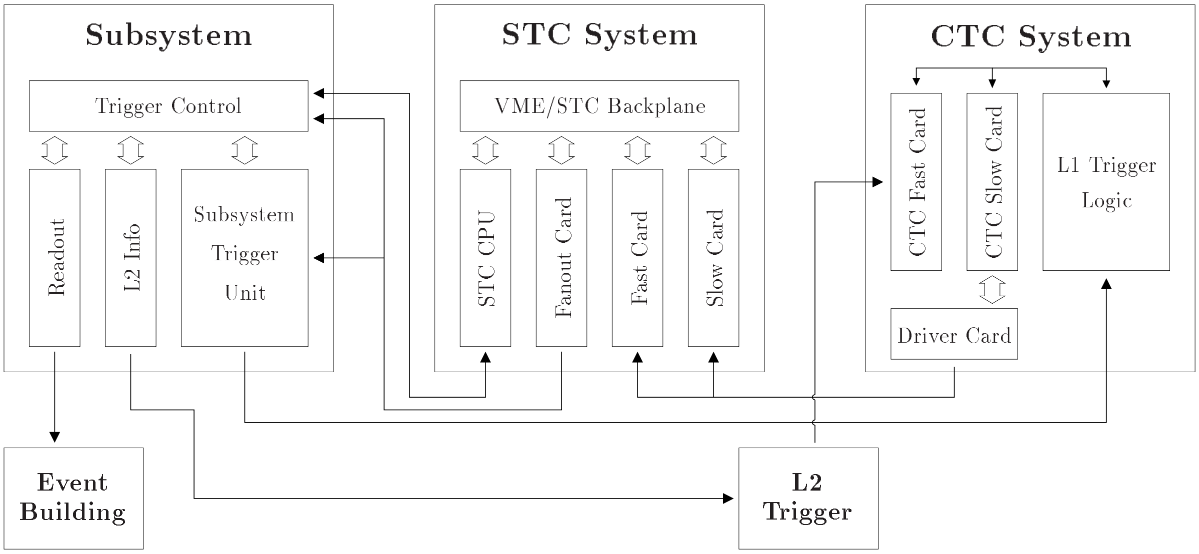


Figure 4.2: *Structure of a standard STC system*

4.2.1 Trigger signals

Several signals are used for the communication between the STC system and the CTC system. Table 4.1 gives a short overview of the different signals. Signals coming from the CTC and going to the STC are called outward signals. Signals generated in the STC and send to the CTC are called inward signals. Fast signals are synchronous to the HERA clock. Slow signals generate interrupt cycles on the VME bus, or the status of the signal can be read or set in special registers on the STC cards. Slow signals are not synchronous to the HERA clock.

Direction	Signal type	Trigger signal	Shortcut
CTC → STC	Fast Signals	Hera Clock	`HCk`
		Pipeline Enable	`PEn`
		L1 Active	`L1Atv`
		L1 Keep	`L1Kp`
		L2 Keep	`L2Kp`
		Fast Clear	`FsClr`
		First Bunch	`FrsBu`
		Filled Bunch	`FillBu`
		Run	`Run`
	Slow Signals	All Front End Ready	`AFER`
		L2 Keep	`L2Kp`
		L3 Keep	`L3Kp`
		L3 Reject	`L3Rej`
		Prepare Run	`PreRun`
		Terminate Run	`TermRun`
		Subsystem Timeout	`SubSyto`
STC → CTC	Slow Signals	Front End Ready	`FER`
		L3 Keep Acknowledge	`L3KpAck`
		L3 Reject Acknowledge	`L3RejAck`

Table 4.1: Trigger signals

Hera Clock (`HCk`): This fast signal is the most important signal distributed by the CTC because all fast signals are synchronized to the HERA clock. The frequency of the Hera clock is $f_{HCk} = 10.40975$ MHz. The time period between two *ep* interactions is $\Delta t_{BC} = 96.0638$ ns. This period of time is named one bunch crossing (BC).

Pipeline Enable (`PEn`): This fast signal shows the subsystem when to start and stop the pipeline for data storage. After a L1 Keep the Pipeline Enable signal drops. It is reactivated after a Fast Clear signal.

L1 Active (`L1Atv`): This fast signal shows the subsystem if the L1 trigger is ready to process trigger decisions. After a L1 Keep the L1 Active signal drops. It is reactivated a programmable number of bunch crossings after the Pipeline Enable Signal (current value: 145 BC).

L1 Keep (`L1Kp`): This fast signal is sent by the CTC after a positive decision of the central L1 trigger decider. After the L1 Keep signal the pipeline is stopped and the subsystem waits for a L2 trigger decision while it prepares the readout.

L2 Keep (`L2Kp`): This signal is sent by the CTC as a fast and a slow signal after a positive decision of the L2 trigger. When a L2 Keep signal is broadcasted the Frontend Ready signal distributed by the subsystem will drop and the readout starts.

Fast Clear (`FsClr`): The Fast Clear signal restarts the cycle of data taking. Three conditions can force a Fast Clear signal:

- All Frontend Ready AND All L3 Keep Acknowledge
- All Frontend Ready AND L3 Reject
- L2 Reject

1 BC after the Fast Clear Signal the Pipeline Enable signal is broadcasted. The L1 Active signal rises 145 BC after the Fast Clear.

First Bunch (`FrsBu`): The maximum number of bunches that can be stored in the accelerator storage rings is 210. The fast signal First Bunch defines the bunch that is counted as the first bunch.

Filled Bunch (`FillBu`): Not all of the possible 210 bunches in the storage rings are filled with particles. The fast signal Filled Bunch gives the option to just trigger the events that come from filled bunches.

Run (`Run`): This fast signal is active when a run and normal data taking is going on.

All Front End Ready (`AFER`): This slow signal is broadcasted by the CTC to inform the subsystem that all other subsystems are ready for a new data take cycle. After the All Frontend Ready signal the Fast Clear signal is broadcasted if all subsystems have acknowledged the L3 Reject or L3 Keep signal. The All Frontend Ready signal is an AND of all Frontend Ready signals of all subsystems.

L3 Keep (`L3Kp`): This slow signal is sent by the CTC after a positive decision of the L3 trigger logic. This can be any time between L2 Keep and L3 timeout. It strictly excludes the L3 Reject signal. The signal means that the event is now allowed to be sent to the central DAQ. For the HERA 2000 upgrade a L3 software trigger is planned.

L3 Reject (`L3Rej`): This slow signal is sent by the CTC at least 10 μs after the L2 Keep signal and earlier than 800 μs after the L2 Keep (L3 timeout). L3 Reject and L3 Keep exclude each other strictly. When the L3 Reject signal is received by the subsystem the readout of the data has to be stopped and the data has to be deleted.

Prepare Run (`PreRun`): The slow signal Prepare Run tells the subsystem to prepare all the different parts of the subsystem for normal data taking and to write the Runstart Record.

Terminate Run (`TermRun`): The slow signal Terminate Run ends the run and gives the subsystem the opportunity to clean up their system units and to write the Run End Record.

Subsystem Timeout (`SubSyto`): This slow signal is sent if the data taking in the event building is not possible or a subsystem does not respond to the CTC.

Front End Ready (`FER`): This signal is sent by the subsystem to the CTC when the hardware is ready to accept a new L1 Keep and the pipeline is prepared for new events by deletion of all old events. At the beginning of a run the Frontend Ready signal is sent when the subsystem is ready for the first L1 Keep.

L3 Keep Acknowledge (`L3KpAck`): This signal is sent by the subsystem to the CTC. It shows that the software of the subsystem has taken note of the L3 Keep.

L3 Reject Acknowledge (`L3RejAck`): This signal is sent by the subsystem to the CTC to indicate that it has noticed the L3 Reject. If the subsystem handles L3 Rejects it sends the signal after finishing its interrupt routine.

4.2.2 Trigger signal timing

Three different possibilities of timing procedures are possible in a normal run situation without an abort of the run or other special cases. The L2 Reject, the L2 Keep with a L3 Reject and with a L3 Keep are possible. Figure 4.3, 4.4 and 4.5 illustrate the different procedures.

Trigger signal timing: L1 Keep, L2 Reject

1. L1 Keep signal switches Pipeline Enable and L1 Active to low.
2. Pipelines stop. Subsystem prepares Readout. Waiting for L2 Keep.
3. L2 Reject (Fast Clear) switches L1 Keep to low. Pipeline Enable rises to high.
4. After 145 BC L1 Active rises to high.
5. System waits for new event.

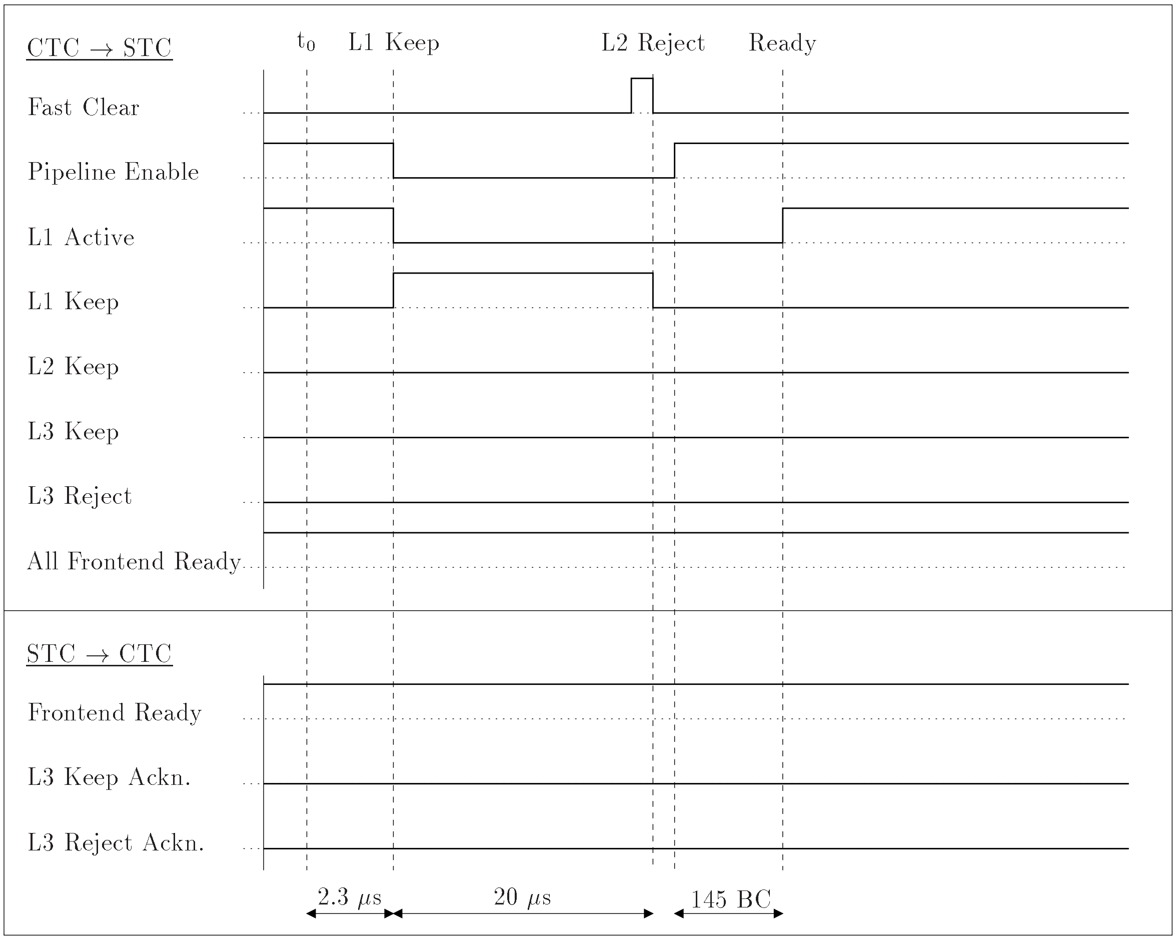


Figure 4.3: *Trigger signal timing – L1 Keep, L2 Reject*

Trigger signal timing: L1 Keep, L2 Keep, L3 Reject

1. L1 Keep signal switches Pipeline Enable and L1 Active to low.
2. Pipelines stop. Subsystem prepares Readout. Waiting for L2 Keep.
3. L2 Keep signal starts readout.
4. Front End Ready and All Front End Ready go low.
5. L3 Reject signal stops readout.
6. After recovering from readout abort subsystem sends Front End Ready signal.
7. CTC sends All Front End Ready after it receives Front End Ready from subsystems.
8. Subsystem sends L3 Reject Acknowledge.
9. Fast Clear resets L1 Keep, L2 Keep and Pipeline Enable.
10. After 145 BC L1 Active rises to high. System waits for new event.

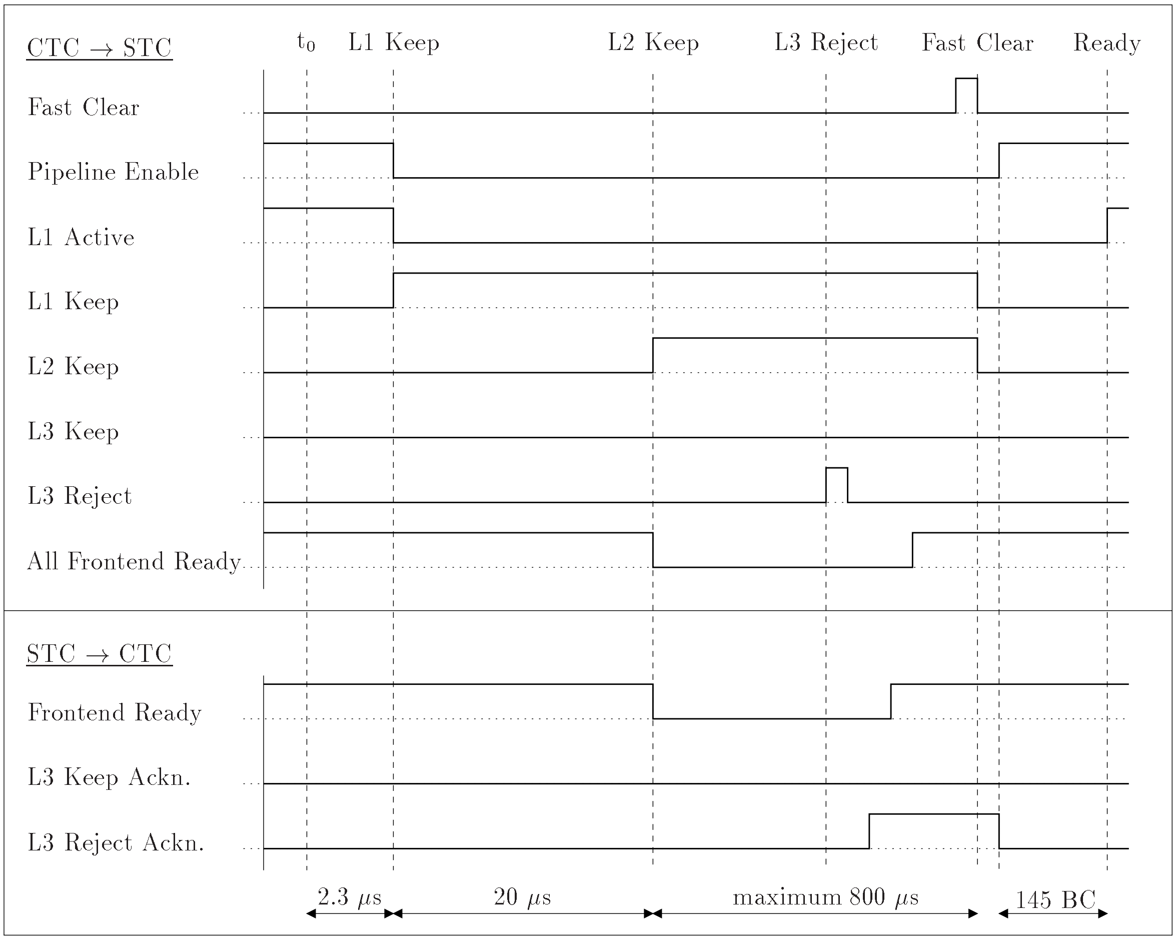


Figure 4.4: *Trigger signal timing – L1 Keep, L2 Keep, L3 Reject*

Trigger signal timing: L1 Keep, L2 Keep, L3 Keep

1. L1 Keep signal switches Pipeline Enable and L1 Active to low.
2. Pipelines stop. Subsystem prepares Readout. Waiting for L2 Keep.
3. L2 Keep signal starts readout.
4. Front End Ready and All Front End Ready go low.
5. L3 Keep validates readout. L3 Keep Acknowledge is sent to CTC.
6. After readout is finished Front End Ready signal is sent.
7. CTC sends All Front End Ready after it receives Front End Ready from subsystems.
8. Fast Clear resets L1 Keep, L2 Keep and Pipeline Enable.
9. After 145 BC L1 Active rises to high.
10. System waits for new event.

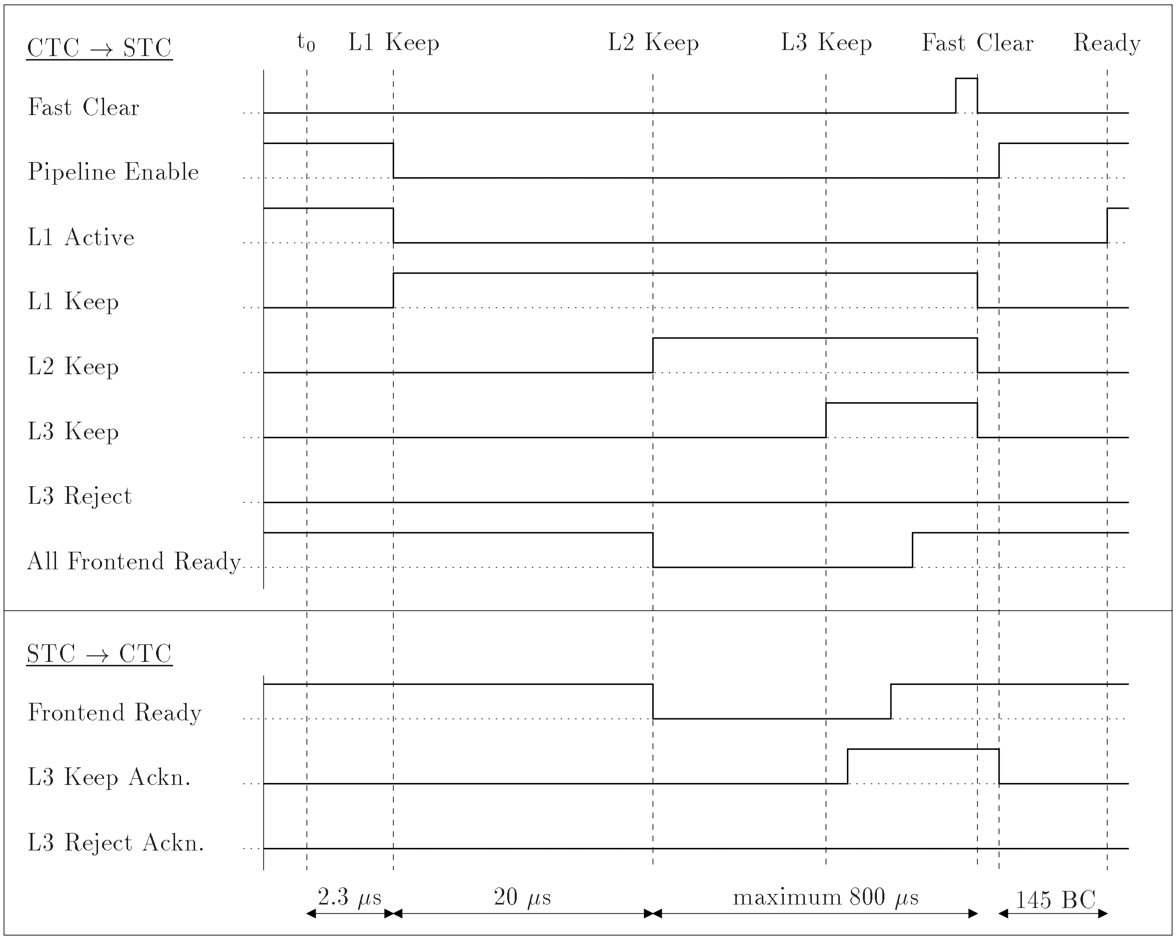


Figure 4.5: *Trigger signal timing – L1 Keep, L2 Keep, L3 Keep*

4.2.3 STC cards

All STC cards are supposed to be housed in a special VME crate. The backplane connection P1/J1 conforms fully to VME standard, while P2/J2 does not. The P2/J2 connection consists of a four row 128 contact connector. It is actually five rows wide with the central row empty. The cards require the non VME ECL supply voltage of -5.2 V. The voltage is fed in through the bottommost contacts of P2/J2. Information on the different cards can be found in [20] (Fast Card), [21] (Slow Card) and [22] (Fanout Card).

Fast Card

The Fast Card is described in [20]. It is connected via a multiconductor twisted pair cable to the CTC. During normal DAQ it simply receives the fast outward signals and passes them on the P2/J2 backplane of the STC crate, where they may be connected via wire wrap to the Fanout Cards. In addition there are some functions in the Fast Card which are not simply passive and serve useful purposes, the `FER` function, scalers and local L1 and L2 trigger logics to generate local L1 and L2 decisions without a connection to the CTC.

Modes There are five basic modes of operation, numbered from 0 to 4. They mean different levels of independence from the CTC.

0: **Submissive**. Normal mode during correlated data taking in the entire detector.
1: **Data Autonomous - L1/L2 Submissive**. The Fast Card receives the clock, L1 and L2 info and the machine related signals from the CTC. The subsystem data will be stored locally and will not go to the event building.
2: **Data and L2 Autonomous - L1 Submissive**. Similar to mode 1 but the STC may do its own L2 decision and timing.
3: **Trigger Autonomous - Clock Submissive**. The STC receives just the HERA clock from the CTC. Trigger actions in the STC are completely independent from the CTC.
4: **Clock Autonomous**. Same as mode 4 but a local quartz oscillator generates the HERA clock in the STC system.

HERA clock circuits The central HERA clock is available via a differential pair of the outward cable or via a BNC connector. In addition a quartz oscillator on the card can generate the HERA clock. A clock monitor circuit checks the clock for errors.

Local L1 logic Every local trigger in the trigger logic has an allocated flip flop which is set in synchronism with the HERA clock if the trigger and gating conditions are fulfilled. The gating conditions can be the L1 Active signal, the Filled Bunch signal or an external input. The flip flops may be read via VME and have an output on the P2 backplane. In mode 3 and 4 three local L1 triggers can be operated. Two different detector triggers intended to accept trigger signals from the subsystems own detector, and an auto-synchronizing test trigger. The local detector trigger can be scaled down not triggering every event. The flip flop of the unscaled trigger has a rear output on the P2 backplane. The gating condition for the triggers are the `L1Atv` signal, the `Run` signal and the `FillBu` signal if needed. The auto-synchronizing test trigger is intended to be set by pulse generators through front or rear inputs or by a bus trigger.

Local L2 control In mode 2 to 4 the subsytem can make its own L2 decisions, either upon the L1 Keep events offered by the STC in mode 2 or upon its own L1 Keeps. The Fast Card does not contain any L2 deciders. A clocked hardwired L2 decider has to be located somewhere not too far away. The Fast Card senses the output level of the decider a presettable number of `HCk` cycles after L1 Keep. Depending on the status of this level the Fast Card takes either the L2 Reject turn or continues into L2 Keep. If there is no local decider a L2 decision may be forced into both ways by writeable control bits.

`FER` logic There are two possible sources for the `FER` signal. Either a flip flop on the Fast Card or an external signal, e.g. the logical AND of many status signals of readout electronics. The selection between these modes is done by a control bit `FER` mode, which may be set through VME. An automatic setting of the `FER` level independent from any data readout can be done for test purposes. A 16 bit number written to a register gives the set delay after the L2 Keep in HERA clock cycles. A one `HCk` long `FER` pulse is available on P2 after `FER` has gone from false to true. This pulse can be used to reset the local L1 flip flops in mode 1 to 4.

Restart logic When the handling of an event is finished by the L2 Reject signal or by the `FER` signal becoming true again, the fast signals must be switched to their normal state in a defined way. In mode 0 this is done by the CTC. For the other modes there is a restart logic on the Fast Card.

Scalers The Fast Card contains several scalers that can be read slowly via the VME bus at a certain time while the scalers continue counting. The following scalers are available: a 8 bit bunch scaler, a 32 bit revolution scaler gated by the `FrsBu` signal, a 40 bit all crossing scaler and a 40 bit crossing during `L1Atv` scaler.

Slow Card

The Slow Card is described in [21]. The main job of the Slow Card is to receive interrupt requests and passing them through VME interrupt cycles to the subsystem processor. The Slow Card is connected to the CTC via a multiconductor flat cable connected to the P2/J2 backplane. This connection follows the RS423 standard. 8 full and 1 reduced interrupt channel are connected through a wire-wrap matrix and three priority encoders to VME interrupters. The interrupt requests can come from the CTC or from a VME write cycle. Two 32 bit gated scalers with common reset for counting L1 Keeps and L2 Keeps are available. A four channel outward information bit connection to the CTC gives information about the status of the CTC.

Information bit logic Four information bits are received from the CTC which can be made available by a local wire wrapping on the P2 connector. Their status can be read from the VME bus and is indicated by front panel LEDs.

Scaler logic There are two 32 bit synchronous and gated scalers on the card for counting the L1 Keep and the L2 Keep. Their clock, gate and common reset inputs are accessible through the P2 connector. The contents of the scalers can be read via VME.

Interrupt request logic Interrupt requests coming from the CTC or a VME write cycle set the appropriate interrupt flip flop. There are eight identical interrupt channels. Eight signals coming from interrupt flip flops and one received from the Fast Card L2 Keep flip flop are logically ANDed with their mask register and can be connected by the user via wire wrapping to one of three priority encoders. Each of these encoders has eight inputs. The OR output of every encoder is connected to the driver of one IRQ line on the VME bus. During the VME acknowledge cycle an interrupt vector (status byte) is placed on the data lines.

Fanout Card

The Fanout Card is described in [22]. The Fanout Card accepts six different outward signals from the STC‘s Fast Card or from other signal sources. It passes them on to the subsystem electronics. The signals can be manipulated, e.g. delayed, disabled, simulated, etc. in the Fanout Card. From the internal logics the signals are fanned out fivefold to identical cable ports and to front NIM outputs. Each cable port contains two differential inputs for inward signals. The inward signals are available at the backplane connector P2/J2 in high level form. A lot of features of the card can be controlled and read via the VME bus.

Outward fast signal logic The outward fast signals (mainly `HCk` and `PEn`) can be manipulated. VME programmable delays can delay the `HCk` and the `PEn` signal. A `PEn` Afterrun logic makes it possible to delay the true to false transition of the `PEn` signal for a number of clock phases. This logic can be programmed via the VME bus. A Gated Clock logic can mix the `HCk` and the `PEn` signal to produce a clock wave that lasts as long as `PEn` is active. The Artificial Signal logic can simulate the `PEn` signal independent from the inputs. The duration of the signal can be programmed via the VME bus and the puls can be triggerd via `NIM` input or the VME bus. The signals will be synchronised to the `HCk` signal. The outputs of this logic can be mapped via jumpers in different ways on the cable ports and `NIM` outputs.

Outward slow signal logic The outward slow signals can be delayed the same way as the fast signals. An Artificial Signal logic can simulate the slow signals in a same way like the `PEn` signal. The outputs of the logic can be mapped on different outputs on the cable ports and the `NIM` outputs.

Inward signal logic The inward signals coming from the subsystem trigger electronics are available on the backplane connector P2/J2 or can be read via the VME bus. A connection test logic can test if a connection is broken. The test logic can be controlled via the VME bus. A logical AND programmed via the VME bus can be formed on the inward signals. The AND can be extended over several cards.

4.3 CIP2000 trigger STC and readout control system

The main control unit of the trigger and readout part of the CIP2000 trigger system is the main trigger CPU. It manages the interrupt handling and the writing and reading of registers in the STC cards via a VIC link to the STC crate. The communication between the different trigger CPUs is implemented via the pVIC link. The software running on the CPUs uses a state machine structure switching from state to state depending on control registers set by the main trigger CPU. The different states of the CPUs are called modes. Every trigger CPU has a mode register showing other CPUs the present mode it is working in. A remote register that can be set from the main CPU shows the trigger CPU what action to take and what mode to switch to. The trigger cards are controlled by the trigger CPUs via the VME bus and by the control cards sending the fast signals from the STC system. The trigger cards have mode and remote registers accessable via the VME bus implemented in the FPGAs. The FPGAs run in a state machine structure similar to the trigger CPUs. The main trigger CPU has a mode and remote register that can be accessed by the CIP2000 main CPU via a VIC connection. Actions like e.g. run aborts are handled here. The only fast signals needed for the CIP2000 trigger system are the Hera clock and the Pipeline Enable signal. These signals are deliverd from the STC fast card via the fanout card to the control cards. The control cards can delay and synchronize the signals before they are sent via the backplane to the trigger cards. The CIP2000 STC system will consist of a slow card, a fast card and one or two fanout cards depending on how the fast signals are delivered to the control cards. The sequence of the different signals and the different steps in the readout in the CIP2000 trigger system are explained in Chapter 5.

Chapter 5

The CIP2000 trigger readout system

5.1 Readout system structure

The readout system of the CIP2000 trigger mainly consists of components already used at the H1 experiment. A part of the readout system structure is shown in Figure 5.1. Four trigger CPUs are managing the data readout from the pipelines on the trigger cards. One trigger CPU has to handle the additional tasks of the communication to the STC and CIP2000 main CPU and to handle the readout to the VMEtaxi card. This CPU is called main trigger CPU. The VMEtaxi card is the link to the optical readout ring of the H1 experiment. Different bus standards are used in the whole trigger readout system. The common bus systems in the H1 experiment are the VME bus and the VSB bus. VIC links connect the VME busses of different crates. A pVIC link builds the connection between the different trigger CPUs. It is based on the local PCI bus structure of the trigger CPUs. Section 5.3 explains the different components and links of the readout system.
The pipelines containing the chamber data are implemented in FPGAs mounted on the trigger cards. The same FPGAs run the trigger algorithm. A certain event window consisting of 3 to 5 events is read from the pipeline. The event window is centered around the designated actual triggered event. It is necessary to read out the whole event window to observe the timing behaviour of the trigger and to make sure that the actual triggered event is read out. The information of the full event window could be used to detect a pile up of events in the LAr calorimeter. The events are stored in a readout register in the FPGAs independent of the pipelines. A VME controller on the trigger card maps the readout register on the VME bus. The trigger CPUs can read out the events via the VME bus in a VME block mode. The trigger CPUs collect the event data and transfer the data to the main trigger CPU. The main trigger CPU merges the data of the different trigger CPUs and sends it to the VMEtaxi card. A data reduction algorithm will be applied on the data by the trigger CPU or by the main trigger CPU. The software on the trigger CPUs and on the main trigger CPU will have the ability of storing and handling different events triggered at different times in a multi event buffer structure. The readout of the data from the pipelines via the VME bus to the trigger CPUs is called horizontal readout. The readout of the data from the CPUs to the main trigger CPU is called vertical readout. The trigger CPUs communicate with the main trigger CPU by the main trigger

CPU setting remote registers in the trigger CPUs forcing the trigger CPUs to switch into a different state. A mode register in the trigger CPUs shows the main trigger CPU the actual state of the CPUs. This communication is done via the pVIC link. A similar communication exists between the trigger CPUs and the trigger cards. The trigger cards have registers called mode and remote registers to show the status of the trigger cards and to force the trigger cards to switch to a different status. This communication is achieved via the VME bus. The main trigger CPU can communicate with the STC system via a VIC link to the STC crate. Interrupts can be passed through to the main trigger CPU and registers and scalers in the STC cards can be read and modified.

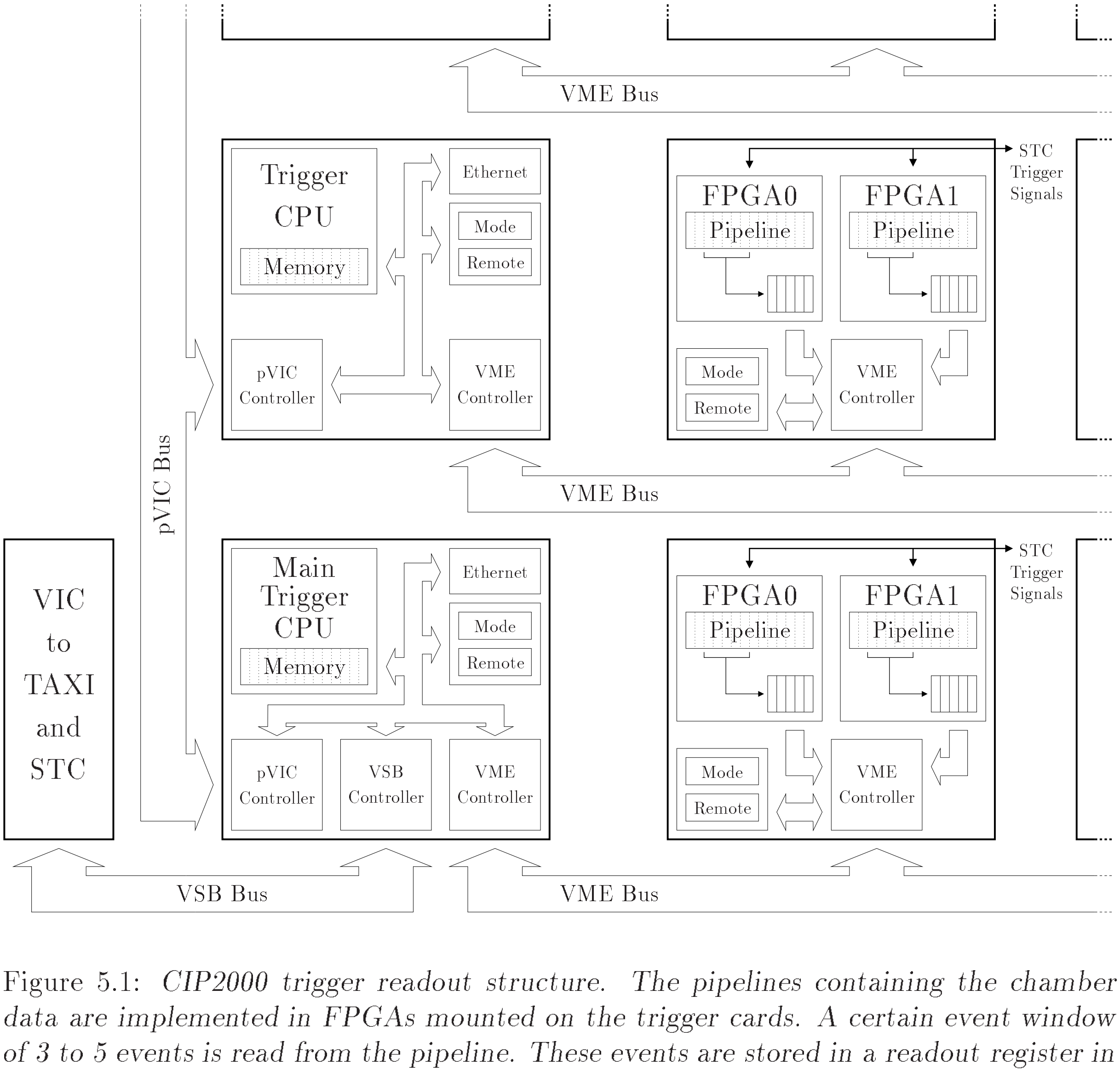


Figure 5.1: *CIP2000 trigger readout structure. The pipelines containing the chamber data are implemented in FPGAs mounted on the trigger cards. A certain event window of 3 to 5 events is read from the pipeline. These events are stored in a readout register in the FPGAs independent from the pipelines. The trigger CPUs can read out the events via the VME bus in a VME block mode. The trigger CPUs collect the event data and transfer the data to the main trigger CPU. The main trigger CPU merges the data of the different trigger CPUs and sends it to the VMEtaxi card.*

5.2 Readout system timing

Figure 5.2 illustrates the different parts of the readout timing. The only fast trigger signals delivered to the trigger cards are the HERA clock and the Pipeline Enable signal. An overview of all trigger signals can be found in Chapter 4. The L1 Keep action is initialized by a falling edge of the Pipeline Enable signal. After the L1 Keep is recognized by the trigger card the pipelines in the FPGAs are stopped and the event window is copied into the readout register. The trigger cards delete the pipelines and show the CPU that the data in the readout registers is valid by setting the mode register. The system waits for the L2 Keep or L2 Reject signal.

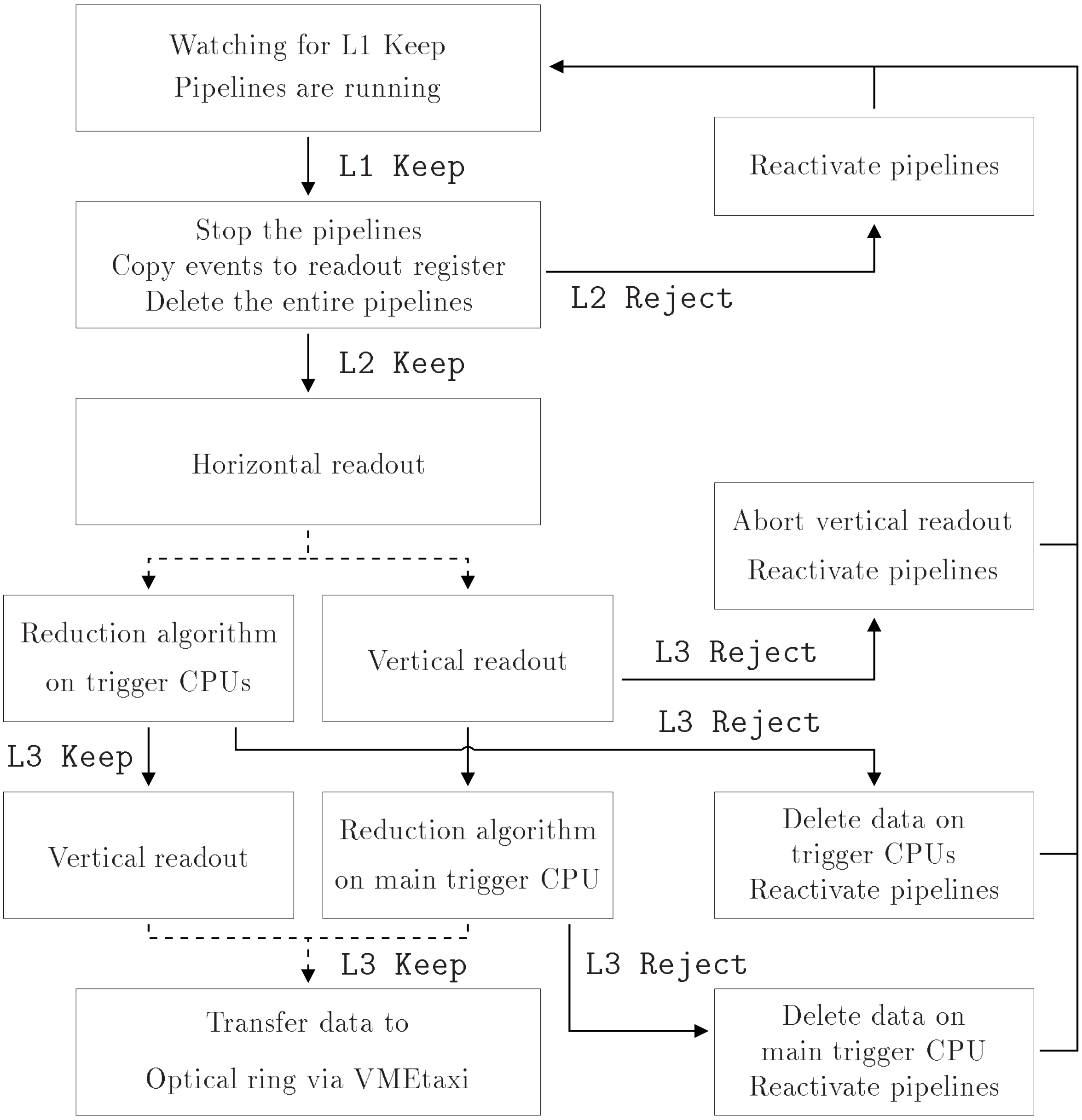


Figure 5.2: *CIP2000 trigger readout timing*

The L2 Reject action is forced by a rising edge of the Pipeline Enable signal. The pipelines are reactivated and the trigger card shows the reject of the event to the trigger CPUs by setting the mode register. The L2 Keep is recognized by the main trigger CPU from an interrupt cycle coming from the STC slow card or by reading a L2 Keep status bit in the STC fast card. The main trigger CPU forces the trigger CPUs to start the horizontal readout by setting the remote registers in the trigger CPUs. The data from the readout registers is now in the memory of the trigger CPUs.

Two different ways of continuing the readout are possible and have to be examined. To reduce the size of the read out data a reduction algorithm has to be applied to the data in the trigger CPUs or in the main trigger CPU. If the reduction algorithm will be applied to the data in the main trigger CPU the data has to be transfered directly to the main trigger CPU without waiting for the L3 Keep or L3 Reject signal. If the L3 Reject is received by the main trigger CPU the vertical readout or the reduction algorithm is aborted, the data is deleted and the pipelines are reactivated. If the L3 Keep signal is received the data is transfered to the VMEtaxi card via the VSB and VIC link. If the data reduction algorithm is applied on the data in the trigger CPUs the system waits for the L3 Keep or L3 Reject signal before the data is transfered to the main trigger CPU, collected and transfered to the VMEtaxi card via the VSB and VIC link. If a L3 Reject signal is received the data is deleted and the pipelines are reactivated. The Frontend Ready, L3 Keep Acknowledge and the L3 Reject Acknowledge signals are set by the main trigger CPU in registers of the STC cards. Other slow signals are handled too in the main trigger CPU.

Every event stored in a pipeline can be read out by a block of 10 VME D32 read cycles. There is one pipeline in every FPGA on the trigger card. 8 pipelines have to be read out for the horizontal readout. The size of the overall data of one event read out in the horizontal readout is 320 byte. In the vertical readout the data in the three trigger CPUs have to be transfered to the main trigger CPU. The data in the trigger CPU part of the main trigger CPU is transfered locally. The size of the overall data of one event read out in the vertical readout is 960 byte. A data transfer rate of 10 Mbyte/s for the VME block mode and the pVIC connection is assumed. The duration of the horizontal and vertical readout for different event window sizes can be calculated from this assumption. The results are summarized in Table 5.1. Chapter 6 shows the results of the experiments made to measure the performance of the horizontal and vertical readout in the actual system.

Event window size[# of events]	Horizontal readout $t_{horizontal}[\mu s]$	Vertical readout $t_{vertical}[\mu s]$
1	30.52	91.55
3	91.56	274.66
5	152.59	457.76

Table 5.1: Assumed duration of the horizontal and vertical readout for a data transfer rate of 10 Mbyte/s for the horizontal and vertical readout.

5.3 Readout system hardware

5.3.1 Connection of CPU and trigger card: VME

The VME bus (**V**ersatile **M**odule **E**urope) is a flexible open-ended bus system widely used as a standard in the H1 experiment which makes use of the Eurocard standard. It was introduced by Motorola, Phillips, Thompson, and Mostek in 1981. VME bus was intended to be a flexible environment supporting a variety of computing intensive tasks. It is defined by the IEEE 1014-1987 standard. A documentation can be found in [14] released by the VITA (**V**MEbus **I**nternational **T**rade **A**ssociation) association.
The bus usage offers a completely memory mapped scheme. Every device can be viewed as an address, or block of addresses. Addresses and data are not multiplexed. The bus allows multiple masters and contains a powerful interrupt scheme. A resource manager is required to handle the interrupts. The VME bus uses a TTL based backplane which, although the system is asynchronous, can obtain a data transfer speed of approximately 10 Mbytes per second.

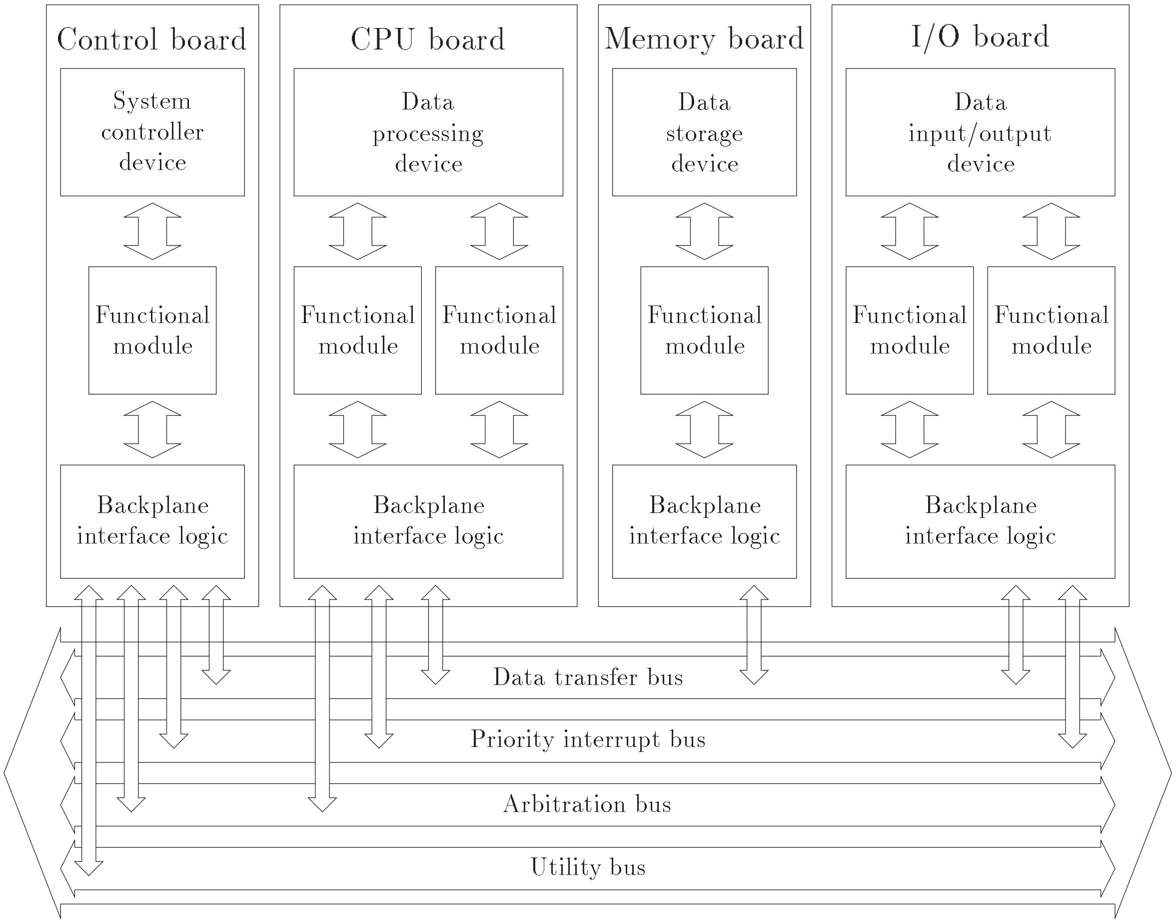


Figure 5.3: *Structure of VME functional modules and buses*

A typical transfer consists of an arbitration cycle to gain bus control, an address cycle to select the register to access and the actual data cycle. Read, write, modify and block transfers are supported. The VME bus system consists of 4 sub buses: the data transfer bus, the arbitration bus, the priority interrupt bus and the utility bus. The data transfer is asynchronous supporting modules with a broad variety of response times. The system is modular and follows the Eurocard standard. Standard VME crates contain 21 slots. The first slot must be used as a crate manager. The usual card sizes are 160 × 216 mm and 160 × 100 mm. Cards of both sizes can be mixed in the same crate. The smaller cards are capable of 8 or 16 bit data transfers. The larger cards can perform 8, 16, 32 or even 64 bit data transfers and can also support a larger address range of 4 Gbytes memory for A32 access instead of 16 Mbytes of memory for A24 access. All VME boards have a P1 connector. Larger cards may have an optional P2 connector. Parts of the P2 connector is used in newer VME standards. Figure 5.3 shows the structure of the functional modules and buses defined by the VME standard.

Arbitration bus A module controlling the bus will drive the bus busy line (`BBSY`) low to show that it is in use. When this line is not low the arbiter module will scan the bus request lines (`BR0`-`BR3`) looking for a pending action. Requests on the `BR3` line have the highest priority. Requests of equal priority are handled by a daisy chain using the bus grant in lines (`BG0IN`-`BG3IN`) and the bus grant out lines (`BG0OUT`-`BG3OUT`). The arbiter module mounted in slot 1 generates the first grant signal passed on to modules of increasing slot number.

Data transfer bus The data transfer bus is used for transfering data between different modules. The data bus lines (`D00-D31`) hold the actual data during a transfer. The address of the register being accessed is presented on the address bus lines (`A01-A31`). The address modifier lines (`AM00-AM05`) indicate the length of the address, the kind of data cycle and the master identifier. The address strobe signal (`AS`) is used to signal the presence of a valid address. The data strobe signals (`DS0`,`DS1`) are used by the master module controlling the data transfer to signal the presence and acceptance of valid data on the bus. Together with the (`LWORD`) signal the data strobe signals present information on the size of the word to be transferred. The `WRITE` signal line is used to distinguish between read and write operations. The data transfer acknowledge (`DTACK`) signal is used by the slave module to signal the completion of a transfer. Errors in this transfer are signaled using the bus error line (`BERR`).

Priority interrupt bus Processors in the system are dedicated to handling interrupts by monitoring the interrupt request lines (`IRQ1-IRQ7`). The interrupt request line `IRQ7` has the highest priority. In response to an interrupt, an address cycle is generated indicating that the request has been acknowledged. The interrupt acknowledge (`IACK`) signal is changed in the arbiter to a signal which is daisy chained down the bus using the interrupt acknowledge in pin (`IACKIN`) and interrupt acknowledge out pin (`IACKOUT`). A data cycle follows where the module requesting the interrupt posts its status and ID.

Utility bus The power is supplied to modules via pins at +5 V, -12 V and +12 V. An optional battery backup of the +5 V supply indicated by the (`+5STDBY`) signal can also be present. The utility bus supports an independent 16 MHz system clock (`SYSCLK`). The system failure signal line (`SYSFAIL`) and the AC failure signal line (`ACFAIL`) indicate global system problems. The system reset signal line (`SYSRESET`) is used for the initialization of the system. An additional data transfer synchronized with the serial clock line (`SERCLK`) can take place along the serial data line (`SERDAT`).

The J1 backplane of the trigger crate is a standard VME D64 backplane. The J2 backplane of the trigger crates is a special backplane with the first 3 slots corresponding to the VME D64 standard and special connectors for the rest of the backplane supporting the VME D64 standard without the standard pin assignment to access the trigger cards and the control cards. The optical receiver cards can be mounted on the back side of the J2 backplane. Additional power supplies of +2.5 V and 3.3 V for the trigger cards are used. The trigger CPUs used in the trigger readout system provide all slot 1 functions of the VME standard like arbitration, interrupt acknowledging and a bus time out function.

5.3.2 Trigger card

A detailed description of the trigger part of the trigger card can be found in Section 3.4.3. Figure 3.10 shows an overview of the trigger card. The main devices on the trigger cards are the two FPGAs APEX 20k400GCC/3 [11] from the company Altera. The FPGAs have an internal memory space to implement a pipeline for the chamber data and sufficient room to run the trigger algorithm. A VME [14] controller build in a Lattice ispL-2048 PLD [15, 16] manages the VME access to the trigger card. The VME controller is build for a A24/D32 VME access to the devices on the trigger card. It supports the VME single cycle mode and the VME block mode. A built in daisy chain makes it possible to span the block of a VME block transfer over several trigger cards. The mapping of the different devices on the trigger card is shown in Figure 5.4.

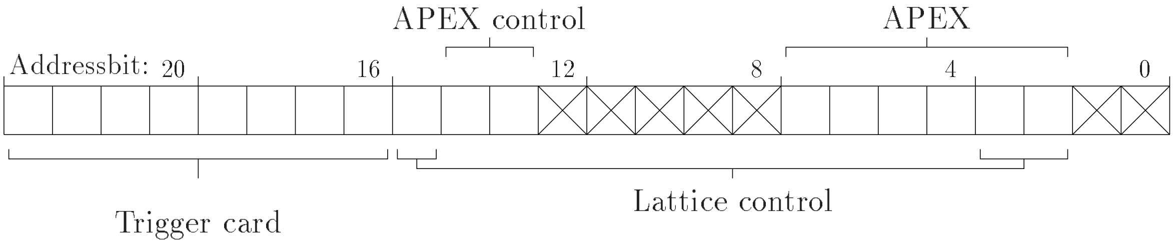


Figure 5.4: *Mapping of the VME resources on the trigger cards.*

The most significant byte of the 24 bit wide address of the VME bus resources specifies the trigger card. Two hexadecimal rotary switches on the trigger card define the VME base address of the trigger card. The Lattice control bits specify the access to registers in the VME controller and control special features like programming the EEPROMs or the FPGAs from the VME bus. The APEX control bits specify the different resources in the FPGAs like the pipelines, remote and mode registers. The APEX bits are used to map an address space in the different resources of the FPGAs. Other address lines are not connected to devices on the trigger card. All 32 data lines are connected to the VME controller and both FPGAs.

5.3.3 CPU board

As trigger CPUs and the main trigger CPU RIO2 8062 PC boards manufactured by the company CES will be used. They are described in detail in [23]. Figure 5.5 shows an overview of the structure of the PC board. The main part of the RIO2 8062 PC board is the PowerPC 603 CPU with the system memory and the L2 cache. These components are connected via a local bus and build the main computing unit of the board. A PCI bridge links the main computing unit to the PCI bus of the PC board. All other components of the board are connected to the PCI bus. An ethernet controller supporting 10/100 baseT ethernet can link the PC board resources to a local area network.
A VME bridge supporting D32/D64 access to the VME board maps the local PCI bus on

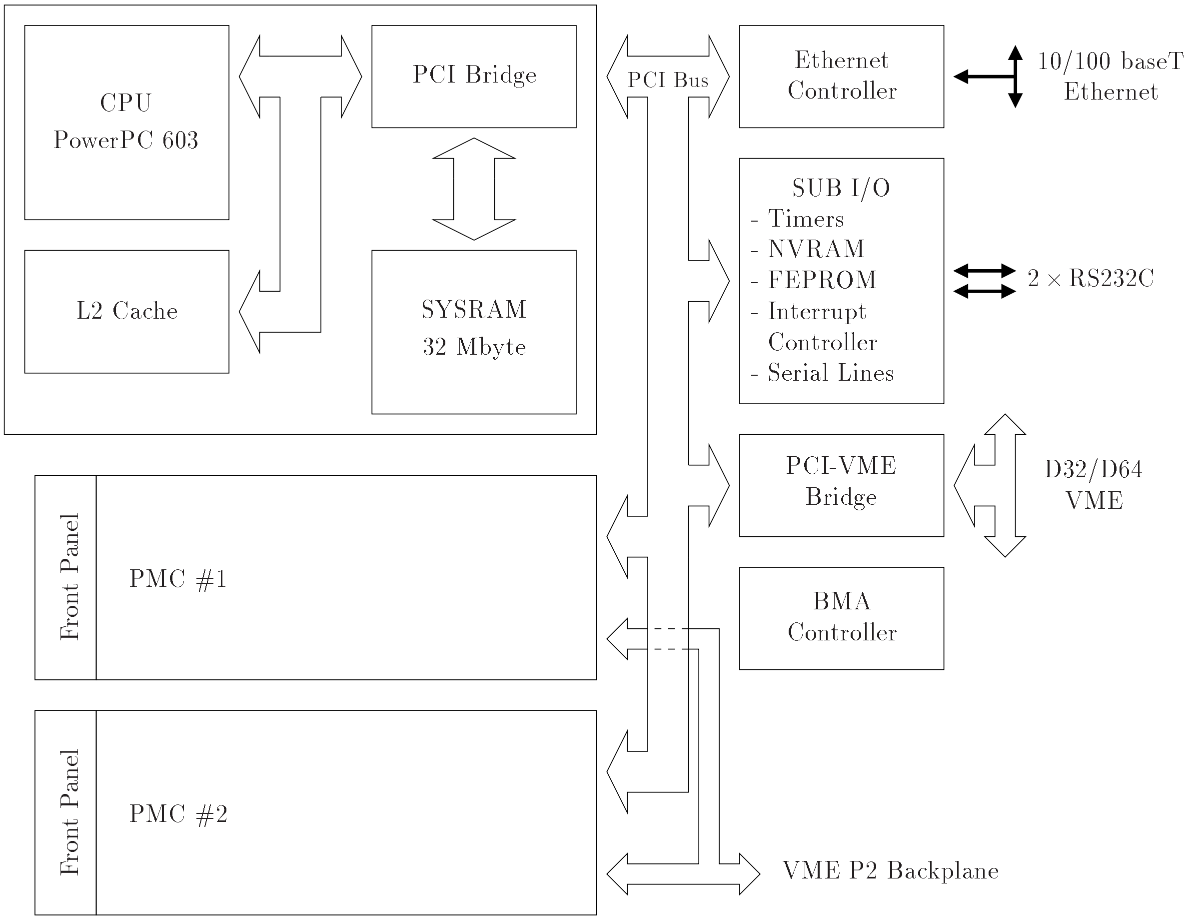


Figure 5.5: *Structure of the RIO2 8062 PC board*

the VME bus and vice versa. A BMA (**B**lock **M**over **A**ccelerator) controller manages the transfer of big amounts of data between the VME bus and the PCI bus. The PC board has different sub I/O resources accessable via the PCI which are timers, an interrupt controller, two serial lines and a FEPROM holding data without a connection to a power supply. Different PMCs (**P**CI **M**ezzanine **C**ard) can be mounted on the PC board for different tasks. In the trigger system PMCs for SCSI adapters, pVIC connections and VSB connections are used. The following resources of the PC board are mainly used in the trigger readout system.

Computing unit The PC boards in the trigger readout system use an IBM PowerPC 603e microprocessor, Mach5, Arthur 740. 32 Mbyte system memory and 1 Mbyte L2 cache are installed. The different parts of the computing unit are connected via a CPU bus.

PCI bridge The IBM 27-82660 PCI bridge is the connection of the computing unit to the PCI bus of the PC board. It manages the mapping of the system memory and registers in the CPU to the PCI bus. Interrupts coming from a device on the PCI bus can be mapped on the CPU. A PCI central arbiter handles the arbitration process of the different PCI devices on the PC board.

VME bridge The VME interface of the PC board incorporates VME master and slave logic. The VME slave mapping is done through a programmable 16 to 128 Mbytes window to map the internal system memory and through a 64 Kbytes window to map resources such as remote reset, message passing FIFO and remote configuration parameters. The memory windows are divided in 2048 pages of 64 Kbytes, each of them owning a page descriptor for the complete PCI address and specific control bits. The VME master access is mapped on two PCI windows. These windows are divided in pages of 64 Kbytes, each of them owning a page descriptor for the complete VME address, address modifier and specific control bits. A complete VME arbitration requester is implemented as well as a full VME slot 1 functionality. VME IACK cycles can also be generated to implement a VME interrupt handler through specific page descriptor initializations.

BMA controller An autonomous VME-PCI block mover (BMA) is integrated to implement efficient data transportation between the local system memory and the VME bus. This block mover allows the implementation of high speed VME transactions in the D32 and D64 mode.

FEPROM An 8 Mbyte flash EPROM is available on the PC board. The PC board can be booted from the FEPROM.

Ethernet controller The ethernet controller PC-Net 79C97x is interfaced directly on the local PCI bus and also to a serial EEPROM storing initial network parameters. All PCI target devices can be addressed by the ethernet controller.

Serial lines Two full modem RS232 serial interfaces are provided on the PC board. They are completely independent.

Interrupt controller The interrupt structure of the PC board is based on the SIC 6351 ASIC. All on board interrupts are routed to the ASIC and dispatched to the PowerPC CPU through the PCI bridge with programmable registers. Interrupt sources can be the VME bus, the ethernet controller, the BMA handler, the serial lines, PMCs or timers.

5.3.4 Connection of CPUs: pVIC

The pVIC bus is a high bandwidth PCI to PCI transparent connection. Remote PCI resources are seen as if they were local. A detailed description of the pVIC system and its components can be found in [24]. This bus system was developed by the company CES. It allows to connect clusters of 15 nodes spanning up to 200 meters while preserving the full PCI throughput (132 Mbyte/s at 32 bit, 33 MHz). An integrated DMA controller allows to perform complex data transactions with a minimal CPU load. Broadcast, and multicast cycles, interrupt dispatching, mailboxes, a mirrored memory and global semaphores provide the hardware support for efficient interprocessor communication. Different transmission technologies like GTL+, differential PECL lines and an optical link are available for the pVIC system. The pVIC module used in the CIP2000 trigger system readout is mounted on a PMC housed in the trigger CPU board.

Master interface The local bus to pVIC bridge is controlled through an outgoing scatter-gather making the complete remapping of the local addressing schema to the pVIC addressing possible. The pVIC uses a PCI memory space window of 32 to 256 Mbyte. This window is tailored in 4092 × 64 kbyte pages. Each individual page is associated with a descriptor supplying the pVIC page parameters.

Slave interface The pVIC slave interface permits access to the pVIC local resources like registers, SSRAM, mirrored memory and the local PCI bus. It receives and decodes the pVIC transaction. The decoding is handled through node id informations. Each pVIC slave has its own unique node id. PVIC broadcasts to all nodes are possible.

Mirrored memory The pVIC mirrored memory allows the building of a shared memory system with a global write and a local read. The mirrored memory is implemented with a true dual port SRAM device.

DMA controller The pVIC interface incorporates a DMA controller. The local DMA controller supports a local read and pVIC write. The DMA ininitialization is controlled by a chaining-block in the local SSRAM. The DMA chaining-block initialization can be done locally or remotely from the pVIC.

Interrupt dispatching The pVIC provides a mechanism to export transparently local interrupts or events to external pVIC nodes. On an event detection the local controller issues a mirrored memory write transaction.

5.3.5 Connection of main CPU and STC system: VIC

The VIC system connects different crates with local bus systems like the VME bus. It uses the the VMV bus structure to transfer the data between the different crates. A detailed description of the VIC modules build by the company CES can be found in [25]. The VMV bus is a multiplexed bus for full 32 bit address and data transfer. Figure 5.6 shows the structure of the in the CIP2000 trigger readout system used VIC module. Two of these modules are connected by two flat cables transmitting signals in differential standard. The module can be accessed by the VME bus and by a VSB bus.

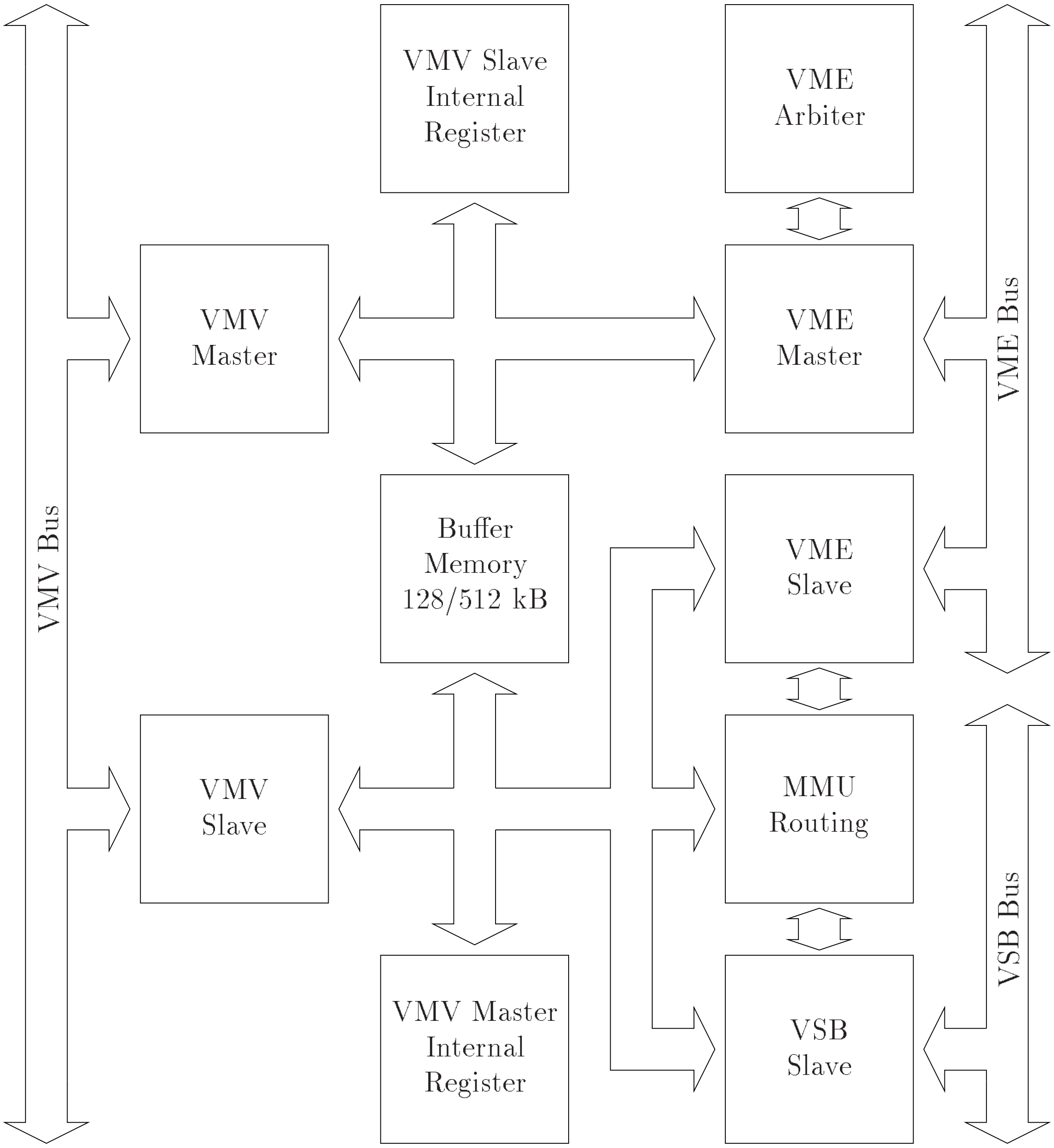


Figure 5.6: *VIC 8250 module structure*

VMV bus The VMV bus is a multiplexed bus for full 32 bit address and data transfer. The electric signals transmitted across the two flat cables are differential. Each signal of the differential bus is terminated at its two physical ends by an active terminator. The VMV bus is asynchronous and uses a signal called vertical timing to provide a correct handshake mechanism between the master and the slave. Three different types of transfer cycles are supported. A direct access on the local VME of the VMV slave module, a direct access on the internal control resources of the VMV slave and interrupt cycles to pass the interrupt information on to the VME bus. The interrupts are transfered multiplexed on the bus interrupt lines.

VMV master The VMV master module drives the VMV bus. Incoming VME or VSB cycles destined for the VMV bus are controlled by this logic. Arbitration phases and handshaking cycles are transparent to the VME and VSB bus. The internal TTL signals are converted into differential signals to drive the VMV flat cables.

VMV slave The VMV slave module decodes the VMV bus cycles and converts them into one of four types of cycles. A cycle to access the VMV internal registers, a cycle to access the VME resources, a cycle to generate VME interrupt acknowledge cycles and a cycle to access the internal buffer memory. The VMV slave internal register module contains informations about the VMV slave module and the VME master interface.

VME master The VME master module converts the decoded VMV cycles into standard VME cycles. The conversion is done in real time at every cycle. A build in VME arbitration module can handle the VME arbitration.

VME slave The VME slave module connects the VIC module to the VME master unit of the VME bus. The VME slave resources are mapped to the VME bus in a 1 Mbyte window in the VME standard address space. The mapping is defined by a hexadecimal rotary switch. The VMV master internal register, the VMV slave internal registers, the internal buffer memory and the MMU routing RAM can be accessed. The VME slave module also transmits the VME bus cycles to the VMV bus through the MMU routing module. The VME slave module can be a VME interruptor.

VSB slave The VSB slave module is similar to the VME slave module with the same abilities. The VSB mapping is done through a 16 Mbyte window with a software programmable offset. After a VME reset the VSB slave port is disabled. The VSB slave module can be a VSB interrupter.

MMU routing module The MMU (**M**emory **M**anaging **U**nit) routing module acts as a dynamic decoder over the VME bus and the VSB bus. The different address fields are divided into 1 Mbyte windows covering the complete addressing range. Each page can be marked to be taken by the VMV master module to be sent over the VMV bus to other VMV slave units. When a page is marked the high address field, the VME destination crate and other specific informations are defined in the associated page descriptors. A total of 8196 page descriptors are stored in the MMU RAM.

Internal buffer memory The 32 bit internal buffer memory module is triple ported via VME, VSB and VMV and locally arbitrated. The memory area up to 512 kbytes is mapped in the VMV master internal register accessible from the VME and VSB bus.

The connection of the main trigger CPU to the STC system mainly uses the ability to read and write registers like scalers and status bits in the STC cards over the VMV bus and to pass interrupts coming from the STC cards to the main trigger CPU.

5.3.6 Connection of main CPU and VMEtaxi card: VSB, VIC

The VSB (**V**ME **s**ubsystem **b**us) bus is a good supplement as a secondary bus for the VME bus. Besides being useful as an extended memory bus, a secondary bus can be used to reduce overall bus traffic on the VME bus. The VSB bus uses the A and C rows of the P2 connector to provide 64 bused signals. A standard VME bus J2 backplane does not use the rows A and C. To support the VSB bus one needs either a piggy-back VSB printed circuit backplane, which can be plugged onto the standard VME bus J2 backplane or a special J2 VSB backplane that has the A and C rows already bused. The VSB bus was approved as an IEEE standard in 1988.

The CIP2000 trigger readout system uses a VSB bus to connect the main trigger CPU to a VIC module in the main trigger crate. A VSB PMC is mounted on the main trigger CPU board to access a VSB sub backplane mounted on the backside of the J2 backplane. The VIC module manages the connection to the VMEtaxi card of the H1 readout system. The VMEtaxi card is the connection to the optical readout ring of the H1 experiment. The VME bus of the main trigger crate and the connection to the VIC module via the VSB backplane are totally independent. Data can be transfered to the VMEtaxi card while the main trigger CPU is managing the horizontal readout via the VME bus.

5.4 Readout system software

The trigger CPUs including the main trigger CPU are running under the operating system LynxOS [26] from the company LynuxWorks. LynxOS is a UNIX compatible and POSIX [27, 28, 29] conformant multi-process and multi-threaded real-time operating system. Drivers for the VME, pVIC and VSB access are provided and supported by the company CES manufacturing the CPU boards. All programs are being written in C/C++. The CPUs are connected by ethernet links making it possible to boot the system from the network. In addition the CPUs will be able to boot from the FEPROMs mounted on the CPU board. One CPU board will have a SCSI adapter to access a local harddisk. This CPU with the harddisk can operate as the file server for the system if no connection to other computers via the ethernet is possible.

All trigger CPUs will run the same program but with different setups adjusted to the environments (e.g. sectors in ϕ) it works in. The setup can be changed at run time. The program structure will have a state machine structure switching from state to state depending on the status of its environment. The different states correspond to the states used in Section 5.2 to describe the trigger readout timing. The communication between the CPUs and other parts of the system is managed by reading and setting of registers in the different components of the system. Interrupts are avoided in the system. The only part where interrupts are used is in the communication of the STC system and the main trigger CPU. A multi event buffer structure will be implemented in every stage of the readout system. This means that different events can be processed at the same time.

Chapter 6

Tests with the CIP2000 trigger readout test system

6.1 Test of the VME bus

To measure the VME bus performance one has to understand how the VME bus transferes the data. Two different transfer types are used in the CIP2000 trigger readout system. These transfer types are the single cycle and the block transfer. In the D32 mode used in the system the single cycle transfer writes or reads a 32 bit longword from a VME address. In the block mode a block of multiple 32 bit longwords are read from or written to a consecutive address space. Figure 6.1 and Figure 6.2 show the timing of the different signals used in the different transfer modes. The signal Address Strobe (`AS*`) shows that

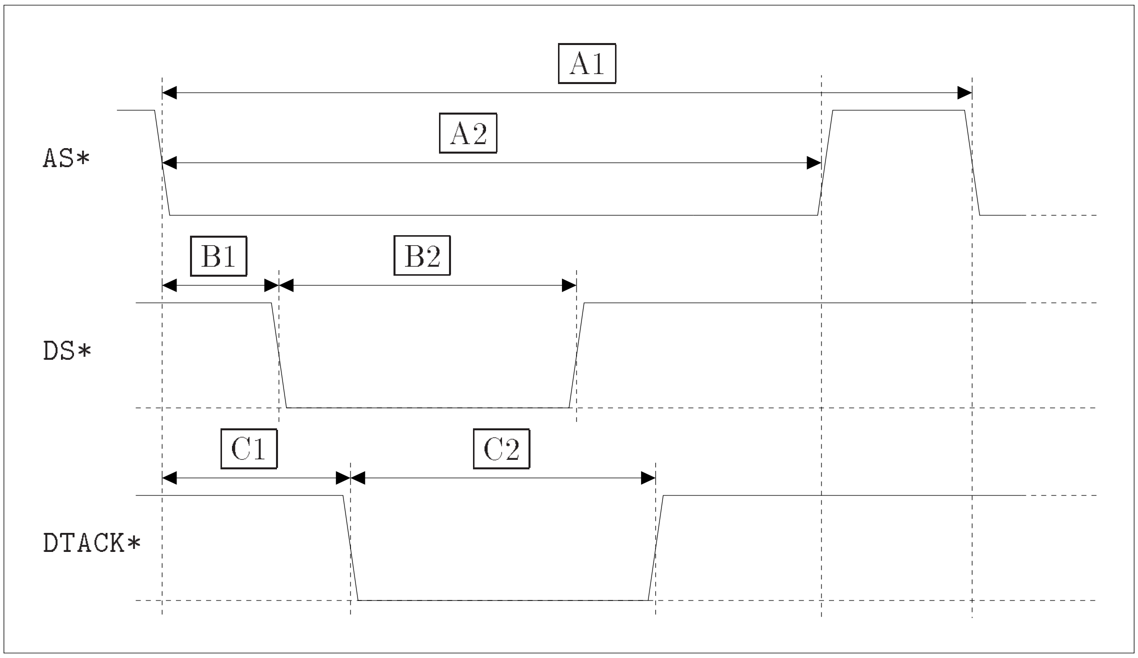


Figure 6.1: *Signal timing of a VME bus single cycle data transfer*

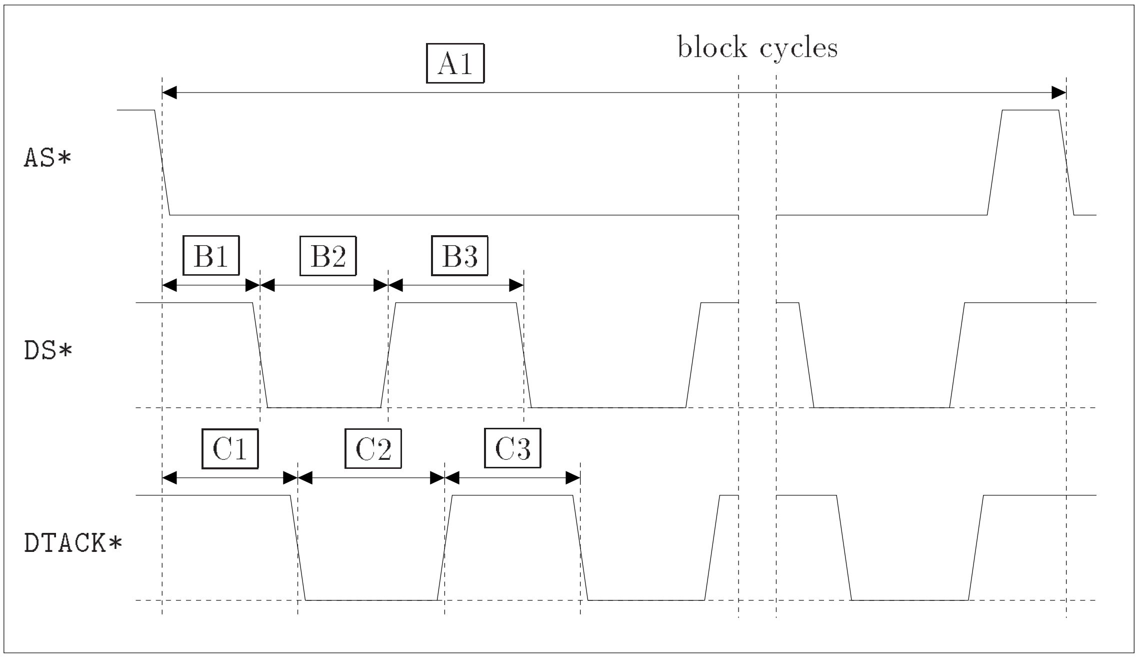


Figure 6.2: *Signal timing of a VME bus block data transfer*

the addresses on the address lines are valid. The Data Strobe (`DS*`) signal shows that the data on the data lines is valid during a write cycle, or that the data has been accepted from the data bus during a read cycle. The address and data line signals are not shown in the figure. The Data Transfer Acknowledge (`DTACK*`) signal is the signal generated by a VME slave to show that the data is available on the data bus during a read cycle, or that data has been accepted from the data bus during a write cycle. During a block mode cycle the `AS*` signal is not released while the `DS*` and the `DTACK*` signals validate the different subcycles of the block transfer. Figure 6.3 shows the structure of the test

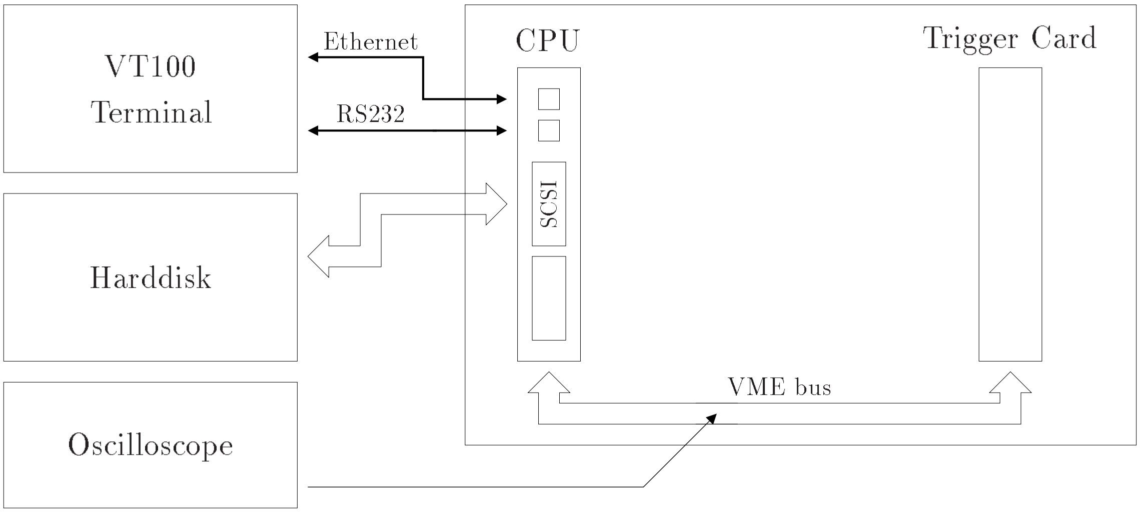


Figure 6.3: *Test system to examine the VME bus performance*

system the different parameters of the VME bus transfer were measured at.
The system uses the CPU board running LynxOS connected to a harddisk via a SCSI PMC. The CPU is accessed via a serial line or after the setup of the network environment via the ethernet connection. A prototype of the trigger cards was used. The FPGAs were programmed the way that they behave like a random accessable memory block. The time period the VME bus needs for a data transfer can be determined by measuring the time between two falling edges of the `AS*` signal within two data transfer cycles.
Table 6.1 and Table 6.2 show the measured timing parameters for the different transfer modes. In one test the VME cycles were called from a test program written in C where the data was further processed by the CPU to test the influence of a program overhead.

Time period	Δt [ns]			
	no program overhead		with program overhead	
	read	write	read	write
$\boxed{\text{A1}}$	820	944	1096	928
$\boxed{\text{A2}}$	608	572	606	586
$\boxed{\text{B1}}$	120	180	122	184
$\boxed{\text{B2}}$	168	196	172	190
$\boxed{\text{C1}}$	200	308	216	308
$\boxed{\text{C2}}$	224	228	214	224

Table 6.1: Measured timing parameters of VME bus data transfers: single cycle mode

Time period	Δt [ns] for 16 cycles			
	no program overhead		with program overhead	
	read	write	read	write
$\boxed{\text{A1}}$	5920	6380	5340	6360
$\boxed{\text{B1}}$	68	100	58	88
$\boxed{\text{B2}}$	172	192	158	208
$\boxed{\text{B3}}$	164	172	174	168
$\boxed{\text{C1}}$	156	208	146	224
$\boxed{\text{C2}}$	200	200	194	200
$\boxed{\text{C3}}$	144	176	152	176

Table 6.2: Measured timing parameters of VME bus data transfers: block mode

The other test used low level commands directly passed to the CPU base operating system. In the block mode the block of data contained 16 cycles of 32 bit data. In the single cycle mode one cycle of 32 bit data was transfered. The performance of the VME bus in the different modes can be calculated by the time period for the data transfer and from the size of the transfered data. Table 6.3 shows the measured data transfer performances for the different transfer modes.

Transfer mode	Data transfer rate [Mbyte/s]			
	no program overhead		with program overhead	
	read	write	read	write
Single cycle mode	4.65	4.04	3.48	4.11
Block mode	10.31	9.57	11.43	9.60

Table 6.3: Measured VME bus data transfer performance

The tests show that the in Section 5.2 assumed transfer rate of 10 Mbyte/s can be reached in the block mode.

6.2 Test of the pVIC bus

To test the performance of the pVIC connection two CPUs are connected via the pVIC link. Figure 6.4 shows the test system used in this test. First it has to be arranged that both CPUs are booting the operating system LynxOS to be able to activate the pVIC drivers. The first CPU is booting the operating system from a local SCSI harddisk. It operates as a boot and file server for the other CPU. The second CPU is booting the operating system via the ethernet connection to the first CPU. The CPUs are controlled via a RS232 connection from a VT100 terminal.

The pVIC link can operate in different modes. A normal transfer mode and a DMA mode are available. Normal transfer mode uses single cycle transfers transfering one 32 bit word data from a local PCI resource of one CPU to a local PCI resource of the other CPU. The main memory of the CPU boards was used as the local PCI resource for the tests. The DMA (**D**irect **M**emory **A**ccess) mode can move big blocks of data without putting a big load on the CPU.

Without a big effort it is not possible to access the hardware of the pVIC bus directly like it has been done for the timing measurement of the VME bus performance. Test programs have been written to measure the performances in the different modes. The programs use the internal realtime clock of the CPU board as a time standard.

In the normal transfer mode 65536 cycles of 32 bit data of a total amount of 0.25 Mbyte

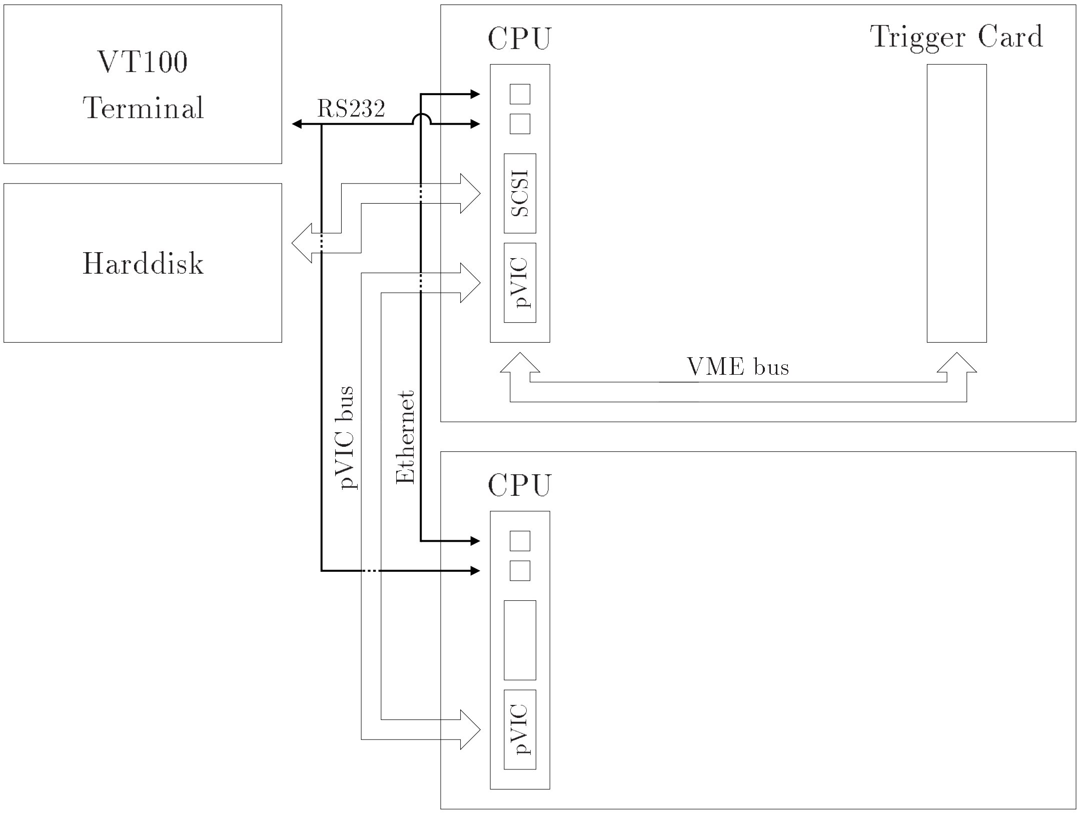


Figure 6.4: *Test system to examine the pVIC connection performance*

have been written from the first CPU to the second CPU and read from the second CPU to the first CPU via the pVIC bus 200 times. The measured time for this transfer of 0.25 Mbyte is (77.4 ± 1.9) ms. From this a performance of (3.23 ± 0.08) Mbyte/s can be calculated. The normal single cycle mode will be used in the system to manage the communication between the CPUs by setting registers in the CPU memory.

The DMA mode will be used in the system to transfer the read out data. A DMA transfer is initialized by writing DMA transfer information in a DMA FIFO of a local pVIC node or a remote pVIC node. Just write transfers are possible. A read transfer is done by initializing a DMA write transfer on a remote pVIC node to the local pVIC node. The size of the DMA FIFO can be set. The transfer rate is strongly dependent on the size of the transfered data. The test program transferes a given number of bytes from the local pVIC node to the remote pVIC node. The time for the transaction is measured and the data transfer rate is calculated. Table 6.4 and Figure 6.5 show the measured data transfer rates depending on the size of the transfered data.

For the CIP2000 trigger system the size of one block transfered from one trigger CPU to the main trigger CPU if five events are read out from the pipeline is 1600 byte if no data reduction algorithm is applied on the data in the trigger CPUs. A transfer rate of over 40 Mbyte/s can be reached in the system. This is four times higher than the assumed transfer rate of 10 Mbyte/s. If one assumes a data transfer rate of 40 Mbyte/s and 3 (5) events are read out from the pipeline the vertical readout can be managed in 68.67 μs (114.44 μs).

Data size [byte]	Transfer rate [Mbyte/s]
4	1.4
16	3.2
32	10.5
64	18.5
128	28.8
256	37.6
512	44.2
1024	44.8
1600	**45.8**
2048	47.1
4096	46.7
8192	46.5
32768	46.5

Table 6.4: Transfer rate of the pVIC connection in DMA mode depending on the size of the transfered data.

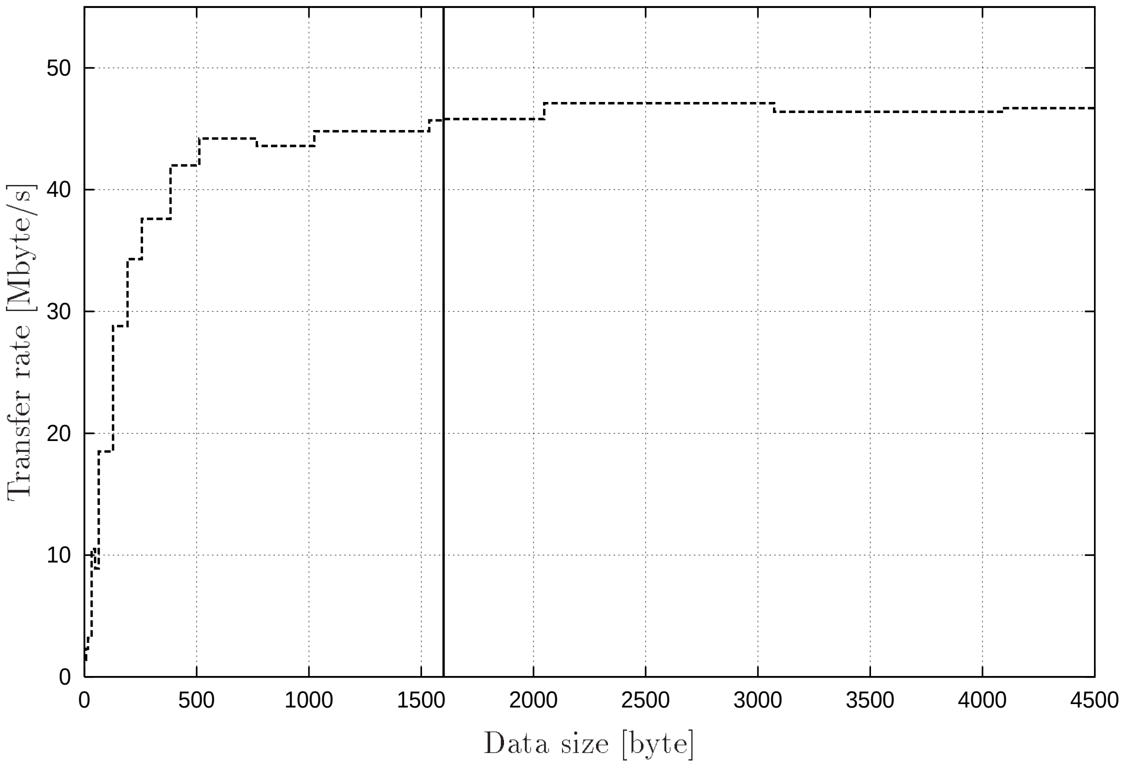


Figure 6.5: *Transfer rate of the pVIC connection in the DMA mode depending on the size of the transfered data. The solid line marks the size of the data that has to be transfered via the pVIC connection in the actual system.*

Chapter 7

Data reduction for the CIP2000 trigger readout

The chamber of the CIP2000 trigger consists of about 9600 pads. The information of 3 (5) events read out from the pipeline has therefore a size of about 3.5 kbyte (5.9 kbyte). It is necessary to read out several events from the pipeline to do timing studies and to be sure that the actual triggered event is read out. In order to keep the overall bandwidth of the H1 readout small, there is a need to reduce the data produced by the trigger system to an amount of about 1 kbyte. The reduction algorithm should be reversible to avoid the loss of information. The following section shows how the data is mapped from the chamber to the memory of the CPU where the reduction algorithm is applied. Two reduction algorithms have been investigated to reduce the data. Both algorithms have been simulated using real H1 events recorded in the existing CIP chamber, mapped on the new chamber geometry.

7.1 Raw data of the CIP2000 trigger

Figure 7.1 shows how the data of half a sector in ϕ is mapped on the CPUs memory. Every CIPiX chip provides the readout for one half of a layer of one sector in ϕ. The chip multiplexes the data four times grouping four adjacent pads to one group. In the figure one of these groups is surrounded by a black box. The groups of pad information are stored in the pipelines of the FPGAs. The figure shows five events of the pipeline forming the event window of events to be read out. The event window shows how many events are read from the pipeline. Three or five events should be read from the pipeline to make sure that the origine event triggering the readout is read out. The figure shows the four steps of the data demultiplexing. The events are read out from the pipeline via a 32 bit VME block cycle. The block cycle for one event is subdivided in ten subcycles. The mapping of the FPGA pipeline memory on the VME cycles is shown in the figure. Two adjacent VME subcycles carry the bit information of one CIPiX. There are four bits left over in the cycles that can be used for parity cross checks of the information. The information of the full event window can be read out in one block cycle. The raw data

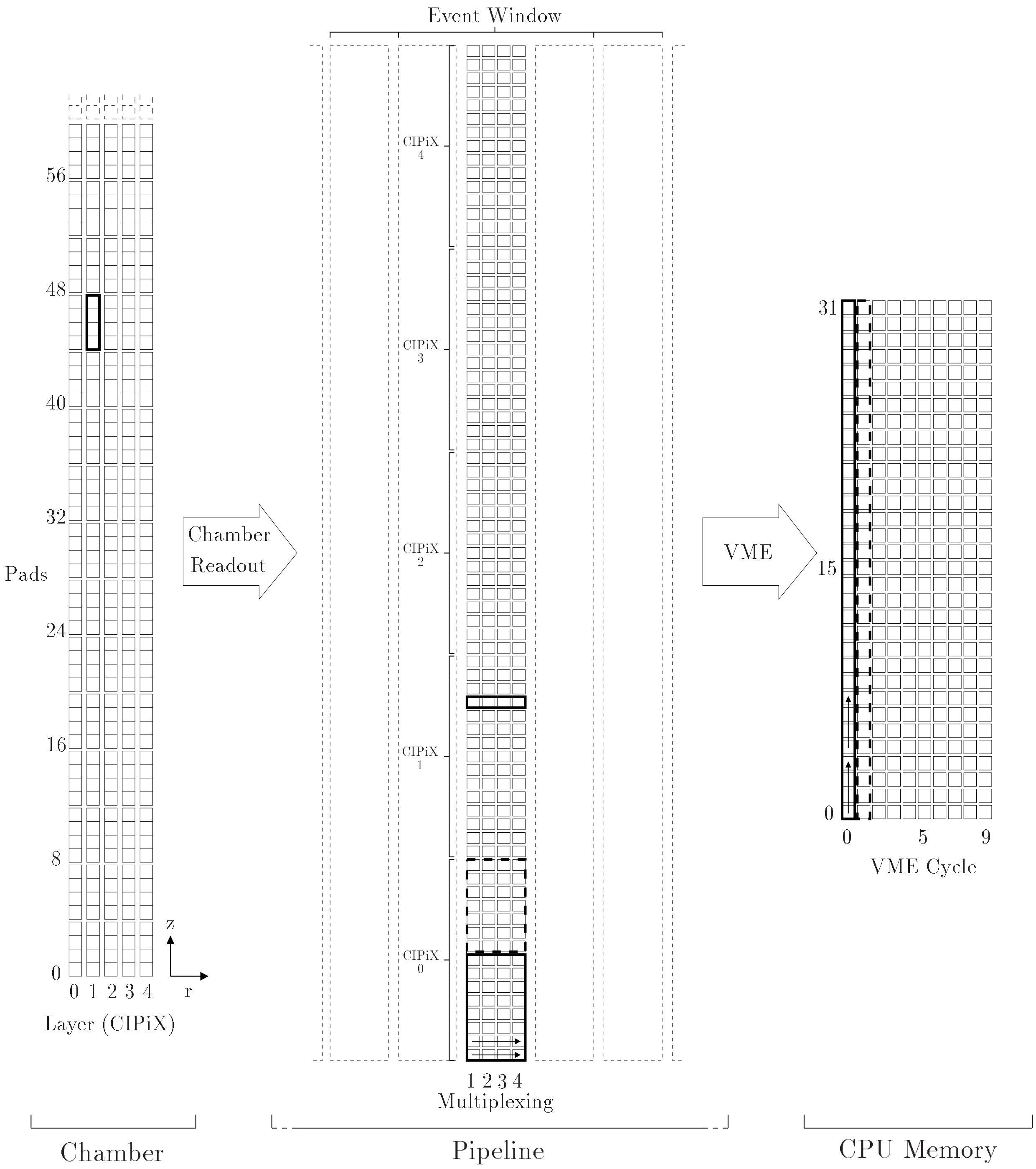


Figure 7.1: *Mapping of the chamber data of half a sector in ϕ on the CPU memory. Every CIPiX chip provides the readout for one half of a layer of one sector in ϕ. The chip multiplexes the data four times grouping the pads in pairs of four adjacent pads (small black box). The pairs of pad information are stored in the pipelines of the FPGAs. The events are read out from the pipeline via a 32 bit VME block cycle. The block cycle for one event is subdivided in ten subcycles. Two adjacent VME subcycles carry the bit information of one CIPiX (solid and dashed box). There are four bits left over in the cycles that can be used for parity cross checks of the information.*

of both FPGAs is merged in the CPU to the full chamber information of one sector in ϕ before the reduction algorithm is applied.

7.2 Data reduction algorithms

The chamber signals are delivered as digital patterns predominantly consisting of only a few active digital bits like a sparse matrix. The reduction algorithm should be reversible to avoid the loss of data. The run length encoding and the zero suppression are two standard solutions for this kind of problem. Figure 7.2 illustrates both reduction algorithms.

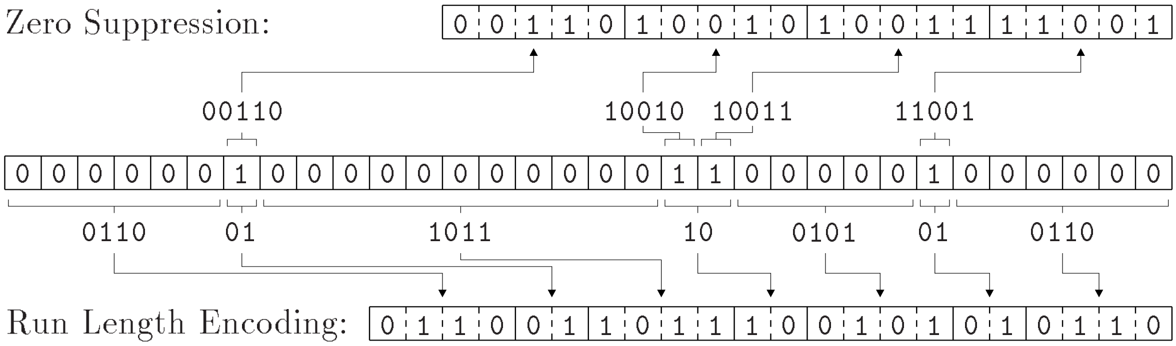


Figure 7.2: *Data reduction algorithms. The Zero suppression algorithm stores the position of a certain digital value in a bit pattern. The run length encoding algorithm stores the number of digits between two digits of a certain digital value.*

7.2.1 Zero supression

The zero suppression algorithm is designed to reduce the data of a bit pattern in form of a block of the size 2^n bit where n is a natural number. The algorithm assignes every digit with a special digital value to the position of the digit in the block as a number of n bit width. The number of digits of the certain digital value in the block has to be stored. This algorithm is referred in the simulations as algorithm 4. Figure 7.2 illustrates how the zero suppression algorithm works.

7.2.2 Run length encoding (RLE)

The algorithm counts how many digits of one digital value which appears most often in the bit pattern are between two digits of the other digital value which does not appear so often in the bit pattern. This number is stored instead of the actual bit pattern. The number has a maximum value depending on the number of bits the number is stored in. This bit width is a fixed number not changed while running the algorithm. Two problems have to be covered in the algorithm. A number of counted digits can be zero. This means that two digits of the same digital value that normally not appear so often in the bit pattern appear directly after another. This problem can be solved by storing the number of digits of both digital values. The maximum number for the digital value not appearing so often in the bit pattern does not have to be as high as for the other digital value. Table 7.1 shows which bit width have been studied. Another problem is if the number of counted digits exceed the maximum number. A zero number for the other digital value has to be stored. The bit width have to be adjusted to the given data structure that has to be reduced. Figure 7.2 illustrates how the reduction algorithm works.

Algorithm	Bit width for digital value 0	Bit width for digital value 1
1	8	8
2	6	2
3	5	3

Table 7.1: *Run length encoding (RLE) algorithms. Bit width for different digital values.*

7.3 Simulation of data reduction algorithms

To simulate the efficiency of data reduction algorithms on the data of the CIP2000 trigger system the program `CIPSIM` has been written. It loads files of fully reconstructed events mapped on the CIP2000 geometry and simulates the whole readout part of the system until the data arrives at the memory of the trigger CPU. The different data reduction algorithms are applied on the data and the size of the resulting data is compared to the size of the original data. Some files include additional noise generated by switching on random pads. Noise levels from 0% to 12% have been investigated.
Three different event signatures have been used. The events can be classified by the four-momentum transfer between the electron and the proton. High q^2, low p_T and background events are investigated. High q^2 events are events with a high four-momentum transfer. Neutral and charged current events are possible. Neutral current events with two hadronic jets opposite to each other are used for the simulation. The data for these events were taken in 1997. Low p_T events are events with very low four-momentum transfer squared ($q^2 \approx 0$). This process is called photoproduction. The electron is nearly not scattered by the proton. Photoproduction events with two jets in the detector from 1995 are used in the simulations. The different background event types are described in Section 3.1. For the simulation proton background events from 1997 are used. The data was taken in a run where only protons were filled in the accelerating storage ring.

7.3.1 Activated pads

To study the reduction algorithms it is important to understand the structure of the data arriving at the CPU memory. The number of activated pads per event gives information about the different event signatures. The pads are activated by particles hitting the active pad area or by switching on random pads to simulate noise. Figure 7.3 shows the distribution of the pads activated per event for different event signatures. Table 7.2 shows the mean value and the standard deviation of the distribution of the activated pads for different event signatures. In the majority of the events less than 10% of the pads are activated. The background events have a big standard deviation and the mean value of the activated pads is high with respect to the other event signatures. Figure 7.4 shows the noise dependence for the mean value of the activated pads for different event signatures. The noise dependency is nearly linear.

Event type	Mean value of activated pads [%]
High q^2	2.23 ± 1.43
Low p_T	3.23 ± 1.59
Background	6.68 ± 4.67

Table 7.2: *Mean value and width of distribution of activated pads: different event types, no noise*

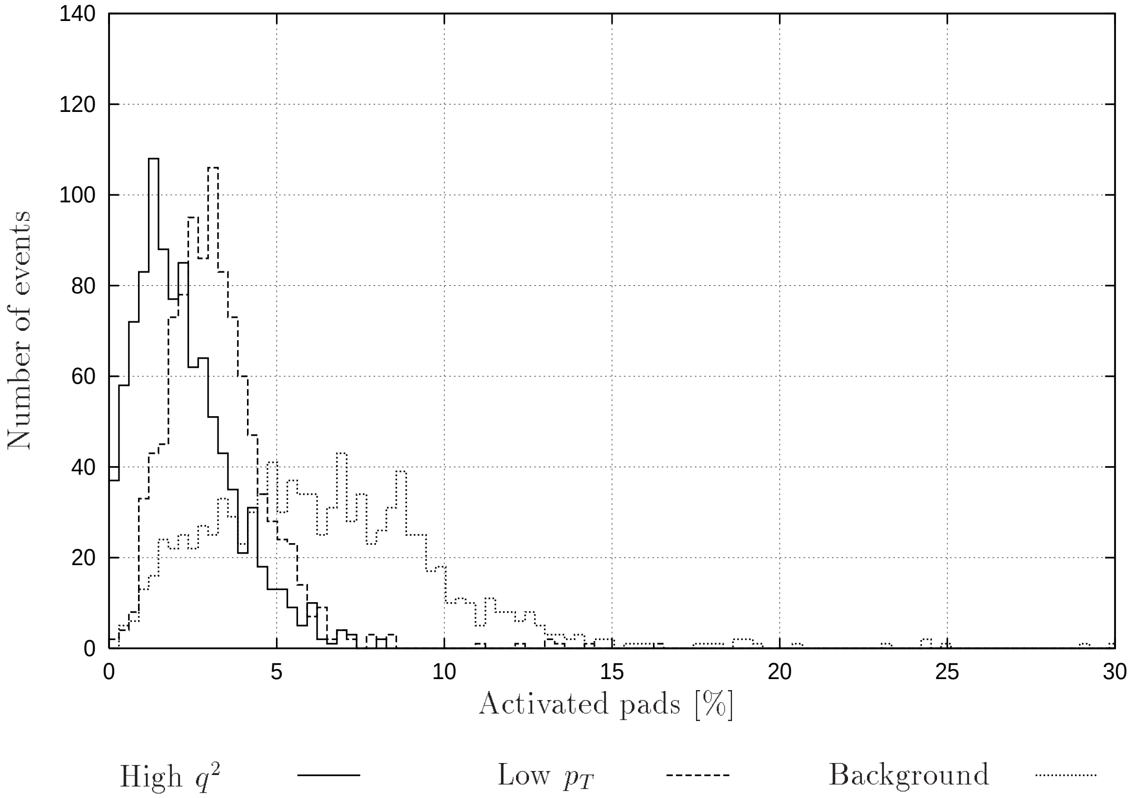


Figure 7.3: *Activated pads: different event types, no noise*

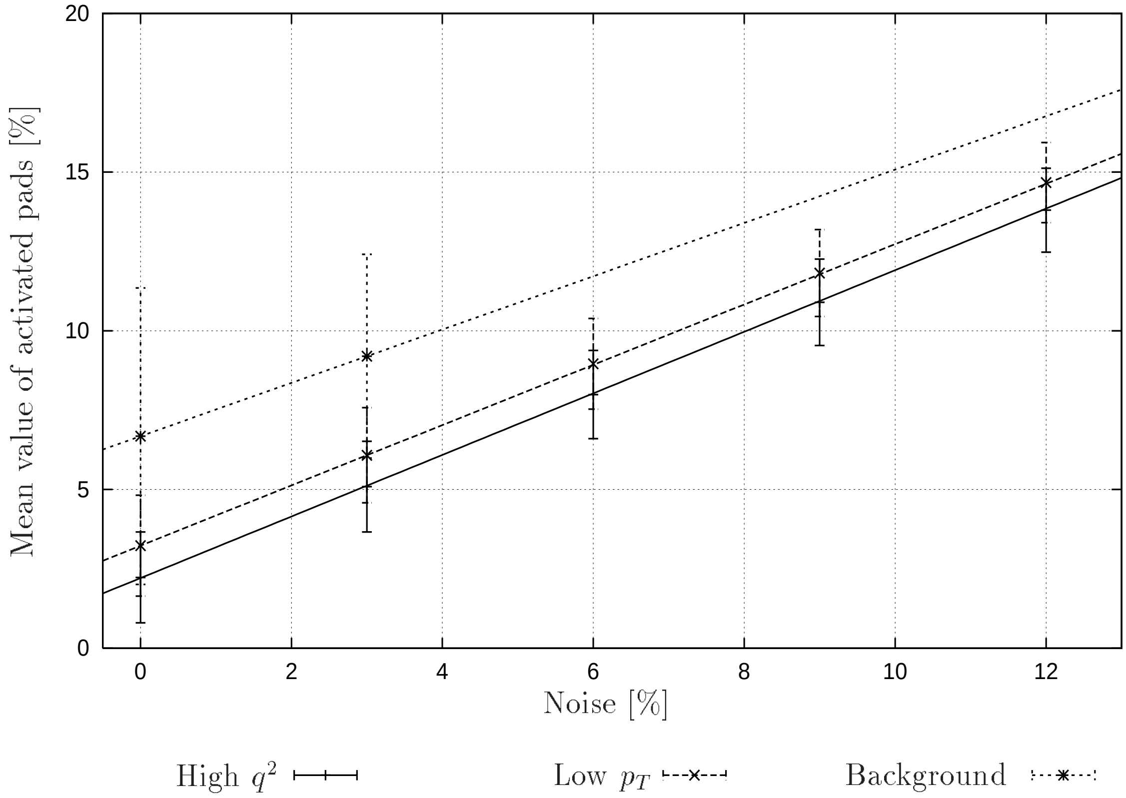


Figure 7.4: *Activated pads: noise dependence for mean value of activated pads*

7.3.2 Data reduction algorithms

Four different data reduction algorithms have been simulated. Algorithm 1-3 are run length encoding algorithms. The differences are explained in Section 7.2. Algorithm 4 is a zero supression algorithm with 8 bit width numbers. Table 7.3 shows the results of the simulations. The ratio of reduced number of bits to the original number of bits is called reduction ratio. The lower the reduction ratio is the smaller will be the resulting data. Figure 7.5 - 7.7 show the effect of the different data reduction algorithms on the different event signatures. Algorithm 2 and algorithm 4 are the most effective algorithms on the

Algorithm	Reduction ratio [%]		
	High q^2	Low p_T	Background
1	36.44 ± 18.67	50.80 ± 21.54	96.26 ± 41.64
2	25.34 ± 7.41	30.83 ± 9.09	51.85 ± 24.75
3	35.83 ± 5.29	39.56 ± 7.25	56.63 ± 21.95
4	22.32 ± 11.44	30.38 ± 12.71	56.74 ± 29.06

Table 7.3: *Reduction ratio and width of the distribution for different algorithms on different event types, no noise*

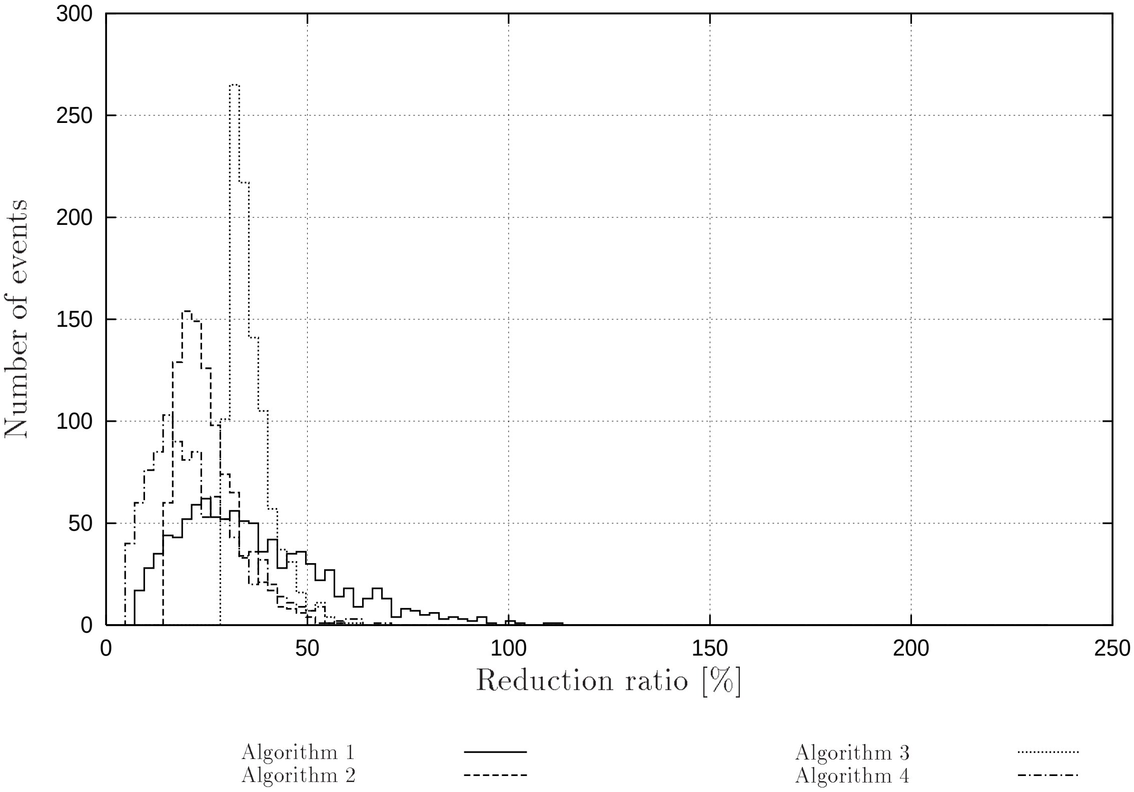


Figure 7.5: *Reduction ratio for different algorithms on high q^2 events, no noise*

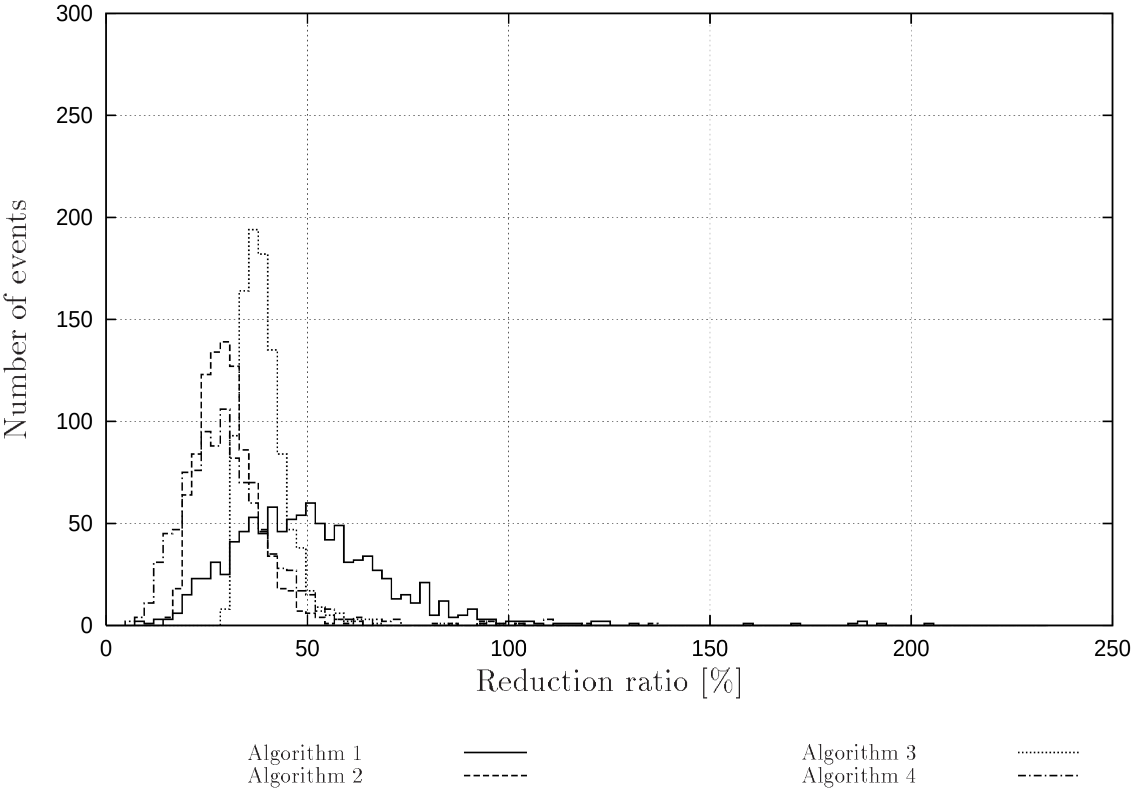


Figure 7.6: *Reduction ratio for different algorithms on low p_T events, no noise*

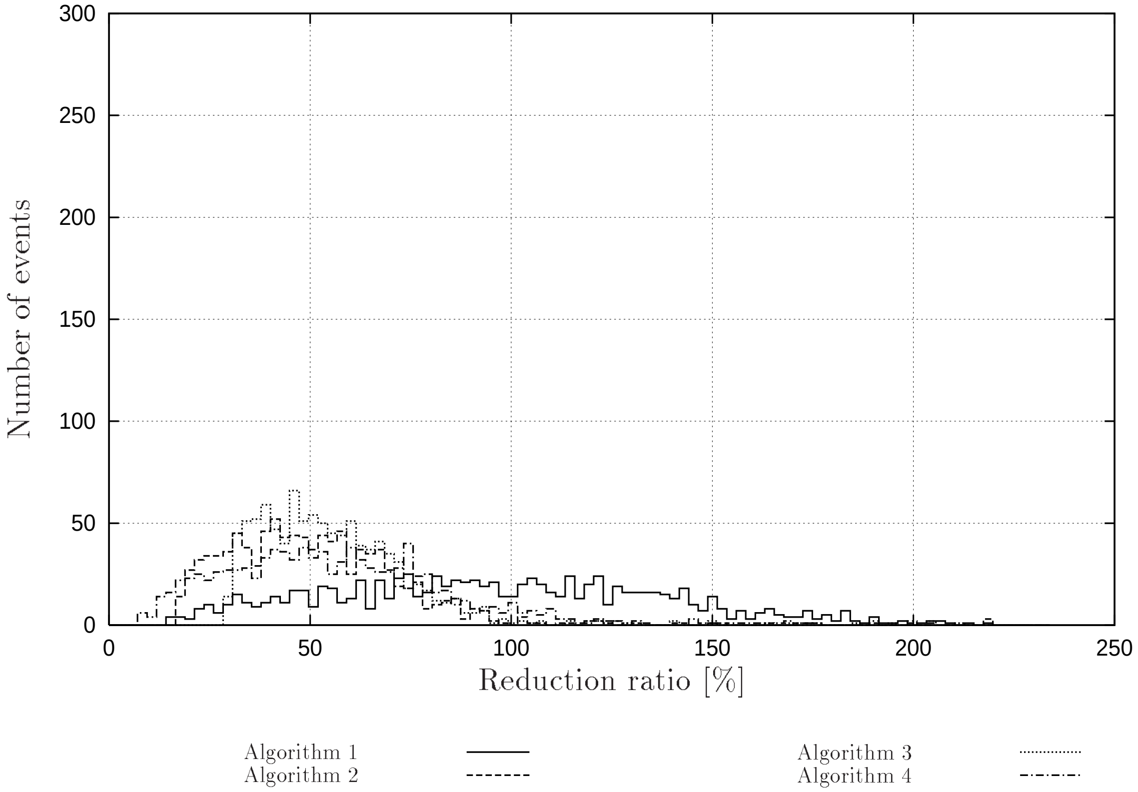


Figure 7.7: *Reduction ratio for different algorithms on background events, no noise*

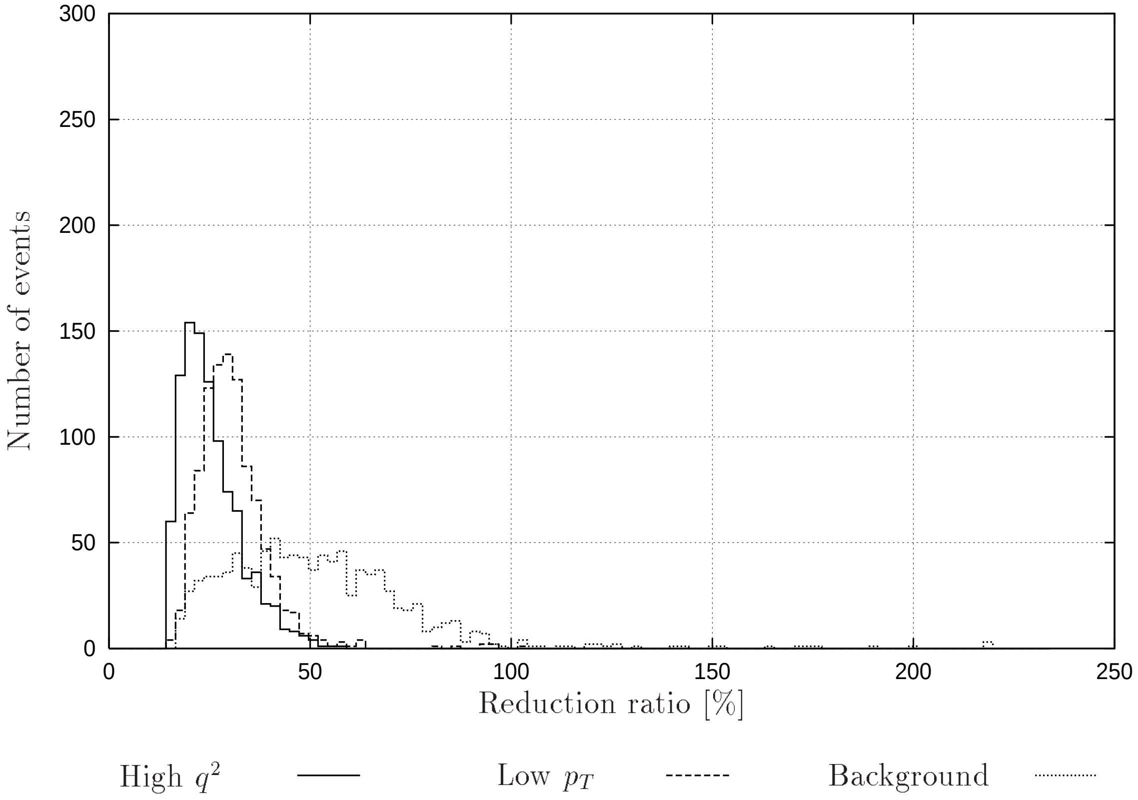


Figure 7.8: *Reduction ratio for algorithm 2 on different event types, no noise*

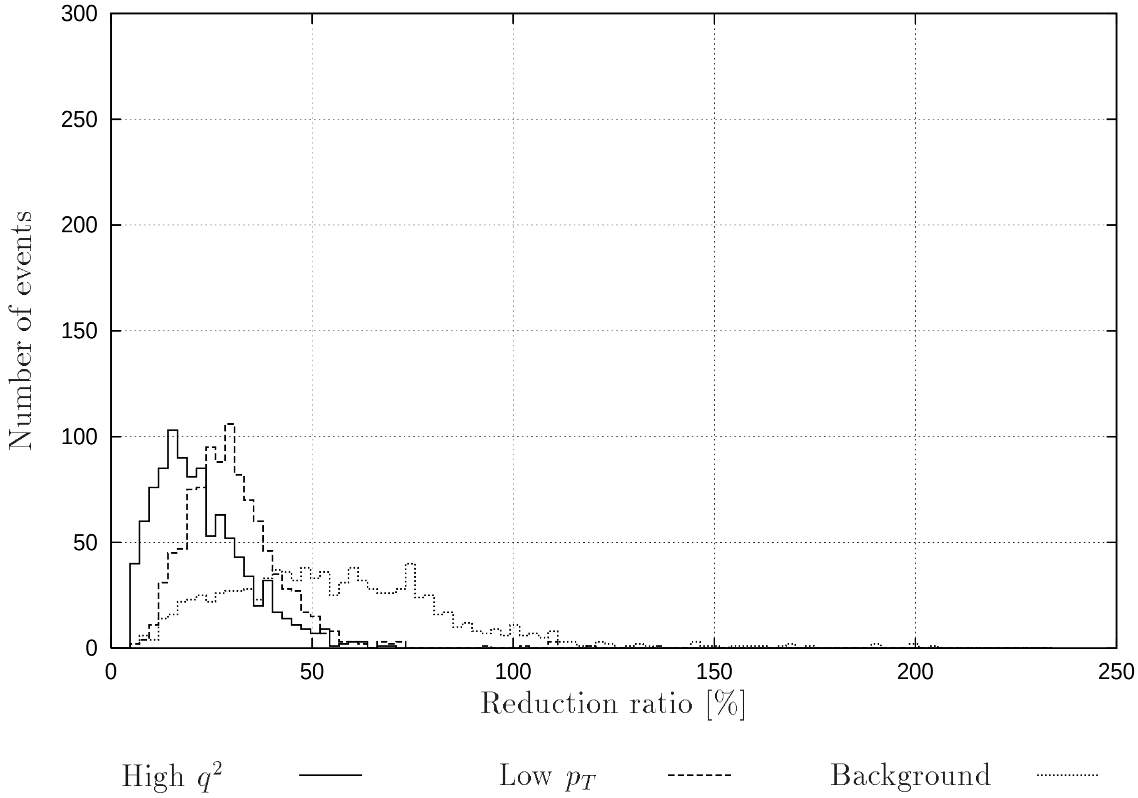


Figure 7.9: *Reduction ratio for algorithm 4 on different event types, no noise*

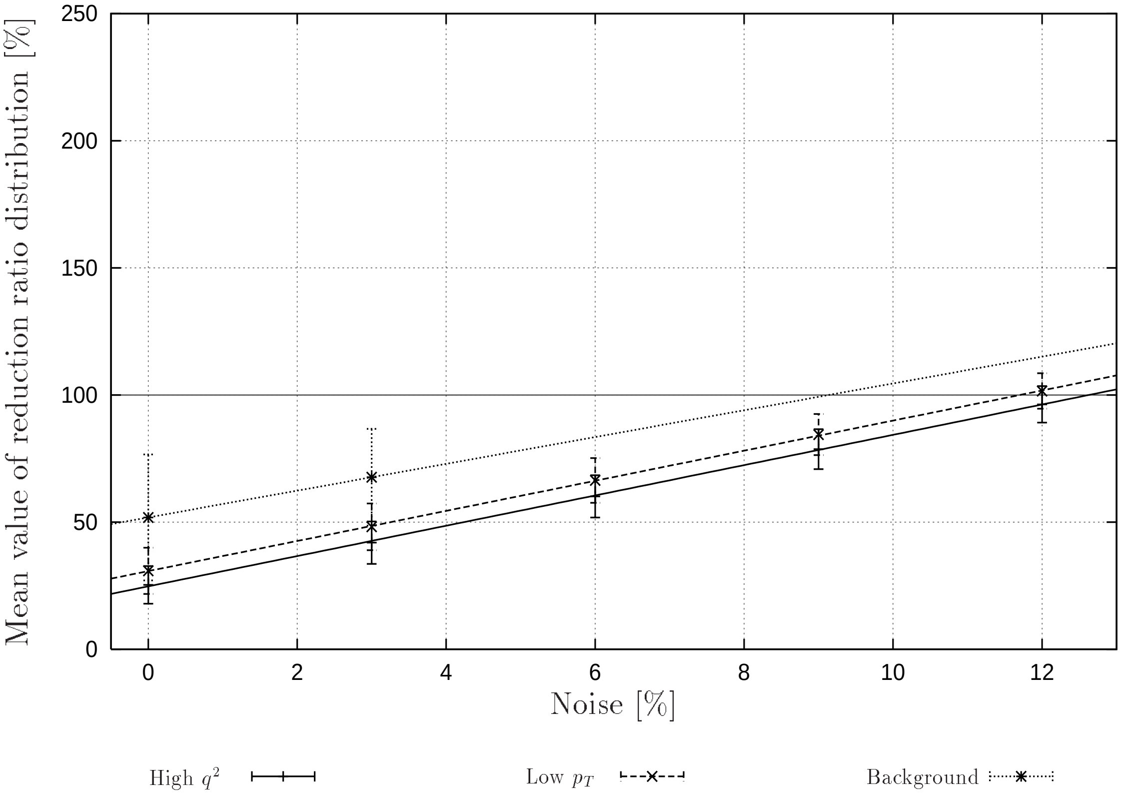


Figure 7.10: *Noise dependence of the reduction ratio for algorithm 2 on different event types*

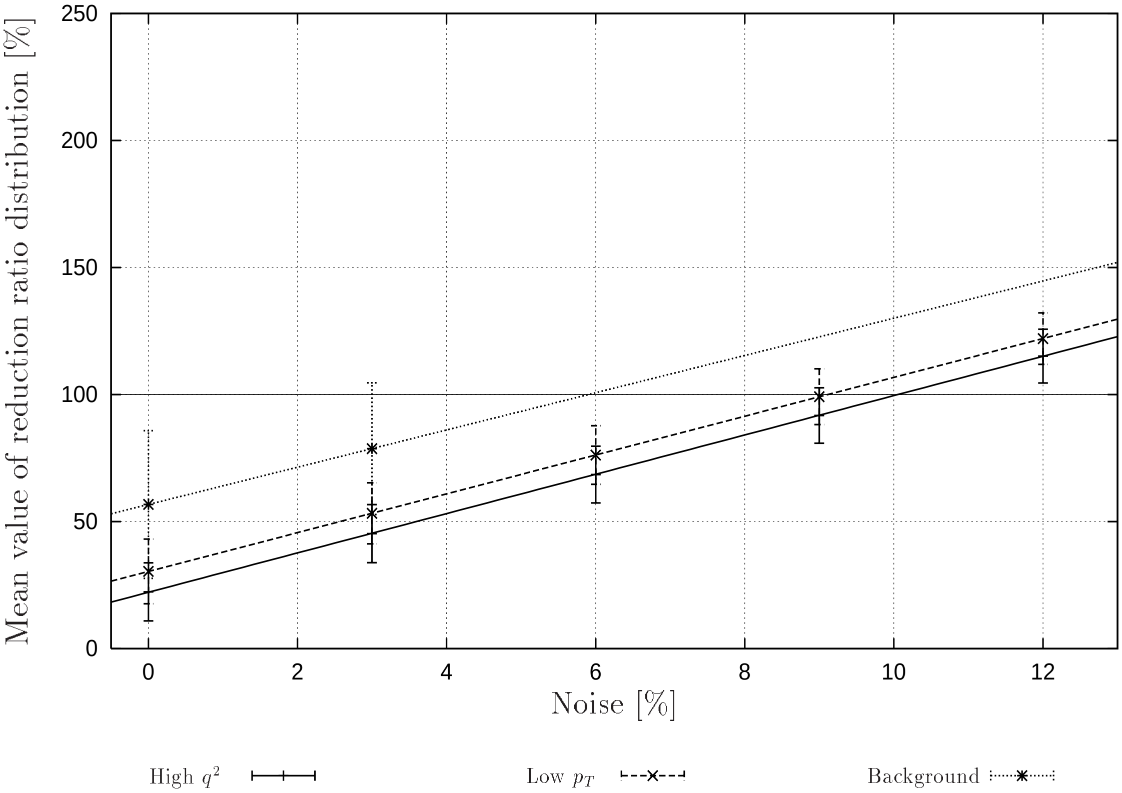


Figure 7.11: *Noise dependence of the reduction ratio for algorithm 4 on different event types*

data. Figure 7.8 and Figure 7.9 show the effect of one certain reduction algorithm on the different event signatures. Figure 7.10 and Figure 7.11 show the effect of noise on the data reduction algorithm 2 and algorithm 4.

7.3.3 Conclusion

The previous section covered the simulations done to reduce the data of one event read out from the pipeline. The performance of the data reduction algorithm on the whole data read out from the pipeline is now investigated. Not all events in the pipeline are like the simulated events. Most of the events surrounding the actual triggered event contain nearly no tracks. Figure 7.12 illustrates models for the distribution of events in the pipeline. The best case is the triggered event surrounded by events without any tracks. The worst case is the whole pipeline full of normal events of one event signature. The expected case is the triggered event with previous events with nearly no tracks and events with decreasing number of tracks after the triggered event. Table 7.4 and 7.5 show the expected ratios resulting from the models of the distribution of events in the pipeline for the reduction algorithm 2 and 4. A noise level of 3% was assumed because of the expected low noise level of the chambers. If 3 (5) events are read out from the pipeline the compressed data

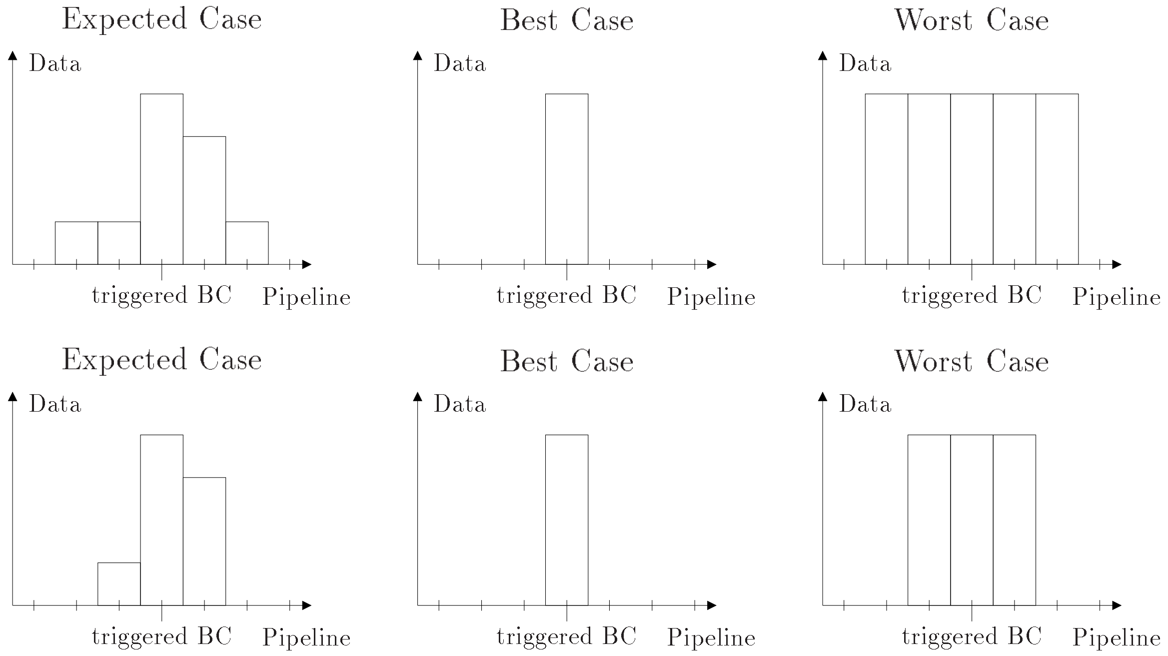


Figure 7.12: *Models for event data in the pipeline. The best case is the triggered event surrounded by events without any tracks. The worst case is the whole pipeline full of normal events of one event signature. The expected case is the triggered event with previous events with nearly no tracks and events with decreasing number of tracks after the triggered event.*

has to be 32% (20%) the size of the original data to get the data size of 1 kbyte. To get the data size of 2 kbyte the compressed data has to be 64% (40%) the size of the original data for 3 (5) events read out from the pipeline. The results show that the goal of a final data size between 1 kbyte and 2 kbyte can be reached. The reduction ratio depends strongly on the distribution of the events in the pipeline. Both reduction algorithms 2 and 4 will be implemented in the final readout system to study both algorithms on real data.

Algorithm 2				
Events read	Event type	Reduction ratio [%]		
		Expected case	Best case	Worst case
3	High q^2	27.95	22.31	41.93
	Low p_T	32.11	24.39	48.17
	Background	45.11	30.89	67.67
5	High q^2	34.94	18.39	41.93
	Low p_T	40.14	19.63	48.17
	Background	56.39	23.53	67.67

Table 7.4: *Reduction ratio for different models of event distribution in the pipeline for algorithm 2*

Algorithm 4				
Events read	Event type	Reduction ratio [%]		
		Expected case	Best case	Worst case
3	High q^2	30.17	17.59	45.26
	Low p_T	35.50	20.25	53.25
	Background	52.48	28.74	78.72
5	High q^2	37.72	12.05	45.26
	Low p_T	44.38	13.65	53.25
	Background	65.60	18.74	78.72

Table 7.5: *Reduction ratio for different models of event distribution in the pipeline for algorithm 4*

Appendix A

Dictionary of abbreviations

ASIC	**A**pplication **S**pecific **I**ntegrated **C**ircuit
BC	**B**unch **C**rossing
BMA	**B**lock **M**over **A**ccelerator
BPC	**B**ackward **P**roportional **C**hamber
BST	**B**ackward **S**ilicon **T**racker
CIP	**C**entral **I**nner **P**roportional Chamber
CIZ	**C**entral **I**nner **z** Chamber
CJC	**C**entral **J**et **C**hamber
COP	**C**entral **O**uter **P**roportional Chamber
COZ	**C**entral **O**uter **z** Chamber
CPU	**C**entral **P**rocessing **U**nit
CST	**C**entral **S**ilicon **T**racker
CTC	**C**entral **T**rigger **C**ontrol
DAQ	**D**ata **Aq**uisition
DESY	**D**eutsches **E**lektronen **Sy**nchrotron
Discr	**Discr**iminator
DMA	**D**irect **M**emory **A**ccess
DMUX	**D**e**mu**ltiple**x**er
ECL	**E**mitter **C**oupled **L**ogic
EEPROM	**E**lectrically **E**rasable **P**rogrammable **R**ead **O**nly **M**emory
FEPROM	**F**lash **E**rasable **P**rogrammable **R**ead **O**nly **M**emory
FNC	**F**orward **N**eutron **C**alorimeter
FPGA	**F**ield **P**rogrammable **G**ate **A**rray
FPS	**F**orward **P**roton **S**pectrometer
FST	**F**orward **S**ilicon **T**racker
FTD	**F**orward **T**rack **D**etector
FTT	**F**ast **T**rack **T**rigger
HERA	**H**adron **E**lektron **R**ing **A**nlage
GTL	**G**unning **T**ransceiver **L**ogic
HV	**H**igh **V**oltage

IEEE	**I**nstitute of **E**lectrical and **E**lectronics **E**ngineers
IRQ	**I**nterrupt **R**e**q**uest
JTAG	**J**oint **T**est **A**ction **G**roup
L1, L2, L3, L4	**L**evel **1**, **2**, **3**, **4** Trigger
LAr	**L**iquid **Ar**gon Calorimeter
Linac	**Lin**ear **Ac**celerator
LVDS	**L**ow **V**oltage **D**ifferential **S**ignaling
MEB	**M**ulti **E**vent **B**uffer
MMU	**M**emory **M**anaging **U**nit
MUX	**Mu**ltiple**x**er
MWPC	**M**ulti**w**ire **P**roportional **C**hamber
NIM	**N**uclear **I**nstrumentation **M**ethods
PCI	**P**eripheral **C**omponent **I**nterconnect
PECL	**P**ositive **E**mitter **C**oupled **L**ogic
PETRA	**P**ositron **E**lektron **T**andem **R**ing **A**nlage
PLD	**P**rogrammable **L**ogic **D**evice
PMC	**P**CI **M**ezzanine **C**ard
pVIC	**P**CI **V**ertical **I**nter**c**onnect
RAM	**R**andom **A**ccess **M**emory
RLE	**R**un **L**ength **E**ncoding
SCSI	**S**mall **C**omputer **S**ystem **I**nterface
SpaCal	**Spa**ghetti **Cal**orimeter
STC	**S**ubsystem **T**rigger **C**ontrol
Sync	**Sync**hronisator
ToF	**T**ime **o**f **F**light
TTL	**T**ransistor-**T**ransistor **L**ogic
VIC	**V**ertical **I**nter**c**onnect
VITA	**V**MEbus **I**nternational **T**rade **A**ssociation
VME	**V**ersatile **M**odule **E**urope
VSB	**V**ME **S**ubsystem **B**us
zVtx	**z**-**V**er**t**e**x** Trigger

Bibliography

[1] H1 Collaboration. The H1 Detector at HERA. DESY H1-96-01, March 1996.

[2] E. Elsen. The H1 trigger and data acquisition system. H1-01/93-262.

[3] W.J. Haynes. VMEtaxi Mark-2 System Software Package. July 1993.

[4] H1 Collaboration. *ep* Physics beyond 1999. H1-10/97-531, October 1997.

[5] D. Baumeister et al. Progress Report on CIP and Level 1 Vertex Trigger. http://www.physik.unizh.ch/groups/grouptruoel/cipupgrade/cip.html, June 1998.

[6] A. Schweizer. Optimierung eines z-Vertex-Triggers für den H1-Detektor bei HERA. February 1999.

[7] J. Becker. H1 CIP-Vertex-Trigger upgrade – Triggerhardware-Simulation. http://www.desy.de/~jbecker, March 1998.

[8] Philips Semiconductors. The I^2C-Bus Specification. Version 2.1, January 2000.

[9] D. Baumeister. CIPiX User Manual. http://wwwasic.kip.uni-heidelberg.de/h1cip/, 1998.

[10] M. Urban. Ein schneller Trigger für H1 bei HERA. May 2000.

[11] Altera. APEX 20k Programmable Logic Device Family. http://www.altera.com, March 2000.

[12] IEEE – Institute of Electrical and Electronics Engineers. 1596.3-1996 IEEE Standard for Low-Voltage Differential Signals (LVDS) for Scalable Coherent Interface (SCI) 1996 . http://www.ieee.org, 1996.

[13] National Semiconductor. LVDS Owner's manual – Moving info with LVDS. http://www.national.com, January 2000.

[14] VITA – VMEbus International Trade Association. VME64 (ANSI/VITA 1-1994). http://www.vita.com, 1994.

[15] Lattice Semiconductor Corporation. Specifications ispLSI 1048E. http://www.latticesemi.com, October 1998.

[16] Lattice Semiconductor Corporation. 1000EA, 1000E and 1000 family architectural description. http://www.latticesemi.com, January 2000.

[17] IEEE – Institute of Electrical and Electronics Engineers. 1149.1B-1994 IEEE Supplement to Standard Test Access Port and Boundary-Scan Architecture. ISBN 1-5593-7497-7, http://www.ieee.org, 1994.

[18] Altera. IEEE 1149.1 (JTAG) Boundary-Scan Testing in Altera Devices. http://www.altera.com, September 2000.

[19] H. Krehbiel. The H1 trigger control system. H1-12/88-101, September 1988.

[20] H. Krehbiel. The fast card of the subsystem trigger controller. October 1989.

[21] J. Olszowska. The slow card of the subsystem trigger controller. August 1990.

[22] H. Krehbiel. The extended fanout card of the H1 STC. July 1991.

[23] CES – Creative Electronic Systems. RIO 8062 Power PC based RISC I/O Board. February 2000.

[24] CES – Creative Electronic Systems. pVIC Systems 4025/4425/7225/8025/8425/8426 PCI to PCI connections. June 1992.

[25] CES – Creative Electronic Systems. VIC 8250 VMV to VME One Slot Interface. June 1992.

[26] LynuxWorks. LynxOS User's Guide. http://www.lynuxworks.com, 1998.

[27] IEEE – Institute of Electrical and Electronics Engineers. 1003.1-1990 Information Technology – Portable Operating System Interface (POSIX). http://www.ieee.org, 1990.

[28] D. Lewine. POSIX Programmer's Guide. March 1994.

[29] B.O. Gallmeister. POSIX.4 – Programming for the Real World. November 1998.

Acknowledgements

I would like to thank the many people I had interesting and fruitful discussions with during the work on this thesis. In particular, I would like to thank:

- Prof. Dr. Franz Eisele and Prof. Dr. Ulrich Straumann for giving me the opportunity to work within the H1 collaboration at DESY.
- Prof. Dr. Volker Lindenstruth for beeing the second referee for this thesis.
- Dr. Felix Sefkow and Jürgen Kroseberg for the revision of this thesis and for the help during my work at the H1 experiment.
- Dr. Alan Campbell for helping me with the setup of the test system.
- Albert Rausch and Torsten Kuelper for the help in all technical questions.
- Dr. Hans-Christian Schultz-Coulon, Dr. Daniel Pitzl, Dr. Martin Zur Nedden and Dr. Guenter Eckerlin for all the discussions about triggers, readout and the H1 experiment.
- Max Urban and all other CIP2000 project members for the good teamwork.

Erklärung:

Ich versichere, daß ich diese Arbeit selbständig verfaßt und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Heidelberg, den

(Unterschrift)