User Logic Development for The Muon Identifier's Common Readout Unit for the ALICE Experiment at The Large Hadron Collider

Prepared by:

Nathan Boyles

BYLNAT001

Supervised by:

Prof. Zinhle Buthelezi

Department of Subatomic Physics, iThemba LABS

Dr. Simon Winberg

Department of Electrical Engineering, UCT

Prof. Amit Mishra

Department of Electrical Engineering, UCT

January 2020

A dissertation submitted to the Department of Electrical Engineering,

University of Cape Town,

in partial fulfilment of the requirements

for the degree of

Master of Science in Electrical Engineering

Declaration

1. I know that plagiarism is wrong. Plagiarism is to use another's work and pretend that it is one's own.
2. I have used the IEEE convention for citation and referencing. Each contribution to, and quotation in, this project report from the work(s) of other people, has been attributed and has been cited and referenced.
3. This project report is my own work.
4. I have not allowed, and will not allow, anyone to copy my work with the intention of passing it off as their own work or part thereof.

Signature of Author ..... Signed by candidate .....

Cape Town
1 January 2020

Abstract

The Large Hadron Collider at CERN is set to undergo major upgrades starting in 2019 resulting in expected centre of mass energy for proton-proton collisions to be the nominal 14 TeV. In light of these upgrades the experiments, namely, ALICE, ATLAS, CMS and LHCB, are required to upgrade their detectors correspondingly. The work contained in this dissertation pertains to the upgrade of the ALICE detector and in particular to the Muon Trigger(MTR) Detector which will be renamed the Muon Identifier (MID). This detector has historically operated in triggered readout manner exchanging trigger signals with the Central Trigger Processor (CTP) when events of interest occur using a minimum bias. The upgrades include a transition from triggered readout to continuous readout time delimiting data payloads using periodic heartbeat signals. Continuous readout however results in data rates several orders of magnitude higher than previous operstion and as such would require vast storage resources for raw data thus a new computing system known as O2 is also being developed for real-time data reduction.

Part of the system used to perform real-time data reduction is based on FPGA technology and is known as the Common Readout Unit. As its name implies, the CRU is common to many detectors regardless of their differences in design. As such, each detector requires customer logic to meet their unique requirements, known as the user logic. This project concerns development of the ALICE MID user logic which will interface to the Core CRU firmware and perform real-time data extraction, reformatting, zero suppression, data synchronization and transmission of the processed data to the Core CRU firmware. It presents the development of a conceptual design and a prototype for the user logic of the ALICE MID. The research methodology employed involved the identification of relevant documentation as well as in-depth meetings with the developers of the periphery systems to ascertain requirements and constraints of the project. The resulting prototype shows the ability to meet the established requirements in effective and optimized manner. Additionally, the modular design approach employed, allows for more features to be easily introduced.

Acknowledgements

Firstly, I'd like to thank iThemba LABS and the SA-CERN Consortium for the financial and administrative support during this project. In particular, the IT department at iThemba LABS were always willing to assist when requested.

My supervisors and mentors, Prof. Zinhle Buthelezi, Prof. Amit Mishra and Dr. Simon Winberg were amazing and showed incredible patience when working giving me freedom to conduct my research in a way that felt natural to me. With their guidance, I was able to overcome many insecurities allowing great personal growth during this period.

As this project is a small cog of a much larger and complex project, a large amount of collegial work was necessary for both initial research as well as ongoing design and development. That been said, the MID-Team was an invaluable sounding board for ideas and development. My South African colleague Sehlabaka Qhobosheane was at my side while we navigated this complex task as well as colleagues from Subatech (Nantes, France) and CERN (Geneva, Switzerland). More specifically, Christophe Renard, Jean-Luc Beney, Dr. Diego Stocco, Dr. Pascal Dupieux provided invaluable guidance on catching up with previous developments on the MID project as well as the necessary information during the project lifespan and I'm tremendously grateful for the patience when working with a young engineer just trying figure things out. The Core CRU team at CERN was always ready to provide support from the very start of this project.

A big shout out to all my friends who held my hand and dealt with all my craziness during my MSc. it was daunting and incredibly reqarding experience for all of us as we worked on our various research. My family although in another city kept me grounded and helped me whenever I needed it.

Finally, I want thank my partner in crime and life Dr. Sariu Ali Didi who gave me the courage to confront a whole new world with tenacity and loved and supported me through the best and worst of it. What a journey for both of us.

Contents

List of Figures

List of Tables

List of Abbreviations

ADULT	—	A DUal Threshold
ALICE	—	A Large Ion Collider Experiment
ASIC	—	Application Specific Integrated Circuit
BC	—	Bunch Crossing
BCID	—	Bunch Counter Identifier
BP	—	Bending Plane
COTS	—	Commercial Off-The-Shelf
CRU	—	Common Readout Unit
CTS	—	Central Trigger System
DAQ	—	Data Acquisition
DCS	—	Detector Control System
DSP	—	Digital Signal Processor
DUT	—	Device Under Test
EPN	—	Event Processing Node
FEE	—	Front-End Electronics
FIFO	—	First-In-First-Out
FLP	—	First Level Processor
FPGA	—	Field Programmable Gate Array
GBT	—	Gigabit Transceiver
GBTx	—	Gigabit Transceiver ASIC
HB	—	Heart Beat
HBF	—	Heart Beat Frame
HBID	—	Heart Beat Identifier
HEP	—	High Energy Physics
IBC	—	Internal Bunch Counter
I2C	—	Inter-Integrated Circuit
LHC	—	Large Hadron Collider
LS2	—	Long Shutdown 2
LTU	—	Local Trigger Unit

LVDS	—	Low-voltage Differential Signal
MCH	—	Muon Tracking Chamber
MFT	—	Muon Forward Tracker
MID	—	Muon Identifier
MTR	—	Muon Trigger
MUX	—	Data Multiplexer
NBP	—	Non-Bending Plane
O^2	—	Online and Offline Computing System
Pb-Pb	—	Lead-Lead
PCIe	—	Peripheral Component Interconnect Express
PLL	—	Phase-Locked Loops
p-Pb	—	Proton-Lead
pp	—	Proton-Proton
QGP	—	Quark-Gluon Plasma
RDH	—	Raw Data Header
RO	—	Read-out
RPC	—	Resistive Plate Chambers
RTL	—	Register Transfer Logic
Run1	—	ALICE operation since the LHC came online in 2009
Run2	—	ALICE operation between between 2015 and 2018 at the LHC
Run3	—	ALICE operation post the LHC Long Shutdown 2
Run4	—	ALICE operation after the LHC Long Shutdown 2 onward
SCA	—	Slow Control Adapter
SDH	—	Standard Data Header
SEU	—	Single Event Upset
TTC	—	Timing, Trigger and Control System
TTS	—	Trigger and Timing Distribution System
VFE	—	Very Front-End Electronics
VME	—	Versa Module Europa
VTRx	—	Versatile Transceiver
VTTx	—	Versatile Twin-Transmitter
ZS	—	Zero Suppression

Nomenclature

Avalanche Mode — the electric field across the gap (and consequently the gas amplification) is reduced and a robust signal amplification is introduced at the front-end level

e-link — Electrical Link

Handshaking — Communication between two nodes that establishes the parameters for continued communication

Injection Testing – Luminosity — The ratio of the number of events detected (N) in a certain time (t) to the interaction cross-section (σ) .It is expressed in units of inverse centimeter square (cm^{-2}) per second (s^{-1}) [9].

Momentum Cut — A threshold at which data acquisition is triggered in the MID for for each Muon passing through the detector

Muon — Subatomic particle belonging to the family of leptons along with electrons and tau particles [9].

Pipelining — A design methodology allowing for the parralization of tasks.

Tracklet — A tracklet is a bit that indicates whether a local card has detected a segment of a track between it's BP and NBP plane. Similarly the tracklet bit for the regional card indicates whether its corresponding card has detected this track.

Streamer Mode — the electric field inside the gap is kept intense enough to generate limited discharges localized near the crossing of the ionizing particle

Chapter 1

Introduction

ALICE [10] is one of the four major experiments on the LHC ring, 27 km in circumference and approximately 100 m underground in both France and Switzerland [11]. It went online in 2009 after over 20 years in development. Its purpose is to advance our understanding of the universe by colliding hadrons together and studying their constituents. The LHC collides particles at unprecedented energies to the order of tera electron volts (TeV) - temperatures equivalent to 10^{12} K, which is about 100 000 times hotter than the core of the Sun. The work done at the LHC is multi-disciplinary and multi-national collaborations of which South Africa participates. The work described in this dissertation pertains to the development of the user logic of the ALICE Muon Identifier (MID) CRU. The MID is the proposed future designation of the ALICE Muon Trigger (MTR) System after LS2. The institute involved in this project is iThemba Laboratory for Accelerator Based Science (LABS).

This dissertation focuses on the development of logic for the Muon Identifier (MID) of the ALICE detector at the CERN LHC [12]. This logic is a custom component specific to the Muon Identifier to performing data processing functions and forms part of the upgrades to the ALICE detector in light of the LHC upgrade to increase the luminosity as more statistics are needed for precision measurements by the experiments. The ALICE collaboration is planning a major upgrade of its detectors during the Long Shutdown 2 (LS2), scheduled to start at the end of 2018. While some systems will be replaced, most other detectors, including the Muon Spectrometer (Tracking and Trigger systems) will receive new front-end readout electronics.

The ALICE detector has historically operated in a triggered manner when physics events of interest are detected at the FEE. This method of data acquisition - using physics triggers, will become very inefficient in view of the upgraded ALICE detector. The upgrade strategy is based on collecting Pb-Pb collisions at luminosities up to $6\text{x}10^{27}\text{cm}^{-2}\text{s}^{-1}$, corresponding to

collision rates of 50 kHz, where collision data will be transferred to the O2 system either upon a minimum bias trigger or in a self-triggered, continuous way[13]. The upgraded detectors will be self-triggered and operate in a continuous readout mode making use of heartbeat triggers at fixed intervals in time [13]. Furthermore, the entire data acquisition chain has to be improved to accommodate this new readout method. Most notably, a new initial real-time data-processing unit based on FPGA technology called the Common Readout Unit (CRU) will be implemented. The CRU will be common to all detectors and is the central interface between The Front-End System, Central Trigger System and the O2 System.

The CRU is being developed for detector data readout, concentration and multiplexing onto the O2 system for event reconstruction and storage, as well as distributing trigger and timing information to the on-detector electronics. Since it will be common to all detectors only core firmware has been produced [4] which all detectors will make use of but the specific user logic for each detector needs to be implemented separately by the relevant detector groups.

This chapter will outline the relevant background information, followed by a description of the problem at hand. Furthermore, the project deliverables to address this problem are defined. The questions being addressed by this work are then clarified and it's scope and limitations are then established .

1.1 Background

The ALICE detector is located in an underground cavern of the 27 km LHC accelerator. It is a dedicated heavy-ion detector (shown in figure 1.1) designed to study the physics of a primordial strongly interacting matter believed to have existed in the early Universe, a few microseconds after the Big Bang known as the quark-gluon plasma (QGP) in Pb-Pb collisions.Additionally, pp and p-Pb collisions are also studied for reference [14].

ALICE physics is achieved through a comprehensive study of hadrons, electrons, muons and photons produced in the above mentioned collisions, up to the highest energies provided at the LHC. It is composed of central barrel detectors (encapsulated in the red-coloured L3 solenoid magnet) as well as the muon spectrometer situated in the forward kinematic region. The muon spectrometer is comprised of the front absorber, 5 tracking stations, an iron filter and 2 muon trigger stations[14]. The work presented in this dissertation focuses on the Muon Trigger System.

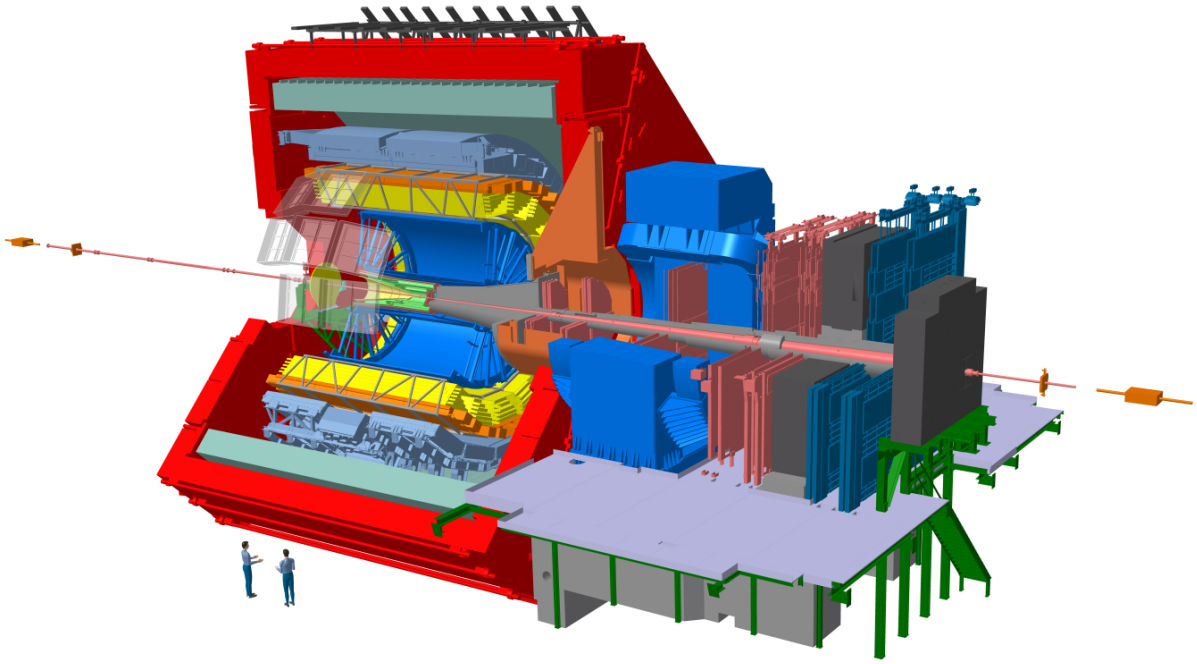


Figure 1.1: The ALICE detector[1]

1.1.1 The Muon Trigger System

The muon trigger system is part of the muon spectrometer, whose physics aim is to study single and di-muon events produced in the decay of various particles (heavy quarks, low-mass vector mesons and electroweak bosons). The MTR was designed for muon identification and to select single and di-muons with transverse momentum above a programmable threshold.

In the MTR, a momentum cut is applied on each individual muon at the trigger level. The purpose is to reduce the probability of triggering on events where unwanted low momentum muons are not accompanied by high momentum muons from the decay of particles of interest. Two momentum cuts (low and high), are performed in parallel by the trigger electronics[12].

The values of the programmable momentum thresholds can vary between 0.5 to 4 GeV/c. The ALICE Central Trigger Processor (CTP) recieves six trigger signals less than 800 ns after the interaction, at 400 MHz. These trigger signals are as follows[12]:

- At least one single muon track above low- (high-) momentum cut
- At least two unlike-sign muon tracks, each of them above low- (high-) momentum cut
- At least two like-sign muon tracks, each of them above low- (high-) momentum cut

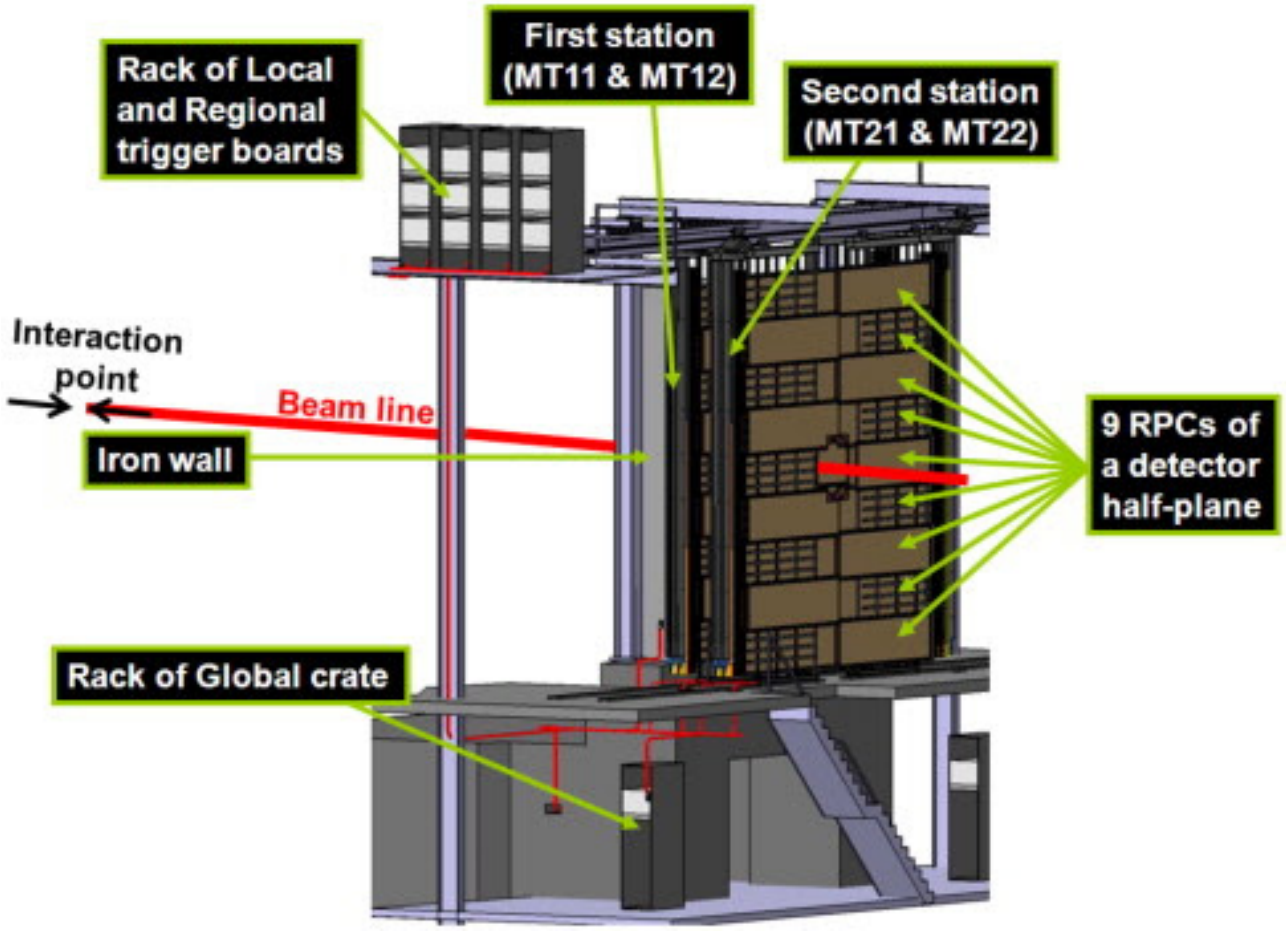


Figure 1.2: MTR System

The MTR consists of four planes of of Resistive Plate Chambers (RPC) arranged in two stations one metre apart. A single plane is comprised of 18 RPC's operating in streamer-mode. An RPC is typically 70x300 cm^2 in size resulting in an active area of 140 m^2. Momentum selection is performed using a position-sensitive trigger detector with spatial resolution less than 1 cm. Cartesian coordinates of RPC hits are determined using segmented strips with pitch and length increasing with their distance from the beam axis.

The RPC's are an air-tight parallel-plate detector with a single gap of 2 mm. The plates are made of low resistivity bakelite($\sim 10^9 \Omega$m) and are filled with a gas mixture composed of 89.7% $C_2H_2F_4$, 10% C_4H_{10} and 0.3% SF_6[15]. The gas relative humidity is kept at 37 % to prevent variations in the bakelite resistivity[15]. The high operating voltages are optimized for each chamber and range from 10.0 to 10.4 kV. The RPC's are equipped with dual-threshold front-end discriminators which have been tailored to the timing properties of the detector and reach the necessary time resolution (1–2 ns) for the identification of the bunch crossing (BC).

The discriminators then send signals to the trigger electronics working in a pipeline mode at 40 MHz. The trigger electronics are arranged hierarchically: local, regional and global. The trigger algorithm searches for a single track using the information of the four RPC's planes that approximately originates from the primary interaction vertex. Additionally, hits in a least 3 of the 4 detector planes in both the bending and non-bending(referring to the axis along which the muons bend due to the presence of the magnetic field in the muon spectrometer) plane are required to define a track. The regional and global levels gather all the signals from the local boards and deliver the single-muon, like-sign and unlike-sign dimuon triggers of the entire detector.

1.2 Problem Description

The transition from a triggered readout mechanism of the MTR to a continuous readout of the MID presents a few a problems. These problems concern the interfacing of the user logic to the Core CRU FW, timing-related issues introduced by higher data throughput and restructuring the data efficiently and in the required formats for the the upstream nodes.

Data being sent from the front-end electronics contains data identifying the local card relative to the regional crate. This data does not indicate a unique location within the MID from which this data originates and thus is in an inappropriate format for use by the O2 system for reconstruction.

The data acquired from all the front-end electronics occurs simultaneously, at fixed periods and is transmitted along many electrical and optical links. However, varying transmission delays will result in the data from the various links arriving asynchronously at the CRU.

While the data payload received from the Front-End system provides the timing information of data acquisition this time period is a sub-period of the period defined in the timing and trigger system. As such, the data arriving at the CRU has no identifying information about the time at which the data was acquired with reference to the high-level timing and trigger system.

Zero suppression in the FES system results in zero's at the CRU interface since the system is streaming data. In particular, planes not transmitting data are indicated in a field in the local card payload. Futhermore, if all the planes interfaced to a local card have no data to trasmit, no data payload is sent from that card. This is indicated in a field in the data payload of the regional link interfaced to that card. Both these zero producing scenarios must sup-

pressed in the final readout.

Data from all the FEE are sent in parallel to the CRU but the O2 system requires fixed packets of data from all the front-end electronics for a specific period in time and arranged in a specific order.

1.3 Focus

The work produced during this MSc focuses on the rudimentary design and development of a prototype of the User Logic for the ALICE MID as well as a testing framework for verification and future development.

The prototype introduced covers top priority requirements from the data acquisition chain of the MID detector, namely:

- Interfacing to the downlink ports of data wrapper provided by the Core CRU firmware for integration
- The identification of valid transmitted data and subsequent extraction of this data
- Reformatting the data payload transmitted from the front-end system into the required format required by the O2 system.
- Synchronization of the data payload essential for performing correct analysis
- Transmission of the data in the order required by the O2 system

The testing methodology developed encompasses three categories which would enable thorough validation and verification in all development stages of the user logic just short of actual testing performed on the physical detector. Tests include simulation-based verification in an HDL test-bench with developed user logic components, partial in-system testing in a hardware testbench comprised of hardware prototypes of the broader DAQ chain and advanced functional testing incorporating the User Logic, the Core CRU FW and a MID front-end system emulator.

1.4 Objectives

The main objective of this project is to produce a prototype to be used as a base for the user logic in the MID detectors DAQ when the upgraded ALICE detector goes online in 2021.

1.5 Methodology Overview

Due to the novel nature of this project and periphery components, the following methodology was utilized for the development of this project.

- The identification of relevant stakeholders
- The sourcing of pertinent technical documentation
- The identification of software and hardware requirements as well as the applicable skills required
- The engagement with stakeholders to elucidate fucntional requirements for the work produced
- Critical analysis of established requirements and possibilities
- Synthesis of the information obtained thus far and a proposal for a possible solution
- Prototyping and evaluation of the proposed solution

  A more comprehensive explanation of the methodology used in this project can be found in Chapter 3.

1.6 Purpose of this Work

The existing readout system of the ALICE MTR is not functionally sufficient for the expected luminosity of the anticipated high data readout expected in the LHC RUN3 phase after LS2 and thus needs a completely new system from the Very Front-End Electronics (VFE) to the computing system used for recalibration and reconstruction [16].

This new readout system is largely designed and in prototype phase. However, a difference in protocol between the front-end system, the trigger and timing system and the computing system requires a custom user logic to interface them.

The purpose of this work is to determine whhether a custom autonomous embedded system can be designed and prototyped that meets the throughput and interfacing requirements of the envisaged ALICE MID in an acceptable time. Furthermore, can it be developed within the constraints of the established hardware and software requirements such as the approved FPGA[2].

1.7 Scope and Limitations

The scope of this project is limited to the design and development of firmware for the MID readout system's uplink path i.e. front-end electronics to the computing farm as shown in figure: 1.3. Involvement with periphery systems is limited to the interfaces and input/output characteristics with those systems. Furthermore, integration with the existing system is only tested through conformance with established requirements and not in practice.

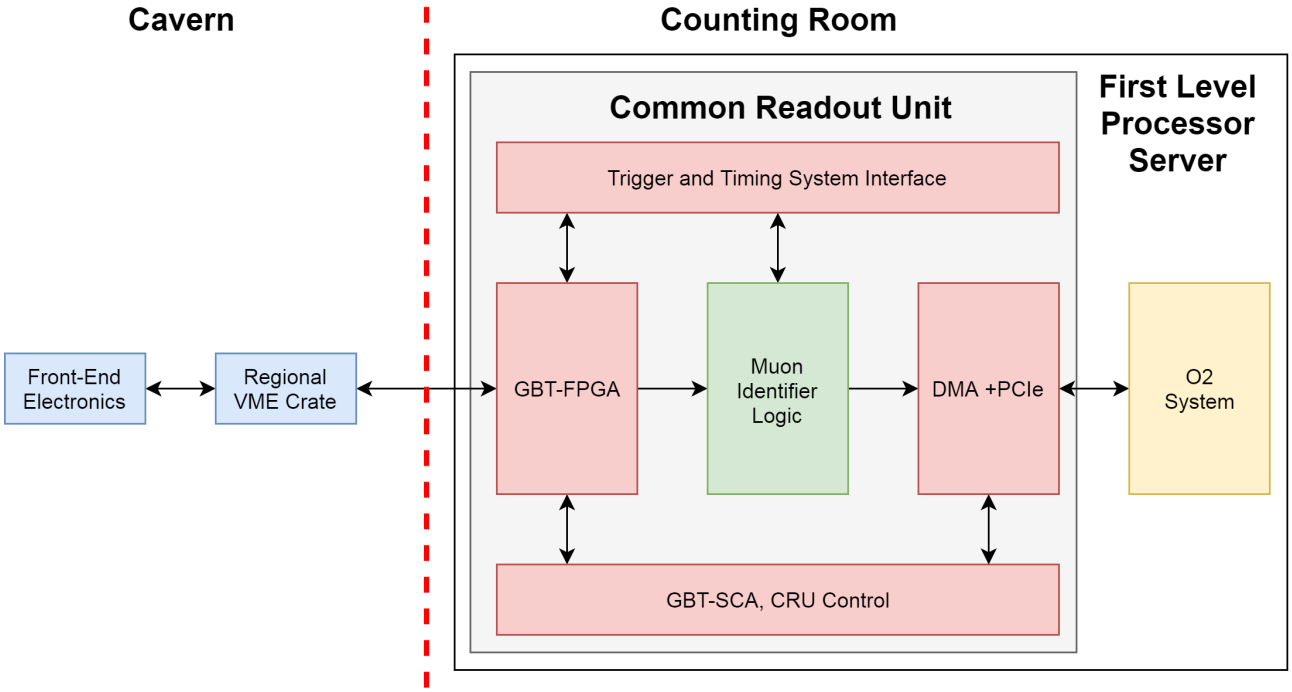


Figure 1.3: MID-CRU Interface [2]

This project was taken up after comprehensive research and development activities of the full ALICE detector was undertaken. As such, technical decisions such as the hardware and software to be used were long established, and they include communication protocols, design tools, programming languages and most relevant, the resource utilization limit of the project in question as outlined in technical notes [2],[4] and [16]. These details are examined in **Chapter 2**. All the periphery systems presented are still under development and therefore might present a change in requirements which may not be reflected at the time of publication of this dissertation.

Although a more comprehensive design verification methodology is produced in section 3.3 this work only performs functional verification of the prototype developed at the conclusion of this MSc.

1.8 Thesis Overview

This dissertation begins with the provision of the background information and examines the problem being addressed. Rudimentary objectives of the project are identified and an overview of the methodology used to achieve these objectives is presented. The hypothesis is outlined and constraints are presented.

Chapter 2 presents pertinent information about the periphery systems to the project in question. In particular, the characteristics of the upgraded hardware and software components of the readout system as they currently stand.

Chapter 3 expands on the methodology used to fully ascertain scope, constraints, objectives and evaluation procedures to successfully complete the project.

Chapter 4 presents a detailed conceptual design of an envisaged system that conforms to the requirements previously established.

Chapter 5 tests the performance of the prototype of the conceptual design previously presented in relevant fields of metric for a project of this nature.

Chapter 6 concludes this dissertation by performing a comparison between the established objectives of the project and the performance of the prototype, results are then presented. Finally, additional work that needs to be done is discussed. This includes architecture not developed in this work and enhancements to the user logic.

Chapter 2

The ALICE MID Readout System

The running conditions of the LHC after LS2 will deliver Pb-Pb collisions of up to $6x10^{27}cm^{-2}s^{-1}$ corresponding to collision rates of 50 kHz for Pb-Pb collisions recorded in minimum bias mode together with dedicated p-Pb and pp reference runs[17][13].

The ALICE upgrade addresses physics topics requiring precise measurements of experimental observable data relevant in the study of the QGP which are characterized by very small signal-to-background ratios making triggered readout very inefficient[13]. The proposed continuous readout mode results in a data output from the detectors roughly 2 orders of magnitude greater than that of Run 1 (2009 - 2013). Continuous readout is significantly more challenging for the O2 system and therefore requires an upgrade. However to minimize cost and requirements from the upgraded O2 system, the ALICE Computing Model for Runs 3 (2021 and onwards) and 4 (post 2023) is designed for a maximal reduction of the data volume read out from the detector as early as possible during the data-flow[18][19]. Detector readout will be activated either by a minimum bias trigger or in a continuous mode. The data volume reduction will be achieved by reconstructing the data in several steps almost synchronously with data taking[17].

2.1 The Muon Trigger to Muon Identifier

The Muon Trigger (MTR) since coming online has operated in a triggered readout mode, however, to cope with the increased luminosity of the LHC in Runs 3 and 4 triggered readout is insufficient. The future designation of this sub-detector will operate in continuous readout mode whilst maintaining its purpose of muon identification thus its renaming to Muon Identifier (MID).

2.1.1 Front-End System

The Front-End System (FES) is comprised of three main components, namely, the detector which consists of Resistive Plate Chambers (RPC), Very-Front-End Electronics (VFE) known as FEERIC and Front-End Electronics (FEE) which consists of 114 local boards and 16 regional boards.

Resistive Plate Chambers

The MTR is a detector built with Resistive Plate Chambers (RPC). When muons pass through the RPC it ionizes the gas, the electrical signal is then picked up inductively on both sides of the detector by means of orthogonal copper strips (1, 2 and 4 cm wide), in the horizontal (bending plane) and vertical (non-bending plane) directions, one for each polarity, allowing a three dimensional hit reconstruction. Strips are placed outside the electrodes[15].

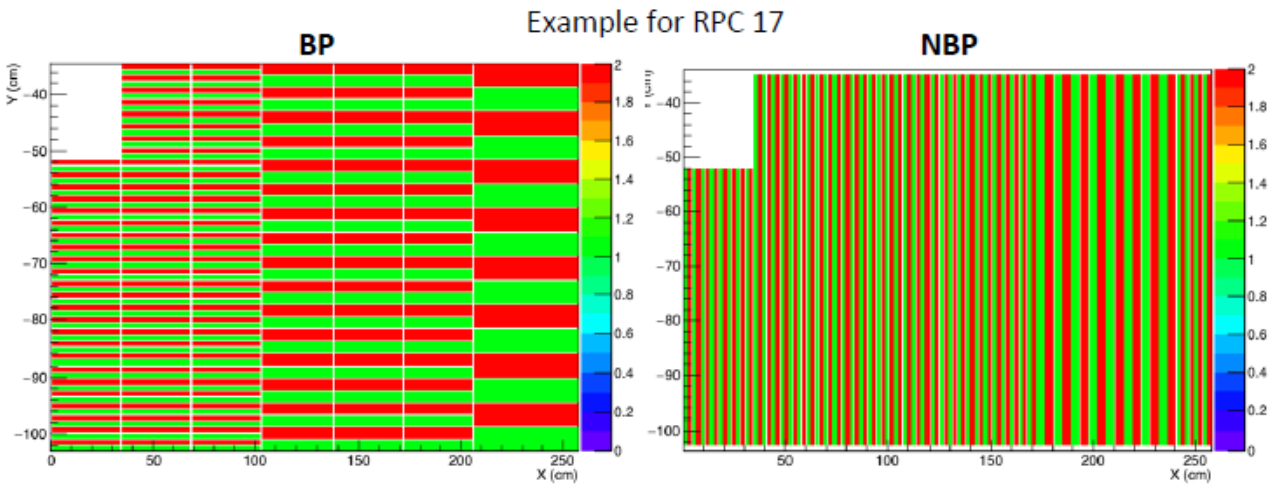


Figure 2.1: RPC Strips in BP and NBP for RPC 17

Front-End Rapid Integrated Circuit

Due to the future operating conditions, the present very front-end cards(VFE) of the Muon Trigger, whose acronym is ADULT, must be replaced to compensate for the gain loss brought on by the aging of the Resistive Plate Chambers[20]. The boards that will replace the ADULT chip is known as FEERIC (Front-End Electronics Rapid Integrated Circuit). Unlike the ADULT chip the FEERIC chip will perform amplification of the analog input signal, enabling operation of the RPC's in avalanche mode, with a much smaller (factor 3-5) charge deposit in the detector. The VFE system consists of 21,000 channels, distributed over 2400 electronics cards equipped

with one or two ASICs[21].

Local and Regional Boards

As previously mentioned, the readout electronics need to be upgraded to allow for continuous readout operation. Thus, new local and regional boards are currently under development at Subatech at Nantes in France.The architecture[3] being developed and tested are presented in this section.

MID Local Cards

Each local card interfaces with 128 front-end LVDS signals from FEERIC ASICS (32 per plane, 16 in both the BP and NBP). The interfaces of these cards are as follows [3]:

- A LVDS local tracklet output indicating whether the card is sending data
- A bidirectional e-Link @320 MB/s (clk40 + Data In + Data Out) sending the data payload to its respective regional card and sending trigger and timing data to the local card.
- An I2C, Config and birectional USB 3.0 interface for Detector Control System operation.
- 1 FPGA
- 8x8 Delay Switch
- Power Supply

MID Regional Cards

A single regional card interfaces to up to 8 local cards via 8 e-links and using the GBT protocol. A regional crate contains 2 regional cards, resulting in up to 16 local cards per crate. Additionally there are 2 internal links in the regional crate (one for each local board). This architecture is depicted in figure: 2.2.

The interfaces of the regional cards are as follows:

- 16 local card tracklet input
- 2 e-links@320 Mb/s for TTC payload

- 8 e-links@320 Mb/s for Data payload for 8 Local cards
- 2 bidirectiional optical GBT links @3.2 Gb/s to CRU for TTC and detector data Payload
- USB 3.0 for DCS operation
- 1 Regional card tracklet output
- 2 GBTx ASICs
- 2 GBT-SCA ASICs
- 1 FPGA
- 1 LVDS Config
- Power Supply

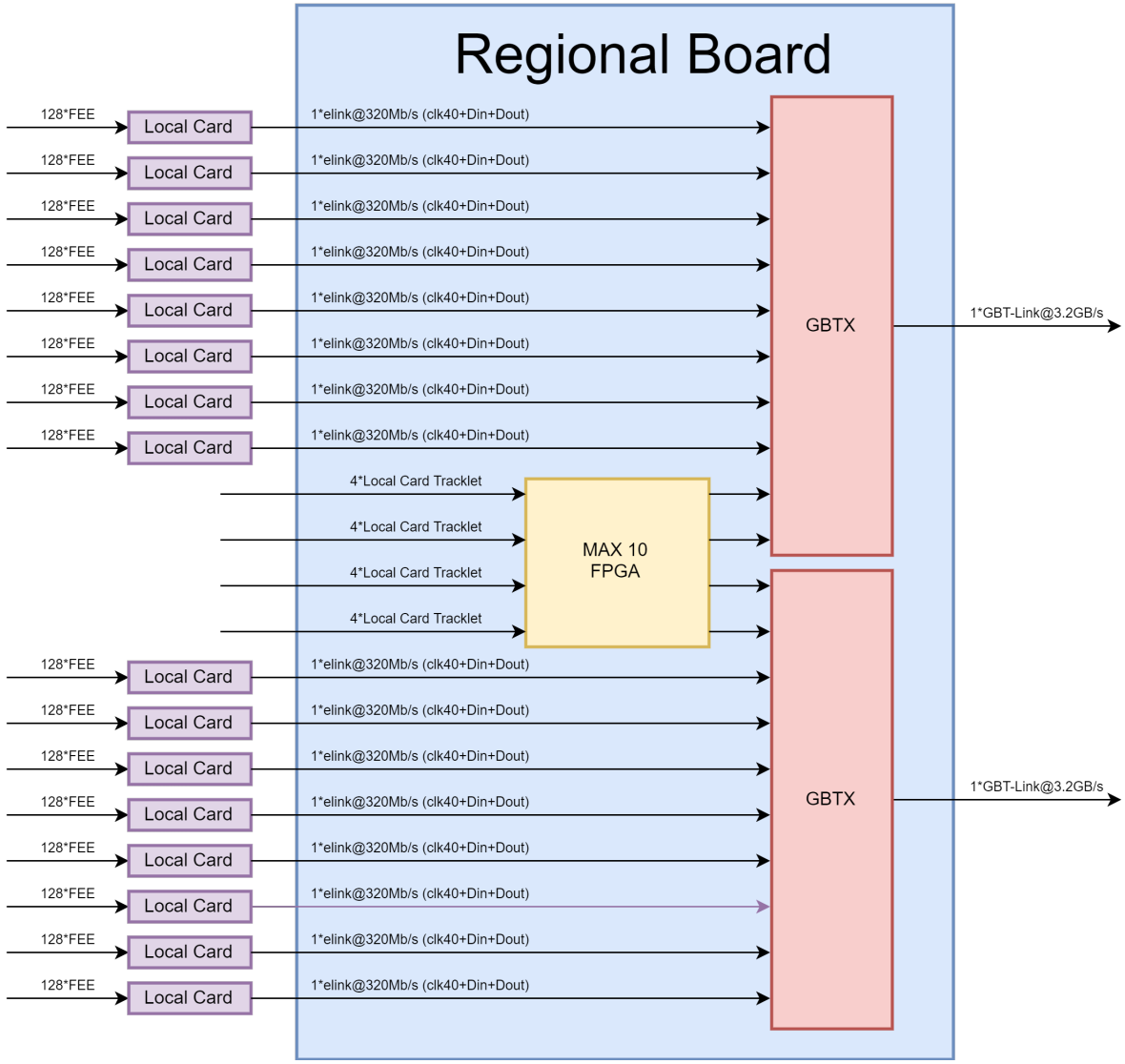


Figure 2.2: Front-End Interface with Regional Board[3]

The format of the data payload produced by the regional cards is tabulated in figure 2.3.

MID J2 Backplane

The MID J2 Backplane interfaces the local and regional cards. Presently its architecture is as follows:

- 16 LVDS lines
- 16 e-link@320 Mb/s
- 16 I2C interfaces

*Coding of SOC, EOC, RESET events in **REGIONAL***	*Number of bits*
START BIT (always '1')	1
CARD TYPE (always '0'=REGIONAL)	1
REGIONAL BUSY ('0'=OK; '1'=FIFO full)	1
REGIONAL DECISION (tracklet)	1
ACTIVE ('0'=OFF; '1'=ON)	1
REJECTING ('0'=OFF; '1'=ON)	1
MASKED ('0'=OFF; '1'=ON)	1
OVERWRITED ('0'=OFF; '1'=ON)	1
SOC	1
EOC	1
PAUSE (always '0')	1
RESUME	1
CALIBRATE	1
SPARE	1
RESET	1
HC	1
Internal bunch counter	16
REGIONAL crate number (0-15)	4
Status: Masks on tracklet inputs	4
N/A	0
Total number of bits	40
Bunches needed to send	5

*Coding of PAUSE, RESUME, CALIBRATE, SPARE, HC events in **REGIONAL***	*Number of bits*
START BIT (always '1')	1
CARD TYPE (always '0'=REGIONAL)	1
REGIONAL BUSY ('0'=OK; '1'=FIFO full)	1
REGIONAL DECISION (tracklet)	1
ACTIVE ('0'=OFF; '1'=ON)	1
REJECTING ('0'=OFF; '1'=ON)	1
MASKED ('0'=OFF; '1'=ON)	1
OVERWRITED ('0'=OFF; '1'=ON)	1
SOC (always '0')	1
EOC (always '0')	1
PAUSE	1
RESUME	1
CALIBRATE	1
SPARE	1
RESET (always '0')	1
HC	1
Internal bunch counter	16
REGIONAL crate number (0-15)	4
Always '0'	4
N/A	0
Total number of bits	40
Bunches needed to send	5

*Coding of **self triggered physics** event in **REGIONAL***	*Number of bits*
START BIT (always '1')	1
CARD TYPE (always '0'=REGIONAL)	1
REGIONAL BUSY ('0'=OK; '1'=FIFO full)	1
REGIONAL DECISION (tracklet)	1
ACTIVE (always '1'=ON)	1
REJECTING (always '0'=OFF)	1
MASKED ('0'=OFF; '1'=ON)	1
OVERWRITED ('0'=OFF; '1'=ON)	1
Always '0'	8
Internal bunch counter	16
REGIONAL crate number (0-15)	4
Data: All tracklet inputs	4
N/A	0
Total number of bits	40
Bunches needed to send	5

Figure 2.3: Regional Card Data Format [3]

- Power supply
- Config interfaces
- 1-to-16 USB 3.0 hub

2.1.2 The Central Trigger System

Data synchronization is paramount in the operation of the trigger and readout systems of the ALICE detector. The existing triggered readout is based on an electronic system, known as the Central Trigger System (CTS), which decides whether an event is of interest and needs to be saved. The CTS is comprised of two components, namely, the Central Trigger Processor(CTP) and the Local Trigger Unit (LTU)[7].

Central Trigger Processor

The CTP is the decision maker of the system. CTP will transmit L0, L1, L2 trigger signals to detectors [22]. When trigger detectors detect the event they were designed for, an L0 trigger signal is transmitted from the CTP to the detector to initialize digitization of the channels. The

detector is then able to perform additional online analysis of the event, if it is still of interest the detector will transmit a L1 trigger signal after which the CTP will transmit an L1 trigger back to the detector. Once a detector has received the L1 trigger siganl it has additional time to continue processing the event. If the event being processed matches the event of interest the detector will transmit an L2 trigger signal, the CTP will then transmit the trigger signal to the detector indicating that it can transmit the processed event.

The planned upgrade to the ALICE sub-detectors aims to move from triggered readout to continuous readout, however some sub-detectors will maintain the ability to perform triggered readout. The continuous readout system makes use of periodic signals called heartbeart triggers approximately every 89.4 ms. This serves a timestamp of the data acquisition called Heartbeat Frames (HBF). There will be new triggers for the ALICE MID along with the heartbeat triggers in continuous readout modes, these listed in table 2.1 [3]. The LTU is transparent in the global mode and emulates the CTP in standalone mode [7].

Table 2.1: ALICE MID FEE Continuous Readout Triggers[3]

Trigger Type	Functionality	Code to FEE
Start of Continuous (SOC)	Start physics data acquisition	0x80
End of Continuous (EOC)	Stop physics data acquisition	0x40
Calibrate (CAL)	Perform front-end test	0x08
Reset (RST)	Reset readout procedure	0x04
health Check (HC)	Check status of readout channels	0x20

Local Trigger Unit

Historically, the sub-detectors in ALICE detector received their trigger signals via a standard interface called a Timing, Trigger and Control (TTC) partition in the Local Trigger Crate (LTC). The TTC partition receives the trigger signals in an internal format from the CTP and generate the signals required to drive the detector [23].

New LTU boards are being developed to perform similar functions in the upgraded detector using the new continuous readout triggers. These new signals will be interpreted by the core CRU firmware, are bitmapped to the format required by the FES and transmitted via the

GBT links. The codes are listed in table 2.1.

CRU-CTP Protocol in Continuous Mode

This section outlines the data protocol between the CTS, the CRU and the GBT front-end as well as a description of the data being transmitted. The CTP-CRU protocol is as follows [7]:

- At every bunch crossing (BC) a TTC-PON message of 77 bits is transmitted by the CTP, indicated by action 1 in figure: 2.4
- When data transmitted from the front end (FE) has been successfully acquired by the CRU and transferred to the Front Level Processor (FLP) the CRU transmits a HB acknowledge message (HBAm), format shown in table 2.3, to the CTP, shown by action 2b in figure: 2.4. The CTP will timeout the reception of a HBAm using a programmable parameter of order $T_{HBAm}8 * 89.4\mu s$.
- The CTP then evaluates the HB decision (HBd) for a given HB frame after the time $T_{HBAm}\mu s$ has elapsed. The HBAd is based on a predefined function of HBAm messages of all CRU, shown by action 3 in figure: 2.4
- The CTP then modifies the HBaccept/HBreject flag in the trigger type in TTC-PON message
- A Heart Beat Map of a given TF (HBMTF) is then created by the CTP after collecting all the HBd corresponding to that TF. Finally, the HBMTF is transmitted asynchronously to the CRU, indicated by action 4 in figure: 2.4

CTP Trigger Message

As mentioned previously the CTP transmits a TTC-PON message of 77 bits to the FEE and CRU's every bunch crossing. The format [7] of which is.

- Trigger type (TTYPE) - 32 bits
- Even identification - 44 bits (Encapsulating an LHC orbit counter (ORBIT) - 32 bits and bunch cross counter (BCID) - 12 bits)
- Trigger type data valid bit (TTVALID) where a value of '1' indicates all other fields (TTYPE, ORBIT, BCID) are valid and a value of '0' indicates that the bits in the other fields can be used for communication e.g. slow control

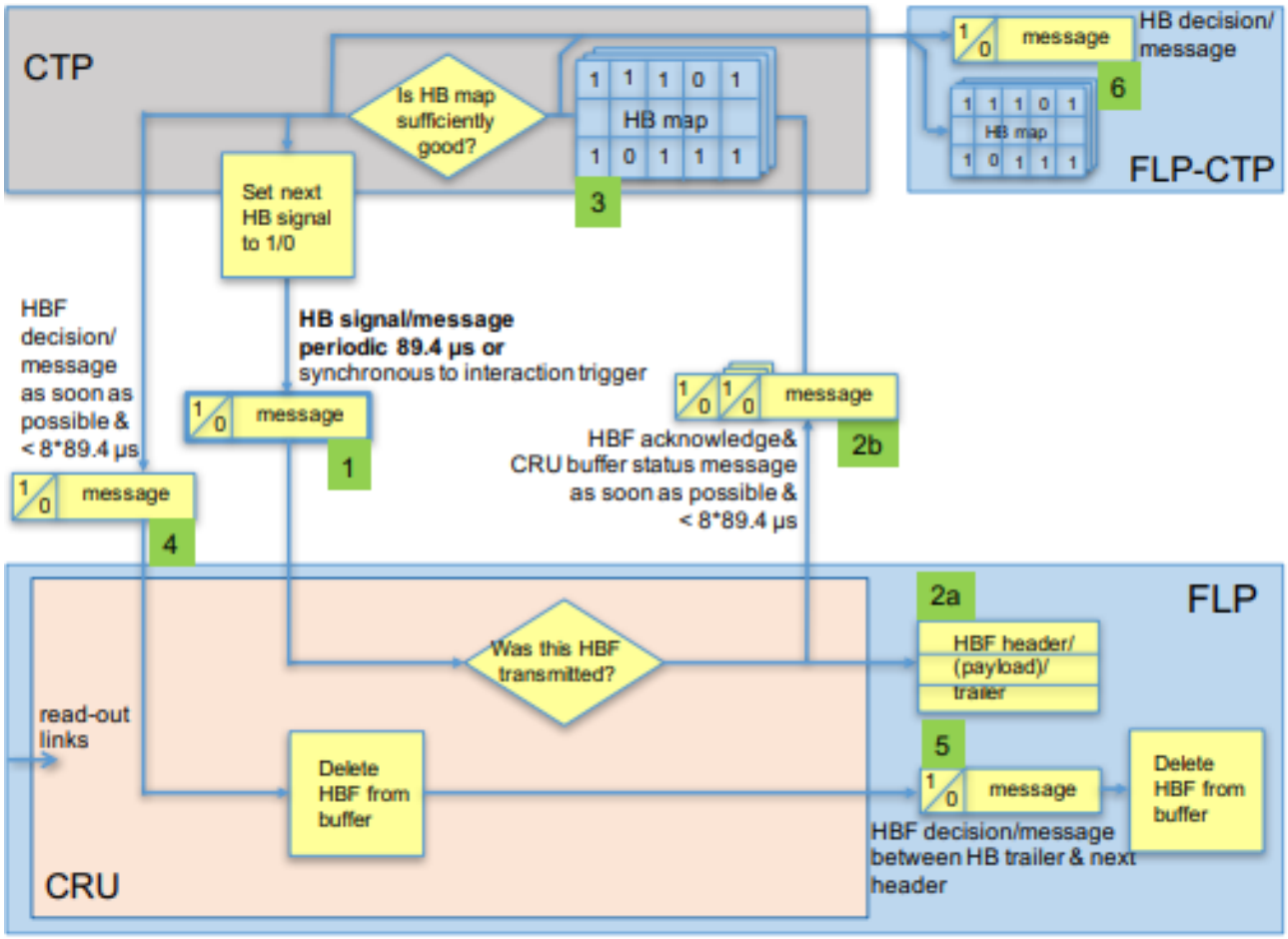


Figure 2.4: CTP-CRU Signal and Message Flow [4]

The format of the TTYPE field is shown in table 2.2. The range [11:0] are used for general purposes whereas the range [29:31] can be used in different ways by different detectors. When the HB bit has a value of '1' this indicates that trigger message is of Heart Beat (HB) type and the corresponding HB trigger type is flagged in the relevant field. The corresponding ORBIT and BCID fields of the message identify that particular HB timeframe. It must be noted that the HB frequency is defined to be equal to the LHC orbit as thus there is only one HB per ORBIT. Also, Trigger messages with HB='1' have BC=0 since HB is synchronized with the LHC orbit.

Table 2.2: Trigger Types transmitted from CTS [7]

Bit	Name	Description
0	ORBIT	ORBIT
1	HB	Heat Beat flag
2	HBr	Heart Beat reject flag
3	HC	Health Check

4	PhT	Physics Trigger
5	PP	Pre Pulse for calibration
6	Cal	Calibration trigger
7	SOT	Start of Triggered Data
8	EOT	End of Triggered Data
9	SOC	Start of Continuous Data
10	EOC	End of Continuous Data
11	TF	Time Frame delimiter
...	...	Time Frame delimiter
29	TPCsync	TPC synchronization
30	TPCrst	TPC reset
31	TOF	TOF special trigger

Table 2.3: HB Acknowledge Message (HBAm) [7]

PON Byte	Size	Payload	Description
0	8 bits	[7:0]	HBID (ORBIT)
1	8 bits	[15:8]	HBID (ORBIT)
2	8 bits	[23:16]	HBID (ORBIT)
3	8 bits	[31:24]	HBID (ORBIT)
4	8 bits	[7:0]	CRU ID
5	2 bits	[9:8]	CRU ID
5	6 bits	[7:2]	Spare
6	1 bits	[0:0]	CRU Acknowledge
6	2 bits	[1:0]	CRU Buffer Status
6	5 bits	[4:0]	Spare

2.1.3 Gigabit Transceiver and Protocol

The planned upgrade to the LHC will see significantly higher radiation doses to the communication links and front-end electronics [24]. To address these issues affecting the communication links a new Rad-Hard optical link based on the custom CERN GBT protocol that was developed [25].

Single Event Upsets (SEU) are a major impairment to error free data transmission in High Energy Physics experiments (HEP) due to the radiation produced. The GBTx tackles this issue by using robust line encoding and error correction, capable of correction single bit and burst errors caused by transmission errors. It operates in 3 modes, namely, Standard, Wide Bus and 8B/10B. The standard mode devotes 30% of the optical link bandwidth to the transmission of Forward Error Correction (FEC). SEU's are also resolved internally in the GBT chips using dedicated design methodologies [25].

GBT ASIC

An ASIC was developed to be used on the front-end electronics of the detectors, referred to as GBTx. Its main features being intolerance to the high radiation doses in the cavern as well as function as a multipurpose bidirectional optical link for HEP Experiments[25].

The bidirectional optical link has three purposes:

- Data acquisition (DAQ) from front-end electronics (FEE)
- Timing, Trigger and Control (TTC) from Central Trigger System (CTS) via the Local Trigger Unit (LTU)
- Slow control from the Detector Control System (DCS)

Logically, there are three "distinct" data paths in the GBT Link. In practice, physical separation is unnecessary and instead the paths are merge on a single optical link [25].

The intended usage of the GBT link is varied for multiple front-end applications and as such it was designed to be highly flexible with many modes of operation, namely: [25]:

- Can be configured to be a bidirectional transceiver, a unidirectional transmitter or a unidirectional receiver.
- Different front-end interface modes and options.
- Extensive features for precise timing control.
- Extensive control and monitoring features.
- A high level of error correction from SEU's and transmission errors.

Gigabit Transceiver Slow Control Adapter

The Gigabit Tranceiver Slow Control Adapter (GBT-SCA) ASIC, part of the GBT chip-set, has the purpose of transmitting control and monitoring signals to the on-detector FEE and performing monitoring operations of detector environmental parameters.

Gigabit Transceiver FPGA IP Core

The Gigabit Transceiver FPGA IP Core (GBT-FPGA) is a fully-fledged VHDL Library developed to emulate the GBTx serial link and test initial GBTx prototypes as well as provide a basic "starter kit" allowing developers to familiarize themselves with the GBT protocol [26].

The Rad-Hard GBTx ASIC is only installed on the Front-End System since this where the high radiation doses occur. As such, GBT-FPGA is now also used in other areas such as the counting room on COTS components, such as an FPGA. The IP Core allows for the usage of GBT Protocol over the GBT-links to interface with the GBTX chips on the FEE.

GBT Frame Format

During a single LHC bunch crossing interval (25 ns) a 120-bit frame is transmitted, resulting in a line rate of 4.8 Gb/s. It is comprised of a 4 bit Frame Header, 32 bits of Forward Error Correction (FEC) and 84 bits for data transmission (resulting in a user bandwidth of 3.36 Gb/s). Slow Control or Internal Control (IC) and External Control (EC) information makes up 4 of the 84 bits, leaving 80 bits for user data transmission which can be used indistinguishably for Data Acquisition (DAQ) and Timing,Trigger and Control (TTC) [25].

2.1.4 Online-Offline System

The O2 Facility contains a hierarchy of processing units comprised of CRU, First Level Processors (FLP), Event Processing Nodes (EPN) as well as networking and storage resources. Each level of this hierarchy performs work on the raw data in real-time until the data is of sufficient quality for storage.

Common Readout Unit

This sections outlines the hardware and functional specifications of the Common Readout Unit (CRU). The hardware specifications are fixed for all ALICE sub-detectors that will utilize the CRU whereas certain functional specifications are detector specific and require custom logic integrated into the core CRU FW.

Hardware Specifications Here is an outline the hardware specifications of the CRU that is currently under development at CERN.

Altera FPGA The main component of the CRU, responsible for data conditioning bi-directionally, is Altera's Arria 10 10AX115S4F45I3SGES.

The functionality of which is the following:

- 1.15 Million Logical Elements
- 8.4 MB Internal Memory
- 1518 Variable Precision DSP blocks
- 3036 18x19 Multipliers
- 32 Fractional PLL's
- 16 I/O PLL's
- 72 12.5 Gbits/s (Max) High Speed Serial Links
- 500 MHz Max Internal Speed

Configuration Devices On power up the FPGA is configured using either the JTAG or PCIe interface using the CvP protocol. This is performed after an EPQS512 quad serial device has downloaded the configuration [2].

Front-End Interface Interfacing with FEE is achieved using 4 optical transmitters and receivers from Avago known as minipods. Each of these transmitters and receivers can handle up to 12 optical links resulting in a maximum of 48 bidirectional links at 10 Gb/s achieving a total bandwidth of 0.48 Tbits/s per CRU board [2].

JTAG Management The Arria 10 FPGA can be programmed through three sources of JTAG, namely:

- An external 10 pin connector located on the front-board
- A JTAG link piloted by the PCIe
- A JTAG link piloted by an embedded USBblaster.

Communication Over Detector GBT Links

There are three groups of communications defined by functionality:

- Trigger related communications
- Detector data related communications
- Slow control related communications

The GBT Uplink will handle the raw 80 bit data stream transmitted from the detectors as independent serial links with custom protocol and forward it to the MID user logic. It is also responsible GBTx and GBT-SCA control communication (receiving the response from the relevant ASIC encoded in the GBT Frame).

The GBT Downlink is responsible for the LHC clock distribution, trigger distribution and GBTx/GBT-SCA control to FEE.

2.1.5 CRU Readout Protocol

The O2 system needs to be able to identify the data packets transferred by the readout electronics. This is achieved by the Raw Data Header (RDH) and must be generated by the user logic of the CRU. Every GBT word (80 bits) corresponds to a RDH word (64 bits) and the remaining 16 bits are reserved and should be 0x0. The format [27] of the RDH is shown in figure: 2.5.

Table 2.4: RDH Field Description

Name	Size	Description
Header Version	8 bits	Header version number

Header Size	8 bits	Size of RDH in bytes (64 bits)
Block Length	16 bits	Size in bytes of the payload (RDH excluded)
FEE ID	16 bits	Unique ID assigned to the FEE
Priority Bit	8 bits	When 0x1 the packet is moved forward with higher priority (for example to report HB frame when buffers are full)
TRG Orbit	32 bits	Trigger orbit (must always be filled if available)
HB Orbit	32 bits	Heartbeat Orbit (must always be filled if available)
HB BC	12 bits	Heart bunch crossing (must always be filled if available)
TRG Type	32 bits	Trigger type set by CTP
Detector Field	16 bits	Detector specific field
PAR	16 bit	Field used by detector to trigger a re-configuration of the FEE
STOP	8 bit	1 bit used to identify the last page (if there are no more pages of 8KB belonging to the same trigger)
Pages Counter	16 bit	Counter to keep track of the different pages belonging to the same trigger

Table 2.5: CRU Reserved Field Description

Name	**Size**	**Description**
Link ID	8 bits	GBT Channel Number where data originates
Memory Size	16 bits	Size in bytes of the data moved in memory (Header + payload)
OFFSET next packet	16 bits	offset in bytes where the next header is located in memory. By default this value is 8K (0x2000)
Packet Counter	8 bit	Counter increased for every packet received in the link
CRU ID	12 bits	Number used to identify the CRU
DATAPATH WRAPPER	4 bits	Number used to identify one of the 2 End Points [0/1]

The readout scheme proposed by Core CRU FW developers is as follows:

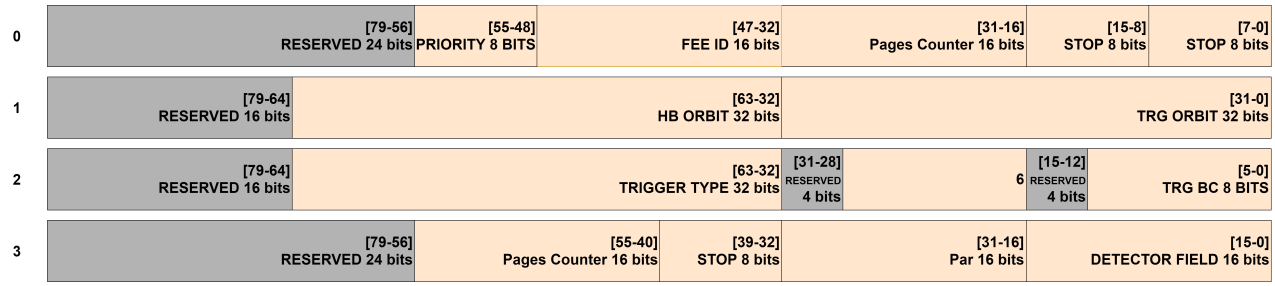


Figure 2.5: RDH field format

- Data can go up to 240MHz
- RDH is provided by CRU User logic in first 2 words (256 bits)
- Packet start with SOP and end with EOP
- Data supposed to use full PCIe data width to maximise throughput

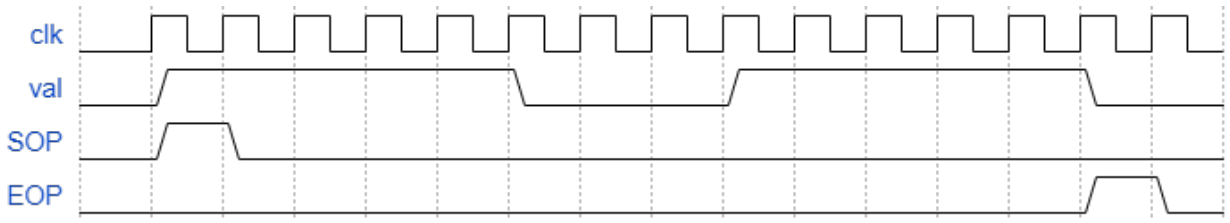


Figure 2.6: Timing diagram of proposed readout scheme control signals)

SOP	EOP	215 — 192	191 — 128	127 — 64	63 — 0
1	0	CRU RESERVED	RDH1	CRU RESERVED	RDH0
0	0	CRU RESERVED	RDH3	CRU RESERVED	RDH2
0	0	DATAUSER1		DATAUSER0	
0	0	DATAUSER3		DATAUSER2	
0	0	DATAUSER5		DATAUSER4	
0	0	DATAUSERx		DATAUSERx	
0	1	DATAUSERn+1		DATAUSERn+1	

Figure 2.7: Proposed readout scheme of data payload via CRU Core FW PCEe data wrapper

First Level Processor

The data received from the ALICE MID FEE is assembled into time frames, currently defined as 22 ms, however this is a programmable parameter. The CRU will send the data per heart-beat frame (HBF) to the FLP[4].

The FLP's of each subdetector organise the HBF's of each subdetector into a sub-time frame (STF). The data is then forwarded to the Event Processing Nodes (EPN) [4].

Event Processing Nodes

The Event Processing Nodes are responsible for assembling the data contained in a STF from each sub-detector into a single Time Frame (TF). The data from a TF is then sent to the O2 system [4].

Chapter 3

Methodology

This chapter outlines the methodology used in the design of the ALICE MID user logic.

As indicated previously, the development of the user logic for the MID was undertaken after much research, planning and development was conducted by collaborators for the ALICE detector upgrade. As such, extensive initial research had to be done to converge on its present stage of development. This research was conducted by reviewing of technical design reports, technical notes, presentations as well as consultations with hardware and software developers of the data acquisition chain since there is little published documentation on these systems. Once this phase was to complete the following methodology was used complete this project:

- User Requirements Analysis and Constraints Identification
- Design Specification
- Firmware Design
- System Evaluation

3.1 User Requirements and Constraints

In this project it is of paramount importance to establish requirements and constraints of any possible solution. The rationale lies in the established fact that it needs to fit seamlessly into a much larger system. The output from the previous research phase now provides the input to requirement identification, requirement analysis and establishing design constraints. These are explained in this section.

3.1.1 User Requirements

Following lengthy discussions with members of the Core CRU team at CERN and collaborators at Laboratoire de Physique de Clermont (LPC), Clermont Ferrand and Subatech in Nantes in France who are currently developing the FEE and the O2 system for the ALICE MID, as well as interpretation of technical documents[17] [2] [16] and following various update meetings the following user requirements were ascertained and corroborated by stakeholders.

Functional Requirements

Consolidation of information obtained from CERN and collaborators resulted in the following basic user requirements were established:

1 Data acquisition of the data payload from the FEE via the Core CRU firmware

2 Synthesis of the MID O2 Header

3 Data reformatting from FEE format to the required O2 format

4 Zero Suppression of elements of the FES not transmitting data

5 Synchronization of the data payload

6 Population of the various data fields of the RDH

7 Data readout and transmission from user logic to O2 system via Core CRU firmware

Requirement Analysis

The data acquired by the ALICE MID will be presented to the CRU Core firmware via 16 optical links using the standard GBT protocol @3.2 Gb/s. The data from each optical link is then deserialized and presented to the MID specific firmware as a 16∗80 bus along with is_valid and is_data lines for each GBT link. This data contains the header information from the specific segment of the 4 planes connected to that specific local card such as status, trigger, timing and card ID information as well as the digitized muon hits.

The 5th byte of data presented to the user logic contains information about which Regional and Local cards produced the data per GBT Link as well as indicating which planes are transmitted data (refer to table: 2.3 for more detail on FEE data format). They do not indicate the unique location in the detector that the data originates from thus the user logic needs to use

the Regional Card ID, Local Card ID and tracklet to determine the exact RPC, the column within that chamber and the row within that column and put this information into the O2 header.

Slight variations in the properties of the multiple optical and electrical links introduces variations in transmission time. Consequently, the data is out of sync on arrival at the CRU and needs to be synchronized in the user logic.

In continuous readout mode data acquired is time delimited using a periodic heartbeat trigger transmitted by the CTS every ∼89us. The period between heartbeat triggers is known as a heartbeat frame (HBF). The FEE upon reception of this trigger via the Core CRU FW initiates the digitization of its channels periodically but at a higher frequency than that on the HB triggers. The data being transmitted contains this internal bunch count of the FEE but not the HB count, this effectively produces data payload in sub-heartbeat frames. This is incompatible with the O2 system since it requires HBID's in the Raw Data Header. Thus the HBID needs to be determined using the internal bunch counter information provided in the FEE data payload as well as BCID in the CTP payload. The trigger that produced that set of data also needs to be added to the RDH which is a simpler matter of populating the respective field.

A key specification on the Core CRU FW PCIe downlink is the packetization of the data payload (limited to 8KB) prior to transmission (More detail outlined in section 2.1.5).

3.1.2 Design Constraints

The MID firmware must be produced with consideration of the following design constraints characterized by the peripheral systems in the DAQ chain:

- It must be programmed in VHDL as stipulated by the Core CRU team
- It must be targeted at the Altera Arria 10 FPGA and programmed using Quartus Prime Pro[28] as this is the FPGA used in the CRU board
- It must be hard-coded logic elements and cannot make use of the processors provided by Altera
- The MID firmware must conform to resource utilization limits after the Core CRU Firmware has been installed on the FPGA

3.2 Design Specification

The user logic will interface with the Core CRU firmware using Altera Avalon streaming interfaces[2] and monitor the input lines for a valid status byte to trigger data acquisition, The data payload being transmitted will then be stored in the appropriate registers for status, trigger, internal bunch counter local/regional card ID's and tracklet on every rising edge of the clock of the FPGA.

Once the Local and Regional Card ID's for that data set have been extracted a valid bit is asserted indicating the start of data reformatting. This results in four O2 headers for the strip patterns corresponding to the four planes of the MID.

The value stored in the internal bunch counter register will be monitored by logic for resets indicating the start of a new HBF and hence an increment in the HBID.

Data synchronization is achieved by storing the data temporarily in a 2-dimensional fifo comprised of an array of fifo's. Each individual fifo corresponds to a local card in each plane. By monitoring the empty signal's of each fifo, the status of data synchronization is known. Once synchronization is achieved the first level of each fifo is read out.

Data belonging to the same HBF will be read out using a merge-sort algorithm arranging the data in ascending order and a pointer ROM that can be modified for different read out schemes for the data payload and will be streamed as an 8KB packet in the order required by O2 system and ordered by internal bunch counter containing the RDH.

3.3 Design Verification

The verification of the designed system for conformance to user requirements will be evaluated using three methodologies common to FPGA development. These perform functional and in-system verification.

3.3.1 Functional Verification

Functional verification is used to test the functional capabilities of an RTL design. More specifically, is determines whether the design meets the specifications established. In this context the RLT design is the MID User Logic.

In this project, functional verification is simulation based using ModelSim[29]. This is achieved by developing an HDL testbench for each module which reads bit vectors (of expected operating scenarios) from a text file and driving them into the device under test (DUT) using a stimuli generator. The outputs are then read by the HDL testbench and written to an output text file. The expected operating clock of 240MHz is simulated in the testbench using clock generator logic. Conformance to requirements is then assessed by inspection. A graphical depiction of this procedure is shown in figure: 3.1.

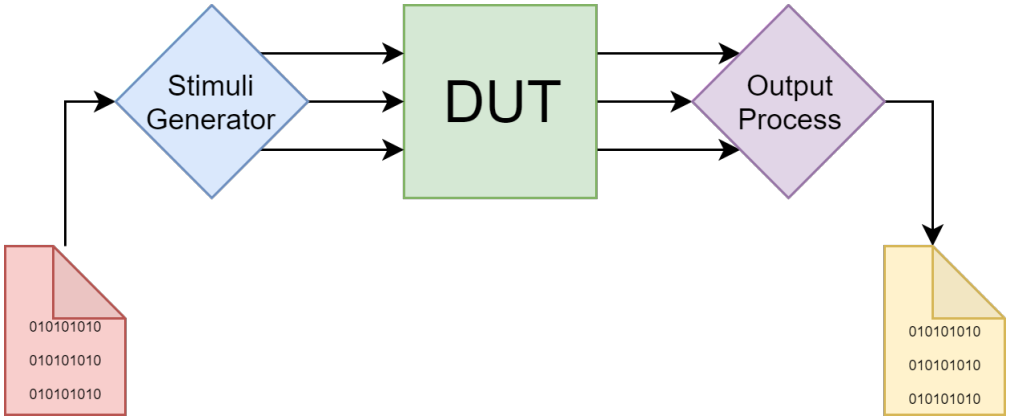


Figure 3.1: Functional Verification in HDL testbench

The tests being conducted aim to ascertain whether modules meet the established requirements of the user logic. The suite of tests that need to be performed for functional verification are as follows:

- Successful extraction and temporary storage of data payload
- Accurate reformatting of data payload
- Successful synchronization of data payload
- Successful transmission of data payload from user logic

The single operating scenario that would encapsulate this entire verification suite would be that of the frond-end test (FET). In the MID a FET is performed to identify dead channels (faulty electronics or broken links) in the data acquisition chain. This is performed periodically by the CTP sending a health check (HC) trigger to the front-end electronics who upon reception of this trigger fire (transmit a "TRUE" signal) which results in a complete data payload. As such, the FET expected data set with no dead channels is used as the simulated data set for functional verification. With the exception of the data extraction and synchronization module,

the vectors are always synchronized when driven into module ports.

More detail on the individual tests performed is explained in section 5.1, Design Verification. Further, improvements to this testbench would have the HDL performing a comparison between the generated file and an expected results output file.

3.3.2 Partial In-System Verification

The MID user logic is a sub-unit with the MID Common Readout Unit and that in itself is a sub-unit of the MID Data Acquisition chain. While functional verification is an important first step in the design and verification process it alone is insufficient to assess conformance to requirements.

It does not include any timing information, nor does it take into consideration changes done to the original design due to implementation and optimization.

The second level of testing requires realistic scenarios using physical hardware. This hardware testbench should represent the actual data acquisition chain. Discussion with collaborators in ALICE yielded the following hardware requirements:

- A high performance control PC to interface all the hardware
- A stand-alone Local Trigger Unit which has the ability to emulate the Central Trigger Processor and its software installed on the control PC
- A MID-Proto board which has the ability to emulate up to 7 local cards
- A Common Readout Board or Arria 10 FPGA development kit in the case of its unavailability with the CRU FW installed
- MID-DAQ and MID-DCS scripts for operation of the MID-Proto board
- An installation of the O2 System on the control PC
- An installation of the CRU SW on the control PC

Configuration of the listed equipment would enable data to be transmitted from a regional link to the O2 system providing a realistic scenario of the operation of the DAQ chain. This enables the performance of the user logic of when it is integrated into a rudimentary data acquisition chain with only one regional link (connected to the MID-Proto Board) whereas there are 16

links in total per CRU in the MID DAQ chain. Successful transmission of correctly reformatted and synchronized data would validate the operation of the user logic.

3.3.3 Advanced Functional Verification

While partial In-System testing is an important and useful step it is insufficient for full design verification. As its name suggests it only partially represents the system as is useful for timing analysis but the full fuctional verification of the user logic can only be performed with the full MID detector, in particular the full detector readout. As this is only possible once the hardware is installed and validated in the detector, a more advanced simulation based functional verification is necessary.

By adapting the HDL local card emulation code developed for the MID-Proto board[3] a full system MID emulator could be developed and incorporated into an HDL testbench that will interface with the core CRU FW and would effectively test the performance of the entire user logic while integrated into the Core CRU FW. The central trigger system can either be provided by the LTU if this functionality is available or using the CTP emulator provided by the core CRU firmware. The same test suite established will be utilized. This procedure is illustrated in figure: 3.2.

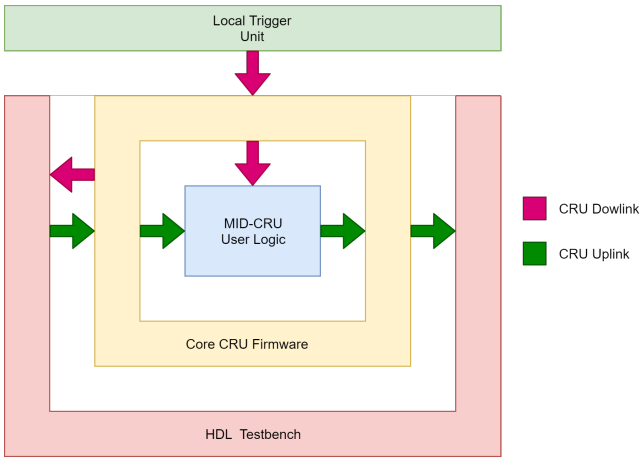


Figure 3.2: Advanced Functional Simulation

Further enhancements to this testbench would result in a complete research and development environment. By adapting the particle physics analysis framework used by physicists in the ALICE experiment to use expected operating parameters of the MID detector the need to wait until the physical MID detector is available for testing is eliminated. GEANT[30] in tandem with FLUKA [31] provide a simulated Monte Carlo particle transport generator through a 3D

model of the MID detector, the output data of which could correspond to the output format of the MID FEE to be used as an input the emulator for the user logic.

Chapter 4

MID User Logic Design

The user logic is designed by employing a modular hierarchical approach with the GBT banks as its base. The motivation for doing so is to facilitate scalability using parameterization, enabling faster development times for continuous improvement and easy adaptation for various testing scenarios. As such, functionality like data extraction, zero suppression and data reformatting for each optical link are handled independently and synchronized before moving to a higher level in the hierarchy, namely, data readout. This approach also allows efficient handling of operational errors such as noisy or broken links along the Front-End System. More details of the user logic designed are explained in this chapter.

4.1 Data Extraction

The data payload from the front-end system is delivered to ALICE MID user logic via the Core CRU firmware unchanged. The ports that interface the user logic to the Core CRU firmware are listed in table: 4.1.

Table 4.1: FE-CRU GBT Uplink Ports[2]

Port Name	Clock Domain	Description
gbt_tx_clk_i[N-1:0]		Input clock
gbt_rx_valid_i[N-1:0]	gbt_tx_clk_i	GBT payload valid bits
gbt_rx_isdata_i[N-1:0]	gbt_tx_clk_i	GBT payload isdata bits
gbt_rx_data_i[N*112-1:0]	gbt_tx_clk_i	GBT payload bits

Data extraction is initiated when a valid status byte is detected. This is achieved by periodically

monitoring the relevant uplink ports for the reception of a valid status byte which is the first byte of a valid data payload from the FEE. When this is true the ports are sampled every clock cycle and the data contained are stored in registers saving the data for use within the user logic. These registers are listed in table: 4.2 and table: 4.3.

Table 4.2: MID User Logic Regional Registers

Register Name	Size	Description
reg_v_stat	8 bits	Stores Regional Card status data
reg_v_trig	8 bits	Stores Regional Card trigger data
reg_v_ibc	16 bits	Stores Regional Card internal Bunch Count data
reg_v_cid	4 bits	Stores Regional Card card ID data
reg_v_trck	4 bits	Stores Regional Card tracklet data

Table 4.3: MID User Logic Local Registers

Register Name	Size	Description
loc_v_stat	8 bits	Stores Local Card status data
loc_v_trig	8 bits	Stores Local Card trigger data
loc_v_ibc	16 bits	Stores Local Card internal Bunch Count data
loc_v_cid	4 bits	Stores Local Card card ID data
loc_v_trck	4 bits	Stores Local Card tracklet data
loc_v_mt22_sp	32 bits	Stores Local Card MT22 strip pattern
loc_v_mt21_sp	32 bits	Stores Local Card MT21 strip pattern
loc_v_mt12_sp	32 bits	Stores Local Card MT12 strip pattern
loc_v_mt11_sp	32 bits	Stores Local Card MT11 strip pattern
loc_v_O2_h_mt22	19 bits	Stores MT22 O2 header
loc_v_O2_h_mt21	19 bits	Stores MT21 O2 header
loc_v_O2_h_mt12	19 bits	Stores MT12 O2 header
loc_v_O2_h_mt11	19 bits	Stores MT11 O2 header

4.2 Data Reformatting

A key requirement of the MID user logic is to receive the data from the front-end system via the Core CRU firmware and reformat the header of the data to correspond to a specific location in the ALICE MID where the data originates.

The Core CRU firmware accepts the incoming payload from the FEE via the GBT optical link and presents it to the user logic as a bus consisting of is_valid, is_data and an 80 bit data

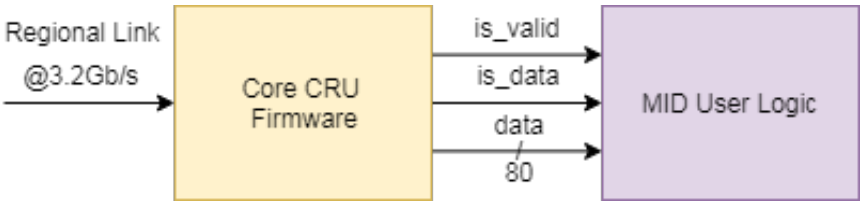


Figure 4.1: GBT Optical link to e-link bus

bus, a sub-set of the bus shown in table: 4.1. The FEE protocol contained in the 80 bit data bus contains byte fragments of the data payload from local card 0 through to 7 as well as byte fragments from the 2 internal links, regional high and low, corresponding to the regional card the local cards are connected to. The information necessary to successfully identify a unique detector location is the card ID embedded in the relevant data streams. Additionally, adjacent RPC's in each plane are connected to the same local cards.

Table 4.4: Data Reformatting Module Uplink Ports

Port Name	Clock Domain	Description
clk_i		Input clock
reset_n_i	gbt_tx_clk_i	Reset
gbt_data_valid_i	gbt_tx_clk_i	GBT valid signal
gbt_isdata_i	gbt_tx_clk_i	GBT data signal
gbt_data_i[79-0]	gbt_tx_clk_i	GBT Data Payload

The required O2 header per data set can be seen in figure: 4.2. The strip pattern for the bending and non-bending is fully described by indicating the RPC detector element where the data originates(i.e. data per plane is split), the column in that particular RPC and finally the position of the local card within that column.

DetElemID[1:72] Fired RPC	Fired Column ID	Local Card ID per Column	Fired NBP Strip Pattern	Fired BP Strip Pattern
7 Bits	8 Bits	4 Bits (max)	16 Bits	16 Bits

Figure 4.2: O2 Header Format Required [5]

Table 4.5: O2_H Uplink Ports

Port Name	Clock Domain	Description
clk_i		Input clock
reset_n_i	gbt_tx_clk_i	Reset
o2_h_valid_i	gbt_tx_clk_i	Input data valid bit
o2_h_syn_reg_id_i[3-0]	gbt_tx_clk_i	Regional card ID Bits
o2_h_syn_loc_id_i[3-0]	gbt_tx_clk_i	Local card bits

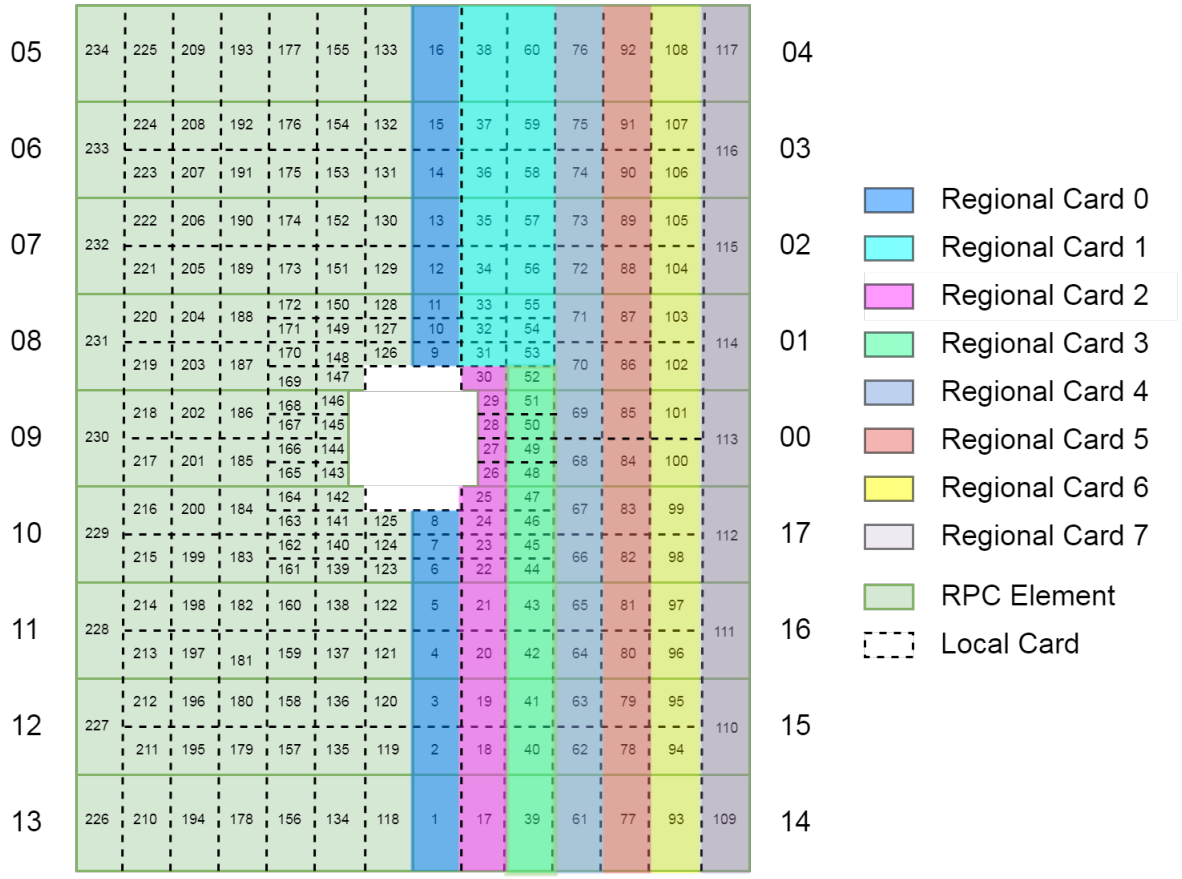


Figure 4.3: MID Detector Mapping for single plane [6]

When data extraction of the local and regional card ID's is complete the corresponding registers where the data is stored is applied to the reformatting component of the user logic (The uplink ports can be seen in table: 4.6) and a valid signal is asserted to start the process. Combinations of regional and local card ID's are then mapped to four RPC elements and also the cards location in that RPC element based on detector geometry, the segmentation of plane MT11 of the MID is shown in figure: 4.3.

Table 4.6: O2_H Downlink Ports

Port Name	Clock Domain	Description
o2_h_valid_o	gbt_tx_clk_i	Data reformatting valid bit
o2_h_mt22_o[18-0]	gbt_tx_clk_i	MT22 unique identifier bits
o2_h_mt21_o[18-0]	gbt_tx_clk_i	MT21 unique identifier bits
o2_h_mt12_o[18-0]	gbt_tx_clk_i	MT12 unique identifier bits
o2_h_mt11_o[18-0]	gbt_tx_clk_i	MT11 unique identifier bits

After successful synthesis of the O2 header each planes BP and NBP strip patterns are attached, which were stored in registers during the data extraction process. The reformatted data is then transmitted to the 2-Dimensional FIFO for data synchronization via the reformatting module's downlink ports seen in table: 4.7.

Table 4.7: Data Reformatting Module Downlink Ports

Port Name	Clock Domain	Description
gbt_rfmt_data_v_o[7-0]	gbt_tx_clk_i	Data Valid bits
gbt_rfmt_data_mt22_7_o[63-0]	gbt_tx_clk_i	LC7 MT22 Data
gbt_rfmt_data_mt21_7_o[63-0]	gbt_tx_clk_i	LC7 MT21 Data
gbt_rfmt_data_mt12_7_o[63-0]	gbt_rx_clk_i	LC7 MT12 Data
gbt_rfmt_data_mt11_7_o[63-0]	gbt_rx_clk_i	LC7 MT11 Data
gbt_rfmt_data_mt22_6_o[63-0]	gbt_tx_clk_i	LC6 MT22 Data
gbt_rfmt_data_mt21_6_o[63-0]	gbt_tx_clk_i	LC6 MT21 Data
gbt_rfmt_data_mt12_6_o[63-0]	gbt_rx_clk_i	LC6 MT12 Data
gbt_rfmt_data_mt11_6_o[63-0]	gbt_rx_clk_i	LC6 MT11 Data
gbt_rfmt_data_mt22_5_o[63-0]	gbt_tx_clk_i	LC5 MT22 Data
gbt_rfmt_data_mt21_5_o[63-0]	gbt_tx_clk_i	LC5 MT21 Data
gbt_rfmt_data_mt12_5_o[63-0]	gbt_rx_clk_i	LC5 MT12 Data
gbt_rfmt_data_mt11_5_o[63-0]	gbt_rx_clk_i	LC5 MT11 Data
gbt_rfmt_data_mt22_4_o[63-0]	gbt_tx_clk_i	LC4 MT22 Data
gbt_rfmt_data_mt21_4_o[63-0]	gbt_tx_clk_i	LC4 MT21 Data
gbt_rfmt_data_mt12_4_o[63-0]	gbt_rx_clk_i	LC4 MT12 Data
gbt_rfmt_data_mt11_4_o[63-0]	gbt_rx_clk_i	LC4 MT11 Data
gbt_rfmt_data_mt22_3_o[63-0]	gbt_tx_clk_i	LC3 MT22 Data
gbt_rfmt_data_mt21_3_o[63-0]	gbt_tx_clk_i	LC3 MT21 Data
gbt_rfmt_data_mt12_3_o[63-0]	gbt_rx_clk_i	LC3 MT12 Data

gbt_rfmt_data_mt11_3_o[63-0]	gbt_rx_clk_i	LC3 MT11 Data
gbt_rfmt_data_mt22_2_o[63-0]	gbt_tx_clk_i	LC2 MT22 Data
gbt_rfmt_data_mt21_2_o[63-0]	gbt_tx_clk_i	LC2 MT21 Data
gbt_rfmt_data_mt12_2_o[63-0]	gbt_rx_clk_i	LC2 MT12 Data
gbt_rfmt_data_mt11_2_o[63-0]	gbt_rx_clk_i	LC2 MT11 Data
gbt_rfmt_data_mt22_1_o[63-0]	gbt_tx_clk_i	LC1 MT22 Data
gbt_rfmt_data_mt21_1_o[63-0]	gbt_tx_clk_i	LC1 MT21 Data
gbt_rfmt_data_mt12_1_o[63-0]	gbt_rx_clk_i	LC1 MT12 Data
gbt_rfmt_data_mt11_1_o[63-0]	gbt_rx_clk_i	LC1 MT11 Data
gbt_rfmt_data_mt22_0_o[63-0]	gbt_tx_clk_i	LC0 MT22 Data
gbt_rfmt_data_mt21_0_o[63-0]	gbt_tx_clk_i	LC0 MT21 Data
gbt_rfmt_data_mt12_0_o[63-0]	gbt_rx_clk_i	LC0 MT12 Data
gbt_rfmt_data_mt11_0_o[63-0]	gbt_rx_clk_i	LC0 MT11 Data

4.3 Zero Suppression

In continuous reaout data is being streamed through the data acquisition chain. This data is not always valid and as such needs to be suppressed from final data readout to the O2 computing system. There are two cases when zero's are introduced.

Firstly, the GBTx's is_valid and is_data signals are always asserted when the they are operating correctly and are not indicated of the validity of the data being transmitted. As such, data extraction should only be initiated when the data being transmitted is valid. The logic responsible for this is outlined in section 4.1.

Another case in which zero data needs suppression is introduced due to the characteristics of the FEE. Data from a particular plane of a local card is only sent when both the BP and NBP of that card sent contain fired strips. Even though the FES is practically transmitting 51.3Gb/s the percentage of FEE transmitting valid data is fractional as can be seen from the results of a study done in table: 4.8. This results in the absence of data for that time period in the related FIFO and hence the AND operation produces false throwing the system out-of-sync. However, if at least one plane is sending data the tracklet information in the FEE header indicates which planes are sending data (a bit is asserted). This information can be used to mask the part of logic not expected to receive data, inject dummy data into the associated fifo and populate a detector map to reject that data in the final readout.

Table 4.8: Anticipated Dataflow to CRU[8]

Pb-Pb@(2*50 kHz)		**pp@200 kHz**
Total data flow to CRU	Mean (max) data flow per link Local -> Regional	Total data flow to CRU
3 Gbit/s (available: 100 Gbit/s)	8 (20) Mbit/s (available: 320 Mbit/s)	0.3 Gbit/s

Similarly, a local card will not transmit data if no strips are fired on any of the planes. The same process of masking the user logic of the local card is not expected to receive data for that time period can be performed, dummy data inserted and rejection in final readout using the detector map. This is possible since regional cards always send data (at least a header) and the tracklet data indicates which local cards are not transmitting data for that time period.

Another case of missing data that needs to be considered is if the optical links are faulty. The two scenarios can occur if these links are unstable or broken. The Core CRU team has suggested additional input signals to the user logic indicating if this is the case with the associated data signals. These additional input signals can be used in the same manner as explained above, namely, masking logic for that GBT block, inserting dummy data and rejecting the dummy data in the final readout.

4.4 Data Synchronization

As mentioned in previous chapters, data acquired from the detector needs to be grouped according to the time it was captured in order to perform correct tracking of particles through the detector. There are two scenarios where data becomes desynchronized. The first case happens due to transmission time variations over the optical and electrical links in the readout system. A compounded issue is that the FEE captures data at a higher frequency (using an internal bunch counter) than the time period of the heartbeat triggers. These sub-heartbeat frames of data need to be labelled with a heartbeat ID of the time period they occurred regardless of the internal bunch count.

To achieve data synchronization a 2-dimensional FIFO is used by building an array of FIFO's, where a single FIFO corresponds to data per local card per plane. An implementation of the 2D-FIFO for a single GBT link is shown in figure: 4.4. The empty signal from each FIFO is

used to determine when data has arrived from all local cards at the beginning of data taking and making use of a local AND operation for each local card from a single GBT link and global AND operation on all GBT links. When 'TRUE', synchronization has been achieved and read-out for consecutive internal bunch count can be performed moving each FIFO up one level and repeating this process.

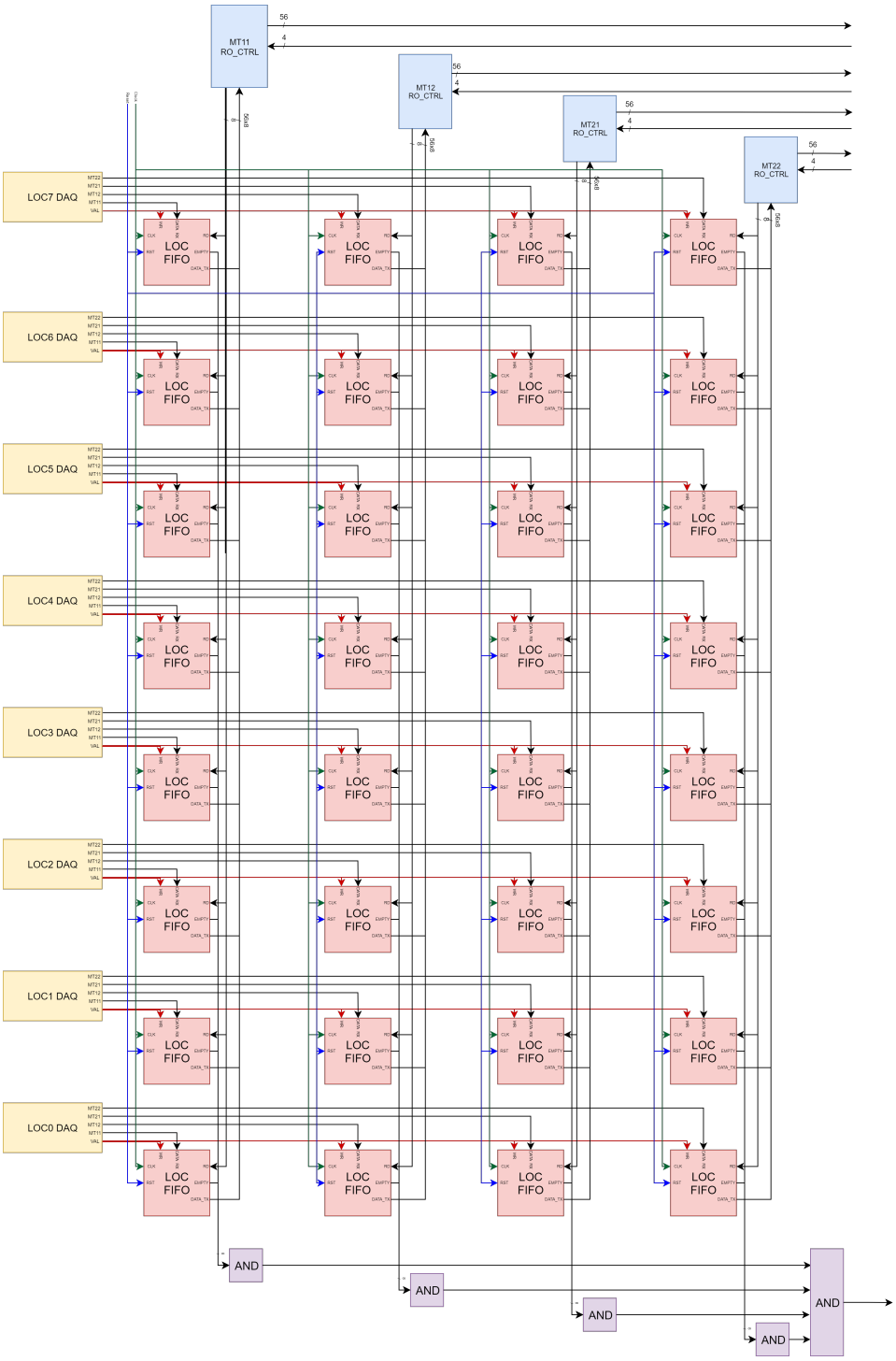


Figure 4.4: Schematic of data Synchronization Module of MID-CRU User Logic

4.5 Population of Raw Data Header

There are fields that need to be populated with information pertaining to the packet of data being transmitted to the core CRU firmware, in particular, the heartbeat trigger that produced that data and the heartbeat ID of that particular trigger.

As mentioned in previous sections, a nuance of the MID FEE is that a single heartbeat trigger produces 2^{16} consecutive data sets within one heartbeat frame. This is due the self-triggered nature of the front-end system at a higher rate that the central trigger system. This results in the same heartbeat ID and heartbeat trigger relating to all 2^{16} sets of data. Additionally, the data payload only provides the internal counter information of the associated FEE and hence the HBID needs to be determined from just that information.

A solution to correctly populating the RDH is to monitor the IBC of each local card. Since the IBC resets whenever a new heartbeat trigger is received from the CTS, this property can be used to solve this problem. By monitoring the register containing the IBC value and incrementing a HBID variable when a reset is detected the HBID of the dataset can be accurately tracked. The trigger variable from the data extraction module can be used as is. These two variables, trigger and HBID will then be populated into the raw data header before final readout for a particular time period.

4.6 Data Readout

A key requirement expressed by O2 experts working on the MID was the specific order in which they want the data readout. This order is indicated in figure: 4.5 depicting plane MT11 using red and blue arrows i.e. from left to right and bottom to top.

The designed method to achieve this is to arrange the data primarily in ascending order of local card number and secondly in ascending order of plane and store this temporarily in memory. Once this has been done, the sequential readout of this memory is done in the required order. The arrangement of the data prior to readout will be performed by a hardware implementation of a merge-sort algorithm, the details of which are explained throughout this section.

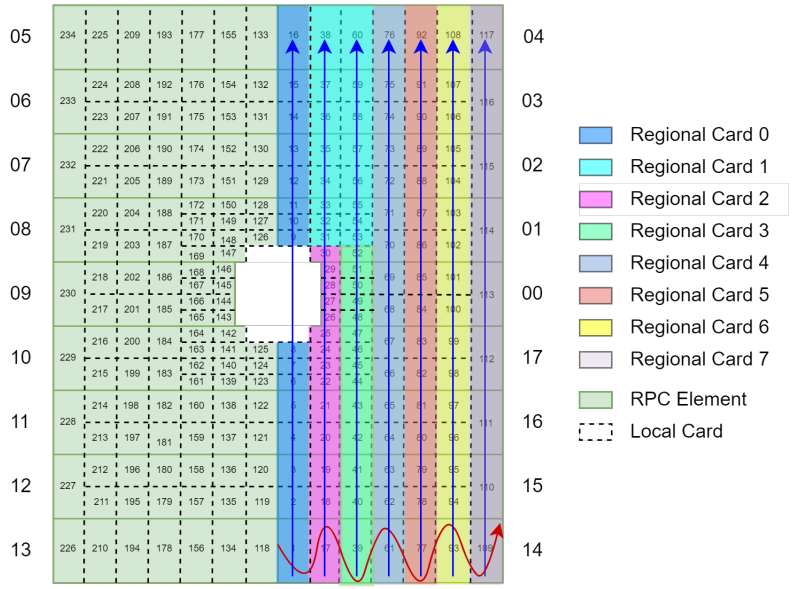


Figure 4.5: Required data readout for O2 Computing System

4.6.1 Stage 1 Readout

Data read out is initiated when synchronization has been achieved, this occurs when all the FIFO's in Stage 1 memory have the first entry containing valid data from the initiation of DAQ indicated by a global AND operation of 2D-FIFO modules. Since there are 16 GBT links containing data from 8 local cards which provide data from 4 planes reading out the Stage 1 serially would 8192 clock cycles. This can be significantly reduced by introducing a merge-sort algorithm. By merging and ordering data in memory blocks before final readout we can utilize pipelining to increase throughput. The segmentation of Stage 1 memory is shown in figure: 4.6.

Applying row and column partitioning to the problem allows 2 levels of pipelining. Each GBT 2D-FIFO bank can be readout concurrently as well as the planes in each GBT 2D-FIFO. Since there only 8 local card data sets per plane per GBT 2D-FIFO it would take only 8 clock cyles to readout the Stage 1 memory instead of 8192 clock cycles which is a significant improvement to throughput.

The readout algorithm also makes use of handshaking signals between stages. This allows constant communication between the stages indicating when they should transmit data to downstream modules or discard data from upstream modules, this process is illustrated in figure: 4.7.

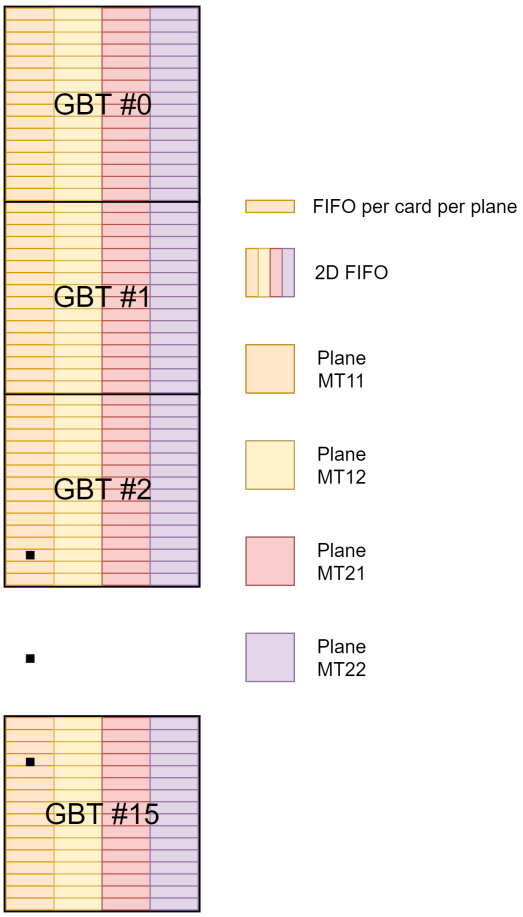


Figure 4.6: Stage 1 Memory Partitioning

These handshaking signals, based on the NI handshaking protocol [32], can be summarized as follows:

- Input valid specifying that the next data has arrived for processing.
- Output valid indicating that the current data produced by the module is valid and ready to be used by downstream modules.
- Ready for output specifying whether the downstream node can accept a new data point.
- Ready for input, indicating if the module can accept a new data during the next clock cycle.

This protocol is sometimes referred as the 4-wire protocol since it involves 4 handshaking signals[33]. Between Stage 1 and Stage 2 are gbt_s2_busy_i (Ready for output) and gbt_rfmt_data_v_o (input valid) shown in table: 4.9 and gbt_fifo_data_v_o (output valid) in table: 4.10. There is no ready for input signal since Stage 1 operates as a FIFO and should always accept new data.

Figure 4.7: Buffered handshaking

Table 4.9: Stage 1 Readout Uplink Ports

Port Name	Clock Domain	Description
clk_i		Input clock
reset_n_i	gbt_tx_clk_i	Reset
gbt_s2_busy_i	gbt_tx_clk_i	Stage 2 RO Busy

gbt_rfmt_data_v_o[7-0]	gbt_tx_clk_i	GBT payloads valid bits
gbt_rfmt_data_mt22_7_o[63-0]	gbt_tx_clk_i	GBT payload bits
gbt_rfmt_data_mt21_7_o[63-0]	gbt_tx_clk_i	GBT payload bits
gbt_rfmt_data_mt12_7_o[63-0]	gbt_rx_clk_i	GBT payload bits
gbt_rfmt_data_mt11_7_o[63-0]	gbt_rx_clk_i	GBT payload bits
...	...	...
gbt_rfmt_data_mt22_0_o[63-0]	gbt_tx_clk_i	GBT payload bits
gbt_rfmt_data_mt21_0_o[63-0]	gbt_tx_clk_i	GBT payload bits
gbt_rfmt_data_mt12_0_o[63-0]	gbt_rx_clk_i	GBT payload bits
gbt_rfmt_data_mt11_0_o[63-0]	gbt_rx_clk_i	GBT payload bits

Table 4.10: Stage 1 Readout Downlink Ports

Port Name	Clock Domain	Description
gbt_fifo_data_v_o	gbt_tx_clk_i	FIFO RO Valid Bit
gbt_fifo_sync_o	gbt_tx_clk_i	Sync Bit
gbt_fifo_data_mt22_o[63:0]	gbt_tx_clk_i	GBT MT22 data
gbt_fifo_data_mt21_o[63:0]	gbt_tx_clk_i	GBT MT21 data
gbt_fifo_data_mt12_o[63:0]	gbt_tx_clk_i	GBT MT12 data
gbt_fifo_data_mt11_o[63:0]	gbt_rx_clk_i	GBT MT11 data

4.6.2 Stage 2 Readout

Since 2 regional links correspond to the same regional crate i.e. local cards 0-15 dual-port RAM is an obvious choice as the base module for Stage 2 memory. This enables the merging of these data sets from the local cards per plane without the need for two separate RAM modules. The advantage of which is optimized resource utilization as well avoiding additional buffering introduced by another RAM module. The segmentation of the Stage 2 memory is illustrated in figure: 4.8.

Stage 2 readout initiates when Stage 1 asserts the input valid handshaking signal i.e. s12_data_v_i this in turn results in Stage 2 deasserting the ready for input handshaking signal i.e. s2_s3_busy_o. Even though s2_s3_busy_o has been deasserted the data being transmitted from Stage 1 remains valid until that cycle of memory readout has completed in which case s12_data_v_i is deasserted. Stage 1 will reassert the input valid signal once the data payload

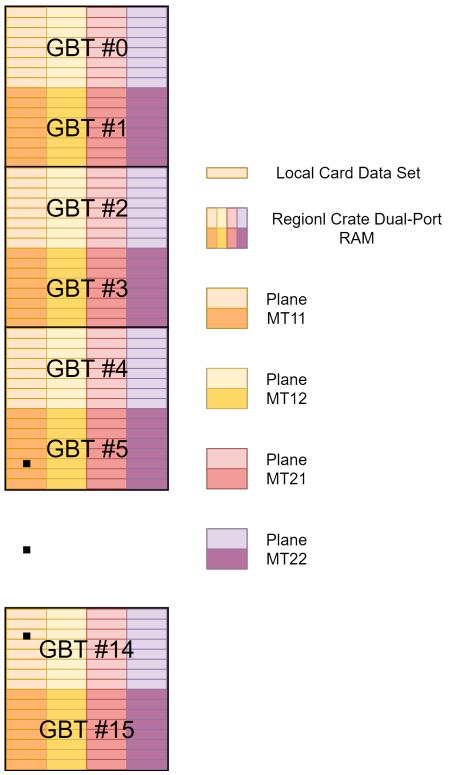


Figure 4.8: Stage 2 Memory Partitioning

transmitted to Stage 2 memory and been successfully transmitted to Stage 3 memory at which point s2_s3_busy_o is reasserted restarting Stage 1 - Stage 2 readout for the next data payload. This procedure is depicted in figure: 4.7. The details of handshaking signals can be seen in table: 4.11 and table: 4.12.

Table 4.11: Stage 2 Readout Uplink Ports

Port Name	Clock Domain	Description
clk_i		Input clock
reset_n_i	gbt_tx_clk_i	Reset
s12_data_v_i	gbt_tx_clk_i	Input ready
reg_7_h_mt22_data_i[63:0]	gbt_tx_clk_i	Reg 7 Plane 4 Data
reg_7_h_mt21_data_i[63:0]	gbt_tx_clk_i	Reg 7 Plane 3 Data
reg_7_h_mt12_data_i[63:0]	gbt_tx_clk_i	Reg 7 Plane 2 Data
reg_7_h_mt11_data_i[63:0]	gbt_tx_clk_i	Reg 7 Plane 1 Data
reg_7_l_mt22_data_i[63:0]	gbt_tx_clk_i	Reg 7 Plane 4 Data
reg_7_l_mt21_data_i[63:0]	gbt_tx_clk_i	Reg 7 Plane 3 Data
reg_7_l_mt12_data_i[63:0]	gbt_tx_clk_i	Reg 7 Plane 2 Data
reg_7_l_mt11_data_i[63:0]	gbt_tx_clk_i	Reg 7 Plane 1 Data
...	...	...

reg_0_h_mt22_data_i[63:0]	gbt_tx_clk_i	Reg 0 Plane 4 Data
reg_0_h_mt21_data_i[63:0]	gbt_tx_clk_i	Reg 0 Plane 3 Data
reg_0_h_mt12_data_i[63:0]	gbt_tx_clk_i	Reg 0 Plane 2 Data
reg_0_h_mt11_data_i[63:0]	gbt_tx_clk_i	Reg 0 Plane 1 Data
reg_0_l_mt22_data_i[63:0]	gbt_tx_clk_i	Reg 0 Plane 4 Data
reg_0_l_mt21_data_i[63:0]	gbt_tx_clk_i	Reg 0 Plane 3 Data
reg_0_l_mt12_data_i[63:0]	gbt_tx_clk_i	Reg 0 Plane 2 Data
reg_0_l_mt11_data_i[63:0]	gbt_tx_clk_i	Reg 0 Plane 1 Data

Table 4.12: Stage 2 Readout Downlink Ports

Port Name	Clock Domain	Description
s23_data_v_o[3:0]	gbt_tx_clk_i	S2 Data Ready Bit
s2_s3_busy_o	gbt_tx_clk_i	s2 Ready for Data bit
s2_ram_data_mt22_o[63:0]	gbt_tx_clk_i	Plane 4 Data
s2_ram_data_mt21_o[63:0]	gbt_tx_clk_i	Plane 3 Data
s2_ram_data_mt12_o[63:0]	gbt_tx_clk_i	Plane 2 Data
s2_ram_data_mt11_o[63:0]	gbt_tx_clk_i	Plane 1 Data

4.6.3 Stage 3 Readout

Stage 3 readout is responsible for merging the data payload from the 8 regional crates from the same plane (MT11, MT12, MT21, MT22) into a single block of memory, depicted in figure: 4.9. The 4 data output signals connect to the 4 data input siganls of Stage 3 as well as 2 handshaking signals, more details on these ports is available in tables 4.13 and 4.14.

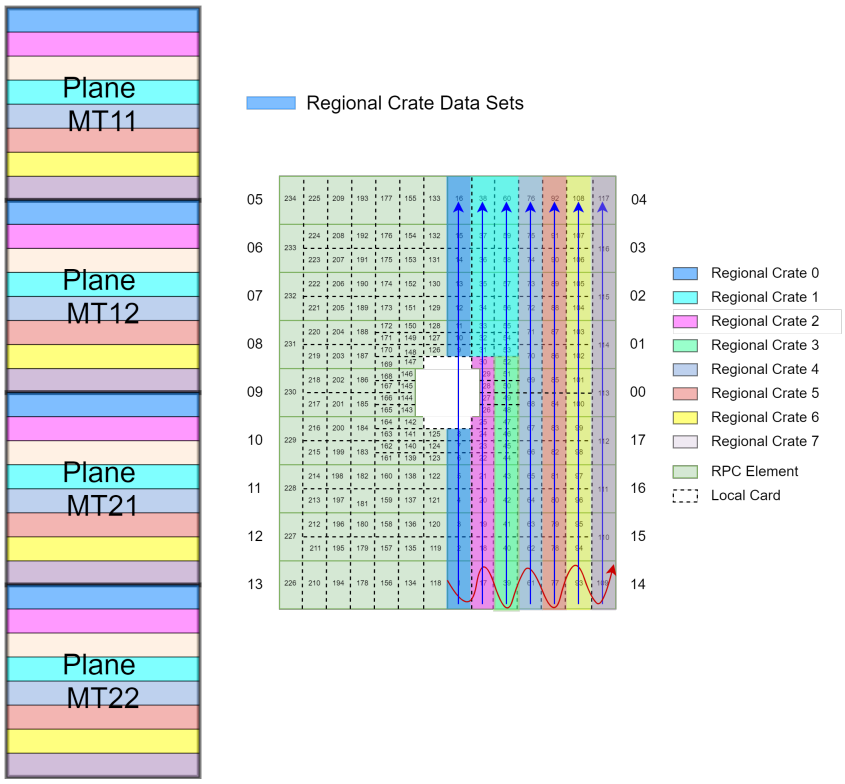


Figure 4.9: Stage 3 Memory Partitioning

Similiarly to Stage 2, Stage 3 readout initiates when it's s23_data_v_i signal (input valid) is asserted by Stage 2 while it's s3_busy_o signal is asserted (ready for input). The 4 columns of the Stage 2 memory are readout concurrently storing the data sets of all the local cards of each regional crate in ascending order in 4 separate RAM modules. Once this process is complete s23_data_v_i is deasserted, s3_ram_data_v_o asserted and the contents of the 4 RAM modules are readout consecutively in order of plane. When Stage 3 readout is complete s23_data_v_i is asserted, s3_busy_o and s3_ram_data_v_o both deasserted indicating that Stage 3 is ready for the next payload. This process is depicted in figure: 4.7.

Table 4.13: Stage 3 Readout Downlink Ports

Port Name	Clock Domain	Description
clk_i		Input clock
reset_n_i	gbt_tx_clk_i	Reset
s23_data_v_i[3:0]	gbt_tx_clk_i	S3 Data Ready Bits
s3_ram_data_mt22_i[63:0]	gbt_tx_clk_i	Plane 4 Data
s3_ram_data_mt21_i[63:0]	gbt_tx_clk_i	Plane 3 Data
s3_ram_data_mt12_i[63:0]	gbt_tx_clk_i	Plane 2 Data

s3_ram_data_mt11_i[63:0]	gbt_tx_clk_i	Plane 1 Data

Table 4.14: Stage 3 RO Downlink Ports

Port Name	Clock Domain	Description
s3_busy_o	gbt_tx_clk_i	S3 RO busy bit
s3_ram_data_v_o	gbt_tx_clk_i	S3 Data Valid bit
s3_ram_data_o[63-0]	gbt_tx_clk_i	S3 RO Data

4.7 Full Configuration

The modules outlined in this chapter form base components for the designed MID User Logic. A simplified depiction of how they are configured is shown in figure: 4.10. Each reformatting and synchronization modules corresponds to a single GBT link and as such there are 16 blocks of these. The stage 2 readout module merges and sorts the data payload for a single internal bunch count according to the regional crate the data came from. Finally stage 3 readout merges and sorts stage 2 data according to the plane that the data came from and reads it out on a single bus consecutively.

Tables 4.15 and 4.16 explain the input and output ports of the full system.

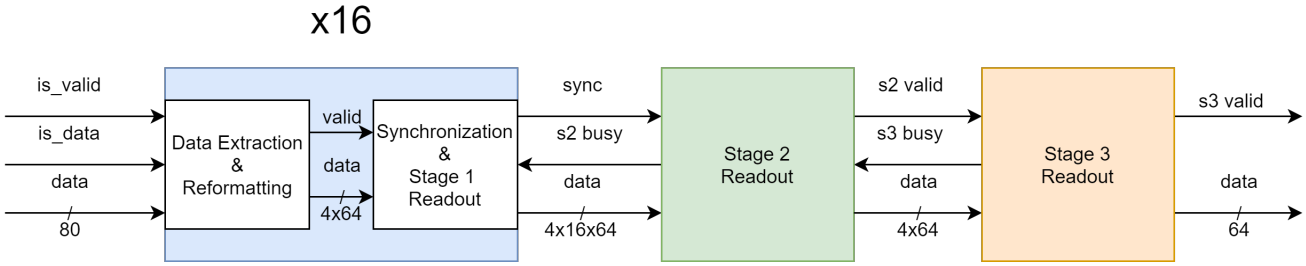


Figure 4.10: Full Configuration

Table 4.15: MID-CRU Uplink Ports

Port Name	Clock Domain	Description
mid_cru_clk_i		Input clocks
mid_cru_valid_i[15:0]	gbt_tx_clk_i	MID-CRU payload valid bits

mid_cru_isdata_i[15:0]	gbt_tx_clk_i	MID-CRU payload isdata bits
mid_cru_data_i[1279:0]	gbt_rx_clk_i	MID-CRU payload bits

Table 4.16: FE-CRU GBT Downlink Ports

Port Name	Clock Domain	Description
mid_cru_valid_i	gbt_tx_clk_i	MID-CRU payload valid bit
mid_cru_data_i[63:0]	gbt_tx_clk_i	MID-CRU payload data bits

Chapter 5

System Evaluation and Results

System evaluation is performed using a simple but effective method. A hierarchical testing procedure is used whereby sub-modules are tested until the top level entity. A file containing bit vectors is imported into a stimuli generator contained in the entity testbench file and applied to the input ports of the device under test (DUT). The output ports of the DUT is sampled then written to an output file using a process and system evaluation is performed by inspection. This procedure is illustrated in the figure: 5.1. The source code for this project can be found in Appendix B.

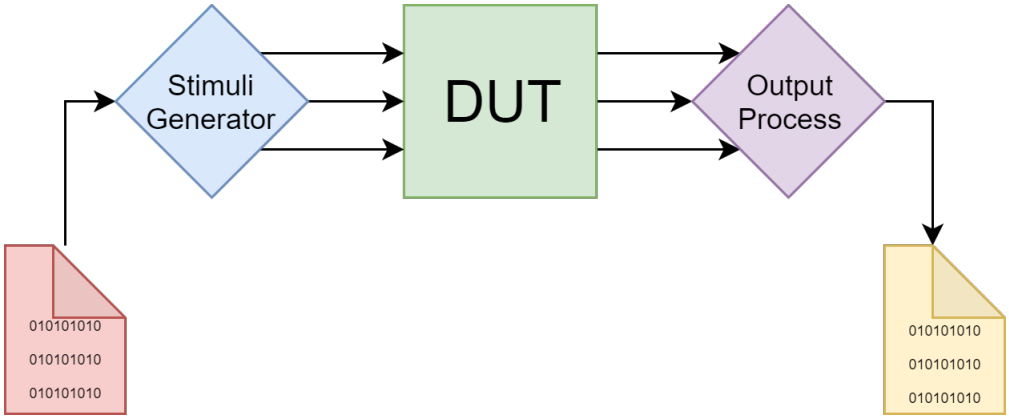


Figure 5.1: Design Verification

5.1 Evaluation of Data Extraction

The data protocol of the front-end cards is such that the 80 bit field of the GBT link is comprised of 10 bytes containing the data payload from the 8 local cards connected to the that

regional card and two internal links the card, for more detail about this protocol is described in section 2.1.1 in particular figure: 2.3. On every clock cycle, a byte of the data payload from each card is transmitted to the CRU.

Using the FEE data formatted as a reference, expected input data is simulated by reading a text file in the HDL testbench for this module and driving it into the module via a stimuli generator. The data used can be seen in figure: 5.2.

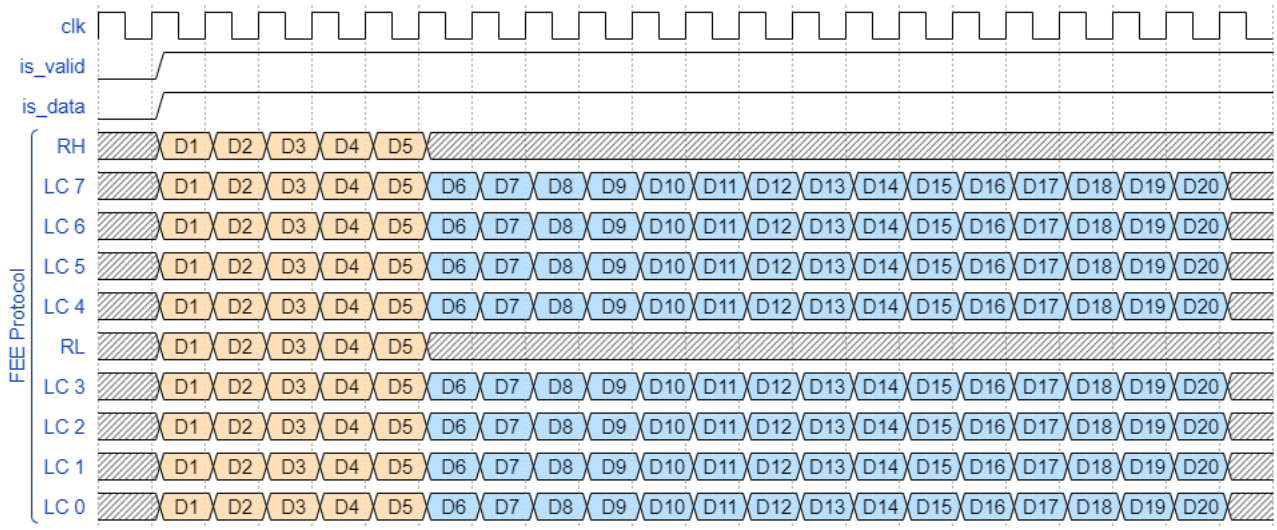


Figure 5.2: Timing diagram depicting test data being driven through data extraction input ports

Data extraction is achieved by storing these bytes in unique registers for use in other modules of the user logic, the details on these registers can be studied in table: 4.3. Observing the registers for data from local card 7 in table: 5.1 it can be seen that on every clock when valid data is being transmitted, these registers are being populated hence operating as designed ('X' represents a 'don't care' condition).

In total 9 registers are used per local card. Assuming 128 local cards for simplicity, this totals 1152 registers in the user logic.

This module has zero latency since the data is stored in these registers on the same clock it is sampled at the input ports.

Table 5.1: Local Card 7 Data storage in variables

CLK	stat	trig	ibc	cid	trck	mt11_sp	mt12_sp	mt21_sp	mt22_sp
1	D1	XX	XXXX	X	X	XXXXXXXX	XXXXXXXX	XXXXXXXX	XXXXXXXX
2	D1	D2	XXXX	X	X	XXXXXXXX	XXXXXXXX	XXXXXXXX	XXXXXXXX
3	D1	D2	D3XX	X	X	XXXXXXXX	XXXXXXXX	XXXXXXXX	XXXXXXXX
4	D1	D2	D3D4	X	X	XXXXXXXX	XXXXXXXX	XXXXXXXX	XXXXXXXX
5	D1	D2	D3D4	D	5	XXXXXXXX	XXXXXXXX	XXXXXXXX	XXXXXXXX
6	D1	D2	D3D4	D	5	D6XXXXXX	XXXXXXXX	XXXXXXXX	XXXXXXXX
7	D1	D2	D3D4	D	5	D6D7XXXX	XXXXXXXX	XXXXXXXX	XXXXXXXX
8	D1	D2	D3D4	D	5	D6D7D8XX	XXXXXXXX	XXXXXXXX	XXXXXXXX
9	D1	D2	D3D4	D	5	D6D7D8D9	XXXXXXXX	XXXXXXXX	XXXXXXXX
10	D1	D2	D3D4	D	5	D6D7D8D9	D10XXXXXX	XXXXXXXX	XXXXXXXX
11	D1	D2	D3D4	D	5	D6D7D8D9	D10D11XXXX	XXXXXXXX	XXXXXXXX
12	D1	D2	D3D4	D	5	D6D7D8D9	D10D11D12XX	XXXXXXXX	XXXXXXXX
13	D1	D2	D3D4	D	5	D6D7D8D9	D10D11D12D13	XXXXXXXX	XXXXXXXX
14	D1	D2	D3D4	D	5	D6D7D8D9	D10D11D12D13	D14XXXXXX	XXXXXXXX
15	D1	D2	D3D4	D	5	D6D7D8D9	D10D11D12D13	D1415XXXX	XXXXXXXX
16	D1	D2	D3D4	D	5	D6D7D8D9	D10D11D12D13	D14D15D16XX	XXXXXXXX
17	D1	D2	D3D4	D	5	D6D7D8D9	D10D11D12D13	D14D15D16D17	XXXXXXXX
18	D1	D2	D3D4	D	5	D6D7D8D9	D10D11D12D13	D14D15D16D17	D18D19D20D21
19	D1	D2	D3D4	D	5	D6D7D8D9	D10D11D12D13	D14D15D16D17	D18D19D20D21
20	D1	D2	D3D4	D	5	D6D7D8D9	D10D11D12D13	D14D15D16D17	D18D19D20D21
21	D1	D2	D3D4	D	5	D6D7D8D9	D10D11D12D13	D14D15D16D17	D18D19D20D21

5.2 Evaluation of Data Reformatting

Data reformatting is achieved in 2 stages, namely, synthesizing the O2 header that uniquely identifies where the data payload originates in the MID detector and attaching the relevant data payload to it's respective header.

After the process of data extraction, the FEE payload has been temporarily stored in the registers outlined in tables 4.2 and 4.3. The local and regional card data is driven through a module responsible for ascertaining the unique location the data originates from LUT's and providing 4 O2 headers, one for each plane (see section 4.2 for more detail). Once the header's are available, their data payloads for each plane are attached to them as they are still being store in registers. This results in an effective mapping from the FEE data format to the required O2 data format.

Testing of the module responsible for synthesizing the O2 header is performed using all possible input combinations of local and regional cards.These input combinations and the respective outputs are tabulated in Appendix A table: A.1.

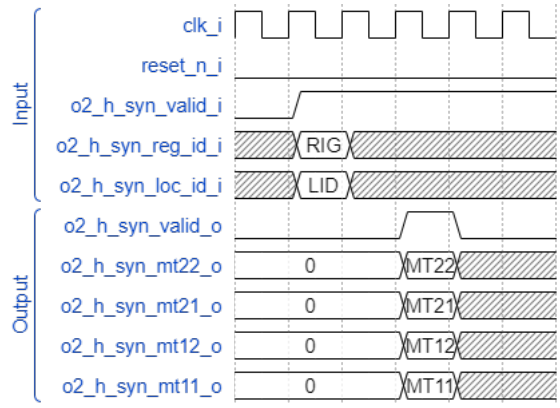


Figure 5.3: Timing diagram depicting test data being driven through O2 header synthesis input ports and output

The results show accurate synthesis of a new O2 header and output data of this module produces 4 sets of data indicating correct operation of the data reformatting module.

A latency of 1 clock cycle occurs in this module due to signals updating on positive clock edges.

5.3 Evaluation of Data Synchronization

In order to evaluate whether data synchronization has been performed correctly in the data synchronization module i.e. the 2D FIFO, data is injected into the ports of this module at varying clock cycles.

The newly reformatted data payload, 64 bits in length, for the 8 local cards and 4 planes are read from a text file and driven into the ports of the 2D FIFO module via a stimuli generator. The data payloads for these cards, however, are injected at different times, simulating anticipated transmission delays (explained in section 4.4). This testing procedure can be seen in figure: 5.4.

Correct data synchronization is evaluated by inspection of output files produced by the HDL testbench of the 2-D FIFO module. The 2-D FIFO is designed to output the data from each plane independently due to column partitioning design methodology and an in sync. Additionally each sub-fifo per plane is read consecutively on each clock cycle while writing to half the memory of an s2 ram module with the other half been written to by the 2-D fifo module corresponding to the second regional card on the same regional crate in parallel.

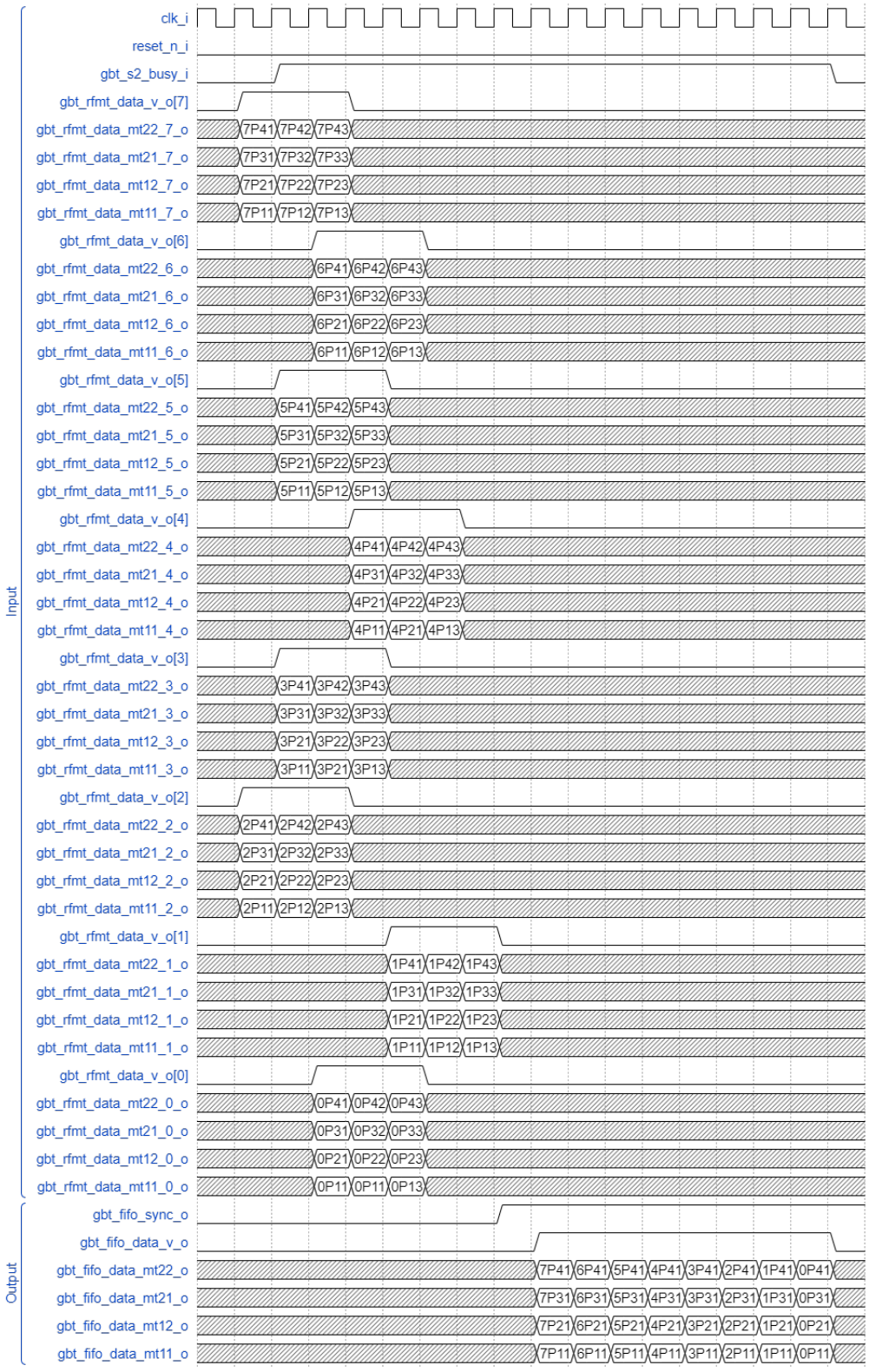
The data contained in figure: 5.4 shows this occurring as expected from the 11 clock cycle to the 18th clock cycle. A subset of the actual data collected from the HDL testbench can be viewed in table: A.2.

The Arria 10 provides two main memory resources, namely: M20K and MLAB's. Due to the limited amount of M20K memory as well the relatively large number of memory blocks needed for the 2D modules, a logical design decision was to make use of the MLAB memory available. Each 2D FIFO module makes use 32 memory blocks comprised of 4Mb (8 per plane corresponding to 8 local cards). Since there are 16 GBT links per CRU the total amount of memory used is calculated as follows:

$$\begin{aligned} Memory\ Used &= (16\ GBT\ links) * (32\ FIFO\ modules) * (64\ words\ depth) * (64\ bits) \\ &= 0.262144MB \end{aligned}$$

Although not explicitly mentioned in section 2.1.4, the Arria 10 provides 1.6 MB of MLAB memory, hence the memory used in for data synchronization i.e. 0.262144 MB accounts for 16.5% of the MLAB memory resources available. Additionally, a depth of 64 words was chosen arbitrarily during development and 8 words would most likely meet the requirements for syn-

Figure 5.4: Timing diagram depicting test data being driven through synchronization module input ports and output



chronization reducing resource usage nearly by a factor of 3 to 5.13%.

A rudimentary calculation of latency is performed by calculating the difference between the time at which synchronization (first data payload) occurs and the time at which the final data set from the first clock cycle is transmitted out the 2-D FIFO.

Figure: 5.4 shows that the data payload from local card 4 is injected last into the module at clock cycle 4 and the synchronized data being transmitted at clock cycle 18. Hence, there is a latency of 14 clock cycles from input to output i.e. $\sim 58ns$

5.4 Evaluation of Readout

Readout of the data payload is performed using a merge-sort algorithm. In actual readout, the final stage of readout organizes the data in the order required by the the O2 system. To display the results of the implementation for this MSc. the data is shown in ascending order of local card, graphically. The results however, still remain valid as simply altering the memory addresses read out does not affect system performance. A detailed description of the readout architecture is available in section 4.6.

5.4.1 Stage One Readout

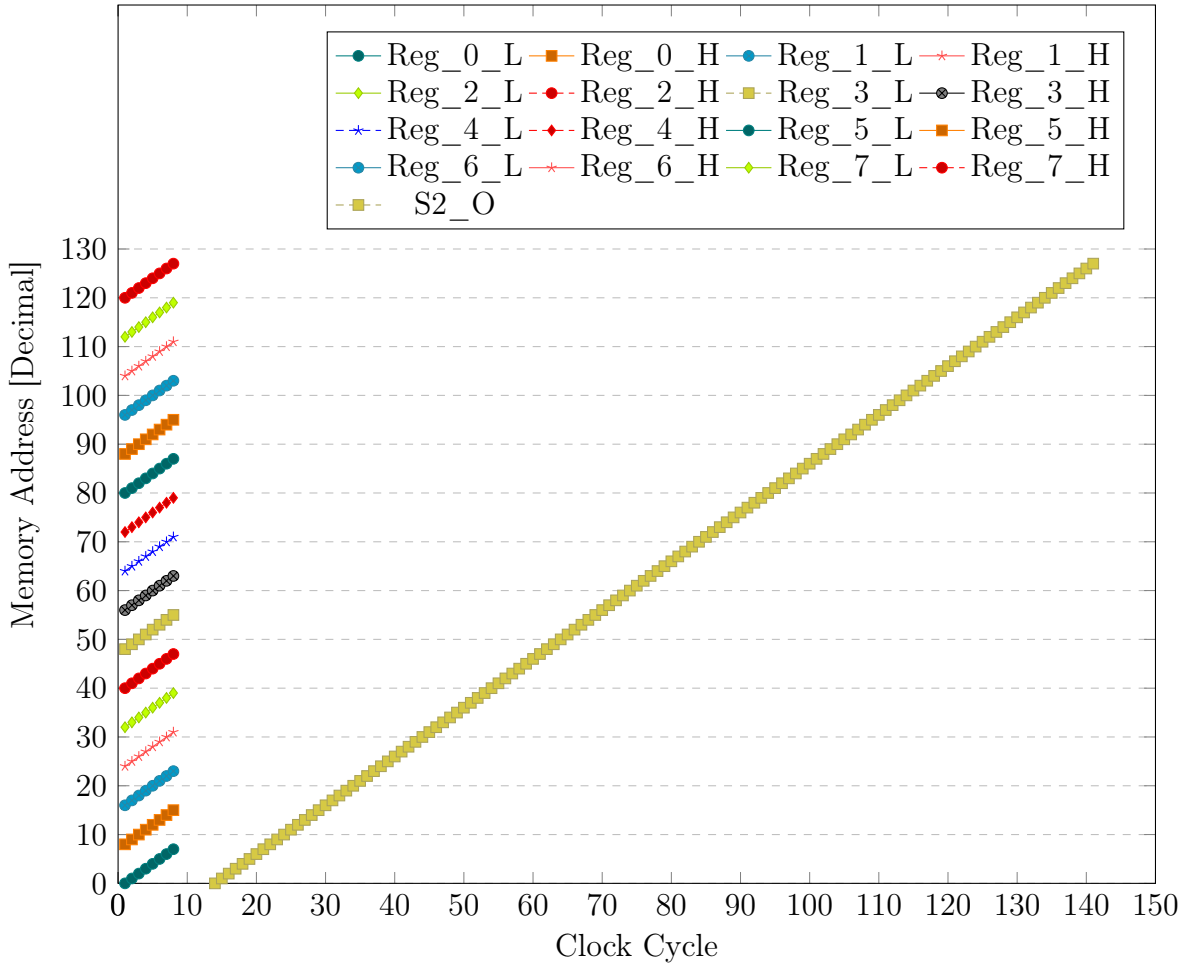
Stage 1 readout is performed once synchronization has been achieved in the 2-D FIFO modules. All 16 2-D FIFO modules need to be in sync for this process to start, once that occurs the first word per local card in each plane is read consecutively. The data read from the two 2D FIFO modules corresponding to the same regional crate are written to the top and bottom half of a single dual-port RAM module.

Data obtained from the output files of the testbench for the 2-D FIFO module is depicted graphically in the timing diagram in figure: 5.6. As can be seen in the figure, the valid signal is asserted as the expected data is been readout from the 11 clock cycle.

5.4.2 Stage Two Readout

Stage 2 readout receives the data payload for half the MID detector. The data payload is transmitted via 16 busses corresponding to the 16 GBT links. The two busses corresponding to

Figure 5.5: Stage 1 output data from 16 2D-FIFO modules from a single plane being written to segments Stage 2 memory and readout via a single bus



the same regional crate are routed to a dual-port RAM module merging the data from each of the two 2D FIFO's, sorting the data in ascending local card order, per plane. Each column in the Stage 2 memory is readout concurrently, resulting in 4 busses (for each plane) transmitting data sets for local cards 0 to 128 in ascending order for a single internal bunch count depicted in figures: 4.8 and 5.5. Once the processing of merging and sorting the data payload is complete, the start of stage 3 readout is initiated using handshaking signals. The S2 RAM module's 4 columns are then readout in parallel from local 0 to 128 consecutively.

This process can be seen in figure: 5.5. Considering the the individual dual-port RAM modules of a column (plane) as sections of a larger 16-port RAM module, the figure shows the data payload for a plane been written to all the segments of the 16-port RAM merging the data from local card 0 to 128 local cards in ascending order.Once this process is complete, stage 3 readout is initiated using handshaking signals.

The transmission of data from Stage 1 memory to Stage 3 memory is facilitated by handshaking signals. Referring to figure: 5.6, the process is as follows:

- Uplink data transmission is initiated when the s12_data_v_i signal (input valid) is asserted by stage 1 logic along with the data payload
- The s2_s3_busy_o (ready for input) signal is asserted indicating it is not ready for new data from the downstream module (besides the data payload currently being transmitted)
- s12_data_v_i signal (input valid) is de-asserted by stage 1 logic when payload is transmitted
- When Stage 2 data is ready for transmission stage 3 s2_s3_data_v_o (output valid) is asserted while the data payload is transmitted
- s2_s3_busy_i (ready for output) is asserted by Stage 3 logic while data is transmitted from Stage 2 until Stage 3 has been readout.

Stage 2 readout performs a read operation consecutively from the first to last memory address of each column in parallel. This process can be observed operating correctly in figure: 5.5. There is however a delay between input and output introduced by buffering caused by the first RAM module. Subsequent buffering is compensated for in the source code.

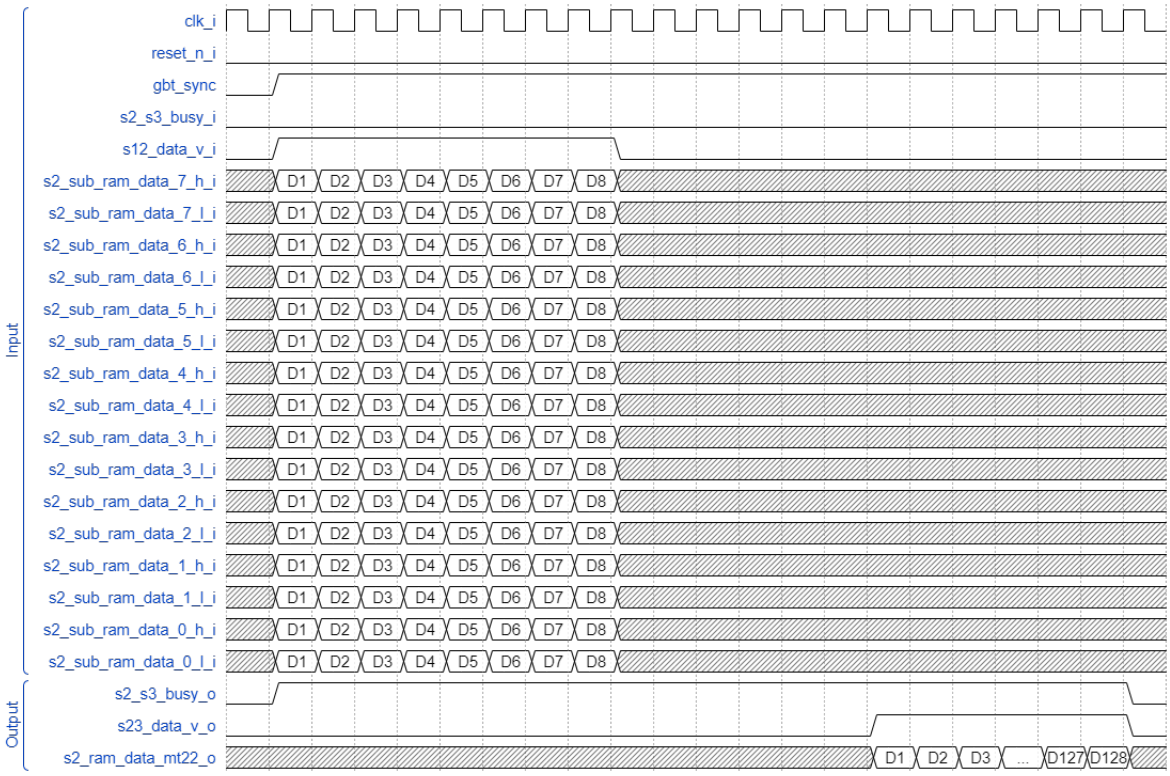
The S2 RAM module is implemented using the M20K internal memory of the Arria 10 FPGA. Using this memory in dual-port mode, accommodating the 16 GBT links and 4 planes results in 32 dual-port RAM modules. As the RAM modules require a width of 64 bits and a depth of 16 words (local card 1 to 16) the memory usage can be calculated as follows:

$$\begin{aligned} Memory\ Used &= (32\ Dual-Port\ RAM\ modules) * (16\ words\ depth) * (64\ bits) \\ &= 0.004096\ MB \end{aligned}$$

Considering, that M20K makes up ∼7MB of internal memory, the Stage 2 readout module utilizes 0.06% of available memory.

The latency of the S2 RAM module is calculated from time the last data set is written to it to the time the last data set is read. This occurs at the 8th and 141st clock cycle respectively. The latency is therefore 133 clock cycles or $\sim 554ns$.

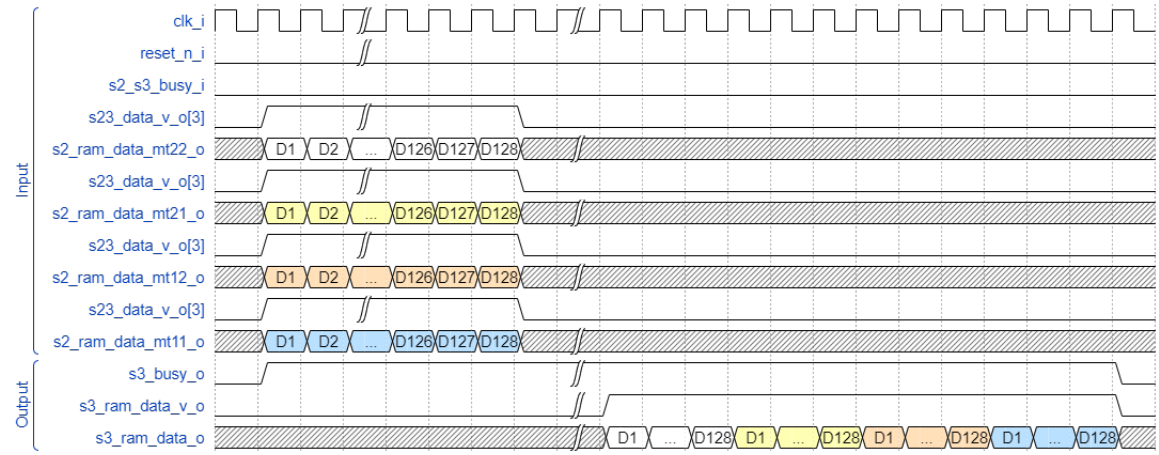
Figure 5.6: Timing diagram depicting test data being driven through Stage 2 readout input ports and output



5.4.3 Stage Three Readout

Stage 3 readout receives the data payload from stage 2 of all 4 planes simultaneously temporarily writing them to single-port RAM modules. Once this process is complete the RAM modules for each plane are read out consecutively merging the data into one data stream and ordering them in from plane MT11 to M22.

Figure 5.7: Stage 3 Readout merging Stage 2 Readout data from 4 S2-RAM module output busses from 4 planes into a single output bus



Data obtained from the output file for the testbench of the s3 RAM module is presented graphically in figure: 5.8. Considering the S3 RAM module as a single 4-port RAM, the figure shows the data payload from each column (plane) of S2 module being written to the 4 segments of the S3 RAM in sync. After the anticipated RAM buffering from the first RAM module, the entire S3 memory is read linearly from bottom to top on each clock cycle operating as desired.

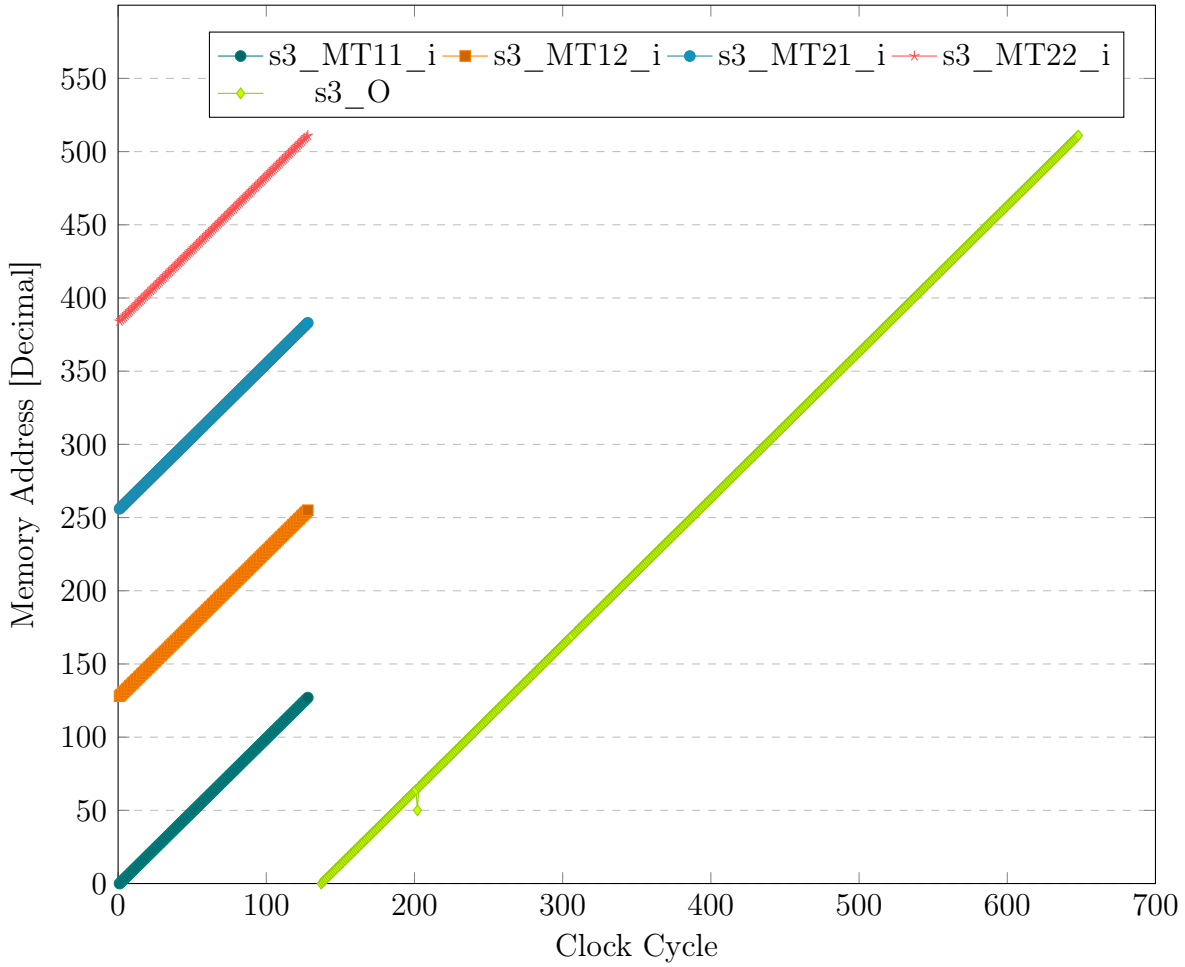
The S3 RAM module similarly to the S2 RAM module implemented using M20K memory. Using 4 blocks of memory in single-port mode, interfaced to the 4 output ports of the S2 module the memory usage can be calculated as follows:

$$\begin{aligned} Memory\ Used &= (4\ Single - Port\ RAM\ modules) * (128\ words\ depth) * (64\ bits) \\ &= 0.004096\ MB \end{aligned}$$

Again, noting that M20K makes up ∼7MB of internal memory, the Stage 3 readout module utilizes 0.06% of available memory

The latency of the s2 RAM module is calculated from the time the last data set is written to it to the time the last data set is read. This occurs at the 128th and 648th clock cycle respectively. The latency is therefore 520 clock cycles or $\sim 2167ns$.

Figure 5.8: Stage 2 output data from 4 Stage 2 memory modules from a single plane being written to segments Stage 3 memory and readout via a single bus



5.5 Evaluation of Hardware Testbench

As indicated in section 3.3 there is a need to test the user logic realistically with the hardware it is expected to operate with in the DAQ chain.

During the course of this MSc the relevant hardware was sourced from Subatech (Nantes, France, CERN (Geneva, Switzerland) and iThemba LABS (Cape Town, South Africa) as well as the knowledge and skills required to configure the complete testbench. However, at the time of submission of this dissertation the CRU board was not yet delivered to iThemba LABS. The Arria 10 is being used as an alternative until the board is available.

The hardware testbench was successfully set up using the CTP emulator available with the core CRU software in RAW data mode (no user logic) and data was successfully transmitted from the MID-Proto board to the O2 System. Initial documentation indicated connection of the LTU to the Arria 10 FPGA development kit (CRU) via the SFP interface, however, this functionality is not currently available in the core CRU firmware and could possibly be removed. This would require the CRU board for the LTU to be connected to the rest of the hardware testbench. Delivery date of the CRU board is unclear dependent on the production in India and France. The due date fell outside of the scope of this project. The hardware topology is illustrated in figure: 5.9.

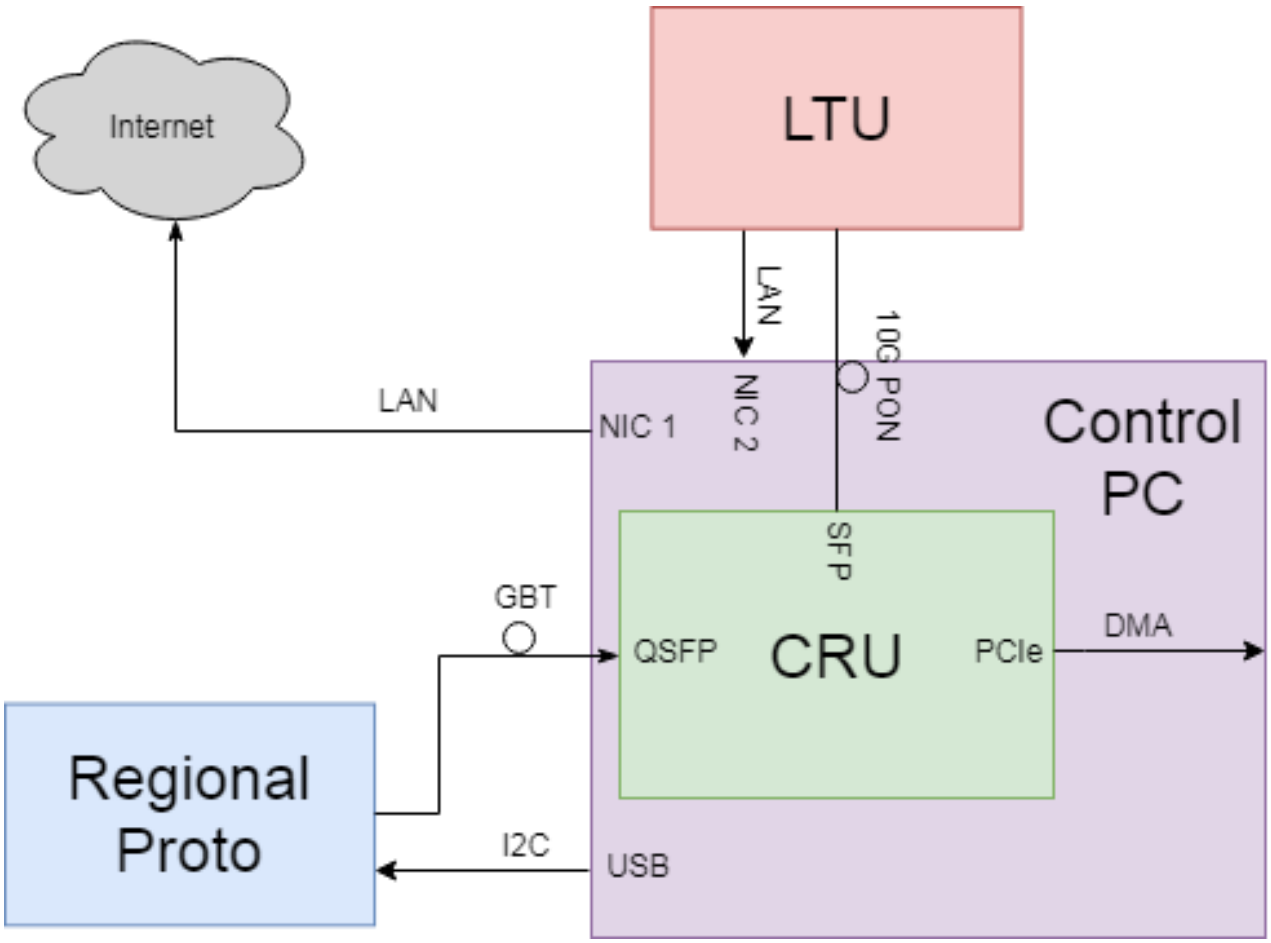


Figure 5.9: Readout Hardware of the MID Testbench

Chapter 6

Conclusions and Future Work

6.1 Conclusions

The Muon Identifier user logic architecture proposed in this work adequately addresses the challenges associated with high-throughput data steaming outlined in Chapter 2. The proposed architecture can effectively extract, reformat, synchronize and readout data transmitted from the MID detector, performing zero suppression and data reduction as required. Although there is still prototyping work needed, refer to chapter 6.2 for more detail, the architecture that was developed and tested shows a functionally sufficient design well within resource constraint limits that can be developed within the time constraints stipulated.

The modularized design employed provides two distinct advantages by leveraging parameterization. Firstly, it can be scaled to test any number of GBT links significantly reducing adaptation time for various testing methodologies and secondly allowing easy adaptation for other detectors with similar requirements by allowing developers easy access to individual components. Additionally, more features can be easily included in the architecture, such as clustering algorithms.

The testing methodology developed in section 3.3 provides a framework for realistically testing the user logic in all design and development stages. By adapting the analysis framework used by physicists for use as MID emulator, a method for continuous maintenance and improvement is also available without an immediate need for testing on the actual MID detector. The readout hardware testbench (shown in section 5.5) configured at iThemba LABS also provides future research and training opportunities.

6.2 Future Work

The prototype developed during this MSc. essentially represents a data routing network with the exception of the data reformatting. As such, future work should focus the architectural development described in Chapter 4 and the development of additional features.

6.2.1 Operating Scenarios

The proof of concept performed in this MSc. required some simplification to reduce it's scope within acceptable parameters. This was achieved by evaluating system performance and conformance to requirements considering just one particular case i.e. a front-end test. Therefore, additional architecture should address the requirements for the various other cases. This section will attempt to elaborate on these cases and possible solutions.

The defining characteristic of a front-end test is that all strips from both the bending and non-bending plane for the entire detector are sending data. In reality, only a small percentage of the readout cards are actually supplying data at any given time period and therefore zero suppression needs to be performed on regional and local links as well as planes not sending data [3] refer to section 4.3 for more detail.

6.2.2 Synthesis of Raw Data Header

The Core CRU firmware requires a header known as RDH and depicted in figure:2.5, to precede the data payload transmitted from the user logic. The two key fields that need to be populated are the trigger and heartbeart ID associated with the data in question. The designed architecture for achieving this is outlined in section 4.5. Since the trigger data is already available in it's respective memory, this needs only to be written to the RDH RAM module, The HBID however, needs additional architecture to be ascertained.

Data acquisition in the Muon Identifier occurs at a higher frequency than the heartbeat period, transmitting data in periods called a sub-heartbeat frame. This period of time is described using an internal bunch counter in the data payload. As such, no information about the actual heartbeat ID is given and can only be determined in the user logic by monitoring resets on this internal bunch counter since this is the effect heartbeat triggers have on the front-end electronics. This logic needs to be developed in future work.

6.2.3 Packetization and Transmission

The scope covered by this work spans the data acquisition from extraction of data transmitted by the core CRU firmware to the processed data transmitted by the end out the readout algorithm. It does not consider the packetization of the data (explained in sections 2.1.4 and 4.6) ,including the RDH, required by the CRU firmware. Consequently, it also does not cover the interfacing of the user logic to the core CRU firmware downlink via packetizing logic and the subsequent data transmission. This functionality needs to be implemented in future work. Refer to the CRU Readout Protocol in section 2.1.5

6.2.4 Future Developments

At the end of the work contained in the scope of this MSc. some developments took place which should be highlighted for future work. Data quality assurance discussions took place regarding technical problems which may arise during data acquisition which are considered in section 4.3. However, how the core CRU firmware will provide this functionality was not established. Additionally, discussions with the MID O2 team, established that the original O2 data format is insufficient since it does not contain a field for internal bunch counter. Both the mechanism for quality assurance and a new O2 data format should be investigated in future work.

Appendix A

Data from HDL Testbenches

Table A.1: Output from O2_h Module

Reg	Loc	O2 Header MT11	O2 Header MT12	O2 Header MT21	O2 Header MT22
0000	0000	0001110100000000001	0100000100000000001	0110010100000000001	1000100100000000001
0000	0001	0001111100000000001	0100001100000000001	0110011100000000001	1000101100000000001
0000	0010	0001111100000000010	0100001100000000010	0110011100000000010	1000101100000000010
0000	0011	0010000100000000001	0100010100000000001	0110100100000000001	1000110100000000001
0000	0100	0010000100000000010	0100010100000000010	0110100100000000010	1000110100000000010
0000	0101	0010001100000000001	0100011100000000001	0110101100000000001	1000111100000000001
0000	0110	0010001100000000010	0100011100000000010	0110101100000000010	1000111100000000010
0000	0111	0010001100000000100	0100011100000000100	0110101100000000100	1000111100000000100
0000	1000	0000001100000000001	0010011100000000001	0100101100000000001	0110111100000000001
0000	1001	0000001100000000010	0010011100000000010	0100101100000000010	0110111100000000010
0000	1010	0000001100000000100	0010011100000000100	0100101100000000100	0110111100000000100
0000	1011	0000010100000000001	0010100100000000001	0100110100000000001	0111000100000000001
0000	1100	0000010100000000010	0010100100000000010	0100110100000000010	0111000100000000010
0000	1101	0000011100000000001	0010101100000000001	0100111100000000001	0111001100000000001
0000	1110	0000011100000000010	0010101100000000010	0100111100000000010	0111001100000000010
0000	1111	0000100100000000001	0010110100000000001	0101000100000000001	0111010100000000001
0001	0000	0000001010000000010	0010011010000000010	0100101010000000010	0110111010000000010
0001	0001	0000001010000000100	0010011010000000100	0100101010000000100	0110111010000000100
0001	0010	0000001010000001000	0010011010000001000	0100101010000001000	0110111010000001000
0001	0011	0000010010000000001	0010100010000000001	0100110010000000001	0111000010000000001
0001	0100	0000010010000000010	0010100010000000010	0100110010000000010	0111000010000000010
0001	0101	0000011010000000001	0010101010000000001	0100111010000000001	0111001010000000001
0001	0110	0000011010000000010	0010101010000000010	0100111010000000010	0111001010000000010
0001	0111	0000100010000000001	0010110010000000001	0101000010000000001	0111010010000000001

Table A.1: Output from O2_h Module

Reg	Loc	O2 Header MT11	O2 Header MT12	O2 Header MT21	O2 Header MT22
0001	1000	0000001010000000010	0010011010000000010	0100101010000000010	0110111010000000010
0001	1001	0000001010000000100	0010011010000000100	0100101010000000100	0110111010000000100
0001	1010	0000001010000001000	0010011010000001000	0100101010000001000	0110111010000001000
0001	1011	0000010010000000001	0010100010000000001	0100110010000000001	0111000010000000001
0001	1100	0000010010000000010	0010100010000000010	0100110010000000010	0111000010000000010
0001	1101	0000011010000000001	0010101010000000001	0100111010000000001	0111001010000000001
0001	1110	0000011010000000010	0010101010000000010	0100111010000000010	0111001010000000010
0001	1111	0000100010000000001	0010110010000000001	0101000010000000001	0111010010000000001
0010	0000	0001110001000000001	0100000001000000001	0110010001000000001	1000100001000000001
0010	0001	0001111001000000001	0100001001000000001	0110011001000000001	1000101001000000001
0010	0010	0001111001000000010	0100001001000000010	0110011001000000010	1000101001000000010
0010	0011	0010000001000000001	0100010001000000001	0110100001000000001	1000110001000000001
0010	0100	0010000001000000010	0100010001000000010	0110100001000000010	1000110001000000010
0010	0101	0010001001000000001	0100011001000000001	0110101001000000001	1000111001000000001
0010	0110	0010001001000000010	0100011001000000010	0110101001000000010	1000111001000000010
0010	0111	0010001001000000100	0100011001000000100	0110101001000000100	1000111001000000100
0010	1000	0010001001000001000	0100011001000001000	0110101001000001000	1000111001000001000
0010	1001	0000000001000000001	0010010001000000001	0100100001000000001	0110110001000000001
0010	1010	0000000001000000010	0010010001000000010	0100100001000000010	0110110001000000010
0010	1011	0000000001000000100	0010010001000000100	0100100001000000100	0110110001000000100
0010	1100	0000000001000001000	0010010001000001000	0100100001000001000	0110110001000001000
0010	1101	0000001001000000001	0010011001000000001	0100101001000000001	0110111001000000001
0011	0000	0001110000100000001	0100000000100000001	0110010000100000001	1000100000100000001
0011	0001	0001111000100000001	0100001000100000001	0110011000100000001	1000101000100000001

Table A.1: Output from O2_h Module

Reg	Loc	O2 Header MT11	O2 Header MT12	O2 Header MT21	O2 Header MT22
0011	0010	0001111000100000010	0100001000100000010	0110011000100000010	1000101000100000010
0011	0011	0010000000100000001	0100010000100000001	0110100000100000001	1000110000100000001
0011	0100	0010000000100000010	0100010000100000010	0110100000100000010	1000110000100000010
0011	0101	0010001000100000001	0100011000100000001	0110101000100000001	1000111000100000001
0011	0110	0010001000100000010	0100011000100000010	0110101000100000010	1000111000100000010
0011	0111	0010001000100000100	0100011000100000100	0110101000100000100	1000111000100000100
0011	1000	0010001000100001000	0100011000100001000	0110101000100001000	1000111000100001000
0011	1001	0000000000100000001	0010010000100000001	0100100000100000001	0110110000100000001
0011	1010	0000000000100000010	0010010000100000010	0100100000100000010	0110110000100000010
0011	1011	0000000000100000100	0010010000100000100	0100100000100000100	0110110000100000100
0011	1100	0000000000100001000	0010010000100001000	0100100000100001000	0110110000100001000
0011	1101	0000001000100000001	0010011000100000001	0100101000100000001	0110111000100000001
0100	0000	0001110000010000001	0100000000010000001	0110010000010000001	1000100000010000001
0100	0001	0001111000010000001	0100001000010000001	0110011000010000001	1000101000010000001
0100	0010	0001111000010000010	0100001000010000010	0110011000010000010	1000101000010000010
0100	0011	0010000000010000001	0100010000010000001	0110100000010000001	1000110000010000001
0100	0100	0010000000010000010	0100010000010000010	0110100000010000010	1000110000010000010
0100	0101	0010001000010000001	0100011000010000001	0110101000010000001	1000111000010000001
0100	0110	0010001000010000010	0100011000010000010	0110101000010000010	1000111000010000010
0100	0111	0000000000010000001	0010010000010000001	0100100000010000001	0110110000010000001
0100	1000	0000000000010000010	0010010000010000010	0100100000010000010	0110110000010000010
0100	1001	0000001000010000001	0010011000010000001	0100101000010000001	0110111000010000001
0100	1010	0000001000010000010	0010011000010000010	0100101000010000010	0110111000010000010
0100	1011	0000010000010000001	0010100000010000001	0100110000010000001	0111000000010000001

Table A.1: Output from O2_h Module

Reg	Loc	O2 Header MT11	O2 Header MT12	O2 Header MT21	O2 Header MT22
0100	1100	0000010000010000010	0010100000010000010	0100110000010000010	0111000000010000010
0100	1101	0000011000010000001	0010101000010000001	0100111000010000001	0111001000010000001
0100	1110	0000011000010000010	0010101000010000010	0100111000010000010	0111001000010000010
0100	1111	0000100000010000001	0010110000010000001	0101000000010000001	0111010000010000001
0101	0000	0001110000001000001	0100000000001000001	0110010000001000001	1000100000001000001
0101	0001	0001111000001000001	0100001000001000001	0110011000001000001	1000101000001000001
0101	0010	0001111000001000010	0100001000001000010	0110011000001000010	1000101000001000010
0101	0011	0010000000001000001	0100010000001000001	0110100000001000001	1000110000001000001
0101	0100	0010000000001000010	0100010000001000010	0110100000001000010	1000110000001000010
0101	0101	0010001000001000001	0100011000001000001	0110101000001000001	1000111000001000001
0101	0110	0010001000001000010	0100011000001000010	0110101000001000010	1000111000001000010
0101	0111	0000000000001000001	0010010000001000001	0100100000001000001	0110110000001000001
0101	1000	0000000000001000010	0010010000001000010	0100100000001000010	0110110000001000010
0101	1001	0000001000001000001	0010011000001000001	0100101000001000001	0110111000001000001
0101	1010	0000001000001000010	0010011000001000010	0100101000001000010	0110111000001000010
0101	1011	0000010000001000001	0010100000001000001	0100110000001000001	0111000000001000001
0101	1100	0000010000001000010	0010100000001000010	0100110000001000010	0111000000001000010
0101	1101	0000011000001000001	0010101000001000001	0100111000001000001	0111001000001000001
0101	1110	0000011000001000010	0010101000001000010	0100111000001000010	0111001000001000010
0101	1111	0000100000001000001	0010110000001000001	0101000000001000001	0111010000001000001
0110	0000	0001110000000100001	0100000000000100001	0110010000000100001	1000100000000100001
0110	0001	0001111000000100001	0100001000000100001	0110011000000100001	1000101000000100001
0110	0010	0001111000000100010	0100001000000100010	0110011000000100010	1000101000000100010
0110	0011	0010000000000100001	0100010000000100001	0110100000000100001	1000110000000100001

Table A.1: Output from O2_h Module

Reg	Loc	O2 Header MT11	O2 Header MT12	O2 Header MT21	O2 Header MT22
0110	0100	0010000000000100010	0100010000000100010	0110100000000100010	1000110000000100010
0110	0101	0010001000000100001	0100011000000100001	0110101000000100001	1000111000000100001
0110	0110	0010001000000100010	0100011000000100010	0110101000000100010	1000111000000100010
0110	0111	0000000000000100001	0010010000000100001	0100100000000100001	0110110000000100001
0110	1000	0000000000000100010	0010010000000100010	0100100000000100010	0110110000000100010
0110	1001	0000001000000100001	0010011000000100001	0100101000000100001	0110111000000100001
0110	1010	0000001000000100010	0010011000000100010	0100101000000100010	0110111000000100010
0110	1011	0000010000000100001	0010100000000100001	0100110000000100001	0111000000000100001
0110	1100	0000010000000100010	0010100000000100010	0100110000000100010	0111000000000100010
0110	1101	0000011000000100001	0010101000000100001	0100111000000100001	0111001000000100001
0110	1110	0000011000000100010	0010101000000100010	0100111000000100010	0111001000000100010
0110	1111	0000100000000100001	0010110000000100001	0101000000000100001	0111010000000100001
0111	0000	0001110000000010001	0100000000000010001	0110010000000010001	1000100000000010001
0111	0001	0001111000000010001	0100001000000010001	0110011000000010001	1000101000000010001
0111	0010	0010000000000010001	0100010000000010001	0110100000000010001	1000110000000010001
0111	0011	0010001000000010001	0100011000000010001	0110101000000010001	1000111000000010001
0111	0100	0000000000000010001	0010010000000010001	0100100000000010001	0110110000000010001
0111	0101	0000001000000010001	0010011000000010001	0100101000000010001	0110111000000010001
0111	0110	0000010000000010001	0010100000000010001	0100110000000010001	0111000000000010001
0111	0111	0000011000000010001	0010101000000010001	0100111000000010001	0111001000000010001
0111	1000	0000100000000010001	0010110000000010001	0101000000000010001	0111010000000010001
1000	0000	0001101100000000001	0011111100000000001	0110001100000000001	1000011100000000001
1000	0001	0001100100000000001	0011110100000000001	0110000100000000001	1000010100000000001
1000	0010	0001100100000000010	0011110100000000010	0110000100000000010	1000010100000000010

Table A.1: Output from O2_h Module

Reg	Loc	O2 Header MT11	O2 Header MT12	O2 Header MT21	O2 Header MT22
1000	0011	0001011100000000001	0011101100000000001	0101111100000000001	1000001100000000001
1000	0100	0001011100000000010	0011101100000000010	0101111100000000010	1000001100000000010
1000	0101	0001010100000000001	0011100100000000001	0101110100000000001	1000000100000000001
1000	0110	0001010100000000010	0011100100000000010	0101110100000000010	1000000100000000010
1000	0111	0001010100000000100	0011100100000000100	0101110100000000100	1000000100000000100
1000	1000	0001000100000000001	0011010100000000001	0101100100000000001	0111110100000000001
1000	1001	0001000100000000010	0011010100000000010	0101100100000000010	0111110100000000010
1000	1010	0001000100000000100	0011010100000000100	0101100100000000100	0111110100000000100
1000	1011	0000111100000000001	0011001100000000001	0101011100000000001	0111101100000000001
1000	1100	0000111100000000010	0011001100000000010	0101011100000000010	0111101100000000010
1000	1101	0000110100000000001	0011000100000000001	0101010100000000001	0111100100000000001
1000	1110	0000110100000000010	0011000100000000010	0101010100000000010	0111100100000000010
1000	1111	0000101100000000001	0010111100000000001	0101001100000000001	0111011100000000001
1001	0000	0001000010000000010	0011010010000000010	0101100010000000010	0111110010000000010
1001	0001	0001000010000000100	0011010010000000100	0101100010000000100	0111110010000000100
1001	0010	0001000010000001000	0011010010000001000	0101100010000001000	0111110010000001000
1001	0011	0000111010000000001	0011001010000000001	0101011010000000001	0111101010000000001
1001	0100	0000111010000000010	0011001010000000010	0101011010000000010	0111101010000000010
1001	0101	0000110010000000001	0011000010000000001	0101010010000000001	0111100010000000001
1001	0110	0000110010000000010	0011000010000000010	0101010010000000010	0111100010000000010
1001	0111	0000101010000000001	0010111010000000001	0101001010000000001	0111011010000000001
1001	1000	0001000010000000010	0011010010000000010	0101100010000000010	0111110010000000010
1001	1001	0001000010000000100	0011010010000000100	0101100010000000100	0111110010000000100
1001	1010	0001000010000001000	0011010010000001000	0101100010000001000	0111110010000001000

Table A.1: Output from O2_h Module

Reg	Loc	O2 Header MT11	O2 Header MT12	O2 Header MT21	O2 Header MT22
1001	1011	0000111010000000001	0011001010000000001	0101011010000000001	0111101010000000001
1001	1100	0000111010000000010	0011001010000000010	0101011010000000010	0111101010000000010
1001	1101	0000110010000000001	0011000010000000001	0101010010000000001	0111100010000000001
1001	1110	0000110010000000010	0011000010000000010	0101010010000000010	0111100010000000010
1001	1111	0000101010000000001	0010111010000000001	0101001010000000001	0111011010000000001
1010	0000	0001101001000000001	0011111001000000001	0110001001000000001	1000011001000000001
1010	0001	0001100001000000001	0011110001000000001	0110000001000000001	1000010001000000001
1010	0010	0001100001000000010	0011110001000000010	0110000001000000010	1000010001000000010
1010	0011	0001011001000000001	0011101001000000001	0101111001000000001	1000001001000000001
1010	0100	0001011001000000010	0011101001000000010	0101111001000000010	1000001001000000010
1010	0101	0001010001000000001	0011100001000000001	0101110001000000001	1000000001000000001
1010	0110	0001010001000000010	0011100001000000010	0101110001000000010	1000000001000000010
1010	0111	0001010001000000100	0011100001000000100	0101110001000000100	1000000001000000100
1010	1000	0001010001000001000	0011100001000001000	0101110001000001000	1000000001000001000
1010	1001	0001001001000000001	0011011001000000001	0101101001000000001	0111111001000000001
1010	1010	0001001001000000010	0011011001000000010	0101101001000000010	0111111001000000010
1010	1011	0001001001000000100	0011011001000000100	0101101001000000100	0111111001000000100
1010	1100	0001001001000001000	0011011001000001000	0101101001000001000	0111111001000001000
1010	1101	0001000001000000001	0011010001000000001	0101100001000000001	0111110001000000001
1011	0000	0001101000100000001	0011111000100000001	0110001000100000001	1000011000100000001
1011	0001	0001100000100000001	0011110000100000001	0110000000100000001	1000010000100000001
1011	0010	0001100000100000010	0011110000100000010	0110000000100000010	1000010000100000010
1011	0011	0001011000100000001	0011101000100000001	0101111000100000001	1000001000100000001
1011	0100	0001011000100000010	0011101000100000010	0101111000100000010	1000001000100000010

Table A.1: Output from O2_h Module

Reg	Loc	O2 Header MT11	O2 Header MT12	O2 Header MT21	O2 Header MT22
1011	0101	0001010000100000001	0011100000100000001	0101110000100000001	1000000000100000001
1011	0110	0001010000100000010	0011100000100000010	0101110000100000010	1000000000100000010
1011	0111	0001010000100000100	0011100000100000100	0101110000100000100	1000000000100000100
1011	1000	0001010000100001000	0011100000100001000	0101110000100001000	1000000000100001000
1011	1001	0001001000100000001	0011011000100000001	0101101000100000001	0111111000100000001
1011	1010	0001001000100000010	0011011000100000010	0101101000100000010	0111111000100000010
1011	1011	0001001000100000100	0011011000100000100	0101101000100000100	0111111000100000100
1011	1100	0001001000100001000	0011011000100001000	0101101000100001000	0111111000100001000
1011	1101	0001000000100000001	0011010000100000001	0101100000100000001	0111110000100000001
1100	0000	0001101000010000001	0011111000010000001	0110001000010000001	1000011000010000001
1100	0001	0001100000010000001	0011110000010000001	0110000000010000001	1000010000010000001
1100	0010	0001100000010000010	0011110000010000010	0110000000010000010	1000010000010000010
1100	0011	0001011000010000001	0011101000010000001	0101111000010000001	1000001000010000001
1100	0100	0001011000010000010	0011101000010000010	0101111000010000010	1000001000010000010
1100	0101	0001010000010000001	0011100000010000001	0101110000010000001	1000000000010000001
1100	0110	0001010000010000010	0011100000010000010	0101110000010000010	1000000000010000010
1100	0111	0001001000010000001	0011011000010000001	0101101000010000001	0111111000010000001
1100	1000	0001001000010000010	0011011000010000010	0101101000010000010	0111111000010000010
1100	1001	0001000000010000001	0011010000010000001	0101100000010000001	0111110000010000001
1100	1010	0001000000010000010	0011010000010000010	0101100000010000010	0111110000010000010
1100	1011	0000111000010000001	0011001000010000001	0101011000010000001	0111101000010000001
1100	1100	0000111000010000010	0011001000010000010	0101011000010000010	0111101000010000010
1100	1101	0000110000010000001	0011000000010000001	0101010000010000001	0111100000010000001
1100	1110	0000110000010000010	0011000000010000010	0101010000010000010	0111100000010000010

Table A.1: Output from O2_h Module

Reg	Loc	O2 Header MT11	O2 Header MT12	O2 Header MT21	O2 Header MT22
1100	1111	0000101000010000001	0010111000010000001	0101001000010000001	0111011000010000001
1101	0000	0001101000001000001	0011111000001000001	0110001000001000001	1000011000001000001
1101	0001	0001100000001000001	0011110000001000001	0110000000001000001	1000010000001000001
1101	0010	0001100000001000010	0011110000001000010	0110000000001000010	1000010000001000010
1101	0011	0001011000001000001	0011101000001000001	0101111000001000001	1000001000001000001
1101	0100	0001011000001000010	0011101000001000010	0101111000001000010	1000001000001000010
1101	0101	0001010000001000001	0011100000001000001	0101110000001000001	1000000000001000001
1101	0110	0001010000001000010	0011100000001000010	0101110000001000010	1000000000001000010
1101	0111	0001001000001000001	0011011000001000001	0101101000001000001	0111111000001000001
1101	1000	0001001000001000010	0011011000001000010	0101101000001000010	0111111000001000010
1101	1001	0001000000001000001	0011010000001000001	0101100000001000001	0111110000001000001
1101	1010	0001000000001000010	0011010000001000010	0101100000001000010	0111110000001000010
1101	1011	0000111000001000001	0011001000001000001	0101011000001000001	0111101000001000001
1101	1100	0000111000001000010	0011001000001000010	0101011000001000010	0111101000001000010
1101	1101	0000110000001000001	0011000000001000001	0101010000001000001	0111100000001000001
1101	1110	0000110000001000010	0011000000001000010	0101010000001000010	0111100000001000010
1101	1111	0000101000001000001	0010111000001000001	0101001000001000001	0111011000001000001
1110	0000	0001101000000100001	0011111000000100001	0110001000000100001	1000011000000100001
1110	0001	0001100000000100001	0011110000000100001	0110000000000100001	1000010000000100001
1110	0010	0001100000000100010	0011110000000100010	0110000000000100010	1000010000000100010
1110	0011	0001011000000100001	0011101000000100001	0101111000000100001	1000001000000100001
1110	0100	0001011000000100010	0011101000000100010	0101111000000100010	1000001000000100010
1110	0101	0001010000000100001	0011100000000100001	0101110000000100001	1000000000000100001
1110	0110	0001010000000100010	0011100000000100010	0101110000000100010	1000000000000100010

Table A.1: Output from O2_h Module

Reg	Loc	O2 Header MT11	O2 Header MT12	O2 Header MT21	O2 Header MT22
1110	0111	0001001000000100001	0011011000000100001	0101101000000100001	0111111000000100001
1110	1000	0001001000000100010	0011011000000100010	0101101000000100010	0111111000000100010
1110	1001	0001000000000100001	0011010000000100001	0101100000000100001	0111110000000100001
1110	1010	0001000000000100010	0011010000000100010	0101100000000100010	0111110000000100010
1110	1011	0000111000000100001	0011001000000100001	0101011000000100001	0111101000000100001
1110	1100	0000111000000100010	0011001000000100010	0101011000000100010	0111101000000100010
1110	1101	0000110000000100001	0011000000000100001	0101010000000100001	0111100000000100001
1110	1110	0000110000000100010	0011000000000100010	0101010000000100010	0111100000000100010
1110	1111	0000101000000100001	0010111000000100001	0101001000000100001	0111011000000100001
1111	0000	0001101000000010001	0011111000000010001	0110001000000010001	1000011000000010001
1111	0001	0001100000000010001	0011110000000010001	0110000000000010001	1000010000000010001
1111	0010	0001011000000010001	0011101000000010001	0101111000000010001	1000001000000010001
1111	0011	0001010000000010001	0011100000000010001	0101110000000010001	1000000000000010001
1111	0100	0001001000000010001	0011011000000010001	0101101000000010001	0111111000000010001
1111	0101	0001000000000010001	0011010000000010001	0101100000000010001	0111110000000010001
1111	0110	0000111000000010001	0011001000000010001	0101011000000010001	0111101000000010001
1111	0111	0000110000000010001	0011000000000010001	0101010000000010001	0111100000000010001
1111	1000	0000101000000010001	0010111000000010001	0101001000000010001	0111011000000010001

Table A.2: Output from GBT FIFO MT11

Time (ns)	valid	RO Data
6.158	U	0xUUUUUUUUUUUUUUUU
10.316	U	0xUUUUUUUUUUUUUUUU
14.474	U	0xUUUUUUUUUUUUUUUU
18.632	U	0xUUUUUUUUUUUUUUUU
22.79	U	0xUUUUUUUUUUUUUUUU
26.948	U	0xUUUUUUUUUUUUUUUU
31.106	U	0xUUUUUUUUUUUUUUUU
35.264	U	0xUUUUUUUUUUUUUUUU
39.422	U	0xUUUUUUUUUUUUUUUU
43.58	U	0xUUUUUUUUUUUUUUUU
47.738	1	0x0000000000000007
51.896	1	0x0000000000000006
56.054	1	0x0000000000000005
60.212	1	0x0000000000000004
64.37	1	0x0000000000000003
68.528	1	0x0000000000000002
72.686	1	0x0000000000000001
76.844	1	0x0000000000000000
81.002	0	0x00FFFFFFFFFFFFFF
85.16	0	0x00FFFFFFFFFFFFFF
89.318	0	0x00FFFFFFFFFFFFFF
93.476	0	0x00FFFFFFFFFFFFFF
97.634	0	0x00FFFFFFFFFFFFFF
101.792	0	0x00FFFFFFFFFFFFFF
105.95	0	0x00FFFFFFFFFFFFFF
110.108	0	0x00FFFFFFFFFFFFFF
114.266	0	0x00FFFFFFFFFFFFFF
118.424	0	0x00FFFFFFFFFFFFFF
122.582	0	0x00FFFFFFFFFFFFFF
126.74	0	0x00FFFFFFFFFFFFFF
130.898	0	0x00FFFFFFFFFFFFFF
135.056	0	0x00FFFFFFFFFFFFFF
139.214	0	0x00FFFFFFFFFFFFFF
143.372	0	0x00FFFFFFFFFFFFFF

Table A.2: Output from GBT FIFO MT11

Time (ns)	valid	RO Data
147.53	0	0x00FFFFFFFFFFFFF
151.688	0	0x00FFFFFFFFFFFFF
155.846	0	0x00FFFFFFFFFFFFF
160.004	0	0x00FFFFFFFFFFFFF
164.162	0	0x00FFFFFFFFFFFFF

Table A.3: Output from S2_RAM MT11

Time (ns)	busy	valid	RO Data
6.158	0	U	Undefined
10.316	0	U	Undefined
14.474	0	U	Undefined
18.632	0	U	Undefined
22.79	0	U	Undefined
26.948	0	U	Undefined
31.106	0	U	Undefined
35.264	0	U	Undefined
39.422	0	U	Undefined
43.58	0	U	Undefined
47.738	0	U	0x0000000000000000
51.896	0	U	0x0000000000000000
56.054	0	U	0x0000000000000000
60.212	0	1	0x00FF00FF00FF00FF
64.37	1	1	0x5555555555555555
68.528	1	1	0x00FF00FF00FF00FF
72.686	1	1	0x5555555555555555
76.844	1	1	0x00FF00FF00FF00FF
81.002	1	1	0x5555555555555555
85.16	1	1	0x00FF00FF00FF00FF
89.318	1	1	0x5555555555555555
93.476	1	1	0x00FF00FF00FF00FF
97.634	1	1	0x5555555555555555
101.792	1	1	0x00FF00FF00FF00FF
105.95	1	1	0x5555555555555555
110.108	1	1	0x00FF00FF00FF00FF

Table A.3: Output from S2_RAM MT11

Time (ns)	busy	valid	RO Data
114.266	1	1	0x5555555555555555
118.424	1	1	0x00FF00FF00FF00FF
122.582	1	1	0x5555555555555555
126.74	1	1	0x00FF00FF00FF00FF
130.898	1	1	0x5555555555555555
135.056	1	1	0x00FF00FF00FF00FF
139.214	1	1	0x5555555555555555
143.372	1	1	0x00FF00FF00FF00FF
147.53	1	1	0x5555555555555555
151.688	1	1	0x00FF00FF00FF00FF
155.846	1	1	0x5555555555555555
160.004	1	1	0x00FF00FF00FF00FF
164.162	1	1	0x5555555555555555
168.32	1	1	0x00FF00FF00FF00FF
172.478	1	1	0x5555555555555555
176.636	1	1	0x00FF00FF00FF00FF
180.794	1	1	0x5555555555555555
184.952	1	1	0x00FF00FF00FF00FF
189.11	1	1	0x5555555555555555
193.268	1	1	0x00FF00FF00FF00FF
197.426	1	1	0x5555555555555555
201.584	1	1	0x00FF00FF00FF00FF
205.742	1	1	0x5555555555555555
209.9	1	1	0x00FF00FF00FF00FF
214.058	1	1	0x5555555555555555
218.216	1	1	0x00FF00FF00FF00FF
222.374	1	1	0x5555555555555555
226.532	1	1	0x00FF00FF00FF00FF
230.69	1	1	0x5555555555555555
234.848	1	1	0x00FF00FF00FF00FF
239.006	1	1	0x5555555555555555
243.164	1	1	0x00FF00FF00FF00FF
247.322	1	1	0x5555555555555555
251.48	1	1	0x00FF00FF00FF00FF

Table A.3: Output from S2_RAM MT11

Time (ns)	busy	valid	RO Data
255.638	1	1	0x5555555555555555
259.796	1	1	0x00FF00FF00FF00FF
263.954	1	1	0x5555555555555555
268.112	1	1	0x00FF00FF00FF00FF
272.27	1	1	0x5555555555555555
276.428	1	1	0x00FF00FF00FF00FF
280.586	1	1	0x5555555555555555
284.744	1	1	0x00FF00FF00FF00FF
288.902	1	1	0x5555555555555555
293.06	1	1	0x00FF00FF00FF00FF
297.218	1	1	0x5555555555555555
301.376	1	1	0x00FF00FF00FF00FF
305.534	1	1	0x5555555555555555
309.692	1	1	0x00FF00FF00FF00FF
313.85	1	1	0x5555555555555555
318.008	1	1	0x00FF00FF00FF00FF
322.166	1	1	0x5555555555555555
326.324	1	1	0x00FF00FF00FF00FF
330.482	1	1	0x5555555555555555
334.64	1	1	0x00FF00FF00FF00FF
338.798	1	1	0x5555555555555555
342.956	1	1	0x00FF00FF00FF00FF
347.114	1	1	0x5555555555555555
351.272	1	1	0x00FF00FF00FF00FF
355.43	1	1	0x5555555555555555
359.588	1	1	0x00FF00FF00FF00FF
363.746	1	1	0x5555555555555555
367.904	1	1	0x00FF00FF00FF00FF
372.062	1	1	0x5555555555555555
376.22	1	1	0x00FF00FF00FF00FF
380.378	1	1	0x5555555555555555
384.536	1	1	0x00FF00FF00FF00FF
388.694	1	1	0x5555555555555555
392.852	1	1	0x00FF00FF00FF00FF

Table A.3: Output from S2_RAM MT11

Time (ns)	busy	valid	RO Data
397.01	1	1	0x5555555555555555
401.168	1	1	0x00FF00FF00FF00FF
405.326	1	1	0x5555555555555555
409.484	1	1	0x00FF00FF00FF00FF
413.642	1	1	0x5555555555555555
417.8	1	1	0x00FF00FF00FF00FF
421.958	1	1	0x5555555555555555
426.116	1	1	0x00FF00FF00FF00FF
430.274	1	1	0x5555555555555555
434.432	1	1	0x00FF00FF00FF00FF
438.59	1	1	0x5555555555555555
442.748	1	1	0x00FF00FF00FF00FF
446.906	1	1	0x5555555555555555
451.064	1	1	0x00FF00FF00FF00FF
455.222	1	1	0x5555555555555555
459.38	1	1	0x00FF00FF00FF00FF
463.538	1	1	0x5555555555555555
467.696	1	1	0x00FF00FF00FF00FF
471.854	1	1	0x5555555555555555
476.012	1	1	0x00FF00FF00FF00FF
480.17	1	1	0x5555555555555555
484.328	1	1	0x00FF00FF00FF00FF
488.486	1	1	0x5555555555555555
492.644	1	1	0x00FF00FF00FF00FF
496.802	1	1	0x5555555555555555
500.96	1	1	0x00FF00FF00FF00FF
505.118	1	1	0x5555555555555555
509.276	1	1	0x00FF00FF00FF00FF
513.434	1	1	0x5555555555555555
517.592	1	1	0x00FF00FF00FF00FF
521.75	1	1	0x5555555555555555
525.908	1	1	0x00FF00FF00FF00FF
530.066	1	1	0x5555555555555555
534.224	1	1	0x00FF00FF00FF00FF

Table A.3: Output from S2_RAM MT11

Time (ns)	busy	valid	RO Data
538.382	1	1	0x5555555555555555
542.54	1	1	0x00FF00FF00FF00FF
546.698	1	1	0x5555555555555555
550.856	1	1	0x00FF00FF00FF00FF
555.014	1	1	0x5555555555555555
559.172	1	1	0x00FF00FF00FF00FF
563.33	1	1	0x5555555555555555
567.488	1	1	0x00FF00FF00FF00FF
571.646	1	1	0x5555555555555555
575.804	1	1	0x00FF00FF00FF00FF
579.962	1	1	0x5555555555555555
584.12	1	1	0x00FF00FF00FF00FF
588.278	1	1	0x5555555555555555
592.436	1	0	0x5555555555555555
596.594	0	0	0x00FF00FF00FF00FF
600.752	0	0	0x00FF00FF00FF00FF

Table A.4: Output from S3_RAM

Clock Cycle	Time (ns)	busy	valid	RO Data
1	6.158	U	U	0xUUUUUUUUUUUUUUUU
2	10.316	U	U	0xUUUUUUUUUUUUUUUU
3	14.474	U	U	0xUUUUUUUUUUUUUUUU
4	18.632	U	U	0xUUUUUUUUUUUUUUUU
5	22.79	1	U	0xUUUUUUUUUUUUUUUU
6	26.948	1	U	0xUUUUUUUUUUUUUUUU
7	31.106	1	U	0xUUUUUUUUUUUUUUUU
8	35.264	1	U	0xUUUUUUUUUUUUUUUU
9	39.422	1	U	0xUUUUUUUUUUUUUUUU
10	43.58	1	U	0xUUUUUUUUUUUUUUUU
11	47.738	1	U	0xUUUUUUUUUUUUUUUU
12	51.896	1	U	0xUUUUUUUUUUUUUUUU
13	56.054	1	U	0xUUUUUUUUUUUUUUUU
14	60.212	1	U	0xUUUUUUUUUUUUUUUU
15	64.37	1	U	0xUUUUUUUUUUUUUUUU

Table A.4: Output from S3_RAM

Clock Cycle	Time (ns)	busy	valid	RO Data
16	68.528	1	U	0xUUUUUUUUUUUUUUUU
17	72.686	1	U	0xUUUUUUUUUUUUUUUU
18	76.844	1	U	0xUUUUUUUUUUUUUUUU
19	81.002	1	U	0xUUUUUUUUUUUUUUUU
20	85.16	1	U	0xUUUUUUUUUUUUUUUU
21	89.318	1	U	0xUUUUUUUUUUUUUUUU
22	93.476	1	U	0xUUUUUUUUUUUUUUUU
23	97.634	1	U	0xUUUUUUUUUUUUUUUU
24	101.792	1	U	0xUUUUUUUUUUUUUUUU
25	105.95	1	U	0xUUUUUUUUUUUUUUUU
26	110.108	1	U	0xUUUUUUUUUUUUUUUU
27	114.266	1	U	0xUUUUUUUUUUUUUUUU
28	118.424	1	U	0xUUUUUUUUUUUUUUUU
29	122.582	1	U	0xUUUUUUUUUUUUUUUU
30	126.74	1	U	0xUUUUUUUUUUUUUUUU
31	130.898	1	U	0xUUUUUUUUUUUUUUUU
32	135.056	1	U	0xUUUUUUUUUUUUUUUU
33	139.214	1	U	0xUUUUUUUUUUUUUUUU
34	143.372	1	U	0xUUUUUUUUUUUUUUUU
35	147.53	1	U	0xUUUUUUUUUUUUUUUU
36	151.688	1	U	0xUUUUUUUUUUUUUUUU
37	155.846	1	U	0xUUUUUUUUUUUUUUUU
38	160.004	1	U	0xUUUUUUUUUUUUUUUU
39	164.162	1	U	0xUUUUUUUUUUUUUUUU
40	168.32	1	U	0xUUUUUUUUUUUUUUUU
41	172.478	1	U	0xUUUUUUUUUUUUUUUU
42	176.636	1	U	0xUUUUUUUUUUUUUUUU
43	180.794	1	U	0xUUUUUUUUUUUUUUUU
44	184.952	1	U	0xUUUUUUUUUUUUUUUU
45	189.11	1	U	0xUUUUUUUUUUUUUUUU
46	193.268	1	U	0xUUUUUUUUUUUUUUUU
47	197.426	1	U	0xUUUUUUUUUUUUUUUU
48	201.584	1	U	0xUUUUUUUUUUUUUUUU
49	205.742	1	U	0xUUUUUUUUUUUUUUUU

Table A.4: Output from S3_RAM

Clock Cycle	Time (ns)	busy	valid	RO Data
50	209.9	1	U	0xUUUUUUUUUUUUUUUU
51	214.058	1	U	0xUUUUUUUUUUUUUUUU
52	218.216	1	U	0xUUUUUUUUUUUUUUUU
53	222.374	1	U	0xUUUUUUUUUUUUUUUU
54	226.532	1	U	0xUUUUUUUUUUUUUUUU
55	230.69	1	U	0xUUUUUUUUUUUUUUUU
56	234.848	1	U	0xUUUUUUUUUUUUUUUU
57	239.006	1	U	0xUUUUUUUUUUUUUUUU
58	243.164	1	U	0xUUUUUUUUUUUUUUUU
59	247.322	1	U	0xUUUUUUUUUUUUUUUU
60	251.48	1	U	0xUUUUUUUUUUUUUUUU
61	255.638	1	U	0xUUUUUUUUUUUUUUUU
62	259.796	1	U	0xUUUUUUUUUUUUUUUU
63	263.954	1	U	0xUUUUUUUUUUUUUUUU
64	268.112	1	U	0xUUUUUUUUUUUUUUUU
65	272.27	1	U	0xUUUUUUUUUUUUUUUU
66	276.428	1	U	0xUUUUUUUUUUUUUUUU
67	280.586	1	U	0xUUUUUUUUUUUUUUUU
68	284.744	1	U	0xUUUUUUUUUUUUUUUU
69	288.902	1	U	0xUUUUUUUUUUUUUUUU
70	293.06	1	U	0xUUUUUUUUUUUUUUUU
71	297.218	1	U	0xUUUUUUUUUUUUUUUU
72	301.376	1	U	0xUUUUUUUUUUUUUUUU
73	305.534	1	U	0xUUUUUUUUUUUUUUUU
74	309.692	1	U	0xUUUUUUUUUUUUUUUU
75	313.85	1	U	0xUUUUUUUUUUUUUUUU
76	318.008	1	U	0xUUUUUUUUUUUUUUUU
77	322.166	1	U	0xUUUUUUUUUUUUUUUU
78	326.324	1	U	0xUUUUUUUUUUUUUUUU
79	330.482	1	U	0xUUUUUUUUUUUUUUUU
80	334.64	1	U	0xUUUUUUUUUUUUUUUU
81	338.798	1	U	0xUUUUUUUUUUUUUUUU
82	342.956	1	U	0xUUUUUUUUUUUUUUUU
83	347.114	1	U	0xUUUUUUUUUUUUUUUU

Table A.4: Output from S3_RAM

Clock Cycle	Time (ns)	busy	valid	RO Data
84	351.272	1	U	0xUUUUUUUUUUUUUUUU
85	355.43	1	U	0xUUUUUUUUUUUUUUUU
86	359.588	1	U	0xUUUUUUUUUUUUUUUU
87	363.746	1	U	0xUUUUUUUUUUUUUUUU
88	367.904	1	U	0xUUUUUUUUUUUUUUUU
89	372.062	1	U	0xUUUUUUUUUUUUUUUU
90	376.22	1	U	0xUUUUUUUUUUUUUUUU
91	380.378	1	U	0xUUUUUUUUUUUUUUUU
92	384.536	1	U	0xUUUUUUUUUUUUUUUU
93	388.694	1	U	0xUUUUUUUUUUUUUUUU
94	392.852	1	U	0xUUUUUUUUUUUUUUUU
95	397.01	1	U	0xUUUUUUUUUUUUUUUU
96	401.168	1	U	0xUUUUUUUUUUUUUUUU
97	405.326	1	U	0xUUUUUUUUUUUUUUUU
98	409.484	1	U	0xUUUUUUUUUUUUUUUU
99	413.642	1	U	0xUUUUUUUUUUUUUUUU
100	417.8	1	U	0xUUUUUUUUUUUUUUUU
101	421.958	1	U	0xUUUUUUUUUUUUUUUU
102	426.116	1	U	0xUUUUUUUUUUUUUUUU
103	430.274	1	U	0xUUUUUUUUUUUUUUUU
104	434.432	1	U	0xUUUUUUUUUUUUUUUU
105	438.59	1	U	0xUUUUUUUUUUUUUUUU
106	442.748	1	U	0xUUUUUUUUUUUUUUUU
107	446.906	1	U	0xUUUUUUUUUUUUUUUU
108	451.064	1	U	0xUUUUUUUUUUUUUUUU
109	455.222	1	U	0xUUUUUUUUUUUUUUUU
110	459.38	1	U	0xUUUUUUUUUUUUUUUU
111	463.538	1	U	0xUUUUUUUUUUUUUUUU
112	467.696	1	U	0xUUUUUUUUUUUUUUUU
113	471.854	1	U	0xUUUUUUUUUUUUUUUU
114	476.012	1	U	0xUUUUUUUUUUUUUUUU
115	480.17	1	U	0xUUUUUUUUUUUUUUUU
116	484.328	1	U	0xUUUUUUUUUUUUUUUU
117	488.486	1	U	0xUUUUUUUUUUUUUUUU

Table A.4: Output from S3_RAM

Clock Cycle	Time (ns)	busy	valid	RO Data
118	492.644	1	U	0xUUUUUUUUUUUUUUUU
119	496.802	1	U	0xUUUUUUUUUUUUUUUU
120	500.96	1	U	0xUUUUUUUUUUUUUUUU
121	505.118	1	U	0xUUUUUUUUUUUUUUUU
122	509.276	1	U	0xUUUUUUUUUUUUUUUU
123	513.434	1	U	0xUUUUUUUUUUUUUUUU
124	517.592	1	U	0xUUUUUUUUUUUUUUUU
125	521.75	1	U	0xUUUUUUUUUUUUUUUU
126	525.908	1	U	0xUUUUUUUUUUUUUUUU
127	530.066	1	U	0xUUUUUUUUUUUUUUUU
128	534.224	1	U	0xUUUUUUUUUUUUUUUU
129	538.382	1	U	0xUUUUUUUUUUUUUUUU
130	542.54	1	U	0xUUUUUUUUUUUUUUUU
131	546.698	1	U	0xUUUUUUUUUUUUUUUU
132	550.856	1	U	0xUUUUUUUUUUUUUUUU
133	555.014	1	U	0xUUUUUUUUUUUUUUUU
134	559.172	1	U	0xUUUUUUUUUUUUUUUU
135	563.33	1	U	0xUUUUUUUUUUUUUUUU
136	567.488	1	U	0xUUUUUUUUUUUUUUUU
137	571.646	1	1	0xF0000000000000000
138	575.804	1	1	0xF0000000000000001
139	579.962	1	1	0xF0000000000000002
140	584.12	1	1	0xF0000000000000003
141	588.278	1	1	0xF0000000000000004
142	592.436	1	1	0xF0000000000000005
143	596.594	1	1	0xF0000000000000006
144	600.752	1	1	0xF0000000000000007
145	604.91	1	1	0xF0000000000000008
146	609.068	1	1	0xF0000000000000009
147	613.226	1	1	0xF000000000000000A
148	617.384	1	1	0xF000000000000000B
149	621.542	1	1	0xF000000000000000C
150	625.7	1	1	0xF000000000000000D
151	629.858	1	1	0xF000000000000000E

Table A.4: Output from S3_RAM

Clock Cycle	Time (ns)	busy	valid	RO Data
152	634.016	1	1	0xF000000000000000F
153	638.174	1	1	0xF0000000000000010
154	642.332	1	1	0xF0000000000000011
155	646.49	1	1	0xF0000000000000012
156	650.648	1	1	0xF0000000000000013
157	654.806	1	1	0xF0000000000000014
158	658.964	1	1	0xF0000000000000015
159	663.122	1	1	0xF0000000000000016
160	667.28	1	1	0xF0000000000000017
161	671.438	1	1	0xF0000000000000018
162	675.596	1	1	0xF0000000000000019
163	679.754	1	1	0xF000000000000001A
164	683.912	1	1	0xF000000000000001B
165	688.07	1	1	0xF000000000000001C
166	692.228	1	1	0xF000000000000001D
167	696.386	1	1	0xF000000000000001E
168	700.544	1	1	0xF000000000000001F
169	704.702	1	1	0xF0000000000000020
170	708.86	1	1	0xF0000000000000021
171	713.018	1	1	0xF0000000000000022
172	717.176	1	1	0xF0000000000000023
173	721.334	1	1	0xF0000000000000024
174	725.492	1	1	0xF0000000000000025
175	729.65	1	1	0xF0000000000000026
176	733.808	1	1	0xF0000000000000027
177	737.966	1	1	0xF0000000000000028
178	742.124	1	1	0xF0000000000000029
179	746.282	1	1	0xF000000000000002A
180	750.44	1	1	0xF000000000000002B
181	754.598	1	1	0xF000000000000002C
182	758.756	1	1	0xF000000000000002D
183	762.914	1	1	0xF000000000000002E
184	767.072	1	1	0xF000000000000002F
185	771.23	1	1	0xF0000000000000030

Table A.4: Output from S3_RAM

Clock Cycle	Time (ns)	busy	valid	RO Data
186	775.388	1	1	0xF0000000000000031
187	779.546	1	1	0xF0000000000000032
188	783.704	1	1	0xF0000000000000033
189	787.862	1	1	0xF0000000000000034
190	792.02	1	1	0xF0000000000000035
191	796.178	1	1	0xF0000000000000036
192	800.336	1	1	0xF0000000000000037
193	804.494	1	1	0xF0000000000000038
194	808.652	1	1	0xF0000000000000039
195	812.81	1	1	0xF000000000000003A
196	816.968	1	1	0xF000000000000003B
197	821.126	1	1	0xF000000000000003C
198	825.284	1	1	0xF000000000000003D
199	829.442	1	1	0xF000000000000003E
200	833.6	1	1	0xF000000000000003F
201	837.758	1	1	0xF0000000000000040
202	841.916	1	1	0xF0000000000000041
203	846.074	1	1	0xF0000000000000042
204	850.232	1	1	0xF0000000000000043
205	854.39	1	1	0xF0000000000000044
206	858.548	1	1	0xF0000000000000045
207	862.706	1	1	0xF0000000000000046
208	866.864	1	1	0xF0000000000000047
209	871.022	1	1	0xF0000000000000048
210	875.18	1	1	0xF0000000000000049
211	879.338	1	1	0xF000000000000004A
212	883.496	1	1	0xF000000000000004B
213	887.654	1	1	0xF000000000000004C
214	891.812	1	1	0xF000000000000004D
215	895.97	1	1	0xF000000000000004E
216	900.128	1	1	0xF000000000000004F
217	904.286	1	1	0xF0000000000000050
218	908.444	1	1	0xF0000000000000051
219	912.602	1	1	0xF0000000000000052

Table A.4: Output from S3_RAM

Clock Cycle	Time (ns)	busy	valid	RO Data
220	916.76	1	1	0xF0000000000000053
221	920.918	1	1	0xF0000000000000054
222	925.076	1	1	0xF0000000000000055
223	929.234	1	1	0xF0000000000000056
224	933.392	1	1	0xF0000000000000057
225	937.55	1	1	0xF0000000000000058
226	941.708	1	1	0xF0000000000000059
227	945.866	1	1	0xF000000000000005A
228	950.024	1	1	0xF000000000000005B
229	954.182	1	1	0xF000000000000005C
230	958.34	1	1	0xF000000000000005D
231	962.498	1	1	0xF000000000000005E
232	966.656	1	1	0xF000000000000005F
233	970.814	1	1	0xF0000000000000060
234	974.972	1	1	0xF0000000000000061
235	979.13	1	1	0xF0000000000000062
236	983.288	1	1	0xF0000000000000063
237	987.446	1	1	0xF0000000000000064
238	991.604	1	1	0xF0000000000000065
239	995.762	1	1	0xF0000000000000066
240	999.92	1	1	0xF0000000000000067
241	1004.078	1	1	0xF0000000000000068
242	1008.236	1	1	0xF0000000000000069
243	1012.394	1	1	0xF000000000000006A
244	1016.552	1	1	0xF000000000000006B
245	1020.71	1	1	0xF000000000000006C
246	1024.868	1	1	0xF000000000000006D
247	1029.026	1	1	0xF000000000000006E
248	1033.184	1	1	0xF000000000000006F
249	1037.342	1	1	0xF0000000000000070
250	1041.5	1	1	0xF0000000000000071
251	1045.658	1	1	0xF0000000000000072
252	1049.816	1	1	0xF0000000000000073
253	1053.974	1	1	0xF0000000000000074

Table A.4: Output from S3_RAM

Clock Cycle	Time (ns)	busy	valid	RO Data
254	1058.132	1	1	0xF0000000000000075
255	1062.29	1	1	0xF0000000000000076
256	1066.448	1	1	0xF0000000000000077
257	1070.606	1	1	0xF0000000000000078
258	1074.764	1	1	0xF0000000000000079
259	1078.922	1	1	0xF000000000000007A
260	1083.08	1	1	0xF000000000000007B
261	1087.238	1	1	0xF000000000000007C
262	1091.396	1	1	0xF000000000000007D
263	1095.554	1	1	0xF000000000000007E
264	1099.712	1	1	0xF000000000000007F
265	1103.87	1	1	0xE0000000000000000
266	1108.028	1	1	0xE0000000000000001
267	1112.186	1	1	0xE0000000000000002
268	1116.344	1	1	0xE0000000000000003
269	1120.502	1	1	0xE0000000000000004
270	1124.66	1	1	0xE0000000000000005
271	1128.818	1	1	0xE0000000000000006
272	1132.976	1	1	0xE0000000000000007
273	1137.134	1	1	0xE0000000000000008
274	1141.292	1	1	0xE0000000000000009
275	1145.45	1	1	0xE000000000000000A
276	1149.608	1	1	0xE000000000000000B
277	1153.766	1	1	0xE000000000000000C
278	1157.924	1	1	0xE000000000000000D
279	1162.082	1	1	0xE000000000000000E
280	1166.24	1	1	0xE000000000000000F
281	1170.398	1	1	0xE0000000000000010
282	1174.556	1	1	0xE0000000000000011
283	1178.714	1	1	0xE0000000000000012
284	1182.872	1	1	0xE0000000000000013
285	1187.03	1	1	0xE0000000000000014
286	1191.188	1	1	0xE0000000000000015
287	1195.346	1	1	0xE0000000000000016

Table A.4: Output from S3_RAM

Clock Cycle	Time (ns)	busy	valid	RO Data
288	1199.504	1	1	0xE0000000000000017
289	1203.662	1	1	0xE0000000000000018
290	1207.82	1	1	0xE0000000000000019
291	1211.978	1	1	0xE000000000000001A
292	1216.136	1	1	0xE000000000000001B
293	1220.294	1	1	0xE000000000000001C
294	1224.452	1	1	0xE000000000000001D
295	1228.61	1	1	0xE000000000000001E
296	1232.768	1	1	0xE000000000000001F
297	1236.926	1	1	0xE0000000000000020
298	1241.084	1	1	0xE0000000000000021
299	1245.242	1	1	0xE0000000000000022
300	1249.4	1	1	0xE0000000000000023
301	1253.558	1	1	0xE0000000000000024
302	1257.716	1	1	0xE0000000000000025
303	1261.874	1	1	0xE0000000000000026
304	1266.032	1	1	0xE0000000000000027
305	1270.19	1	1	0xE0000000000000028
306	1274.348	1	1	0xE0000000000000029
307	1278.506	1	1	0xE000000000000002A
308	1282.664	1	1	0xE000000000000002B
309	1286.822	1	1	0xE000000000000002C
310	1290.98	1	1	0xE000000000000002D
311	1295.138	1	1	0xE000000000000002E
312	1299.296	1	1	0xE000000000000002F
313	1303.454	1	1	0xE0000000000000030
314	1307.612	1	1	0xE0000000000000031
315	1311.77	1	1	0xE0000000000000032
316	1315.928	1	1	0xE0000000000000033
317	1320.086	1	1	0xE0000000000000034
318	1324.244	1	1	0xE0000000000000035
319	1328.402	1	1	0xE0000000000000036
320	1332.56	1	1	0xE0000000000000037
321	1336.718	1	1	0xE0000000000000038

Table A.4: Output from S3_RAM

Clock Cycle	Time (ns)	busy	valid	RO Data
322	1340.876	1	1	0xE0000000000000039
323	1345.034	1	1	0xE000000000000003A
324	1349.192	1	1	0xE000000000000003B
325	1353.35	1	1	0xE000000000000003C
326	1357.508	1	1	0xE000000000000003D
327	1361.666	1	1	0xE000000000000003E
328	1365.824	1	1	0xE000000000000003F
329	1369.982	1	1	0xE0000000000000040
330	1374.14	1	1	0xE0000000000000041
331	1378.298	1	1	0xE0000000000000042
332	1382.456	1	1	0xE0000000000000043
333	1386.614	1	1	0xE0000000000000044
334	1390.772	1	1	0xE0000000000000045
335	1394.93	1	1	0xE0000000000000046
336	1399.088	1	1	0xE0000000000000047
337	1403.246	1	1	0xE0000000000000048
338	1407.404	1	1	0xE0000000000000049
339	1411.562	1	1	0xE000000000000004A
340	1415.72	1	1	0xE000000000000004B
341	1419.878	1	1	0xE000000000000004C
342	1424.036	1	1	0xE000000000000004D
343	1428.194	1	1	0xE000000000000004E
344	1432.352	1	1	0xE000000000000004F
345	1436.51	1	1	0xE0000000000000050
346	1440.668	1	1	0xE0000000000000051
347	1444.826	1	1	0xE0000000000000052
348	1448.984	1	1	0xE0000000000000053
349	1453.142	1	1	0xE0000000000000054
350	1457.3	1	1	0xE0000000000000055
351	1461.458	1	1	0xE0000000000000056
352	1465.616	1	1	0xE0000000000000057
353	1469.774	1	1	0xE0000000000000058
354	1473.932	1	1	0xE0000000000000059
355	1478.09	1	1	0xE000000000000005A

Table A.4: Output from S3_RAM

Clock Cycle	Time (ns)	busy	valid	RO Data
356	1482.248	1	1	0xE000000000000005B
357	1486.406	1	1	0xE000000000000005C
358	1490.564	1	1	0xE000000000000005D
359	1494.722	1	1	0xE000000000000005E
360	1498.88	1	1	0xE000000000000005F
361	1503.038	1	1	0xE0000000000000060
362	1507.196	1	1	0xE0000000000000061
363	1511.354	1	1	0xE0000000000000062
364	1515.512	1	1	0xE0000000000000063
365	1519.67	1	1	0xE0000000000000064
366	1523.828	1	1	0xE0000000000000065
367	1527.986	1	1	0xE0000000000000066
368	1532.144	1	1	0xE0000000000000067
369	1536.302	1	1	0xE0000000000000068
370	1540.46	1	1	0xE0000000000000069
371	1544.618	1	1	0xE000000000000006A
372	1548.776	1	1	0xE000000000000006B
373	1552.934	1	1	0xE000000000000006C
374	1557.092	1	1	0xE000000000000006D
375	1561.25	1	1	0xE000000000000006E
376	1565.408	1	1	0xE000000000000006F
377	1569.566	1	1	0xE0000000000000070
378	1573.724	1	1	0xE0000000000000071
379	1577.882	1	1	0xE0000000000000072
380	1582.04	1	1	0xE0000000000000073
381	1586.198	1	1	0xE0000000000000074
382	1590.356	1	1	0xE0000000000000075
383	1594.514	1	1	0xE0000000000000076
384	1598.672	1	1	0xE0000000000000077
385	1602.83	1	1	0xE0000000000000078
386	1606.988	1	1	0xE0000000000000079
387	1611.146	1	1	0xE000000000000007A
388	1615.304	1	1	0xE000000000000007B
389	1619.462	1	1	0xE000000000000007C

Table A.4: Output from S3_RAM

Clock Cycle	Time (ns)	busy	valid	RO Data
390	1623.62	1	1	0xE000000000000007D
391	1627.778	1	1	0xE000000000000007E
392	1631.936	1	1	0xE000000000000007F
393	1636.094	1	1	0x30000000000000000
394	1640.252	1	1	0x30000000000000001
395	1644.41	1	1	0x30000000000000002
396	1648.568	1	1	0x30000000000000003
397	1652.726	1	1	0x30000000000000004
398	1656.884	1	1	0x30000000000000005
399	1661.042	1	1	0x30000000000000006
400	1665.2	1	1	0x30000000000000007
401	1669.358	1	1	0x30000000000000008
402	1673.516	1	1	0x30000000000000009
403	1677.674	1	1	0x3000000000000000A
404	1681.832	1	1	0x3000000000000000B
405	1685.99	1	1	0x3000000000000000C
406	1690.148	1	1	0x3000000000000000D
407	1694.306	1	1	0x3000000000000000E
408	1698.464	1	1	0x3000000000000000F
409	1702.622	1	1	0x30000000000000010
410	1706.78	1	1	0x30000000000000011
411	1710.938	1	1	0x30000000000000012
412	1715.096	1	1	0x30000000000000013
413	1719.254	1	1	0x30000000000000014
414	1723.412	1	1	0x30000000000000015
415	1727.57	1	1	0x30000000000000016
416	1731.728	1	1	0x30000000000000017
417	1735.886	1	1	0x30000000000000018
418	1740.044	1	1	0x30000000000000019
419	1744.202	1	1	0x3000000000000001A
420	1748.36	1	1	0x3000000000000001B
421	1752.518	1	1	0x3000000000000001C
422	1756.676	1	1	0x3000000000000001D
423	1760.834	1	1	0x3000000000000001E

Table A.4: Output from S3_RAM

Clock Cycle	Time (ns)	busy	valid	RO Data
424	1764.992	1	1	0x3000000000000001F
425	1769.15	1	1	0x30000000000000020
426	1773.308	1	1	0x30000000000000021
427	1777.466	1	1	0x30000000000000022
428	1781.624	1	1	0x30000000000000023
429	1785.782	1	1	0x30000000000000024
430	1789.94	1	1	0x30000000000000025
431	1794.098	1	1	0x30000000000000026
432	1798.256	1	1	0x30000000000000027
433	1802.414	1	1	0x30000000000000028
434	1806.572	1	1	0x30000000000000029
435	1810.73	1	1	0x3000000000000002A
436	1814.888	1	1	0x3000000000000002B
437	1819.046	1	1	0x3000000000000002C
438	1823.204	1	1	0x3000000000000002D
439	1827.362	1	1	0x3000000000000002E
440	1831.52	1	1	0x3000000000000002F
441	1835.678	1	1	0x30000000000000030
442	1839.836	1	1	0x30000000000000031
443	1843.994	1	1	0x30000000000000032
444	1848.152	1	1	0x30000000000000033
445	1852.31	1	1	0x30000000000000034
446	1856.468	1	1	0x30000000000000035
447	1860.626	1	1	0x30000000000000036
448	1864.784	1	1	0x30000000000000037
449	1868.942	1	1	0x30000000000000038
450	1873.1	1	1	0x30000000000000039
451	1877.258	1	1	0x3000000000000003A
452	1881.416	1	1	0x3000000000000003B
453	1885.574	1	1	0x3000000000000003C
454	1889.732	1	1	0x3000000000000003D
455	1893.89	1	1	0x3000000000000003E
456	1898.048	1	1	0x3000000000000003F
457	1902.206	1	1	0x30000000000000040

Table A.4: Output from S3_RAM

Clock Cycle	Time (ns)	busy	valid	RO Data
458	1906.364	1	1	0x30000000000000041
459	1910.522	1	1	0x30000000000000042
460	1914.68	1	1	0x30000000000000043
461	1918.838	1	1	0x30000000000000044
462	1922.996	1	1	0x30000000000000045
463	1927.154	1	1	0x30000000000000046
464	1931.312	1	1	0x30000000000000047
465	1935.47	1	1	0x30000000000000048
466	1939.628	1	1	0x30000000000000049
467	1943.786	1	1	0x3000000000000004A
468	1947.944	1	1	0x3000000000000004B
469	1952.102	1	1	0x3000000000000004C
470	1956.26	1	1	0x3000000000000004D
471	1960.418	1	1	0x3000000000000004E
472	1964.576	1	1	0x3000000000000004F
473	1968.734	1	1	0x30000000000000050
474	1972.892	1	1	0x30000000000000051
475	1977.05	1	1	0x30000000000000052
476	1981.208	1	1	0x30000000000000053
477	1985.366	1	1	0x30000000000000054
478	1989.524	1	1	0x30000000000000055
479	1993.682	1	1	0x30000000000000056
480	1997.84	1	1	0x30000000000000057
481	2001.998	1	1	0x30000000000000058
482	2006.156	1	1	0x30000000000000059
483	2010.314	1	1	0x3000000000000005A
484	2014.472	1	1	0x3000000000000005B
485	2018.63	1	1	0x3000000000000005C
486	2022.788	1	1	0x3000000000000005D
487	2026.946	1	1	0x3000000000000005E
488	2031.104	1	1	0x3000000000000005F
489	2035.262	1	1	0x30000000000000060
490	2039.42	1	1	0x30000000000000061
491	2043.578	1	1	0x30000000000000062

Table A.4: Output from S3_RAM

Clock Cycle	Time (ns)	busy	valid	RO Data
492	2047.736	1	1	0x30000000000000063
493	2051.894	1	1	0x30000000000000064
494	2056.052	1	1	0x30000000000000065
495	2060.21	1	1	0x30000000000000066
496	2064.368	1	1	0x30000000000000067
497	2068.526	1	1	0x30000000000000068
498	2072.684	1	1	0x30000000000000069
499	2076.842	1	1	0x3000000000000006A
500	2081	1	1	0x3000000000000006B
501	2085.158	1	1	0x3000000000000006C
502	2089.316	1	1	0x3000000000000006D
503	2093.474	1	1	0x3000000000000006E
504	2097.632	1	1	0x3000000000000006F
505	2101.79	1	1	0x30000000000000070
506	2105.948	1	1	0x30000000000000071
507	2110.106	1	1	0x30000000000000072
508	2114.264	1	1	0x30000000000000073
509	2118.422	1	1	0x30000000000000074
510	2122.58	1	1	0x30000000000000075
511	2126.738	1	1	0x30000000000000076
512	2130.896	1	1	0x30000000000000077
513	2135.054	1	1	0x30000000000000078
514	2139.212	1	1	0x30000000000000079
515	2143.37	1	1	0x3000000000000007A
516	2147.528	1	1	0x3000000000000007B
517	2151.686	1	1	0x3000000000000007C
518	2155.844	1	1	0x3000000000000007D
519	2160.002	1	1	0x3000000000000007E
520	2164.16	1	1	0x3000000000000007F
521	2168.318	1	1	0x10000000000000000
522	2172.476	1	1	0x10000000000000001
523	2176.634	1	1	0x10000000000000002
524	2180.792	1	1	0x10000000000000003
525	2184.95	1	1	0x10000000000000004

Table A.4: Output from S3_RAM

Clock Cycle	Time (ns)	busy	valid	RO Data
526	2189.108	1	1	0x10000000000000005
527	2193.266	1	1	0x10000000000000006
528	2197.424	1	1	0x10000000000000007
529	2201.582	1	1	0x10000000000000008
530	2205.74	1	1	0x10000000000000009
531	2209.898	1	1	0x1000000000000000A
532	2214.056	1	1	0x1000000000000000B
533	2218.214	1	1	0x1000000000000000C
534	2222.372	1	1	0x1000000000000000D
535	2226.53	1	1	0x1000000000000000E
536	2230.688	1	1	0x1000000000000000F
537	2234.846	1	1	0x10000000000000010
538	2239.004	1	1	0x10000000000000011
539	2243.162	1	1	0x10000000000000012
540	2247.32	1	1	0x10000000000000013
541	2251.478	1	1	0x10000000000000014
542	2255.636	1	1	0x10000000000000015
543	2259.794	1	1	0x10000000000000016
544	2263.952	1	1	0x10000000000000017
545	2268.11	1	1	0x10000000000000018
546	2272.268	1	1	0x10000000000000019
547	2276.426	1	1	0x1000000000000001A
548	2280.584	1	1	0x1000000000000001B
549	2284.742	1	1	0x1000000000000001C
550	2288.9	1	1	0x1000000000000001D
551	2293.058	1	1	0x1000000000000001E
552	2297.216	1	1	0x1000000000000001F
553	2301.374	1	1	0x10000000000000020
554	2305.532	1	1	0x10000000000000021
555	2309.69	1	1	0x10000000000000022
556	2313.848	1	1	0x10000000000000023
557	2318.006	1	1	0x10000000000000024
558	2322.164	1	1	0x10000000000000025
559	2326.322	1	1	0x10000000000000026

Table A.4: Output from S3_RAM

Clock Cycle	Time (ns)	busy	valid	RO Data
560	2330.48	1	1	0x10000000000000027
561	2334.638	1	1	0x10000000000000028
562	2338.796	1	1	0x10000000000000029
563	2342.954	1	1	0x1000000000000002A
564	2347.112	1	1	0x1000000000000002B
565	2351.27	1	1	0x1000000000000002C
566	2355.428	1	1	0x1000000000000002D
567	2359.586	1	1	0x1000000000000002E
568	2363.744	1	1	0x1000000000000002F
569	2367.902	1	1	0x10000000000000030
570	2372.06	1	1	0x10000000000000031
571	2376.218	1	1	0x10000000000000032
572	2380.376	1	1	0x10000000000000033
573	2384.534	1	1	0x10000000000000034
574	2388.692	1	1	0x10000000000000035
575	2392.85	1	1	0x10000000000000036
576	2397.008	1	1	0x10000000000000037
577	2401.166	1	1	0x10000000000000038
578	2405.324	1	1	0x10000000000000039
579	2409.482	1	1	0x1000000000000003A
580	2413.64	1	1	0x1000000000000003B
581	2417.798	1	1	0x1000000000000003C
582	2421.956	1	1	0x1000000000000003D
583	2426.114	1	1	0x1000000000000003E
584	2430.272	1	1	0x1000000000000003F
585	2434.43	1	1	0x10000000000000040
586	2438.588	1	1	0x10000000000000041
587	2442.746	1	1	0x10000000000000042
588	2446.904	1	1	0x10000000000000043
589	2451.062	1	1	0x10000000000000044
590	2455.22	1	1	0x10000000000000045
591	2459.378	1	1	0x10000000000000046
592	2463.536	1	1	0x10000000000000047
593	2467.694	1	1	0x10000000000000048

Table A.4: Output from S3_RAM

Clock Cycle	Time (ns)	busy	valid	RO Data
594	2471.852	1	1	0x10000000000000049
595	2476.01	1	1	0x1000000000000004A
596	2480.168	1	1	0x1000000000000004B
597	2484.326	1	1	0x1000000000000004C
598	2488.484	1	1	0x1000000000000004D
599	2492.642	1	1	0x1000000000000004E
600	2496.8	1	1	0x1000000000000004F
601	2500.958	1	1	0x10000000000000050
602	2505.116	1	1	0x10000000000000051
603	2509.274	1	1	0x10000000000000052
604	2513.432	1	1	0x10000000000000053
605	2517.59	1	1	0x10000000000000054
606	2521.748	1	1	0x10000000000000055
607	2525.906	1	1	0x10000000000000056
608	2530.064	1	1	0x10000000000000057
609	2534.222	1	1	0x10000000000000058
610	2538.38	1	1	0x10000000000000059
611	2542.538	1	1	0x1000000000000005A
612	2546.696	1	1	0x1000000000000005B
613	2550.854	1	1	0x1000000000000005C
614	2555.012	1	1	0x1000000000000005D
615	2559.17	1	1	0x1000000000000005E
616	2563.328	1	1	0x1000000000000005F
617	2567.486	1	1	0x10000000000000060
618	2571.644	1	1	0x10000000000000061
619	2575.802	1	1	0x10000000000000062
620	2579.96	1	1	0x10000000000000063
621	2584.118	1	1	0x10000000000000064
622	2588.276	1	1	0x10000000000000065
623	2592.434	1	1	0x10000000000000066
624	2596.592	1	1	0x10000000000000067
625	2600.75	1	1	0x10000000000000068
626	2604.908	1	1	0x10000000000000069
627	2609.066	1	1	0x1000000000000006A

Table A.4: Output from S3_RAM

Clock Cycle	Time (ns)	busy	valid	RO Data
628	2613.224	1	1	0x1000000000000006B
629	2617.382	1	1	0x1000000000000006C
630	2621.54	1	1	0x1000000000000006D
631	2625.698	1	1	0x1000000000000006E
632	2629.856	1	1	0x1000000000000006F
633	2634.014	1	1	0x10000000000000070
634	2638.172	1	1	0x10000000000000071
635	2642.33	1	1	0x10000000000000072
636	2646.488	1	1	0x10000000000000073
637	2650.646	1	1	0x10000000000000074
638	2654.804	1	1	0x10000000000000075
639	2658.962	1	1	0x10000000000000076
640	2663.12	1	1	0x10000000000000077
641	2667.278	1	1	0x10000000000000078
642	2671.436	1	1	0x10000000000000079
643	2675.594	1	1	0x1000000000000007A
644	2679.752	1	1	0x1000000000000007B
645	2683.91	1	1	0x1000000000000007C
646	2688.068	1	1	0x1000000000000007D
647	2692.226	1	1	0x1000000000000007E
648	2696.384	1	1	0x1000000000000007F
649	2700.542	1	0	0xXXXXXXXXXXXXXXXX

Appendix B

Software Source Code

The source code for this project is publicly available on GitHub.
https://github.com/hermes-lab/MID-CRU

Appendix C

EBE Faculty: Assessment of Ethics in Research Projects

ETHICS APPLICATION FORM

Please Note:
Any person planning to undertake research in the Faculty of Engineering and the Built Environment (EBE) at the University of Cape Town is required to complete this form **before** collecting or analysing data. The objective of submitting this application *prior* to embarking on research is to ensure that the highest ethical standards in research, conducted under the auspices of the EBE Faculty, are met. Please ensure that you have read, and understood the **EBE Ethics in Research Handbook** (available from the UCT EBE, Research Ethics website) prior to completing this application form: http://www.ebe.uct.ac.za/ebe/research/ethics1

APPLICANT'S DETAILS		
Name of principal researcher, student or external applicant		Nathan Boyles
Department		Electrical Engineering
Preferred email address of applicant:		nathanh.boyles@gmail.com
If Student	Your Degree: e.g., MSc, PhD, etc.	MSc
	Credit Value of Research: e.g., 60/120/180/360 etc.	180
	Name of Supervisor (if supervised):	Associate Professor Amit Mishra
If this is a researchcontract, indicate the source of funding/sponsorship		iThemba LABS/ SA-CERN
Project Title		User Logic Development for The Muon Identifier's Common Readout Unit for the ALICE Experiment at The Large Hadron Collider

I hereby undertake to carry out my research in such a way that:

- there is no apparent legal objection to the nature or the method of research; and
- the research will not compromise staff or students or the other responsibilities of the University;
- the stated objective will be achieved, and the findings will have a high degree of validity;
- limitations and alternative interpretations will be considered;
- the findings could be subject to peer review and publicly available; and
- I will comply with the conventions of copyright and avoid any practice that would constitute plagiarism.

APPLICATION BY	Full name	Signature	Date
Principal Researcher/ Student/External applicant	Nathan Boyles		06/01/2020
SUPPORTED BY	Full name	Signature	Date
Supervisor (where applicable)	Amit Mishra		4-1-2020

APPROVED BY	Full name	Signature	Date
HOD (or delegated nominee) Final authority for all applicants who have answered NO to all questions in Section 1; and for all Undergraduate research (Including Honours).			
Chair: Faculty EIR Committee For applicants other than undergraduate students who have answered YES to any of the questions in Section 1.			

Bibliography

[1] The ALICE Collaboration, "Real-time data processing in the alice high level trigger at the lhc," 12 2018.

[2] E. David, T. Kiss, F. Costa, and J. Imrek, "Cru specification." http://twiki.cern.ch/twiki/pub/ALICE/CruHwFwSwDev/CRU_Specification_v0.7.pdf, 2016.

[3] C. Renard, S. Martinez, J. Béney, and P. Pichot, "Readout chain muon identifier: Readout architecture, functionalities and prototypes." https://indico.cern.ch/event/614004/contributions/2516144/attachments/1459304/2269519/20170530_mid_rdo5.pdf, 2017. ALICE Muon Meeting 15 May 2017, Grotta Giusti, Tuscany, Italy.

[4] A. Kluge and P. V. Vyvre, "The detector read-out in alice during run 3 and 4." https://twiki.cern.ch/twiki/pub/ALICE/CruHwFwSwDev/ALICErun34_readout.pdf, 2016.

[5] MID Group, "Proposed o2 data format." Private Communication.

[6] D. Stocco, "Mid detector mapping." Private Communication.

[7] O. Bourrion, D. Evans, J. Imrek, A. Jusko, M. Krivda, J. Kvapil, R. Lietava, L. A. P. Moreno, O. Villalobos-Baillie, and E. J. Willsher, "Interface between cts-cru and cts-detector front ends trigger note for developers." Private Communication, 2018.

[8] D. Stocco, "The muon identifer readout project production readiness review." Internal Communication, 4 2018.

[9] W. Herr and B. Muratori, "Concept of luminosity." http://cds.cern.ch/record/941318/files/p361.pdf.

[10] The ALICE Collaboration.

[11] The ALICE Collaboration, "Alice." https://home.cern/science/accelerators/large-hadron-collider.

[12] The ALICE Collaboration, "The alice experiment at the cern lhc." http://iopscience.iop.org/article/10.1088/1748-0221/3/08/S08002/pdf, 2008.

[13] The ALICE Collaboration, "Upgrade of the alice readout and trigger system, technical design report." https://cds.cern.ch/record/1603472/files/ALICE-TDR-015.pdf, 2013. ALICE-TDR-015.

[14] P. Chochula, L. Jirden, A. Augustinus, G. de Cataldo, C. Torcato, P. Rosinsky, L. Wallet, M. Boccioli, and L. Cardoso, "The alice detector control system." https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5446521, 2010.

[15] M. Marchisone, "Performance of a resistive plate chamber equipped with a new prototype of amplified front-end electronics in the alice detector." http://iopscience.iop.org/article/10.1088/1742-6596/889/1/012011/pdf, 2017.

[16] E. David, T. Kiss, and F. Costa, "Cru user requirements." https://twiki.cern.ch/twiki/pub/ALICE/CruHwFwSwDev/CRU_User_Requirements_v0.9.pdf.htm, 2016.

[17] The ALICE Collaboration, "Upgrade of the online - offline computing system." https://cds.cern.ch/record/2011297/files/ALICE-TDR-019.pdf, 2015.

[18] G. Apollinari, O. Brüning, T. Nakamoto, and L. Rossi, "High luminosity large hadron collider hl-lhc." https://arxiv.org/ftp/arxiv/papers/1705/1705.08830.pdf, 2015.

[19] CERN, "The longer term lhc schedule." https://lhc-commissioning.web.cern.ch/lhc-commissioning/schedule/LHC-long-term.htm.

[20] P. Dupieux, B. Joly, F. Jou've, and R. Vandaaele, "Upgrade of the alice muon trigger electronics." http://iopscience.iop.org/article/10.1088/1748-0221/9/09/C09013/pdf, 2014.

[21] S.Manen, P.Dupieux, B.Joly, F.Jouve, and R.Vandaele, "Feeric, a very-front-end asic for the alice muon trigger resistive plate chambers." https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6829539, 2016.

[22] e. M.Krivda and N. Meth.A, "The integration of the alice trigger system with sub-detectors." https://www.sciencedirect.com/science/article/pii/S0168900209017951/pdfft?md5=19b72c650829b5eac86c7656c503231c&pid=1-s2.0-S0168900209017951-main.pdf, 2009.

[23] D. Evans, S. Fedor1, I. Kralik1, G. T. Jones, P. Jovanovic, A. Jusko, R. Lietava, L. Sandor1, J. Urban2, and O. Villalobos-Baillie, "Alice trigger system." https://cds.cern.ch/record/814257/files/p277.pdf.

[24] M. B. Marin, S. Baron, S. Feger, P. Leitao, E. Lupu, C. S. andP. Vichoudisa, and K. Wylliea, "The gbt-fpga core: features and challenges." http://iopscience.iop.org/article/10.1088/1748-0221/10/03/C03021/pdf, 2016.

[25] P. Moreira, J. Christiansen, and K. Wyllie, "Gbtx manual." https://espace.cern.ch/GBT-Project/GBTX/Manuals/gbtxManual.pdf, 2016.

[26] The GBT-FPGA Team, "Gbt-fpga user guide." https://espace.cern.ch/GBT-Project/GBT-FPGA/Manuals/GBT_FPGA20_User_Guide_v1_40.pdf, 2016.

[27] Core CRU Team, "Alice run 3 raw data header (rdh v4)." Private Communication, 2017.

[28] "Quartus prime." https://www.intel.com/content/www/us/en/software/programmable/quartus-prime/overview.html.

[29] "Modelsim." https://www.mentor.com/products/fv/modelsim/.

[30] "Geant 4: A simulation toolkit." https://geant4.web.cern.ch/.

[31] "Fluka." http://www.fluka.org/fluka.php.

[32] "Scheduling timing using handshaking signals (fpga module)." https://zone.ni.com/reference/en-XX/help/371599P-01/lvfpgaconcepts/fpga_handshaking/.

[33] "Stategies for pipeling logic." https://zipcpu.com/blog/2017/08/14/strategies-for-pipelining.html. Accessed: 2019-09-22.