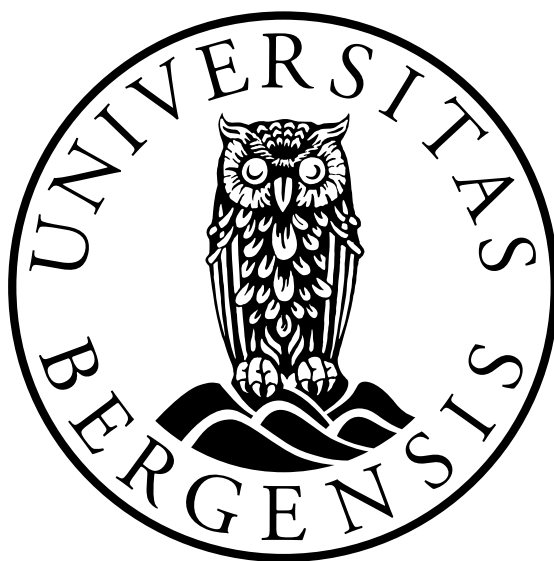


The ALICE TPC Readout Electronics

Design, performance optimization and verification of the DAQ circuit

Attig ur Rehman



Dissertation for the degree of Philosophiae Doctor (PhD)

Department of Physics and Technology
University of Bergen

October 2012



Acknowledgements

Today, while writing this page I am thinking about all those gem like personalities I have met during my stay at CERN and UiB. Starting from the recent most, first of all I am indebted and thankful to my supervisors Kjetil Ullaland and Dieter Røhrich for the opportunity to complete my studies at University of Bergen. Kjetil Ullaland's guidance and occasional discussions has provided me new insights to improve and enhance my skills in electronics design engineering. Special thanks to Dieter Røhrich for his prompt responds resolving the administrative issues, and to Bianca Ross for helping me to solve the practical issues of the first days in Bergen.

My gratitude to Luciano Musa, Peter Braun-Munzinger and Harald Appelshaeuser for arranging the financial matters during my Ph.D. studies and stay in Bergen. I owe sincere and earnest thankfulness to Luciano Musa, for his supervision throughout my work at CERN. Much obliged for his simulating guidance, excellent teachings and the training (driven by personal example), these are among the best professional and personal assets of my life. I am obliged to Christian Lippmann for his corrections, comments and suggestions to improve this dissertation. His patience and pleasant guidance at ALICE ACR, while debugging and commissioning of the RCU firmware has been highly valuable. Meeting Johan Alme's for the first time was pleasant and technically enriched same as the recent most, always thankful for his valuable advices related to the RCU design during my initial days at CERN. I am thankful to Ketil Røed, his guidance related to the radiation effects on the RCU and irradiation tests of electronics components was always helpful. Special thanks to Dominik Fehlker for his support in practical matter during my stay in Bergen and helpful support related to the DCS board related issues. I am thankful to Csaba Soos and Torsten Alt for sharing their experiences and always being helpful in problems related to the FPGA based hardware design. I am grateful to Filippo Costa for his contributions and working with him has been always a pleasure.

To name all of my coffee mates at CERN would be out of scope, but the most memorable ones are Magnus Mager and Stefan Rosseger. Their company to discuss any technical or a philosophical issues is like "guaranteed to enrich" every moment spent with them!

Last but not least, I have been grateful in all my life to Muhammad Shahid Rafique, Zaheer Abbas Shamsi and Chong ee Chin for their valuable support and kind help in those days when I was starting to learn the principles of life and learning in Malaysia.

Abstract

ALICE (A Large Ion Collider Experiment) is a dedicated heavy-ion experiment at CERN's LHC (Large Hadron Collider). It is designed to study the physics of strongly interacting matter and the quark-gluon plasma in heavy-ion collisions. It contains a large volume Time Projection Chamber (TPC) as its main tracking device. The ALICE TPC is the largest ever built gaseous TPC, both in terms of dimensions and number of read-out channels (557,578). A total number of 128 channels are packed in one TPC Front End Card (FEC) and 4,356 FECs are distributed over 216 independent readout partitions. Each readout partition steered by a single Readout Control Unit (RCU) functions as an independent unit in the data acquisition system of the TPC. The RCU functions as an interface between the FECs, Data Acquisition system (DAQ), the Trigger and Timing Circuit (TTC) and the Detector Control System (DCS). The ALICE TPC readout electronics is in operation since the start of the LHC in November 2009. The primary objectives of the work carried out for this thesis includes the design and implementation the RCU firmware. The RCU firmware is used by four different ALICE detectors including TPC, PHOS, FMD and EMCAL. Design and implementation of the RCU firmware required flexibility and configurability as a key challenge to meet the requirements of these different detectors. Novel techniques to verify the integration of the RCU firmware with other components of the readout system are developed and set as standard procedures for its commissioning to be used at the ALICE experiment.

The optimization of the readout electronics to meet the required performance of the TPC data acquisition system was the second key objective of this thesis. Several techniques and methods to improve the readout rate were implemented in the RCU firmware and its interface to FECs. The performance of the TPC readout electronics is characterized and evaluated. The measured performance of the ALICE TPC during p-p and Pb-Pb collisions is presented in this thesis.

Robustness and reliability of the TPC readout system is addressed as the third main objective of this thesis. The RCU firmware is featured to monitor and detect erroneous situations of all its interfaces involved in an event readout, throughout all their transactions. These features and error handling mechanisms are verified and presented along with their results. The RCU firmware was subjected to fault injection test to characterize the effects of SEUs during the event readout. A new technique to correct the SEU effects in any readout partition was developed, implemented and verified as a part of the work carried out for this thesis. This technique ensures that the TPC FEE does not cause an unplanned termination of the data taking session in the ALICE experiment.

Contents

Acknowledgements	i
Abstract	iii
1 The ALICE Experiment	1
1.1 Introduction	1
1.2 The detectors of ALICE	3
1.3 Experiment control system	4
1.4 Detector control system	4
1.5 Central trigger processor	5
1.6 Data acquisition system	6
1.7 High level trigger	7
2 The ALICE Time Projection Chamber (TPC)	9
2.1 Introduction	9
2.2 Working principle of the TPC	10
2.3 Field cage and the drift volume	10
2.3.1 Inner and outer containment vessels	11
2.3.2 Inner and outer field cage vessels	11
2.3.3 The high voltage central electrode	11
2.4 Readout chambers	11
2.5 Gating grid system	12
2.6 Calibration systems	12
2.6.1 Calibration pulser	12
2.6.2 Laser calibration	13
2.7 Front end electronics of the TPC	13
2.7.1 Front end card	14
2.7.1.1 Preamplifier shaper chip	14
2.7.1.2 ALICE time projection chamber readout chip	15
2.7.1.3 Board controller	16
2.7.2 The RCU motherboard	16
2.7.2.1 DCS card	17
2.7.2.2 SIU card	18
2.7.3 Busy box	18
3 RCU Firmware	19
3.1 Functional requirements	19
3.2 Functional description	20
3.2.1 Clock management	20

3.2.2	Reset and initialization of the system	20
3.2.3	Configuration and verification	21
3.2.4	Monitoring and safety	22
3.2.5	Trigger handling	22
3.2.5.1	Test trigger	22
3.2.5.2	CTP trigger	23
3.2.6	Event readout	23
3.2.6.1	Full readout	24
3.2.6.2	Sparse readout	25
3.2.7	Data format	26
3.3	Architecture of the firmware	28
3.3.1	RCU clock manager	29
3.3.2	ALTRO interface module	29
3.3.2.1	ALTRO bus driver	30
3.3.2.2	Channel readout unit	30
3.3.2.3	Event readout unit	31
3.3.3	Instruction sequencer	31
3.3.4	Result unit	31
3.3.5	RCU decoder	34
3.3.6	Event manager	36
3.3.7	Data assembler	37
3.3.8	Monitoring and safety module	38
3.3.9	SIU interface unit	39
3.3.10	DCS interface unit	39
3.3.11	Trigger receiver unit	40
3.4	Implementation of the RCU firmware	40
3.5	Simulation and verification of the RCU firmware	40
3.5.1	Simulation and functional verification	42
3.5.1.1	Simulation controller	42
3.5.1.2	DCS stimuli process	42
3.5.1.3	SIU stimuli process	43
3.5.1.4	CTP stimuli process	44
3.5.1.5	FEC stimuli process	44
3.5.1.6	I ² C stimuli process	45
3.5.2	Hardware test of the complete readout system	45
3.5.2.1	Hardware setup for the test	45
3.5.2.2	Hardware test for the compatibility of complete readout system	46
3.5.2.3	Hardware test with real time data verification	47
3.5.3	Final qualification and laser run	49
3.6	Conclusion	50
4	Characterization and performance evaluation of the ALICE TPC readout system	51
4.1	Characterization of the TPC readout system	51
4.1.1	Configuration of the readout electronics	51
4.1.2	Trigger and clock synchronization	52
4.1.3	Monitoring and safety	54
4.1.4	Event readout	55

4.2	Evaluation of readout performance in the test set-up	57
4.2.1	Test methodology	57
4.2.1.1	Full readout	57
4.2.1.2	Sparse readout	60
4.2.1.3	Comparison of full and sparse readout modes	65
4.2.2	Test of multi-event buffering and readout	65
4.2.2.1	Objectives of the test	67
4.2.2.2	Test scheme and hardware set-up	67
4.2.2.3	Results and analysis	69
4.2.2.4	Evaluation of performance with multi-event buffering	69
4.3	Performance of the TPC readout at the ALICE experiment	71
4.3.1	Readout performance during p-p collisions	72
4.3.1.1	Readout partitions 0 and 2	72
4.3.1.2	Readout partition 1	73
4.3.1.3	Readout partitions 3,4 and 5	74
4.3.1.4	Performance summary for p-p collisions	74
4.3.2	Readout performance during Pb-Pb collisions	75
4.3.2.1	Readout partitions 0 and 2	75
4.3.2.2	Readout partition 1	77
4.3.2.3	Readout partitions 3, 4 and 5	77
4.3.2.4	Performance summary for Pb-Pb collision	77
4.4	Summary	78
5	Stability and robustness of the ALICE TPC readout system	81
5.1	Robustness of the TPC readout electronics	81
5.1.1	The ALTRO interface	82
5.1.1.1	Continuous monitoring of the ALTRO control lines	82
5.1.1.2	Monitoring of the ALTRO handshake protocol	83
5.1.1.3	Verification of the trailer word from the ALTRO channel	84
5.1.1.4	Test methodology	84
5.1.2	Handling of erroneous trigger sequences	84
5.1.3	The DDL interface	85
5.1.3.1	Handling of the X_{off}	86
5.1.3.2	Test methodology	86
5.2	Radiation effects and the TPC readout electronics	86
5.2.1	Radiation mitigation in the ALTRO chip	87
5.2.2	Radiation mitigation in the board controller	87
5.2.3	Radiation mitigation in the DCS card	87
5.2.4	Radiation mitigation in the SIU card	87
5.2.5	Radiation mitigation in the RCU	88
5.2.6	Measurement of SEUs in the TPC readout electronics	91
5.2.7	Handling of radiation effects on the TPC readout electronics	94
5.2.7.1	Periodic and interrupt driven reconfiguration	94
5.2.7.2	Configuration of the readout partition during readout	96
5.3	Summary	97
6	Conclusions and Outlook	99
A		103

B	111
List of Abbreviations	113

List of Figures

1.1	The location of the four main LHC experiments	1
1.2	Evolution of the universe and cosmological transition phase of matter after the Big Bang [15]	2
1.3	The ALICE experiment	3
1.4	The ALICE experiment control system and its hierarchy	4
1.5	The ALICE detectors and the DCS hierarchy	5
1.6	The ALICE central trigger processor and related sub systems	6
1.7	Hierarchy of The ALICE data acquisition system	6
1.8	Functional overview of the ALICE high level trigger	7
2.1	An overview of the ALICE TPC	9
2.2	Working principle of the ALICE TPC	10
2.3	Pulser systems of the TPC	12
2.4	3D view of the laser tracks in the drift volume [30]	13
2.5	Top level architectural view of the TPC front end electronics	14
2.6	Single front end card (packed with 128 channels)	15
2.7	The ALTRO single channel data processing and transport chain. Each ALTRO integrates 16 channels and a single FEC encapsulates 128 channels	16
2.8	The RCU motherboard; with its daughter cards and interface to FECs	17
3.1	Flow chart of the full readout	24
3.2	Parallel processes during the sparse readout	25
3.3	Flow chart of the sparse readout	26
3.4	The ALTRO data format [19]	27
3.5	Top level architecture of the RCU firmware	29
3.6	Components of clock manager	29
3.7	Sub-modules of the ALTRO interface module	30
3.8	Instruction sequencer and result unit	32
3.9	RCU decoder and its sub modules	35
3.10	Structure of the event manager	36
3.11	Sub-modules of the data assembler	37
3.12	Monitoring and safety module top level	38
3.13	SIU interface unit top level	39
3.14	DCS interface unit top level	39
3.15	Trigger receiver unit top level	40
3.16	Relative size and utilization of the logical resources by modules of the RCU firmware	41
3.17	Architecture of the RCU test bench	43
3.18	Hardware setup of one single TPC readout partition	46

3.19	Test set-up for hardware test	46
3.20	Flow chart of real time data verification	48
3.21	Final qualification of RCU firmware by the laser run	50
4.1	Timing diagram of a single channel write transaction	52
4.2	ALICE trigger synchronization with the TPC sampling clock	53
4.3	Synchronization of the TPC pulser systems	54
4.4	Communication between the RCU and the FEC board controller on the FC bus	55
4.5	Fixed time overhead of the channel readout command	56
4.6	Time to execute EVLRDO command to board controller	57
4.7	Relation of number of samples per channel with readout time and event size	58
4.8	Measurements of full readout in test set-up for the partition with maximum number of channels	59
4.9	Time profile of Sparse readout (SRDO) for a single FEC (128 channels)	61
4.10	Channel hits, event size and readout time	62
4.11	Channel hits, dead-time and events size	62
4.12	Measured readout times for all 6 readout partitions for random event size with sparse readout mode	64
4.13	Comparison of full and sparse readout for increasing number of channel hits	66
4.14	The principle of multi-event buffering with four buffers and event readout	67
4.15	Calibration pulser to test multi-event buffering in the TPC	68
4.16	Storage of time shifted pulses in MEB according to test scheme	68
4.17	Test results of MEB functionality	70
4.18	Effect of different setting of MEB on readout rate	71
4.19	Average readout times for six different readout partitions during p-p collisions	73
4.20	Average readout performance of 6 readout partitions during p-p collisions	75
4.21	Average readout times for six different readout partitions during Pb-Pb collision	76
4.22	Comparison of three partitions with equal number of channels	77
4.23	Readout time vs centrality of Pb-Pb collision	78
5.1	Flow chart of the fault injection test	89
5.2	Bitmap of the injected faults causing an error in the event readout	89
5.3	Total faults injected and the failures in the RCU	90
5.4	The measurements of SEUs in the ALTRO chips [17]	92
5.5	The measurements of SEUs in the RCU firmware [27]	93
5.6	Recovery procedures for FEC related errors	95
5.7	Summary of EoR reasons for the period of 30 days from the ALICE E-logbook	98
A.1	Two examples of script files to simulate the RCU firmware	103
A.2	Waveform view of the simulation in progress for the ALTRO bus interface	105
A.3	Decoded event data by the SIU stimuli process, received from the RCU firmware (being simulated)	106
A.4	Screen snap shot of the stress test with 4 buffers at high trigger rate	107
A.5	Bit-wise event data verification of the RCU firmware using collisions data for all six readout partitions	108
A.6	Xilinx VP2 device utilization after final implementation of RCU firmware	109

List of Tables

3.1	Commands to issue resets of different hierarchy	21
3.2	Structure of the event data block	27
3.3	ALICE common data header	27
3.4	Structure of the event payload	28
3.5	Structure of RCU trailer	28
3.6	Format of instructions loaded in instruction memory	31
3.7	Internal instruction set of the sequencer	32
3.8	Result of a FEC read transaction	32
3.9	Result of a FEC write transaction	33
3.10	Result of a FEC command transaction	34
3.11	Result of the ISEQ wait internal command	34
3.12	Result of an internal loop instruction	34
3.13	Result of end sequence	34
3.14	Configuration registers of the RCU, for detailed description see [23]	35
3.15	The MSM command and register set	38
3.16	Device utilization summary for the targeted device: 2vp7ff672-7	42
3.17	Commands for simulation tasks	44
3.18	Compatibility tests of the complete integrated readout system	47
4.1	The configuration of the ALTRO sampling clock in ALTRO interface register [23]	53
4.2	Frequency and start delay selection of the FC bus (for details [23])	55
4.3	Full readout time with number of samples and event size (for complete table see appendix B, table B.2)	60
4.4	Measured timing of four steps in sparse readout	61
4.5	Event data used in performance evaluation	71
4.6	Average Readout time for p-p collisions of all partitions	74
4.7	Average readout time for Pb-Pb collisions for all partitions	78
5.1	Status bits in two registers (“ALTBUSSTATUS” and “ALTBUSST_TRSF” [23]) for the ALTRO bus control lines	82
5.2	Bit level details of the error and status register of the ALTRO interface	83
5.3	Results of the fault injection test [7]	90
5.4	Details of the bits to signal the PAC request in the ALICE CDH	97
5.5	The longest data taking sessions of the ALICE experiment (September 2010 - July 2012) [4]	97
A.1	Final qualification of the RCU firmware at the ALICE TPC	104

B.1	Comparison of readout time for sparse and full readout mode with increasing number of channel hits	111
B.2	Full readout time with number of samples and event size	112

Chapter 1

The ALICE Experiment

An overview of CERN and the Large Hadron Collider (LHC) is presented in this chapter. It includes a brief description of the ALICE experiment and its sub systems. A short summary of the key objectives of ALICE physics is given in the following section. The detectors of the ALICE experiment are summarized in one section whereas the systems which are common to all detectors are discussed in their respective sections.

1.1 Introduction

The LHC is located at CERN near Geneva on the boarder between France and Switzerland. The LHC is a particle accelerator which is built 100 meter underground in a circular tunnel with circumference of 27 km. The particles accelerated in the form of rotating beams in opposite directions are steered such that they intersect each other allowing for the collisions at four different locations. These collisions are studied by four different experiments installed at the respective locations as shown in figure 1.1.

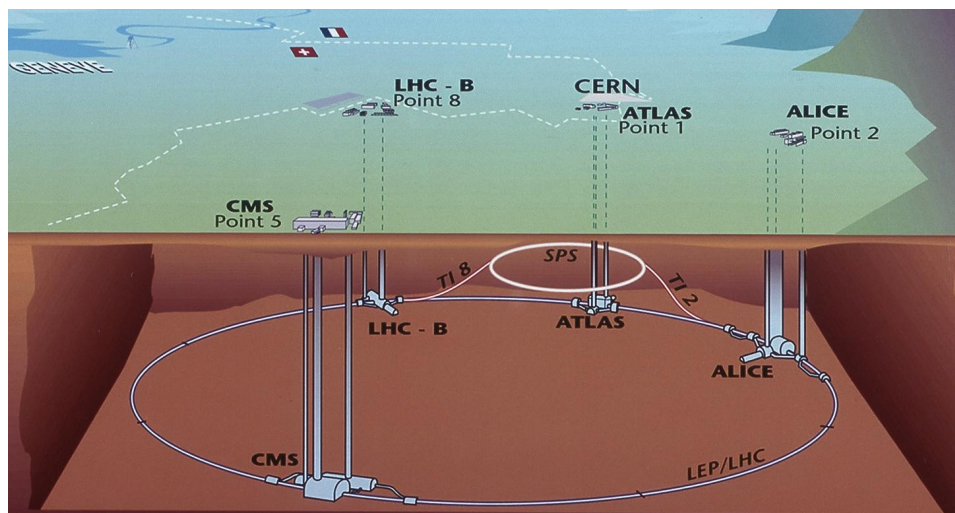


Figure 1.1: The location of the four main LHC experiments

- **ALICE (A Large Ion Collider Experiment)**

The ALICE experiment is dedicated to study heavy ion collisions and the physics related to the generation of the Quark Gluon Plasma (QGP), where quarks and gluons are no

longer bound to each other as in hadrons. The ALICE experiment is the main topic of this chapter.

- **ATLAS (A Toroidal LHC ApparatuS)**

The main goal of the ATLAS experiment is to discover the Higgs particle, investigate its properties and investigate the physics at TeV level. Its main goals include the confirmation or improved measurements of the standard model.

- **CMS (Compact Muon Solenoid)**

The CMS experiment is dedicated to the search for the Higgs particle like the ATLAS. Its other objectives include the study of physics beyond the standard model, such as looking for the evidence of super symmetry.

- **LHCb (LHC beauty)**

The LHCb experiment is focused towards B physics. The aim of this experiment is to study rare decays via precise measurements of Charge Parity CP violating effects in B hadrons.

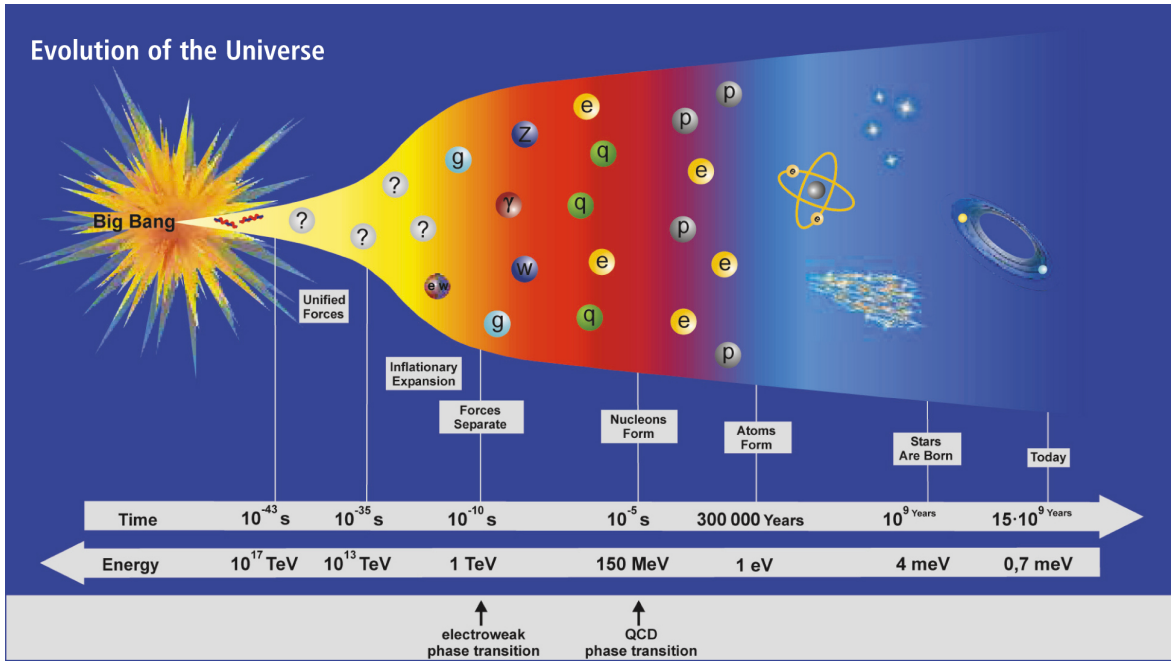


Figure 1.2: Evolution of the universe and cosmological transition phase of matter after the Big Bang [15]

The conceptual evolution of the universe following the Big Bang and the phase transition of the matter to reach the current form is represented in Figure 1.2. As described in the Big Bang theory, the life time of the Quark Gluon Plasma is anticipated as 0.5 micro seconds after the creation of the universe. The drop in pressure and temperature initializes a process called freeze out, in which the quarks and the gluons are bound together to form different kinds of hadrons. These hadrons include the hadrons which constitutes the basic building blocks of atomic nuclei. The ALICE experiment is specifically designed to study heavy ions collisions like Pb-Pb at the LHC. Central Pb-Pb collisions possess enough energy to create the Quark Gluon Plasma. The high temperature achieved in the collision frees the quarks

from their hadron bags into a hot plasma of quarks and gluons which in turn undergoes a phase transition from the Quark Gluon Plasma back into a hadron gas. The observation of the transient quark gluon plasma state is not directly possible. Its existence and its properties can only be detected by either penetrating probes like photons or by its footprints in the hadronic final state. The ALICE experiment provides momentum measurement of charged particles, energy measurement of neutral particles and Particle IDentification (PID) within a limited phase space coverage.

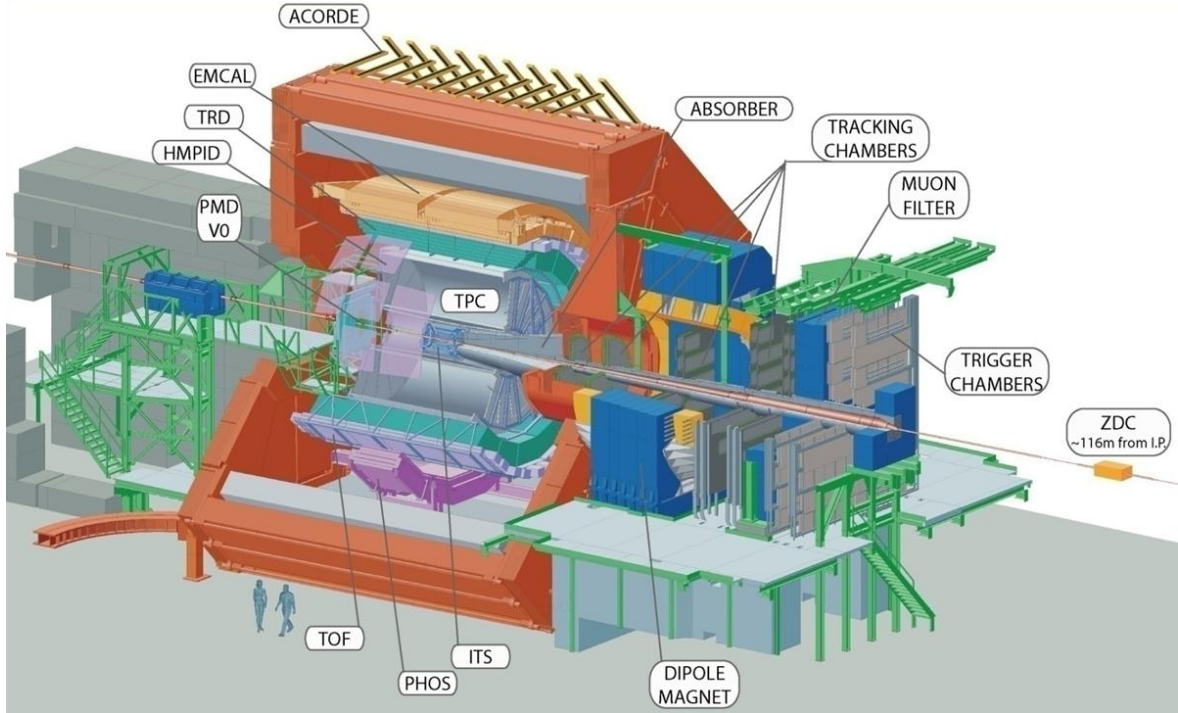


Figure 1.3: The ALICE experiment

1.2 The detectors of ALICE

The ALICE experiment has been built by a collaboration of over 1000 physicists and engineers from 105 institutes distributed over 30 countries. Its overall dimensions are $16 \times 16 \times 26 \text{ m}^3$ with a total weight of approximately 10,000 tons. A schematic layout of the ALICE detector setup is presented in figure 1.3. All of the sub detector systems of the experiment are described in detail in their respective Technical Design Reports. A short summary of the sub detectors and the layout based on [1] is presented here. The central system of ALICE is installed inside a large solenoidal magnet (L3-magnet) which provides the main magnetic field of 0.5 T. The central barrel contains an Inner-Tracking System (ITS) and a cylindrical Time-Projection Chamber (TPC). The ITS consists of six planes of high-resolution silicon pixel (SPD), drift (SDD) and strip (SSD) detectors. It provides high resolution tracking and precise vertex reconstruction. The tracking is performed by combining the information from the TPC, the main tracking detector with the information coming from the other detectors. Three particle identification detectors consisting of the Time Of Flight (TOF), the Ring-Imaging Cerenkov (HMPID) and the Transition-Radiation Detector (TRD) are installed within the L3 magnet. The TOF and HMPID are devoted to the identification of hadrons at intermediate momentum,

using two different techniques. The information from the TRD is used for the identification of electrons. The particle identification is enhanced further through the measurement of the specific energy loss by the central tracking detectors (the TPC and the ITS). A highly granular crystal calorimeter (PHOTon Spectrometer — PHOS) is used for the measurement of photons. The electromagnetic calorimeter (EMCal) is used to measure particle jets. All detectors except HMPID, PHOS and EMCal cover the full azimuth in the central barrel. There are several other smaller detectors (FMD,PMD,Dimuon Spectrometer,T0,V0,ZDC) located at small angles, which are used for global event characterization and trigger generation.

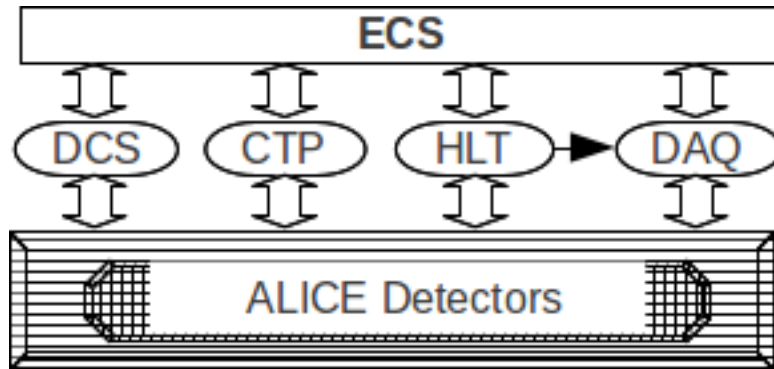


Figure 1.4: The ALICE experiment control system and its hierarchy

1.3 Experiment control system

The Experiment Control System (ECS) is the top level framework to control the ALICE experiment as shown in the figure 1.4. ECS is responsible of performing a set of synchronized activities on the “online systems” that control the operation of the detectors. The “online systems” of the ALICE include the Central Trigger Processor (CTP), the Detector Control System (DCS), the Data AcQuisition system (DAQ) and the High Level Trigger (HLT). To ‘Run’ the experiment implies to control and operate all detectors and the online systems of ALICE. A unified interactive view of the sub detector systems and related online systems is provided to facilitate the operator to conduct multiple operations on the experiment. ALICE is operated under a hierarchical architecture, in which the ECS acts as a top level layer between the “online systems” and all of the detectors of the ALICE experiment. The ALICE experiment can be operated either as a whole (during the LHC operation for the physics production) or independently for different detectors (partitions). Independent operation of the detectors is important for the installation, commissioning or testing of the system. A partition can contain one or more detectors and can be operated autonomously and concurrently.

1.4 Detector control system

The primary task of the ALICE DCS is to ensure the safety of the ALICE experiment during its operation. All of the experimental equipment is monitored, controlled and operated remotely via the DCS. The DCS provides an interface for monitoring and control of all services and parts of each detector. The services include gas systems, cooling, ventilation and control of the front end electronics. The DCS is a heterogeneous system where the tasks are distributed over multiple resources including personal computers and embedded computing devices. The

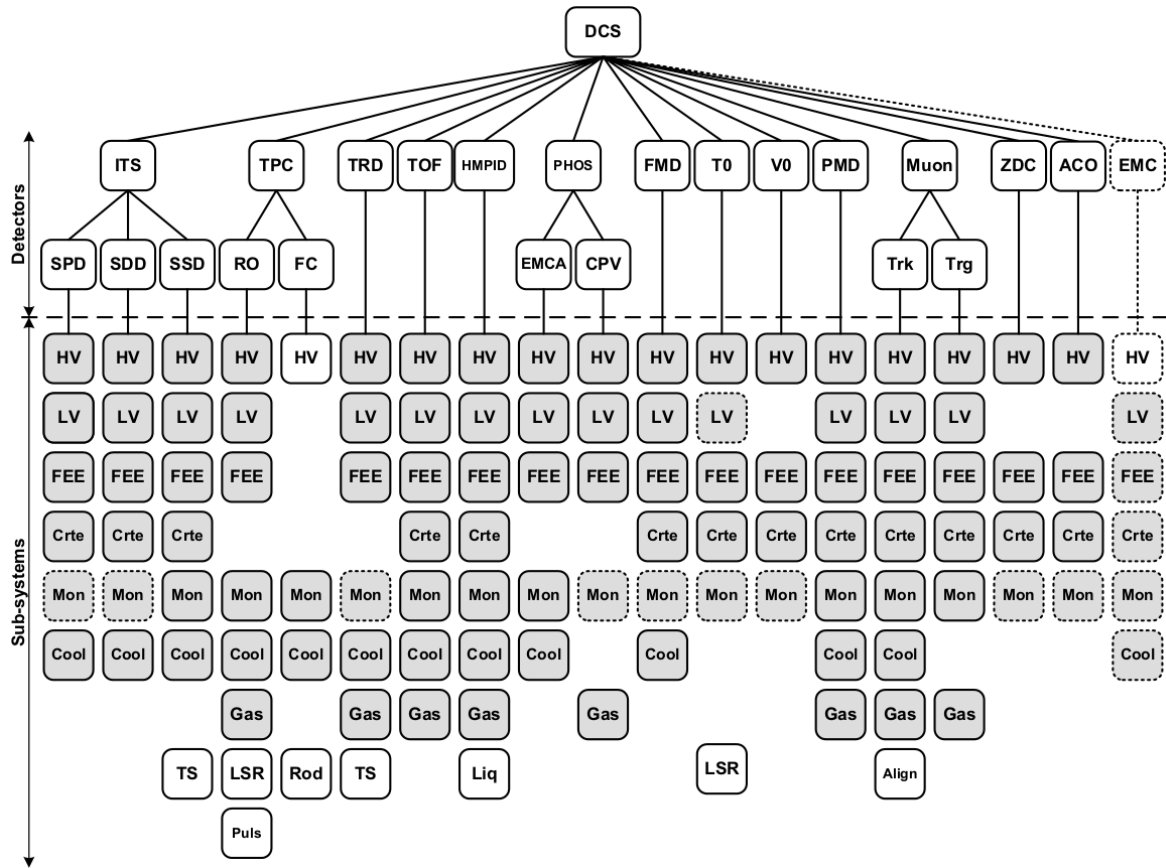


Figure 1.5: The ALICE detectors and the DCS hierarchy

DCS is based on a configurable and scalable architecture to ensure and allow independent and parallel operation of all the different components. The supervisory layer of the DCS is implemented in a commercial Supervisory Control and Data Acquisition (SCADA) system: Prozess Visualisierungs- und Steuerungs-System (PVSS) [16]. The DCS is structured in three physical and logical layers: 1) Supervisory (PVSS) 2) Control (e.g. intercom layers and FEE server) 3) Field layer (e.g. RCUs and FECs). It provides a user interface to perform the different tasks related to the operation of the experiment. The control layer connects to different detectors and their sub systems, updating the supervisory control layer with received information concerning the status of the system, or responding to commands issued. The DCS can be utilized for the configuration of the Front End Electronics (FEE) of the detectors. The monitoring and control system of the detectors is detached from the event readout data flow to keep a redundant access to the front end electronics. The thresholds of critical operational parameters for example high voltage, current, pressure and temperature of the sub systems for all detectors are configured for their normal values. These parameters are measured and updated continuously to the supervisory layer of the DCS to confirm the safe operation of the experiment.

1.5 Central trigger processor

The ALICE CTP is designed to process and synchronize the information related to an event from all the triggering detectors in ALICE. The CTP is responsible to send the correct sequences of trigger signals to all detectors to perform the readout of interesting events. It also

facilitates the requests for the calibration triggers from the detectors and generates special trigger sequences to perform pre-defined tasks. The basic interface of the CTP in the complete system is shown in figure 1.6. The trigger information from the trigger detectors and LHC clock signals are the inputs to the CTP, which are processed to create trigger decisions. The ECS interacts with the CTP to start and stop the generation of trigger sequences. The CTP receives a “busy” signal from the readout detectors, informing that a detector is not able to handle any more trigger due to the handling of the previous triggers. Trigger sequences sent from the CTP are discussed in section 3.2.5.2.

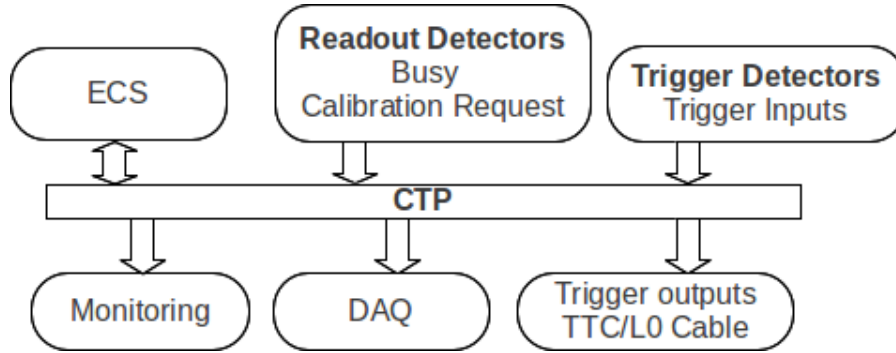


Figure 1.6: The ALICE central trigger processor and related sub systems

1.6 Data acquisition system

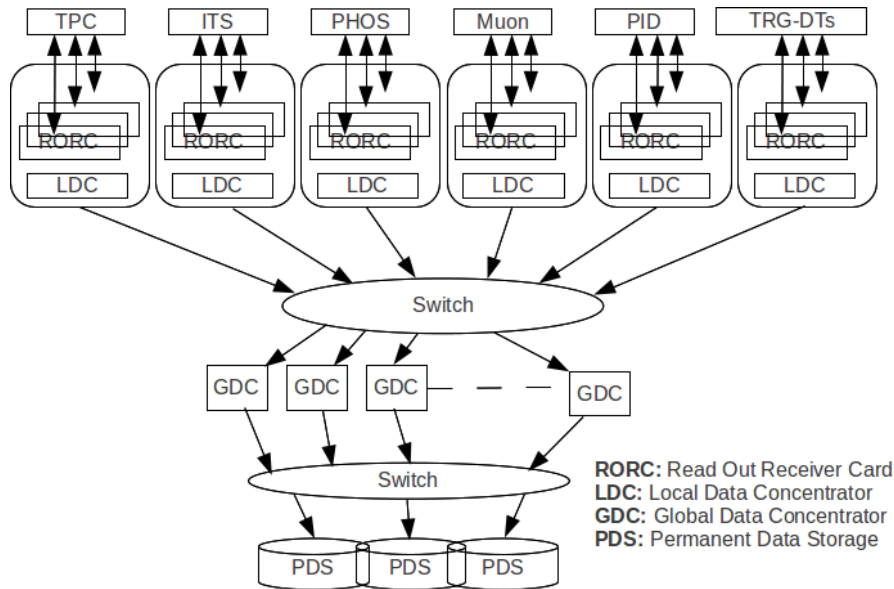


Figure 1.7: Hierarchy of The ALICE data acquisition system

The ALICE Data Acquisition System (DAQ) system realizes the data link between the front end readout electronics of the detector up to the permanent data storage and to the HLT computing farm. The complete chain of the DAQ consists of the hardware and the software interface of the DAQ to the readout electronics of the detector. The DAQ hardware consists of an optical interface called detector data link (DDL). On the FEE it includes the Source Interface

Unit (SIU) card. On the receiver side it includes the Data Interface Unit (DIU) mounted on the ReadOut Receiver Card (RORC) which is hosted on a Peripheral Card Interface (PCI) interface of a personal computer named as Local Data Concentrator (LDC). The SIU card is connected to the DIU via the DDL. Each of the LDC can interface to multiple RORCs, each RORC can host a maximum of two DDLs as shown in the figure 1.7. The LDCs are connected to the Global Data Concentrators (GDC) via standard switches to form the event building network. A number of checks on the event data received at GDC level are performed to verify the consistency and availability of the event fragments from all of the LDCs of the respective detectors before the event is saved to the permanent storage.

1.7 High level trigger

The High Level Trigger (HLT) combines and processes the event data online from all major detectors of ALICE in a large computing cluster. The prime objective of the online processing of the data is to select the relevant part of the huge amount of incoming data and to compress the data volume to fit the available storage bandwidth. The HLT accomplishes its objectives by performing following functions on the event data.

- **Trigger**
Interesting and important events based on detailed online analysis of its physics observables can be selected to be saved in permanent storage.
- **Selection**
Regions of Interests (significant fractions of single events) can be configured to select certain parts of an event.
- **Compression**
The event size can be reduced by advanced data compression without any loss of the contained physics.

Data processing and online reconstruction of the event data is performed such that the physics information is preserved and available to evaluate the importance of the event or any specific part of it based on the physics objectives. As shown in figure 1.8 data processing is performed

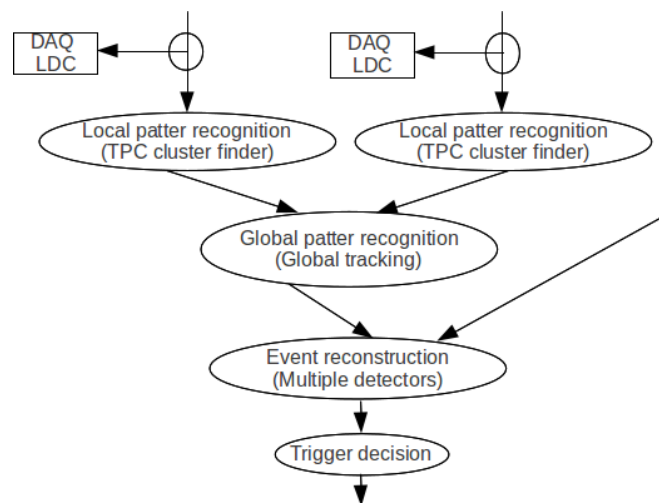


Figure 1.8: Functional overview of the ALICE high level trigger

at different stages in the hierarchy and the complete information is utilized to generate the trigger to store the event.

Chapter 2

The ALICE Time Projection Chamber (TPC)

The main objectives and working principle of the ALICE TPC is discussed in this chapter. An overview of its sub systems and the basic building blocks are presented with emphasis on the core features and functions of the front end electronics.

2.1 Introduction

The ALICE experiment incorporates the TPC as its main device for tracking and particle identification. It is installed in the central barrel of the ALICE experiment.

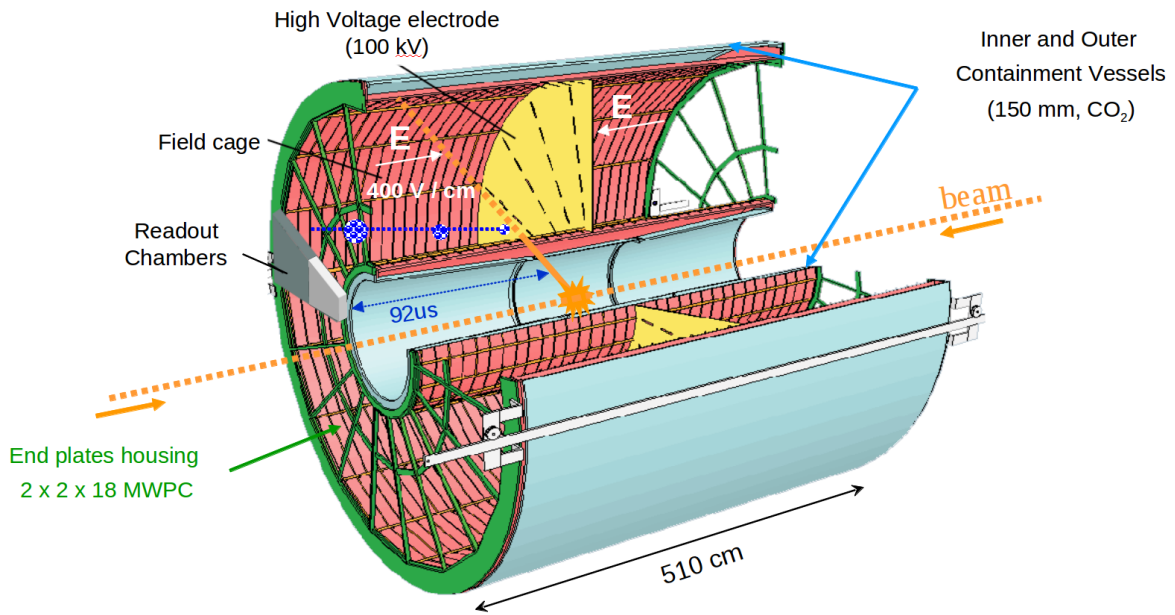


Figure 2.1: An overview of the ALICE TPC

The function of the TPC can be considered as a 3 dimensional imaging device utilized to record the ionization of the gas by the charged particles crossing it in order to track the high energy particles in the drift volume. The TPC is used for the identification of the particles produced in collisions through the accurate measurement of their energy loss in the medium. The architectural layout of the ALICE TPC is shown in Fig 2.1. The cylindrical shaped gas volume has a size of 88 m^3 which is filled with 90% Ne - 10% CO_2 . The main building blocks of the ALICE TPC are described in following sections.

2.2 Working principle of the TPC

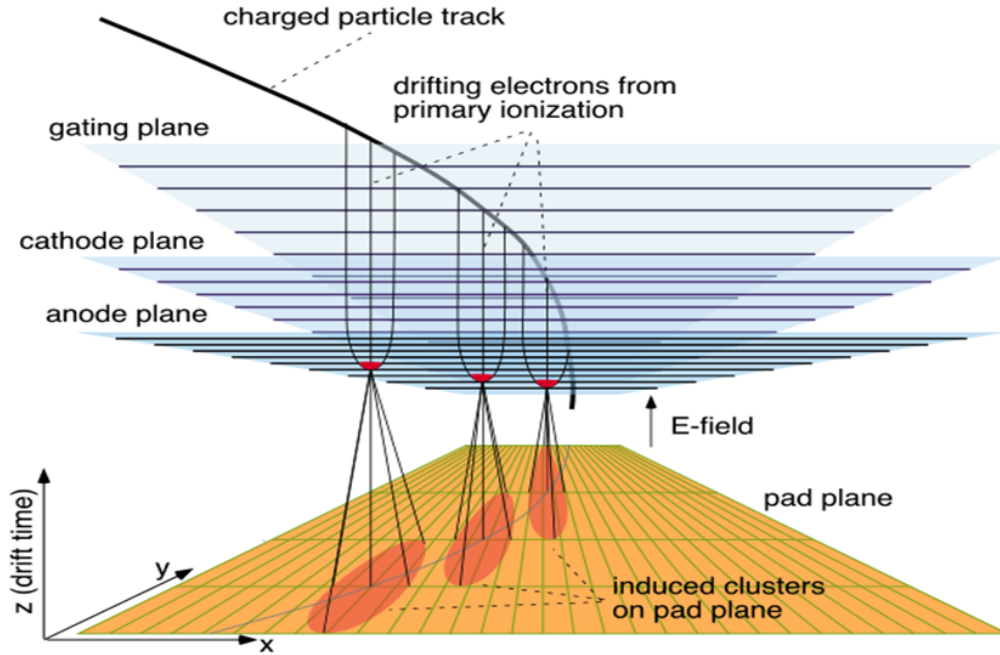


Figure 2.2: Working principle of the ALICE TPC

A charged particle is tracked in the drift volume since the ionization of the gas in the volume leaves a trace in the form of electron-ion pairs, along the path of particles as shown in figure 2.2. The parameters of its trajectory in a known magnetic field are utilized to measure the momentum of the particle. The electrons and ions produced in the ionization process are accelerated by the applied external electric field and get drifted towards the Read Out Chambers (ROCs) and the Central Electrode (CE) respectively. An electron avalanche is produced in the high electric fields in the vicinity of the anode wires in the ROCs, the primary ionization signal is amplified by this avalanche process. The electrical signal induced by the positive space charge onto a segmented pad plane is shaped, amplified, digitized, processed and readout from each readout pad by the data acquisition system.

2.3 Field cage and the drift volume

The field cage establishes a uniform electrostatic field in the drift region, to transport the primary ionization electrons towards the readout end plates. The mechanical structure for the foundation of other elements of the detector and for the precise positioning of the ROCs is provided by the field cage. The field cage mainly consists of four cylindrical vessels (field cage and containment vessels) and central electrode. The containment vessels are separated from the field cage vessels by an insulation gap. Key functions of these vessels are [2]:

- Mechanical stability and high assembly precision
- Leak tightness of the gas volume
- Heat shield against variations in temperature
- Containment of the drift high voltage network

2.3.1 Inner and outer containment vessels

The outer containment vessel is the outermost and largest in diameter, cylindrical component of the TPC whereas the inner containment vessel is the inner most and smallest cylinder of the TPC. The containment vessels surround the field cage vessels and provide a gas tight and grounded enclosure at the inner and outer diameter of the TPC. They are separated from the field cage vessels by a insulation gap filled with CO_2 . Even though these vessels are different in their size and position in the assembly of the TPC, yet both of these vessels are essentially responsible for the similar functions [6].

2.3.2 Inner and outer field cage vessels

These cylindrical vessels essentially function to define the gas volume of the TPC. They are equipped with coarsely segmented guard rings to avoid the accumulation of charge on the vessel surface.

2.3.3 The high voltage central electrode

As shown in the figure 2.1, the gas volume is divided in two back to back drift regions by the central electrode. The central electrode and the end plates of the readout chambers define the drift field of the TPC. The central electrode consists of stretched mylar foil which is flattened over circular shaped inner and outer Aluminium rims.

2.4 Readout chambers

The ALICE TPC readout chambers are based on conventional Multi Wire Proportional Chambers (MWPC) with cathode pad readout. Readout chambers of two different sizes are arranged in 18 trapezoidal sectors. This radial segmentation of a single TPC readout sector is identified as Inner ReadOut Chamber (IROC) and Outer ReadOut Chamber (OROC). The MWPC is composed of three different wire grids and the cathode pad plane.

- **Gating wire grid:** It functions as a control element of the entrance between the amplification and the drift regions. It admits or prevents electrons from the drift region to drift into the amplification region and at the same time it prohibits the ions created in previous events to drift back into drift region [8]. This gating functionality is accomplished by applying opposite levels of biasing voltages to the gating grid wires by a dedicated gating grid system discussed in section 2.5.
- **Cathode wire grid:** The cathode wire grid shapes the electric field in the amplification region. Many of the positive ions produced during the amplification through avalanche process are collected at the cathode wires.
- **Anode wire grid:** The electrons from the drift region cross the cathode wire grid and drift towards the anode wire grid, which is on positive high voltage. The strong electrical field gradient in the vicinity of the anode wires creates electron avalanches for each primary electron, amplifying them and producing new electron-ion pairs [28].
- **Cathode pad plane:** The electrons are collected on the anode grid wires and slowly drifting ions towards the cathode plane induce a mirror charge on the pad plane which

is collected as the measured signal [8], where each pad corresponds to a single readout channel of the TPC. The charge collected on each pad is further processed by the signal conditioning and processing chain as discussed in section 2.7.

2.5 Gating grid system

The ALICE TPC is equipped with one gate pulser system for each of the two TPC sides. The gate pulser system is responsible to bring the gating grids of the proportional wire chambers from an opaque (closed) state to a transparent (open) state and back to opaque state. In the transparent state the electrons from the drift region can enter the amplification region. In the opaque state electrons are prohibited to reach into the amplification region. One of the most important function of the gating grid system is to prevent the positive ions drifting back from the amplification to enter the drift volume of the TPC [2]. Key components in the gating grid system are shown in figure 2.3(a). The gating pulser firmware is implemented using standard readout control unit (RCU) motherboard as described in section 2.7.2.

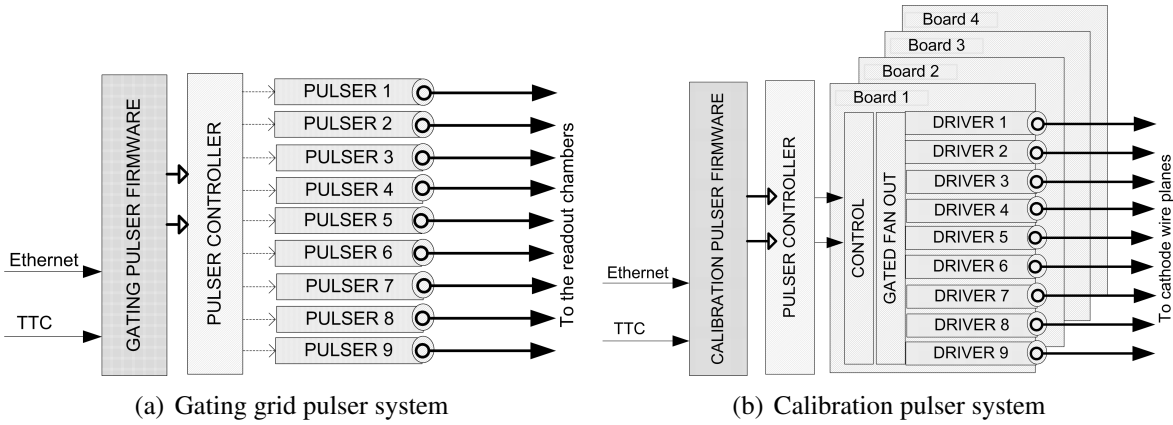


Figure 2.3: Pulser systems of the TPC

2.6 Calibration systems

The ALICE TPC contains a large drift volume, where the drifting electrons and ions are sensitive to temperature, pressure and gas mixture variations. Minor variations in the drift field parameters and even a slight imperfection of the electric field inside the field cage may affect the trajectories of the particles and the drift time. The calibration of the drift time can compensate these effects by including the variations in final reconstruction of the particle trajectories. The readout electronics for each channel can be subjected to different manufacturing variations affecting the shape of the analog signal. It is required to calibrate the readout electronics to estimate noise, pedestal, gain and other parameters of each readout pad. These calibrations are performed by means of the laser calibrations system and the calibration pulser system which are discussed here.

2.6.1 Calibration pulser

The calibration pulser system is used to inject charge into the pads by pulsing the cathode wire planes of the MWPC readout chambers of the TPC. A voltage step is applied to the

cathode wire plane, due to the capacitive coupling between cathode plane and the pads a signal proportional to the amplitude of the voltage step is generated to the readout electronics [2]. The shape of the applied voltage pulse and the position of the signal relative to the trigger of the readout is configurable. In figure 2.3(b) a block diagram of the setup of a calibration pulser system is shown. Each side of the TPC is equipped with such a setup. The system is controlled by a circuit implemented in a FPGA inside the controller module. It is connected to a DAC which generates the output signal to control the voltage drivers. The calibration pulser system is controlled by a standard RCU motherboard (section 2.7.2) with modified firmware. The pulser system is accessed via Ethernet link through the DCS board (section 2.7.2.1) [25].

2.6.2 Laser calibration

The ALICE TPC is equipped with a laser system to calibrate the imperfections in its drift volume [2]. The laser calibration system is used to pulse UV laser beams in the drift volume. Six laser rods carry four fixed bundles, each bundle consists of seven micro mirrors. These mirrors are utilized to extract the laser beams in order to shoot them on known angular distances into the drift volume [30]. In total 168 such laser beams are reflected into each side of the TPC drift volume. The laser calibration system and its procedure is described in detail in the reference listed here [30].

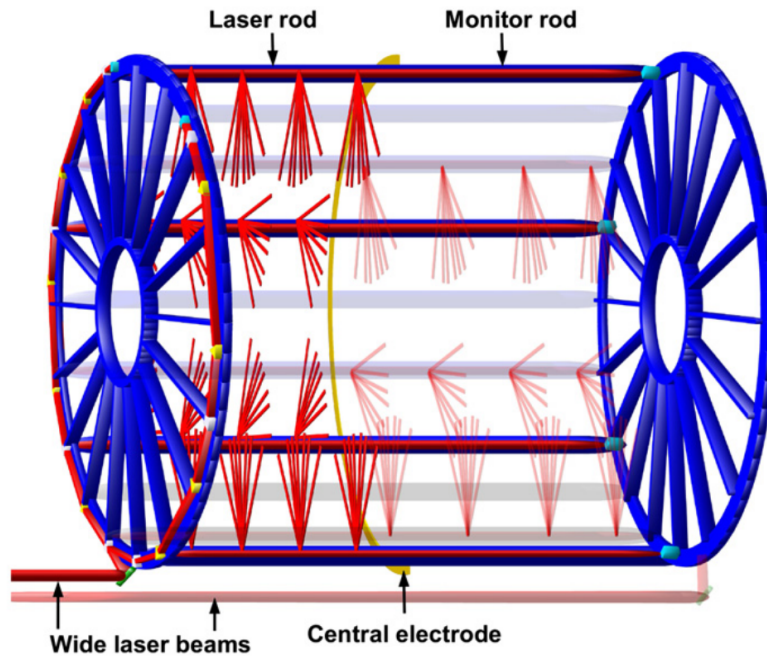


Figure 2.4: 3D view of the laser tracks in the drift volume [30]

2.7 Front end electronics of the TPC

The front end electronics for the readout of the signal is installed on both sides of the TPC readout plane. The readout plane is divided in 18 trapezoidal sectors. The azimuthal segmentation (sector) of the TPC readout plane covers 20 degrees as shown in figure 2.5. A sector is further divided into six readout partitions, containing different number of the front end cards (FECs) as shown in the figure 2.5. The two inner most readout partitions covers the Inner

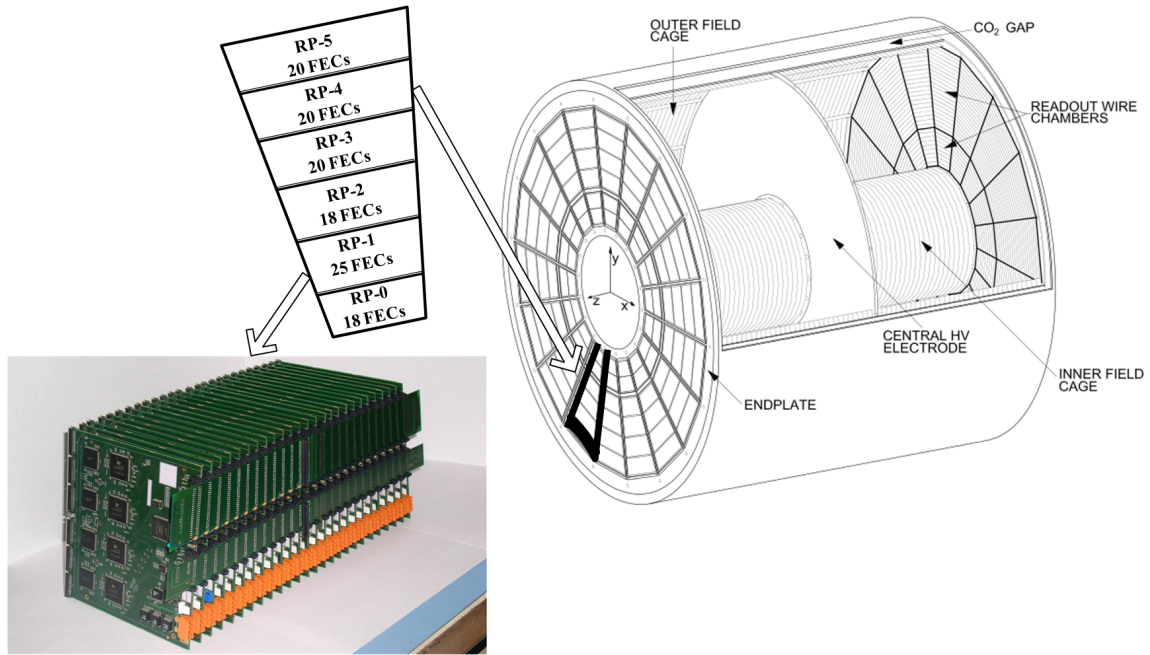


Figure 2.5: Top level architectural view of the TPC front end electronics

ReadOut Chamber (IROC) whereas the outer most four partitions covers the Outer ReadOut Chamber (OROC). Each readout partition is further divided into two branches; branch A and B. Both of these branches consist of equal number of FECs except in the readout partition 1 where branch A contain 13 FECs as compared to 12 FECs in the branch B.

2.7.1 Front end card

The TPC front end card is manufactured of an eight layer PCB which is equipped with eight ALice TPC ReadOut (ALTRO) chips and eight PreAmplifier ShAper (PASA) chips, encapsulating in total 128 readout channels. Each FEC is connected to the readout plane through 6 flexible kapton cables inserted into on-board connectors. Each FEC is equipped with a board controller to monitor and control the communication of the FEC to the RCU. There are two buses for the communication between the RCU and the FEC; the ALTRO bus and the Front-end Control (FC) bus. The ALTRO bus is a 40-bit parallel data bus with asynchronous control signals that is shared between the ALTRO chips and the board controller on the FEC, and is used for configuration and data readout. The FC bus is a modified I²C protocol that is used for monitoring.

2.7.1.1 Preamplifier shaper chip

The PASA [31] is a custom made analog integrated circuit, used to integrate, amplify and shape the charge collected on the TPC pad. The PASA is made up of low input impedance amplifier based on a CSA (Charge Sensitive Amplifier) followed by a fourth order semi Gaussian pulse shaper. The PASA is implemented in the AMS CMOS 0.35 μ m technology.

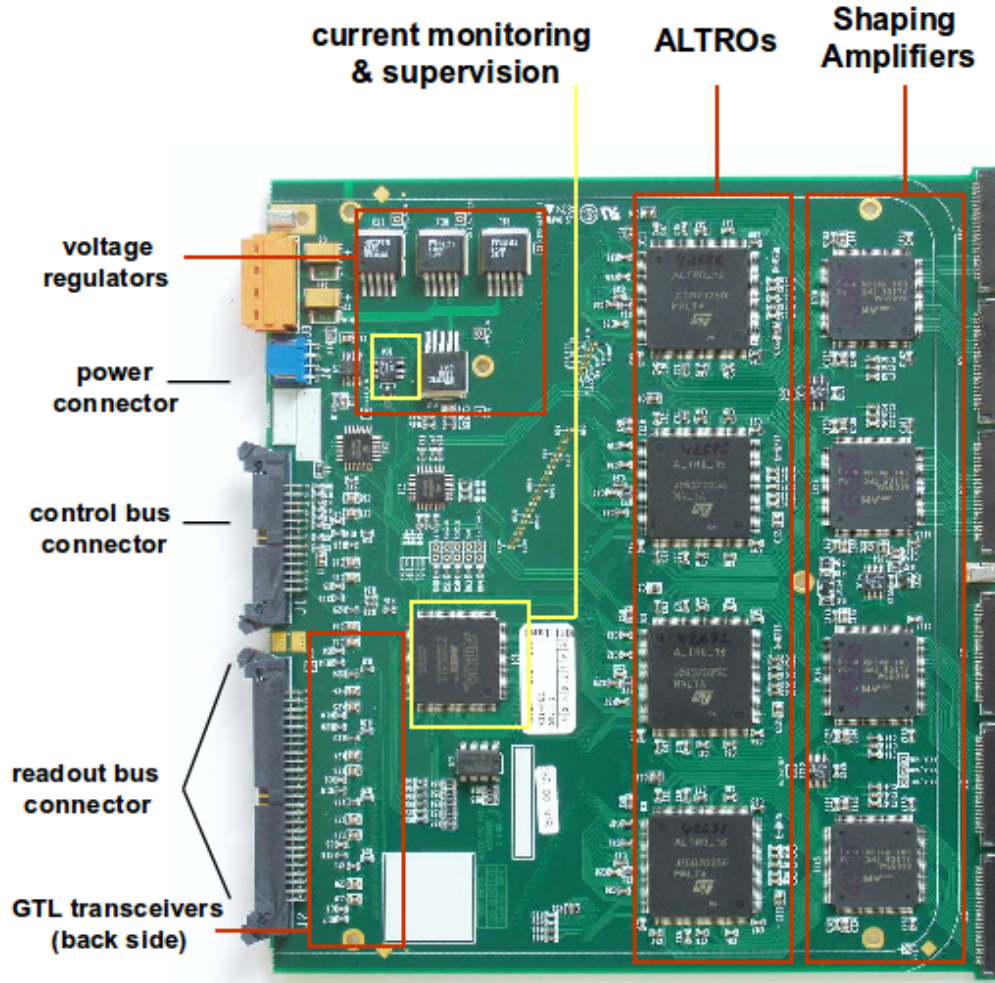


Figure 2.6: Single front end card (packed with 128 channels)

2.7.1.2 ALICE time projection chamber readout chip

The ALTRO is a mixed signal integrated circuit specially designed to meet the needs of the ALICE TPC. It consists of a 10-bit ADC (Analog Digital Converter) in addition to a digital circuitry for the signal processing. The output of the PASA is connected to the built-in ADC of the ALTRO. There are 16 independent channels integrated in one IC, implemented in 0.25 μm CMOS process operating concurrently on the analog signals coming from 16 independent inputs (PASA outputs). Each of these channels is composed of ADC, Baseline Correction and Subtraction 1 (BCS1), Tail Cancellation Filter (TCF), Baseline Correction and Subtraction 2 (BCS2), Zero Suppression Unit (ZSU), Data Formatting Unit (DFU) and Multi Event Buffer (MEB) [20].

There are two frequency domains in the ALTRO, first one is driven by the sampling clock for the ADC and common control logic of digital processing blocks, and the second one is driven by the readout clock which controls the readout of the event data from the MEB. The ADC runs on the sampling clock, the influence of the readout clock on the signal is minimized by avoiding all kind of switching activity during the signal acquisition and conversion time on readout clock. The ALTRO needs two levels of trigger, the first one named as level 1 (L1) is required to buffer the event data in the MEB whereas second trigger named as level 2 (L2) validates the event data by incrementing the write pointer to the MEB in the channel. The

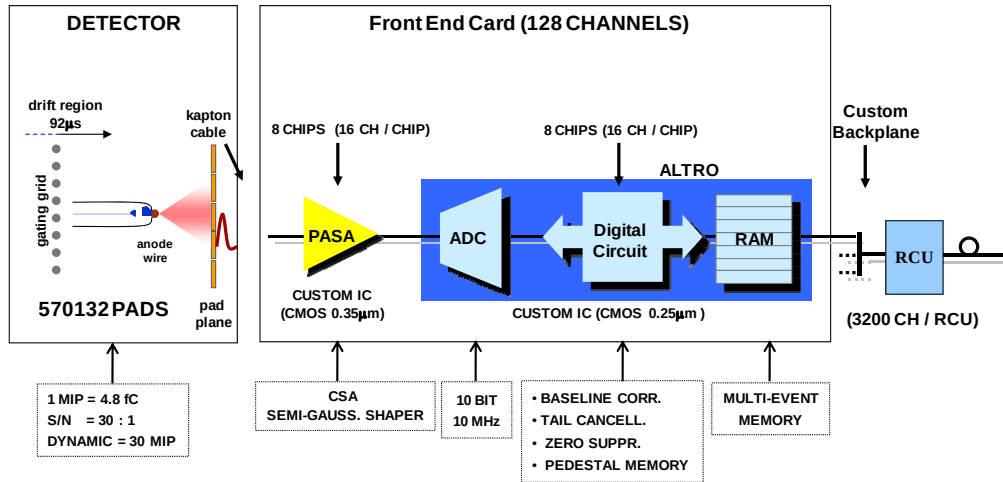


Figure 2.7: The ALTRO single channel data processing and transport chain. Each ALTRO integrates 16 channels and a single FEC encapsulates 128 channels

maximum length of an event to be stored in the MEB is 1008 samples. The latest event is kept in the MEB until the data is shipped out via the ALTRO bus. Each MEB of a channel has a capacity of buffering up to eight events. If two L1 trigger pulses are received by the ALTRO chip, skipping the corresponding L2 trigger pulse after the first L1 pulse then the first acquisition is discarded and overwritten by the acquisition initialized by the second L1 pulse [20].

2.7.1.3 Board controller

The Board Controller handles the low level monitoring and controlling tasks on the FEC. The board controller is responsible to monitor the health status of the FEC by measuring the current levels, voltage levels and temperatures on the FEC. These measured parameters are compared to configurable thresholds and an interrupt is issued to notify the RCU via a dedicated line if any monitored value violates a configurable threshold. It controls the ALTRO bus signals and polarity of the GTL drivers used in the communication network with the RCU.

The most important role of the board controller in the event readout is to enable the sparse readout functionality of the FEC, i.e. enable the RCU to read the channels where event data have been buffered and labeled as a 'hit' by skipping all empty channels. The sparse readout and the function of the board controller is described in section 3.2.6.2.

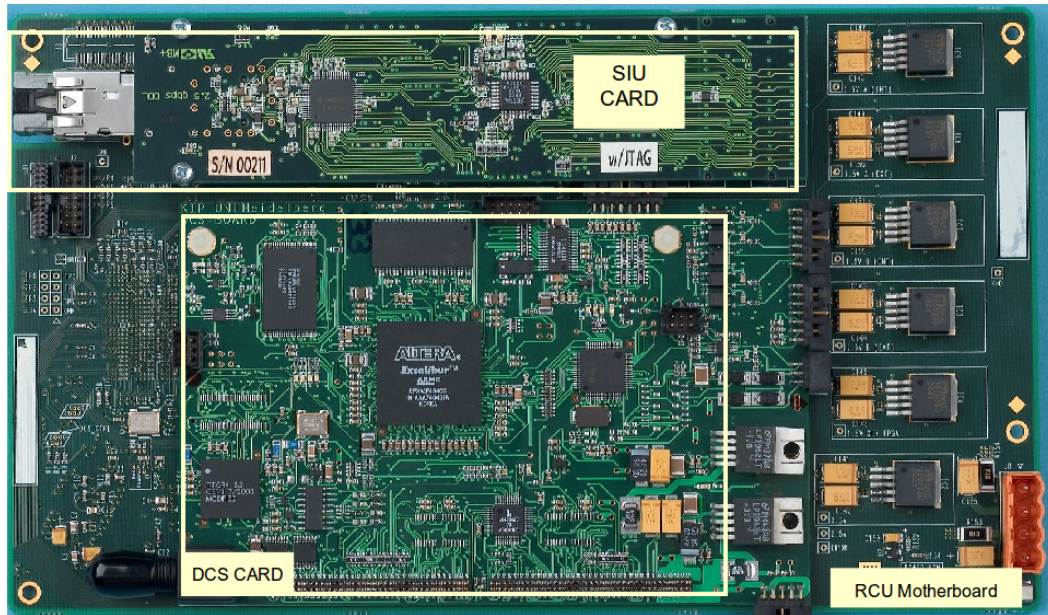
2.7.2 The RCU motherboard

Each readout partition contains a RCU motherboard. It provides a platform to address FECs for configuration and monitoring (DCS) and event data readout (DAQ). This interface is implemented on the RCU motherboard by hosting two daughter cards; the SIU card and the DCS card to interface with the DAQ and the DCS respectively as shown in the picture in figure 2.8(a). It directly connects to the FECs in the readout partition via the ALTRO interface bus and a FC serial link. The key component of the motherboard is the Xilinx Virtex II pro FPGA. This FPGA is programmed with the RCU firmware to establish the communication between all the agents of online sub systems of the ALICE experiment. Moreover, the RCU motherboard is equipped with an ACTEL flash based FPGA, which monitors the configuration of the

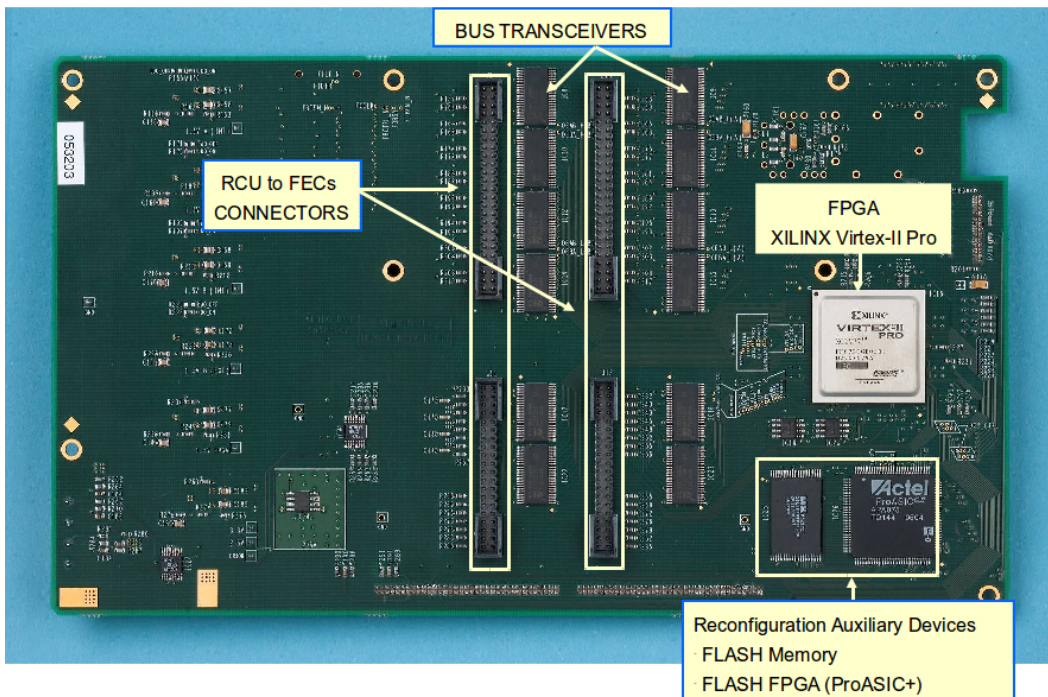
Xilinx device (RCU firmware) and corrects any single bit flip due to radiation effects [26].

2.7.2.1 DCS card

The DCS board functions as a monitoring and control node inside of the DCS network. It hosts a TTCRx [21] chip which receives the trigger sequences, messages and clock via optical link from the CTP. The DCS board is equipped with an on-board crystal oscillator to generate a local clock. It provides the system clock to the RCU board from either of the two sources. The



(a) An assembled RCU motherboard



(b) The RCU motherboard (FECs interface side)

Figure 2.8: The RCU motherboard; with its daughter cards and interface to FECs

firmware to implement the functionality of the DCS card is realized in an ALTERA *Excalibur* FPGA (EPXA1) device featuring a built-in ARM processor core [6]. The FPGA firmware and the Linux based customized embedded operating system of the DCS board is stored on a flash memory. This embedded minimalistic Linux system facilitates the access to the DCS board and Xilinx device via Ethernet. It may for example be used to program the Xilinx device (the RCU firmware). The DCS board is interfaced to the RCU firmware via dedicated 32 bit wide data and 16 bits wide address bus. This DCS bus interface is utilized to communicate to the FECs for the monitoring and safety of the FEC or the configuration of ALTRO parameters via the RCU.

2.7.2.2 SIU card

The SIU card is responsible for the communication between the FECs and the DAQ via the RCU firmware. It acts as a remote agent of the DAQ system to the front end electronics. It translates DAQ commands to the FECs through the RCU firmware to configure and perform the event readout. The SIU card is based on the ACTEL *ProASIC+* flash based FPGA, which implements the DDL communication protocol and additional logical interface between the RCU firmware and the DAQ. The SIU hardware utilizes a full duplex optical transceiver which makes the conversion between the electrical data and the optical serial data to and from the DDL.

2.7.3 Busy box

The busy box is a FPGA based component of the readout system which verifies that the event data corresponding to the received trigger from the CTP has been acquired from the FEE to the DAQ system. The busy box receives similar trigger input from the CTP as the RCU firmware. It asserts a busy signal to the trigger system indicating that the readout electronics is busy performing readout of the received trigger and not able to receive the new trigger. Each trigger sent from the CTP is marked with a special event identification number (event ID). This event ID is sent to the DAQ as a part of the event data header by the RCU firmware. The busy box communicates to the DAQ to receive the event ID, it compares and verifies that the event ID received from the CTP and the DAQ are similar to release the busy signal to trigger system. As discussed earlier each ALTRO channel is able to store four consecutive events in the MEB. In order to activate the mode of operation of the ALTRO chip where all four event buffers are used to store the event data ($MEB > 1$), the corresponding configuration has to be activated in the busy box. In that case the busy box asserts the busy signal to the CTP only where all four MEBs in the ALTRO channels are occupied with four events and the DAQ is waiting for the four events to be received from the RCU.

Chapter 3

RCU Firmware

The design, implementation and optimization of the RCU firmware are the primary objectives of the work carried out for this thesis. This chapter describes the details of the RCU firmware covering the key functional requirements, the hardware implementation, the functional verification and commissioning procedures.

3.1 Functional requirements

The RCU functions as a bridge to interface the FECs to the online systems (DDL, DCS and CTP) of the ALICE experiment. The RCU is required to be able to interact concurrently or individually with the online systems of ALICE to meet the objectives described as under.

- To initialize and reset all of the memory elements to their default values.
- To receive the system clock from the DCS card and to provide the readout and sampling clocks to the ALTRO chips and the system clock to the SIU daughter card.
- To accept, verify and execute a complete set of instructions on the front-end electronics from the DAQ and the DCS. These instructions may consist of read and write operations on the complete addressable range of the ALTRO chip and the board controller memories or special commands to control the status of FECs.
- To record the result of each read and write transaction performed on the FECs.
- To accept, verify and process the trigger sequences from the CTP and issue the L1 and the L2 triggers to the FECs.
- To format the event data received from the ALTRO channels according to the ALICE data format and transmit it to the DAQ.
- To detect erroneous situation from any front end card during an event readout and report it without interrupting the readout.
- To monitor all of the physical parameters of each FEC and read two temperature sensors on the RCU board.

3.2 Functional description

The functionality¹ of the RCU firmware is divided into five main tasks. Each task can be invoked individually to verify its compliance with the functional requirements set by the readout system. These tasks share the hardware resources and different design modules in the RCU firmware. Each task is described in detail in its respective section.

3.2.1 Clock management

The RCU receives the differential clock signal of 40 MHz frequency from the DCS board. This clock is used to generate the following five different clock signals.

- **RCU system clock**

The RCU clock is the system clock for the RCU firmware, all other clock signals are driven from this clock. The frequency of this clock is fixed to 40 MHz.

- **Readout clock**

This clock signal is delivered to the FECs on both branches in a readout partition. The readout clock for both branches A and B is similar and driven at 40 MHz frequency from the same source in the RCU firmware.

- **ADC sampling clock**

The sampling clock is configurable for four different frequencies. This clock is driven from the readout clock to keep the fixed phase relation between both clocks delivered to the ALTRO chips. The sampling clock is delivered to both branches as differential output from the RCU. The phase of the sampling clock is reset upon the Start Of Data (SoD) trigger (section 3.2.5.2) sequence received from the CTP to make sure that all of the ALTRO channels begin the data acquisition at the same time.

- **SIU clock**

The SIU clock with 40 MHz frequency is provided to the SIU daughter card on the RCU board. This clock is driven from the system clock of the RCU.

- **I²C clock**

The I²C clock is driven from RCU clock as well. This clock is provided to the FECs on both branches to communicate to the board controller on the FEC through the FC bus. It is configurable for four different frequencies.

3.2.2 Reset and initialization of the system

The mechanism to reset the system is handled with different types of commands to minimize the recovery time from a faulty or erroneous situation by localizing the effect of a corrective action. The RCU firmware is designed to provide special signals to initialize and reset the internal components of the RCU firmware and front end cards to their default operational state. The RCU firmware is implemented with the following methods to address the need to reset the RCU or the FECs. The methods are realized by special set of high level commands from the DCS or the DDL. There is no single command or a signal to reset the memories

¹Functional description is explained as it is perceived at design phase of the RCU firmware. Functional description of the RCU may have different detector specific descriptions due to its usage but the core functionality to define the objectives of the design remains the same.

implemented as block rams (BRAM), instead these are initialized with a fixed pattern of “0xf” upon boot up of the system.

Reset type	The DCS command code	The DDL command code
Global reset	0x5300	0x2
FEC reset	0x5301	0x8
RCU reset	0x5302	0x4

Table 3.1: Commands to issue resets of different hierarchy

- **Internal reset on power up**

This reset signal is generated internally in the RCU upon the programming of the RCU firmware (Xilinx device) with the configuration file. It clears all the memory elements including registers, counters and switches the states of finite state machines of the RCU firmware to their default states.

- **Global system reset**

The global reset can be issued by the DCS or the DDL to initialize the RCU and the FECs by executing the command listed in table 3.2.2. It clears all of the status and error registers of the RCU firmware and the ALTRO. The configuration registers of the RCU firmware as discussed in section 3.2.3 are not reset with this command. However the ALTRO configuration registers are cleared as a consequence. it is required to reconfigure the ALTRO chips after a global reset.

- **RCU reset**

The RCU reset is capable of clearing all the error and status registers of the RCU firmware and reset the finite state machines to their default states. This reset signal can be asserted by issuing the commands as listed in table 3.2.2. This reset is useful in such cases where it is required to bring the RCU in default and error free state without affecting the FECs.

- **FEC reset**

The FEC reset is capable of clearing all the error and status registers of the ALTRO chips and the board controller firmware and resetting the finite state machines to their default states. This reset signal can be asserted by issuing the commands as listed in table 3.2.2. The FEC reset clears the configuration registers of the ALTRO chips and it is required to reconfigure the ALTRO chips after a FEC reset.

3.2.3 Configuration and verification

The configuration of the complete readout system is the most basic and initial task. It can be performed by the DCS or the DDL in order to prepare the detector for event readout. It is carried out after resetting and initialization of the complete readout system including all of the RCU boards and the FECs. The main tasks performed to accomplish the configuration are listed here.

- All of the RCU configuration registers as listed in table 3.14 are written to configure the RCU. The switch lines to control the power to the FECs are controlled by a configuration register in the RCU firmware. The parameters through the registers in table 3.14 define

various timing thresholds related to the ALTRO bus interface. Acceptance windows for the trigger sequences as described in [5] are also set by writing into the respective configuration registers of the trigger receiver module [5]

- The readout list memories in the RCU are configured with the addresses of channels to be included in the event readout. The ALTRO channels with the addresses listed in readout list memories are common for both of the event readout modes described in section 3.2.6
- All of the FECs in a partition are configured by writing the configuration registers of the ALTRO chips [20] and the board controllers. The configuration of the FECs is carried out by executing multiple write instructions packed in a sequence as discussed in section 3.3.3.

After configuration of the front end electronics, verification of the written data is essential to minimize the chances of malfunctioning or failure of the readout system. The RCU configuration registers and memories are verified by reading and comparing with the written value. The result of each command executed on ALTRO bus is recorded in the result memory in the RCU firmware as described in section 3.3.4. Contents of the result memory are available to the DCS or DDL to verify the success of the configuration executed through RCU.

3.2.4 Monitoring and safety

The FECs in a readout partition on both branches are monitored for their physical parameters including analogue current & voltage, digital current & voltage and the temperature of each FEC. The thresholds (upper limits) of these parameters are configurable in the board controller. The board controllers on all FECs can assert an interrupt to the RCU indicating that the measured value of any critical parameter has reached its maximum threshold. The monitoring and safety procedures in the RCU are accessible only from the DCS. The RCU updates the temperature values from both sensors on the RCU board and makes these measurements available to be readout by the DCS.

3.2.5 Trigger handling

The ALTRO chip requires two levels of triggers to acquire and store the event data as described in section 2.7.1.2. The RCU issues these trigger pulses to the ALTRO chips. The RCU is configurable to accept triggers from three different sources discussed here.

3.2.5.1 Test trigger

- **Software trigger**

The RCU accepts a software command from the DCS to trigger the readout electronics. This command can be issued only from the DCS side because before issuing a trigger the direction of the DDL is set to receive the data from the RCU. This software command is processed to issue a L1 pulse to the FECs followed by a L2 pulse after a programmable latency. When the readout electronics is triggered by the software trigger the communication between the busy box and the RORC is not part of the complete system. The software trigger is primarily used to verify the integrity of the event data formatted and shipped by the RCU as discussed in section 3.5.2.3.

- **Hardware trigger**

The RCU can be configured to accept an input trigger from an external source or a pulse generator on a dedicated input line to the RCU. A single pulse is issued from the external source and delivered as the L1 trigger to the FECs, the L2 pulse is generated internally in the RCU firmware after a programmable latency.

3.2.5.2 CTP trigger

Under normal operating conditions the ALICE TPC is triggered by the CTP. The RCU receives the trigger sequences through a dedicated optical fiber. These trigger sequences are decoded and verified for their validity related to timing parameters and the messages [5]. A trigger sequence is declared as “valid sequence” to acquire the event data from the FECs if it consists of L0, L1 accept (L1a) and L2 accept (L2a) triggers. The trigger sequences are declared as erroneous trigger sequences if the timing requirements or the sequence of its contents are violated as discussed in reference [5]. There are three special trigger sequences issued from the CTP to the detectors.

- **Start of data trigger**

The SoD trigger is the first trigger sequence sent from the CTP to start the readout of the ALICE experiment. This trigger sequence invokes all the detectors to respond sending a SoD event to the DAQ [9]. The ECS waits for the DAQ to report that all of the detectors in the respective configuration have responded correctly. Handling and processing of the SoD event after reception confirms that all the online systems and detectors in the data stream of the ALICE experiment [9] are ready.

- **Synchronization trigger**

The SYNCHronization (SYNCH) trigger is a special trigger sequence which can be used to check and validate the status of the ALICE detector during data taking. The detectors respond to the reception of a SYNCH trigger by sending a SYNCH event to the DAQ indicating their coherence in the data taking. This trigger sequence can be requested multiple times during a data taking period.

- **End of data trigger**

The End Of Data (EoD) trigger is the last trigger sequence sent from the CTP to mark the end of data taking session of the ALICE experiment. The detectors respond to the reception of the EoD trigger by preparing and sending an End Of Data (EoD) event to DAQ [9]. The ECS waits for the DAQ to report that all of the detectors in the respective configuration have responded correctly.

3.2.6 Event readout

During event readout the RCU interacts with the FECs through the ALTRO interface to acquire the event data and transmits this data on the DDL in parallel to receiving the trigger sequences from the CTP. The RCU begins to read out event data from all ALTRO channels after the L1 pulse followed by a L2 pulse are issued to the FECs (in all readout partitions). The readout list memories in the RCU firmware are written with the addresses of the ALTRO channels at configuration time (section 3.2.3). The RCU is required to read out and transmit the event data from all channels listed in the readout list memory following the order in which channels are listed. The RCU scans and processes every single channel address listed in readout list

memory of each branch to accomplish the event readout. The RCU can perform the event readout in two different read out modes as described in the following.

3.2.6.1 Full readout

The RCU performs full readout by addressing all channels listed in the readout list memory of each branch. The RCU starts the readout process concurrently on both branches, by reading the respective channel addresses from the readout list memories. The RCU firmware is equipped with two independent data memories for the two branches. A flow chart of the full readout is shown in figure 3.1(b). Three logical processes operate concurrently in the RCU during full readout, as shown in figure 3.1(a). The channel readout process begins to scan the readout list memory to fetch the channel address for read out on both branches. The channel

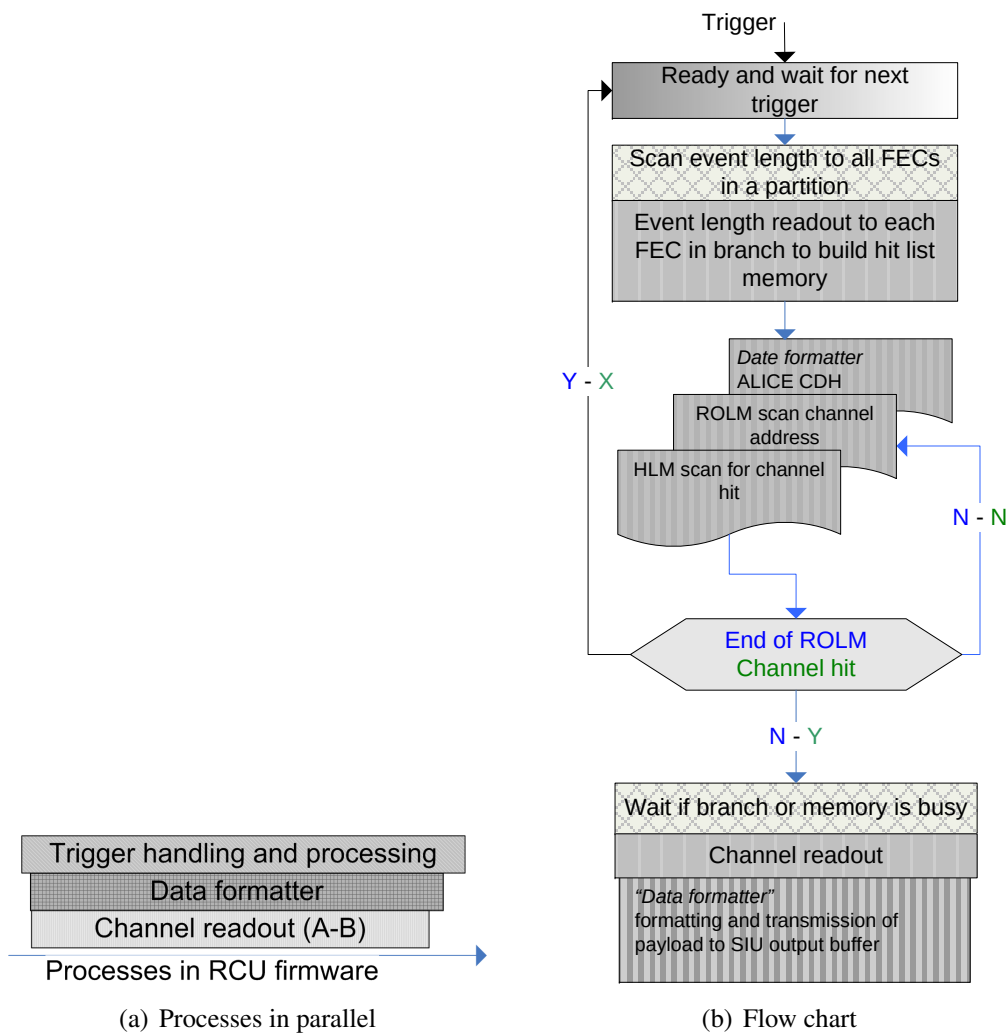


Figure 3.1: Flow chart of the full readout

address from the readout list memory is checked to verify that the channel address belongs to an active front end card. If the corresponding FEC is not powered on then the channel is skipped and the next location in memory is accessed to get the next channel to be read out. The CHannel ReaDOut (CHRDO) [19] command is issued to the channel and the RCU waits for the channel to transfer the event data to one of the available data memories of the corresponding branch in the RCU. The complete readout process is monitored through watchdog

timers to detect any kind of timing violation during the data transfer to RCU. In case of any ALTRO protocol violation or erroneous situation the RCU skips the channel and continues to the next channel for readout after reporting the error encountered in respective error registers [23]. In parallel to accessing and processing the channel addresses from the readout list memories on both branches, the process of data formatting and packing reads the event header from the trigger receiver to construct the common data header (CDH) according to the ALICE data format. The CDH is written to the output data buffer in the SIU interface unit (SoB) described in section 3.3.9 and this process waits till the first channel is completely readout. These two processes interact with each other through the status of four event data memories. The status of each data memory may have three possible states; 1) free: when a data memory does not contain event data and is available, 2) engage: The memory is locked by the channel readout process to write the event data from a channel, 3) ready: The readout process sets the status as ready for the data formatting process to read and write the formatted data to the SoB. The data formatting process polls on the status of data memories, waiting for its status to be marked as “ready”. This process resets the status of the memory to “free” after the channel data is formatted and written to the SoB. The event readout is completed after the data from all of the channels listed in the readout list memories has been read out into the data memories and from there written to the SoB, terminated by the RCU trailer words. The processes described above then stay in idle state waiting for the next trigger sequence from CTP. An independent process runs in the RCU to receive and process the trigger sequences from the CTP. This process gets activated upon SoD trigger from the CTP till the EoD trigger is received. This process is responsible of selecting the right parameters to build the event data packet as explained in the data format (section 3.2.7)

3.2.6.2 Sparse readout

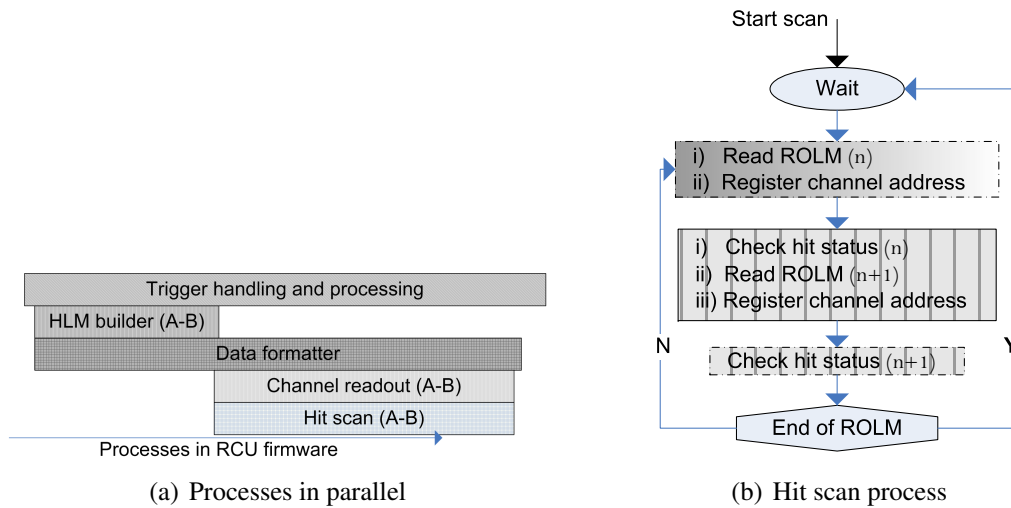


Figure 3.2: Parallel processes during the sparse readout

The RCU is able to perform sparsified readout, where the empty ALTRO channels are not addressed by the RCU to acquire the event data. This mode involves one additional phase as compared to full readout, where the status of each channel in terms of channel hit or empty is transferred to the RCU before starting the event readout. Two processes related to sparse readout get activated upon a L2a pulse of a valid trigger sequence from the CTP. The HLM

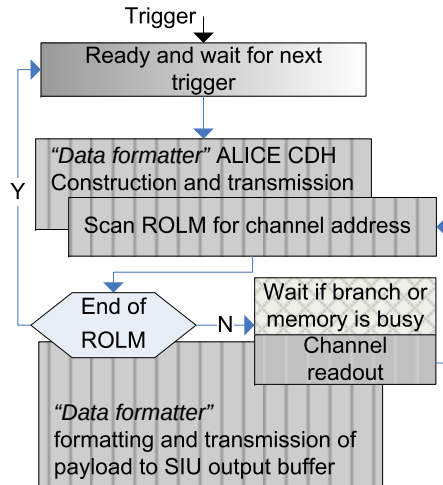


Figure 3.3: Flow chart of the sparse readout

builder process shown in figure 3.2(a) issues the SCEVL command [14] in broadcast mode to all FECs in a readout partition and waits for 90 μ s. This time is used for transferring the status of each channel from all ALTRO chips to the board controller [14]. As a next phase of this process each FEC is addressed to update the hit list memory in the RCU. The board controller from every FEC writes four words of 40 bit width updating the status of 128 channels. A data formatter process begins to construct and transfer the CDH to the SoB. Three processes (channel readout, hit scan and data formatter) run in parallel to complete an event readout. The hit scan process is of key importance in sparse readout mode. It verifies the hit status of each channel from the hit list memory. The readout process starts to address the channel (marked with hit status=1) to acquire the event data. While the readout process acquires the data from any channel, the hit scan process continues to scan for the consecutive channel addresses from the readout list memory. The hit scan process may pause the scanning if two consecutive channels are found with hit status. The data formatter waits for the first data memory to get ready before it starts to format and transmit the event data to the SoB. These processes continue for both branches in parallel. Consequently, in this readout mode event data is transmitted to the DDL only from non empty channels.

3.2.7 Data format

The format of the event data transferred from the FEC to the RCU is based on the ALTRO data format as shown in figure 3.4 from the reference [19]. It consists of 10 bit data word sampled by ADC and the 10 bit word indicating the time stamp of the data packets. The data word and time stamp are shown as S_{xy} , T_{xy} respectively in figure 3.4. These 10 bit words are packed into 40 bit words of the ALTRO format, if the number of data words is not multiple of 4 a padding of “0x2aa” is used to fill the remaining space in the ALTRO data word. At the end of an event packet each channel transmits the 40 bit ALTRO channel trailer [19]. The trailer is composed of the size of the payload and of the channel address. The channel address represents the geographical address of the channel in the TPC and identifies to which readout pad the data packet is associated [19].

The structure of the RCU data format is shown in table 3.2. The CDH is constructed according to the ALICE specifications. It is followed by the event data and the RCU trailer. The CDH contains the information related to the received trigger from the CTP. The basic struc-

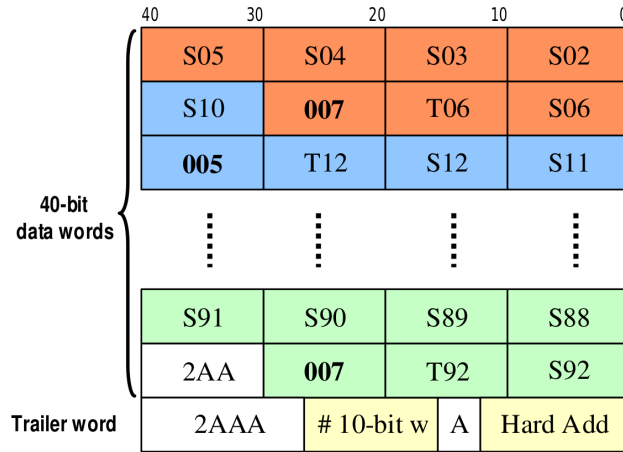


Figure 3.4: The ALTRO data format [19]

Common Data Header (8 words)
Event payload (Variable number of words, minimum 0)
RCU Trailer (9 words)

Table 3.2: Structure of the event data block

ture of the CDH is shown in table 3.3. All information in the CDH except the error and status bits are copied from the event info received in trigger sequences. These fields words 4 [27:12] and word 3 [31:24] are set by the RCU.

The RCU transforms the 40 bit event data from the data memories into 32 bit data. The two

No.	Word description [31:0]				
0	Block length [31:0]				
1	Version [31:24]	MBZ [23:22]	L1 trigger message [21:14]	0	Bunch crossing [11:0]
2	MBZ [31:24]		Event ID (Orbit number) [23:0]		
3	PAC request [31:28]		RCU version [27:24]	Participating sub-detectors[23:0]	
4	MBZ [31:28]	Status and error bits [27:12]		Mini event ID [11:0]	
5	Trigger Classes Low [31:0]				
6	ROI low [31:28]		MBZ [27:18]	Trigger classes high [17:0]	
7	ROI high[31:0]				

Table 3.3: ALICE common data header

most significant bits of the 32 bit word in the RCU data format are used as an identifier of the word as shown in tables 3.4 and 3.5. The first three 10 bit words of the first 40 bit word are packed into the first single 32 bit word written to the SoB. It concatenates the last 10 bit word to the next two 10 bit words from the next location of data memory. The extra padding added by the ALTRO chips to complete a 40 bit word is removed by the RCU. The RCU does not differentiate between time stamp and data words. The ALTRO Channel trailer constitutes of the channel address and the block length of the event data sent by the channel. The ALTRO channel trailer is the last word sent from an ALTRO channel in the event data, but the RCU sends the trailer word as the first word of a channel data packet. The RCU verifies these

two fields of information before sending the channel trailer to the DDL. If any of the channel trailer is found corrupted for channel address or block length then the corrupted field is corrected and marked in the trailer as erroneous by setting the “E” error bit in the ALTRO channel header as shown in table 3.4. At the end of the payload, if the number of data words is not exactly a multiple of 3, then the empty space is filled with “0x0” padding as shown in table 3.4.

ID [31:30]	Word description [29:0]		
01	Error [29]	ALTRO Channel header[28:0]	
00	DW_{n-6}	DW_{n-5}	DW_{n-4}
00	DW_{n-3}	DW_{n-2}	DW_{n-1}
00	DW_{n-0}	0x0 padding ₁	0x0 padding ₂
01	Error [29]	ALTRO Channel header[28:0]	
00	DW_{n-4}	DW_{n-3}	DW_{n-2}
00	DW_{n-1}	DW_{n-0}	0x0 padding ₁
01	Error [29]	ALTRO Channel header[28:0]	
00	DW_{n-2}	DW_{n-1}	DW_{n-0}
01	Error [29]	ALTRO Channel header[28:0]	

Table 3.4: Structure of the event payload

The RCU trailer is the final construct of the event data packet. The RCU trailer words are marked in the ID fields of the 32 bit words sent to the DDL [23]. Details of the RCU trailer words are described in table 3.5.

ID[31:30]	Parameter	Code[29:26]	Description[25:0]
10	Pay Load Length	0000	Number of 32 bit words
10	Error Register 1	0001	Error Registers for Branch A and B
10	Error Register 2	0010	Read out Errors
10	Error Register 3	0011	Number of Altro Trailer Errors
10	Act FEC A	0100	Active Front End Cards for Branch A
10	Act FEC B	0101	Active Front End Cards for Branch B
10	RDO CONFIG 1	0110	Readout configuration Register 1 of ALTRO
10	RDO CONFIG 2	0111	Readout configuration Register 2 of ALTRO
11	RCU ID	1000	Current RCU parameters

Table 3.5: Structure of RCU trailer

3.3 Architecture of the firmware

The architecture of the RCU firmware is partitioned based on its function as an interface between the three online systems of ALICE (DCS, DDL, TTC) and the FECs. The top level architecture and the implemented modules are shown in figure 3.5. Details of each module are given in the following sections.

3.3.1 RCU clock manager

The RCU Clock manager is equipped with three Digital Clock Manager (DCM) modules [36]. The DCM primitive in Xilinx FPGA (Virtex II - pro) is used to implement a digital frequency synthesizer of 40MHz frequency. The RCU clock manager accepts a differential clock signal from the DCS board. The DCM Main as shown in figure 3.6 is used to generate the system clock of the RCU. The SIU daughter card gets its system clock from the RCU, an independent DCM is utilized to provide the clock of 40MHz frequency. The DCM FEC in figure 3.6 is dedicated to generate the readout clock for the ALTRO chips, this clock is also used to generate the ADC sampling clock. The sampling clock can be configured for three different frequencies as mentioned in table 3.14.

3.3.2 ALTRO interface module

The ALTRO Interface Module (AIM) communicates on the ALTRO bus for the transactions to the FECs in both branches of a readout partition. The AIM is responsible for read and write transactions to all FECs in a readout partition. The ALTRO chip has a set of commands to perform specific tasks [19] and the AIM handles these command transactions as a special case of write operations from the RCU to the FEC. The AIM performs the readout of the event data from the channels to accomplish the readout of a complete event. The AIM is partitioned based on its three above mentioned functional objectives. Three sub-modules are instantiated

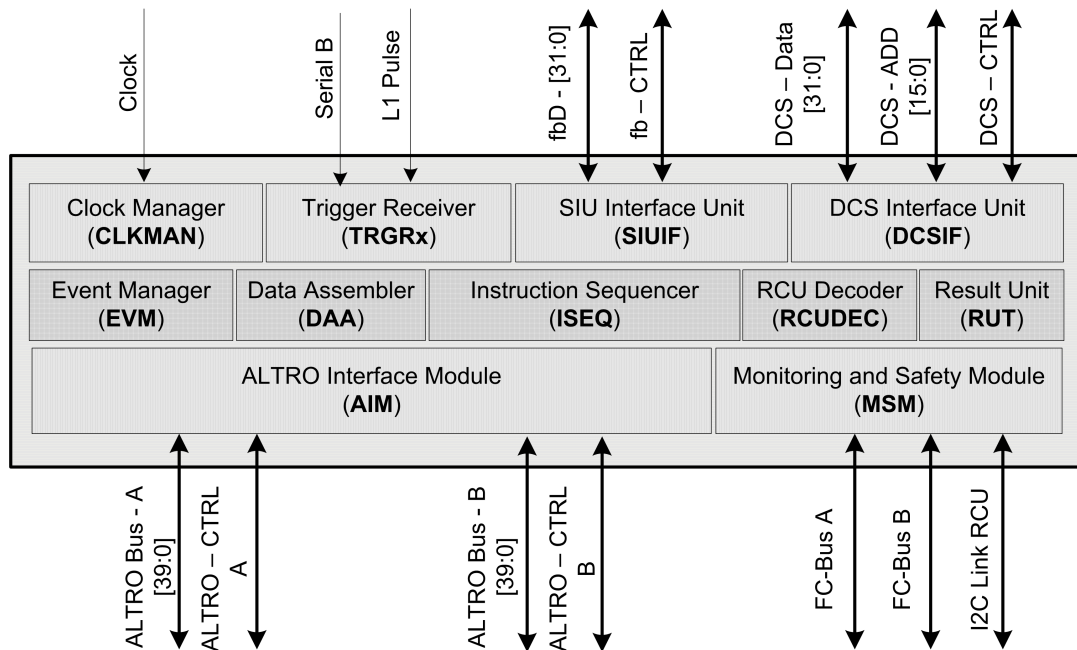


Figure 3.5: Top level architecture of the RCU firmware

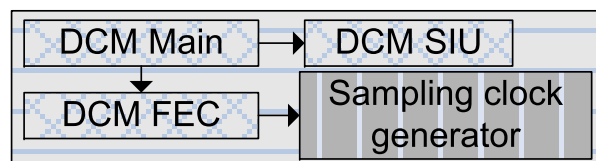


Figure 3.6: Components of clock manager

for each branch and work so that both branches are able to operate concurrently. These three sub-modules are discussed in the following.

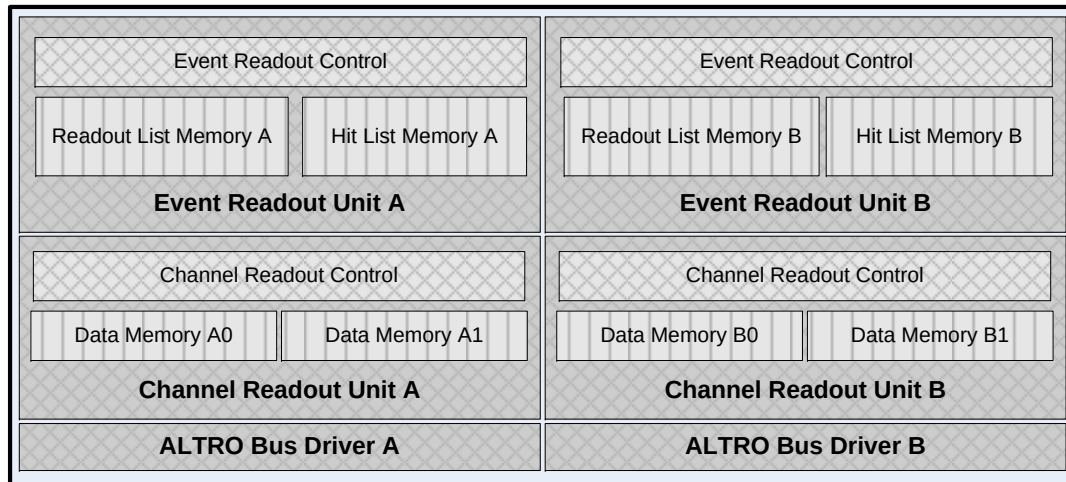


Figure 3.7: Sub-modules of the ALTRO interface module

3.3.2.1 ALTRO bus driver

The ALTRO Bus Driver (ABD) controls the transactions on the ALTRO bus. It accepts the orders from one of these three modules; the instruction sequencer (section 3.3.3), the CHannel ReaDOut Unit (CHRDO) or the EVent ReaDOut Unit (EVRDO). It delivers the result of each transaction to the result unit (section 3.3.4) to store in the result memory. It controls the direction of the GTL bus drivers on the ALTRO bus and sets the right polarity of the GTL drivers to complete the data transfer to or from the RCU. It monitors the timing and sequence of the ALTRO handshake protocol for all transactions and in case of any violation the relevant bit in error and status register [23] is updated. The ALTRO bus protocol for three transactions based on the reference [19] along with its signals is summarized here. Each transaction on the ALTRO bus is initialized by the RCU by asserting the control line named as command strobe (CSTB) and driving the data lines of the ALTRO bus following the ALTRO instruction format. A special bit in the ALTRO address field is set to decide if the transaction is targeted to an ALTRO chip or a board controller. If the transaction is of write or command type then the control line named as write enable (WEN) is asserted to indicate the operation (read or write) on the bus. The CSTB line is released after the acknowledge (ACKN) line is asserted by the addressed unit, indicating the completion of the transaction. A read operation on the ALTRO bus is completed with the assertion of ACKN indicating that the requested data from the addressed location has been placed on the bus.

The ABD consists of two finite state machines (FSM) for read and write transactions. Both of these state machines share the watchdog timers and the data registers.

3.3.2.2 Channel readout unit

The CHannel ReaDOut (CHRDO) unit contains two data memories as shown in figure 3.7 and a finite state machine to control the channel readout operation along with all required watchdog counters and data registers. Each one of these two data memories is used alternatively to avoid dead-time introduced by the data assembler (section 3.3.7) in reading the data memories and

transmitting the data to the DDL. The CHRDO unit receives a valid channel address from top level (event readout unit) to read out the event data from the addressed channel. The CHRDO unit on each branch is responsible to accomplish the channel readout for all the channels evaluated as valid by the event readout unit. It continues to run till end readout signal is asserted by the event readout unit.

3.3.2.3 Event readout unit

The EVeNT RaDOuT (EVRDO) unit consists of the event readout control unit, readout list and hit list memories as shown in figure 3.7. The event readout control unit reads the readout list memory and provides a valid address to the CHRDO unit after checking it against the list of active FECs. A channel is skipped if its address does not belong to a currently powered FEC (active FEC). This unit decides to verify the hit status from the hit list memory if the readout mode is configured as sparse mode. The EVRDO control unit continues to scan the readout list memory till end-of-readout-list-memory marker (0xffff) is found in the memory. The RPINC command [19] to all ALTRO chips is issued on both branches through the ABD by this unit after the last channel has been read out.

3.3.3 Instruction sequencer

As shown in figure 3.8(a) the instruction sequencer is equipped with an instruction memory which is configured to communicate to the FECs according to the instruction format in table 3.6. The instruction sequencer control unit is capable to execute a specific number of instructions named a sequence. This unit is capable of performing specific functions internally as listed in table 3.7. The status of the instruction sequencer is visible to all other modules in the RCU firmware to avoid any conflict on internal data handling or communication to FECs. When the instruction sequencer status is shown as busy no other module is allowed to access the ABD or the instruction memory.

Instruction ID [21:20]	Instruction Parameters [19:0]	
00 FEC-RD	ALTRO Instruction format [19]	
01 FEC-CMD	ALTRO Instruction format [19]	
10 FEC-WR	ALTRO Instruction (Address) [19] ALTRO Instruction (Data)	
11 ISEQ-CMD	ISEQ internal command code [19:16]	Parameters[15:0]

Table 3.6: Format of instructions loaded in instruction memory

3.3.4 Result unit

The result unit is responsible to record the result of each instruction executed by the instruction sequencer. It consists of the result memory and the result control unit module as shown in figure 3.8(b). The RCU can be configured to receive the contents of result memory as an event payload upon reception of an EoD trigger (section 3.2.5).

Command	Instruction code	Description
Loop	[11][0001]	Execute the instructions in loop for defined number of times
Wait	[11][0010]	Wait for defined number of clock cycles
ENDSEQ	[11][0100]	End of specific sequence (set of instructions)
ENDMEM	[11][1000]	End of Instruction Memory

Table 3.7: Internal instruction set of the sequencer

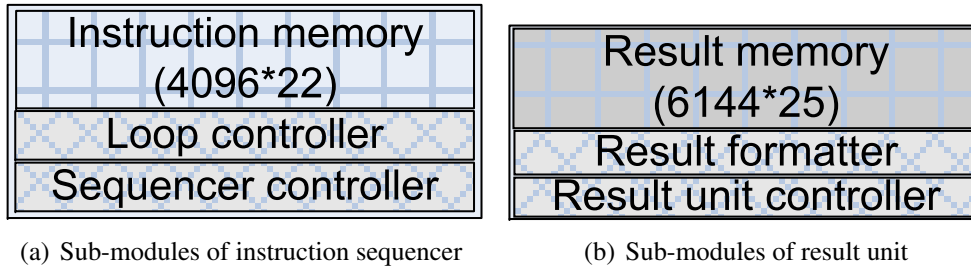


Figure 3.8: Instruction sequencer and result unit

• Reading FEC registers

All configuration registers, memories and status registers of the ALTRO chip and board controller can be read by the RCU by executing a read transaction on the ALTRO bus. The result of the read transaction including the content of the addressed location is written to the result memory. The result of each read instruction comprises two locations of the result memory [23]. The data received from the addressed location is written to the second location in the result memory for each respective read transaction. A status bit shows the success or failure of the execution of the read instruction. The content of the result memory for a single read transaction is shown in table 3.8. The RCU firmware verifies the content of the read command to the FEC before it is executed. If the instruction written in the memory does not comply with the ALTRO format then the ISEQ error code indicates the type of error in the instruction. If the read instruction fails during the execution the error code generated from the ALTRO interface module describes the nature of the error encountered during communication on the ALTRO bus [23].

(a) Successful

Inst. ID [24:21]	Error [20]	Description [19:0]
0000	0	Addressed location
0000	0	Data from the addresses location

(b) Failed

Inst. ID [24:21]	Error [20]	Description [19:0]
0000	1	Addressed location
0000	1	MBZ[19:11]-AIM error code [10:6]-ISEQ error code [5 :0]

Table 3.8: Result of a FEC read transaction

- **Writing FEC registers**

For a write operation on any of the addressable memory location of the ALTRO chip, the RCU executes the write transaction on the ALTRO bus. The result of each write instruction comprises two result memory locations: the first location contains the result of the command along with the address of the location whereas the data written to the addressed location is written to the next consecutive location in the result memory. The success or the failure of the execution of the write instruction is shown by a status bit as shown in table 3.9. A single location in the result memory could have been used to record the result of a single write transaction but two consecutive location are allocated to keep the consistency with respect to the result of a single FEC read transaction. If a write transaction is completed in broadcast mode then the result is still recorded with similar contents indicating the broadcast bit in the field of the addressed location. The RCU verifies the content of the write command to the FEC before it is executed. If the instruction written in the memory does not comply with the ALTRO instruction format or if the address does not belong to the allowed range, the instruction is not executed and the error code indicates the type of error in the instruction [23]. If the write instruction fails during the execution of the instruction then the error code generated from the AIM is written to the result memory. It describes the nature of the error encountered during communication from the RCU to the FEC.

- **FEC commands**

The ALTRO chip can be addressed by six different commands [19]. These commands can be executed individually to an ALTRO or issued in broadcast mode. The result of a command transaction is recorded in the result memory as shown in table 3.10.

- **ISEQ wait**

This command is used as an internal command of the instruction sequencer (ISEQ) to wait for a specified number of clock cycles. The result of this instruction is loaded in the result memory as shown in table 3.11.

- **ISEQ loop**

This command is used as an internal instruction of the sequencer to repeat a set of FEC instructions for a specific number of times, which is specified along with the command. The result of a loop command is written to the result memory according the format shown in table 3.12.

(a) Successful

Inst. ID [24:21]	Error [20]	Description [19:0]
0010	0	Addressed location
0010	0	Data written to the location

(b) Failed

Inst. ID [24:21]	Error [20]	Description[19:0]
0010	1	Addressed location
0010	1	MBZ[19:11]-AIM error code [10:6]-ISEQ error code [5 :0]

Table 3.9: Result of a FEC write transaction

- **ISEQ end**

This command is used as an internal command to the sequencer to end a sequence. The result of this instruction is loaded in the result memory along with the address of the last instruction executed. Table 3.13 shows the result of execution for the ENDSEQ as written in the result memory.

3.3.5 RCU decoder

The RCU decoder manages the interface of the DDL and the DCS to the complete address range of registers, memories and the command set of the RCU. It consists of five sub-modules as shown in figure 3.9, each one dedicated to a specific task described in the following.

- **Bus arbiter**

The RCU registers and memories can be accessed from the DCS or the DDL. Access to

(a) Successful

Inst. ID [24:21]	Error [20]	Description [19:0]
0001	0	Addressed location
0001	0	MBZ

(b) Failed

Inst. ID [24:21]	Error [20]	Description[19:0]
0001	1	Addressed location
0001	1	MBZ[19:11]-AIM error code [10:6]-ISEQ error code [5 :0]

Table 3.10: Result of a FEC command transaction

Inst. ID [24:21]	Error [20]	MBZ[19:16]	Description[15:0]
0101	0	0000	Number of wait clock cycles

Table 3.11: Result of the ISEQ wait internal command

(a) Successful

ID [24:21]	Error [20]	MBZ[19:16]	Description [15:0]
0100	0	0000	number of loops[15:12],loop Address[11:0]

(b) Failed

ID [24:21]	Error [20]	MBZ[19:16]	Description [15:0]
0100	1	0000	erroneous loop [15:12],number of errors[11:0]

Table 3.12: Result of an internal loop instruction

ID [24:21]	Error [20]	MBZ[19:12]	Description[11:0]
0111	0	X00	Address of last executed instruction

Table 3.13: Result of end sequence

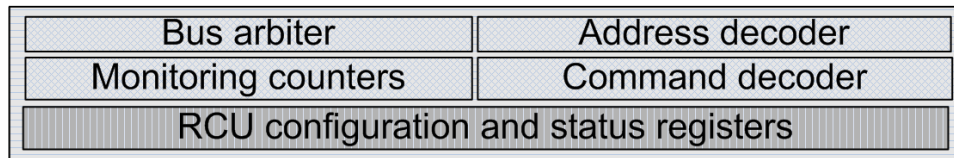


Figure 3.9: RCU decoder and its sub modules

Name	Address	Width	Access	Description
ACTFECLIST	0x5100	32	RW	Active FEC List
ALTROIF	0x5101	18	RW	Instruction error check selection: [17:16], CSTB programmable delay: [15:14] Tsampling / TLHC: [13:10] Nr. Samples / channel : [9:0]
TRGCONF	0x5102	17	RW	Trigger Source: [16:14], Trigger Mode:[13], L2 latency w.r.t. L1: [12:0].
RDOMOD	0x5103	4	RW	Mask RDYRX [3] , Sparse readout enable: [2], Execute sequencer on SOE/EOE: [1] , MEB mode:[0].
ALTROCFG1	0x5104	20	RW	Copy of the ALTRO-DPCFG1 register
ALTROCFG2	0x5105	25	RW	Copy of the ALTRO-DPCFG2 register
RCU version	0x5106	24	R	Last release date of RCU firmware version
BP version	0x5107	1	RW	Mapping of ACTFECLIST to CSW lines 0 for TPC and FMD, 1 for PHOS.
RCUID	0x5108	9	RW	RCU Identification register
SCEVL DLY	0x510A	13	RW	Time to perform SCEVL command
EN INT BA	0x510B	2	RW	Enable to interrupt operation
SCLK FREQ	0x800C	2	RW	Frequency of the slow control clock

Table 3.14: Configuration registers of the RCU, for detailed description see [23]

each memory and register in RCU is provided through an internal data and address bus named as RCU bus. The bus arbiter is responsible to decide the arbitration of the RCU bus to DCS or DDL. The DCS interface unit and SIU interface unit request the bus by asserting a bus request signal and the arbiter responds with the bus granted signal to each request from either interface. If the DCS and the DDL request the bus on the same clock edge, priority is given to the DCS because a single transaction from the DCS requires the occupation of the bus for two clock cycles, whereas DDL access requires minimum of sixteen clock cycles due to the DDL data exchange protocol [32].

- **Address decoder**

The address decoder decodes the address on the RCU address bus to provide the access of a memory location in the RCU to DDL or DCS. The enable signals to the registers and memories are generated after decoding of the address.

- **Command decoder**

The command decoder decodes the commands on the RCU address bus and initializes the respective unit in the RCU for execution of the command. It provides a start signal to the respective control finite state machine of a module.

- **Watchdog and monitoring counters**

There are a number of parameters in the RCU which are counted and accessible in read-only mode. They include all the ALTRO control signals and trigger pulses. The details of these counters along with their addresses are listed in reference [23].

3.3.6 Event manager

The event manager consists of an event manager controller and a trigger interface manager. The trigger interface manager is further partitioned into CTP trigger control and test trigger control modules as shown in figure 3.10.

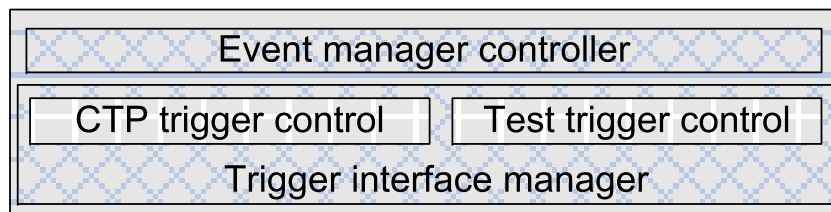


Figure 3.10: Structure of the event manager

- **Event manager control**

The event manager control unit interacts with the readout unit by issuing the trigger pulses of all types including L0, L1a, L1r, L2a and L2r. It receives the status of the readout unit and the data assembler to monitor and control the event readout. It registers the number of triggers in the pipeline and exhibits the status of the event handling. At the end of the event readout this unit issues the RPINC [19] command to the FECs. This unit measures the event readout time for every trigger with respect to the L2a pulse till the last word from the RCU is shipped to the DDL.

- **Trigger interface manager**

The trigger interface manager is partitioned into two blocks based on the trigger sources (section 3.2.5). Software triggers and hardware triggers are handled as test trigger by the RCU. The test trigger control unit receives the software command or the input pulse to generate triggers. This unit issues the L1 pulse to the FECs followed by a L2 pulse after a configurable delay.

The CTP trigger control unit manages the interface to the trigger receiver unit in the RCU firmware. Trigger pulses are decoded by the trigger receiver unit [5] and the CTP trigger control unit issues the respective pulses to the FECs. It communicates to the event readout unit and the data assembler to perform the readout for a valid trigger sequence [5].

3.3.7 Data assembler

The data assembler transmits the event data to the DDL. It is responsible to construct the CDH, read and format the event data from the data memories or the result memory and to construct the RCU trailer. The data assembler transmits the event data according to the format discussed in section 3.2.7. The data assembler is partitioned in four different modules to accomplish these tasks.

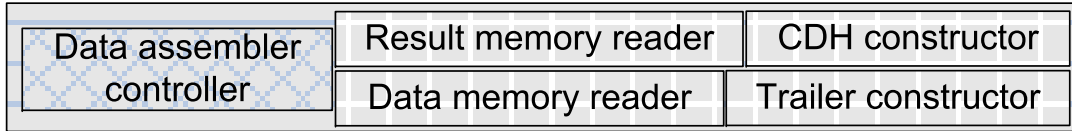


Figure 3.11: Sub-modules of the data assembler

- **Data assembler controller**

The data assembler controller consists of a finite state machine to control the three modules to construct the three different parts of the event data packet. It communicates to the readout unit to begin the event data formatting. This control unit signals to the SIU interface unit to mark the start and end of an event.

- **CDH constructor**

The CDH constructor module is started by the data assembler controller as soon as a L2 pulse is received. It reads the event FIFO (section 3.15) to extract the information for the CDH if the trigger source is set to CTP. When the trigger source is any of the test triggers it generates a dummy CDH mentioning that the trigger type is of test type.

- **Data memory reader**

This unit is responsible to read the event data from data memories, convert 40 bit data word to 32 bits data words and write these words to the SoB. It consists of a finite state machine and the respective address counters and data registers. This unit handles the control of the data flow to the SIU interface unit. A special signal X_{off} from the SIU interface unit to this module is assigned to indicate the link status. “Link full” signal is a standard DDL signal which is asserted when the link can not receive any more data from the front-end-electronics [32]. The SIU interface unit propagates this status to the data memory reader to pause the data transmission. The handling of the data flow is of very high importance in the system. It can be asserted anytime during the event data readout resulting in immediate pause of the data transmission, the data transmission must resume later for the same channel and event. This unit reports to the data assembler controller after the complete event data is sent to the SoB.

- **Result memory reader**

The RCU firmware can be configured to send the content of the result memory as payload for the SoD trigger or the EoD trigger from the CTP. This feature is useful when the event data is required to be analyzed with reference to the ALTRO chip configuration. This module in the data assembler reads the result memory and writes its content as event payload to the SoB.

- **Trailer constructor**

The details of the trailer words are mentioned in table 3.5. This module is responsible

of constructing and writing the trailer word to the SoB. The link status is checked before writing every word to the SoB. If the link full status is signaled from the SIU interface unit the write process pauses till the link is again ready to receive the remaining trailer words.

3.3.8 Monitoring and safety module

The Monitoring and Safety Module (MSM) is responsible to monitor the physical parameters of all the FECs in a partition and the temperature of the RCU board. This module is interfaced to the DCS interface unit on a dedicated data and address bus. This dedicated bus is utilized while the RCU bus is occupied for configuration or the event readout. The specific address range starting from 0x8000 is assigned to this module as mentioned in table 3.15. The access to this address range is not available from the DDL side. This module is partitioned in three sub-blocks based on the functionality and interface.

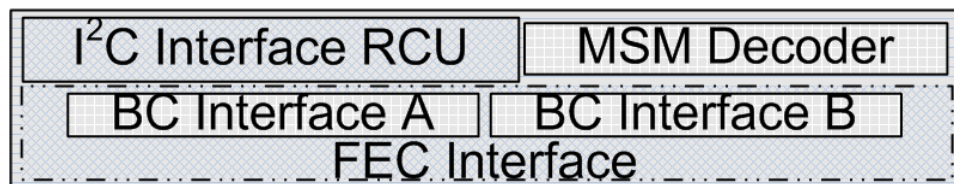


Figure 3.12: Monitoring and safety module top level

Name	Address	Width	Access	Description
SLC-RS-REG	0x8002	21	R	Result of the command after execution
SLC-ST-REG	0x8003	31	R	Status of the executed transaction
SLC-AD-REG	0x8005	16	RW	Address register
SLC-DT-REG	0x8006	16	RW	Data register
SLC-EX-CMD	0x8010	16	T	Execute command
RCU-EX-CMD	0x801A	16	T	Update RCU board temperature
RCU-BD-TMP	0x800D	16	R	Register indicating temperature of RCU

Table 3.15: The MSM command and register set

- **MSM Decoder**

The MSM decoder contains all status and command registers. It decodes the address and commands received from the DCS interface unit.

- **FEC Interface A-B**

The FEC interface unit for monitoring is divided into two blocks, each one dedicated to a single branch in the readout partition. This unit communicates to the board controller on the FEC to access the updated values of physical parameters. The FEC interface communicates to the MSM decoder to get the commands issued from the DCS to initialize the transaction on a FEC. The FEC interface for each branch generates its independent clock with configurable frequency for the serial communication on the FC bus.

- **RCU I²C Interface**

The RCU board is equipped with two on-board temperature sensors with ADCs. This

block communicates to these ADCs to start the conversion of the temperature from sensors. It begins by configuring the ADCs, then it updates the temperature values upon request from the DCS.

A read or write process to any of the board controller registers [14] using the FC bus is carried out by the following four steps listed here:

1. Write instruction in address register
2. Write data in data register (for write transaction only, not required for read)
3. Execute command (0x8010)
4. Read result register

After successful execution of this procedure the contents of the result register contains the contents of the addressed register of the board controller.

3.3.9 SIU interface unit

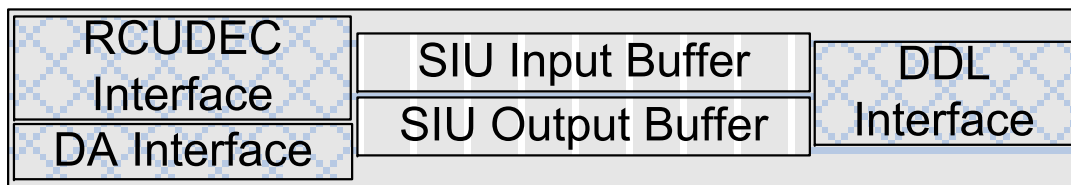


Figure 3.13: SIU interface unit top level

The SIU interface unit functions as an interface to the SIU daughter card, which is the first component in the DDL to a readout partition. It is interfaced to RCU decoder to access the complete addressable range of RCU memories and commands through the RCU bus. A dedicated interface is assigned for the readout of event data. It contains two FIFO modules which work as input data buffer from the DDL and output readout buffer. The SIU interface unit implements the standard DDL protocol according to the DAQ interface to front-end-electronics [32].

3.3.10 DCS interface unit

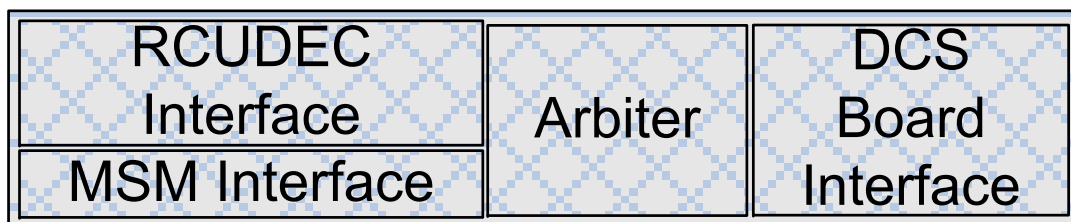


Figure 3.14: DCS interface unit top level

The DCS interface unit functions as an interface to the DCS daughter card, which acts as a local agent of the ALICE DCS installed on the readout partition. It is interfaced to the RCU

decoder to access the complete addressable range of the RCU memories and registers through the RCU bus. It consists of two interface control units and a bus arbiter [5]. Bus arbitration is required when the DCS addresses the MSM register set of the memories under RCU bus. The MSM module in the RCU is interfaced to the DCS interface unit via a dedicated bus [5] enabling continuous monitoring of the FECs and the RCU board.

3.3.11 Trigger receiver unit

The trigger receiver unit (TRGRx) is an interface to the CTP implemented in the RCU firmware. It is responsible to process and decode the trigger sequences from the CTP. The trigger interface between the RCU and the DCS board consists of two signals named as “serial channel B” and “L0/L1 pulse” [5]. This module decodes the data received on serial channel B and L0/L1 triggers. It communicates to the event manager to deliver the trigger pulses corresponding to the received trigger sequence. The data assembler reads the event FIFO in this module to construct the event CDH. The main components [5] of the TRGRx unit are shown in figure 3.15. The TRGRx unit is assigned the address range from 0x4000 - 0x4fff for status and configuration parameters. The busy box firmware and the RCU firmware are equipped with the same TRGRx modules to keep the consistency in decoding and accepting the trigger sequences from the CTP.

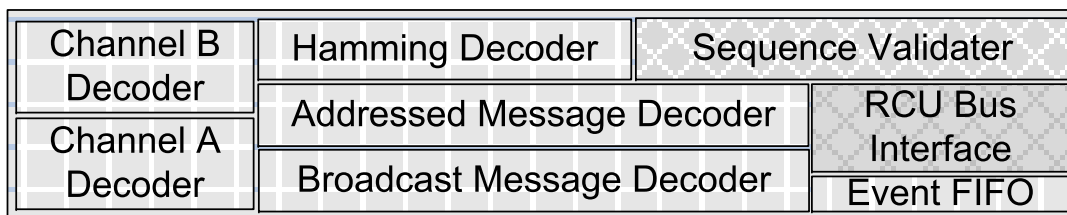


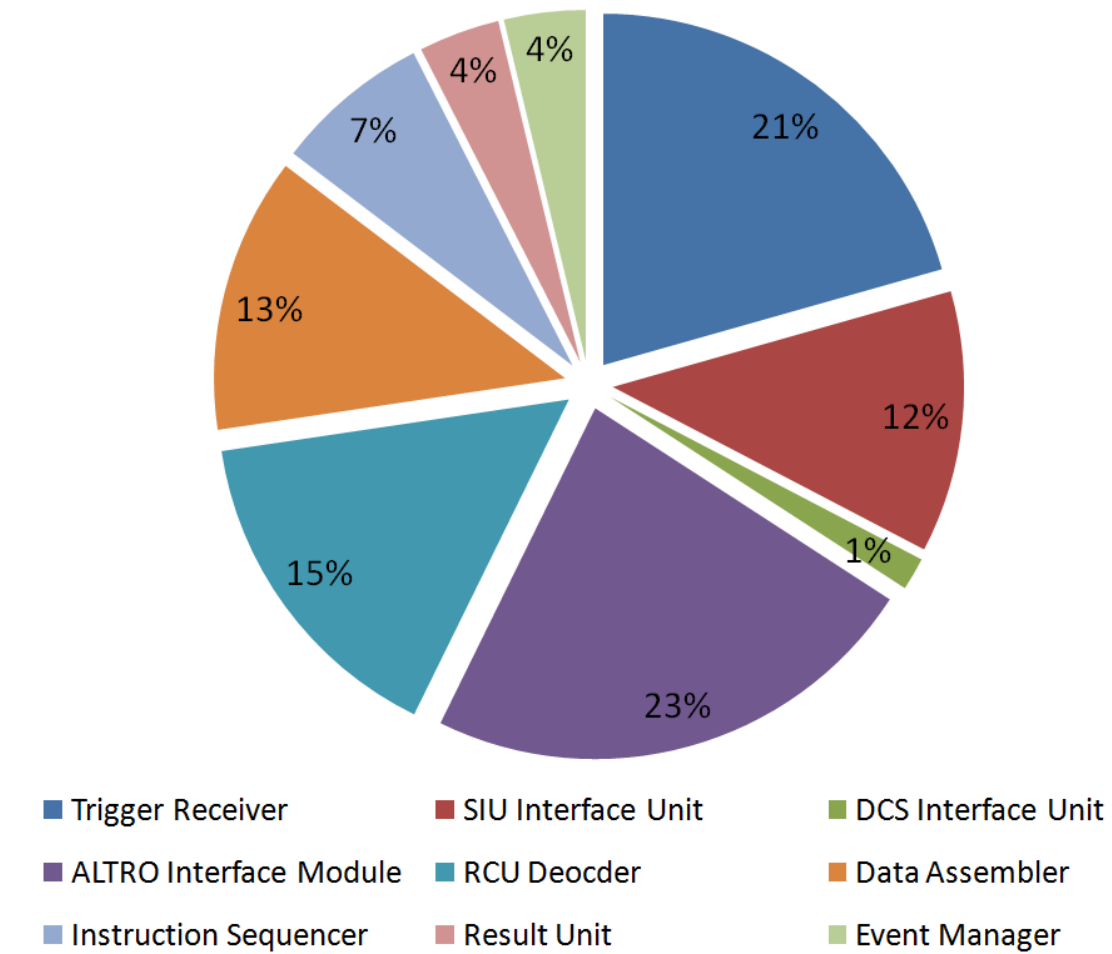
Figure 3.15: Trigger receiver unit top level

3.4 Implementation of the RCU firmware

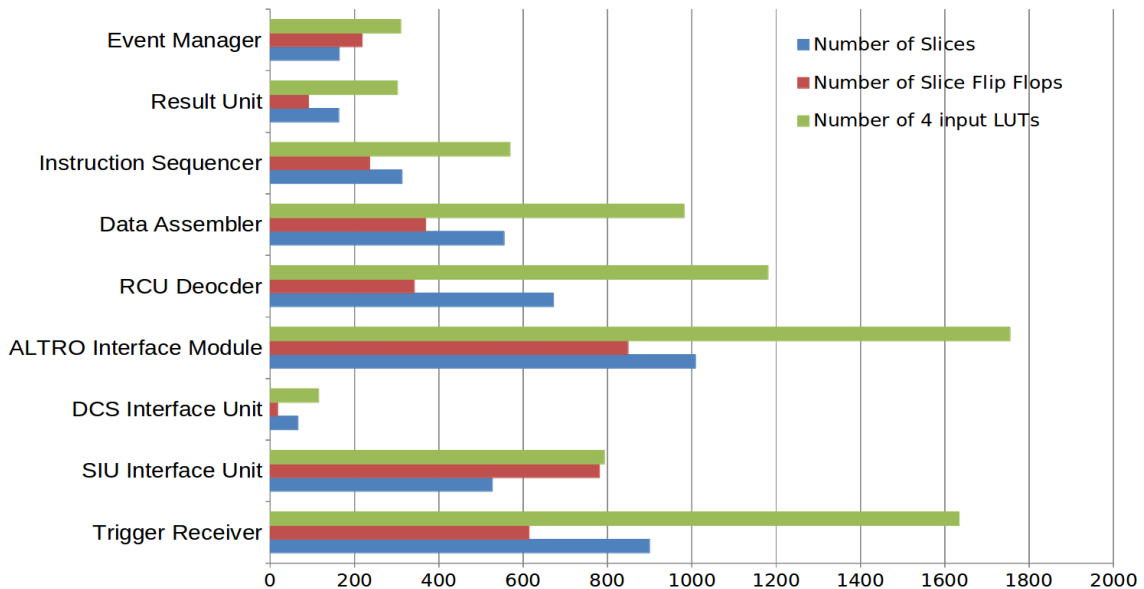
All modules of the RCU firmware are written in VHDL. The RCU firmware is synthesized and implemented targeting the Xilinx FPGA device “XC2VP7-7-ff672” using the Xilinx ISE software suit. Figure 3.16 shows the logical resources used by each of the modules of the RCU firmware. The ALTRO interface module and the trigger receiver unit are the two largest modules of the RCU firmware as shown in figure 3.16(a). As shown in table 3.16, the logical resources of the FPGA device have been utilized 99% by the RCU firmware. The top level view of the Xilinx device is presented in appendix A (figure A.6), as a comparison to illustrate the space and resources utilization.

3.5 Simulation and verification of the RCU firmware

As described earlier the RCU functions as the key interface between CTP, DAQ, DCS and FECs for all kind of transactions to accomplish event readout. All of the components in the readout system interact with the RCU concurrently and independently. The RCU is required to report any kind of inconsistency or functional failure of any component. Simulation, verification and reliable testing of the RCU firmware to confirm its functionality and detection of



(a) Pie chart to display the relative size of each module



(b) Utilization of logical resources by main modules

Figure 3.16: Relative size and utilization of the logical resources by modules of the RCU firmware

Name of the device resource	Available	Used	Percentage usage
Slices	4928	4912	99
Slice Flip Flops	9856	4342	44
4 input LUTs	9856	8719	88
bonded IOBs	396	248	62
TBUFs	2464	220	8
BRAMs	44	33	75
DCMs	4	3	75
GCLKs	16	8	50

Table 3.16: Device utilization summary for the targeted device: 2vp7ff672-7

any kind of error or failure is a complex task. These tasks are accomplished by three different independent procedures as discussed in following sections.

3.5.1 Simulation and functional verification

A simulation of each module in RCU firmware was carried out during the development and implementation phase. Test benches written in VHDL for each module are integrated to build a comprehensive simulation environment for the complete RCU firmware. The top level architecture of the test bench is shown in figure 3.17. This simulation environment is used for functional² simulation of the RCU firmware and After Place and Route (APnR) simulation. The results and the log files generated for functional and APnR simulations are compared to verify the timing and the event data handling by the RCU firmware.

3.5.1.1 Simulation controller

The simulation controller accepts a text file which lists the tasks to be carried out during a simulation. These tasks are assigned specific names used as commands in the simulation environment based on their role in the simulation as listed in table 3.17. A text file name as “script.txt” lists the tasks in a sequence for a complete simulation. The simulation controller generates two reports named as “sim-result.log” and “dcs-siu.log”. “sim-result.log” is written to record each transaction carried out from the DDL or the DCS along with the data and the location addressed. Each transaction is recorded with a time stamp in this file. The file “dcs-siu.log” contains the data transferred between the RCU firmware and the DDL or the DCS along with the time stamp in terms of clock cycles. Both of these files are mainly used to verify the execution time of any transaction and verify the handling and processing of data by the simulated modules inside the RCU firmware. Two examples of “script.txt” performing different simulations are shown in appendix A (figure A.1).

3.5.1.2 DCS stimuli process

This process emulates the DCS in the simulation environment. It accepts the tasks from the simulation controller as listed in the script.txt file. It translates the text based command to activate the DCS control signals, address and data bus.

²Functional simulation & verification in this work is referred to RTL level simulation of the each module and complete RCU firmware before synthesis

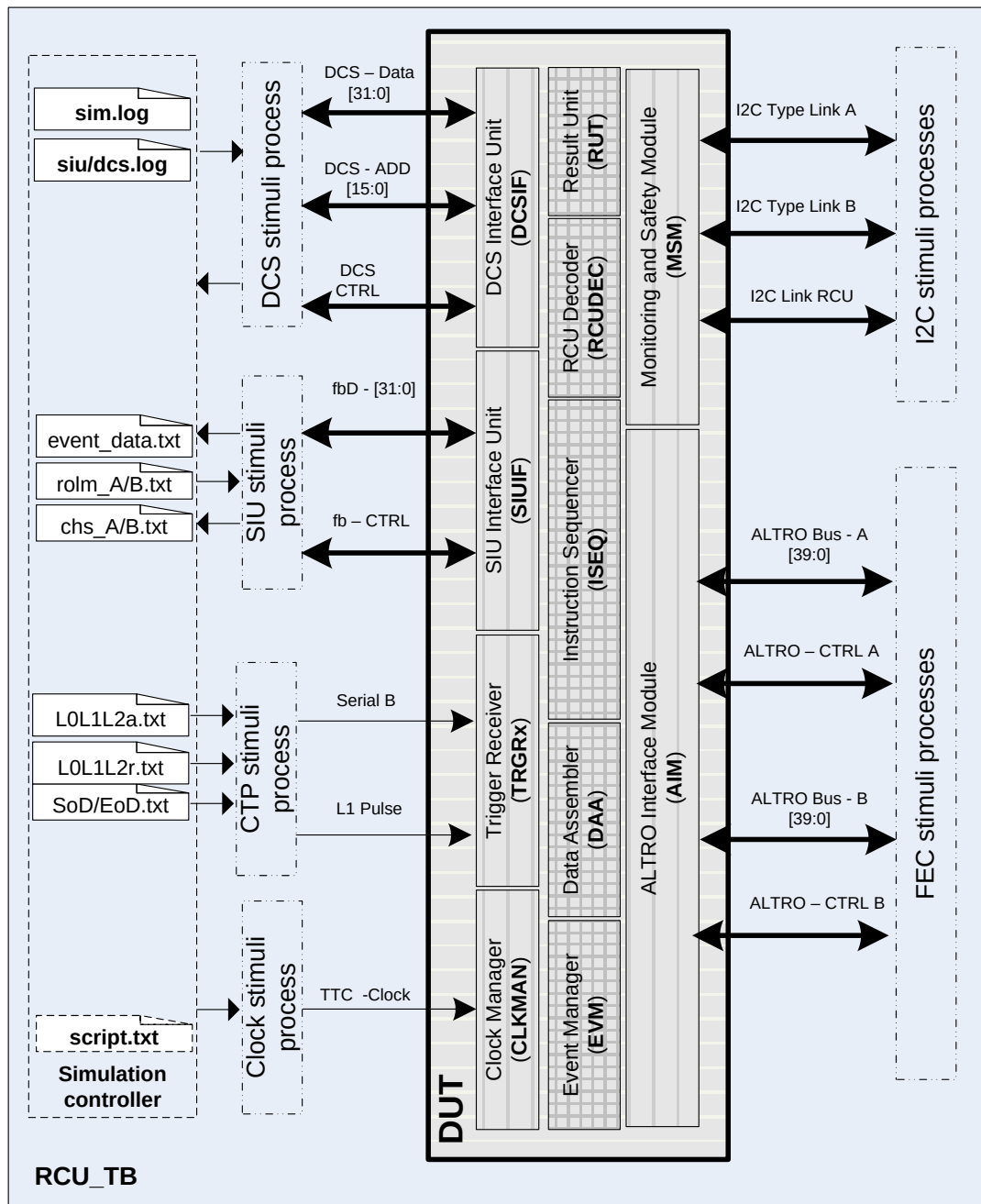


Figure 3.17: Architecture of the RCU test bench

3.5.1.3 SIU stimuli process

This process emulates the role of the DDL in the simulation environment. It accepts the tasks from the simulation controller as listed in the “script.txt” file. This stimuli implements a set of transactions based on the DDL functions to the front-end-electronics as listed in table 3.17. All of the DDL functions for front-end-electronics are translated according to the DDL communication protocol [32]. This stimuli is used to read and write all of the RCU memories and it receives the event data. The “ENSUIU” command is issued to the simulation controller followed by the tasks to be accomplished from the DDL to the RCU. The readout list memory is configured by this process by writing the channel addresses from the file named as “rolm.txt”. This file contains simulated event sizes for the respective channel as well. The SIU stimuli

Command	Stimuli process	Description
DCSWR	DCS	Write transaction from DCS
DCSRD	DCS	Read transaction from DCS
ENSIU	SIU	Enable DDL stimuli
TXGAP	SIU	Transfer gap (wait command)
ENCTP	simulation controller	Enable CTP stimuli
EOBTR	SIU	End of block transfer from DDL
STBWR	SIU	Begin a memory block write from DDL
RDYRX	SIU	Open DDL to receive event data
SOD	CTP	Send SoD event from CTP stimuli
L0L2a	CTP	Send L0L1L2a trigger sequence
L0L2r	CTP	Send L0L1L2r trigger sequence
EOD	CTP	Send L0L1L2r trigger sequence
-END-	simulation controller	To terminate a stimuli in simulation

Table 3.17: Commands for simulation tasks

process receives the data from the Device Under Test (DUT) “RCU firmware” in the simulation, and writes into a result file named “event-data.txt”. Received data is decoded by this process and the written file contains all three segments of the event data packet according to the RCU data format. The ALTRO channel trailers are extracted from data and written to a separate file to verify that the channel addresses found in the event payload are consistent with the addresses listed in the readout list memory. An example of “event-data.txt” is shown in appendix A (figure A.3).

3.5.1.4 CTP stimuli process

The CTP stimuli process is responsible to issue four different trigger sequences to the DUT (the RCU firmware). It is initialized by the simulation controller when any of the trigger sequences as listed in the table 3.17 (CTP) are found in the “script.txt”. This process reads the data from three files “L0L2a.txt, L1L2r, SoD.txt and EoD.txt” to generate trigger sequences and asserts respective bits on the serial B channel and L1 pulse line. The process is started by the simulation controller upon the command “ENCTP” to accomplish the tasks listed in “script.txt”. The simulation environment can send a number of any triggers sequences by calling the respective tasks.

3.5.1.5 FEC stimuli process

The FEC stimuli process consists of two sub-functions, one dedicated for each branch in a partition. This process communicates on the ALTRO data bus and the ALTRO control signals. It accepts the ALTRO read, write and command transactions to verify that the RCU firmware abides by the timing requirements of the control protocol. For read transactions this process returns a dummy value verifying the read process by the RCU firmware.

The main function of this process is to interact with the RCU firmware to simulate and transmit the event data for both modes of event readout. For full readout mode this process verifies that the CHRDO command from the RCU firmware meets the ALTRO protocol for timing requirements. It reads the file “rolm.txt” to get the event size to be sent from the addressed channel. This process can send a fixed pattern or an ascending ramp pattern as event data.

The ascending ramp pattern is used to verify that the padding (0x2aa) is removed correctly as discussed in section 3.4.

3.5.1.6 I²C stimuli process

This stimuli process communicates to the RCU firmware on the I²C interface to the temperature sensors and on the FC bus to the board controllers. It verifies that the I²C protocol is followed. It verifies that the RCU firmware executes all three transactions (read, write and command) to board controller on the full range of the configurable frequency of the clock.

3.5.2 Hardware test of the complete readout system

Hardware tests of the RCU firmware were carried out in the laboratory in order to confirm the compatibility of the RCU firmware with all other components in the readout system. The most critical aspect of this test is to verify that the system does not get stuck due to a varying size of the data volume at random trigger rate. A real time verification is done to confirm that the data received from the FECs by the RCU and shipped to the DAQ is not corrupted and that it is formatted according to the RCU data format.

3.5.2.1 Hardware setup for the test

The hardware setup consists of one single TPC readout partition as shown in figure 3.18. The partition used to test the readout system has several components, which are described here.

- **Single readout partition**

This single readout partition consists of the maximum number of readout channels for one readout partition in the TPC. It consists of two branches, branch A and branch B containing 13 and 12 FECs respectively. The partition can be configured to emulate other partitions where the number of front end cards are 20, or 18. These 25 FECs are controlled by one single RCU board containing the two daughter cards, the SIU card and the DCS card as shown in figure 3.18.

- **Local trigger unit**

The Local Trigger Unit (LTU) is used to emulate the ALICE CTP by generating trigger sequences to perform the event readout. The trigger sequences are fully programmable for their timing parameters through software. They can be generated with random or fixed delays with known and random errors. These trigger sequences can be utilized to test the robustness and the error reporting capabilities of the readout system [3].

- **Busy box**

The busy box controls the consistency between the event readout and the CTP as discussed in section 2.7.3. For every trigger issued by the LTU, the busy box verifies that the respective event data packet has been received by the DAQ before a new trigger is issued to the readout system.

- **DAQ computer**

The DAQ computer with the DATE (ALICE Data Acquisition and Test Environment) software installed is used to emulate the data acquisition environment in the ALICE experiment. This computer is equipped with three RORCs such that it can connect to three different readout partitions at the same time.

- **DCS computer**

The DCS computer is used to communicate with the DCS card on the RCU motherboard. The main functions carried out by this computer are to configure the Xilinx device on the RCU board and to monitor the FECs and the RCU board. In the test environment this computer is used to host the communication and monitoring environment between the LTU, the busy box and the RCU board.

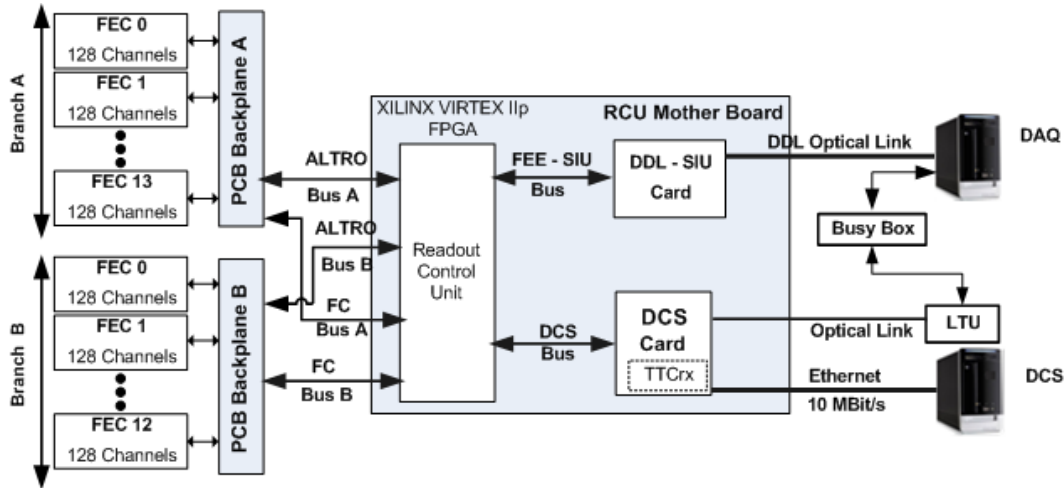
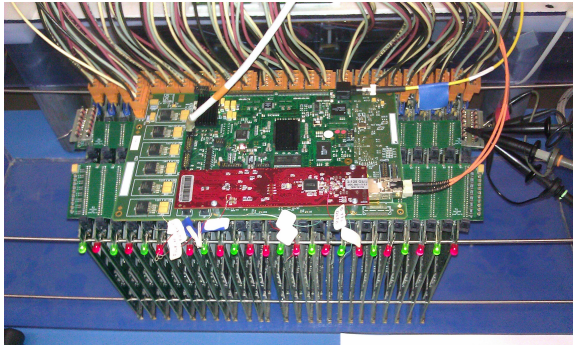
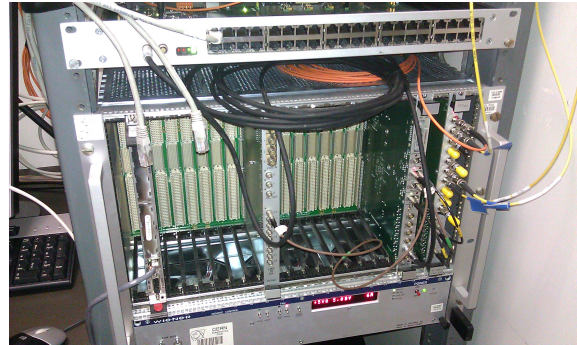


Figure 3.18: Hardware setup of one single TPC readout partition



(a) Single readout partition with 25 FECs



(b) Busy box and LTU (CTP emulator)

Figure 3.19: Test set-up for hardware test

3.5.2.2 Hardware test for the compatibility of complete readout system

The single TPC readout partition in the test setup is configured under various settings and stress tested for a specified number of hours as listed in table 3.18. The purpose is to verify the event handling with zero suppressed data with variable volume at various input trigger rates. Two different trigger schemes are applied to the system under test. A continuous input trigger with fixed bunch crossing (fix BC) implying a fixed time between the triggers by the LTU. In the real world the arrival of the triggers is however randomly spaced in time. A continuous stream of triggers with random time spacing is applied to emulate this situation. The average trigger rate as mentioned in the first column of table 3.18 is varied from 100 Hz to 300 kHz to cover the maximum range of input trigger rate.

Trigger Settings (rate - kHz) (time spacing)	RP(number of channels) (2304/2560/3200)	MEB settings (busy box)	Duration (hours)
0.10 - Fix BC / random	RP0 / RP1 / RP3	1-2 / 4	4
0.25 - Fix BC / random	RP0 / RP1 / RP3	1-2 / 4	4
0.50 - Fix BC / random	RP0 / RP1 / RP3	1-2 / 4	4
0.75 - Fix BC / random	RP0 / RP1 / RP3	1-2 / 4	4
1.0 - Fix BC / random	RP0 / RP1 / RP3	1-2 / 4	4
1.0 - Fix BC / random	RP0 / RP1 / RP3	1-2 / 4	4
3.0 - Fix BC / random	RP0 / RP1 / RP3	1-2 / 4	8
5.0 - Fix BC / random	RP0 / RP1 / RP3	1-2 / 4	8
7.0 - Fix BC / random	RP0 / RP1 / RP3	1-2 / 4	8
9.0 - random	RP0 / RP1 / RP3	1-2 / 4	8
10.0 - random	RP0 / RP1 / RP3	1-2 / 4	8
20.0 - random	RP1	1-2 / 4	8
30.0 - random	RP1	1-2 / 4	12
50.0 - random	RP1	1-2 / 4	12
100.0 - random	RP1	2 / 4	12
200.0 - random	RP1	2 / 4	12
300.0 - random	RP1	2 / 4	24

Table 3.18: Compatibility tests of the complete integrated readout system

As an example, figure A.4 in appendix A shows the console screen while the test is being carried out in the laboratory. The hardware test setup is configurable for all possible numbers of FECs in any readout partition. The tests are conducted for the full range of the number of channels in the TPC readout partitions. All combinations of the test configurations mentioned in table 3.18 are verified for full and sparse readout modes and the trigger counters on the LTU and the RCU along with the number of events received by the DDL and the busy box are continuously monitored during the tests.

3.5.2.3 Hardware test with real time data verification

As described earlier, the RCU firmware receives the event data from all channels in a readout partition and transmits the data then to the DDL according to the channel list specified in the readout list memory. It formats the data according to the ALICE data format as shown in tables 3.2 and 3.4. The removal of the ALTRO channel padding [19], the verification of the channel headers and the concatenation of data words during readout requires extensive verification to avoid any kind of data corruption or misalignment. Data verification during an event readout is carried out through two different procedures. For each of the procedures the pedestal memories in the ALTRO channels are written with known test patterns or known event data. The main steps of the procedures to conduct hardware data verification are briefly discussed here.

- **Data verification for full readout**

The data verification for the full read mode is carried out to meet two main objectives; 1) To verify the event data handling and formatting by four different data patterns loaded in the pedestal memories 2) To test the handling of events during the complete usage of the ALTRO multi event buffers.

Such a test is carried out in the following steps as shown also in the flow chart in figure 3.20(a).

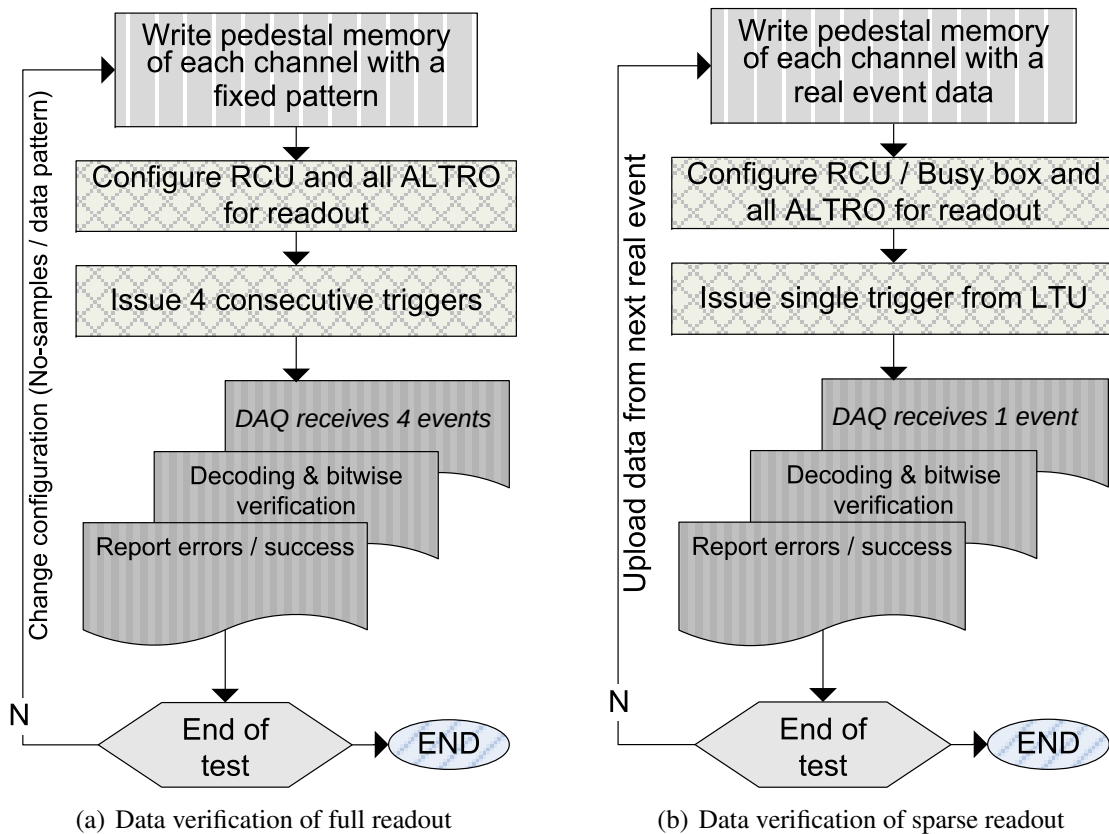


Figure 3.20: Flow chart of real time data verification

1. Write pedestal memories with fixed patterns

The pedestal memories in the ALTRO chips are configured with a known pattern after each iteration of the test. The test is based on four iterations utilizing four different patterns.

2. Configure RCU and ALTROs for readout

The RCU memories are configured to perform full readout. The ALTRO chips are configured to read the contents of the pedestal memories after each trigger and send them to the RCU as event payload. The data pattern written to the pedestal memories is changed after 1000 events, whereas the number of samples from each channel is configured for each event. In this way the number of 10 bit samples is varied from 5 to 1008 in order to vary the amount of data sent to the RCU.

3. Issue four consecutive test triggers

Four consecutive software triggers are issued from the DCS without opening the DAQ link so all of the event buffers on ALTRO channels are occupied with data.

4. Decoding and bit wise verification

The DAQ link is opened to receive four event data packets from the RCU. First the format of each event packet is decoded and verified to meet the required format shown in table 3.4. The received event data packet is decoded and compared with the pattern written in the pedestal memories. The event data is checked for the order of data from the channels. Any kind of corruption in the data or alignment is reported. This test may be used as well to find and locate any bit which stuck at '0' or at '1' in the ALTRO event memory.

- **Data verification for sparse readout mode**

As described earlier the sparse readout is efficient because empty channels are skipped to minimize the dead-time due to the CHRDO command from the RCU to the ALTRO channels. It involves building the hit list memory in the RCU by the FEC board controller. The verification of the correct functionality of this mode of operation is carried out by replaying real events from proton-proton collisions. The data set of 2300 events from different readout partitions was selected for replay on the partition in the test set-up. The test procedure as shown in the flow chart in figure 3.20(b) is described here. As an example, figure A.5 in appendix A shows the console screen while the test is being carried out in the laboratory.

1. **Write pedestal memories with real event data**

The data from real event is decoded to extract the channels to be written with their respective event data. This data is written to the pedestal memories of the specific ALTRO channels. This step is repeated for each event during this test.

2. **Configure RCU and ALTROs for readout**

The RCU is configured to perform sparse readout. The ALTRO chips are configured to read the contents of pedestal memories after each trigger and send it to the RCU as event payload. Zero suppression and number of samples defining ALTRO data acquisition window are set and the readout list memories are written with the complete list of channels.

3. **Issue single trigger from LTU**

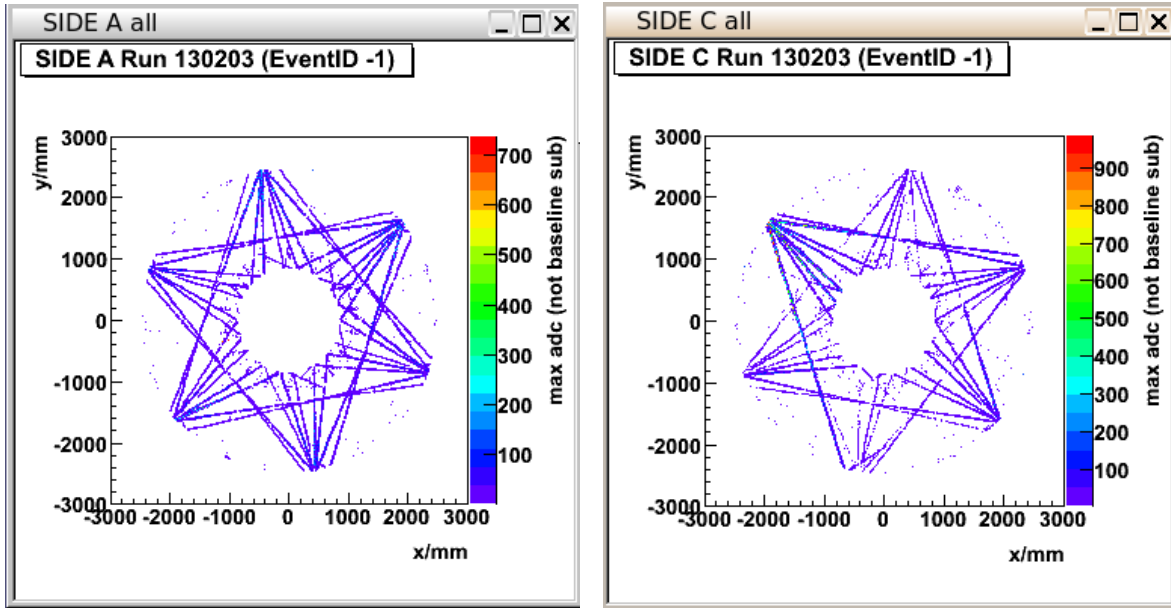
The LTU is used to emulate the CTP in this test. It issues a valid trigger sequence to acquire the event data. The busy box receives the same trigger sequence. The LTU, the busy box and the RCU firmware are configured with same trigger related configurations to keep the consistency in terms of decoding of trigger sequences.

4. **Decoding and bit wise verification**

The DAQ link is opened to receive the event data packet from the RCU. First the busy box verifies that the event ID received by the DAQ in the CDH matches with the event ID sent by the LTU to the RCU. If the event ID sent by the DAQ to the busy box and the one sent by the LTU does not match, the test is terminated and a failure is reported. After verification of the event ID, the format of each event packet is decoded and verified to meet the required format shown in table 3.4. The received event data packet is decoded and compared to the event data written in the pedestal memories. The event data is checked for the order of channels data. Any kind of corruption in data or alignment is reported.

3.5.3 Final qualification and laser run

After the compatibility test and the data verification in the test set-up, all of the Xilinx devices (RCU) in the readout partitions of the ALICE TPC are programmed with the tested RCU firmware. A final qualification of the event data handling is carried out at varying trigger rates with different configurations of the ALICE TPC. Both readout modes are tested for the complete TPC as listed in tables A.0(a) and A.0(b) of appendix A. The objectives of these tests are the same as the compatibility tests conducted in the test set-up. The trigger mode is set to random and the input trigger rate is varied to cover a wide range. All trigger counters from the CTP emulator, RCU firmware and busy box are continuously monitored to verify the consistency in number of issued triggers, data flow and number of events received.



(a) Laser event on TPC A - side

(b) Laser event on TPC C - side

Figure 3.21: Final qualification of RCU firmware by the laser run

Finally the TPC laser system (section 2.6.2) is utilized to verify the event data by reconstructing the tracks produced by the laser. For this purpose the laser system is activated and the laser tracks are generated synchronously with the arrival of the trigger sequences that also control the readout from the RCUs. The data is reconstructed as shown in figure 3.21 for a final verification of the event data handling by the RCU firmware and by the complete readout system. After this the RCU firmware is qualified for release and to be utilized in physics data taking mode.

3.6 Conclusion

The RCU firmware is the key component of the TPC FEE for the event data readout, monitoring and control of the FEE components. It was designed, developed and implemented successfully to meet all of the requirements of the readout and data acquisition system. It was extensively simulated to verify its functionality and was tested in the laboratory for the verification of its compatibility with the other components in the readout system. New test procedures to verify the complete readout system have been developed and used for the commissioning of the RCU firmware.

Chapter 4

Characterization and performance evaluation of the ALICE TPC readout system

The TPC readout system is divided into 216 independent readout partitions. Each partition is triggered through an independent link from the CTP and is steered by a single RCU linked to the DAQ through a DDL for data acquisition. A single readout partition in a test set-up is used to characterize the basic parameters related to the performance of the full TPC readout system. A complete performance evaluation of the readout system is carried out with the TPC at the ALICE experiment. This chapter describes the procedures to assess the performance of the ALICE TPC readout system. The results are presented and discussed in detail in the respective sections.

4.1 Characterization of the TPC readout system

Every transaction between the RCU and the FEC is characterized for its timing behavior. The hardware set-up of one single TPC readout partition is described in section 3.5.2.1 and shown in figure 3.19.

4.1.1 Configuration of the readout electronics

Configuration of the complete TPC readout system is carried out concurrently for all of the readout partitions. The TPC readout electronics can be configured either through the DCS cards or through the DDL links. The DCS and the DAQ use identical values for the configuration parameters from a common data base. The time taken to configure a single RCU depends on the number of channels to be readout in the readout partition. This time varies for three different sets (partitions) of front end cards. The procedure to configure the FECs by the RCU through the instruction sequencer is discussed in section 3.2.3 and section 3.3.3.

A single write transaction is initialized from the RCU by asserting CSTB and WEN lines to the FEC following the ALTRO bus protocol, which is summarized in section 3.3.2.1. Each write transaction to the FEC by the RCU takes $0.55\ \mu\text{s}$, the measured timing diagram of a single write transaction is shown in figure 4.1. Each ALTRO channel contains 7 configuration registers and 1024 locations for pedestal memory. A single location of the pedestal memory is written through two write transactions, in the first transaction the address of the pedestal memory location is written to the PMADD register followed by writing data for that location in the PMDATA [20]. There are 6 global configuration registers for each ALTRO chip which are common to all of the channels in the chip. The ALTRO global and the channel specific

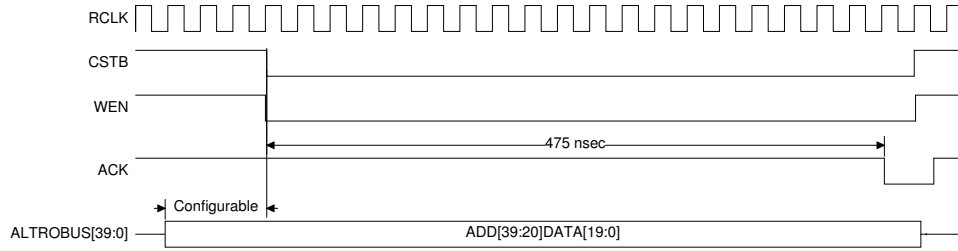


Figure 4.1: Timing diagram of a single channel write transaction

registers are configured for the parameters related to data processing. Full configuration of an ALTRO along with its pedestal memories requires 32885 write instructions from the RCU. The RCU takes a total time of 144 msec to configure a single FEC with 8 ALTRO chips. The total time needed to configure different partitions with three sets of FECs from the DCS is 2.67 seconds, whereas the DDL requires 1 second [11] configuring with identical values through similar transactions between the RCU and the FECs. This time does not include the time needed to transfer the data from the configuration data base to the RCU. It varies depending on the mode of configuration (DCS/DAQ). The DDL is primarily designed and used for data acquisition in the ALICE experiment, it is capable of configuring the front end electronics (RCU and FECs) as well. When the TPC is set to get configured by the DDL, configuration process is initialized and carried out on all LDCs concurrently and carried out in parallel on all readout partitions of the TPC.

The DCS or the DDL configures the instruction memory in the RCU with sequences to execute on FECs. All of the 216 RCUs run these sequences independent of each other (section 3.3.3). Each register in all ALTRO channels can be written in broadcast, where the ALTRO address field is set with specific register address along with asserting the broadcast bit in the ALTRO address field [20]. In the current configuration schemes each register in an ALTRO channel is written independently for its specific configuration, so the writing of ALTRO and channel parameters in broadcast mode is not in practice. The configuration from the DCS is executed via the Front End Electronics (FEE) server, a software which runs independently in all of the 216 readout partition on their respective DCS boards [16]. The configuration from the DDL is executed on all readout partitions in parallel to each other.

4.1.2 Trigger and clock synchronization

The RCU provides the readout clock and the ADC sampling clock to the front end cards. The sampling clock is configurable for four different frequencies as listed in table 4.1. The trigger sequence from the CTP which is synchronized with the 40 MHz clock is received by the RCU. As described in 2.7.1.2 the ALTRO chip requires trigger pulses of two levels; the L1 and the L2. The L1 pulse to the FECs is synchronized with respect to the sampling clock and it is asserted for two clock cycles. The L2 pulse is issued from the RCUs to all of the FECs after decoding of the L2 message and this pulse is synchronized with the readout clock (40 MHz) [20]. The trigger sequences which are issued from the ALICE CTP are synchronized with the 40 MHz clock, whereas the TPC sampling clock (ALTRO ADC clock) is of 10 MHz frequency. The trigger sequence is issued and decoded with respect to the 40 MHz clock. The L1 pulse which is issued from a clock domain of 40 MHz frequency to a clock domain with 10 MHz frequency can have four values for the phase difference with respect to the ALTRO sampling clock as shown in figure 4.2(a). This phase difference in arrival of the L1 pulse is calculated

by the RCU and sent in the RCU trailer [23].

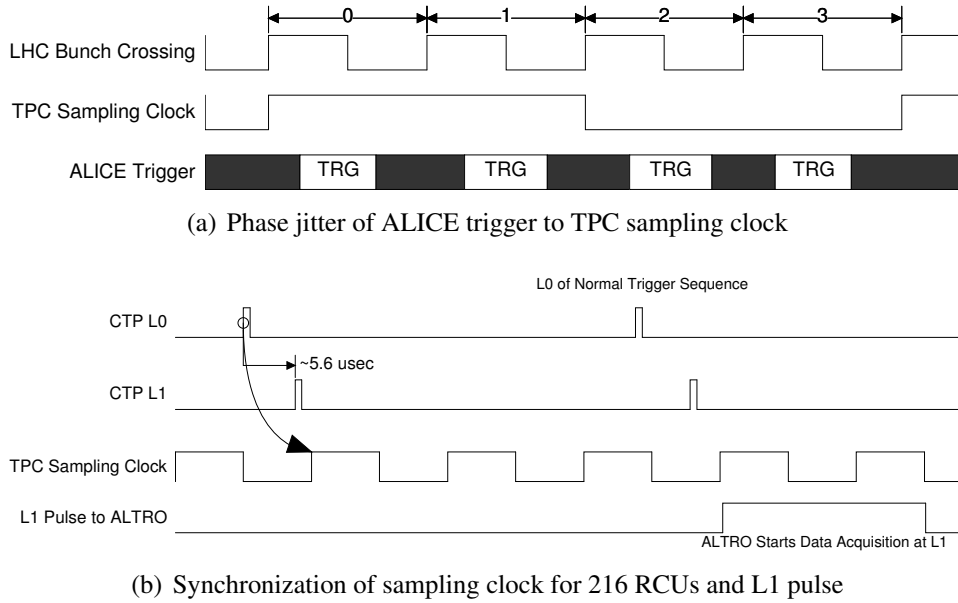


Figure 4.2: ALICE trigger synchronization with the TPC sampling clock

- **The synchronization of Readout Partitions**

The L1 trigger is issued from the RCU to the FECs in all partitions. It is extremely important to make sure that the data acquisition in all of the ALTRO chips begins at the same time so the sampling clock delivered from all of the RCUs must be synchronized. Synchronized start of the data acquisition in all of ALTRO chips is important to avoid jitter in the event data. Jitter in the event data will affect position resolution of the tracks in the TPC. This synchronization is done upon receiving the L0 pulse of the SoR trigger sequence as shown in figure 4.2(b). The synchronization of the sampling clock from all of the 216 RCUs is carried out by issuing a command by the DCS or the DDL. After configuration, a command named as arm synchronization is issued to reset the sampling clock upon the SoR trigger [23]. This trigger sequence is a special class of trigger issued from the CTP [3], where the event data is not readout. The synchronization of all of the 216 RCUs is essential to make sure that all of the ALTRO chips start and finish their data acquisition for a given number of samples on the same clock edge of the TPC sampling clock.

Settings	Sampling clock
00	20 MHz
01	10 MHz
10	5 MHz
11	2.5 MHz

Table 4.1: The configuration of the ALTRO sampling clock in ALTRO interface register [23]

- **The synchronization of the TPC pulser systems**

The TPC is equipped with two pulser systems; the gating grid pulser and the calibra-

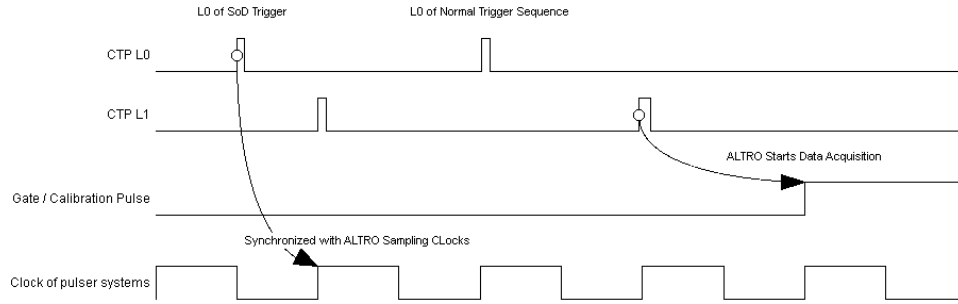


Figure 4.3: Synchronization of the TPC pulser systems

tion pulser as explained in section 2.5 and section 2.6 respectively. The state transition of the gate pulser is carried out upon the L0 pulse or the L1 pulse (configurable) by issuing a pulse of configurable duration from gating the grid pulser controller to the voltage drivers [24]. This pulse is required to be synchronized with the TPC sampling clock to synchronize the start of data acquisition in the ALTRO chip and the opening of the gating grid (section 2.5). A copy of the ALTRO sampling clock is generated internally in the gating pulser firmware and this clock is used to synchronize the gating pulse from the pulser controller. A mechanism to synchronize the system clock of the gating pulser firmware with the ALTRO sampling clock is implemented in the gating pulser firmware [24]. As shown in figure 4.3, the system clock is reset to get the same phase as the TPC sampling clock and upon the L0 pulse of the SoR trigger sequence.

The calibration pulser is used to issue a pulse which is configurable for its shape and amplitude on TPC cathode pads (section 2.6). The mechanism similar to the one implemented in the gating grid pulser firmware to synchronize the pulse from calibration pulser is implemented in the calibration pulser firmware as shown in figure 4.3.

4.1.3 Monitoring and safety

The RCU monitors its own board temperatures at two different locations. The RCU communicates to the board controller of each FEC in the partition to readout the updated values of the measured parameters on the FC bus. Both of these functions are described as under.

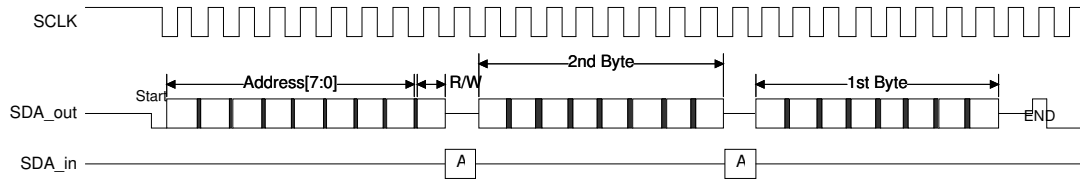
- **Monitoring of the RCU board**

The RCU board is equipped with two temperature sensors, one placed next to the voltage regulator and the second one next to the DCS board connector. The communication between both corresponding ADCs and the RCU firmware is based on the I²C protocol. The RCU gets these updated temperatures values as I²C master.

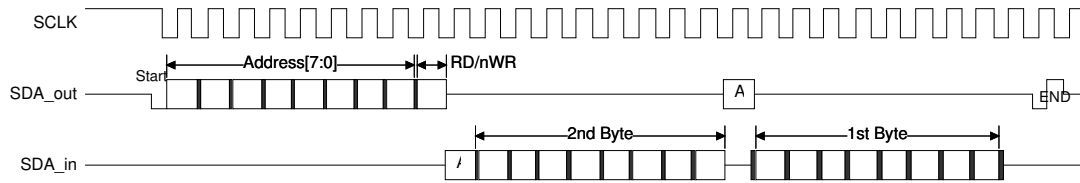
- **Monitoring of front end cards**

Temperatures, voltages and current levels of a front end card are measured by two 4 channel ADCs. The board controller on each FEC continuously monitors and updates these parameters. The RCU communicates to all of the board controllers through the FC bus [14] as described in section 2.7.1. The RCU board is used as main readout controller and monitoring device for four different detectors. The back planes (printed circuit boards) to mount the front end cards for different partitions and detectors vary in their length. This difference in length may lead to different propagation delays for the serial data lines of the FC bus. The clock generated for the FC bus is configurable for its frequency and the start delay of a serial line as listed in table 4.2 and shown in figure

4.4. The *SCLK*, *SDA_out* and *SDA_in* are the adopted acronyms for the serial clock and data lines from the I²C standard [14].



(a) Write transaction between the RCU and the board controller on the FC bus



(b) Read transaction between the RCU and the board controller on the FC bus

Figure 4.4: Communication between the RCU and the FEC board controller on the FC bus

Settings	Frequency SCL	start delay
00	5 MHz	250 nsec
01	2.5 MHz	500 nsec
10	1.25 MHz	1 μ sec
11	0.625 MHz	2 μ sec

Table 4.2: Frequency and start delay selection of the FC bus (for details [23])

This FC bus between the RCU and the FECs provides an alternative path to all board controller configuration and status registers. The RCU is able to read the updated and measured values of voltages, currents and temperature of the FEC from the board controller. The FEE server reads these updated values from all the FECs through the RCU and monitoring status is displayed on the graphical user interface panel [16].

4.1.4 Event readout

The RCU is configurable to perform the event readout in two different modes as explained in section 3.2.6. There are four different commands involved to perform an event readout. Readout of a single event requires execution of two or four different commands to accomplish the readout with a selected mode. The timing characteristics of each command as measured in the test set-up are discussed as under.

- **Scan Event Length (SCEVL)**

This command is issued only once for an event readout when the RCU is configured to run in sparse readout mode (section 3.2.6.2). The SCEVL command is addressed to the board controller on the FEC. It is issued in broadcast mode from the RCU to all front end cards in a readout partition. After issuing it the RCU waits for a configurable time. During this time each board controller reads the channel hit status from all ALTRO chips

on the respective FEC. The total time required to complete this command for the TPC is configured as $90.6 \mu\text{s}$.

- **Event Length Readout (EVLRO)**

This command is issued to each FEC in a partition for an event readout when the RCU is configured to run in the sparse readout mode. The EVLRO command is addressed to the FECs in both branches in a partition at the same time. After being addressed the board controller from every front end card writes four 40-bit words to the hit list memory in the RCU via the ALTRO bus. The total time to complete this command is $0.825 \mu\text{s}$, the measured time for this transaction is shown in figure 4.6 along with all events on the ALTRO control lines during the execution [14]. The execution of this command finishes at the same time in both branches in those partitions (0, 2, 3, 4, 5) where the number of FECs are equal for both branches. In readout partition 1, branch A contains 1 FEC more than branch B so the execution of this branch takes an additional time of $0.825 \mu\text{s}$.

- **Channel Readout (CHRO)**

This command is executed by the RCU for each channel of the ALTRO chip to receive event data regardless of the readout mode of the RCU. It is executed on both branches concurrently to address the channels on branch A and B. Details of the ALTRO control signals and the ALTRO bus protocol are discussed in section 3.3.2.1, figure 4.5 shows the timing characteristics of this command, minimum time to complete it is $0.475 \mu\text{s}$ where a single 40-bit word is transferred from a single ALTRO channel to the RCU.

- **Read Pointer Increment (RPINC)**

This command is common for both readout modes, and executed only once per event readout. The RPINC command is issued after every event data is read out from all of the channels. This command is issued in broadcast mode on both branches in a readout partition. It releases a buffer of the data memory in the ALTRO channel, making it available for writing data of a new event. The event buffers in each channel of the ALTRO chip are used and released on a FIFO basis [20]. The total time taken to execute this command is $0.475 \mu\text{s}$.

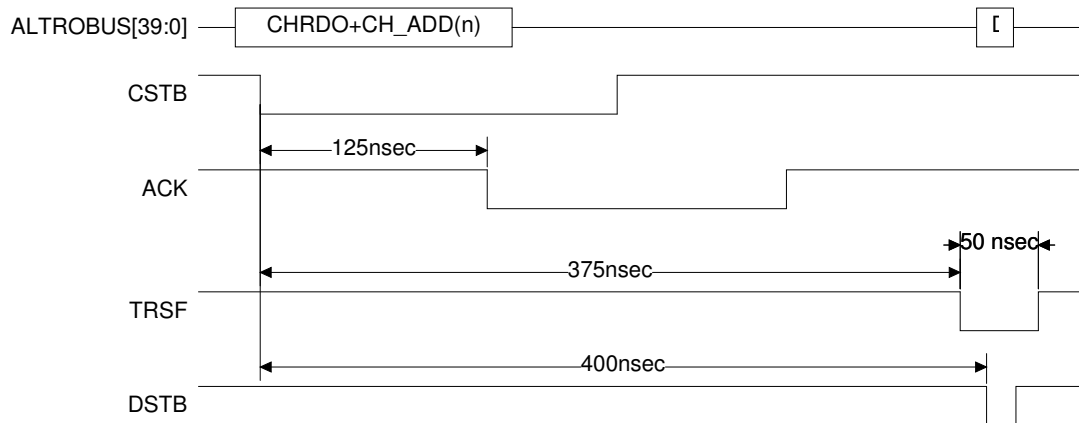


Figure 4.5: Fixed time overhead of the channel readout command

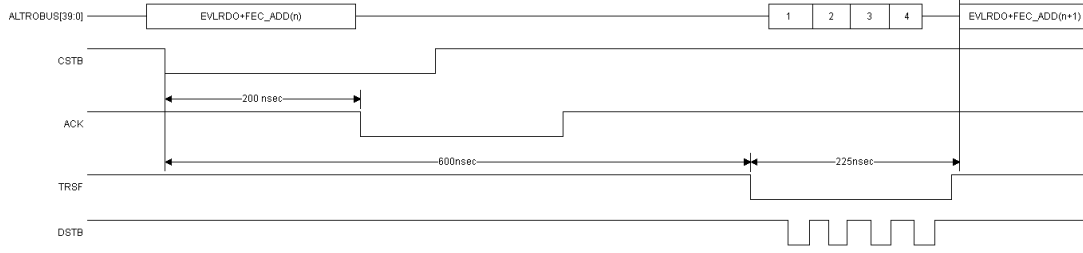


Figure 4.6: Time to execute EVLRDO command to board controller

4.2 Evaluation of readout performance in the test set-up

Event readout in the TPC readout system is performed concurrently in all readout partitions. Readout partitions vary in their maximum readout rate due to the amount of data resulting from particle collisions. The readout list memory for each branch in the RCU is configured with the addresses of all channels to be read out. In the TPC all ALTRO chips are configured for same number of samples to acquire, triggered with the L1 pulse. Since the overall readout rate of the TPC will be limited by the slowest partition, the performance of the readout electronics is assessed by doing a number of different readout tests. All these tests take account of all the variables affecting the readout rate in a single partition.

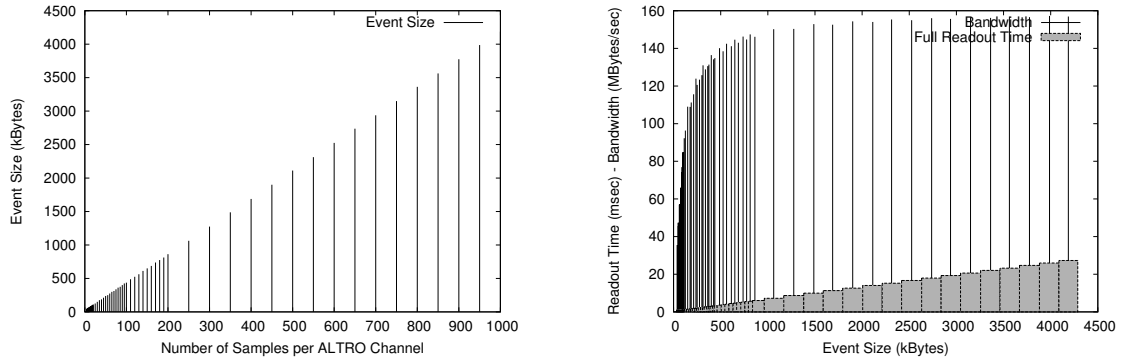
4.2.1 Test methodology

Readout times of an event for both readout modes are measured in the laboratory set-up shown in figure 3.18, where a single partition is used with a maximum number of FECs as described in section 3.5.2.1. These measurements are used to benchmark the performance limits of a single partition by testing for full range of readout parameters. All the tests conducted in the laboratory have been repeated at least five times to check the repeatability. Results from these tests are verified against simulations for event size and readout time. The RCU starts to process for event readout only after a valid L2 trigger sequence is received and it starts to transmit the data to DAQ as soon as the first channel readout is accomplished. An event readout is considered completed when the RCU has shipped off all the event data from all channels followed by the RCU trailer. The RCU signals the completion of an event data packet to the SIU interface unit by asserting a signal named as End Of Event (EOEV). The RCU firmware is implemented with a special counter which measures the duration between the L2 accept pulse and the pulse on the EOEV with a resolution of 25 nsec. The value of this counter is registered and read after every completed event readout to record as readout time for the respective event.

4.2.1.1 Full readout

When event readout is performed in full readout mode, all ALTRO channels configured in the readout list memory are addressed for readout. In the test set-up readout list memories of the RCU for branch A and B are configured to address all channels in the respective branches. As shown in figure 4.7(a) the event size is a direct function of the number of non-zero-suppressed samples acquired for an event readout with corresponding number of samples per channel, the event size and corresponding readout time are listed in table 4.3. In this test full readout is performed for a complete range of number of samples with non-zero-suppressed data. The

total time taken to perform full readout of a partition can be sectioned into three parts as listed in table 4.3.



(a) Number of samples per readout channel and event size (b) Number of samples per readout channel and event readout time

Figure 4.7: Relation of number of samples per channel with readout time and event size

- **Fixed overhead for CHRDO handshake protocol**

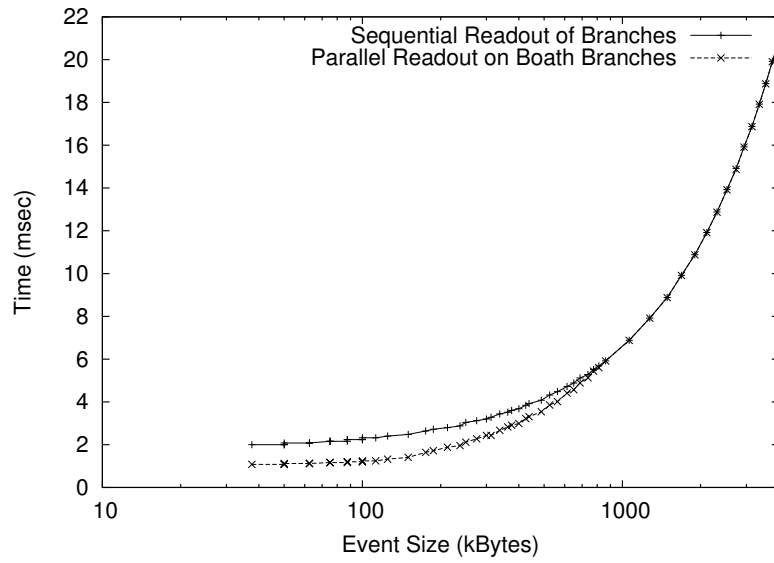
Each ALTRO channel is addressed with CHRDO command to start the acquisition of data from the respective channel, as shown in figure 4.5 it takes 400 nsec before the first valid data word is transferred from the ALTRO chip, in addition to this time another 25 nsec is needed when the ALTRO chip releases the transfer line after the transmission of a data block is completed. The transfer line “TRSF” is passed through a filter protecting it from metastability when it is received on RCU side. The RCU senses the release of the transfer line and asserts the next CHRDO after one clock cycle. The maximum number of channels in one branch is 1664 for partition 1 branch A which contains 13 FECs. The overhead of CHRDO command for a single event readout is 665.6 μ sec. This duration is a part of the time measured as “FEC-RCU” in table 4.3.

- **Data transfer time to RCU from FEC (ALTRO channel)**

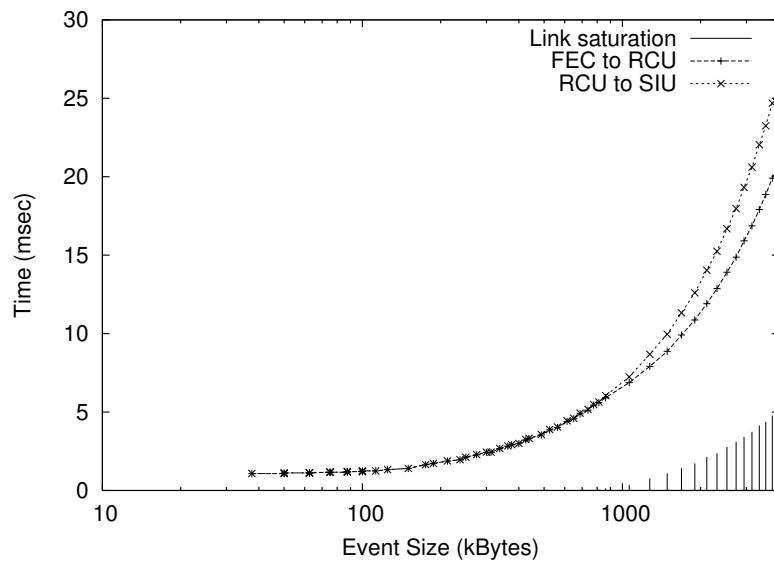
In table 4.3, column [BR-A Busy] and [BR-B Busy] list the time taken to transfer the data from all channels in both branches to the data memories in RCU. Readout is carried out concurrently on both branches, [FEC-RCU] column shows the total time of an event from FEC to RCU. This time includes the fixed overhead of CHRDO command as well. The advantage of carrying out the channel readout concurrently on both branches is evident by comparing the sum of busy times for both branches and the time [FEC-RCU] as shown in figure 4.8(a). The time to transfer the data from FEC to RCU data memories is precisely a linear function of the event. Figure 4.8 shows the results along with the bandwidth utilized for the event readout. Data transfer time for branch A in partition 1 is more than branch B due to one additional FEC, whereas the partitions with a equal number of FECs in both branches are measured for having equal data transfer times.

- **Data transfer time from RCU to DDL**

The channel readout unit (section 3.3.2.2) in the RCU firmware contains two memory banks for each branch in the partition, named as A0, A1, B0 and B1. These memories are read by the data assembler (section 3.3.7) in the RCU firmware to pack and send the



(a) Parallel vs sequential readout of both branches



(b) Measured components of readout time

Figure 4.8: Measurements of full readout in test set-up for the partition with maximum number of channels

event data to the SIU interface unit. The SIU ships this data on the DDL. The channel readout unit, the data assembler and the SIU interface unit operate on the same clock frequency (40MHz). When each ALTRO channel is sending data with an average of 200 samples making a total event size of 750 kbytes then data assembler and SIU start to introduce a dead-time in shipping the data to the DDL. This dead-time in readout

Nsamp (10-bit words)	Event Size (Bytes)	Total Time (msec)	BR-A-Busy (msec)	BR-B-Busy (msec)	FEC-RCU (msec)	RCU-SIU (msec)
100	448068	3.32	2.03	1.88	3.30	3.32
200	883268	6.04	3.07	2.84	5.91	6.04
400	1728068	11.32	4.76	5.15	9.91	11.32
600	2585668	16.68	7.23	6.68	13.91	16.68
800	3443268	22.04	8.60	9.31	17.90	22.04
1000	4288068	27.32	11.39	10.52	21.90	27.32

Table 4.3: Full readout time with number of samples and event size (for complete table see appendix B, table B.2)

system does not appear when the event size from a readout partition is smaller than 750 kBytes, because the fixed dead-time in executing CHRDO provides additional time to the data assembler to transmit the data to DDL. As shown in figure 4.8(b) when the event size tends to exceed 750 kBytes this dead-time imposed by the link saturation is dominant to limit the maximum readout rate.

4.2.1.2 Sparse readout

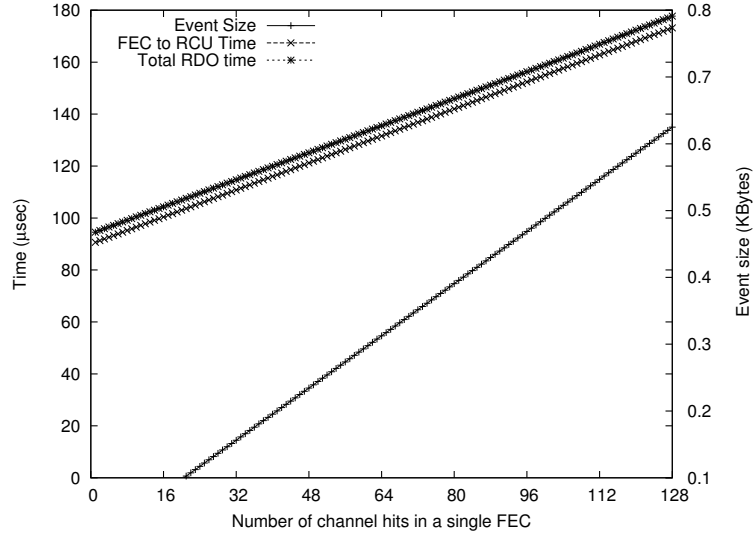
The functional description of the sparse readout is explained in section 3.2.6.2. Sparse readout of an event is completed in two different phases by the RCU. Same as full readout the RCU initializes an event readout after a trigger sequence with L2 accept is received from CTP. The RCU issues the SCEVL command to all board controllers to read the ADEVL [14] register from all of the ALTRO chips on the respective FEC. This register exhibits the hit status of each channel in ALTRO. After the board controller has finished the scanning all of the ALTRO chips for channel status, the EVLRDO command is issued from RCU to each FEC in the partition. Both of these commands are executed concurrently on both branches in a partition, measured times for these commands are listed in table 4.4. After the hit list memory is updated by the board controllers the RCU starts the next phase of sparse readout, which is similar to full readout in terms of CHRDO command. Channel address is fetched from the readout list memory and it is checked for its status in the hit list memory. If the status is '1' (showing a hit) then channel is addressed to acquire the event data. If the channel does not show a hit in its status then the next channel address is read from the readout list memory and this process continues till the end of memory or a channel is found with a hit. The procedure to read a channel address from the readout list memory and checking its status takes 25 nsec for each channel. The readout list memory for branch A in partition 1 contains a maximum number of 1664 channels. The minimum time to scan the readout list memory for any empty event is 41.6 μ sec. This time is calculated, simulated and measured in the laboratory set-up. The readout time for sparse readout is not only a function of the channels configured in the readout list memory and number of samples but also of the channel hits. Two different events with same amount of data may have different readout times. Same is true when two different event show same readout time with different amount of data for each event. The situation described earlier is due to the readout time dependency on the number of channel hits and the event size.

- **Basic characterization with single partition**

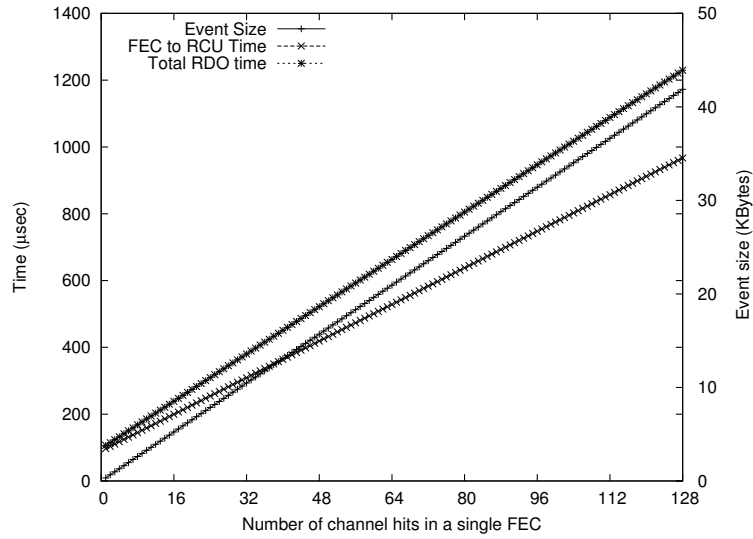
Characterization of sparse readout is performed by two different tests in the test set-up. In the first test sparse readout for a single FEC (128 channels) is evaluated to get the

Transaction	Execution	Time
SCEVL	Broadcast	90.6 μ sec
EVL RDO	FECs:Parallel on Branch A-B	11 μ sec
CHRDO	Channel:Paraller on Branch A-B	425 nsec+(acquisition time)
RPINC	Broadcast	475 nsec

Table 4.4: Measured timing of four steps in sparse readout



(a) Number of channel hits and SRDO time for 10 samples per channel



(b) Number of channel hits and SRDO time for 1000 samples per channel

Figure 4.9: Time profile of Sparse readout (SRDO) for a single FEC (128 channels)

complete time profile with respect to number of channels hits and variation in data size. In the second test, a complete readout partition with 3200 channels distributed over two branches is evaluated for its time profile. For the first test, pedestal memories of all channels in a single front end card are configured with different patterns. Zero-suppression threshold is varied to get a change in number of channel hits incrementing by one for every event. The readout time and the event size is recorded for the detailed analysis.

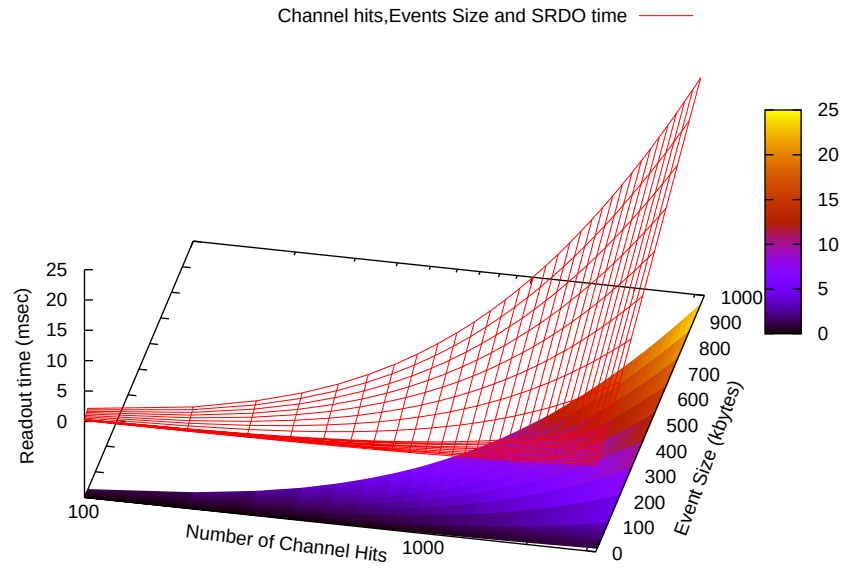


Figure 4.10: Channel hits, event size and readout time

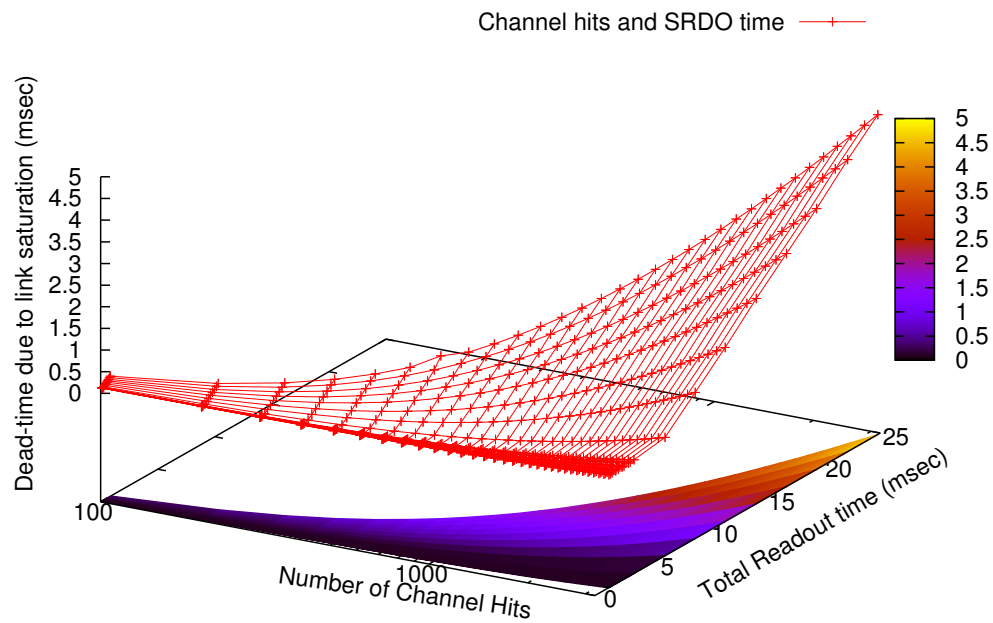


Figure 4.11: Channel hits, dead-time and events size

The total readout time consists of FEC to RCU time and the time taken by RCU to ship this data to DDL (SIU). The efficiency of sparse readout lies in skipping empty channels to avoid the fixed overhead of the ALTRO CHRDO command ($0.475 \mu s$). As discussed earlier scanning of readout list memory and channel readout is carried out concurrently. The time needed to skip empty channels can be formulated as $(\text{No of channels} \times 25 \text{ nsec} + 25 \text{ nsec})$. Figures 4.9(b) and 4.9(a) show the time profile for the readout of channels including fixed overhead of the SCEVL and EVLRDO commands to build the hit list memory. The EVLRDO is addressed only to a single FEC, all others FECs are skipped without any extra overhead in time. All channels were configured to send 10 (10-bit) samples, to emulate events with lower data volume. The time profile of readout time in figure 4.9(a) shows that the time from a FEC to RCU and from RCU to SIU does not add any extra overhead in the readout time for smaller events. The difference due to the ALICE CDH and the RCU trailer (section 3.3.7) stayed the same for complete range of channel hits. Same test is repeated where ALTRO channels are configured to send the maximum event size (1000 samples). Time profile shown in figure 4.9(a) represents the case where the greater number of consecutive channel hits start to effect the difference in FEC to RCU and total readout time (16-32 channel hits). Since each channel hit contributes a fix overhead ($0.475 \mu s$), so this difference increase linearly with the increase in number of channels with a hit. When the number of consecutive channel hits within one branch increases, the advantage of scanning of hit status in parallel also start to get diluted because the scanning process can not continue more than two channels. It needs to wait till the respective ALTRO bus and data memory become available after finishing with the previous channel found with a status of having a hit.

The event readout is performed concurrently on two branches in a partition, so after establishing the time profile of channels hits for a single FEC with two sets of numbers of samples the test is repeated for complete partition with a wider range of number of samples and channel hits. It is repeated for the following number of samples [10,20,.....100,200,...900] from each channel. For each event size the number of channel hits are varied in a step of 100 channels per event, starting from 100 to 3200 channels in the partition. It is important to mention that different events with the same event size may have different readout times in sparse readout due to difference in number of channel hits. Figure 4.10 presents a plot where the readout times for different events is shown for the range of event sizes in test. The plot for total sparse readout time in figure 4.10 shows that there are a number of events with same sizes but different readout times. When these events are checked against the number of channel hits it confirms that the difference is due to the number of channel hits between both events. A total of 576 event readouts were carried out to measure the total readout time and the time from FEC to RCU for the event data. Complete time profile with increasing number of channel hits resulting in increasing amount of event data for full readout partition is presented in figure 4.11. The difference between the total readout time for each event and the time taken for event data transfer from the FEC to the RCU is plotted on z-axis in figure 4.11. Number of channel hits are presented on x-axis on log scale along with the total readout time scaled on y-axis. There are two different phenomena affecting the total readout time; 1) when a higher number of channels are hit with the data, the fixed overhead of channel readout (425 nsec) tends to introduce latency and diluting the effect of skipped channels, 2) this aspect which has been discussed earlier is related to the link saturation for the larger data volumes exceeding 750 kbytes from one single readout partition. These two sources make the difference of total readout time and the FEC to the RCU time to

increase for the events with large data size as shown in figure 4.11. This effect of link saturation has been observed for full readout mode as well and is shown in figure 4.8(b).

- **Evaluation with real events for all partitions in test set-up**

This test is carried out for the evaluation of sparse readout mode emulating all six read-

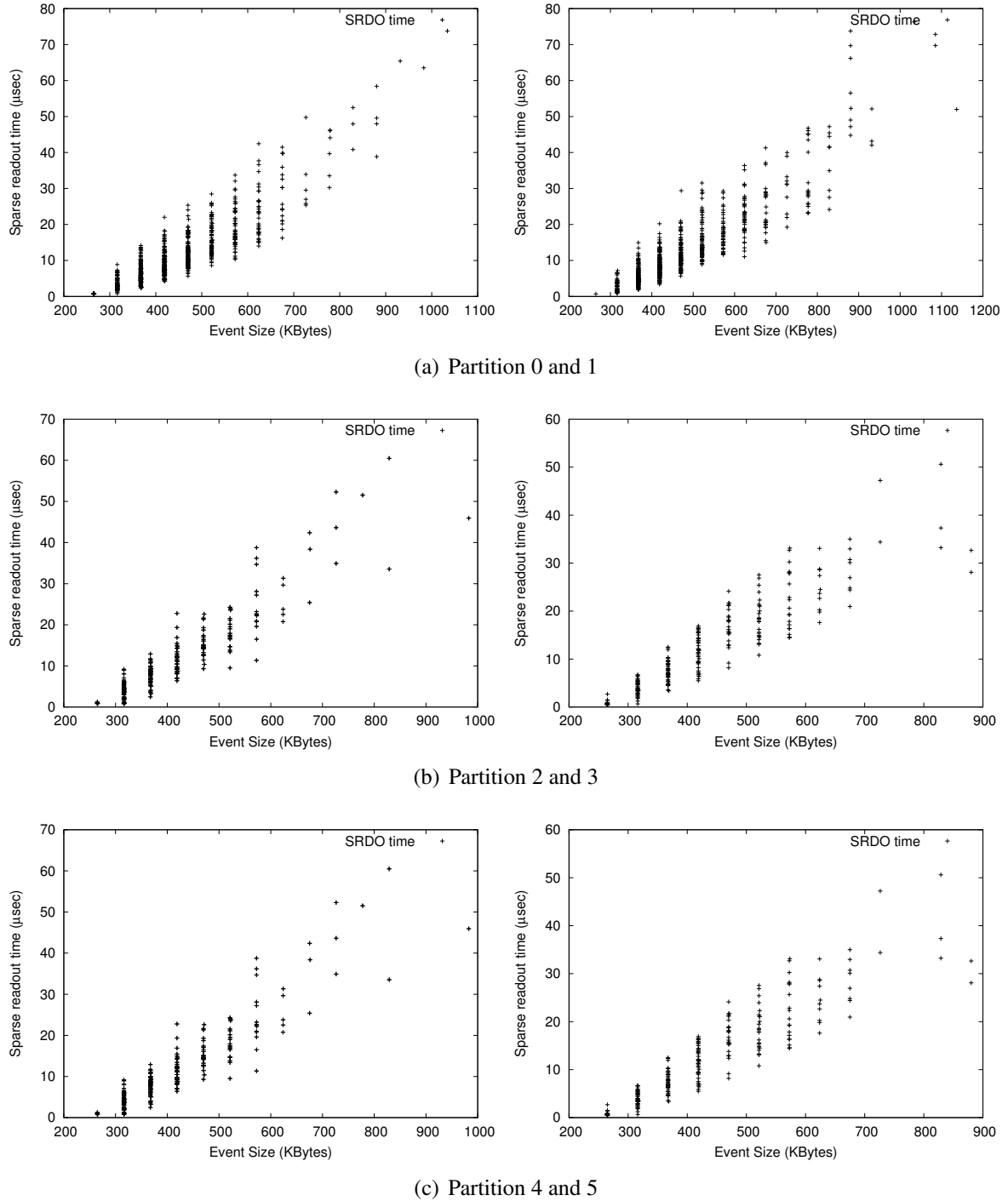


Figure 4.12: Measured readout times for all 6 readout partitions for random event size with sparse readout mode

out partitions of the TPC by replaying real events in the test set-up. The objective of this test is to confirm the trend of readout time for random number of channels hits and

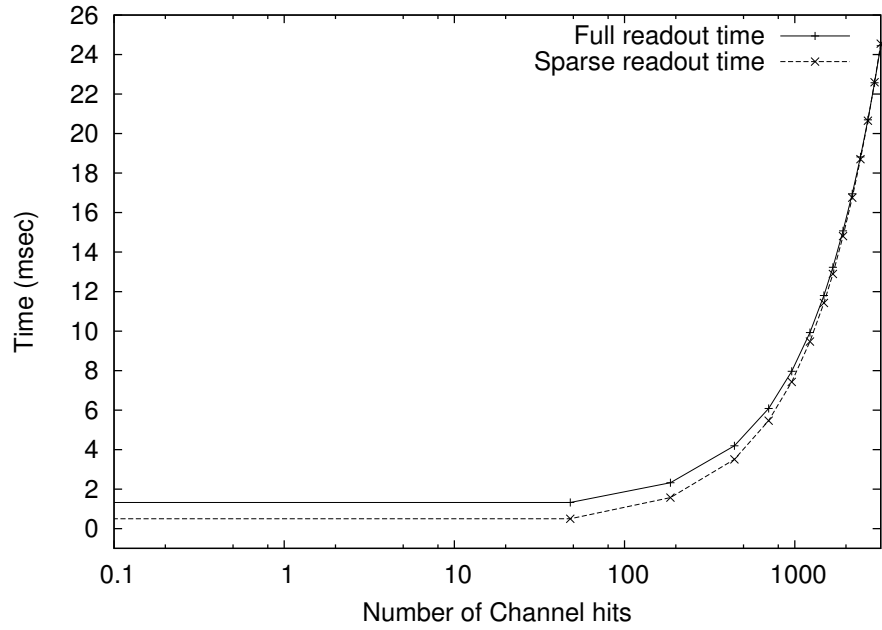
random event size for all different partitions. A total of 2300 real events were selected for replay by the sparse readout in single partition set-up. This partition has a maximum number of FECs so the emulation of all possible combinations of the FECs was performed to test the readout time for each partition. Measured readout times for various event sizes from each readout partition are shown in figure 4.12. Event size varies due to random number of channel hits and due to varying amount of zero-suppressed event data from each channel. Each channel sends a different amount of data for every event due to the settings of fixed pedestal and zero-suppression threshold. Events with the same event size are shown to have different readout times in all readout partitions, for example in figure 4.12(a) for readout partition 1, events with sizes of 1Kbytes to 9Kbytes were measured for having same readout times. A similar trend is visible for all the partitions plotted in figure 4.12. Event data is checked to confirm the number of channel hits and the payload size of events with same readout times to confirm that results of real events follow the time profile established by characterization.

4.2.1.3 Comparison of full and sparse readout modes

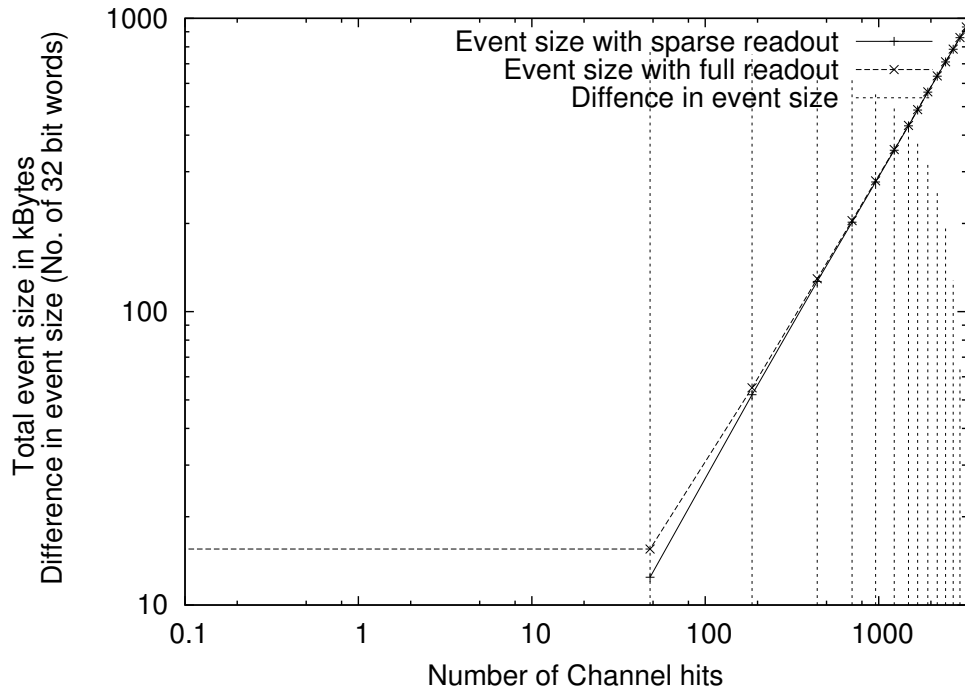
Sparse readout and full readout modes follow different schemes in addressing the channels for readout (section 3.2.6). This basic difference of addressing all channels in a partition and addressing only those with data makes sparse readout highly efficient for the events with lower multiplicity as compared to full readout. This efficiency of sparse readout is gained because of gathering the hit information of all channels in prior to performing the channel readout. Sparse readout gains advantage in total readout time by skipping an empty channel in 25 nsec, whereas full readout takes 450 nsec. This advantage of sparse readout starts to get diluted when the number of channels hits per FEC starts to increase. For such events where most of channels show a hit, sparse readout gets slower because of its overhead of the SCEVL and the EVLRDO commands. Empty channels do not send any data for the sparse readout whereas for the full readout empty channels send one 32 bit word to the DDL. Lower amount of data for sparse readout contributes an advantage in the bandwidth utilization of the DDL link. Both readout modes were tested to compare their time and event size profile with respect to the increasing number of channel hits. Figures 4.13(a) and 4.13(b) show that at the higher number of channels hits for both modes leads to have same readout time and payload size. When all channels have event data to transfer to RCU, sparse readout mode will be slower by at least $101.6 \mu\text{sec}$ due the overhead of building the hit list memory.

4.2.2 Test of multi-event buffering and readout

The multi-event buffering scheme refers to the feature of ALTRO chip to buffer the data of more than one event on the chip. Each ALTRO channel is equipped with a data memory (1024×40 bits) to buffer the event data, which can be partitioned in either 4 or 8 buffers. Acquisition of an event begins in ALTRO upon arrival of L1 trigger, the event data is stored in the data memory [8]. The data is validated when a L2 trigger is received by the ALTRO and it is kept till the CHRDO command is received from the RCU as described in section 3.2.6. If another L1 trigger arrives while transmitting the event data of the previous event, the data transmission from the ALTRO to RCU is paused and the next event is stored in the next available data buffer. The busy box can be configured to allow four consecutive triggers before asserting a busy signal to the CTP [18] as shown in figure 4.14. Storage of multi-events on the ALTRO chip is essential to reduce the deadtime introduced by the readout electronics.



(a) Trend of readout time and channel hits



(b) Trend of event size and channel hits

Figure 4.13: Comparison of full and sparse readout for increasing number of channel hits

Reducing deadtime of the detector does not mean to have a faster readout but it will reduce the time when the detector is not available to accept and process new triggers.

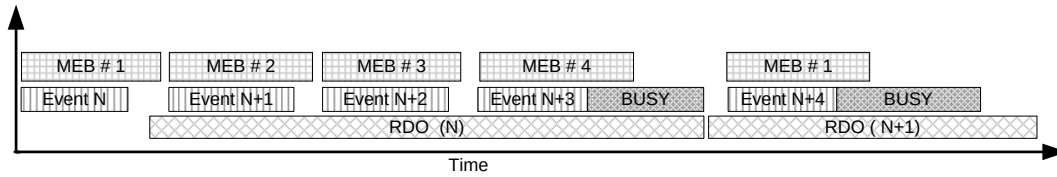


Figure 4.14: The principle of multi-event buffering with four buffers and event readout

4.2.2.1 Objectives of the test

Multi-event buffering and readout at a randomly varying input trigger rate is a complex and challenging task. Any kind of switching activity on the readout clock during the acquisition of a new event in parallel to an event readout may introduce systematic errors in the acquired data. The communication between the busy box and the DAQ to verify the event IDs and handling of multiple trigger sequences during a readout by the RCU require extensive verification. Special tests were conducted to verify the following objectives:

- **Multi-event handling and storage in ALTRO**

First key objective of this test is to verify that all ALTRO chips mounted on the TPC are able to perform acquisition of new event data during readout of previous event without any kind of corruption of in new acquisition. After complete acquisition ALTRO must resume the readout of the previous event, these two key functions were verified in this test.

- **Verification of readout control in RCU**

RCU keeps track of number of triggers received from the CTP and completed event readouts sent to the DAQ. When a new trigger sequence is received during a readout, RCU needs to pause the data transmission to DAQ and resumes as soon as new data is received from the ALTRO. The RCU is required to transmit the event data from the channels according to the list of channels in the readout list memory. The control of data flow at high trigger rates with random spacing between the events is a challenging task. This characteristics of RCU firmware is tested as third key objective of the tests conducted at random trigger rate.

- **Handling of multi-event in complete readout system**

Readout system is configured to run with 4 multi-event buffers in busy box configuration. When the system is operated with single buffer, busy is asserted as soon as the first trigger is received, whereas when the busy box is configured to allow four consecutive triggers then busy is asserted only if none of the 4 consecutive events are not received by the DAQ.

4.2.2.2 Test scheme and hardware set-up

The test was conducted on the TPC, involving the calibration pulser system (section 2.6) as the source of the signal to be acquired by the front end electronics of the TPC.

- **Test scheme**

The calibration pulser firmware is identical to the RCU firmware in receiving and processing the trigger sequences from the CTP. Two key operations are synchronized in order to perform the test; 1) the start of data acquisition in ALTRO upon the L1 trigger

2) a voltage pulse with predefined delay is issued from calibration pulser to the cathode wire plane of the readout chambers. The delay with respect to L1 is known so the pulse which appears in the event data will present the shift in time with respect to the start of acquisition. The calibration pulser issues the pulse with shifted time which is a multiple of $100\ \mu\text{s}$ for every new trigger received from CTP. This process is repeated for 7 times, shifting each pulse in time and repeated over 5,000 events.

- **Modification of the calibration pulser firmware**

The calibration pulser firmware was modified to implement the test scheme described above. A new module named as pulser sequencer is implemented with sequence memory of (1024×32) size. The sequencer memory is loaded with predefined sequences

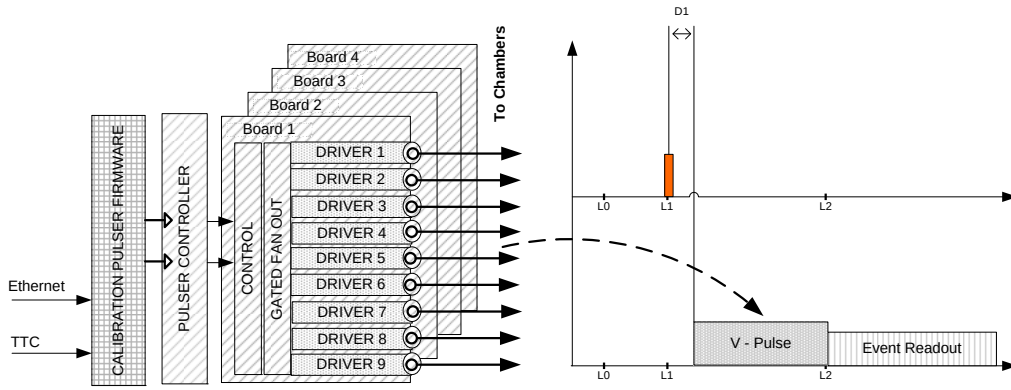


Figure 4.15: Calibration pulser to test multi-event buffering in the TPC

to execute on arrival of the L2 trigger. The pulser sequencer executes pre-configured sequences to vary the parameters of pulse (coarse delay [25]). The coarse delay is measured relative to the arrival of the L1 trigger pulse till the pulse from the pulser controller. Each sequence varies the coarse delay of pulse as shown in figure 4.16. The first

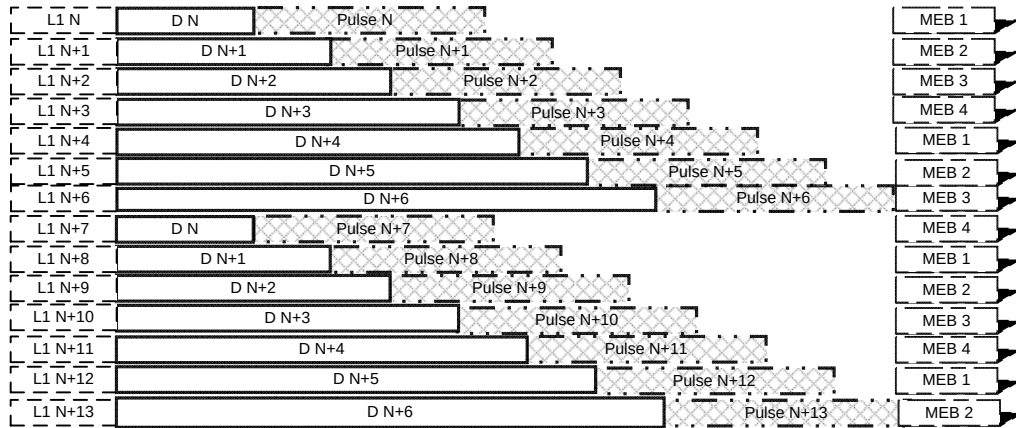


Figure 4.16: Storage of time shifted pulses in MEB according to test scheme

sequence in the sequence memory is executed at SOR trigger sequence. The parameters set by this sequence will change the default settings of the pulser controller, changes are applied to the first event of data acquisition, this first event is stored in MEB1 as shown in figure 4.16. This procedure is followed so that each buffer in ALTRO chips is filled with a pulse with shifted time from the reference point (start of acquisition) as shown in figure 4.16.

- **Busy box and LTU**

The busy box is configured to allow four triggers to be delivered to the RCU before it receives the event IDs of respective events. In such a setting the busy box will assert the busy to LTU only when none of the event IDs of four triggers is received from the DAQ. Triggers from LTU were issued with random time space among them at varying rate. The trigger rate was tuned so that all four buffers in ALTRO were used to store the test pulses. It was needed to tune to avoid such situations where the trigger rate from LTU is too high, then ALTRO chip may use only the first buffer to store new events. Two registers in busy box were continuously monitored to confirm that the number of events being handled vary with respect to the input trigger rate. These tests were repeated five times with recording of 5,000 events for each test.

4.2.2.3 Results and analysis

The readout system was set to run at varying trigger rates with random spacing in time among the trigger sequences for more than one hour of continuous operation. The input trigger rate was varied from 100Hz to 100kHz, all status registers of RCU and busy box were kept under observation to check any kind of erroneous situation in any sub-system. All the triggers issued and processed to perform event readout were consistent indicating that the event and data flow control is robust and ready to handle and process the multiple events.

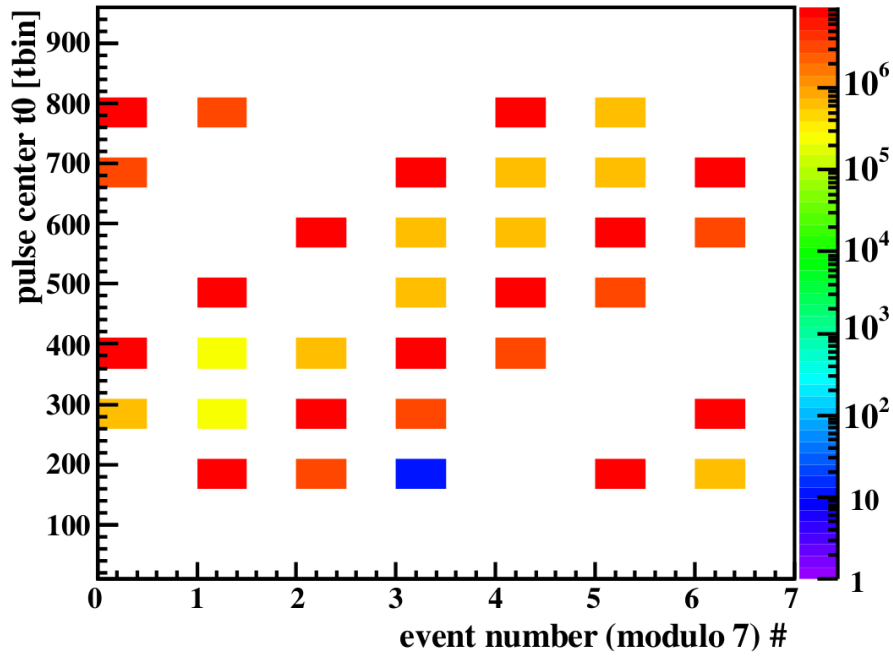
The recorded data under manually tuned trigger rate was reconstructed and analyzed to check the quality and consistency of the data for the shifted pulse. In the first iteration data was found corrupted and misaligned in terms of the shifted time as shown in figure 4.17(a) [29]. The Y-axis in figure 4.17(a) shows the timebin (10 timebins = 1 μ s) when the test pulse is found in the event data, the X-axis of figure 4.17(a) represents the event data found within the acquisition window. The pulses were issued with shifted time as multiple of 100 μ s for every new trigger, this shift in time is labeled as “D” in the test scheme shown in figure 4.16. The randomly shifted data in acquisition window along Y-axis and wrong distribution of the amount of events in X-axis shows that the data was corrupted at some stage during the test.

The problem in the readout system was debugged and the reason for the corrupted data was found. Data was corrupted due to the overlap situation between execution of the sequence to shift the pulse by the calibration pulser and ALTRO acquisition for next event. Readout tests were repeated with modified test scheme providing enough time (30 μ s) to change the position of new pulse from the calibration pulser. The data from the repeated test was decoded again, the mechanism to handle multi-event and their storage in the buffers was verified as presented in figure 4.17. The test pulse issued from the calibration pulser with known shift in time was found as expected under similar settings of the busy box and input trigger rate from the LTU.

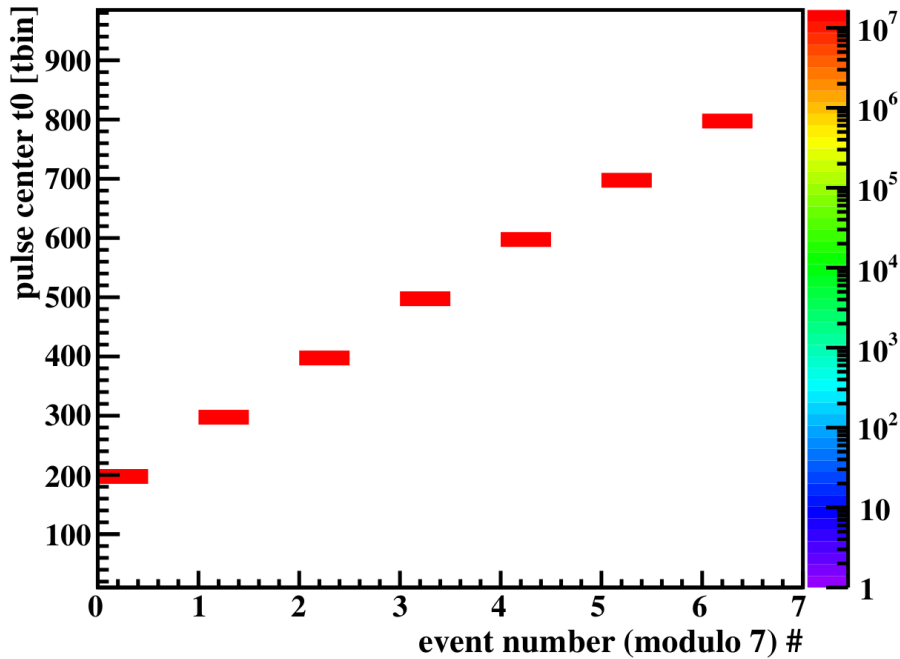
4.2.2.4 Evaluation of performance with multi-event buffering

After the mechanism of multi-event buffering and control of data flow was verified, the effect of the multi-event buffering on readout rate was evaluated. This evaluation was performed using the same test set-up. In this test a standard single pulse with fixed latency was applied to all readout channels using the calibration pulser. The busy box settings and configuration of the RCU was modified to evaluate the readout rate with different settings of multi-event buffer (1,2,4).

Sparse readout mode faced a hardware limitation at front end card level in using 4 multi-event buffers. The ALTRO control lines to deliver the L1 trigger to ALTRO are not available during the scan event length execution, because during this time the board controller disables the



(a) Corrupted and misaligned data due to erroneous test procedure



(b) Decoded data with shifted pulse and predefined time

Figure 4.17: Test results of MEB functionality

GTL drivers to ALTRO chips from the RCU. To overcome this short coming a special mode is implemented in the busy box, this mechanism enables the readout system to run with 2 event buffers.

The relation between the event readout rate and the input trigger rates under different settings of multi-event buffers is shown in figure 4.18. The readout rate for all tested mode falls approximately within the same range (100Hz-150Hz) where the input trigger rate is at the order of 200Hz. When the input trigger rate increase more than 500Hz, the effect of multi-event buffer is distinguishable in the figure 4.18. When the readout system is running with the single

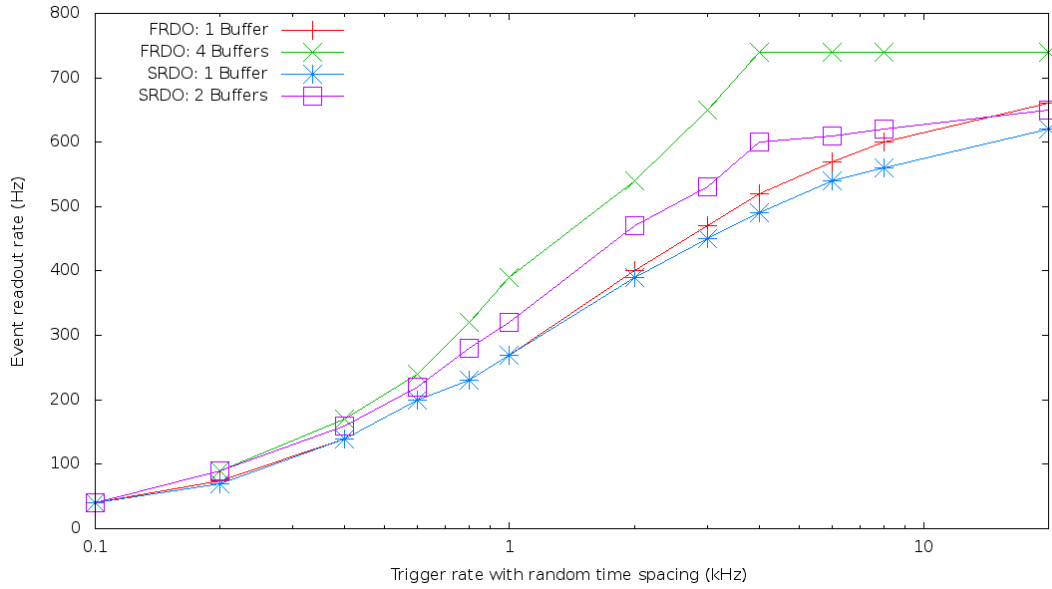


Figure 4.18: Effect of different setting of MEB on readout rate

buffer it starts to reject the input trigger significantly at input trigger rate of 1 kHz and above. During this time the system is busy in performing either the acquisition or the readout of an event. When the system is configured to use 4 multi-event buffers then the TPC readout system starts to exhibit a lower deadtime and higher event readout rate by rejecting less number of triggers at higher trigger rates.

4.3 Performance of the TPC readout at the ALICE experiment

TPC detector has been installed into the ALICE experiment and since 2007 up to the start of the LHC it has been commissioned with triggers on cosmic particles. The LHC started its operation in the fall of 2009. Since then the TPC has recorded event data for more than one billion of proton-proton (p-p) collisions at center of mass energy of 7 TeV. In November 2010 the TPC recorded more than 50 million Pb-Pb events during three week of LHC heavy ion (Pb-Pb) operation at energy level of 2.76 TeV per nucleon pair.

Beam Type	Beam Energy	Number of events
p-p	900 GeV	30,000
p-p	2.76 TeV	15,000
p-p	7 TeV	30,000
Pb-Pb	2.76A TeV	5,000

Table 4.5: Event data used in performance evaluation

- **Tests for evaluation of performance**

The performance of readout electronics for the complete TPC is evaluated by replaying event readout with the data recorded during different periods of LHC operation. A data set is selected based on different energy levels of LHC operation for p-p and Pb-Pb collisions as listed in table 4.5. Tests to evaluate the performance were carried out with all the events data mentioned in table 4.5 but the results are analyzed and presented for

the maximum energy levels of 7 TeV for p-p and 2.76A TeV for Pb-Pb collisions. ALTRO chips can be configured to send the contents of pedestal memory as event data. This special mode of the ALTRO operation is utilized to process the event data through the complete readout chain to measure the readout time of each event. All RCUs in 216 partitions were configured through normal configuration process from the DCS. The test procedure consisting of the following five steps is repeated for each event.

1. Writing real event data to pedestal memories of all ALTRO chips
2. Configuring RCU to perform sparse readout
3. Trigger and record the sparse readout time
4. Configuring RCU to perform full readout
5. Trigger and record the full readout time

4.3.1 Readout performance during p-p collisions

Over 15,000 events of p-p collisions are replayed according to the procedure described above to record the readout times of both modes for each event. Data acquisition of the TPC was carried out with single buffer, and the RCUs were configured to conduct sparse readout during p-p operation of the LHC. There are 36 partitions of each type (0,1,2,3,4,5) in the complete TPC readout system. TPC readout rate can not be faster than the slowest partition, so readout times of all partitions are analyzed to check the slowest readout partition. Results are evaluated for each set of partitions to compare among the partition with similar number of front end cards.

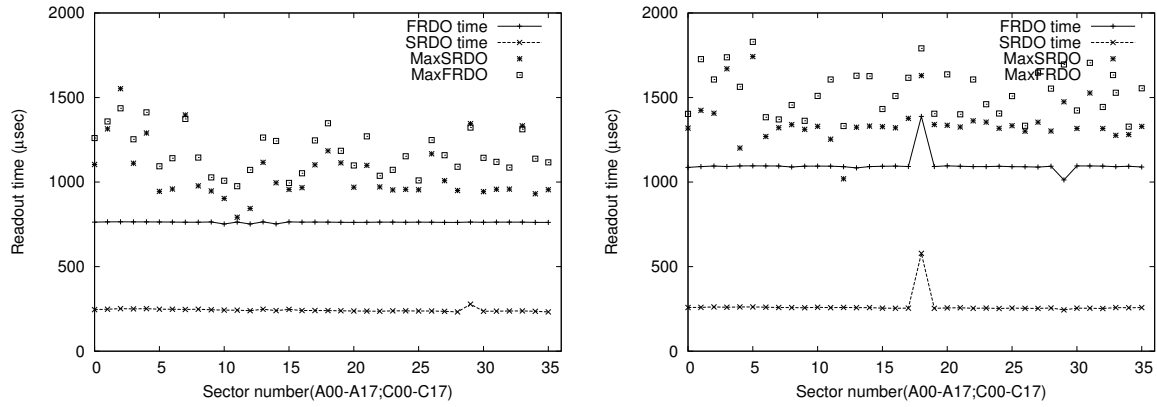
4.3.1.1 Readout partitions 0 and 2

Readout Partition 0 (RP-0) is the inner most partition, first one in the radial direction in every sector of the TPC. It contains 9 front end cards on each branch, constituting a total of 2304 readout channels. There are 36 partitions of this type in the TPC, 18 on each A and C side. Readout times of these 36 partitions were compared to evaluate their behavior for each event. Average full and sparse readout times for these partitions are plotted in figure 4.19(a). These partitions differ with negligible (200-300 nsec) amount of average time for sparse and full readout modes, whereas the maximum time for each partition for the slowest event vary significantly. These events are taken as exceptions which are the result of random events causing these partitions to send more data. Sparse readout mode is evaluated to be efficient by an average factor of 2.65 as compared to the full readout mode for this partition during the p-p collisions.

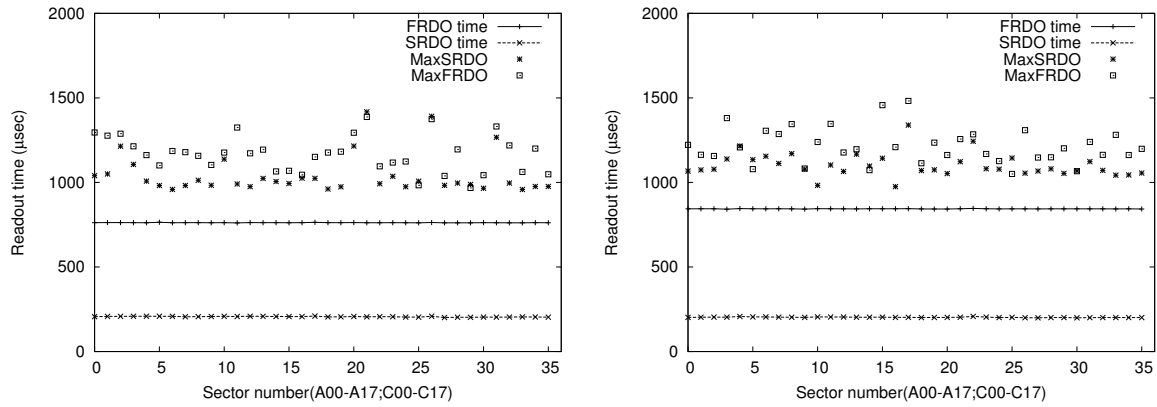
Readout Partition 2 (RP-2) is the third one in radial direction away from the interaction point. It contains similar number of the channels as RP-0. Readout times of these 36 partitions of RP-2 type were compared to evaluate their behavior for each event. The averages of full and sparse readout times for these partitions is plotted in figure 4.19(b). These partitions differ with negligible amount of average times as for the RP-0. Sparse readout mode is evaluated to be as efficient by a average factor of 3.2 as compared to the full readout mode for this partition during the p-p collisions.

RP-0 and RP-2 are compared for readout times because of having same number of channels. Average full readout time of RP-2 is less that RP-0 by 0.16 percent (126 nsec). It is evidently consistent with the characterization made in lab, where readout times for full readout are dominated by the fix overhead of the CHRDO command. Average sparse readout time of RP-2 is

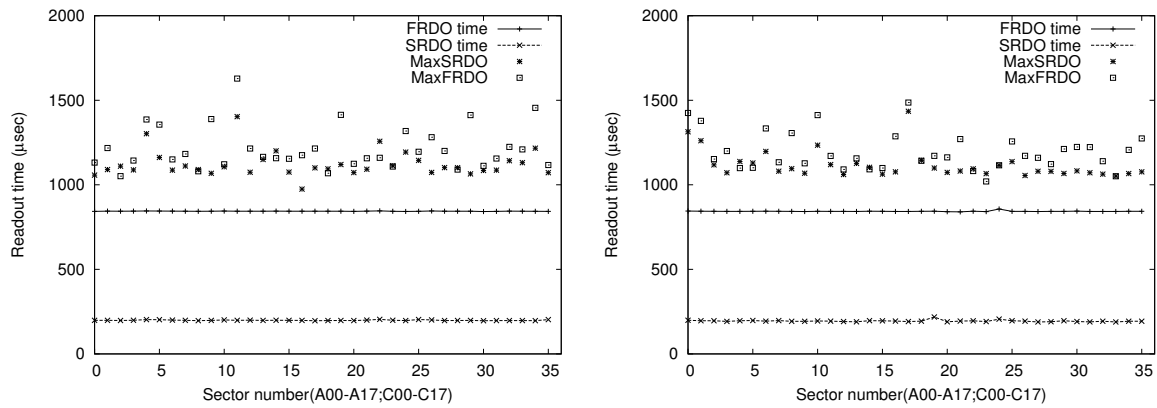
less than RP-0 by 15 percent (27 nsec) where efficiency is gained for sparse readout by less number of channel hits. Average event readout time for both partitions experienced during p-p collision is listed in table 4.6.



(a) Partitions 0 and 1



(b) Partitions 2 and 3



(c) Partitions 4 and 5

Figure 4.19: Average readout times for six different readout partitions during p-p collisions

4.3.1.2 Readout partition 1

Readout Partition 1 (RP-1) is the second inner most partition in every sector of the TPC. It contains 25 front end cards, constituting a total of 3200 readout channels. This readout

partition is unique of its type in terms of number of channels. There are 36 partitions of this category in the TPC. When 36 partition of type RP-1 are compared among each other it is found that RP-1 for the sector C-00 has a higher data volume and readout time for both modes of readout. This difference can be due to the channels with higher levels of noise in this partition, which always shows a hit during sparse readout and sends data of higher volume for the full readout. Event data and the respective readout times were checked to find the reasons for higher readout time, it is found there are around 3000 event with higher readout times which bias the average readout time of this specific partition in the sector C-00.

Sparse readout for the RP-1 is evaluated as faster by 4.65 times in comparison to the full readout, this factor is calculated by averaging over 36 partitions of type RP-1 for more than 15,000 events. Measured readout times for RP-1 are found larger than all other partitions as listed in table 4.6 and shown in figure 4.19 in respective sector as expected due to the higher level of channel occupancy resulting in larger data volume and maximum number of channels hits. The RP-1 type partition is the slowest partition limiting the readout rate of TPC for any operational session (run). Exact position of the partition may vary among different events depending on the physics and collision parameters.

4.3.1.3 Readout partitions 3,4 and 5

Readout partitions 3,4 and 5 are three outer most readout partitions in each sector of the TPC. All of these three partitions contain 10 FECs on each branch comprising total of 2560 readout channels in each partition. Readout times of all 36 partitions of RP-3 are compared against each other and the average difference for sparse mode and full readout mode are calculated as 100 nsec and 350 nsec respectively. The difference is negligible to effect the over all readout performance of the TPC. A similar trend is found for the RP-4 and RP-5 for the readout times of full and sparse readout modes. Readout times of these three partitions are listed in table 4.6. When these three partitions having equal number of channels are compared their behavior reflects not only the characterization but also the impact of their geographical position on the event size. The readout time for the sparse mode decreases from RP-3 to RP-5 reflecting less number of the channel hits due to their distance from the interaction region.

4.3.1.4 Performance summary for p-p collisions

Partition	Radial position (cm)	SRDO (μ s)	FRDO (μ s)	DIFF (μ s)
RP-0	100	207.34	757.42	518.94
RP-1	120	235.71	1097.90	862.18
RP-2	149	179.60	757.30	577.69
RP-3	175	179.29	847.50	668.20
RP-4	203	176.42	839.72	663.29
RP-5	208	171.32	839.20	667.88
TPC AVG (P1+130μs)	—	365.71	1227.9	862.18

Table 4.6: Average Readout time for p-p collisions of all partitions

Readout time measured to evaluate the TCP performance for p-p collisions did not include the drift time of TPC detector and the time taken by the DAQ and busy box communication to verify the validity of an event. This time is approximated in total 130 μ s and added to the measured readout times to calculate the maximum readout rate which was experienced by the

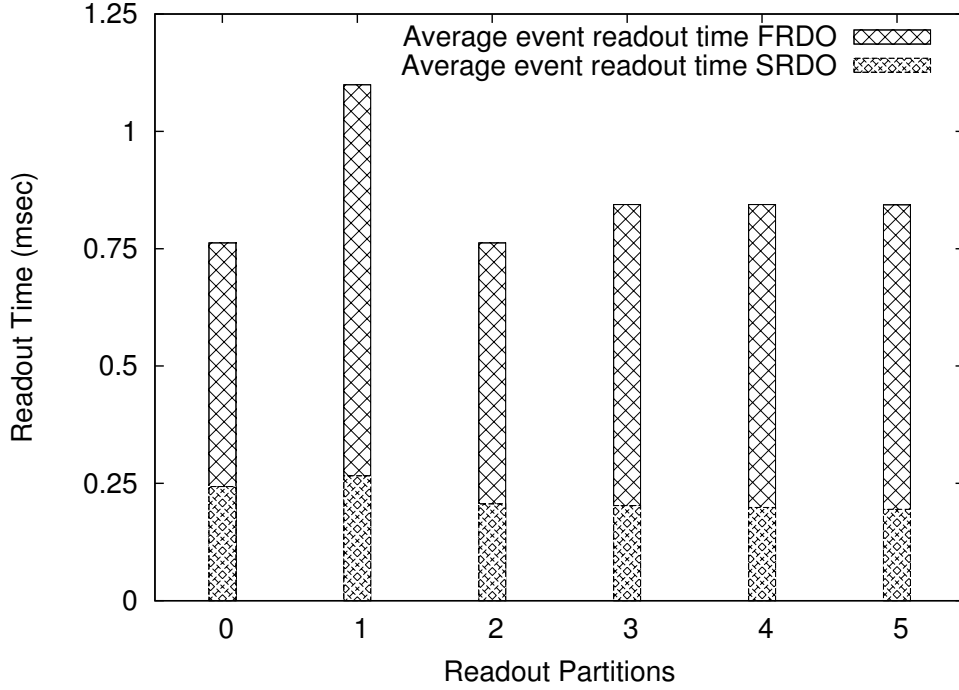


Figure 4.20: Average readout performance of 6 readout partitions during p-p collisions

TPC readout system for the LHC operation with p-p colliding beams. Readout times of both modes for 36 partitions of each type measured over 15,000 events are averaged and listed in table 4.6. The ALICE TPC experienced an average readout rate of 2.7 kHz at 7 TeV, which is limited by the readout partition 1 as listed in table 4.6 and shown in figure 4.20.

4.3.2 Readout performance during Pb-Pb collisions

In November 2010, the LHC changed the beam mode to Pb-Pb collisions at an energy of 2.76A TeV per nucleon pair. Pb-Pb data of 5000 above events was replayed with the same procedure as described for p-p collisions data. In the tests conducted, readout times were measured for both readout modes to evaluate the performance during heavy ion run of LHC in 2010. These tests are repeated for both modes so the evaluation can confirm the optimal mode of readout for Pb-Pb collisions during in 2011. The readout mode during Pb-Pb collisions in 2010 was set to sparse readout with single buffer. These tests evaluated the performance of the TPC readout during 2010, and the conducted tests for full readout facilitate projecting the performance of the TPC readout system for more central Pb-Pb collisions. Evaluation procedure is kept same the as p-p collisions data by analyzing the readout time for each partition in the TPC and averaged over the complete number of events and analyzed with respect to partitions of the same number of FECs.

4.3.2.1 Readout partitions 0 and 2

Basic characteristics of both partitions for their physical location and number of channels has been discussed in section 4.3.1.1. Average readout times for 36 partitions of type RP-0, are compared and shown in figure 4.21(a). Average difference among 36 partitions for full and sparse readout does vary but with a negligible amount of time (200-300 nsec) and this varia-

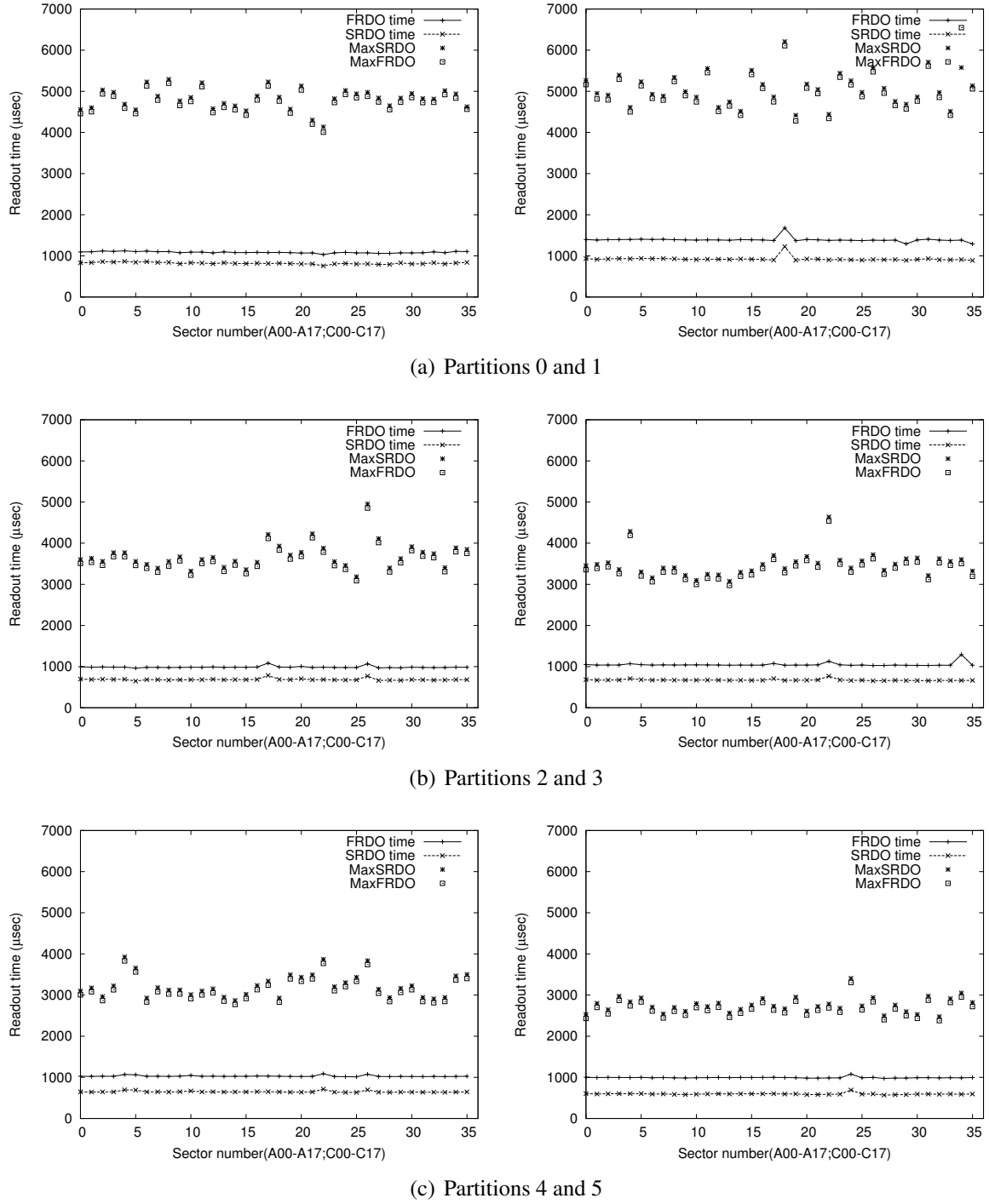


Figure 4.21: Average readout times for six different readout partitions during Pb-Pb collision

tion does not affect the accumulative performance of the complete TPC. Average readout rate calculated for sparse and full readout modes is listed in table 4.7. Sparse readout for RP-0 is found to be faster by 24 percent than the full readout.

36 partitions of type RP-2 are plotted in figure 4.21(b) average readout times among 36 partitions of RP-2 type shows a negligible difference in time for both modes. Sparse readout for RP-2 is found to be more efficient by 30 percent in average. The performance of RP-0 and RP-2 for Pb-Pb collision data follows the trend established in characterization. Readout times for full and sparse modes of RP-0 is higher than RP-2 due to more channel hits and higher volume

of data. Average readout time with sparse readout for RP-2 is less by 16 percent whereas full readout is less by 8 percent than RP-0, this difference is clearly in line with the fundamental difference between full and sparse readout.

4.3.2.2 Readout partition 1

The event readout times of RP-1 are shown for 36 partitions in figure 4.21(a). As expected readout partition 1 encapsulating maximum number of channels exhibits maximum readout time as compared to all other readout partitions. All 36 partitions show the similar readout times except readout partition for sector C00. This partition is found having more data for the majority of the events and number of channels hits as compared to other partitions as discussed in section 4.3.1.2. Average difference between full readout time and sparse readout time for this partition is 467 micro seconds which is 33 percent of average full readout time.

4.3.2.3 Readout partitions 3, 4 and 5

All of these three partitions contains equal number of 2560 readout channels. Readout times for 36 partitions of each type are plotted in figure 4.21(b) and 4.21(c). Readout times of each partition among each type appears to be of negligible difference having no influence to limit the TPC readout rate for Pb-Pb collisions data readout. Average readout times for both modes of these three partitions are listed in table 4.7. It is shown in figures 4.21(b) and 4.21(c) as listed

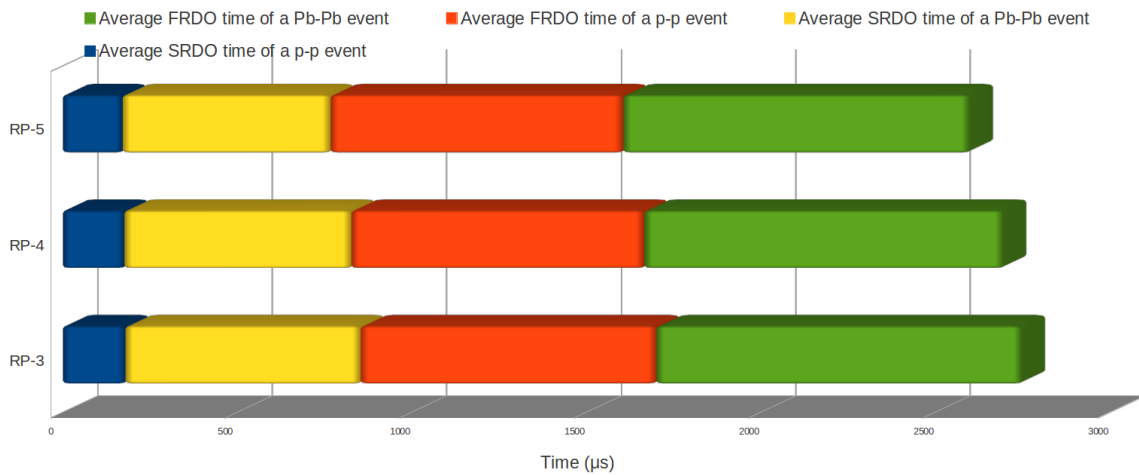


Figure 4.22: Comparison of three partitions with equal number of channels

in table 4.7 that readout times for these three partitions decreases gradually from partition 3 to 5. Although these partition contain similar number of channels but the amount of data and channel hits vary based on their radial position in the sector. All channels are addressed for readout so the decrease in full readout time from RP-3 to RP-5 is around 10 percent, whereas sparse readout times decreases by 13 percent.

4.3.2.4 Performance summary for Pb-Pb collision

Readout time measured to evaluate the TCP performance for Pb-Pb collisions did not include the drift time of TPC detector and the time taken by the DAQ and the busy box communication to verify the validity of event id. Same as p-p collision data this time is approximated as total of 130 micro seconds and added to the measured readout times to calculate the maximum

Partition	Radial position (cm)	SRDO (μ s)	FRDO (μ s)	DIFF (μ s)
P0	100	820.82	1084.95	264.12
P1	120	926.36	1394.09	467.72
P2	149	686.47	988.08	301.61
P3	175	673.98	1048.75	374.76
P4	203	650.28	1029.46	379.17
P5	208	595.65	993.61	397.95
TPC AVG (P1+130μs)	—	1056.36	1524.09	467.72

Table 4.7: Average readout time for Pb-Pb collisions for all partitions

readout rate which was experienced by TPC readout system for LHC operation with heavy ions colliding beams. Readout times for 36 partitions of each type measured over 6,000 events are averaged as listed in table 4.7.

4.4 Summary

The readout time for an event is a function of the number of channels to be addressed to perform readout and the amount of the zero suppressed event data on each channel. The volume

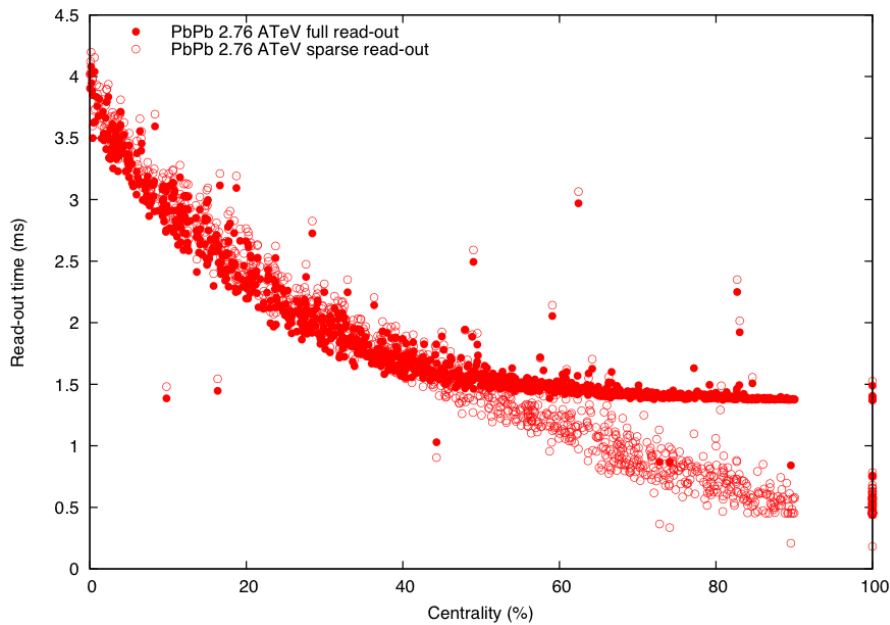


Figure 4.23: Readout time vs centrality of Pb-Pb collision

of data and number of channel hits from a readout partition mainly depends on the radial distance of the partition from the interaction region and on the beam type. The data volume of an event depends on the parameters determining the trigger generation for a given collision as well. Centrality of a heavy ion collision reflects the number of particles generated and it relates to the amount of data per channel and number of channel hits in the TPC. The readout time of an event as a function of the centrality is shown in figure 4.23.

The ALICE TPC readout electronics has experienced an average readout time of 366 μ s and 1056 μ s during LHC operation which corresponds to an average readout rate of 2.7 kHz

and 946 Hz for p-p and Pb-Pb collisions respectively. The tests conducted, the results obtained and the analysis show that the ALICE TPC data acquisition system fulfills all the design requirements and the goals set in the specifications [2].

Chapter 5

Stability and robustness of the ALICE TPC readout system

A fault tolerant and stable operation of the TPC data acquisition system is essential to meet the physics objectives of ALICE, especially since the TPC is the main device for tracking charged particles and particle identification. This chapter describes the fault tolerant techniques, testing methods, error reporting and recovery procedures that were developed in order to ensure a reliable operation. The effects of radiation on the readout electronics play a major role in determining the reliability of the complete readout system. Single Event Upset (SEU) mitigation techniques implemented in different components of the TPC FEE are discussed in the following sections.

5.1 Robustness of the TPC readout electronics

The TPC front-end electronics contains five different PCBs implementing five independent physical interfaces. It consists of 4,356 FECs, steered by 216 assembled RCU boards interfacing them to the online systems of ALICE. The manufacturing variation in the PCBs and in the mounted integrated circuits, temperature and power gradients caused by varying switching activity are key factors which may affect the operational robustness of the TPC FEE. These factors may affect the normal operation by introducing transient glitches or delayed switching of the critical control signals, resulting in a fault in the control or in the data flow of the readout system. A malfunctioning of a circuit can originate in various reasons ranging from an occasionally affecting design bug to a failure due to an SEU in the circuit. The RCU functioning as a bridge interfacing the FECs and the online systems of ALICE plays a key role in realizing the robustness and stability of the data acquisition system. Logical faults may get ignored and removed with a new event readout or may cause a complete failure of the readout system depending on the location and time profile of the fault. As an example two operational errors caused by logical faults are presented here:

1. All channels listed in the readout list memories are expected to send the event data to their respective RCUs, as described in section 3.2.6. Unstable handling of the failure of any channel to transmit the event data to the RCU, or a misaligned flow of the event data in the RCU may cause the DAQ waiting for event fragments from the TPC leading to stop a data taking session in ALICE. In other words the reliability of the complete TPC readout system depends on the stable and robust operation of a single readout channel of the TPC.

2. The RCU must respond to all of the valid trigger sequences with correctly placed event ID in the CDH. A corrupted event ID due to any fault may lead to unplanned termination of the data taking session of ALICE.

Faults and errors in a complex system such as the TPC FEE are unavoidable, but their effects are required to be eliminated or minimized before they can affect the quality of the data or cause an unplanned termination of the data taking session of the ALICE experiment. The RCU firmware is implemented with special techniques to handle erroneous or unexpected conditions at each of its interfaces to other components (FECs, DDL, trigger) as discussed in the following sections.

5.1.1 The ALTRO interface

The ALTRO bus is based on the GTL (Gunning Transistor Logic) [33] which is a low-voltage signaling technology. The ALTRO bus protocol including all possible transactions is summarized in section 3.3.2.1. Variations in the volume of event data and trigger rate may lead to different switching activity on the GTL based backplanes in the FEE. These fluctuations in the switching activity may cause glitches or delayed (skewed) switching on the control lines of the ALTRO bus.

The ALTRO chip receives *CSTB* (section 3.3.2.1) and the trigger pulses from the RCU. These signals are protected by glitch filters in the ALTRO chip to block unwanted switching activity on the lines. The RCU receives two control signals driven by the FEC, *ACKN* and *TRSF*. These signals are filtered for metastability and glitches by two cascaded flip-flops in the RCU firmware. The RCU firmware features a number of status and errors registers which can be read out to find the reason for an erroneous condition on the ALTRO bus. These will be discussed in the following:

(a) Refreshed every clock cycle		(b) Sampled upon TRSF line	
[0]	<i>CSTB</i> , branch A	[0]	<i>CSTB</i> , branch A
[1]	<i>ACK</i> , branch A	[1]	<i>ACK</i> , branch A
[2]	<i>TRSF</i> , branch A	[2]	<i>TRSF</i> , branch A
[3]	<i>CSTB</i> , branch B	[3]	<i>CSTB</i> , branch B
[4]	<i>ACK</i> , branch B	[4]	<i>ACK</i> , branch B
[5]	<i>TRSF</i> , branch B	[5]	<i>TRSF</i> , branch B
[6]	ALTRO error line A	[6]	L1 of ALTRO
[7]	ALTROerror line B	[7]	L2 of ALTRO

Table 5.1: Status bits in two registers (“ALTBUSSTATUS” and “ALTBUSST_TRSF” [23]) for the ALTRO bus control lines

5.1.1.1 Continuous monitoring of the ALTRO control lines

All of the ALTRO control lines are monitored by the RCU for their status, which is refreshed every clock cycle. This monitoring is carried out by the RCU to verify that the ALTRO control lines preserve an expected status throughout the operation. The “TRSF” line is asserted when the ALTRO channel transfers the event data to the RCU, all of the ALTRO control signals are sampled upon assertion of the “TRSF” line to verify that none of the other control lines is asserted. Two status registers are accessible from the DCS to check the status of the ALTRO control lines, details of these registers are shown in table 5.1.

5.1.1.2 Monitoring of the ALTRO handshake protocol

The ALTRO bus is implemented with an asynchronous control handshake to the RCU for all transactions. The RCU must abide by this protocol (section 3.3.2.1) while addressing any FEC for any type of transaction. The ALTRO protocol is supervised through event driven watchdog timers, which monitor the total time any signal has been asserted during any transaction. This time is used to compare with expected value. The RCU firmware implements with two error and status registers, one dedicated for each branch to record any kind of violation of the ALTRO protocol during any transaction between the RCU and the FECs. Table 5.2 lists the details of all bits in the status and error register. The board controller also follows the ALTRO protocol to complete a transaction. The word “source” is used in table 5.2 to refer to both units (ALTRO or board controller) on the addressed FEC.

[0]	The ALTRO error line
FSM-Write	
[1]	ALTRO error line is asserted before <i>CSTB</i>
[2]	<i>ACKN</i> is asserted before <i>CSTB</i>
[3]	Source asserts ALTRO error line while waiting for <i>ACKN</i>
[4]	Source does not assert <i>ACKN</i> and time out
[5]	Source asserted ALTRO error line while waiting for command to complete
[6]	<i>ACKN</i> is not released by source
[7]	Source asserts the error line while waiting for broadcast command
FSM-Read	
[8]	Source asserted error line after “ALTRO bus” is driven from RCU
[9]	Source asserted error line while RCU waits for <i>ACKN</i>
[10]	<i>ACKN</i> is asserted before <i>CSTB</i>
[11]	Source asserts ALTRO error line while waiting for <i>ACKN</i>
[12]	Source does not assert <i>ACKN</i> and timed out
[13]	Source asserts ALTRO error line while waiting for release of <i>ACKN</i>
[14]	<i>ACKN</i> is not released by source
FSM-Readout	
[15]	FSM write ended with error when started for <i>CHRDO</i>
[16]	Source asserts ALTRO error line before asserting <i>TRSF</i> line
[17]	<i>TRSF</i> is not asserted (time out)
[18]	Source asserts ALTRO error line while data being transferred to RCU
[19]	<i>TRSF</i> line never got released from the source

Table 5.2: Bit level details of the error and status register of the ALTRO interface

If the ALTRO protocol is violated during a read or write operation, the respective bit in the status and error register is set and the result of the transaction is written in the result memory. The RCU communicates on the ALTRO bus through the instruction sequencer (section 3.3.3). Upon the failure of any transaction the instruction sequencer continues with the next instruction in the sequence.

When an ALTRO channel violates the readout protocol or does not complete the event readout due to an erroneous situation, the channel readout unit skips that channel addressing and continues by the next channel. A list of known errors during readout is provided in table 5.2. It is very complex to determine the root cause of any error during data taking but the most impor-

tant aspect the stable and reliable operation of the readout system, is accomplished by skipping the erroneous ALTRO channel and registering the error in the status and error register.

5.1.1.3 Verification of the trailer word from the ALTRO channel

The trailer word sent from each of the ALTRO channel consists of two pieces of critical information. It carries the address of the channel read out by the RCU and the block length (number of 10-bit words) of the event data packet [19]. This trailer word is stored in the RCU data memory, and the RCU verifies it by verifying if it matches the channel addressed by the RCU and the number of 40-bit words received as the event data from the channel. If the address of the channel addressed by the RCU and the address in the trailer word do not match, or if the block length in the trailer word is different than the number of words counted by the RCU, then the RCU will correct this word for the corrupted field in the newly constructed channel header according to the RCU data format (section 3.2.7).

5.1.1.4 Test methodology

The errors listed in table 5.2 were injected one by one on the ALTRO bus interface between the FEC and the RCU during the simulations, in order to test the respective path of error detection and reporting in the RCU firmware. It is inefficient and complex to emulate multiple erroneous conditions on the ALTRO interface in the simulation environment of the RCU firmware. Two different methods used to test the error detection and reporting by the RCU firmware are described as following.

- **Functional verification through simulation**

The communication between the RCU and FECs on the ALTRO interface was verified by various kinds of simulations as discussed in section 3.5. The FEC stimuli in the simulation environment can be configured to simulate the error handling mechanism under test. The complete simulation of the error handling mechanism was verified for its timing behavior, the status of the signals and the respective bits in the error register. An example of the simulation procedures and the waveforms is shown in Appendix B.1. All of the errors listed in table 5.2 were generated using the FEC stimuli and the handling mechanism of each individual error is verified accordingly.

- **Hardware based error emulation**

Multiple erroneous conditions were not simulated, as this is an inefficient and complex task. A special module in the RCU firmware was implemented to accept specific commands to introduce run time readout errors. These errors include all the errors related to protocol violation as mentioned in table 5.2 and also a mismatch of the ALTRO trailer. The complete system was verified for its behavior in handling multiple errors during readout. The results from these tests were compared and verified against the specifications and functional simulations. This special module, which was implemented to test and verify the error handling mechanism is not part of the final implementation of the RCU firmware.

5.1.2 Handling of erroneous trigger sequences

A valid trigger sequence consists of two trigger pulses, the L0 and the L1 and two messages of eight 8-bit words (L1 message and L2 message) from the ALICE CTP. The order of words

in a message and the time between trigger pulses is verified by the trigger receiver unit in the RCU firmware [6]. The verification of the trigger sequences and a stable handling of erroneous trigger sequences in all of the 216 RCUs and the busy box is essential for a reliable operation of the TPC data acquisition system. Inconsistency in the verification procedure of a trigger sequence between the RCU and the busy box may result in a lock-up situation. The busy signal from the busy box to the CTP may for example remain asserted for a trigger sequence which is rejected due to errors in the verification procedure, indicating that data acquisition from the RCU to the DAQ is in progress. Errors in the trigger sequences are reported in the CDH according to the ALICE scheme for handling trigger errors [13]. The detection and reporting of erroneous trigger sequences is verified by two methods: 1) Functional simulation; 2) Hardware test in the laboratory setup as discussed in the following

- **Functional verification through simulation**

The communication between the RCU and the CTP to issue trigger sequences is modeled for the simulation environment as discussed in section 3.5.1.4. The CTP stimuli in the simulation environment is configurable to issue four valid trigger sequences as listed in table 3.17. This is used to simulate the decoding of trigger sequences in the RCU. Timing thresholds set in the configuration registers of the trigger receiver unit are varied to create violations with respect to valid trigger sequences issued by the CTP stimuli. The complete logical processing chain from detecting a timing violation till the reporting in the ALICE CDH is verified against the specifications.

- **Hardware-based error emulation**

The test setup in the laboratory as described in section 3.5.2.1 is used to verify the handling of erroneous trigger sequences from the ALICE CTP. The CTP emulator and the LTU are used to issue trigger sequences with known errors to the RCU and to the busy box. In these tests the status and error register in the busy box and in the RCU are continuously monitored to verify and confirm that the errors in trigger sequences issued and detected by the RCU and the busy box are similar. A single TPC readout partition is triggered with a mixture of valid and erroneous trigger sequences emulating different errors. The event data packets received from the RCU for the valid and erroneous trigger sequences are decoded to verify that the reporting of errors in the ALICE CDH meets the required format in the reference [13].

5.1.3 The DDL interface

As shown in figure 1.7, three TPC readout partitions are connected to a single LDC through the DDL from three respective SIUs. The SIU translates the higher level commands and controls the event data flow from the RCU to the DAQ. The RCU receives two signals from the SIU to control the event data flow [32];

1. Ready to receive (*RDYRX*): it validates the status of link indicating that it is ready to transmit the event data from the RCU to the SIU.
2. Link full (*X_{off}*): assertion of this signal indicates that the RCU must stop the data transmission to the SIU, showing that the link is not able to accept data anymore.

The RCU firmware is implemented with a special mechanism to handle the activity on these two signals and to control the flow of event data from the RCU to the SIU.

5.1.3.1 Handling of the X_{off}

The assertion of the X_{off} to pause the event data transmission from the RCU to SIU is also named “back pressure” from the DAQ to the front end electronics. The X_{off} handling is complex due to its asynchronous assertion for an un-deterministic duration. It affects one complete chain of the event readout, pausing the data assembler (section 3.3.7) and event readout unit (section 3.3.2.3). The RCU is required to resume data transmission after the link is indicated to be available again by the deactivation of the X_{off} .

5.1.3.2 Test methodology

The basic mechanism handling the assertion of X_{off} in the finite state machines of the data assembler module is verified by various simulations. The back pressure involves all components of the DAQ, so an extensive handling of X_{off} in the laboratory setup was done to maximize the test coverage. The test setup in the laboratory as described in section 3.5.2.1 was used to test the handling of X_{off} and data integrity after the resumed transmission from the RCU to the SIU.

5.2 Radiation effects and the TPC readout electronics

Radiation effects in integrated circuits are caused by energy deposition from an ionizing particle traversing through the sensitive nodes in the circuit. Energy deposited may cause a transient current pulse named Single Event Transient (SET). SETs may cause a fault in the digital integrated circuit only if the resulting pulse coincides with the clock edge and changes the stored value in a memory element (flip-flop) [10]. An incorrectly stored value in a memory element is generally referred to as a Single Event Upset (SEU). In other words, in a digital integrated circuit an SEU occurs when an ionizing particle changes the stored value in a memory element from “1” to “0” or vice versa, as explained in [10]. In SRAM FPGAs, the effect of the SEU can be classified into two kinds of errors:

1. Soft Error: when an SEU modifies the value in a memory element stored by the operation of the implemented circuit, this kind of fault may get cleared by restarting the digital circuit or by restarting its operation
2. Hard Error: when an SEU changes the value of the configuration memory of the FPGA fabric, modifying the implemented circuit in the FPGA. This fault can only be corrected by reprogramming the FPGA device.

Radiation induced effects on the TPC readout electronics can not be neglected because of their placement into areas of high particle flux caused by the collisions of beam particles in the interaction region of the ALICE experiment [26]. SEUs in memory elements are the widely experienced repercussions of these radiation effects on digital integrated circuits. These SEUs may cause a range of erroneous situations in the readout electronics, depending on the severity level of the error produced by the SEU. Common errors may include corrupted event data, delayed event readout, missing data from certain FECs or ALTRO channels and readout being stuck permanently. These erroneous situations of the readout system influence the data recording efficiency of the TPC and the quality of acquired data as well.

Each TPC FEC is mounted with one SRAM FPGA and eight ALTRO chips resulting in the utilization of a total of 4,356 board controller FPGA devices and 34,848 ALTRO chips. A

total of 216 RCU boards, assembled with two daughter cards, contains 432 SRAM FPGA devices (RCU and DCS) and 216 flash based SIU FPGA devices. Different kinds of mitigation techniques and recovery procedures are implemented to cope with errors caused by SEUs, as described in the following sections:

5.2.1 Radiation mitigation in the ALTRO chip

All the FSMs controlling the interface and data path are protected against a single SEU by Hamming coding that implements an algorithm to correct single bit errors. Protecting state registers of the FSMs through single bit Hamming Code ensures that values of the state registers do not change to illegal states values. Double bit errors in the state registers are detected and recorded only in the errors and status register of the ALTRO [20].

5.2.2 Radiation mitigation in the board controller

The board controller is a SRAM FPGA based circuit. Its functions of controlling the GTL bus drivers and of updating the hit-list memory in the RCU in sparse readout are essential for the stability of the event readout. The configuration of the FPGA is loaded into the device from an on-board EPROM upon powering up the FEC. The firmware of the board controller is not implemented with any type of mitigation technique to cope with radiation effects. At the start of each physics data taking session, reconfiguration of the FPGA firmware is performed by issuing the command “CNFG-BC”, which is issued to all RCUs and reconfigures the board controller firmware on all FPGAs by turning off the voltage regulators for a period of 100 μ s.

5.2.3 Radiation mitigation in the DCS card

The DCS card is discussed in section 2.7.2.1. The DCS board firmware realized its FSMs with one-hot implementation technique. The ComStat register is mitigated against SEUs using Triple Modular Redundancy (TMR) and voting techniques [5]. The key control signals from the DCS board firmware to the RCU are filtered using a shift register and the logical AND operation of shifted values is used to get the final value of each output signal. This implementation avoids glitches or temporary bit-flips due to radiation effects, preventing erroneous resetting of the DCS firmware or unplanned changes in the mode of operation [5].

The trigger interface on the DCS board is the key component in the readout system, which is realized using the TTCRx chip. The TTCRx chip is fabricated in radiation-hard SOI technology [21], which is highly immune to SEUs. Moreover, the control logic and configuration registers are protected against single bit errors by Hamming error detection and correction techniques.

5.2.4 Radiation mitigation in the SIU card

The main firmware of the SIU card is implemented in ACTEL *ProASIC+*[®] FPGA which provides the DDL protocol and the control logic of its interface to the RCU. The ACTEL *ProASIC+* FPGA is a flash FPGA, which uses latches for its configuration memory, that are not affected by an SEU [12]. However, SEU can still occur in the internal RAM, or in standard logic cells. In the SIU firmware, internal RAM is used in implementing FIFOs and parity bits are used to detect errors, and these errors are signaled in the “Data Transfer SStatus Word” (DTSTW) [32].

5.2.5 Radiation mitigation in the RCU

Active Partial Reconfiguration (APR) is utilized to correct the SEU in the configuration memory of the RCU FPGA without interrupting the operation of its firmware. It implies that an SEU is detected and corrected immediately after it has occurred, hence minimizing the risk of an operational fault in the RCU. The APR is carried out by reading and writing the smallest portion of data that can be read from, or written to, the configuration memory in one operation (called frame). The APR of the RCU firmware can be operated in two different modes:

1. Scrubbing: the configuration memory of the RCU is continuously overwritten regardless of the occurrence of an SEU in the configuration memory;
2. Frame Read-back Verification and Correction (FRVC): the configuration memory is read from the RCU one frame after the other, compared with the corresponding frame from the on board flash memory and if a bit flip is found the correct status of the bit from flash is restored in the configuration memory of the RCU [26].

The APR of the RCU firmware is implemented using a flash based FPGA (Actel ProASICplus[®] APA075 [39]) and a flash memory device. The main configuration controller is implemented in the flash based FPGA device, and the configuration frames and files needed for the APR are stored on the flash memory device. The design and implementation of the APR for the RCU firmware is the main topic of the work done in this reference [26]. The APR is a solution to correct SEUs and prevent the accumulation of SEUs in the configuration memory, but it does not inhibit the occurrence of the SEU. This technique is able to detect and correct the SEU in the configuration memory but it is not able to remove any possible fault induced in the operation of the RCU in the short time before it is corrected. Life time of a single SEU in the RCU firmware spans from 350 μ s to 150 ms [26]. The RCU clock operates at 40 MHz corresponding a clock period of 25 ns and minimum readout time of a single event (empty) under sparse readout mode is 147 μ s. A relatively long, minimum life time of a single SEU as compared to the minimum event readout time clearly indicates that the APR technique may not be able to protect the RCU firmware from the consequences of SEUs. A clock period of 25 ns also poses a potential risk of propagating logical faults into the circuit caused by SEUs. To study the robustness and effects of SEUs in the configuration memory of the RCU firmware under such timing conditions, the fault injection technique [26] was used.

The idea is to flip a single bit in the configuration memory of the RCU firmware to realize the effects of an SEU during an event readout. The pedestal memories of ALTRO channels were written with real events. Triggered by the LTU, the readout system was configured to send the contents of the pedestal memories. The test was completed following the steps shown in figure 5.1 (explained in [7]). Three different consequences of the injected faults were experienced:

1. Stuck readout: this was the most critical effect of the injected bit-flip in the configuration memory of the RCU firmware, resulting in complete failure, stopping the event readout. This type of error is named as reliability errors;
2. Data corruption: the event data received did not match with the data written in the pedestal memories, showing that the original event data written into pedestal memories was modified in the RCU;
3. Loss of data: event data did not contain the complete data from all the corresponding channels.

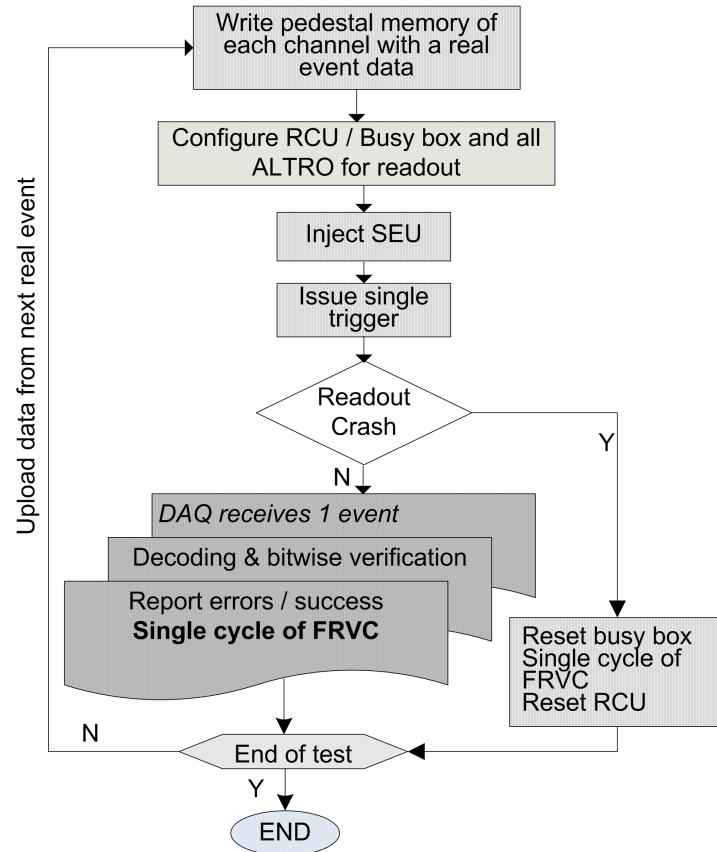


Figure 5.1: Flow chart of the fault injection test

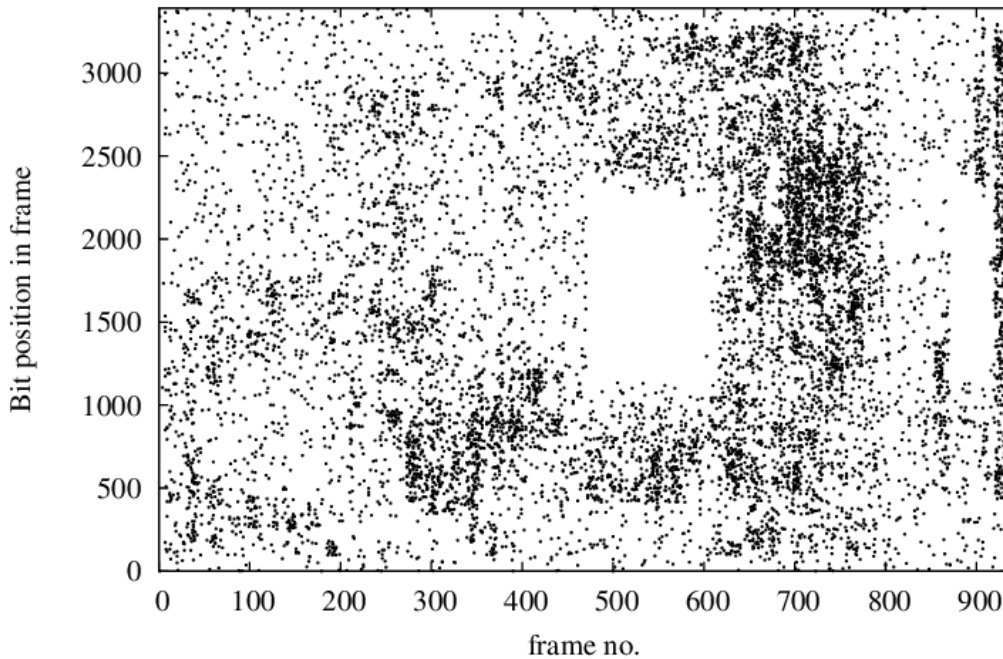
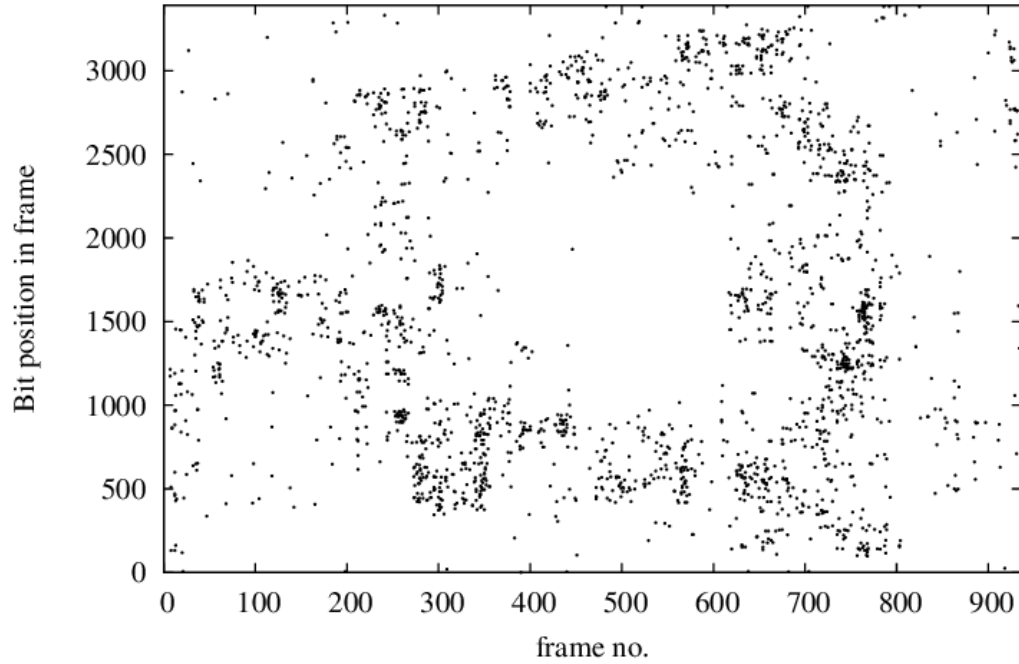


Figure 5.2: Bitmap of the injected faults causing an error in the event readout

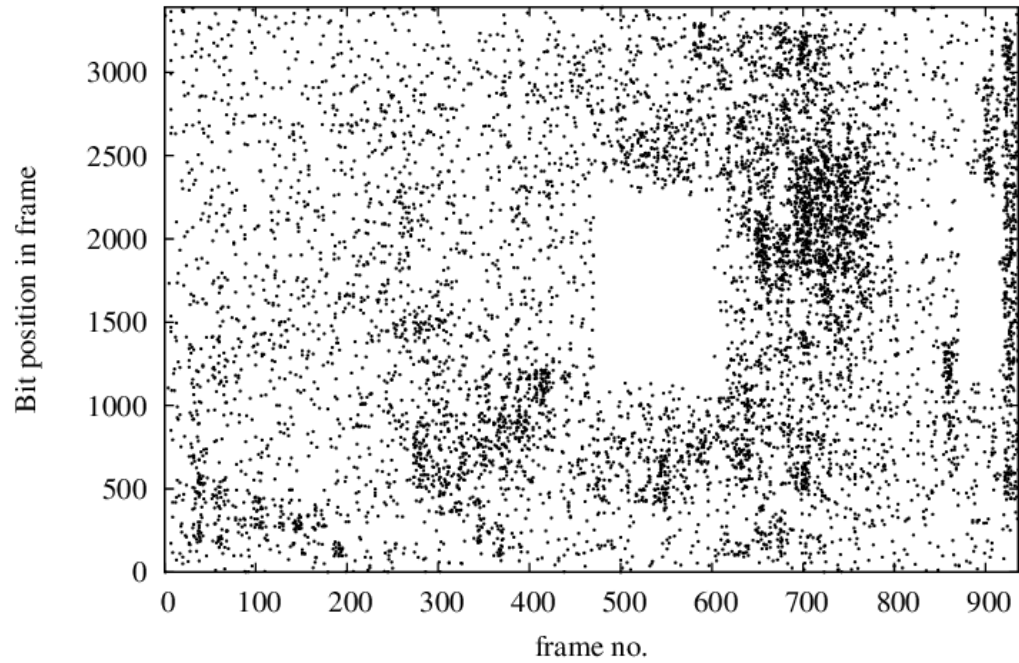
Two types of errors (2 & 3) are named as performance errors. There are 3,161,344 configuration bits to be programmed for the Xilinx Virtex-2 vp7 device excluding BRAMs primitives of the Xilinx [34]. A total of 206,151 bits were flipped to inject a fault in the configuration

Fault type	Number of Faults	SEUs/Fault
Total	10,341	19.9
Stuck (reliability)	2,210	93.5
Loss of data	2,499	82.6
Corrupted data	5,632	36.6

Table 5.3: Results of the fault injection test [7]



(a) Injected faults resulting in reliability errors



(b) Injected faults resulting in performance related errors

Figure 5.3: Total faults injected and the failures in the RCU

memory of the RCU firmware. This correspond to about 6.5% of the total amount of bits in the configuration memory of the device. The configuration bits from 1000 to 2100 between frame number from 450 to 650 as shown in figure 5.3 are excluded from the fault injection, this part shows the built-in core of power processor in FPGA device [35]. However, the tested bits are randomly distributed across the configuration bits in the FPGA device. The main goal to find possible faults due to SEUs and their possible consequences on the event readout at the ALICE experiment is met by this test. The analyzed results are shown in table 5.3. These results are used to get an estimate of the correlation between the rate of the RCU failures with respect to the measured SEU rate. Detected and corrected SEUs are continuously monitored by the DCS and the collected data has been analyzed for a period with p-p collisions from start of May until end of August in 2011 [27]. A linear correlation between integrated luminosity and the SEU rate was observed with a mean value of 0.49 SEU/nb^{-1} on all of 216 RCUs on the TPC detector. The calculated rate of reliability faults leading to cause unplanned termination of data taking is $0.005 \text{ faults/nb}^{-1}$, which leads to an expected rate of 0.13 reliability faults per day in all of 216 RCUs. It is difficult to find and relate the root cause of each fault at the ALICE TPC due to the other sub-systems involved in normal operation. During a normal data taking session of ALICE, faults related to data quality and performance are detected after the processing of the data received by the DAQ which is too late to reach the cause in the TPC readout system due to its continuous operation. Reliability faults cause the TPC readout system to stop and lead to an unplanned termination of data taking session in ALICE, such cases are analyzed to find the fault and respective source. Most of the logical faults induced due to SEUs can be tracked by analyzing the status of critical bits in the status registers. All of the EoR reasons from start of May till mid of June 2011 were analyzed to find the EoR reasons caused by reliability faults induced by SEUs in the RCU firmware. This period falls within the duration which was analyzed to determine the relation between SEUs and integrated luminosity in [27]. A total number of 39 data taking sessions were terminated due to the TPC FEE, 5 out of 39 reasons might have been resulted from faults induced by SEUs. It corresponds to 0.11 reliability faults per day as compared to expected 0.13 faults per day. Increase in luminosity during the four months period can be a reason for slightly higher expected fault rate. Only the detection of reliability faults induced by SEUs does not prevent an unplanned termination of data taking session, to cope with complex consequences of SEUs and their expected increase with luminosity is handled through a new technique implemented at higher operational level as discussed in section 5.2.7.2.

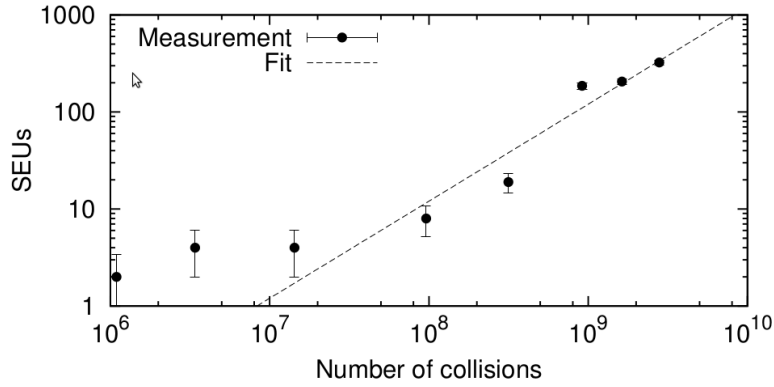
5.2.6 Measurement of SEUs in the TPC readout electronics

Each ALTRO channel contains 10,240 SRAM cells constituting 1kByte pedestal memory with word size of 10-bits. These memory cells were not used in the processing of the event data during the LHC operation. There are 16 channels in each of the 34,848 ALTRO chips on the TPC, providing a total number of $5.7 \cdot 10^9$ SRAM cells to measure SEUs during the LHC operation. The measurement of SEUs is carried out by monitoring the bit flips caused by the p-p collisions at energy level of 7 TeV [17].

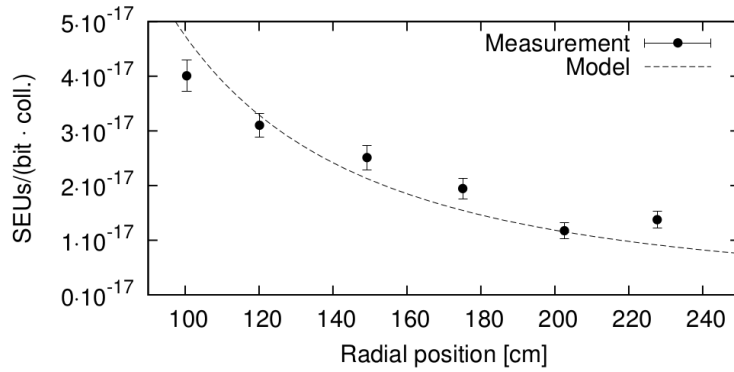
1. Configuration and verification of the pedestal memories

The memories were configured with specific data patterns (“0101010101” or “1111111111”), then the memories were read back in order to verify their contents against the used pattern. This allows the detection of bits which were permanently stuck at ‘1’ or ‘0’.

2. Operation of TPC under radiation



(a) Relation of SEUs with number of collisions during the total period of measurements



(b) Radial distribution of SEUs per *bit · collisions*

Figure 5.4: The measurements of SEUs in the ALTRO chips [17]

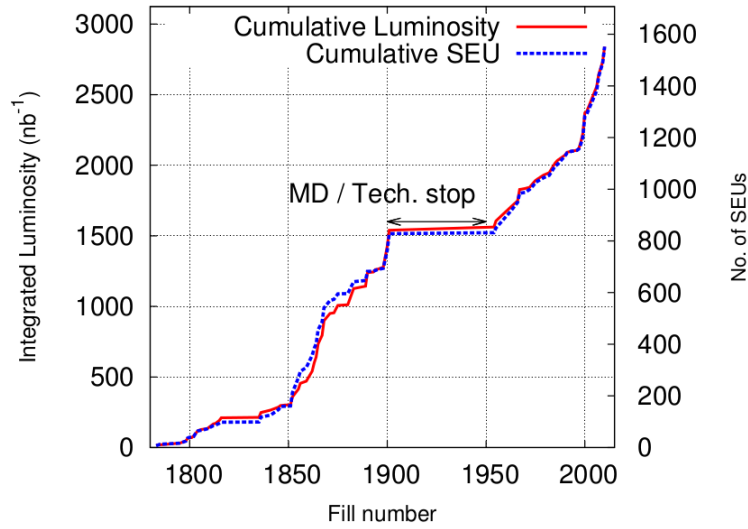
The configured pedestal memories were left while the TPC operated in the radiation environment.

3. Read back of the pedestal memories

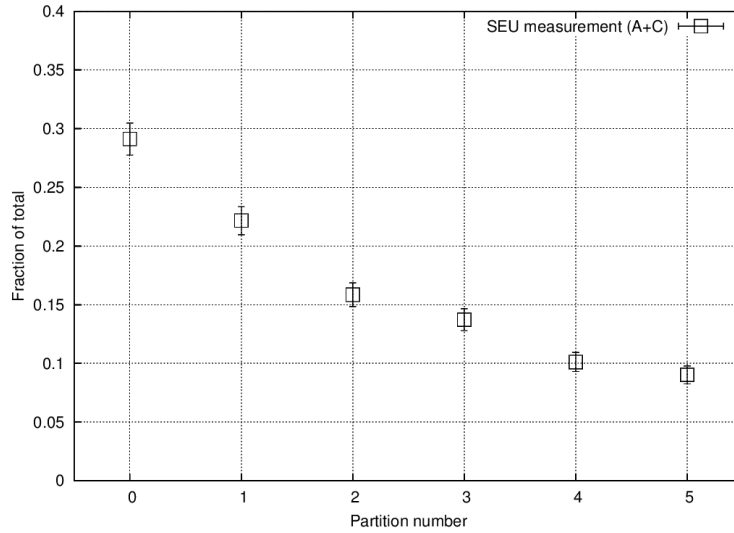
The memories were read back and all locations were again checked against their reference data.

The number of bit errors from all readout partitions was analyzed and figure 5.4(a) shows the radial dependence of the measured SEUs from the interaction point. The number of SEUs was correlated with the integrated number of collisions, which is calculated from the rate of collisions reported by the V0 detector of ALICE for the duration of the measurements. SEUs in the pedestal memories clearly indicates a linear dependence on the number of collisions, as shown in figure 5.4(b). The bit error rate in the ALTRO chip at a given collision rate can be approximated from the proportionality constant of $(1.2 \pm 0.1) \cdot 10^{-9}$ per collision [17].

The APR solution to correct SEUs in the RCU firmware is discussed in section 5.2.5. The FRVC has been operational in all of the RCUs at the ALICE TPC since the beginning of the LHC operation. The ACTEL FPGA as a main controller of the APR contains a dedicated counter to count the detected and corrected SEUs [26] along with the register providing the address of the corrected frame in the configuration memory of the RCU. The counted number of detected and corrected SEUs in the RCU firmware is monitored continuously by the FEE server on each of the DCS boards. Whenever a change is detected in the counted value, the new value of the counter and the respective frame number are time stamped and stored in a database. The counted number of SEUs during the LHC operation for the period of May to August 2011



(a) Relation of SEUs with integrated luminosity during the total period of measurements



(b) Radial distribution of SEUs in the RCU firmware

Figure 5.5: The measurements of SEUs in the RCU firmware [27]

was analyzed for its relation with the integrated luminosity of the LHC fills. A comparison of the integrated luminosity and the total number of SEUs in the configuration memory of the RCU is shown in figure 5.5(a). A linear relation between the integrated luminosity per fill and SEUs detected is demonstrated in figure 5.5(a). This dependence is not clearly visible at the lower range of luminosity but it becomes prominent with the increased number of SEUs at a higher value of integrated luminosity [27].

Both of these measurements were carried out during different periods of the LHC operation and on two different devices (an ASIC (ALTRO) and an FPGA (RCU)). The radial dependence is observed for the detected SEUs in both devices. The measurement carried out by the ALTRO pedestal memories shows a linear dependence of SEUs on the number of collisions and the analysis done for the SEUs in the RCU configuration memories shows a linear dependence on the integrated luminosity. These measurements clearly not only justify the implemented radiation mitigation techniques but also show a support to detect and handle the errors induced

by SEUs at a higher level in the readout system, as discussed in the following section.

5.2.7 Handling of radiation effects on the TPC readout electronics

There are three important limiting factors which may affect the reliability of the complete data acquisition system of the ALICE TPC, listed in the following.

1. The RCU firmware is a complex design, interfacing with six different physical interfaces and running at least four important concurrent processes interacting with each other during an event readout as detailed in section 3.2.6. The configuration memory of the RCU (Xilinx FPGA) is protected by the APR technique through the ACTEL FPGA but the internal logic implemented in the RCU firmware is not protected with any mitigation technique to handle soft errors or faults triggered by SEUs in the configuration memory. Severe area constraints were faced while implementing the mitigation technique in the RCU firmware, which are due to the maximum utilization (see section 3.4) of the FPGA resources.
2. The second weak point of the system are faults induced due to SEUs in the data path and configuration registers of the ALTRO chip. These memory cells are not protected against SEUs and the bit errors in the configured parameters may lead to chip level corruption or loss of the event data.
3. The board controller FPGA on the FEC is programmed at the start of each data taking session, ensuring cleaning of the SEUs in the configuration memory or in the operation logic. It is not protected against any fault induced during data taking, imposing a third concern for the reliable operation of the TPC FEE.

The periodic and fault driven reconfiguration of these registers during data taking was a foreseen solution to correct radiation induced errors before a fault becomes propagated in the readout system causing an unplanned termination of the data taking.

5.2.7.1 Periodic and interrupt driven reconfiguration

An originally foreseen procedure to handle the errors induced on the TPC FEE during data taking was a periodic reconfiguration during an abort gap between the circulated beam or a reconfiguration of the readout partition driven upon an interrupt from the RCU to the DCS board as shown in figure 5.6(a). The RCU is able to assert an interrupt signal to the DCS board indicating that either the RCU or any of the FEC in the readout partition has encountered an error which requires immediate corrective action by the DCS. The RCU pauses the readout by not sending the latest event to the DAQ, but it keeps a record of the triggers received from the CTP. Consequently the busy signal to the CTP will not be released from the busy box, hence no more triggers will be issued from the CTP. This pause in readout and trigger provides enough time for the DCS to correct the fault and inform the RCU to resume the readout.

This technique is not adopted as a procedure for handling the effects of SEUs on the FECs due to the following reasons.

1. Reliability of the DCS board

The stability of the DCS board while functioning under a radiation environment poses the weakest point concerning the reliability of the TPC FEE. Reasons causing an unplanned termination of the data taking sessions in the period between the 2nd May 2011

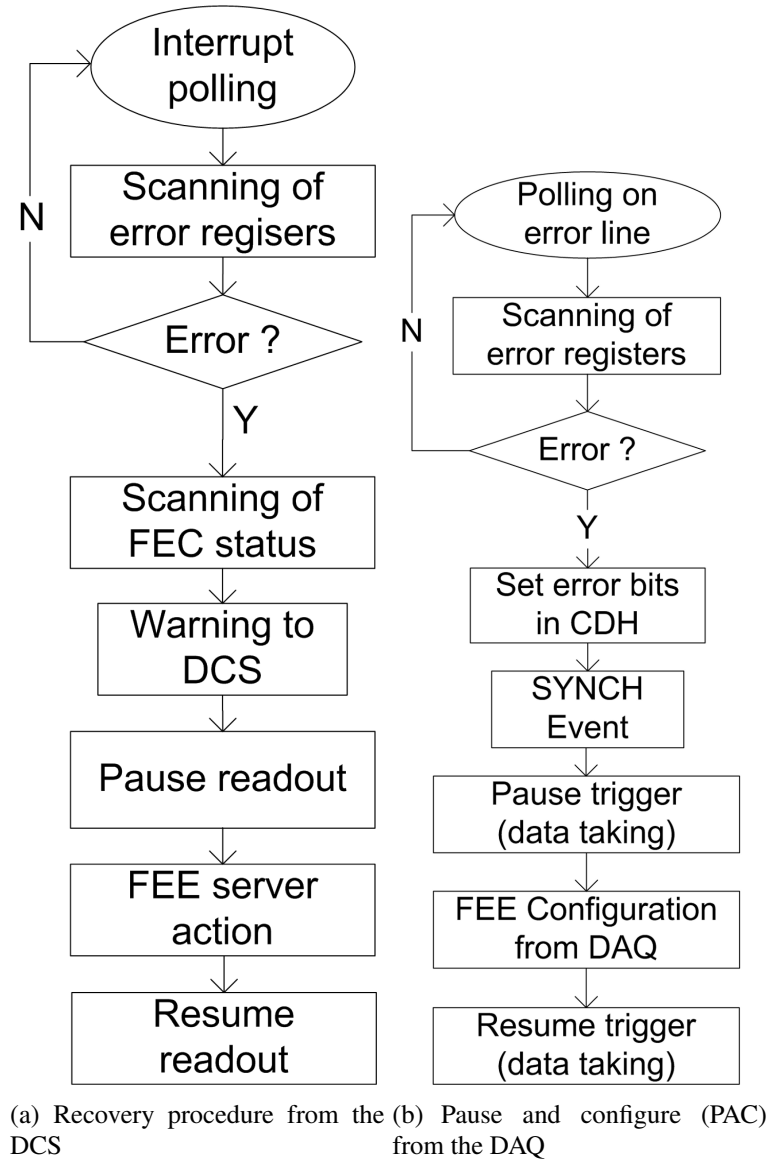


Figure 5.6: Recovery procedures for FEC related errors

and 16th June 2011 were analyzed. It was confirmed that 28% [22] of the unplanned termination of data taking sessions at ALICE were caused by the erroneous behavior of the DCS boards, whereas the FECs were responsible for only 9% of such cases.

2. Inconsistency in event handling

One idea was to stop the triggers from the CTP through the busy box, by not sending an event to the DAQ and keeping it locally in the RCU till the erroneous readout partition is recovered by the DCS. But an artificial pause of the event readout caused by a single TPC readout partition may lead to inconsistency in the number of total events received from the TPC readout partitions. For example, four consecutive triggers may have been received before the busy signal from the busy box is detected by the CTP. In principle such an inconsistency could be overcome if the readout partition which is reconfigured by the DCS to recover from the radiation effects sends four empty events. Other readout partitions will send complete events including the event data acquired from the FECs [22].

In a situation where any of the erroneous FEC (affected by SEUs) in a readout partition requires to get reconfigured from such a DCS board which itself is in an erroneous condition due to radiation effects, the DCS will not be able to provide a reliable solution for correcting errors induced due to SEUs in the FECs or in the RCU during data taking. An alternate solution to this technique was developed and implemented as discussed in the following section.

5.2.7.2 Configuration of the readout partition during readout

A new procedure for handling radiation induced faults in the FECs has been developed and implemented to reconfigure a complete single readout partition without terminating the data taking session at ALICE. It is called Pause And Configure (PAC), reflecting its function that the trigger is paused to configure the FEE. It is carried out from the DDL side by reversing the direction of the link during data taking. The flow chart of the complete procedure is shown in figure 5.6(b). The ALICE CDH as described in section 3.2.7 is constructed in the RCU and sent as a first part of each event data. The fourth word of the ALICE CDH is modified, implementing a new field named “PAC request [31:28]”; details of each bit is shown in table 5.4. The key steps involved in this procedure are listed as following.

- **Detection of the erroneous condition**

The RCU keeps polling the ALTRO error lines and monitors the errors related to channel readout. If an ALTRO error line is detected as permanently asserted by the RCU, the respective bit in the CDH, as shown in table 5.4 is set to “1”, making a request for the PAC. The ALTRO control lines are monitored and verified for their assertions according to the protocol discussed in section 3.3.2.1, and if any of the control line are found permanently asserted then the respective bit (bit[30]) of the PAC field is set to “1”, indicating the erroneous behavior of the ALTRO bus in the readout partition. The verification of the ALTRO trailer is explained in section 5.1.1.3; bit[29] in the PAC field is set to “1” if the trailer word from any channel is found to be corrupted. It is important to note that this bit is asserted when corrupted data is found in the previous event as compared to the one currently being handled by the RCU. The channel trailer is verified while event data is read from the data memories but the CDH of the current event is sent before verification of the trailer. Consequently if this bit is found in the PAC request field, it indicates that the previous event contained corrupted data trailers from the ALTRO channels.

- **Request to pause the trigger and configure**

The CDH of each event is decoded by the DAQ and the PAC bits are evaluated to decide the severity of the fault in the readout partition. The ALTRO control line, being permanently asserted, shows a severe situation which requires an immediate response from the DAQ to proceed to configuring the complete partition. The DAQ is configurable to act upon the PAC request by decoding the severity level of the fault and tolerating a specific number for events, but the communication between the ECS and the DAQ may span to a few seconds so in most cases the PAC request is made without deciding about the severity of the fault.

- **Handling of event readout and SYNCH event**

As shown in figure 5.6(b), the DAQ communicates to the ECS requesting the SYNCH event to be issued from the CTP. All of the TPC readout partitions are required to respond with SYNCH event data, indicating that there is no event readout to be processed or left in the data memories to be shipped from the RCU. After the DAQ receives the SYNCH

event, it proceeds to closing the DDL links and reconfiguring the TPC FEE by executing identical commands as at the start of data taking. After the configuration, the DAQ restores the DDL link to receive event data from the RCUs and communicates its status to the ECS to resume the triggers from the CTP.

[28]	“0”
[29]	Corrupted ALTRO data is detected by RCU during data taking
[30]	Any of the ALTRO control lines are detected as stuck at '0' or '1'
[31]	The ALTRO error line is asserted from branch A or B

Table 5.4: Details of the bits to signal the PAC request in the ALICE CDH

The PAC solution was completely tested in the hardware setup in the laboratory (section 3.5.2.1). It was extensively verified that reversing the direction of the DDL during event readout does not cause any kind of malfunctioning in the RCU. This technique provides a solution to a permanently stuck situation of the event readout by resetting the busy box before reconfiguration of the TPC FEE.

5.3 Summary

The reliable operation of the TPC data acquisition system depends on the robustness of the readout operation of a single channel. Fault tolerance techniques were implemented in the RCU firmware to ensure the reliability of all the interfaces and operations conducted by the RCU. These techniques were simulated, verified and tested in the laboratory. The effect of

Run number	Beam type and energy	Duration
133282	p-p at 7 TeV	9 hours 51 minutes
138275	Pb-Pb at 2.76 ATeV	8 hours 4 minutes
184968	p-p at 8 TeV	6 hours 38 minutes

Table 5.5: The longest data taking sessions of the ALICE experiment (September 2010 - July 2012) [4]

these fault tolerant techniques have been well demonstrated during data taking sessions at the ALICE experiment. The monitoring of the FECs and the RCUs may get affected by SEUs but the DCS framework of the ALICE TPC is implemented in such a way that it does not interfere with the data taking. Fatal errors, for example a mis-match of the event ID in the busy box, can cause an unplanned termination of the data taking sessions, but the proposed technique (PAC) can be used to ensure that the TPC data acquisition system will not cause an unplanned termination of the data taking session. The PAC is implemented but it is not yet completely functional due to practical issues of handling the SYNCH event in the HLT. The TPC contains 216 DDLs out of total 468 DDLs in the ALICE experiment, constituting 46% of the total links used in the ALICE DAQ. EoR reasons [22]¹ of the data taking sessions for a randomly selected period of 30 days (16 July 2012 till 16 August 2012) is analyzed and figure 5.7 shows that the TPC is responsible only for 6% of EoR reasons caused by the detectors of ALICE. The final implementation of the PAC is expected to handle all of the EoR reasons due to the TPC FEE (reliability and performance). The summary of three data taking sessions under three different

¹Restricted access

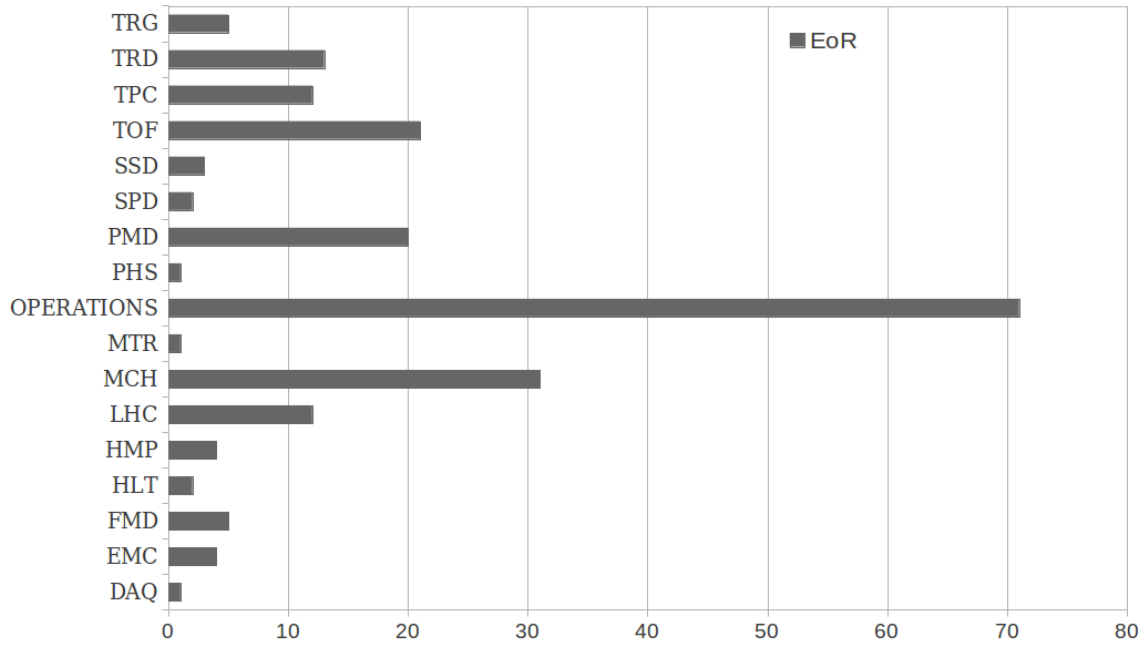


Figure 5.7: Summary of EoR reasons for the period of 30 days from the ALICE E-logbook

operational conditions is presented in table 5.5. The listed EoR reasons of these data taking sessions were not originated by the TPC readout system.

Analyzed EoR reasons and long data taking sessions at the ALICE experiment demonstrate that the TPC data acquisition system preserves a high level of reliability and robustness.

Chapter 6

Conclusions and Outlook

This thesis has mainly focused on the design, implementation, full scale verification and optimization of the data acquisition circuit of the ALICE TPC. The heart of the data acquisition system, the RCU firmware has been designed, implemented and successfully commissioned to be used on the ALICE TPC. State of the art simulation techniques to test and to verify the functionality of the RCU firmware are developed and utilized. New methods to test the compatibility of the RCU firmware with other components of the readout system are defined and set as standard commissioning procedure of the RCU firmware before it is used at the ALICE TPC. The RCU firmware is currently being used by four different detectors of the ALICE experiment including TPC, PHOS, EMCAL and FMD. The firmware is configurable and has demonstrated successful operation by meeting the requirements for front-end electronics of all these four detectors.

The design of the RCU firmware is large, utilizing almost 99% of the FPGA resources. Its size is large mainly due to the amount of implemented “intelligence” in the circuit used for verifying the instructions, monitoring the status of its interfaces, diagnosing and reporting erroneous conditions and resuming the normal operation of the RCU. Consequently, having more than six parallel threads (processes) in its circuit interacting with each other during its normal operation complicates the verification and simulation of the firmware. Parallel processes in the hardware not only increase the complexity but also consume more FPGA resources required for their independent synchronous (counters, register) and combinational (comparators, multiplexers) logic. SRAM FPGAs are commonly used to provide flexibility to improve or add new features in the operation of a digital circuit, but such a large and complex design reduces the level of conceived flexibility by providing extremely limited resources for any new feature and requiring more time for the simulation and for the system level verification. Two system level aspects to optimize the data acquisition system are discussed in the following conclusions. These aspects directly influence the design of the RCU firmware and may have simplified its implementation in various aspects. The RCU is required to configure the FEE of all four detectors mentioned above by executing the instructions packed in a sequence written in the instruction memory. The RCU records the result of each transaction in the result memory. This feature could have been implemented at software level, saving at least 15% of the FPGA area resources. Handling the back pressure during readout is complex because the X_{off} has to be checked before writing every single 32-bit word of event data to the buffer in the SIU interface unit. The effect of the X_{off} has to be propagated to pause the event readout and resume it from the next data word of the same channel, increasing the design complexity by having two branches working in parallel. A DDL without any back pressure would not only reduce the

complexity, minimizing the time spent for verification, but would also save FPGA resources. A simplified version of the RCU firmware may offer enough area and FPGA resources to implement SEU protection technique for the event readout chain in the RCU firmware.

The performance optimization of the event readout to meet the design specifications of the ALICE TPC data acquisition system has been one of the key objectives of this thesis. A systematic evaluation of the readout performance is done, from a single channel readout to the complete event readout of the TPC. It has been demonstrated that the performance of the ALICE TPC data acquisition has been unprecedented by preserving a better performance than the required specifications.

The robustness of the TPC readout electronics has been focused on as one of the key challenges along with optimizing the readout system. The reliability of the complete TPC readout system depends on the fault tolerant operation in acquiring the event data from a single TPC readout channel. The data path and configuration registers of the ALTRO chip and the implemented logic of the RCU firmware are not protected by any mitigation techniques against SEUs. It therefore poses a potential risk for the reliable operation of the TPC data acquisition system. A new technique (PAC) is proposed and implemented to handle the radiation induced effects on the TPC FEE. It offers a reliable solution to an unplanned termination of the data taking session and to a permanently stuck situation due to a reliability fault in the FEE. The PAC is adopted as a standard technique to recover a detector from an erroneous situation without aborting the data taking session of the ALICE experiment.

Appendices

Appendix A

```

TXGAP 5 us
DCSWR 0x4000 0x00201F09 -- TRGRx CONF
DCSWR 0x5100 0x00FF00FF -- FEC ACTIVE LIST
DCSWR 0x5101 0x0001c7c5 -- ALTROIF
DCSWR 0x5102 0x00013fff -- TRIGCONF
DCSWR 0x5103 0x00000000 -- RDOMOD
DCSWR 0x5104 0x00000000 -- DPCFG1
DCSWR 0x5105 0x00000000 -- DPCFG2
DCSWR 0x5107 0x00000000 -- BP_VERSION
DCSWR 0x5108 0x00000000 -- RCU ID
ENSIU
TXGAP 10 us
ENCTP
TXGAP 180 us
-END-
** SIU commands **
EOBTR
STBWR 0x00001000
TXGAP 1 us
EOBTR
STBWR 0x00001800
TXGAP 1 us
RDYRX
TXGAP 500 us
-END-
SOD--
L0L2a
SYN--
L0L2a
L0L2a
EOD--
-END-

TXGAP 1 us
DCSWR 0x4000 0x00201F09 --
DCSWR 0x400A 0x002800F8 --
DCSWR 0x5100 0xFFFFFFFF -- ACTIVE FEC LIST
DCSWR 0x5300 0xFFFFFFFF -- GLOBAL RESET
DCSWR 0x8005 0x00001B0B -- FC BUS ADDRESS
DCSWR 0x8010 0xFFFFFFFF -- EXECUTE COMMAND
DCSWR 0x8005 0x00005005
DCSWR 0x8010 0xFFFFFFFF
TXGAP 50 us
DCSWR 0x8005 0x00001B06
DCSWR 0x8010 0xFFFFFFFF
DCSWR 0x8005 0x00005805
DCSWR 0x8010 0xFFFFFFFF
TXGAP 50 us
DCSWR 0x8005 0x00001B0A
DCSWR 0x8010 0xFFFFFFFF
DCSWR 0x8005 0x00005902
DCSWR 0x8010 0xFFFFFFFF
TXGAP 50 us
DCSWR 0x8005 0x00004805
DCSWR 0x8010 0xFFFFFFFF
DCSWR 0x8005 0x00005C04
DCSWR 0x8010 0xFFFFFFFF
TXGAP 50 us
DCSWR 0x8005 0x00004902
DCSWR 0x8010 0xFFFFFFFF
TXGAP 50 us
DCSWR 0x8005 0x00004C04
DCSWR 0x8010 0xFFFFFFFF
TXGAP 50 us
DCSWR 0x8005 0x00005005

```

- (a) An example of script to perform event readout including the SOD and the EOD events
- (b) An example of script verifying transactions on the FC bus

Figure A.1: Two examples of script files to simulate the RCU firmware

(a) Test of full readout

TRG Rate (Hz)	TRG Mode	MEB settings	Duration (Minutes)
50k	Random	1	5
25k	Random	1	5
12.5k	Random	1	5
5k	Random	1	5
3k	Random	1	5
1k	Random	1	5
50k	Random	4	5
25k	Random	4	5
12.5k	Random	4	5
5k	Random	4	5
3k	Random	4	5
1k	Random	4	5

(b) Test of sparse readout

TRG Rate (Hz)	TRG Mode	MEB settings	Duration (Minutes)
50k	Random	1	5
25k	Random	1	5
12.5k	Random	1	15
5k	Random	1	5
3k	Random	1	10
1k	Random	1	10
50k	Random	2	10
25k	Random	2	10
12.5k	Random	2	10
5k	Random	2	5
3k	Random	2	10
1k	Random	2	10

Table A.1: Final qualification of the RCU firmware at the ALICE TPC

Event Readout on Both Branches (ALTRO BUS A – B)

Page

SOBDin= Data being written into output buffer

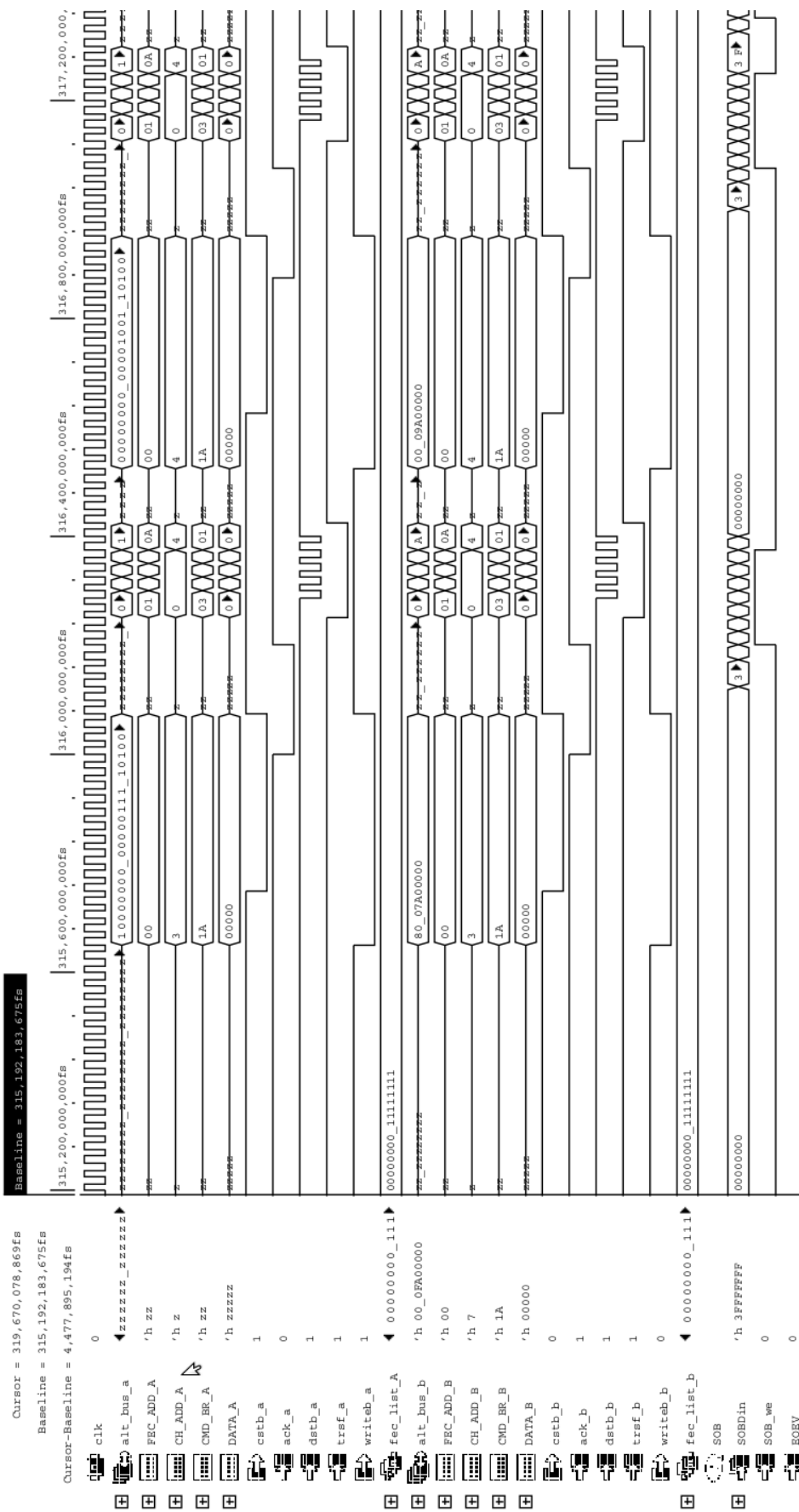


Figure A.2: Waveform view of the simulation in progress for the ALTRO bus interface

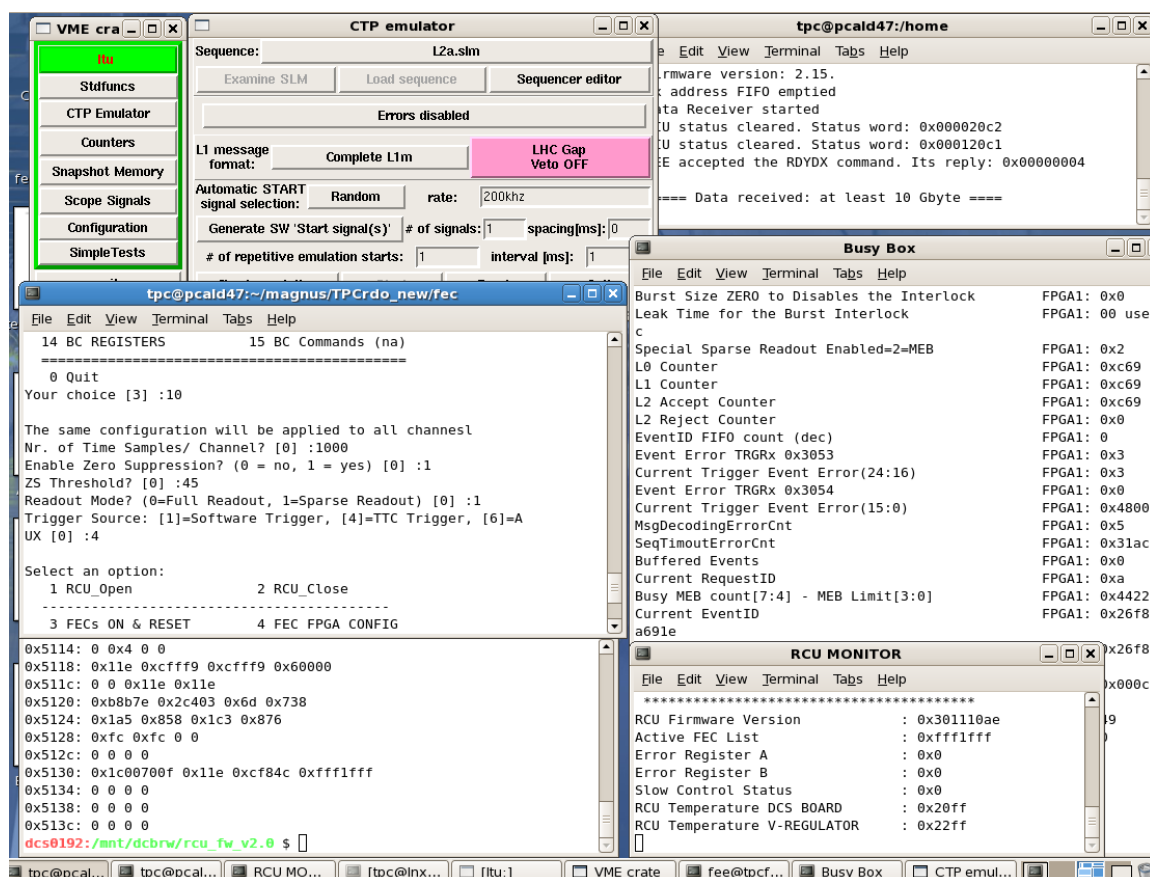
```

CDH Start: FFFFFFFF
Data : 020EC4DE MSW : 020 2ndMSW : 3B1 LSW : 0DE
Data : 003AB278 MSW : 003 2ndMSW : 2AC LSW : 278
Data : 02012EDF MSW : 020 2ndMSW : 04B LSW : 2DF
Data : 01000378 MSW : 010 2ndMSW : 000 LSW : 378
Data : 0000012E MSW : 000 2ndMSW : 000 LSW : 12E
Data : 00000000 MSW : 000 2ndMSW : 000 LSW : 000
RCU Trailer : 80000000 Number of 32 Bit Words :00000000
RCU Trailer : 84000000 Error Register Branch A - B :00000000
RCU Trailer : 88000000 Raed out Error Register :00000000
RCU Trailer : 8C000000 No. of Block LengthThe Mismatch :0000
No. of Channel Address Mismatch :000
RCU Trailer : 900000FF Actiev Front End Card Br. A :000000FF
RCU Trailer : 940000FF Actiev Front End Card Br. B :000000FF
RCU Trailer : 98000000 Copy of Altro Register DPCFG1 :00000000
RCU Trailer : 9C0F1421 Copy of Altro Register DPCFG2 :00F1421
EOF RCU TR. : E0020009 Parameter Code:8 Value:00200009
BYE BYE

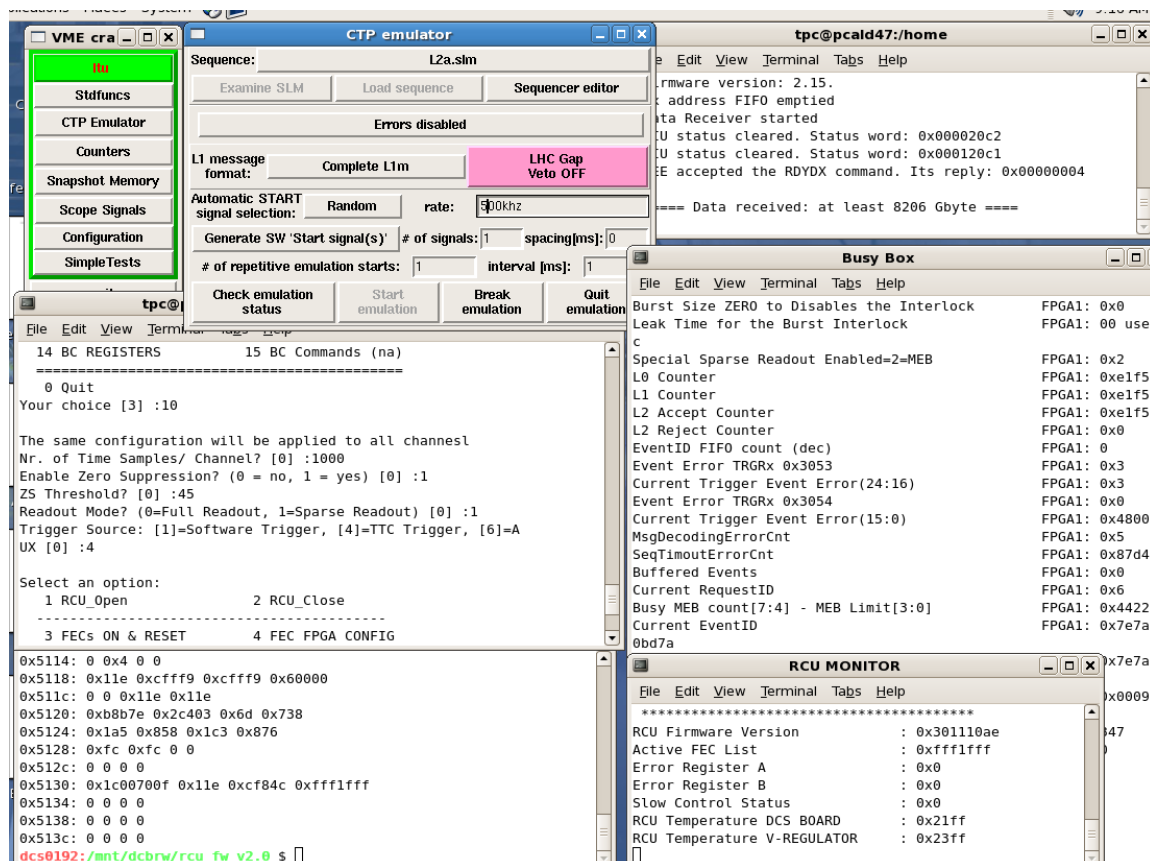
CDH Start: FFFFFFFF
Data : 020080EA MSW : 020 2ndMSW : 020 LSW : 0EA
Data : 009E32DC MSW : 009 2ndMSW : 38C LSW : 2DC
Data : 024CEDC7 MSW : 024 2ndMSW : 33B LSW : 1C7
Data : 010000EA MSW : 010 2ndMSW : 000 LSW : 0EA
Data : 00024CED MSW : 000 2ndMSW : 093 LSW : 0ED
Data : 00000000 MSW : 000 2ndMSW : 000 LSW : 000
Altro Trailer : 40100000 Ch. Add:000 Branch A:Block Length:010
Data : 01003C0E MSW : 010 2ndMSW : 00F LSW : 00E
Data : 00D0300B MSW : 00D 2ndMSW : 00C LSW : 00B
Data : 00A02408 MSW : 00A 2ndMSW : 009 LSW : 008
Data : 00701805 MSW : 007 2ndMSW : 006 LSW : 005
Data : 00400C02 MSW : 004 2ndMSW : 003 LSW : 002
Data : 00100000 MSW : 001 2ndMSW : 000 LSW : 000
Altro Trailer : 40100800 Ch. Add:800 Branch B:Block Length:010
Data : 01003C0E MSW : 010 2ndMSW : 00F LSW : 00E
Data : 00D0300B MSW : 00D 2ndMSW : 00C LSW : 00B
Data : 00A02408 MSW : 00A 2ndMSW : 009 LSW : 008
Data : 00701805 MSW : 007 2ndMSW : 006 LSW : 005
Data : 00400C02 MSW : 004 2ndMSW : 003 LSW : 002
Data : 00100000 MSW : 001 2ndMSW : 000 LSW : 000
Altro Trailer : 40100001 Ch. Add:001 Branch A:Block Length:010
Data : 01003C0E MSW : 010 2ndMSW : 00F LSW : 00E
Data : 00D0300B MSW : 00D 2ndMSW : 00C LSW : 00B
Data : 00A02408 MSW : 00A 2ndMSW : 009 LSW : 008
Data : 00701805 MSW : 007 2ndMSW : 006 LSW : 005
Data : 00400C02 MSW : 004 2ndMSW : 003 LSW : 002
Data : 00100000 MSW : 001 2ndMSW : 000 LSW : 000
Altro Trailer : 40100801 Ch. Add:801 Branch B:Block Length:010

```

Figure A.3: Decoded event data by the SIU stimuli process, received from the RCU firmware (being simulated)



(a) Start of the stress test in the test set-up



(b) End of the stress test in the test set-up

Figure A.4: Screen snap shot of the stress test with 4 buffers at high trigger rate

```

File Edit View Terminal Tabs Help
read BB... read RCU... done.
Event 2279/2324 (index=1707, partition=5, ddlID=891): read BB... load ALTRO...
read BB... read RCU... done.
Event 2280/2324 (index=1352, partition=5, ddlID=911): read BB... load ALTRO...
read BB... read RCU... done.
Event 2281/2324 (index=1703, partition=5, ddlID=875): read BB... load ALTRO...
read BB... read RCU... done.
Event 2282/2324 (index= 41, partition=5, ddlID=975): read BB... load ALTRO...
read BB... read RCU... done.
Event 2283/2324 (index=1699, partition=5, ddlID=883): read BB... load ALTRO...
x42a3: mismatch in channel payload
written data file: /local/tpc/raw1699/TPC_883.ddl
read data file: /tmp/test.dat
read BB... read RCU... done.
Event 2284/2324 (index= 38, partition=5, ddlID=871): read BB... load ALTRO... open DDL... trigger!... check event data...
read BB... read RCU... done.
Event 2285/2324 (index=2606, partition=5, ddlID=843): read BB... load ALTRO... open DDL... trigger!... check event data...
read BB... read RCU... done.
Event 2286/2324 (index=1698, partition=5, ddlID=863): read BB... load ALTRO... open DDL... trigger!... check event data...
read BB... read RCU... done.
Event 2287/2324 (index=1692, partition=5, ddlID=931): read BB... load ALTRO... open DDL... trigger!... check event data...
read BB... read RCU... done.
Event 2288/2324 (index=2609, partition=5, ddlID=911): read BB... load ALTRO... open DDL... trigger!... check event data...
read BB... read RCU... done.
Event 2289/2324 (index=1364, partition=5, ddlID=855): read BB... load ALTRO... open DDL... trigger!... check event data...
read BB... read RCU... done.
Event 2290/2324 (index=1688, partition=5, ddlID=919): read BB... load ALTRO... open DDL... trigger!... check event data...
read BB... read RCU... done.
Event 2291/2324 (index= 34, partition=5, ddlID=883): read BB... load ALTRO... open DDL... trigger!... check event data...
read BB... read RCU... done.
Event 2292/2324 (index=1655, partition=5, ddlID=919): read BB... load ALTRO... open DDL... trigger!... check event data...
read BB... read RCU... done.
Event 2293/2324 (index=1653, partition=5, ddlID=923): read BB... load ALTRO... open DDL... trigger!... check event data...
read BB... read RCU... done.
Event 2294/2324 (index=1637, partition=5, ddlID=907): read BB... load ALTRO... open DDL... trigger!... check event data...
read BB... read RCU... done.
Event 2295/2324 (index=1629, partition=5, ddlID=983): read BB... load ALTRO... open DDL... trigger!... check event data...
read BB... read RCU... done.
Event 2296/2324 (index=1416, partition=5, ddlID=859): read BB... load ALTRO... open DDL... trigger!... check event data...
read BB... read RCU... done.
Event 2297/2324 (index=1426, partition=5, ddlID=899): read BB... load ALTRO... open DDL... trigger!... check event data...
read BB... read RCU... done.
Event 2298/2324 (index=2620, partition=5, ddlID=899): read BB... load ALTRO... open DDL...

```

RCU Monitoring Registers

```

***** RCU Monitoring Registers*****
*****
RCU Firmware Version      : 0x30110ae
Active FEC List           : 0x3ff03ff
Error Register A          : 0x0
Error Register B          : 0x0
Slow Control Status       : 0x0
RCU Temperature DCS BOARD : 0x21ff
RCU Temperature V-REGULATOR : 0x22ff

```

(a) Example of a single data word mismatch found; data files are saved for debugging

```

File Edit View Terminal Tabs Help
Event 665/2324 (index=2082, partition=0, ddlID=796): read BB... load ALTRO... open DDL...
Event 666/2324 (index= 785, partition=0, ddlID=818): read BB... load ALTRO... open DDL...
Event 667/2324 (index=2061, partition=0, ddlID=786): read BB... load ALTRO... open DDL...
el payload
read BB... read RCU... done.
Event 668/2324 (index=2075, partition=0, ddlID=770): read BB... load ALTRO... open DDL...
Event 669/2324 (index=2074, partition=0, ddlID=800): read BB... load ALTRO... open DDL...
Event 670/2324 (index= 770, partition=0, ddlID=772): read BB... load ALTRO... open DDL...
Event 671/2324 (index= 777, partition=0, ddlID=768): read BB... load ALTRO... open DDL...
Event 672/2324 (index=2065, partition=0, ddlID=780): read BB... load ALTRO... open DDL...
Event 673/2324 (index=1714, partition=1, ddlID=781): switching FECs... load ROLM... read
ad BB... read RCU... done.
Event 674/2324 (index= 774, partition=1, ddlID=819): read BB... load ALTRO... open DDL...
Event 675/2324 (index= 775, partition=1, ddlID=833): read BB... load ALTRO... open DDL...
Event 676/2324 (index=2062, partition=1, ddlID=823): read BB... load ALTRO... open DDL... trigger!... check event data... read BB... read RCU... done.
Event 677/2324 (index=2070, partition=1, ddlID=817): read BB... load ALTRO... open DDL... trigger!... check event data... read BB... read RCU... done.
Event 678/2324 (index= 780, partition=1, ddlID=793): read BB... load ALTRO... open DDL... trigger!... check event data... read BB... read RCU... done.
Event 679/2324 (index=2078, partition=1, ddlID=825): read BB... load ALTRO... open DDL... trigger!... check event data... read BB... read RCU... done.
Event 680/2324 (index=2058, partition=1, ddlID=823): read BB... load ALTRO... open DDL... trigger!... check event data... read BB... read RCU... done.
Event 681/2324 (index= 784, partition=1, ddlID=805): read BB... load ALTRO... open DDL... trigger!... check event data... read BB... read RCU... done.
Event 682/2324 (index=2081, partition=1, ddlID=781): read BB... load ALTRO... open DDL... trigger!... check event data... read BB... read RCU... done.
Event 683/2324 (index= 764, partition=1, ddlID=791): read BB... load ALTRO... open DDL... trigger!... check event data... read BB... read RCU... done.
Event 684/2324 (index=2083, partition=1, ddlID=817): read BB... load ALTRO... open DDL... trigger!... check event data... read BB... read RCU... done.
Event 685/2324 (index=2055, partition=1, ddlID=839): read BB... load ALTRO... open DDL... trigger!... check event data... read BB... read RCU... done.
Event 686/2324 (index= 762, partition=1, ddlID=775): read BB... load ALTRO... open DDL... trigger!... check event data... read BB... read RCU... done.
Event 687/2324 (index= 760, partition=1, ddlID=823): read BB... load ALTRO... open DDL... trigger!... check event data... read BB... read RCU... done.
Event 688/2324 (index= 791, partition=1, ddlID=797): read BB... load ALTRO... open DDL... trigger!... check event data... read BB... read RCU... done.
Event 689/2324 (index= 792, partition=1, ddlID=777): read BB... load ALTRO... open DDL... trigger!... check event data... read BB... read RCU... done.
Event 690/2324 (index=2084, partition=1, ddlID=779): read BB... load ALTRO... open DDL... trigger!... check event data... read BB... read RCU... done.
Event 691/2324 (index= 758, partition=1, ddlID=801): read BB... load ALTRO... open DDL... trigger!... check event data... read BB... read RCU... done.
Event 692/2324 (index=2087, partition=1, ddlID=813): read BB... load ALTRO... open DDL... trigger!... check event data... read BB... read RCU... done.
Event 693/2324 (index=2090, partition=1, ddlID=815): read BB... load ALTRO... open DDL... trigger!... check event data... read BB... read RCU... done.
Event 694/2324 (index= 754, partition=1, ddlID=823): read BB... load ALTRO... open DDL... trigger!... check event data... read BB... read RCU... done.
Event 695/2324 (index=2091, partition=1, ddlID=795): read BB... load ALTRO... open DDL... trigger!... check event data... read BB... read RCU... done.
Event 696/2324 (index=2092, partition=1, ddlID=839): read BB... load ALTRO... open DDL... trigger!... check event data... read BB... read RCU... done.
Event 697/2324 (index= 802, partition=1, ddlID=819): read BB... load ALTRO... open DDL... trigger!... check event data... read BB... read RCU... done.
Event 698/2324 (index= 803, partition=1, ddlID=823): read BB... load ALTRO... open DDL... trigger!... check event data... read BB... read RCU... done.
Event 699/2324 (index= 751, partition=1, ddlID=793): read BB... load ALTRO... open DDL... trigger!... check event data... read BB... read RCU... done.
Event 700/2324 (index= 750, partition=1, ddlID=839): read BB... load ALTRO... open DDL... trigger!... check event data... read BB... read RCU... done.
Event 701/2324 (index= 749, partition=1, ddlID=777): read BB... load ALTRO... open DDL... trigger!... check event data... read BB... read RCU... done.
Event 702/2324 (index= 807, partition=1, ddlID=797): read BB... load ALTRO... open DDL... trigger!... check event data... read BB... read RCU... done.
Event 703/2324 (index= 808, partition=1, ddlID=835): read BB... load ALTRO... open DDL... trigger!... check event data... read BB... read RCU... done.
Event 704/2324 (index=2093, partition=1, ddlID=799): read BB... load ALTRO... open DDL... trigger!... check event data... read BB... read RCU... done.
Event 705/2324 (index= 811, partition=1, ddlID=839): read BB... load ALTRO... open DDL... trigger!... check event data... read BB... read RCU... done.
Event 706/2324 (index=2045, partition=1, ddlID=813): read BB... load ALTRO... open DDL... trigger!... check event data... read BB... read RCU... done.
Event 707/2324 (index= 747, partition=1, ddlID=813): read BB... load ALTRO... open DDL... trigger!... check event data... read BB... read RCU... done.
Event 708/2324 (index= 743, partition=1, ddlID=795): read BB... load ALTRO... open DDL... trigger!... check event data... read BB... read RCU... done.
Event 709/2324 (index= 815, partition=1, ddlID=797): read BB... load ALTRO... open DDL... trigger!... check event data... read BB... read RCU... done.
Event 710/2324 (index=2038, partition=1, ddlID=771): read BB... load ALTRO...

```

RCU Monitoring Registers

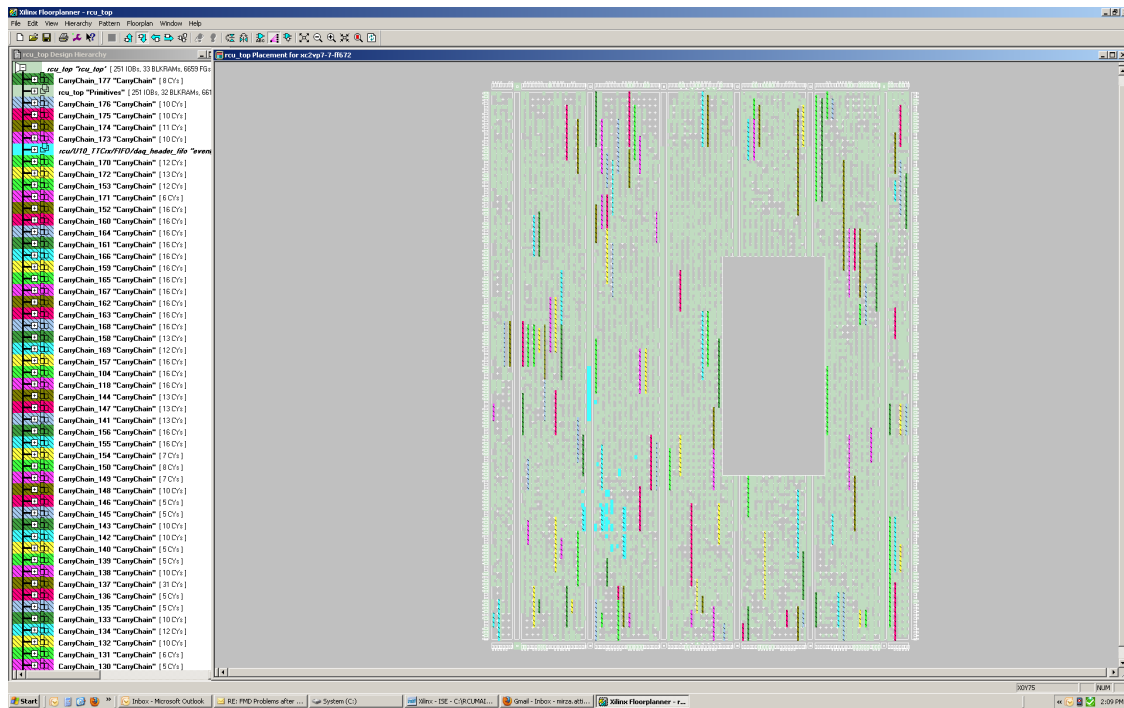
```

***** RCU Monitoring Registers*****
*****
RCU Firmware Version      : 0x30918ae
Active FEC List           : 0xfffffff
Error Register A          : 0x0
Error Register B          : 0x0
RCU Temperature DCS BOARD : 0x1dff
RCU Temperature V-REGULATOR : 0x1eff

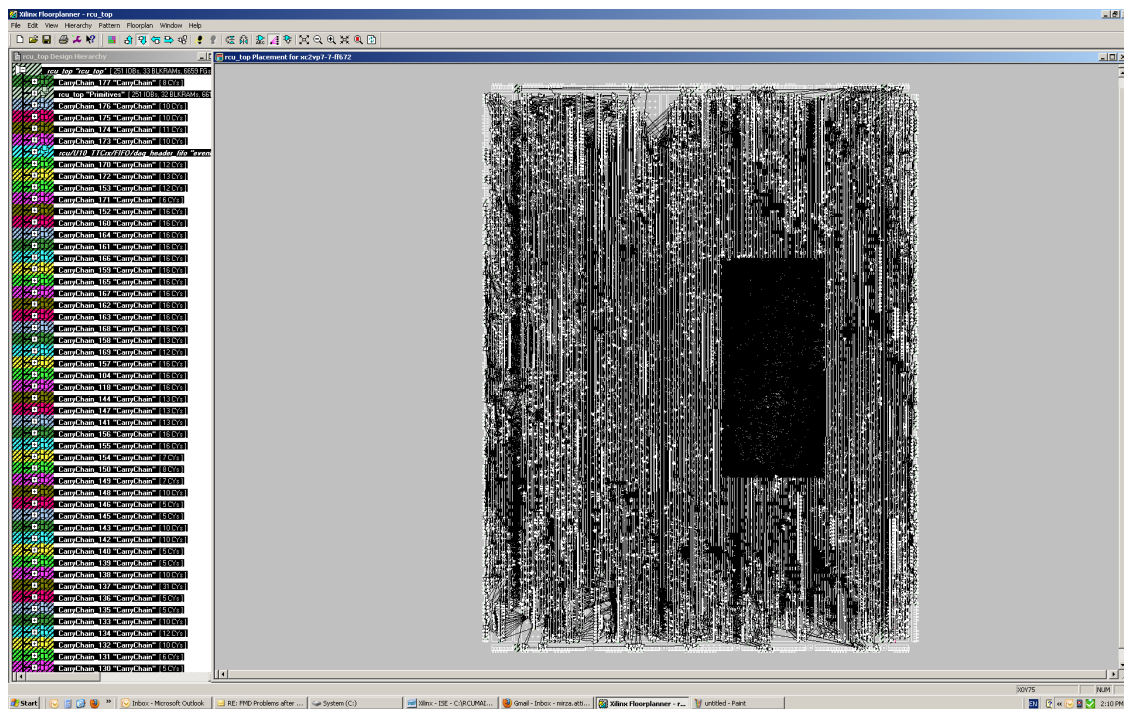
```

(b) Automatic transition in test from partition 0 to partition 1

Figure A.5: Bit-wise event data verification of the RCU firmware using collisions data for all six readout partitions



(a) Top level device view before the final implementation



(b) Top level device view after the final implementation

Figure A.6: Xilinx VP2 device utilization after final implementation of RCU firmware

Appendix B

Number of ch. Hits	FRDO P.L. Size (Bytes)	SRDO P.L. Size (Bytes)	FRDO Time (msec)	SRDO Time (msec)
3200	956800	956800	24.44	24.5496
2944	880512	880256	22.57	22.6008
2688	804224	803712	20.69	20.652
2432	727936	727168	18.82	18.7032
2176	651648	650624	16.94	16.7544
1920	575360	574080	15.07	14.8056
1676	500187	498663	13.22	12.89025
1484	442058	440342	11.79	11.42365
1228	365770	363798	9.92	9.471675
960	285830	283590	7.96	7.42525
702	209340	206842	6.08	5.466925
442	132652	129894	4.19	3.50345
186	56364	53350	2.32	1.569475
48	15886	12734	1.32	0.4983
0	3200	0	0.99	0.14525

Table B.1: Comparison of readout time for sparse and full readout mode with increasing number of channel hits

Nsamp	Event Size	Total Time	BR-A-Busy	BR-B-Busy	FEC-RCU	RCU-SIU
10	64068	1.12	1.08	0.99	1.11	1.12
20	115268	1.24	1.20	1.11	1.24	1.24
30	153668	1.40	1.28	1.19	1.40	1.40
40	192068	1.72	1.41	1.30	1.72	1.72
50	243268	1.96	1.49	1.38	1.95	1.96
60	281668	2.28	1.62	1.49	2.27	2.28
70	320068	2.44	1.70	1.57	2.43	2.44
80	371268	2.84	1.83	1.68	2.82	2.84
90	409668	3.00	1.91	1.76	2.98	3.00
100	448068	3.32	2.03	1.88	3.30	3.32
110	499268	3.56	2.12	1.95	3.54	3.56
120	537668	3.88	2.24	2.07	3.85	3.88
130	576068	4.04	2.32	2.15	4.41	4.04
140	627268	4.44	2.45	2.26	4.72	4.44
150	665668	4.60	2.53	2.34	4.57	4.60
160	704068	4.92	2.66	2.45	4.88	4.92
170	755268	5.16	2.74	2.53	5.12	5.16
180	793668	5.48	2.87	2.64	5.43	5.48
190	832068	5.64	2.95	2.72	5.59	5.64
200	883268	6.04	3.07	2.84	5.91	6.04
250	1088068	7.24	3.57	3.30	6.87	7.24
300	1305668	8.68	4.11	3.80	7.91	8.68
350	1523268	9.96	4.61	4.26	8.87	9.96
400	1728068	11.32	4.76	5.15	9.91	11.32
450	1945668	12.60	5.65	5.22	10.87	12.60
500	2163268	14.04	6.19	5.72	11.92	14.04
550	2368068	15.24	6.69	6.18	12.87	15.24
600	2585668	16.68	7.23	6.68	13.91	16.68
650	2803268	17.96	7.73	8.16	14.87	17.96
700	3008068	19.32	8.27	7.64	15.91	19.32
750	3225668	20.60	8.77	8.10	16.86	20.60
800	3443268	22.04	8.60	9.31	17.90	22.04
850	3648068	23.24	9.81	9.06	18.88	23.24
900	3865668	24.68	10.35	9.56	19.90	24.68
950	4083268	25.96	10.85	10.02	20.86	25.96
1000	4288068	27.32	11.39	10.52	21.90	27.32

Table B.2: Full readout time with number of samples and event size

List of Abbreviations

ADC : Analogue-Digital Converter
ALTRO : ALice Tpc Read-Out
ALICE : A Large Ion Collider Experiment
ASIC : Application-Specific Integrated Circuit
ATLAS : A Toroidal Lhc ApparatuS
ACORDE : Alice COsmic Ray DEtector
ACKN : Acknowledge
BB : BusyBox
BRAM : Block RAM
BC : Board Controller
CERN : Conseil Européenne pour la Recherche Nucléaire
CTP : Central Trigger Processor
CSTB : Command Strobe
CE : Central Electrode
CMS : Compact Muon Solenoid
DAQ : Data AcQuisition
DSTB : Data Strobe
DCS : Detector Control System
DIU : Destination Interface Unit
DDL : Detector Data Link
DAC : Digital to Analogue Converter
DRORC : Destination Read-Out Receiver Card
DCAL : Di-jet CALorimeter
EMCAL : Electro-Magnetic CALorimeter
ECS : Experiment Control System
FEE : Front-End Electronics
FEC : Front-End Card
FMD : Forward Multiplicity Detector
FPGA : Field Programmable Gate Array
FC : Front-end Control
GTL : Gunning Transceiver Logic
GUI : Graphical User Interface
GDC : Global Data Concentrator
HLT : High Level Trigger
I²C : Inter-Integrated Circuit
IROC : Inner Read-Out Chamber
ITS : Inner Tracking System
IM : Instruction Memory
LDC : Local Data Concentrator

LHCb : LHC-Beauty
HMPID : High Momentum Particle Identification Detector
L : Level
LHC : Large Hadron Collider
MSM : Monitoring and Safety Module
MUON : MUON spectrometer
MWPC : Multi-Wire Proportional Chamber
OROC : Outer Read-Out Chamber
PHOS : PHOton Spectrometer
PVSS : ProzessVisualisierungs- und Steuerungs-System
PASA : PreAmplifier ShAper
PMD : Photon Multiplicity Detector
PID : Particle IDentification
QGP : Quark Gluon Plasma
RCU : Readout Control Unit
RM : Result Memory
ROC : Read-Out Chamber
RAM : Random Access Memory
RP : Readout Partition
SIU : Source Interface Unit
SRAM : Static RAM
SCADA : Supervisory Control And Data Acquisition
SC : Slow-Control
SPD : Silicon Pixel Detector
SDD : Silicon Drift Detector
SSD : Silicon Strip Detector
TDR : Technical Design Report
TPC : Time Projection Chamber
TRD : Transition Radiation Detector
TRSF : Transfer
TTCRX : TTC Receiver
TTC : Timing, Trigger and Control
TOF : Time Of Flight
VHDL : VHSIC (Very High Speed Integrated Circuit) Hardware Description Language
ZDC : Zero Degree Calorimeter

Bibliography

- [1] ALICE-COLLABORATION. *Technical Proposal for A Large Ion Collider Experiment at the CERN LHC*. CERN, December 1995.
- [2] ALICE-COLLABORATION. *ALICE Technical Design Report of the Time Projection Chamber*. CERN, January 2000.
- [3] ALICE-COLLABORATION. *ALICE Technical Design Report of the Trigger, Data Acquisition, High-Level Trigger and Control System*. CERN, January 2004.
- [4] ALICE-TPC. *ALICE Electronic Log Book*¹. ALICE, August 2012.
- [5] ALME, J. *A Trigger Based Readout and Control System operating in a Radiation Environment*. PhD thesis, University of Bergen, Bergen, 2008.
- [6] ALME, J., ET AL. *The ALICE TPC, a large 3-dimensional tracking device with fast readout for ultra-high multiplicity events*, 2010.
- [7] ALME, J., ET AL. *Radiation Tolerance Studies using Fault Injection on the Readout Control FPGA Design of the ALICE TPC Detector*, 2012.
- [8] BRAMM, R. *Characterization of ALICE TPC Readout Chip*. PhD thesis, University of Frankfurt, Frankfurt, 2005.
- [9] CARENA, F., ET AL. *Start of data and end of data events in ALICE*. ALICE, September 2005.
- [10] CARRO, L., ET AL. *Fault-Tolerance Techniques for Sram-Based FPGAs*. Springer, Frontiers in electronic testing, Netherland, 2006.
- [11] COSTA, F. *Private communication*. CERN, April 2012.
- [12] DÉNES, E., ET AL. *Radiation Tolerant Source Interface Unit for the ALICE Experiment*, 2005.
- [13] FORMENTI, F., ET AL. *Trigger error handling and reporting*. ALICE, November 2006.
- [14] GONZÁLEZ-GUTIÉRREZ, C., ET AL. *The ALICE TPC readout control unit*, 2005.
- [15] JANSSEN, K. Nic brochure online: Elementary particle physics, March 2005.
- [16] LARSEN, D. T. *Monitoring and calibration of the ALICE time projection chamber*. PhD thesis, University of Bergen, Bergen, 2009.

¹Restricted access

- [17] MAGER, M., MUSA, L., REHMAN, A., AND SZCZEPANKIEWICZ, A. Measurement of single event upsets in the ALICE-TPC front-end electronics.
- [18] MUNKEJORD, M. *Development of the ALICE Busy Box*. Bergen, 2007.
- [19] MUSA, L. *ALICE TPC Readout chip User Manual*. CERN, June 2002.
- [20] MUSA, L., ET AL. *The ALICE TPC front end electronics*, 2003.
- [21] P. MOREIRA, J. CHRISTIANSEN, A. M., AND TOIFL, T. *TTCrx Reference Manual*. CERN, October 2004.
- [22] REHMAN, A., AND COSTA, F. *TPC configuration of TPC during run time*. ALICE TPC weekly meeting, September 2011.
- [23] REHMAN, A., AND MUSA, L. *RCU Firmware User Manual*. CERN, April 2008.
- [24] RENFORDT, R., AND REHMAN, A. *Gating Grid Pulser Firmware User Manual*. CERN, January 2009.
- [25] RENFORDT, R., AND REHMAN, A. *Calibration Pulser User Manual*. CERN, January 2010.
- [26] RØED, K. *Single Event Upsets in SRAM FPGA based readout electronics for the Time Projection Chamber in the ALICE experiment*. PhD thesis, University of Bergen, Bergen, 2009.
- [27] ROED, K., ALME, J., FEHLKER, D., LIPPMANN, C., AND REHMAN, A. First measurement of single event upsets in the readout control FPGA of the ALICE TPC detector. *JINST* 6 (2011), C12022.
- [28] ROSSEGGER, S. *Simulation & Calibration of the ALICE TPC including innovative Space Charge Calculations*. PhD thesis, Graz University of Technology, Graz, 2009.
- [29] ROSSEGGER, S. *Private communication*. CERN, June 2011.
- [30] SØGAARD, C. *Measurement of Forward-Backward Charged Particle Correlations with ALICE*. PhD thesis, Niels Bohr Institute, University of Copenhagen, 2012.
- [31] SOLTVEIT, H., STACHEL, J., BRAUN-MUNZINGER, P., MUSA, L., GUSTAFSSON, H., ET AL. The Preamplifier shaper for the ALICE TPC-Detector. *Nucl.Instrum.Meth.* A676 (2012), 106–119.
- [32] SOOS, C., AND RUBIN, G. *ALICE DDL - Hardware Guide for the front-end Designers*. CERN, January 1998.
- [33] TEXAS INSTRUMENTS INC. *GTL/BTL: A Low-Swing Solution for High Speed Digital Logic*, 1997.
- [34] XILINX. *Virtex-II and Virtex-II Pro X Platform FPGAs: Complete Data Sheet*, 4.2 ed. Xilinx INC., November 2007.
- [35] XILINX. *Virtex-II and Virtex-II Pro X Platform FPGAs: User Guide*. Xilinx INC., November 2007.
- [36] XILINX. *Digital Clock Manager (DCM) Module*. Xilinx INC., April 2009.