Low Power Giga Bit Transceiver (lpGBT) Test-Suite

Sander Jensen

Master's thesis in Physics

University of Bergen
Department of physics and technology

December, 2023

Acknowledgements

I want to begin by extending my appreciation to Professor *Johan Alme*, my supervisor, for dedicating a significant amount of his valuable time to mentor me through both the academic and mental aspects of my thesis. I am grateful to the *Institute of Physics and Technology* for providing the necessary facilities and a professional work environment, and a special thanks to the Microelectronics group for making me feel welcome from the start, contributing to a positive and collaborative atmosphere.

To my fellow students in room 312, thank you for brightening my days and providing the motivation to get to the university even on the greyest of days. Your collective encouragement has played a significant role in making this academic journey both enriching and enjoyable. A special shout-out to *Louis*, the office dog, for expanding your duties from sleeping to providing some therapy as well.

I want to express my gratitude to my family. My mother, who consistently supports me and would be proud even if I pursued a career as a bank robber. Thanks to my father and grandfather for nurturing my passion for science and technology. And to my sisters, thank you for being delightfully annoying.

Finally, I give a huge thanks to *Alica* for being a motivator in my life, diverting my attention during stressful moments with ramblings about plants and paint colors.

Abstract

The ALICE detector, one of the key instruments at CERN's Large Hadron Collider (LHC), is dedicated to capturing and measuring the properties of particles generated in high-energy particle collisions. To enhance its capabilities for studying these collisions, the ALICE detector is set to incorporate the FoCal sub-detector for the next experiment run. This sub-detector consists of various sensors for data collection, including pixel layers. UiB has taken the lead in incorporating these pixel layers, which are equipped with custom sensor chips designed to gather extensive data from particle collisions. A challenge lies in efficiently offloading this data from the sensors to servers for later analysis. This is particularly challenging due to the substantial volume of data and the impact the radioactive environment created by the collisions have on electronic components. To address this challenge, a comprehensive system is required, featuring custom-made components designed to withstand radiation near the sensors and commercially available components in radiation-free areas. Central to this data offloading process is the Versatile Link+ ecosystem, a set of components designed by CERN engineers to bridge these two environments. To successfully integrate these components with the pixel layers, a thorough understanding of their operation and utilization is crucial.

This thesis aims to explore and gain this understanding, encompassing the specifications crucial for both the pixel layers, and the Versatile Link+ ecosystem. Furthermore, it initiates the development of a test-suite for this ecosystem, serving as a practical platform for hands-on experience with its components. This test-suite holds the potential to contribute to the final design of the pixel layers readout system. The primary focus of this thesis has been on the development of FPGA designs responsible for managing communication with the lpGBT chip, the front-end component of the Versatile Link+ ecosystem. These FPGA designs play an important role in facilitating data exchange between the back-end and the front-end of the system, ensuring the establishment of a functional link. Extensive verification of the designs have been conducted through simulations, employing independent testbenches and a simulation platform developed by CERN for a complete Versatile Link+ ecosystem simulation.

Contents

List of Figures

List of Tables

List of Footnotes

Acronyms

ADC Analog to Digital Converter.

ALICE A Large Ion Collider Experiment.

ALPIDE ALICE Pixel Detector.

ASIC Application-Specific Integrated Circuit.

BC Bunch Crossing.

BFM Bus Functional Model.

CAN Controller Area Network.

CDC Clock Domain Crossing.

CDR Clock and Data Recovery.

CERN European Organization for Nuclear Research.

CLPS CERN Low Power Signaling.

CMOS Complementary Metal-Oxide Semiconductor.

COTS Commercial Off-The-Shelf.

CRC Cyclic Redundancy Check.

CRU Common Readout Unit.

CTP Central Trigger Processor.

DAC Digital to Analog Converter.

DAQ Data Acquisition.

DCS Detector Control System.

e-Links Electrical links.

EC Extrnal Control.

EOP End Of Packet.

FEC Forward Error Correction.

FIFO First-In-First-Out.

FLP First Level Processor.

FMC FPGA Mezzanine Card.

FoCal Forward Calorimeter.

FPGA Field Programmable Gate Array.

FSM Final State Machine.

GBTx GigaBit Transceiver.

GPIO General-Purpose Input/Output.

GUI Graphical User Interface.

HB Heart Beat.

HDLC High-Level Data Link Control.

HEP High-Energy Physics.

HGCROC High Granularity Calorimeter ReadOut Chip.

HIC High Integration Chip.

HL-LHC High Luminosity LHC.

HPC High Pin Count.

IB Inner Barrel.

IC Internal Control.

IO Input/Output.

IP Interaction Point.

IPbus Interconnect Protocol bus.

ITS Inner Tracking System.

ITS2 Inner Tracking System for Run 3.

LHC Large Hadron Collider.

LHCb Large Hadron Collider beauty.

LM Latency Measurement.

lpGBT Low Power GigaBit Transceiver.

LS Long Shutdown.

LSB Least Significant Bit.

LVDS Low-Voltage Differential Signaling.

MAPS Monolithic Active Pixel Sensor.

MGT Multi Gigabit Transceiver.

MSB Most Significant Bit.

MT Mechanical Transfer.

MUX Multiplexer.

OB Outer Barrel.

PCB Printed Circuit Board.

PCI Peripheral Component Interconnect.

PLL Phase-Locked Loop.

PRBS Pseudorandom Binary Sequence.

QGP Quark-Gluon Plasma.

RDH Raw Data Header.

ROM Read Only Memory.

RU Readout Unit.

SC Slow Control.

SerDes Serializer/Deserializer.

SEU Single Event Upset.

SOP Start Of Packet.

SRAM Static Random-Access Memory.

TC Transition Card.

TDR Technical Design Report.

TIA Trans-Impedance Amplifier.

TTC Timing and Trigger Control.

UDP User Datagram Protocol.

UiB University of Bergen.

UVVM Unviersal VHDL Verification Methodology.

VHDL VHSIC Hardware Description Language.

VHSIC Very High-Speed Integrated Circuit.

VL Versatile Link.

VL+ Versatile Link Plus.

VLDB+ Versatile Link Plus Demonstrator Board.

VTRx Versatile Transceiver.

VTRx+ Versatile Link Plus Transceiver.

VTTx Versatile Twin-Transmitter.

ZIF Zero Insertion Force.

1 Introduction

Elementary particles are considered to be the fundamental building blocks from which all matter and forces arise. In the field of nuclear physics, elementary particles are the focus of study to understand the fundamental nature of the universe. The two main categories of elementary particles are fermions and bosons, whereas fermions are the building blocks of matter, bosons carry energy and forces[1].

1.1 The Study of Quark-Gluon Plasma

The fundamental composite particles, protons, neutrons and other hadrons, are formed by quarks, a type of fermions, bound together through strong nuclear forces. This strong nuclear force is carried by bosons called gluons. Quarks and gluons which are also referred to as partons[1], are always found in bound states within hadrons, where their interactions are confined and balanced by the strong force. Only under specific extreme conditions can quarks and gluons exist in a state where they are relatively free and not confined within hadrons. This state is known as the Quark-Gluon Plasma (QGP)[2], which is believed to have existed in the very early moments of the universe, shortly after the Big Bang[2].

On the border between France and Switzerland, just outside Geneva is one of the world's largest and most prominent research centers for scientific research situated, the European Organization for Nuclear Research (CERN). In a 27-kilometre ring-shaped tunnel 100 metres underground sits the most powerful particle accelerator ever built, the Large Hadron Collider (LHC). The accelerator hosts several major particle detectors that capture and analyse the particles produced during high-energy collisions. Each detector is designed to study specific types of particles and interactions, one detector in particular, the ALICE (A Large Ion Collider Experiment) detector is designed to study the physics of heavy-ion collisions.

The conditions required to produce the quark-gluon plasma are not present under normal circumstances but can be created in high-energy particle collisions, such as those that occur in heavy ion collisions at particle accelerators like the LHC. By colliding heavy ions like lead nuclei at ultra-high energies, the extreme conditions needed to produce the QGP can briefly be recreated. The ALICE detector is equipped with a complex set of sub-detectors that capture and measure the properties of the particles produced in these collisions. Figure 1.1 shows a schematic view of the sub-detectors and components that compose the ALICE detector. The sub-detectors include tracking devices to trace the paths of charged particles, time-of-flight detectors to measure par-

[1] Partons refers to the constituents of hadrons, they include both quarks and gluons.

[2] The Big Bang is the leading scientific theory that describes the origin and evolution of the universe.

ticle velocities, and calorimeters[3] to measure particle energy. Signatures that are characteristic of the formation and evolution of the QGP can be identified by studying the patterns of particle production, energy distribution, and the correlations between different particles.

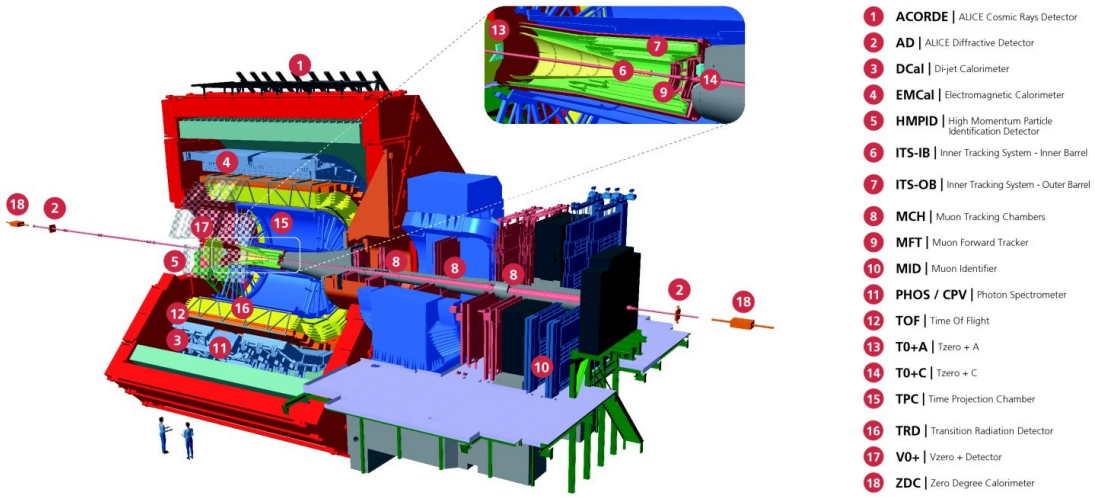


Figure 1.1: ALICE Schematics as during RUN3.[3]

The sub-detectors, along with various other components and systems are designed to study the QGP and understand the behavior of matter under extreme conditions. The dynamic nature of nuclear physics demands that these parts evolve in tandem with scientific advancements. Technological progress is also a driving factor for the constant development, offering opportunities to enhance the capabilities of the detector through the integration of cutting-edge materials, sensors, and electronics. This evolution serves to optimize experiments, enabling fine-tuning of the detector's settings and parameters to extract the most information from each collision event. Furthermore, as particle accelerators like the LHC achieve higher energies and luminosities, continuous development becomes essential to handle the increased number of particles and wider range of energies that come with these improved conditions. Handling the monumental volumes of data generated, requires ongoing innovations in data storage, processing and analysis techniques.

The LHC, where ALICE is located, operates in cycles of data-taking runs that include shutdown periods in between. The Long Shutdown (LS) provide a dedicated timeframe to perform major tasks that cannot be accomplished during shorter maintenance breaks. Figure 1.2 shows the long term schedule for these periods of data-taking and shutdowns. Run 3, which began in 2022 is expected to continue until the end of 2025. Following Run 3, there will be a three-year-long shutdown period known as LS3. An advancement intended to be incorporated into the ALICE detector during the LS3 is the Forward Calorimeter (FoCal) sub-detector.

[3]A calorimeter is a detector used to measure the energy of particularly high-energy particles such as electrons, photons and hadrons.

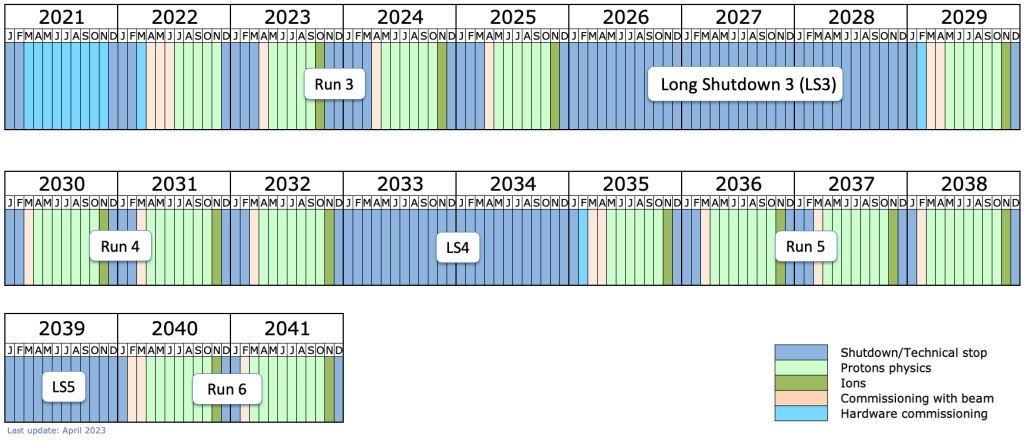


Figure 1.2: Longer term LHC schedule.[4]

1.2 Extending the Scope of ALICE

The FoCal sub-detector extends the scope of ALICE by adding new capabilities to explore the distribution of quarks and gluons within nucleons and nuclei at small values of the momentum fraction[4]. FoCal consists of a sampling electromagnetic calorimeter (FoCal-E) and a hadron calorimeter (FoCal-H). The electromagnetic part of the detector includes multiple layers of tungsten and silicon. Each layer has a certain ”granularity”, which refers to the size and resolution of the individual elements within the layer. The layers are divided into pads and pixels. The pad layers are responsible for measuring the energy and shape of particle showers. The pixel layers are crucial for distinguishing between specific particle interactions involving photons and neutral pions. The high granularity of the pixel layers allows for very precise spatial separation and identification of these particles and interactions. The hadronic part of the FoCal sub-detector uses a conventional metal/scintillating calorimeter with high granularity to accurately measure and distinguish between different types of hadronic particles, providing good resolution and energy compensation.[5]

Fully utilizing the capabilities of the detector elements are of paramount importance in nuclear physics experiments. The quality of the data collected directly influences the accuracy and precision of the scientific conclusions drawn from the experiment. A Common Readout Unit (CRU) is used to control and manage data acquisition from detector elements. A CRU is comprised of several systems, including Timing and Trigger Control (TTC), Data Acquisition (DAQ) and Slow Control (SC), which play essential roles in ensuring successful data collection and analysis.

The TTC systems ensure that detectors are fully synchronized and activated at the right moment, enabling the collection of data from events of interest and enhancing the efficiency of data acquisi-

[4]Momentum fraction refers to the fraction of the total momentum of a particle that is carried by a specific component, such as a parton within a larger particle like a proton or a nucleus.

tion. Accurate timing and precise triggering are essential for capturing data from relevant collision events while minimizing noise and irrelevant information. The DAQ system efficiently collects, digitizes, and records data from detectors. It manages the flow of data at high rates and ensures that data from various sensors are properly correlated to reconstruct complete event information. The SC systems ensure that the experimental conditions are stable and within specified ranges to maintain the accuracy and reliability of the data collected. In combination, these systems enable researchers to acquire data with precise timing, trigger on relevant events, collect complete information from detectors, and ensure the detectors operate in optimal conditions.

As crucial as the corresponding systems themselves, reliable data paths for TTC, DAQ, and SC information are vital in both directions: from the counting room to the detector (downlink) and from the detector to the counting room (uplink). To establish this link between the detector and the counting room, the FoCal sub-detector will use the Low Power GigaBit Transceiver (lpGBT) on the detector (front-end) and a lpGBT core implemented within a Field Programmable Gate Array (FPGA) in the counting room (back-end), coupled with the Versatile Link Plus (VL+). The VL+ consists of the Versatile Link Plus Transceiver (VTRx+) at the front-end and commercial optical transceivers at the back-end. Passive components, including optical fibers, connectors, fan-out patch-cords, and trunk cables, establish the connection between the VL+ front- and back-end. A block diagram of the link between the counting room and the detector is shown in Figure 1.3.

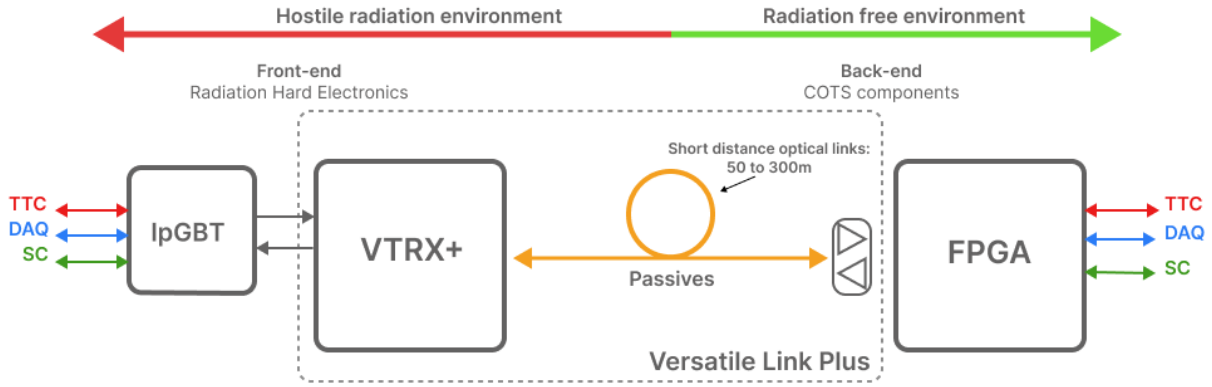


Figure 1.3: Overview of the the link between the detector and the counting room. Showing the lpGBT, the FPGA, and the components of the Versatile Link Plus: VTRx+ transceiver, passives and commercial back-end optical transceiver.

While the back-end electronics operate in a radiation free environment and can utilize Commercial Off-The-Shelf (COTS) components, the front-end electronics operate in a hostile radiation environment which requires custom made components. The Versatile Link Plus Transceiver (VTRx+) is a full custom device that combines a radiation-hardened transceiver chipset with a radiation qualified laser diode array and photodiode. "*The lpGBT is a radiation tolerant Application-Specific*

Integrated Circuit (ASIC) that can be used to implement multipurpose high speed bidirectional optical links for high-energy physics experiments" [6, p. 3]. The logical pathways for TTC, DAQ, and SC converge onto a bidirectional optical link, leveraging the inherent properties of optical communication to enable simultaneous transmission of distinct signals in both directions.[6][7]

1.3 Objective

Multiple institutes are engaged in the FoCal project, distributing the duties for the detector elements among them. The University of Bergen (UiB) assumes the primary responsibility for the pixel layers. This thesis will center its attention on the readout chain associated with the pixel layers. The readout is derived from the Inner Tracking System for Run 3 (ITS2)[8] with only minor adaptations. Noteworthy modifications within the scope of this thesis encompass:

- The Readout Unit (RU) which is based on the ITS2 RU, will have updated components where needed.
  - The lpGBT will replace the GigaBit Transceiver (GBTx).
  - The VL+ which includes the VTRx+, will replace the Versatile Transceiver (VTRx) and the Versatile Twin-Transmitter (VTTx).
- The CRU will be updated with lpGBT core FPGAs.

Successfully incorporating the lpGBT and the VL+ optoelectornic components into the RU requires a thorough grasp of their operation and the essential configuration procedures. This leads to the central objective of this thesis, which is to evaluate the lpGBT, acquiring an in-depth understanding of its functionality and the necessary steps for its configuration. Particular attention will be directed towards the back-end integration, comprehending the methods of communication with the lpGBT, as well as grasping the transmission and management of incoming data.

Commencing the evaluation of the lpGBT requires setup and proper operational understanding of the Versatile Link Plus Demonstrator Board (VLDB+), housing both the lpGBT and the VTRx+. The evaluation board allows for testing a wide range of the lpGBT's configurable settings and provide access to almost all of it signals.

An emulation system, mimicking a CRU will be developed to create a testing environment for the lpGBT. This setup requires a dedicated independent testbench to verify its intended functionality. Once the system's performance aligns with its design objectives, it will be integrated into the lpGBT simulation to validate operation with these elements. Upon successful completion of the aforementioned stages, the testing environment will be applied to the physical lpGBT chipset on the VLDB+.

When this system is finalized, it may also streamline the process for the RU prototype under development. This is due to the uncertainty about the availability of a functional CRU compatible with the lpGBT within the timeframe of RU prototype development. Even if the CRU becomes available. there remains the possibility that it might lack the necessary flexibility to effectively test the RU prototype.

1.4 Chapter overview

Chapter 2 - A High-Granularity Forward Calorimeter (FoCal) introduces the FoCal subdetector, a proposed enhancement for the ALICE detector at CERN. This chapter provides a detailed exploration of the measurement objectives and design considerations for the different components of FoCal, with a specific focus on the pixel layers dedicated to the electormagnetic component.

Chapter 3 - Readout Electronics focuses on the examination of readout electronics, components that are important for facilitating a dependable and efficient exchange of information within the experimental setup. This chapter provides an analysis of the readout systems for both ITS2 and the FoCal-E pixel layers, offering insights into their design and operations. Furthermore, a dedicated section within this chapter provides a detailed examination of the lpGBT ASIC.

Chapter 4 - lpGBT Test-Suite introduces the key components required to establish a complete test-suite around the lpGBT ASIC. The chapter provides an in-depth examination of the design of two components for testing the lpGBT: a serial control module and a payload module. Furthermore, it presents the design for a specific communication slave developed for the serial control module.

Chapter 5 - lpGBT Test-Suite Verification and Testing outlines the steps performed to ensure verification of both the serial control module and its dedicated communication slave. This chapter delves into the specifics of diverse testbench architectures crafted to support thorough simulations and verification procedures.

Chapter 6 - Discussion and Conclusion summarizes the core content of this thesis. It features a discussion of the work, its implications, and the significance of the results. Furthermore, it outlines the future work required to enhance and finalize the test-suite. This chapter provides a critical reflection on the journey taken in this thesis and offers a compass for the path ahead.

2 A High-Granularity Forward Calorimeter (FoCal)

The essence of the FoCal physics program is directed towards tackling demanding measurements that come with specific design and performance requirements. This section aims to provide an understanding of these measurements and the approach that FoCal intends to employ in accomplishing them. Provided is an overview encompassing the various constituent elements comprising the FoCal sub-detector. This chapter delves into their distinctive measurement objectives and design intricacies. The foundational components of FoCal encompass the Hadronic Calorimeter (FoCal-H) and the Electromagnetic Calorimeter (FoCal-E). Special attention will be devoted to the FoCal-E, delving deeper into its design and operation.

2.1 Background and Motivation

The FoCal sub-detector is a proposed enhancement for the ALICE detector at the CERN LHC. As of the third quarter of 2023, the FoCal experiment is in its conclusive phases of crafting a Technical Design Report (TDR). The sub-detector will be installed during LS3, should it receive approval, with operational readiness aligned for data acquisition as run 4 initiates in 2029. The planned position for the FoCal sub-detector is situated outside the solenoid magnet, at a distance of seven meters from the ALICE interaction point, as illustrated in figure 2.1.

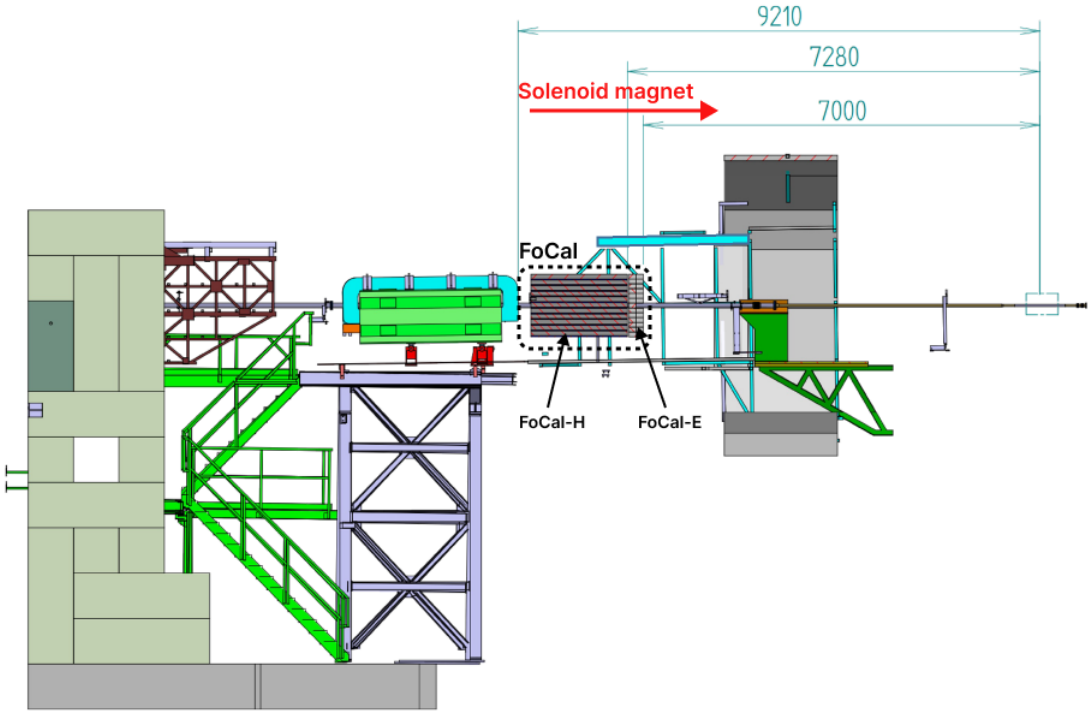


Figure 2.1: FoCal sub-detector layout in the ALICE experimental area.[9]

The inclusion of the FoCal sub-detector will enhance the ALICE detector by enabling precise investigation of the inner structure of nucleons and nuclei at very fine scales and low energy levels. The primary objective is to unravel the distribution of partons within these particles, in a domain that has yet to be thoroughly explored. The specific region of interest relates to a domain characterized by elevated parton density within nucleons. This heightened density is thought to give rise to non-linear dynamics[5], where the interactions between partons become complex due to their close proximity. FoCal will analyze these phenomena across a spectrum of collision types, encompassing proton-proton (pp) as well as proton-lead (p-Pb) interactions.[5]

2.2 FoCal-H (Hadronic Calorimeter)

While the FoCal-H is an integral component of the FoCal sub-detector, it does not occupy a central role in this thesis. However it is noteworthy that the FoCal-H will utilize the lpGBT ASIC for its RUs. The FoCal-H is designed as a high-granularity copper-scintillating fiber spaghetti calorimeter. The expected dimensions of the FoCal-H is approximately $90 \times 90 \times 110cm^3$, offering a resolution of about $2 \times 2cm^2$[9]. In a spaghetti calorimeter, thin scintillating fibers or strips are arranged in a grid-like pattern. Scintillating materials emit light when charged particles pass through them. These fibers are sensitive to the energy deposited by particles, and the emitted light is collected and measured. By analyzing the amount of light produced, the energy of the particles that interacted with the fibers can be determined[10]. The term ”spaghetti” comes from the appearance of these fibers, which can resemble thin strands of pasta as illustrated in figure 2.2.

Figure 2.2: The scintillation fiber bundles for the FoCal-H prototype used in the 2021 test beam.[11]

[5]Non-linear dynamics refers to a situation in which the relationship between different variables or components is not proportional or straightforward.

This type of calorimeter design is chosen for its high granularity, meaning it can provide detailed information about the distribution of energy within a particle shower, where multiple particles are produced by a single high-energy particle. The FoCal-H enhances the measurement of energy around a detected photon that is considered separate from any other particle interactions. This enhancement leads to better identification of prompt photons[6]. The FoCal-H also helps with the measurements of clusters of particles resulting from high-energy collisions, also know as jets[9].

2.3 FoCal-E (Electromagnetic Calorimeter)

As previously indicated, the silicon layers within the FoCal-E component exhibit distinct levels of granularity, specifically involving pixel and pad layers. An individual FoCal-E module encompasses 20 layers, with two of these layers being pixels, while the remaining 18 layers are pads. Figure 2.3 showcases the configuration of the FoCal-E module, providing an illustrative representation of the layered granularity levels. The FoCal-E, positioned as closely as possible to the FoCal-H, will comprise a collective total of 22 FoCal-E modules which are stacked in pairs of 11 on both sides of the beam pipe[9].

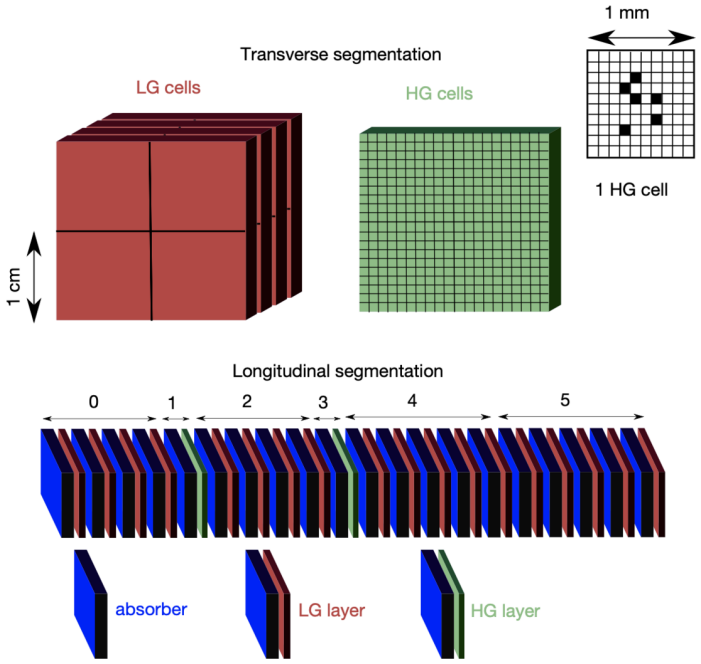


Figure 2.3: Schematic view of an FoCal-E module, depicting the layers of tungsten absorber and silicon sensor. 18 layers of pad sensors with low granularity cells, and 2 layers of pixel sensor with high granularity cells at position 5 and 10.[9]

[6]prompt photons refers to photons that are produced promptly in particle interactions, rather than being the result of subsequent decays or interactions of other particles.

The FoCal-E is designed as a Si+W (Silicon + Tungsten) sampling calorimeter. This type of calorimeter combines layers of silicon detectors and tungsten absorber material to accurately measure the energy of particles as they pass through the detector. The silicon detectors, constructed from semiconductor materials, possess the ability to identify charged particles as they traverse the material. By measuring the ionization produced by these charged particles, the energy of the particles can be determined[12]. The tungsten layers efficiently triggers electromagnetic showers and absorbs a notable fraction of the particle's energy[13]. The high-density of tungsten leads to the formation of showers in a more confined region, thereby minimizing transverse dispersion[14]. FoCal-E is dependent on a small transverse shower size to achieve optimized shower separation and to minimize the occupancy effects. Consequently, assisting the FoCal sub-detector in achieving its most challenging measurements.

When electromagnetic particles such as photons and electrons interact with the detector material, they give rise to electromagnetic showers. These showers consists of a cascade of secondary particles created through consecutive interactions. By designing the detector to minimize the shower sizes, it becomes possible to accurately distinguish between showers produced by photons and showers originating from other particles like electrons and hadrons, and thereby optimize photon shower separation. Additionally, reducing shower sizes also lowers the likelihood of multiple showers from various particles overlapping within the same detector elements. This, in turn minimizes the occurrence of high occupancy and ensures that each detected signal corresponds more reliably to a single particle interaction.[9][15]

In summary, the FoCal-E component of the sub-detector is designed with two main objectives. To ensure that the detector can effectively distinguish between electromagnetic showers produced by photons and those produced by other particles. And to minimize the occurrence of multiple particles triggering the same detector elements, which helps maintain accurate energy measurements and reduce potential uncertainties.

2.3.1 Pad Layers

The 18 pad layers within each FoCal-E module primarily serve the purpose of measuring the energy of electromagnetic showers and offer insights into the longitudinal shower profile by individually reading out each layer. Each pad layer undergoes readout and digitization by the High Granularity Calorimeter ReadOut Chip (HGCROC). Subsequently, data is transmitted via CERN Low Power Signaling (CLPS) lines from the HGCROC to *concentrator boards*, responsible for executing zero-suppression[7] and then transmitting the data to the counting room. A concentrator board manages power control and gathers data from four rows of five front-end PCBs equipped with the HGCROC.

[7]Zero-suppression refers to the removal of redundant zeros from data.

To transmit the data to the counting room, the board utilizes two lpGBTs along with the VTRx+ and optical fibers, effectively providing all the necessary uplink and downlink capabilities required for control, monitoring, and data acquisition.

2.3.2 Pixel Layers

In each FoCal-E module, the two pixel layers are equipped with ALICE Pixel Detector (ALPIDE) Monolithic Active Pixel Sensor (MAPS)[8] chips custom-developed for the ALICE ITS. These ALPIDE chips feature a matrix of 1024×512 pixels, along with Digital to Analog Converters (DACs), Analog to Digital Converters (ADCs), and digital peripheral components[16]. The ALPIDE chips original design, targeted tracking environments with relatively low occupancy rates. Both test beam results and simulations confirm the chips capability to effectively manage the expected occupancy levels, such as those encountered in electromagnetic showers[5]. A single pixel module is formed by arranging six strings, each containing 15 or 12 chips, resulting in a total of 90 or 72 ALPIDE chips. Beyond the strings, the module includes a Printed Circuit Board (PCB) which serves as a Transition Card (TC) connecting the module to the off-detector electronics, along with a mechanical structure that securely houses the module and its components[9].

The ALPIDE features a single design but offers three modes of operation, including Inner Barrel (IB), Outer Barrel (OB) Master, and Outer Barrel (OB) Slave, each tailored to specific use cases. These distinct modes exhibit varying behaviors in terms of slow control interface, data interface, and data transmission rates. The chips selected mode is permanently set and cannot be reconfigured via any configuration registers. For the slow control interface, the ALPIDE chip features two ports, with port activation determined by the operational mode of the chips. Specifically, the IB chips exclusively utilize the differential control port, while the OB Slave chips utilize the single-ended control port. The OB Master chip employ both control ports, serving as a central hub for slow control. The OB Master chip efficiently relays control transactions received via the differential control port to the OB Slave chips through the single-ended control port.[17]

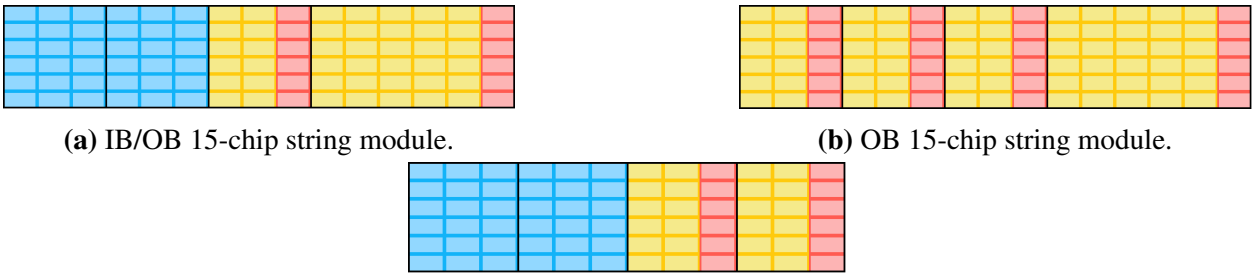
The ALPIDE also provides two interfaces for data transmission: a serial transmission port and a parallel data port. Table 2.1 outlines which data transmission interfaces are used by the various modes and their corresponding transmission speeds. The Serial data ports used by OB Master and IB chips serves to transmit data from the ALPIDE to the RU. This link employs 8b/10b encoding, a method that utilizes 10 bits to convey 8 bits of data. Further details on data encoding and decoding will be covered in 3.4, `Enhancing Data Transmission Reliability`. As a result, the IB chip and the OB Master chip achieves an effective data rate of 960Mbps and 320Mbps respectively. In contrast, the parallel data link is shared by the OB Slave chips and the OB Master.

[8] A MAPS chip integrates sensitive elements and readout circuits onto a single die.

ALPIDE Mode	Serial Link Utilized	Parallel Port Utilized	Description
IB	Yes	No	Programmable Serial Link Speed. Default is 1200 Mb/s, but can optionally use 400 Mb/s or 600 Mb/s
OB Master	Yes	Yes	Samples the parallel port and forwards stream to Serial Link Transmission. Serial Link transmission at 400 Mb/s
OB Slave	No	Yes	Transmits data over parallel port. Data is then to be serialized and transmitted on Serial Link by Master

Table 2.1: Summary of transmission interface utilization for the ALPIDEs depending on modes of operation.[17]

OB Slave chips transmit their data from the parallel data port to the OB Master, where it undergoes serialization before being transmitted to the RU through the serial link. Unlike the serial link, data on this port is not encoded, but it maintains the same data rate as the effective 320Mbps rate of the OB Masters serial port.[17]



(a) IB/OB 15-chip string module.

(b) OB 15-chip string module.

(c) IB/OB 12-chip string module.

Figure 2.4: The three different pixel modules that comprise a full 22 module pixel layer. Blue cells represents IB ALPIDEs, red cells represents OB Master ALPIDEs and yellow cells represents OB Slave ALPIDEs.

Each pixel module is comprised of six identical multi-chip strings, the operating modes of the ALPIDEs within these strings depend on their position relative to the beam pipe. There are three different configurations of strings, known as the IB/OB 15-chip string, OB 15-chip string, and IB/OB 12-chip string. The strings are organized into four groups, starting from the point closest to the beam pipe. In the IB/OB 15-chip string, there are two groups of three IB chips, followed by OB chips arranged in groups of three and six. As for the OB 15-chip string, it consists of three groups, each comprising three chips, followed by one group comprising six chips, all operating in OB mode. In the IB/OB 12-chip string, the first three groups are identical to the IB/OB 15-chip string, but the last group consists of three OB chips instead of six. Each OB group includes one

OB Master chip, with the remaining chips operating in OB Slave mode.[9] Since each individual pixel module consists of six identical strings, there are three distinct pixel modules, as illustrated in Figure 2.4.

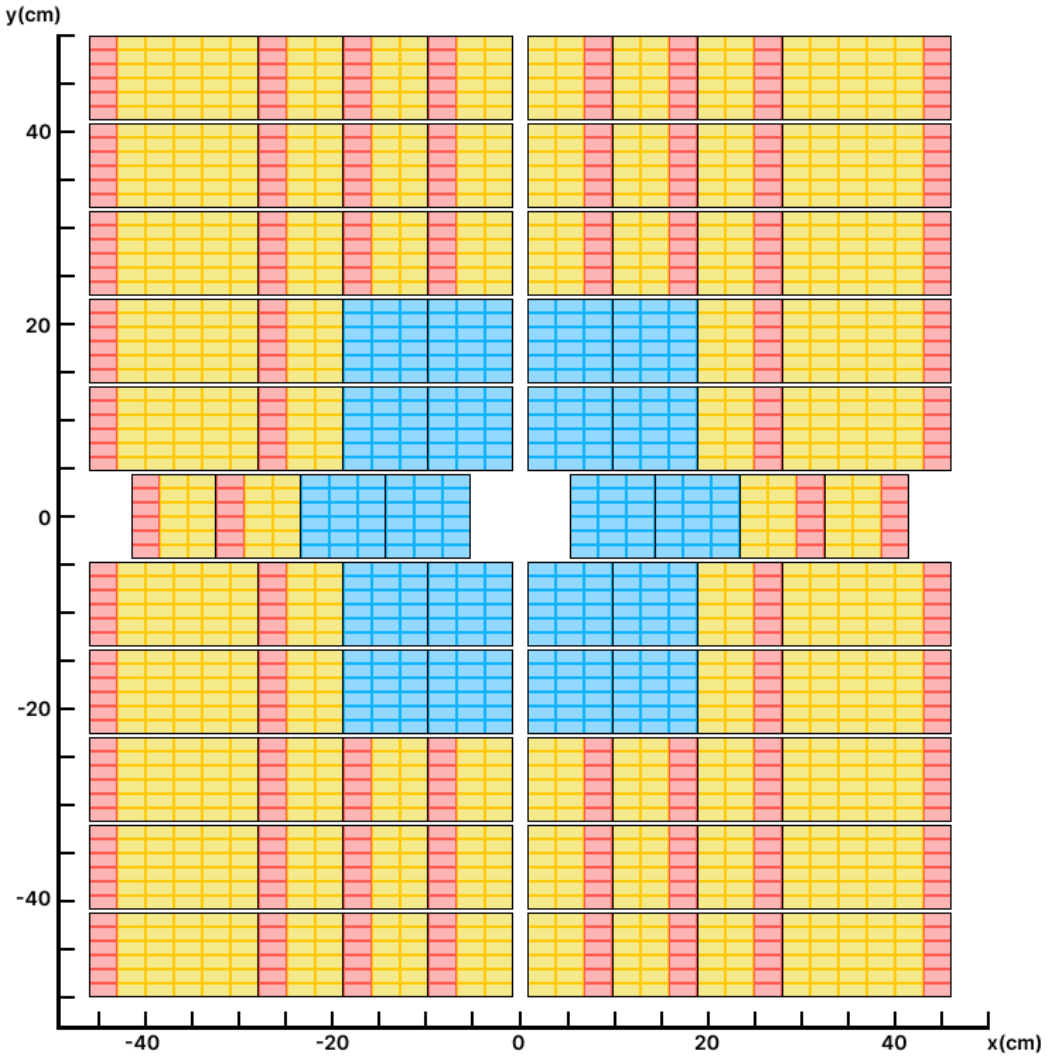


Figure 2.5: Schematic view of the arrangement of pixel modules (individually represented in Figure 2.4) in a complete 22 module pixel layer.

In Figure 2.5, the pixel modules depicted in Figure 2.4 are arranged according to the planned configuration of the 22-module pixel layer, the blank space in the middle is where the beam pipe will go. The chips positioned closest to the beam pipe are expected to record a higher frequency of hits compared to those situated further away, resulting in a greater volume of data to be transmitted. This consideration influences the arrangement of pixel modules, with a cluster of IB chips strategically placed around the beam pipe. As one moves away from the beam pipe, the occupancy gradually decreases, and the readout can be efficiently managed by an OB Master chip in

conjunction with two OB Slave chips. At the outermost edges of the layer, where the occupancy is significantly lower, a single OB Master chip is capable of handling the readout for itself and five OB Slave chips. Simulations indicates that the data rates for this configuration comfortably remain within the link speed specification.[9] Comprehending and predicting the data rates of the pixel modules hold significance when it comes to designing the RU, and determining the necessary quantity required to effectively manage an entire pixel layer.

2.4 General Readout of FoCal

This section provides a brief explanation on how components of the FoCal sub-detector trigger data collection, along with an overview of the general data retrieval path. A comprehensive examination of the readout chain for the pixel layers is presented in Chapter 3, `Readout Electronics`.

The LHC Clock

The LHC operates by circulating two beams of particles in opposing directions, these beams consist of discrete bunches of particles rather than continuous beams. The LHC ring is divided into 3564 distinct bunch positions, which represent locations around the ring where bunches of particles may be present but are not necessarily occupied. These positions are spaced approximately 25 ns apart from each other. Each time two of these bunches intersect (Bunch Crossing (BC)), it creates the potential for collision events that are of great interest to the experiments conducted at the LHC. These crossings occur at a frequency of 40.079 MHz, giving rise to what is known as the BC clock or the LHC clock. This clock signal is distributed to all four experiments to ensure precise synchronization of the detectors and various components within each experiment. It allows them to precisely determine when a BC event occurs at their specific location.[18]

Triggering data readout

A trigger serves the fundamental role of instructing different detector sub-systems when to initiate data recording. In context of particle detectors, triggers play a vital role in managing data volume, selecting relevant events, optimizing resource usage, and enhancing the efficiency of data analysis. Since the triggering schemes hold less significance in this thesis, a concise overview is provided to offer context. What is essential to establish is that the diverse sensor and readout technologies of the various detector sub-systems necessitate distinct trigger systems. The systems utilized for FoCal-E pads and FoCal-H are engineered to operate synchronously with the LHC clock mentioned earlier. This operation requires receiving triggers that are precisely synchronized to this clock, with a fixed latency concerning the collision time. In contrast, the ALPIDEs used in FoCal-E pixel layers are designed to function in a continuously triggered mode, where they receive an external

trigger at a consistent frequency, typically set at 100 kHz by default. When utilizing diverse trigger systems, careful planning, synchronization, and coordination among sub-systems are important to ensure seamless operation between them, leading to accurate event reconstruction and reliable data analysis. If it is advantageous to utilize correlated triggers, the pixel layers also provides the possibility of operating in a triggered mode, where the trigger signal is derived from an external source, such as hit information from the pad layers.[9]

Readout

After on-detector readout, both the pad and pixel layers of FoCal-E, as well as FoCal-H utilize the lpGBT for transmitting data from the front-end to the back-end. Optical fibers connected to the lpGBT lead to a CRU, serving as an interface bridging the front-end and the computing farm. These CRUs incorporate synchronization and pre-processing steps before forwarding the data to a server known as the First Level Processor (FLP). Each FLP is designed to accommodate up to three CRUs, each responsible for collecting data from up to 24 lpGBT links. The estimated allocation includes ten CRUs for the pads, seven for the pixels, and three for FoCal-H, resulting in a total of 20 CRUs.[9]

3 Readout Electronics

The Readout chain refers to the path of the signals from detection to acquisition. Within this path, readout electronics represents the components responsible for ensuring a dependable and efficient exchange of information. This assembly includes both on-detector and off-detector readout components, complemented by passive elements such as fibers and adapters, which establish communication between the various components.

This chapter is dedicated to the examination of these components and passives, specifically within the context of the FoCal pixel layer. As the pixel layer RU design significantly leverages the one originally developed for ITS2, the readout electronics employed here will be introduced, along with an exploration of its data transfer protocol configuration. A review of the planned readout chain for the pixel layers will be provided, with specific attention devoted to the RU and the utilization of the lpGBT ASIC within the RU. A general understanding of its functionality, operation, and configuration will be presented. Additionally, the chapter will explore the techniques and protocols implemented by the FoCal pixel layer readout to enhance the reliability of data transmission.

3.1 ITS2 Readout

The readout electronics planned for implementation in the FoCal detector draw significant inspiration from the readout electronics originally designed for the ITS2. The information presented in this section is primarily derived from S.V.Nesbø's thesis titled *"Readout Electronics for the Upgraded ITS Detector in the ALICE Experiment"*[16]. This section provides an overview of the ITS2 pixel configuration and delves into a more detailed exploration of the RU, its interfaces, and its constituent components.

3.1.1 ITS2 pixel layout

As FoCal incorporates strings featuring three distinct ALPIDE configurations, the ITS2 employs two configurations known as High Integration Chips (HICs), each containing either nine ALPIDEs in IB mode or 14 ALPIDEs in OB mode. The nine chips in the IB HIC are arranged in a single line. The OB HICs consists of two parallel lines, each housing seven chips, with one chip per line operating in OB Master mode, while the remaining chips operate in OB Slave mode. A Stave represents the smallest functional unit of the ITS2 detector, comprising HICs as well as structural and operational components. The ITS2 layers are categorized into three regions: Inner Layers,

Middle Layer, and Outer Layers. The staves within the Inner Layers are relatively simple, featuring a single IB HIC where all ALPIDEs drive their individual signals, resulting in nine high-speed links from an IB stave to the RU. The OB stave is available in two configurations, one equipped with eight OB HICs for the Middle Layers and another with 14 OB HICs for the Outer Layers, housing a total of 112 and 196 ALPIDEs, receptively. Since each OB HIC is managed by a two OB Masters for data transmission, a Middle Layer stave requires 16 data links to the RU, while an Outer Layer stave requires 28 data links.[19]

3.1.2 Readout Unit Interface

To enable high-speed data transmission from the staves to the counting room, specialized readout electronics positioned in close proximity to the detector is required. At the heart of this system is an FPGA-based RU that streamlines the retrieval of data from the links and then transmits it to the counting room using optical links. These optical links also serve the additional functions of facilitating control, configuration, and monitoring of the ALPIDEs. The high-speed Multi Gigabit Transceivers (MGTs) integrated into the FPGA, which are employed by the data links from the IB staves, represents the primary preference for handling data from interfaces operating at rates exceeding 1 Gbps. However, it is important to note that these transceivers have a minimum line rate requirement of 500 Mbps[20], rendering them incompatible for use in the front-end of the 400 Mbps data links of the OB staves. As a result, a General-Purpose Input/Output (GPIO)-based front-end is required for the OB staves.[16]

To facilitate the transfer of clock, control, and data signals to and from the staves, a specialized *Samtec Twinax Firefly* cable assembly is employed. Due to space limitations on the RU, the FireFly connectors are positioned on a separate PCB, functioning as a mezzanine card[9]. This intermediary PCB, refereed to as the transition board, establishes connections with the RU FPGA, providing access to 28 transceiver pairs and 28 standard Low-Voltage Differential Signaling (LVDS) GPIO pairs on the FPGA. The signals from the staves are directed to the transceiver or GPIO pairs through five FireFly connectors on the transition board: one connector for the IB stave interface, establishing connections with transceiver pairs, and four for the OB stave interface, linking the LVDS pairs. The utilization of a transition board extends the adaptability of the ITS RU, making it compatible with other detectors that also rely on ALPIDEs but need a different configuration of connections.[16]

For the interface between the RU and the CRU in the counting room, the ITS2 employs the GBTx

[9]Mezzanine cards are designed to be compact and contain various electronic components, connectors, and circuitry specific to their intended purpose. Commonly used in electronics and computing systems to expand or customize the capabilities of the baseboard.

in conjunction with the Versatile Link (VL) optical system. This digital ecosystem operates in a manner similar to its successor, the system involving Versatile Link Plus (VL+) and lpGBT as mentioned in the introduction. It differs in that it offers lower data rates and reduced radiation tolerance. Each GBTx chip provides a user bandwidth of 3.2 Gbps in both the uplink and downlink directions[21]. The interface for a single ITS2 RU is established using three GBTx chips, one VTTx with dual uplinks for data, and two Versatile Transceivers (VTRxs). These transceivers collectively provide two downlinks for control and trigger signals, while one uplink is shared for both data and control purposes. The entire readout chain, spanning from the ALPIDE staves through the RU to the CRU, is illustrated in Figure 3.1.

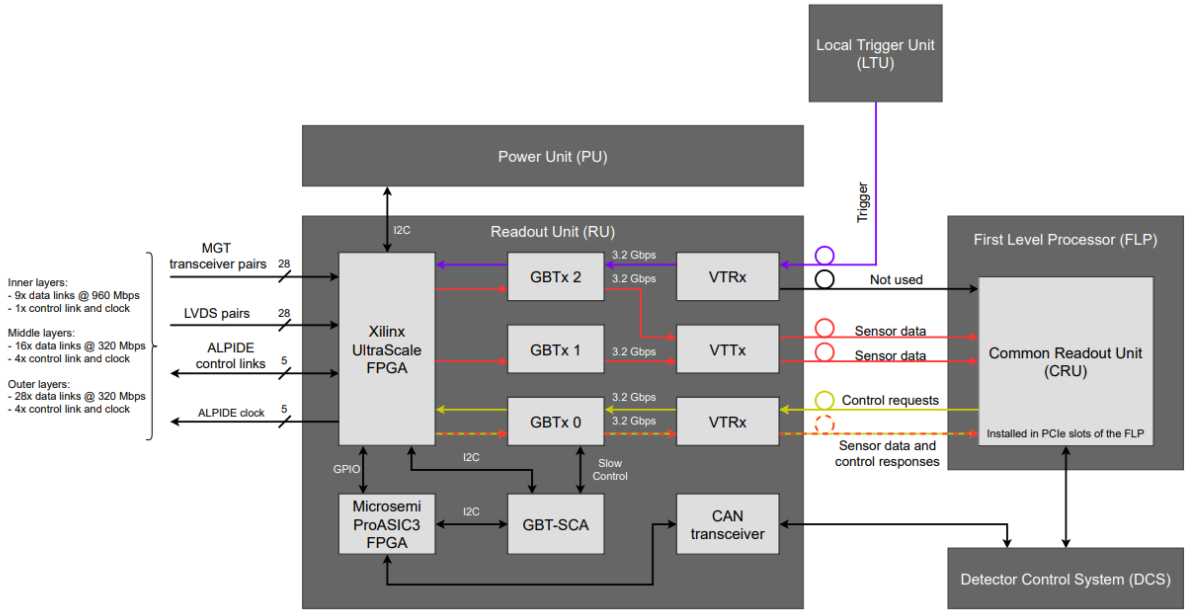


Figure 3.1: Readout chain for the ITS2.[16]

3.1.3 Readout Unit FPGAs

The ITS2 RU comprises two FPGAs. The primary FPGA is responsible for processing high volumes of data through the various high-speed interfaces previously explored. This is a *Xilinx UltraScale XCKU060*, which is based on Static Random-Access Memory (SRAM) technology. SRAM is a type of volatile memory that loses its data when power is not supplied, necessitating FPGA reconfiguration after a power-on reset. Furthermore, the configuration memory of these FPGAs is susceptible to Single Event Upsets (SEUs) when operating in a radiation environment. As a result, a scrubbing[10] process is required to ensure that the configuration remains intact throughout

[10]Scrubbing involves the continuous refreshing of the FPGA configuration memory to correct any bit flips caused by radiation.

the operation. To address these requirements, the primary FPGA is coupled with a flash memory and an auxiliary flash-based FPGA. The external memory is a non-volatile memory that stores the configuration image when the RU is without power, while the auxiliary FPGA is responsible for handling the configuration and scrubbing process. Flash memories are not susceptible to SEU-induced failures from alpha or neutron radiation, making them dramatically more reliable in radiation environments[22]. Employing FPGAs within the RU provides the flexibility for remote updates and modifications to functionality post installation. This capability is particularly significant since the RUs are challenging to physically access and are subjected to radiation exposure.

In the ITS2, a single RU is assigned to each stave. To ensure the integrity of the ALPIDE data links, these RUs must be positioned in proximity to the Interaction Point (IP) within a radiation-intensive environment. The selection and design of the components used, take into account the anticipated radiation levels, with a significant safety margin of ten.

3.2 ITS2 Data Transfer Format

A primary objective of the readout electronics is to collect and consolidate data from the ALPIDEs within the detector and then transmit this data upstream for storage and analysis. A data transfer format includes a set of specifications, rules, and conventions that define how data is structured, organized and transmitted between components. Its purpose is to guarantee the accurate and reliable transmission, reception, and interpretation of data by various components, maintaining a standardized and orderly approach. This section will explore how data produced by the ALPIDEs is organized, packaged, compressed, and uniquely identified as it traverses different stages within the readout chain, spanning from acquisition to storage.

3.2.1 Front-End Format

Every stave within ITS2 is linked to an individual RU, the quantity of data links utilized for this connection depends on the specific stave configuration. As previously detailed, nine data links are utilized for Inner Layer staves, 16 for Middle Layer staves, and 28 for Outer Layer staves. Data is transmitted in 8-bit blocks over these links, adhering to the ALPIDE data protocol. Upon reaching the primary FPGA, the ALPIDE data undergoes 8b10b decoding before being directed along data lanes corresponding to each data link. Data on each lane is independently and simultaneously processed. The 8-bit data blocks are organized into 80-bit words known as GBT words, comprising nine 8-bit data words and an 8-bit lane identifier indicating the source IB or OB master ALPIDE from which these nine data bytes originate. A round-robin multiplexer[11] gathers GBT words pro-

[11] A round-robin multiplexer is used to cyclically select and gather data from multiple sources in a sequential order.

duced by each data lane and organizes them into sequential order for transmission to the CRU, forming CRU Data Packets. In order to facilitate the recognition of CRU Data Packets, a Raw Data Header (RDH) comprising four GBT words is incorporated at the beginning of the packet. This header contains a range of information to ensure reliable transmission and analysis of data packets. Notably, it encompasses details such as RU identification, detector identification, trigger information, and quality control data.[23]

The GBT words used for CRU Data Packets have two configurations that are distinguished by an additional data valid bit. When the data valid bit is set to logical 1, it signifies user data, specifically the GBT words for ALPIDE data and the four RDH GBT words. In contrast, when the data valid bit is 0, it indicates control GBT words. There are four defined control words: Idle, Start Of Packet (SOP), End Of Packet (EOP), and Single Word Transaction (SWT). GBT data streams can consists of either a single control word, which can be either IDLE or SWT, or a stream of CRU Data packets, which are marked by the presence of SOP and EOP control words. The transfer of data from the RU to the CRU is controlled by Heart Beat (HB) triggers from the Central Trigger Processor (CTP). A HB represents the time required for a bunch to complete a full orbit around the LHC ring. With each HB trigger, a HB frame, comprising a minimum of two CRU data packets, must be conveyed from the RU to the CRU. Each packet within a HB frame commences with a SOP, followed by an RDH, and concludes with an EOP. If there is specific ALPIDE data to be transmitted for that HB, it is included in the CRU Data Packet. A maximum of 512 GBT words, excluding the SOP and EOP, can be included in the same CRU Data Packet. If more data needs to be transmitted, it must be sent in a second packet. Figure 3.2 illustrates the organization of GBT words from Outer Barrel Staves sent from the RU to the CRU.[23][16]

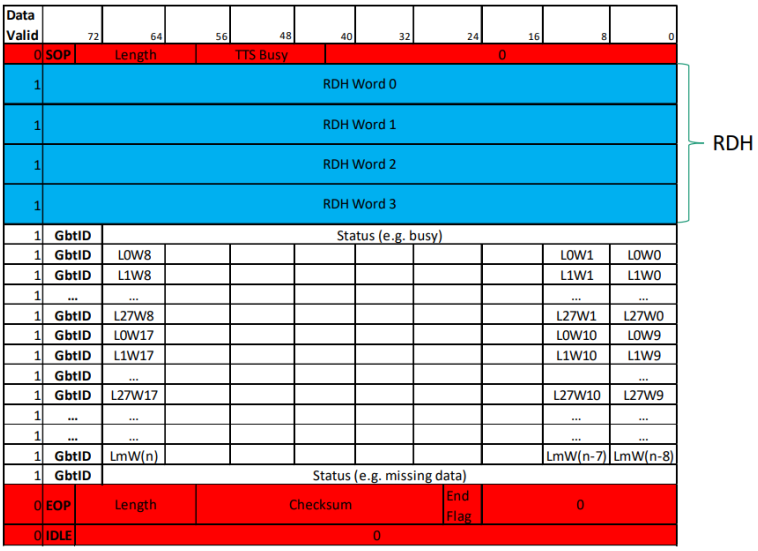


Figure 3.2: CRU Data Packet sequence for Outer Barrel Staves.[23]

3.2.2 Back-End Format

In the back-end, the data is received and undergoes processing by the CRU before being transferred to the FLP. The connection between the CRU and the FLP is engineered to facilitate the parallel transfer of 128 bits of data in each data transfer cycle. To fully utilize this bandwidth with respect to the 80-bit GBT words received from the RU, a dedicated logic is essential for packing the GBT words into a 128-bit structure. The packing involves filling the 128-bit words with 80-bit GBT words in a sequential manner. If there is not sufficient room for a complete GBT word within a 128-bit word, it is segmented and included in the next 128-bit word. Figure 3.3 illustrates one of two configurations employed for the packing process. The GBT words within the RDH of CRU Data Packets will not be subjected to the packing process. Instead, the initial word of the RDH will be enhanced with an additional 64 bits of data, including details like GBT channel identification, CRU identification, and memory size. The remaining GBT words within the RDH are filled with zeros to complete the 128-bit word.[24]

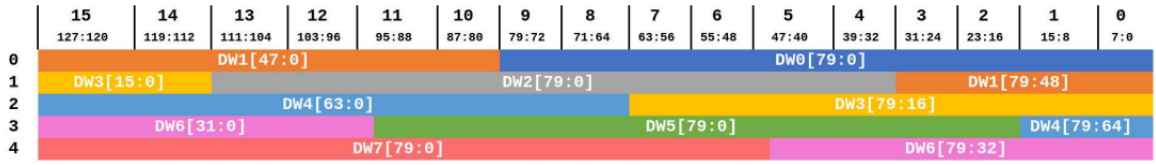


Figure 3.3: 80-bit GBT words packed in 128-bit words.[24]

3.3 FoCal-E Pixel Layer Readout

Chapter 2, `FoCal`, delved into the design and arrangement of pixel strings. This section will give an overview of the various components employed for facilitating communication between the pixels and the counting room, as well as the manner in which these components are interconnected. *If otherwise not stated, the information provided in this section is sourced from the Technical Design Report of FoCal[9], which is currently in writing. The design of the FoCal RU is also in development, the ultimate design therefore may deviate from the details examined in this section.*

3.3.1 Physical Datapath from ALPIDEs to RU

Each IB and OB Master ALPIDE utilizes a LVDS line for the transmission of data from the pixel strings. This leads to a total of either eight data lines for an IB/OB string or four lines for an OB string. In addition to the data links, each string incorporates a dedicated line for clock, a control line for every group of three IB ALPIDEs, and a shared control line for the OB Master ALPIDEs. This results in an additional four or two LVDS lines for the IB/OB or OB strings, respectively.

Table 3.1 provides a summary of the arrangement and number of lines in a six-string IB/OB or OB module.

I/O	Description	Link	Port	Number
IB/OB pixel module				
LVDS	High-speed IB data	1200 Mbps	Out	36
LVDS	High-speed OB data	400 Mbps	Out	12
LVDS	Control	40 MHz	Bi	18
LVDS	Clock	40 MHz	In	6
OB pixel module			**Total**	**72**
LVDS	High-speed IB data	400 Mbps	Out	24
LVDS	Control	40 MHz	Bi	6
LVDS	Clock	40 MHz	In	6
			Total	**36**

Table 3.1: Number of lines per IB/OB and OB pixel module[9]

The LVDS lanes originating from a pixel string is consolidated and connected through a Zero Insertion Force (ZIF) connector. An intermediate PCB, serving as a TC, supplies power to the ALPIDEs and routes the signals from the ZIF connectors to FireFly connectors, enabling the interface with the RUs. One TC is responsible for routing signals from six ZIF connectors originating from the six strings in a pixel module. As indicated in Table 3.1, the IB/OB module employs twice as many LVDS lines compared to an OB module. Consequently, the number of required FireFly connectors varies. When the TC interfaces with an IB/OB module, six FireFly connectors are employed, while three connectors suffice when it interfaces with an OB module. If the need arises to position the RUs farther than eight meters from the detector due to space constraints or the desire to reduce radiation exposure, an active repeater board can be utilized to ensure a dependable connection.

3.3.2 The Readout Unit

Unlike the ITS2 setup, where one RU is dedicated to each stave regardless of its configuration, the RU for the FoCal pixel layer will accommodate either one IB/OB module or three OB modules. Similar to the ITS2 RU, it will need distinct connections to manage the IB and OB links. In essence, the FPGA must be equipped with 36 transceivers for the IB ALPIDE data links and a total of 72 LVDS Input/Output (IO) ports for the OB ALPIDE data links. The front-end communication must also support connections for 18 clock signals and 18 bidirectional control links. To accommodate the distinct interfaces of either the IB/OB module or the three OB modules the FireFly connectors from the TC will establish connections with the RU through a mezzanine connector board designed specifically for the connected module type. The RU will link all signals with the mezzanine connector board, which will then selectively route only the necessary signals

for the particular configuration to the FireFly connectors. Figure 3.4 provides an overview of the RU for the FoCal pixel layer. The left portion of the figure showcases the connections between the mezzanine connector boards and the RU.

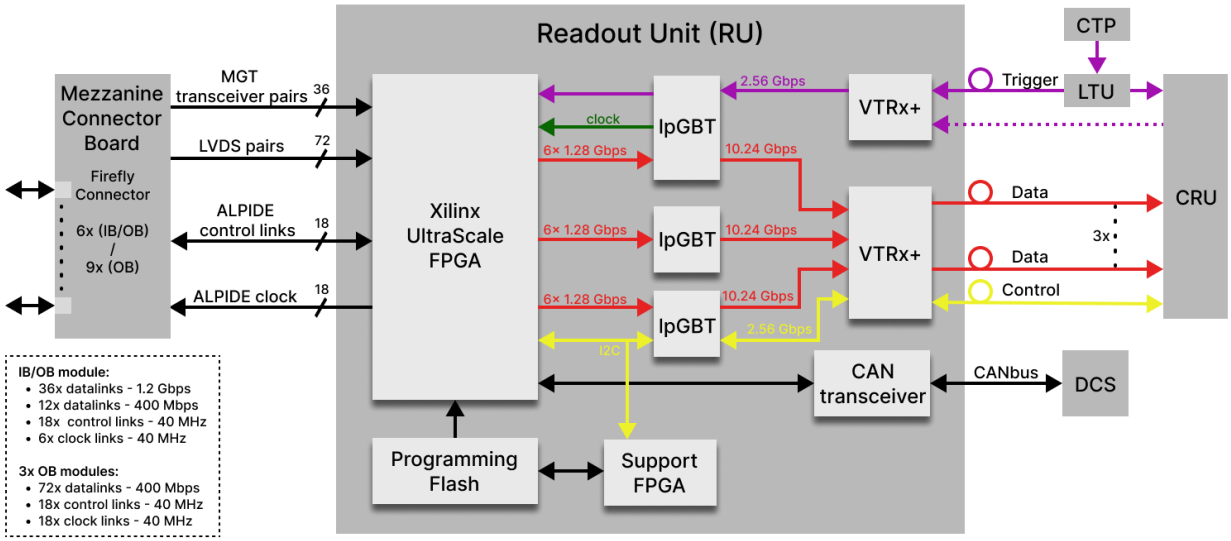


Figure 3.4: Readout Unit for the FoCal pixel layer.

The FPGA system in the proposed FoCal pixel RU shares many similarities with that of the ITS2 RU. Like the ITS2 setup, configuration and scrubbing of the primary FPGA are handled using a flash memory and an auxiliary flash-based FPGA. The recommended primary FPGA, which manages the processing of high volumes of data, is the *Xilinx Kintex Ultrascale KU085*, representing an upgrade from the one utilized in the ITS2. Additionally, another feature inherited from the ITS2, though not previously mentioned, is the inclusion of a Controller Area Network (CAN) bus link to the Detector Control System (DCS), serving as a backup communication channel.

Following data processing by the primary FPGA, the data is directed via Electrical links (e-Links) at a rate of 1.28 Gbps to lpGBT chips. Each RU is equipped with three lpGBT chips, with each chip receiving data from six e-Links. The lpGBT represents an improvement over the GBTx employed in the ITS2, enabling the implementation of high-speed bidirectional optical links that transmits data to the counting room at a rate of 10.24 Gbps. The clock signal is delivered to the front-end systems via the lpGBT, which generates a clock signal synchronized with the LHC clock from the downlink data stream. The lpGBT chips are accompanied by VTRx+ optical transceivers, responsible for driving the optical links from the RU to the off-detector CRU. The VTRx+ also signify an enhancement compared to the components utilized in ITS2, specifically the VTTx and the VTRx. Furthermore, in addition to their role in data transmission, one set of lpGBT and VTRx+ components is dedicated to the trigger link, while another set handles the control link. Detailed

exploration of both the lpGBT and the VL+ system will be presented in subsequent sections of this chapter.

3.3.3 Back-end Data Acquisition

The optical fibers originating from the RU are directed to the CRU located in the counting room. The length of this connection can extend up to the VL+ specification limit of 150 meters[7]. The CRUs are based around high-performance FPGA processors featuring multi-gigabit optical inputs and outputs. Given that the counting room is located in a radiation-free environment, COTS components are employed for the back-end electronics, including the optical receivers and transmitters responsible for establishing connections with the VTRx+[6].

The CRU functions as a common interface linking the front-end and computing systems. Additionally, it can serve as an interface for the trigger and timing system when it is advantageous to use the CRU to deliver triggers to the pixel layers[25]. The CRU, currently under development, is built upon the *PCIe40* hardware initially designed for the Large Hadron Collider beauty (LHCb). The *PCIe40* card, a Peripheral Component Interconnect (PCI) express card [12], serves as an interface between the CRU and the FLP. In the context of data acquisition, the cards primary function involves consolidating data originating from a single collision event[26]. Additionally, the CRU incorporates an FPGA that utilizes an lpGBT-core for the interface between the CRU and RUs to ensure compatibility with the lpGBT protocol. The lpGBT-FPGA aligns its encoding and decoding methodologies with those supported by the front-end ASIC, enabling accurate data encoding and decoding across the communication link.

As the CRUs are still in the development phase, the final specifications have not yet been firmly established. The allocation of CRUs and FLPs for the pixel layers will depend on the data rates of the system. The CRUs have the capacity to manage input data rates of up to 480 Gbps, and it is presumed that they can be configured to gather data from as many as 24 lpGBT links. For the pixel layers, one CRU is planned to interact with four IB/OB or OB RUs, effectively collecting data from a total of 12 lpGBT links. In total, seven CRUs will be employed for the pixel layers, with five designated for IB/OB RUs and two for OB RUs.

The fiber connections will be carefully arranged to maintain a balanced total input data rate to the CRUs and to ensure that the output data rates stay within the available bandwidth of the FLP. The maximum bandwidth for the PCI Express interface between the CRU and the FLP is 100 Gbps, while the maximum bandwidth for the FLP output data is 80 Gbps. Due to these data rates, it is expected that each FLP will be associated with one CRU, even though it is possible to connect up

[12] A PCI express (PCIe) card is a hardware component that can be inserted into a PCI Express slot to provide additional functionality or connectivity.

to three CRUs to a single FLP. After data is gathered in the FLPs, the subsequent stages of the chain will follow the standard ALICE Data Acquisition process. Figure 3.5 and 3.6 illustrates the interconnections in a simplified readout chain from ALPIDEs to FLP involving IB/OB modules, and OB modules respectively.

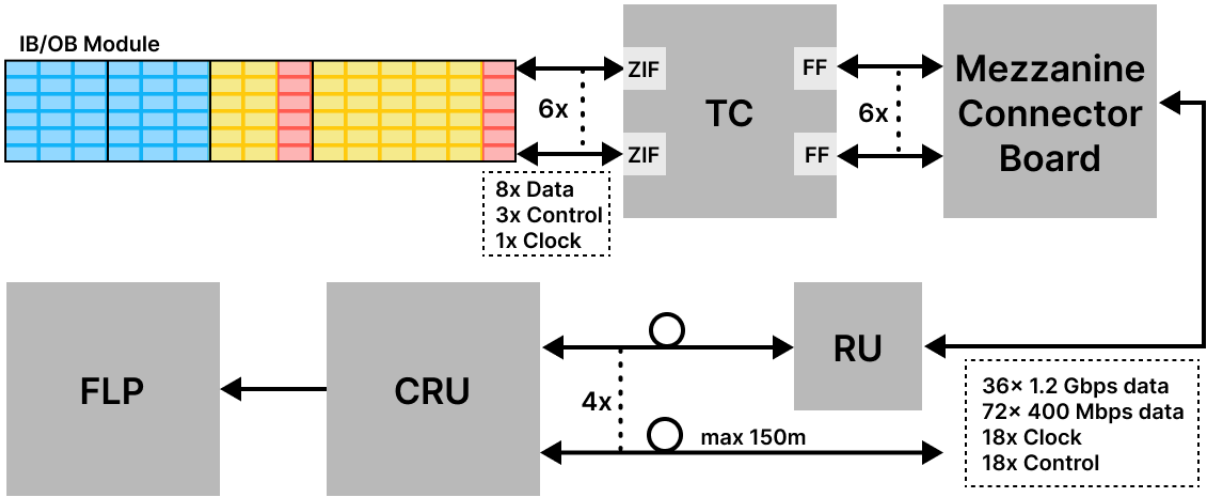


Figure 3.5: A simplified sketch of the readout chain for IB/OB pixel modules.

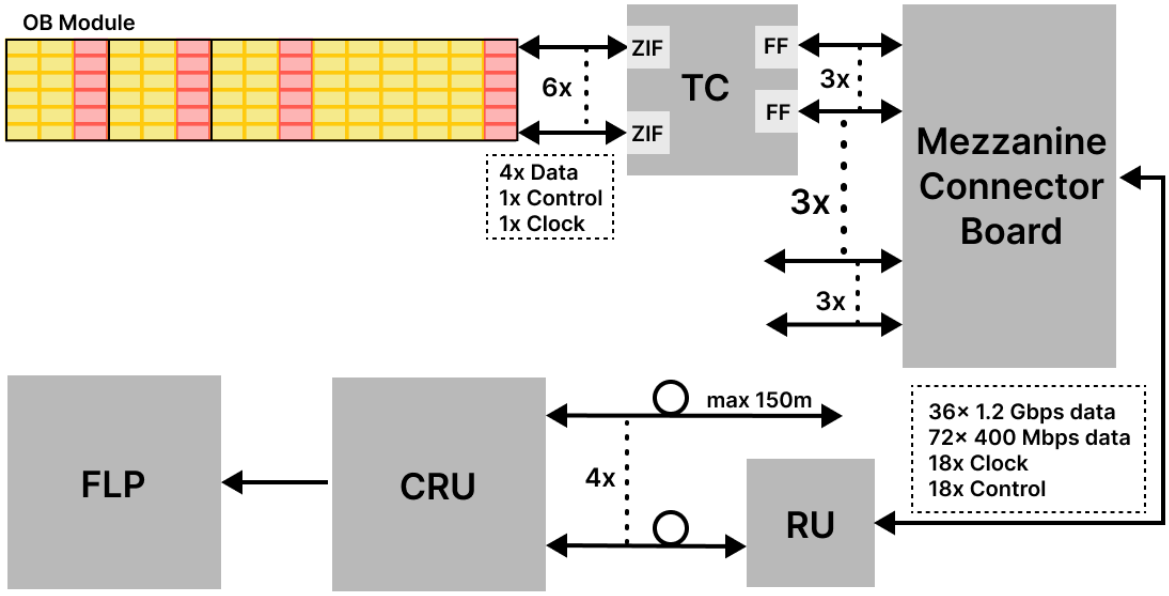


Figure 3.6: A simplified sketch of the readout chain for OB pixel modules.

3.4 Enhancing Data Transmission Reliability

In the realm of data transmission, ensuring the reliable and efficient transfer of information is paramount. High-stakes applications, such as those found in the high-energy nuclear physics and advanced communication systems, demand not only high data rates but also robust mechanisms to cope with the challenges of transmission. This section delves into the key techniques and protocols implemented by the FoCal pixel layer readout to enhance the reliability of data transmission. The section will explore how High-Level Data Link Control (HDLC) provides a framework for efficient communication between devices, investigate how Interleaving helps combat burst errors, and examine the role of Scrambling in maintaining signal integrity. It will also uncover the vital functions of Forward Error Correction (FEC) in correcting data errors and explore how 8b/10b encoding combines data efficiency with error detection.

3.4.1 High-Level Data Link Control (HDLC)

HDLC is a communication protocol which is utilized for the transmission of data between devices. It establishes a standardized mechanism for devices to efficiently and reliably send and receive data. HDLC offers a range of features and operational modes that make it a versatile and flexible data communication protocol. The following paragraphs will provide a broad overview of its implementation, with a particular emphasis on the essential features utilized in the FoCal readout system.

The data is transmitted in packets structured as frames, each comprising control information, data payload, and error-checking codes. A distinctive bit pattern known as the frame delimiter is used to distinguish the beginning and end of the frame. In HDLC, this delimiter typically comprises the 8-bit sequence '01111110'. To enable the receiver to differentiate other data within the frame from the delimiter, a technique called bit-stuffing is employed for the data bits situated between the start and end frame delimiters. Bit-stuffing is a process initiated by the transmitter, where it examines the bitstream for the occurrence of five consecutive '1' bits. When identified, the transmitter inserts a '0' bit after the fifth '1' bit, regardless of the value of the sixth bit. This introduced '0' bit serves the purpose of preventing the delimiter pattern from appearing within the transmitted frame. On the other end of the communication link, the reverse operation of destuffing is preformed. The receiver examines the incoming bit sequence for the specific pattern of five consecutive '1' bits followed by a '0' bit. When this pattern is identified, the '0' bit is removed, ensuring that the received data is successfully reconstructed as the original data content.[27]

Additional key features of HDLC includes addressing and error detection. Frames can be directed to specific devices, enabling multiple devices to share the same communication channel.

Each device uses its assigned address to determine wheter an incoming frame is intended for it. Furthermore, HDLC frames are equipped with error-checking code, such as Cyclic Redundancy Check (CRC), to identify errors that might occur during transmission. The CRC process involves the transmitter analyzing the data and generating a checksum based on its contents. This checksum is then appended to the data, forming an extended bit sequence transmitted to the receiver. Upon arrival at the receiver, the data undergo processing to create a local checksum, which is then compared to the received one. A match suggests error-free data reception, while a mismatch indicates the presence of errors, prompting the receiver to request the re-transmission of the corrupted frame[27]. This approach is advantageous for systems operating in a radiation-intensive environment where the probability of data corruption is elevated.

In addition to its error-checking capabilities, HDLC is considered a sound practice in nuclear physics experiments susceptible to radiation-induced disturbance for a variety of other reasons. Firstly, HDLC demonstrates remarkable efficiency in terms of bandwidth utilization, effectively minimizing overhead[13] and catering to the high-speed data transmission demands frequently encountered in nuclear physics experiments. Being a standardized protocol, HDLC's uniform behavior streamlines system design and guarantees effortless compatibility. This wider-ranging compatibility simplifies the process of integrating it into existing setups and allows for customization to meet the requirements of radiation-sensitive environments. Moreover, the extensive track record of the protocol in critical applications, including those exposed to radiation, underscores its reliability and performance.

3.4.2 Interleaving

Interleaving is a data processing technique employed to enhance the robustness of data transmission. It involves rearranging data elements in a specific manner before transmission. This rearranging serves to disperse data errors caused by radiation or other disturbances, across multiple data blocks. Consequently, this dispersion reduces the amount of consecutive errors, resulting in improved efficiency for error correction algorithms like CRC or FEC by simplifying error detection and correction[6]. Additionally, interleaving can contribute to balance the data rates in situations where they vary or where data from different sources are combined. This helps prevent potential data processing bottlenecks and ensures the efficient utilization of all data sources. A graphical representation of the data error dispersion achieved through interleaving is depicted in Figure 3.7.

[13] Overhead refers to the additional data or information that is included in the transmission beyond the actual payload data.

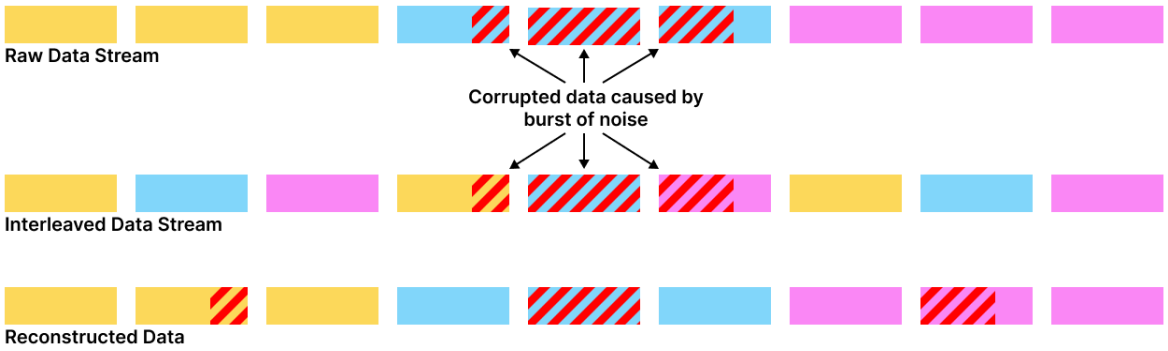


Figure 3.7: Graphical representation of the data error dispersion achieved through interleaving.

3.4.3 Scrambling

For precise sampling of an incoming bit stream, numerous systems, including the lpGBT, utilize a synchronized clock produced by a local Clock and Data Recovery (CDR) circuit. This circuit derives its clock signal by closely tracking the frequency and phase of the incoming serial bit stream. To operate effectively, CDR circuits rely on the presence of transitions between 0 and 1 within the bit stream. The more frequent these transitions occur, the better the phase and frequency information can be tracked. As there is no guarantee that transmitted data naturally contains a high density of such transitions, the transmitter can employ a process called scrambling. Scrambling is the process of deliberately introducing controlled randomness into the data before transmission. This balances the distribution of 0s and 1s in the data stream, thereby reducing the presence of long sequences of 0s and 1s and increasing the frequency of transitions. The reverse process, known as descrambling, is performed by the receiver to restore the original data.[6]

3.4.4 Forward Error Correction (FEC)

The primary goal of FEC is to enhance the reliability and integrity of data transfer, particularly in situations where data can be susceptible to noise, interference or SEUs. FEC is a technique used in data communication to detect as well as correct errors that may occur during the transmission process, thereby eliminating the need for retransmission. This capability proves especially valuable in scenarios where requesting a retransmission of corrupted data may be challenging or costly, such as in High-Energy Physics (HEP) applications, which often involve high data rates.

In order to facilitate data correction, the original data is encoded by combining it with extra bits generated through a specific algorithm. This results in a larger data frame that includes both the original information and redundant bits, which is then transmitted to the receiver. The receiver utilizes these extra bits to perform error checking and, in the event of errors, employs the sup-

plementary information to correct them. While this inclusion of extra bits enhances transmission robustness, it does come at the expense of data bandwidth.

3.4.5 8b/10b encoding

8b/10b encoding is a binary encoding technique employed to provide reliable data transmission while maintaining a balanced distribution of 0s and 1s, thereby facilitating clock recovery. In this scheme, each set of 8 data bits is mapped to a 10-bit code. This mapping ensures that the transmitted data stream exhibits frequent transitions between 0s and 1s, enabling the receiver to reconstruct the clock signal from the data stream. Additionally, the encoding method encompasses error-detection mechanisms, with specific bit patterns reserved for error detection rather than data encoding. If the receiver detects these reserved bit patterns, it signifies the presence of an error. Although some overhead is introduced by the 8b/10b encoding, it represents a minimal cost for the crucial features it provides, ensuring the accuracy of data reception.

3.5 Low Power Giga Bit Transceiver (lpGBT)

Collecting the vast amount of data from high-energy particle detectors exposed to substantial doses of ionizing radiation is a complex endeavor. The forthcoming upgrade of the LHC to the High Luminosity LHC (HL-LHC) in 2029 will expose the detectors to even higher levels of ionizing radiation due to increased energy. This higher energy will also generate a greater number of events, resulting in an increased volume of data that needs to be transmitted and processed. The lpGBT ASIC, which has been briefly mentioned throughout this thesis, is specifically tailored to meet the heightened demands imposed by the new operating conditions of the HL-LHC, as the previously used GBTx in ITS2 will no longer suffice. Given that the lpGBT is engineered by a team of highly skilled professionals, this section will not delve into the intricate technical details of its operation. Instead, the emphasis will be on gaining a general understanding of how the lpGBT functions and exploring its configuration and external operations in more detail. *The content in this section is derived from the lpGBT manual[6] unless stated otherwise.*

3.5.1 Functionality

SEUs pose a significant challenge for ensuring error-free data transmission in High-Energy Physics (HEP) applications. To address this issue, the lpGBT ASIC has been purposefully designed to mitigate the effects of radiation induced by the high-energy particle collisions that take place in the HL-LHC environment. The chip itself leverages a highly radiation-qualified commercial 65 nm Complementary Metal-Oxide Semiconductor (CMOS) technology, incorporating specialized layout techniques to mitigate the impact of radiation in internal logic and registers. Furthermore,

the lpGBT incorporates a robust line coding and error correction scheme to counteract the impact of radiation on the transmission lines. The ASIC serves as the core component for establishing a reliable high speed optical link that utilizes a single link for bidirectional transmission of various data types. For the FoCal pixel layer RU, the data types conveyed over the link primarily include slow control information and the data collected by the ALPIDEs. Additionally, the link has the capability to transmit trigger information if required. The details regarding the transmission of these multiple data types over the link will be elaborated upon in 3.6, `Versatile Link`.

The connections linking the lpGBT to the counting room are optical links categorized as downlink and uplink. To ensure data integrity, both the uplink and downlink frame incorporates Forward Error Correction (FEC) coding, which safeguard all elements within the link against errors that may occur due to noise or SEUs. The lpGBT employs *Reed-Solomon* Codes[28], a frequently employed FEC code known for its robust error correction capabilities, especially in scenarios where errors can occur in bursts. The downlink is responsible for transmitting data from the counting room to the lpGBT, this data is transferred at a speed of 2.56 Gbps with a 64-bit frame, and employs FEC12, which is capable of correcting up to 12 consecutive bits. The uplink, on the other hand, which plays the role of transmitting data from the lpGBT to the counting room, is configurable to accommodate various data rates and FEC codes. It can be programmed to operate at either 5.12 Gbps with a 128-bit frame or 10.24 Gbps with a 256-bit frame, and the FEC coding can be adjusted to either FEC5 or FEC12. Consequently, the effective data rate is determined by the specific configuration of the uplink.

In addition to its flexible uplink configuration, the lpGBT also offers versatility in terms of transceiver modes. These modes includes simplex transmitter, simplex receiver, and transceiver, all of which have an influence on the its functionality, operation, and configurability. The lpGBT chips utilized in the FoCal pixel RU are planned to operate in **Transceiver** mode specifically configured for a worst-case scenario featuring data rates of 10.24 Gbps and FEC12 encoding. Consequently, this particular combination of transceiver mode and configuration will receive primary attention in this section. When operating in **Transceiver** mode, the lpGBT makes use of both the uplink and downlink, functioning as a bidirectional data communication link between the front-ends and the counting room.

As mentioned earlier, the lpGBT system incorporates an lpGBT FPGA on the counting room side of the connection. This FPGA is employed to ensure compatibility with the lpGBT protocol, serving as a transmitter for the downlink and a receiver for the uplink. To ensure precise sampling of the incoming data stream by both the lpGBT FPGA and the lpGBT ASIC, they utilize CDR and Phase-Locked Loop (PLL) circuits to generate a high speed clock that perfectly matches the frequency and phase of the incoming serial bit stream. As outlined in section 3.4, frequent transi-

tions between 0 and 1 are required to ensure the efficient operation of these circuits. Consequently, the initial data processing step applied to the data intended for transmission is scrambling. The next step involves encoding the scrambled data with a FEC code, specifically FEC12 for both the uplink and downlink in the FoCal pixel context. Furthermore, the lpGBT includes an interleaving step after FEC encoding, streamlining the error correction process for the FEC decoder by preventing the occurrence of concentrated error bursts. Finally, the data is serialized and transmitted over the optical link using a laser driver. At the receiving ends, the reverse process is employed to reconstruct the original data. First the serial data is deserialized, followed by de-interleaving and decoding with necessary error correction, and ultimately, the data is descrambled. Figure 3.8 depicts a visual representation of the sequential transmission processes executed by the lpGBT for both the uplink and downlink.

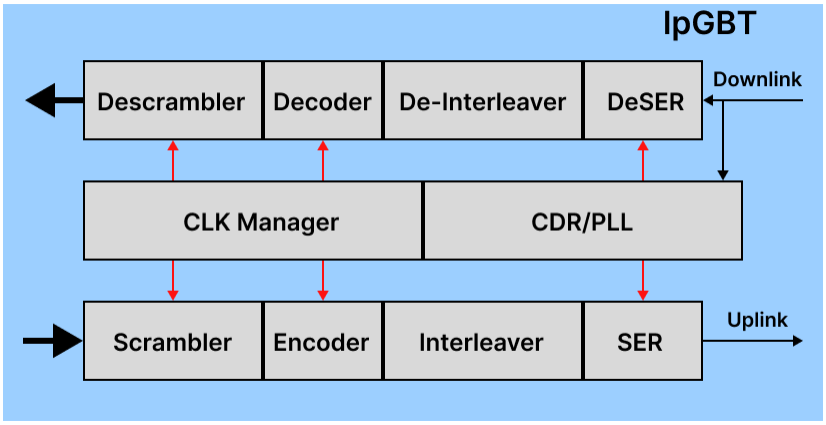


Figure 3.8: Simple illustration of the transmission processes executed within the lpGBT.

3.5.2 E-links

The communication interface between the lpGBT and the front-end is established using IO e-Links paired with associated e-Ports. These e-Links employ CLPS for transmitting data using the Most Significant Bit (MSB)-first convention between the front-end and the lpGBT, while also providing clock signals to the front-end. The e-Ports are organized into groups of four, forming what is referred to as an e-Group. Within each e-Group, the allocation of the total available bandwidth can be programmed to be transmitted via one, two, or four e-Ports. The total available bandwidth in each e-Group is equally distributed among its active e-Ports.

The programmability of the e-Ports varies between input (uplink) and output (downlink) e-Links due to the asymmetry in bandwidth between the two. Specifically, the total available bandwidth for an output e-Group is 320 Mbps, whereas for an input e-Group, it is 1280 Mbps. Additionally,

the programmability for the input e-Ports also depends on the specific configuration of uplink bandwidth and FEC encoding in use. The type of FEC encoding impacts the number of available e-Groups, with seven e-Groups for FEC5 and six e-Groups for FEC12. Furthermore, the configured uplink bandwidth determines the maximum bandwidth allocated to the active e-Links within each e-Group. Table 3.2 presents details regarding e-Group programmability and associated information for output e-Links, as well as for input e-Links within the context of an uplink configuration of 10.24 Gbps with FEC12 encoding.

eLinks	Output (Downlink)			Input (Uplink: 10.24 Gbps, FEC12)		
Bandwidth [Mbps]	80	160	320	320	640	1280
Links per group	4	2	1	4	2	1
Maximum number of eLinks	16	8	4	24	12	6
Active Channels	0, 1, 2, 3	0, 2	0	0, 1, 2, 3	0, 2	0
Frame-channel mapping	{c3[1:0], c2[1:0], c1[1:0], c0[1:0]}	{c2[3:0], c0[3:0]}	{c0[7:0]}	{c3[7:0], c2[7:0], c1[7:0], c0[7:0]}	{c2[15:0], c0[15:0]}	{c0[31:0]}

Table 3.2: E-Group programmability and associated information for output e-Links, and input e-Links within the context of an uplink configuration at 10.24 Gbps with FEC12 encoding.[6].

3.5.3 Frame

From a user's perspective, both the downlink and uplink frames consists of three fields: the Internal Control (IC) field, the Extrnal Control (EC) field, and the Data field. Both the IC field and EC field consist of two bits, consistently providing a bandwidth of 80 Mbps, regardless of the link direction. In the downlink frame, the IC field serves to transmit control information from the counting room to the lpGBT, while in the uplink frame, it conveys lpGBT status information in response to a command previously received in the downlink IC field. The EC field is associated with the EC e-Port, handling either the transmission of control information from the downlink frame, or transporting data received by the port to the counting room in the uplink frame. It is important to emphasize that these fields within the lpGBT frame are exclusively used when the chip is operating in **Transceiver** mode.

Unlike the IC and EC fields, the length of the data field, and consequently the user-available bandwidth, varies for the downlink and uplink frames. In the case of uplink frames, the length of the data field also varies depending on the uplink configuration. In the downlink frame, the data field composed of 32 bits, carries user data to the front-end via the e-Links, providing a user-available

bandwidth of 1.28 Gbps. Conversely, in the uplink frame, it accommodates data from the data input e-Ports, offering a user-available bandwidth of 7.68 Gbps when the uplink is configured to operate at 10.24 Gbps with FEC12.

Downlink

Alongside the user fields, both the uplink and downlink frames include system fields. For the downlink frame, these system fields encompass a header field and the FEC field. Given the consistent 64-bit structure of the downlink frame, the FEC field comprises 24 bits. The header field marks the initiation of the frame, presenting a fixed pattern denoted as `H[3:0] = "1001"`. The arrangement of the frame prior to the final interleaving step is depicted in Figure 3.9. This frame follows the MSB-first convention during transmission, meaning that H[3] will be transmitted first, with FEC[0] being transmitted last.

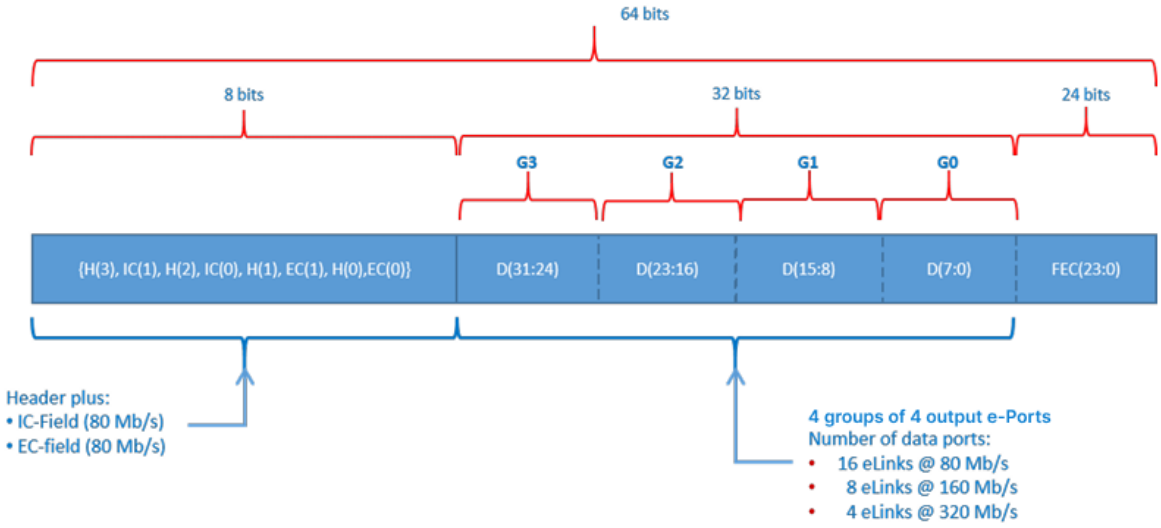


Figure 3.9: Downlink frame structure before interleaving.[6]

Figure 3.9 also illustrates the division of the 32-bit data field into four bytes, with each byte linked to an e-Port group. The initial data byte transmitted, Data[31:24], is allocated to e-Group 3. Subsequently, the data bytes are assigned to e-Group 2 and 1, concluding with the last byte of the data field transmitted, Data[7:0] allocated to e-Group 0.

Uplink

In the context of an uplink configuration at 10.24 Gbps, the frame extends to a length of 256 bits. The size of the FEC field, which depends on the error correction strength and the length of the frame, consumes 48 bits within the frame to implement FEC12 encoding, resulting in an error

correction strength of 24 bits. Alongside the FEC field, the uplink frame incorporates additional system fields, namely a two-bit header field (`H[1:0] = "10"`) and a Latency Measurement (LM) field.

The LM field serves the purpose of estimating the round-trip latency of the transceiver link, excluding the e-Links. This means estimating the time it takes for data to travel from the lpGBT FPGA to the lpGBT ASIC and then back. It achieves this by returning the two bits from the downlink IC-field, which is only operational in **Transceiver** mode. The LM field is padded with a number of leading zeros which varies depending on the specific uplink configurations. In the case of the 10.24 Gbps and FEC12 configuration, the field is padded with eight bits of leading zeros. The complete field allocation for the uplink frame in this configuration is outlined in Table 3.3.

Field	Frame	Header	IC	EC	LM	Data	FEC
Number of bits	256	2	2	2	10	192	48

Table 3.3: Uplink frame field allocation for 10.24 Gbps and FEC12 configuration[6]

The data field is segmented into 32-bit, with each segment corresponding to an e-Group. In the case of FEC12 encoding, the data field is sourced from six input e-Groups. As the uplink frame also follows the MSB-first convention, the initial 32-bit segment received, Data[191:160], originates from e-Group 5. The allocation of subsequent e-Groups is specified in Table 3.4, which also outlines the frame arrangement before the final interleaving step. In this arrangement, H[1] represents the MSB, and FEC0 is the Least Significant Bit (LSB). A corresponding table for the downlink frame is available in Appendix C.

Frame	Function	eGroup
FRMUP[47:0]	FEC[47:0]	
FRMUP[79:48]	Data[31:0]	0
FRMUP[111:80]	Data[63:32]	1
FRMUP[143:112]	Data[95:64]	2
FRMUP[175:144]	Data[127:96]	3
FRMUP[207:176]	Data[159:128]	4
FRMUP[239:208]	Data[191:160]	5
FRMUP[249:240]	{8'b0, DownIC[1:0]}	
FRMUP[251:250]	EC[1:0]	
FRMUP[253:252]	IC[1:0]	
FRMUP[255:254]	H[1:0]	=2'b10

Table 3.4: Uplink frame structure before interleaving for 10.24 Gbps and FEC12 configuration[6].

3.5.4 Configuration and Control

From the previous sections, it is clear that the lpGBT is a versatile device offering various configurations and adjustable settings. Among those discussed include transmission modes, uplink data rate, FEC coding, active e-Links, and their bit rates. There are several other settings beyond the ones listed. To set up the lpGBT for specific operations, both external configurations pins and internal configuration registers are employed. Thirteen configuration pins are utilized for the lpGBT. Among them are the **MODE** pins that determine the fundamental operating mode of the ASIC. To enable the lpGBT to work in **Transceiver** mode, with an uplink data rate of 10.24 Gbps and FEC12 encoding, these four pins should be set to `MODE[3:0] = "1111"`. There are also four **ADR** pins responsible for setting the LSBs of the lpGBT chip address for control interfaces. During chip operation, the logic level of the configuration pins must remain constant.

Along with the configuration pins, the lpGBT is equipped with numerous registers that help manage the chip. While there are various methods to access, read, or write these registers, the available options depend on the operation mode. The I2C interface slave, a serial communication protocol, is the most universal method, allowing access to write all writable registers and read from every register across any operation mode. The IC channel mirrors the capabilities of the I2C interface slave but is limited to the **Transceiver** mode. The EC channel can serve a similar control function for the lpGBT in the **Simplex** modes.

The configuration registers are grouped into four distinct categories. The initial category encompasses Read/Write/Fuse registers. During startup, these can be loaded with values from either e-fuses or Read Only Memory (ROM), which are programmed beforehand. Once the system has powered up, modifications to these registers can be made via the I2C interface or the serial control interface. The next category is the Read/Write registers, which can only be modified using the I2C or serial control interface. The third category consists of Read/Clear registers, utilized for event counters; any write access to these resets their count value to zero. The last category encompasses Read Only registers, dedicated to monitoring the status of internal blocks of the lpGBT.

Slow Control

The EC[1:0] and IC[1:0] fields within the lpGBT frame serve as a dedicated interfaces for managing slow control applications. In `Transceiver` mode, the IC field serves as the means to control and monitor the operation of the lpGBT. Simultaneously, the EC field is externally accessible, enabling the establishment of a slow control link to other chips operating in `Simplex` mode through their respective EC e-Ports. The control and monitoring of the lpGBT operation involves accessing its internal registers. When communicating with these registers, the IC fields from consecutive

frames are demultiplexed to form 8-bit words which follows a specific frame-based protocol. The protocol for transmitting data to the lpGBT for both *write-read* and *read-only* operations can be found in Table 3.5.

Write-Read	ID	Description	Parity check	Read Only
	A	Frame delimiter 8'b01111110	No	
r/w = 0	B	lpGBT address[6:0] + R/W bit	Yes	r/w = 1
	C	Command [7:0]	Yes	
	D	Number of data words [7:0]	Yes	
	D	7'b0 + Number of data words [8:8]	Yes	
	E	Memory address [7:0]	Yes	
	E	Memory address [15:8]	Yes	
	F	1st data [8 bits]	Yes	N/A
	F	...	Yes	N/A
	F	nth data [8 bits]	Yes	N/A
	G	Parity word [7:0]	Yes	
	A	Frame delimiter 8'b01111110	No	

Table 3.5: Serial control frame structure sent to the lpGBT for a write-read and a read-only operation.[6]

This protocol employs a HDLC-based approach, featuring a start and end delimiter as well as bit-stuffing and destuffing to ensure data and delimiter distinction. The 8-bit words listed in Table 3.5 are transmitted in a top-down manner, where the initial word following the delimiter comprises the specific lpGBT address and the read/write bit. Notably, each 8-bit word is transmitted using the LSB-first convention, with the read/write bit being transmitted as the first bit of the 8-bit word in the MSB of the IC field, IC[1].

Subsequently within the frame, there is a command word, in the case of lpGBT version 1, this word is not considered during downlink operations and should be configured as 8'b0. The next two words transmitted serve to indicate the quantity of data words to be either written in a write operation or read in a read operation. Out of the 16 bits within these two word, only nine bits are utilized, allowing for a maximum indication of 511 words. In cases where multiple data words are to be written or read, the memory address is incremented, enabling access to registers located in consecutive addresses. Following this, the successive words correspond to n data words intended for a write operation, no such words are transmitted in the case of a read operation. Prior to the end delimiter, the last word transmitted is the parity word, each bit in this word is the result of a parity calculation involving the corresponding bit position across all the bytes in the frame, excluding the delimiters. The lpGBT computes the same parity word from the received frame and verifies it against the transmitted one. In the event of a mismatch, no data will be written to the registers.

When the lpGBT receives a serial control frame, it acknowledges it by transmitting a corresponding frame on the uplink channel, provided that the address in the received frame matches the specific address of the lpGBT. The format of this returned frame aligns with the structure outlined in Table 3.5, except for the Command word. In the received frame, this particular word comprises seven zeros followed by a status bit for the parity verification. A successful parity check is indicated by this bit being set to 1. Between serial control frame transactions, the IC bits within the uplink frame will be set to a high state, this same action should be taken by the user for the IC bits in the downlink frame. It is worth emphasizing that in order to make use of the serial control channel, it is essential to set up fundamental configurations. These configurations can be achieved by either programming e-fuses or loading predefined settings from the internal ROM. The configuration parameters include settings for PLL/CDR, equalization, line drivers, and EC/IC channels.

3.6 Versatile Link

The lpGBT ASIC and lpGBT FPGA, which were previously discussed, are integral components of the Versatile Link ecosystem responsible for establishing the connection between the detector and the counting room. Another key element within this ecosystem is the VL+ system, which serves to create an optical interface connecting the ASIC and the FPGA. The on-detector optical interface is facilitated by VTRx+, while the back-end optical interface relies on COTS components that adheres to the system specifications.

3.6.1 VTRx+

The VTRx+ consists of various components responsible for establishing the optical interface. For the downlink, it incorporates a Trans-Impedance Amplifier (TIA) paired with a photodiode. This combination handles the amplification of the tiny current generated by the photodiode in response to received light, converting it into a more robust voltage signal, suitable for interpretation by the lpGBT. For the uplink, the VTRx+ employs a quad laser driver connected to a laser diode array. These laser drivers manage the current supplied to the laser diode, which, in turn, results in the emission of coherent light. While the downlink channel operates at 2.56 Gbps using just a single TIA, the VTRx+ module is equipped with a quad laser driver, establishing four separate uplink channels, each capable of operating at speeds of up to 10.24 Gbps. Which enables the simultaneous data transmission from four lpGBT ASICs. This design approach is taken to accommodate the asymmetrical bandwidth requirements between the downstream and upstream directions. Additionally, the VTRx+ can be configured to adapt to diverse user demands. Through I2C programming, the laser diver can be configured to disable channels that are not in use, either to meet lower uplink bandwidth requirements or when there is no need for either a downlink or

uplink connection.[29]

Since the VTRx+ is specifically designed for front-end detector operations, it is constructed to endure radiation exposure and remain functional within a magnetic field. Furthermore, it is engineered with a compact design and minimal thickness, all the while maintaining reliable optical coupling and electrical connectivity. In order to establish connections within the fiber optic network, an optical fibre pigtail, which is a length of terminated optical fiber is attached to the VTRx+ interface. The VTRx+ pigtail is terminated by a female Mechanical Transfer (MT) connector. System installation necessitates an MT-MT connection, involving the manual addition of guide pins before creating the connection. These guide pins are held in place by the MT spring clips that secures the MT connectors.[29]

3.6.2 The Versatile Link+ Demo Board (VLDB+)

The VLDB+ is a board specifically designed for the purpose to gain a thorough understanding of the Versatile Link ecosystem and to facilitate efficient testing of systems that utilize both the lpGBT and the VTRx+. This evaluation kit provides the opportunity to asses a wide range of configurable settings offered by the lpGBT. Virtually all lpGBT signals are accessible, for instance, the external configuration pins via switches, the e-Links through FPGA Mezzanine Card (FMC) High Pin Count (HPC) connectors, and the high-speed uplink and downlink channels via the MT connector on the optical pigtail connected to the VTRx+. Out of the 12 optical lines available on the MT connector, only two are utilized for communication with the lpGBT to maintain simplicity. Line six is designated for the uplink channel, while line seven serves as the downlink channel.[30]

To ensure the accurate configuration of the lpGBT, there is a dedicated control toolkit called PiGBT, built around the RaspberryPi. This toolkit is specifically designed for programming the chip using the lpGBT I2C controller, which can be accessed through a *RaspberryPi 4* and a PiGBT web application. This toolkit streamlines the efficient configuration of the lpGBT, allowing users to adjust various settings, including data transmission mode, data rates, FEC encoding, e-Group configurations, and more.[31] Detailed user manuals, providing step-by-step instructions for system setup and utilization, are available for both the VLDB+ evaluation kit[30] and the PiGBT control toolkit[31].

4 lpGBT Test-Suite

To ensure the reliable testing of the Versatile Link ecosystem, it is necessary to undertake the design and development of several elements. These elements serve as the backbone of the testing process, allowing for accurate assessment and validation of the systems functionality. However, in order to design these components effectively, a good understanding of their intended roles is essential. This understanding encompasses not only what these elements are required to accomplish but also a comprehensive grasp of the existing components within the ecosystem, including their specifications and operational intricacies. This knowledge base forms the foundation upon which the design and development efforts can be strategically and reliably executed, ultimately contributing to the successful testing and validation of the Versatile Link ecosystem.

At the start of the master thesis project, design and development priorities were established given the limited knowledge of the system and its requirements at that early stage. However, as the project has progressed, a deeper understanding of the system has emerged, revealing that some of the initial priorities and design choices may not have been the most optimal. This chapter will provide an explanation of the designed elements, while also shedding light on alternative design solutions that have surfaced over the course of the projects development. These alternatives represent valuable insights gained during the ongoing development offering the potential for improved outcomes and efficiency in the final design.

4.1 A Complete Test System

As previously discussed, the communication interface between the lpGBT and the front-end relies on IO e-Links paried with associated e-Port connections. Additionally, the communication interface between the lpGBT ASIC and the back-end is established through optical links connected to an lpGBT core FPGA. To facilitate effective communication with and data transmission through the lpGBT for testing purposes, a set of elements must be developed, both for the back-end and front-end interfaces of the system.

In the back-end, these elements play a critical role in managing the data flow into the lpGBT core FPGA, which is intended for transmission to the lpGBT ASIC. Simultaneously, they handle the data received by the FPGA from high-speed links connected to the ASIC. These elements ensure data management, simplifying data handling and ensuring the accuracy of data transmission. Furthermore, they are essential for precise processing and storage of received data, ensuring that it

can be correctly checked and verified against the expected received data. Similarly, in the front-end, specialized elements are needed to mange the data flow going in and out of the lpGBT ASIC through e-Links. These components efficiently facilitate data exchange between the front-end and the lpGBT ASIC, enabling transmission and reception of data via e-Links. By designing and implementing these key elements for both interfaces, the testing purposes becomes manageable and robust, ensuring that data transmission is accurate and verifiable throughout the Versatile Link ecosystem.

At the outset of the project, allocation of priorities among different segments of the test system was established in accordance with the available knowledge and understanding of the system requirements at that time. Recognizing the critical need to asses communication with the lpGBT through the serial IC channel, a strategic decision was made to initiate the development process by focusing on the elements for the back-end interface. This decision was motivated by the fact that controlling the data flow of the IC bits in both the uplink and downlink frame provides the capability to both write and read internal registers in the lpGBT. Thus, by concentrating on the back-end interface, it will be possible to effectively address and test the essential components required for successful communication with the lpGBT via the serial IC channel through the VL+, laying a strong foundation for the broader testing and validation efforts. The priorities within the development process will be presented below, with higher-priority elements listed first. Further details regarding the specific components and elements associated with each priority will be explored in subsequent sections, providing an insight into the development road map and the key elements driving the project forward.

- Establish data flow management for the back-end lpGBT core FPGA interface.
  - Enable bidirectional IC channel communication.
  - Enable bidirectional data payload control.
  - Establish IPbus communication for various back-end interface elements.
- Establish data flow management for the front-end e-Link interface.
- Implement Pseudorandom Binary Sequence (PRBS) generator and checker for front-end to back-end data transmission. Or a similar approach for complete VL+ uplink testing.
- Establish IPbus communication for all the elements in the test system.
- Construct IPbus API abstraction for the various IPbus modules in the test system.
- Facilitate the offloading of uplink payload data through PCI Express.

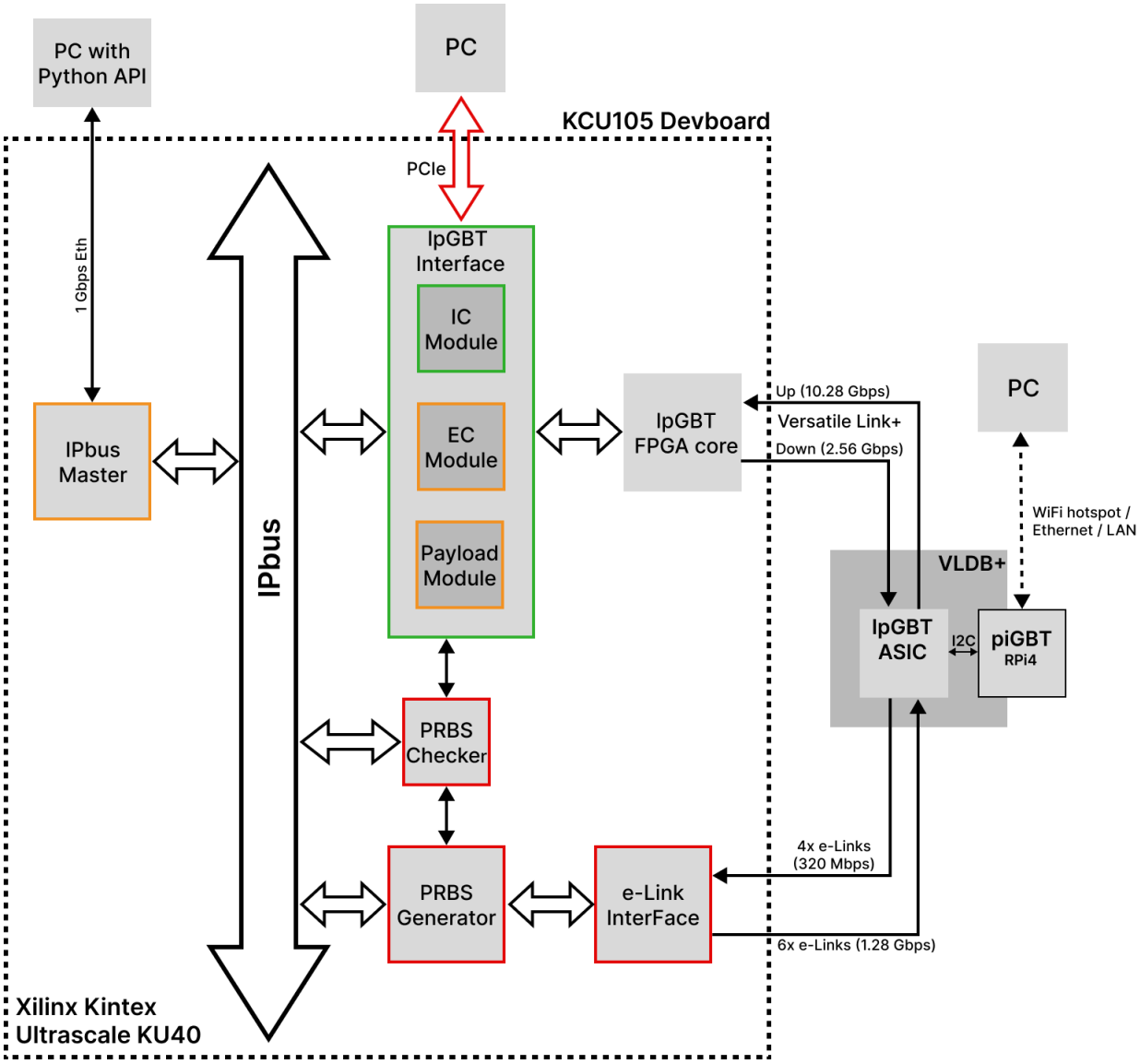


Figure 4.1: Block diagram of the essential components for the lpGBT Test-suite.

Figure 4.1 presents a simplified schematic of the necessary components for the envisioned lpGBT Test-suite. The schematic features the **lpGBT interface** component, which contributes to establishing a robust connection with the back-end lpGBT core FPGA. This component exhibits a well-structured design, divided into three primary segments. Each of these segments represents a dedicated module engineered to exercise control over the three user data fields present in both the uplink and downlink frames. These data fields include the IC, EC, and payload field. Through this segmentation, the lpGBT interface component ensures precise and efficient management of data within both the uplink and downlink frames, facilitating communication while maintaining the integrity of the frame structure. The schematic further encompasses an **e-Link interface** component,

symbolizing the connection established with the front-end lpGBT e-Ports via e-Links. This component is strategically coupled with a data producer module, conceptualized as a PRBS generator, to streamline the process of generating data intended for transmission through the VL+ ecosystem. Additionally, a data checker module is integrated to validate the incoming data received by the lpGBT Interface component. This setup ensures that data flows seamlessly between the front-end and back-end components, while the data accuracy and integrity are verified, enhancing the reliability of the entire system.

In addition to the modules for establishing connections and managing data within the VL+ ecosystem, the schematic also highlights the presence of Interconnect Protocol bus (IPbus) firmware. This firmware serves as an important component for enabling control of all the modules in the test-system, offering a high level of flexibility and user-friendliness. Through an Ethernet connection to a user-operated PC, the IPbus firmware is paired with software to enable the user to oversee and command various aspects of the system operation. This centralized control mechanism enhances the systems versatility and accessibility, ensuring that it aligns with the user specific requirements and objectives.

The IPbus firmware, in conjunction with the diverse modules and the lpGBT core, will be implemented on a *KCU105 Development Board*. This board is equipped with a powerful *Xilinx Kintex Ultrascale KU40 FPGA*, providing the necessary computational capabilities and flexibility for effectively managing the data flow and communication within the system. An implementation example for the lpGBT core FPGA is thoughtfully provided for the *KCU105 Development Board*. This example serves as a valuable reference and practical guide for configuring the FPGA on this specific hardware platform. It allows users to understand how to adapt and optimize the lpGBT core for seamless integration with the KCU105 board, ensuring that the communication and data management between the core and other system components function smoothly.

The schematic reveals the presence of a PCI Express connection on the development board, this connection serves as a valuable asset for testing purposes, particularly in the context of data offloading for storage and subsequent analysis. This capability is particularly relevant because the CRU responsible for transmission of data between the VL+ ecosystem and the FLP also employs the PCI Express. Therefore, leveraging the same communication interface for testing purposes ensures alignment with established data transfer standards and allows for an evaluation of data transfer, storage, and analysis capabilities on the board.

4.2 The Serial Control Module

As elaborated in section 3.5.4, **Slow Control**, the IC field within the lpGBT serves as a specialized interface for handling slow control applications, particularly when the ASIC operates in transceiver mode. Within this context, the downlink IC field serves as the conduit for accessing the internal registers housed within the lpGBT ASIC, enabling various operations and manipulations of these registers. Conversely, the uplink IC field is responsible for conveying information that aligns with the specific operation performed on the lpGBT registers. This bidirectional IC field communication mechanism ensures efficient management of slow control applications and seamless interaction with the internal lpGBT ASIC registers.

Both the downlink and uplink frames provide two IC bits accessible from the lpGBT core FPGA, forming a communication channel. However, to successfully utilize this communication mechanism, it is required to have both transmission and reception capabilities in place. As mentioned earlier, adhering to a predefined protocol and considering bit stuffing are critical aspects of ensuring successful communication. This section will provide insights into the design of a dedicated module aimed at facilitating this communication, offering an understanding of its structure and functionality.

The development of this IC field communication module enables testing of communication with the lpGBT ASIC. Through this module, internal registers can be both written to and subsequently read using the serial communication channel, allowing for the verification of successful operation. Additionally, the operations performed can be further confirmed through the I2C interface slave to ensure that the IC channel operates as expected. This provides a straightforward approach to confirming an operational VL+ connection between the front-end and back-end components.

The module has been tailored for seamless operation in serial communication through the IC field within the lpGBT frame, specifically when the lpGBT ASIC is operating in transceiver mode. However, due to its adherence to the serial control frame standards, this versatile module can also be employed for serial control communication through the EC field when the lpGBT operates in the simplex modes. Its adaptability across different modes of operations underscores its versatility and utility in various scenarios.

4.2.1 Downlink

The IC module is structured into two distinct segments, each with its own unique role. One segment is responsible for overseeing the transmission of IC bits from the lpGBT core FPGA to the lpGBT ASIC, while the other segment is tasked with receiving and processing the IC bits traveling in the opposite direction. The discussion will commence by delving into the IC module functionality for

enabling the transmission of the IC bits on the downlink, followed by an exploration of how it manages the reception on the uplink.

The portion of the IC module responsible for communicating with the downlink plays a crucial role in delivering IC bits to the lpGBT frame. To successfully execute this, the downlink IC module must fulfill several requirements, which are outlined below.

- It must ensure that the two IC bits in each lpGBT frame are transmitted in the correct order, adhering to the specific frame-based protocol outlined in Table 3.5 of 3.5.4, **Slow Control**.
- It must enable bit stuffing to prevent the payload data from containing more than five consecutive 1s, ensuring that it can be distinguished from the delimiter.
- It must ensure that each IC field is transmitted at the correct timing within the lpGBT frame.

Loading the Serial Control Frame

The downlink segment of the IC module is structured as an entity employing two processes, as depicted in Figure 4.2. The initial process, illustrated as "*p_load_frame*", operates by employing an internal array to systematically store 8-bit words in the appropriate sequence. The 8-bit control words for the serial frame are extracted from the data supplied to dedicated ports corresponding to the lpGBT address, memory address, and the number of payload words. By retrieving data for the overhead from independent ports, the module provides better control over ensuring that the correct data is arranged in the precise order required within the array. This approach also allows for processing these data inputs on the same clock edge, enhancing the overall efficiency of the module.

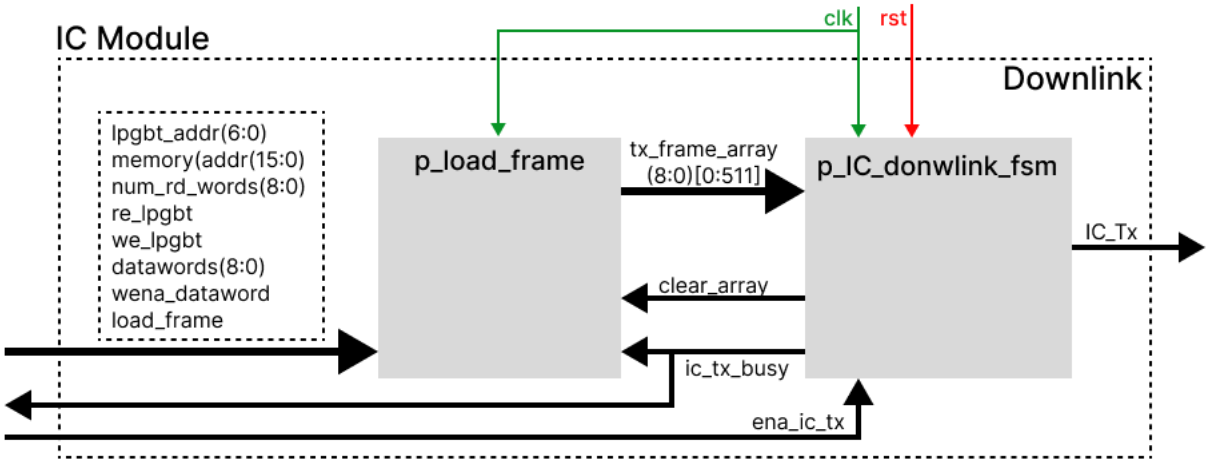


Figure 4.2: Block diagram of the downlink portion of the IC Module

In the context of the lpGBT version one, it is notable that the command word within the serial control frame is not currently considered. To address this, the module takes the initiative to automatically append this word to the array as 8'b0. This ensures that this control word is accounted for, even if it does not have an immediate role in the current version of the lpGBT. Additionally, the module maintains a port dedicated to the command word within its entity structure. This foresight allows for future updates and enhancements to the module, ensuring that it remains adaptable and ready to accommodate changes or additions related to this control word in upcoming versions or configurations of the lpGBT system.

The Read/Write bit in the serial control frame is determined based on two enable signals. To enable a serial control write operation, the **Write Enable** signal must be set to a high state, while the **Read Enable** signal must be maintained at a low state. Any other combination of these signals will result in the Read/Write bit being set to a high state, consequently enabling a serial control read operation. This design precaution ensures that registers are not unintentionally written, mitigating the risk of accidental write operations in cases where both enable signals are high or low simultaneously. When all signals for the overhead are appropriately applied to their respective ports, the activation of a **Load Frame Enable** signal to a high state initiates the process of loading these values into the array. This action prepares the frame for transmission, making it ready for a read operation of the internal lpGBT registers.

During a write operation to the registers, additional payload data must be included in the serial control frame, along with the loading the overhead. This payload data, consisting of 8-bit words, is automatically inserted into the array at the appropriate position from a designated payload data port. This insertion occurs on the rising edge of the clock signal when the **Payload Write Enable** is set to a high state. This process can be repeated up to 510 times, resulting in a total of 511 8-bit words of data that than can be written to the internal lpGBT registers. It is worth mentioning that the order in which the overhead are loaded and the payload data is written is not significant. However, it is crucial to ensure that enough payload data is loaded to match the number of words indicated by the control word, *number of payload words*, which specifies the quantity of data words to be written. In cases where there are fewer words in the array than indicated by the control word, any extra words will be written as *8'b0* to the lpGBT registers, ensuring that the operation proceeds without issues. In retrospect, this user requirement could be eliminated by introducing conditional execution for the transmission process, thereby preventing the transmission from commencing if an insufficient amount of data is loaded.

Serialization and Bit-Stuffing

Once all the necessary data for the serial control frame has been loaded into the array, the process of serial transmission can be initiated. Due to the requirement of bit-stuffing for all the data in the frame except for the start and end delimiters, the serialization process is carried out using a Final State Machine (FSM), as depicted in Figure 4.3.

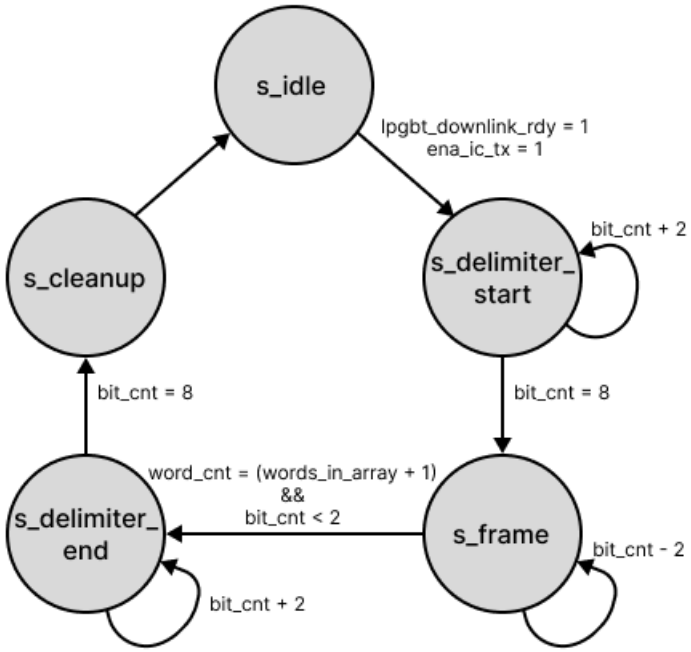


Figure 4.3: State diagram of the final state machine in the downlink portion of the IC module.

Before transmission is initiated, the state machine remains in the idle state, waiting for an enable signal to start the transmission process, all the while maintaining the IC bits at a high state. The initiation of frame transmission is triggered by the **IC TX Enable** signal, prompting several key actions. These actions include creating a copy of the array, calculating the parity word specific to this frame, which is then appended to the array copy, and preparing the first two bits of the delimiter for transmission. These preparatory steps are executed before entering the *delimiter start* state. In this state, the transmission process of the delimiter commences, once complete, the state machine transitions into the *frame* state.

During the *frame* state, the serial transmission of the control and data words within the serial control frame occurs. This process begins by concatenating the two first 8-bit words of the array within a shift register. This shift register operates by shifting out two bits during each clock period in a LSB fashion. As the first 8-bit word is nearing completion of its shift, a new 8-bit word is fetched from the array and appended to the shift register. This cycle repeats until all the words contained in the array have been successfully transmitted. The decision to append the 8-bit words

consecutively within the shift register, as opposed to completing the shift of one 8-bit word before moving to the next, is a strategic choice made to effectively handle the bit-stuffing process applied to the frame. This approach is instrumental in ensuring that bit-stuffing can be consistently applied throughout the frame.

Since data is transmitted two bits at a time, the bit stuffing is implemented by checking two five-bit fields for the occurrence of consecutive 1s. When two bits are shifted out of the frame, the first five bits in the shift register are examined. If these five bits are not all 1, the second to sixth bits of the shift register are then examined. If five consecutive 1s are detected in either of these fields, a zero is inserted into the frame immediately after the field. It is worth noting that once the first occurrence of consecutive 1s is detected and a zero is inserted, the next field is not checked. This is because the insertion of the zero already addresses the issue. In Figure 4.4, a detailed illustration of the shift register operation is presented, offering insights into the key processes. This includes the appending of 8-bit words, the two field checking for five consecutive 1s, and the insertion of 0 bits as needed. Additionally, the figure showcases the continuous serial shifting of bits, providing a visual representation of how data is managed within the shift register during the course of its operation. The bits shifted out are flipped before being added to the IC field to accommodate the MSB-first convention of the field.

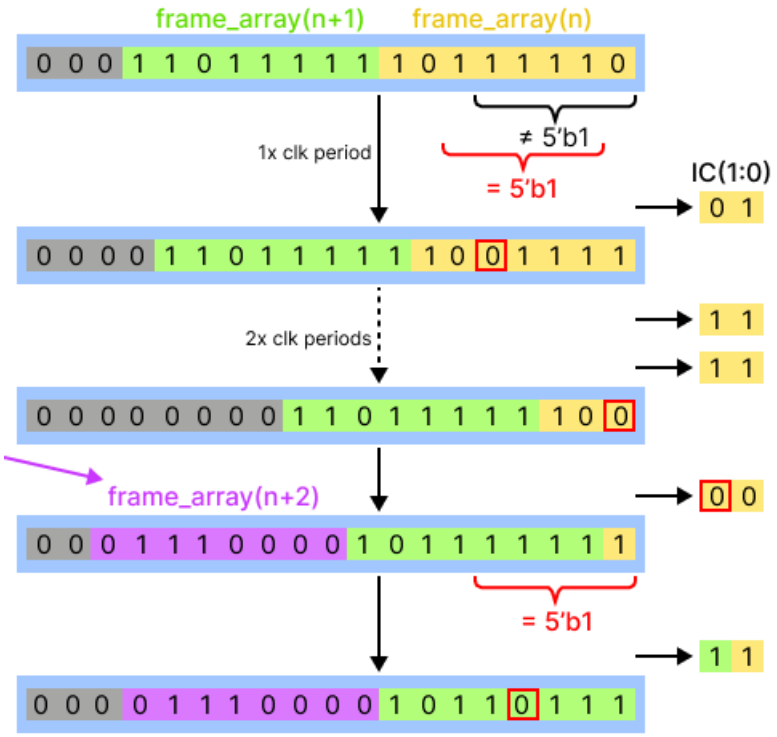


Figure 4.4: Shift register operation with word appending and bit-stuffing in the downlink portion of the IC module.

Upon the completion of transmitting all the words within the serial control frame array, the state machine progresses to the *delimiter end* state. In this state, the system sends the delimiter signifying the conclusion of the current data frame, before proceeding to the *cleanup* state. Here, internal signals and variables that played a role in the transmission process are reset to their initial states, ensuring a clean slate for the next data transmission. Finally, the state machine returns to the *idle* state, ready to receive and transmit a new data frame as it awaits the next communication task.

4.2.2 Uplink

The portion of the IC module that handles the interface with the uplink IC field is important for establishing bidirectional IC channel communication. Its primary function involves reconstructing serial control frames from the IC bits contained within the uplink lpGBT frame. This reconstructed serial control frame received on the uplink serves as a direct response to user-initiated operations executed through the downlink serial control interface. In contrast to the downlink portion of the IC module, which is responsible for data transmission, the uplink counterpart functions inversely by deserializing the incoming IC bits, processing the data, and storing it in an organized manner. This ensures that the data becomes easily accessible to the user. Outlined below are the key requirements for the successful implementation of the uplink portion of the IC module, ensuring the effective reconstruction of the uplink serial control frame.

- The prompt and precise sampling of incoming IC bits.
- The module should support bit destuffing to ensure accurate deserialization and frame reconstruction.
- The reconstructed frame must be stored efficiently and in an organized manner.

Sampling and Reconstruction

The process of reconstructing the uplink serial control frame involves a series of discrete operational steps. To manage these steps effectively, the uplink portion of the IC module is structured as an independent entity, employing a FSM to oversee and coordinate the various operations. Figure 4.5 illustrates a block diagram representing the uplink entity.

Prior to the IC bits entering the FSM, the entity employs double registers to pre-process the incoming data. This is done in order to minimize the risk of metastable values entering the circuit, which can occur due to clock domain differences. However, it is worth noting that later in the discussion, the implementation of a Clock Domain Crossing (CDC) element between the lpGBT core FPGA and the communication modules will be addressed. This element effectively transitions the signals

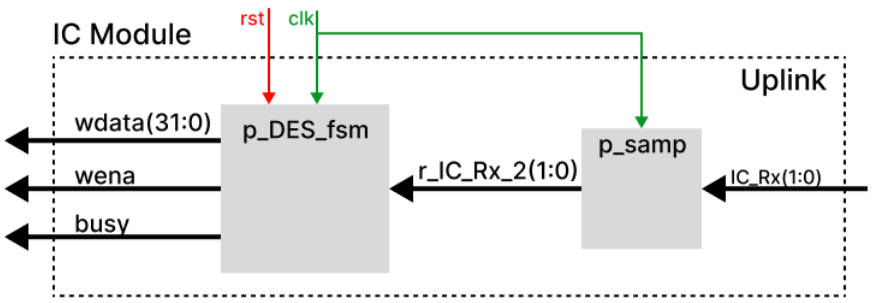


Figure 4.5: Block diagram of the uplink portion of the IC Module.

to the clock domain used by the communication modules before their entry, making the double registers redundant in this context.

To ensure the accurate initiation of the reconstruction process for the serial control frame, the FSM continuously monitors the incoming IC bits while in the *idle* state. Given that the IC bits remain in a high state between serial control frame transactions, the presence of IC(1:0) = ”01” serves as a trigger, prompting the FSM to transition to the *check* state. In this state the data stream is examined for the presence of a delimiter. If a delimiter is detected, the FSM proceeds to the *data* state, where it begins the process of reconstructing the serial control frame. However, if no delimiter is identified, the FSM performs cleanup operations and returns to the *idle* state, resuming its monitoring of the incoming IC bits. The state diagram representing the FSM used within the uplink entity is presented in Figure 4.6.

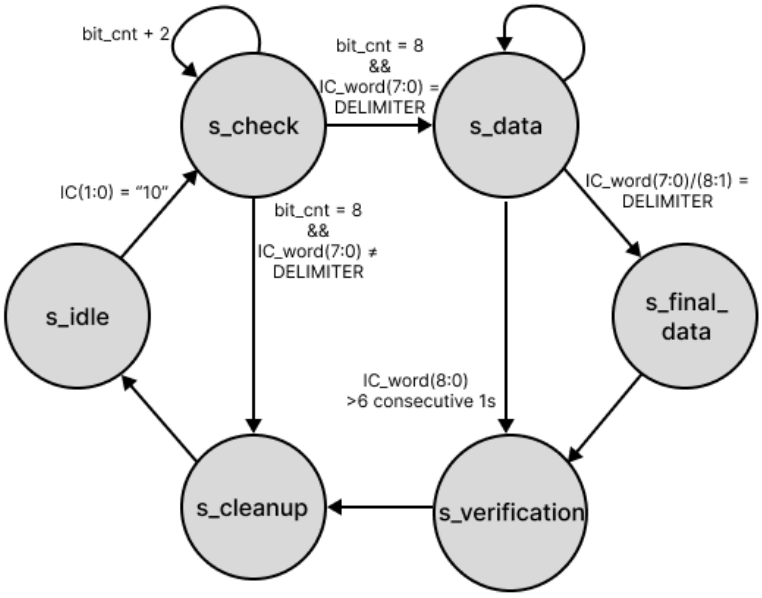


Figure 4.6: State diagram of the final state machine in the uplink portion of the IC module.

During the *data* state, a series of operations occur, including deserialization, destuffing and progressive storing. The two IC bits are deserialized onto two distinct vectors. One vector serves the purpose of word verification, while the other is utilized for destuffing operations. This approach is essential because using a single vector would make it impossible to distinguish whether a deserialized 8-bit word serves as a delimiter or merely represents destuffed data. The vector employed for destuffing operations is vulnerable to the introduction of asymmetry, because two bits are added during each deserialization step, while on bit is removed during destuffing. Given this it is essential for the system to keep track of both the number of bits added and removed. This tracking allows the process to precisely determine which eight bits within the vector correspond to a specific 8-bit word in the serial control frame protocol. Once the process has determined that the vector containing destuffed data has received a minimum of eight new bits, it retrieves an 8-bit word from this vector, taking into account whether destuffing introduced any asymmetry. This retrieved word is then temporarily stored, ensuring precise data alignment and that the integrity of the serial control frame protocol is maintained. Figure 4.7 illustrates the asymmetry introduced to the destuffing vector.

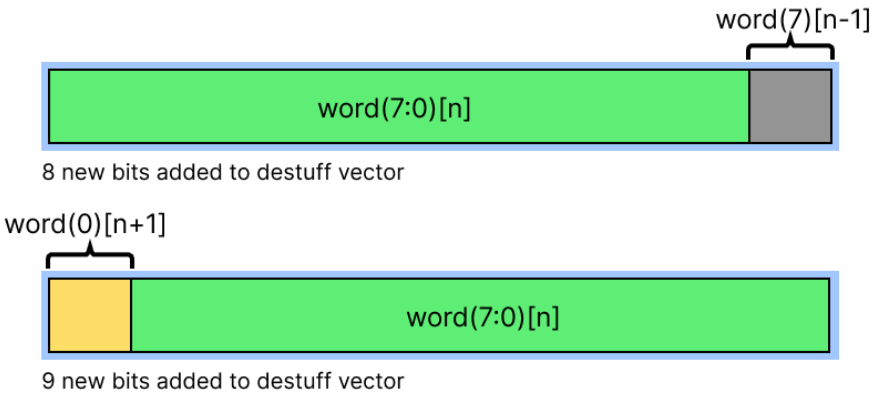


Figure 4.7: Illustration of asymmetry in the destuffing vector.

Computing the Parity Word

In addition to the FEC decoding carried out by the versatile link to ensure error-free transmission of the lpGBT frame, an extra layer of error-checking is provided for the serial control frame through the inclusion of a parity word. The transmitter calculates this even parity word and adds it to the frame. Upon reception of the frame, the receiver independently computes its own parity word for the received frame and compares it to the transmitted parity word to check for any discrepancies. The calculations of this parity word is a stepwise process undertaken by the uplink entity. It entails progressive Exclusive OR (XOR) operations between each of the temporarily stored 8-bit words and the evolving parity word. The parity word is updated each time a new 8-bit word is

deserialized. Initially, during the commencement of the frame, the parity word is set to all zeros, forming a baseline for the first XOR operation within the frame. This process results in a parity word where each bit corresponds to the parity of all the bits in the same position across all the 8-bit words in the frame, with the exclusion of the delimiters. Figure 5.2 in Section 5.1.2 provides a visual representation of a waveguide segment extracted from simulation, offering insight into the process of calculating the parity words.

Storing the Serial Control Frame

During the *data* state, a concurrent operation involves assembling the temporarily stored 8-bit words sequentially to from 32-bit words. Once four new 8-bit words have been successfully deserialized and processed, a complete 32-bit word is written to a First-In-First-Out (FIFO) buffer for storage. This approach is adopted to align with the 32-bit payload transmission frame used in the IPbus protocol, which is utilized for user communication with the module. The details of IPbus operation and implementation will be discussed in subsequent sections. Table 4.1 provides an example of how the 8-bit words are organized within the 32-bit words written to the FIFO, in accordance with the serial control frame protocol.

32-bit words	**(31:24)**	**(23:16)**	**(15:8)**	**(7:0)**
1st word	7'b0 & payload_size(8)	payload_size(7:0)	command(7:0)	lpgbt_addr(6:0) & rw_bit
2nd word	data(7:0)	data(7:0)	memory_addr(15:8)	memory_addr(7:0)
3rd word	8'b0	8'b0	parity_word(7:0)	data(7:0)
4th word	8'b0	8'b0	8'b0	6'b0 & corrupt_frame & parity_check

Table 4.1: Example of organization of 8-bit words within the 32-bit words written to the uplink FIFO.

The FSM undergoes a transition out of the *data* state under two conditions: either upon recognizing a delimiter or upon detecting a data stream with more than six consecutive 1s. In the latter scenario the FSM transitions into the *verification* state, before executing cleanup procedures and reverting to the *idle* state to resume monitoring. Conversely, when a delimiter is recognized, the FSM transitions into the *final data* state. In this state, the process considers how many new 8-bit words have been received since the last FIFO write operation and writes a 32-bit word into the FIFO accordingly. It ensures that any fields without new words are filled with zeros to maintain the integrity of the data structure within the FIFO.

After the *final data* state, the FSM progresses into the *verification* state, designed to offer users supplementary information regarding the transmission of the serial control frame. This state plays a role in verifying the correctness of the frame by examining the parity word and addressing the possibility of aborted receptions. In the *verification* state, an additional 32-bit word is written to the FIFO buffer, where only the two LSBs are actively used, while the remaining bits are padded with zeros. When the LSB of this word is set to a high state, it serves as an indicator that the reception of the frame was terminated due to the identification of a data stream containing more than six consecutive 1s. The bit following the LSB serves as a marker to check whether the parity word computed by the uplink entity matches the one received in the serial control frame, signified by this bit being set to a high state. Upon writing this final word to the FIFO, the FSM initiates cleanup procedures before transitioning back to the *idle* state.

As an additional point to consider, given that the uplink serial control frame directly corresponds to operations executed through the downlink serial control interface, the quantity of received data words and their subsequent storage in the FIFO can typically be readily determined under normal operations. However, to further enhance the overall usability and reliability of the module, it would be advantageous to implement a mechanism that effectively segregates individual serial control frames. This separation mechanism would serve the purpose of facilitating the clear distinction between different frames, especially when multiple frames are stored in the FIFO while prior frames await readout. Such an organizational feature contributes significantly to a more structured and manageable system. Moreover, this separation mechanism would prove particularly useful in scenarios involving terminated transmission, where the quantity of data words in the FIFO may become indeterminate.

4.3 The Payload Module

In contrast to the Serial Control Module, the module responsible for controlling the payload data field within the lpGBT frame from the lpGBT core FPGA interface exhibits a notably simpler structural design. This simplicity is due to the absence of complex protocols and bit-stuffing requirements at this particular layer. However, it is important to emphasize that this module primarily serves as a straightforward testing tool, offering a platform for loading payload data onto the downlink lpGBT frame for transmission and offloading received payload data from the uplink lpGBT frame. Processing payload data in accordance with specific data transfer formats for a fully operational system will introduce a significantly higher degree of complexity.

4.3.1 Facilitating Data Loading and Offloading

The payload module features simply two FIFO buffers tailored to the specific requirements of both the uplink and downlink payload fields. In the case of the downlink payload, the FIFO width is set at 32 bits. However, for the uplink payload, configured to accommodate lpGBT uplink operations at 10.24 Gbps with FEC12 encoding, the width is extended to 192 bits. Figure 4.8 illustrates a block diagram representing the payload module.

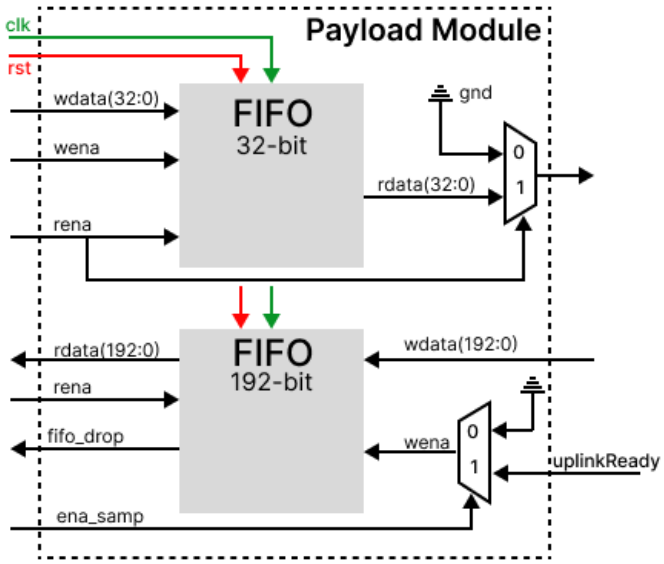


Figure 4.8: Block diagram of the payload module.

The downlink operation of the payload module involves a straightforward process: data is written into the FIFO by the user, and transmission of this data via the downlink is initiated by setting the read enable signal to a high state. While the read enable signal remains high, the module efficiently loads data onto the lpGBT frame during each clock period. For enhanced transparency during testing, a Multiplexer (MUX) is integrated between the payload module and the subsequent component. The primary function of this MUX is to pull the payload data low when the read enable signal transitions to a low state. This mechanism streamlines the recognition of data flow within the system, contributing to effective testing and monitoring processes.

In the uplink operation of the payload module, the FIFO is constantly updated with data retrieved from the lpGBT frame via the lpGBT core FPGA interface. The trigger for writing data into the FIFO is a signal indicating that the FPGA has received a new frame, acting as the write enable signal. This approach ensures that the FIFO is updated with fresh data as soon as new lpGBT frames are received. The FIFO receives this write trigger at a frequency of 40MHz, which means that every 25 nanoseconds, an lpGBT frame containing 192 bits of payload data is written to the

FIFO. This results in a rate of 40 million frames being processed every second.

To streamline testing and manage the substantial influx of data, various measures should be implemented. A MUX is suggested to control the writing of data into the FIFO. This feature allows users to enable or disable data input, providing time to process the data already residing in the FIFO. Additionally, a drop mechanism within the FIFO to address scenarios where the FIFO approaches full capacity while data remains unread is beneficial. In such cases, during each clock period, both the read and the write index within the FIFO are incremented when new data is written. This strategy prevents the FIFO from reaching its maximum capacity, ensuring uninterrupted data flow without the risk of overflow. To facilitate thorough testing of uplink payload data, it becomes essential to implement specialized user logic that operates within the existing system. This could involve the creation of dedicated modules such as a data checker, designed to verify that the received frame aligns with the one originally sent. Ensuring accurate synchronization between the transmitted and received frames is crucial, as it enables the checker to precisely correlate received data with the corresponding transmitted data.

4.3.2 Utilizing E-port Frame Allocation for Testing

Managing the substantial volume of data received on the uplink requires thoughtful design strategies to streamline testing processes. Leveraging the correlation between e-Groups on the lpGBT ASIC and frame allocation, coupled with the configurability of active e-Ports within each group, allows the system to selectively target and process specific segments of the frame individually. This approach simplifies the testing and processing by breaking down the complex task into more structured and manageable components, ensuring a more efficient and effective testing environment.

As detailed in section 3.5, **lpGBT**, payload data field within the lpGBT frame is organized into segments of either 16 or 32 bits, depending on the uplink data rate configuration. Each of these segments corresponds to an e-Group on the lpGBT ASIC. The number of e-Group segments allocated in the lpGBT frame is directly tied to the FEC encoding configuration of the uplink. For instance, in a 10.24 Gbps uplink configuration, each segment consists of 32 bits, and with FEC12 encoding, the frame accommodates six such segments, thereby utilizing six e-Groups. Additionally, the e-Groups offer flexibility in terms of configuring how many of the four e-Ports within each group are active.

In the case of the 10.24 Gbps configuration, a bandwidth of 1280 Mbps is allocated among the active e-Ports within the group. Consequently, the frame allocation for each port is divided among the 32 bits of the e-Group field in the lpGBT frame. When an e-Group is configured to utilize

only one e-Port, all 32 bits in the field directly correlate with the data provided through that specific e-Port. Conversely, if all four e-Ports within a group are configured to be active, the 32-bit field is distributed evenly among these four e-Ports, resulting in 8 bits of that particular field corresponding to the data provided through one specific e-Port. This configurability and segmentation of data enable flexibility in handling and processing data within the lpGBT frame. Figure 4.9 provides an illustrative representation of the lpGBT frame allocation for e-Ports with varying e-Group configurations.

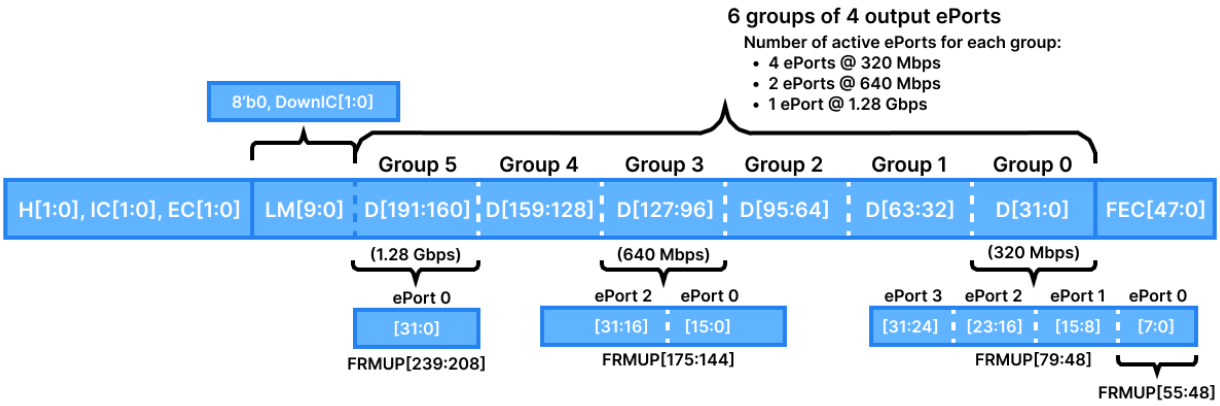


Figure 4.9: Uplink frame structure before interleaving, with lpGBT frame allocation for e-Ports with varying e-Group configurations.

By configuring each e-Group on the lpGBT ASIC to utilize four e-Ports, the testing for the VL+ uplink channel can be broken down into distinct segments, with each element focusing on 8 bits of the 192-bit payload field. This arrangement allows for the pairing of a front-end element with a corresponding back-end element. Effectively managing data flow on the e-Link connected to a specific e-Port in the front-end and processing the 8 bits in the frame that corresponds to that particular port in the back end. This pairing mechanism can be implemented for all e-Ports within each e-Group, simplifying the verification process for the uplink data transmission through every e-Port on the lpGBT ASIC. It enables comprehensive and independent testing of each individual port, while also providing a more straightforward foundation for debugging in instances where issues may be related to a specific e-Port.

The testing approach for the VL+ downlink channel can also be structured into separate segments. The configuration of the downlink e-Groups allows for the correlation of two, four, or eight bits of the downlink lpGBT frame with individual e-Ports. However, it is important to note that unlike the IC fields, the uplink and downlink channels, in the context of the payload fields, lack direct correlation and can operate independently. Given this, it is advantageous to prioritize the development of testing elements for the uplink payload fields. This focus is driven by the recognition that trans-

mitting data from the front-end to the back-end constitutes the most critical utilization of the VL+ ecosystem within the context of the FoCal Pixel RU. Emphasizing uplink payload transmission testing ensures that this crucial data transfer process is thoroughly evaluated and validated.

4.4 lpGBT Core FPGA Interface

The main design requirement for the test suit is to establish interfaces with the lpGBT core FPGA and the electrical links from the lpGBT ASIC. Designing an interface with a component involves several key considerations to ensure that the connection and communication between the component and the interface are reliable, safe and efficient. The main steps and requirements involved in designing an interface include understanding the component specifications, understanding the communication protocol, testing and verification, scalability and future compatibility.

In the context of the lpGBT core FPGA interface, ensuring synchronization with the lpGBT core is also a critical factor. While the communication modules operate on the LHC clock, which runs at 40 MHz, the lpGBT core FPGA is designed to function with the MGT clock. This MGT clock is derived from a reference clock that matches the upstream line-rate, operating at 320MHz. This architectural configuration employs a multicycle path architecture with a 1 to 8 ratio. In practical terms, it means that the lpGBT core FPGA carries out eight clock cycles for every cycle of the LHC clock. To facilitate a seamless transition of data between the communication modules and the lpGBT core FPGA, a mechanism know as CDC becomes essential. The CDC element plays an important role in enabling data transfer between the 40 MHz and the 320 MHz clock domains. It ensures that data changes occur on the interconnections between these domains on every rising edge of the 40 MHz clock while guaranteeing that the data crossing over in either direction aligns with the respective clock edge. A practical example of the CDC implementation tailored for transitioning between 40 MHz and 320 MHz clock domains are provided, this example can be directly applied for testing purposes.

Figure 4.10 provides a visual representation of the interconnections for the critical uplink signals originating from the lpGBT FPGA core, passing through the CDC, and ultimately reaching the various communication modules. The transmission of the uplink lpGBT frame is depicted as a 234-bit vector. This vector accommodates different uplink configurations, ensuring flexibility. Within this vector, specific bits are allocated to each control module based on their requirements. In the case of a 10.24 Gbps and FEC12 configuration, bits 233 to 206 are padded with zeros, while bits 201 to 192 are reserved for the Latency Measurement (LM) field.

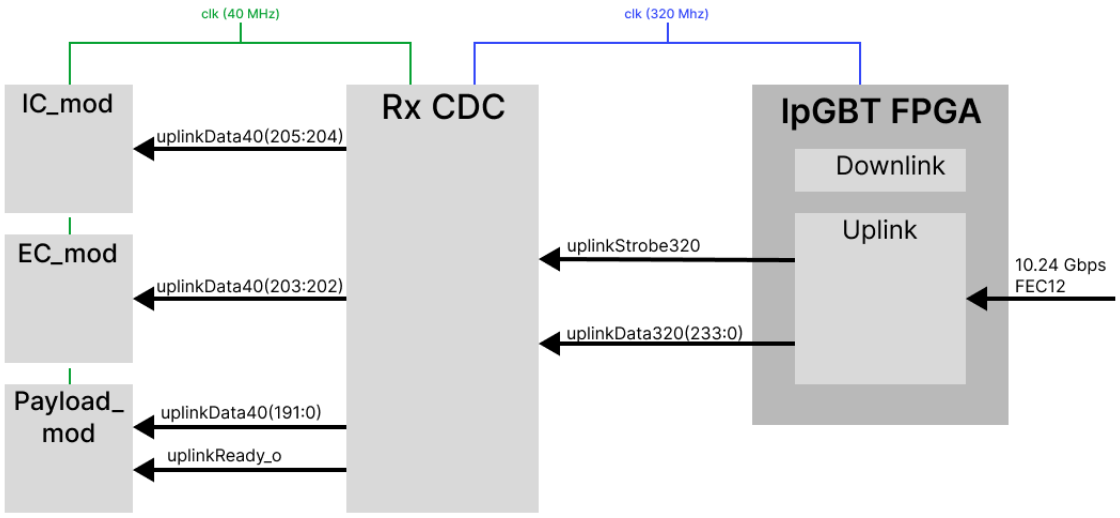


Figure 4.10: Block diagram of critical uplink signal paths from lpGBT FPGA Core through CDC to various modules.

Figure 4.11 illustrates a timing diagram that demonstrates the transition of a frame from the 320 MHz clock domain in the lpGBT FPGA core to components operating of the 40 MHz clock domain through the CDC element. In this diagram, "*data_a*" represent the frame that is accumulated within the lpGBT FPGA core, originating from the uplink serial stream. This data is the initial input to the CDC process. "*data_a_reg*" represents the frame that the CDC captures from "*data_a*" once it has accumulated a complete frame. Subsequently, "*data_b*" represents the data that the CDC passes on to the components operating within the 40 MHz clock domain. This transformed data is what becomes available for processing in the new clock domain. Lastly, "*strobe_a*" serves as an important signal sent from the lpGBT FPGA core to the CDC. It acts as a trigger, signaling to the CDC that "*data_a*" contains a new complete frame ready to be captured.

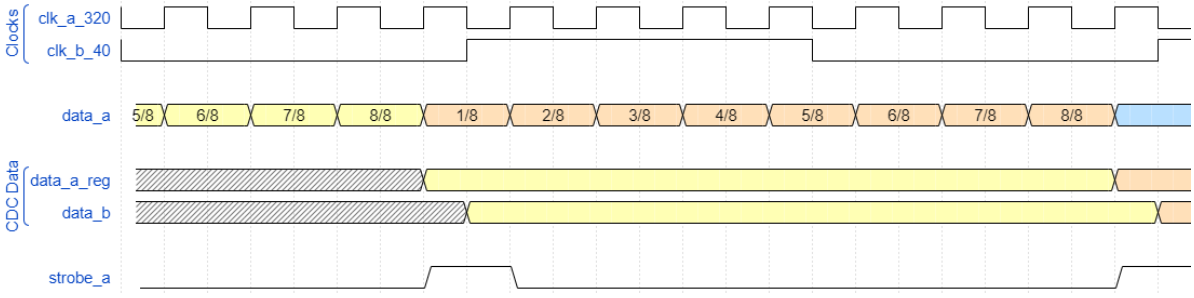


Figure 4.11: Timing diagram of frame transition from 320 MHz to 40 MHz clock domain via CDC element.

4.5 Establishing IPbus Communication with the modules

The modules discussed earlier serve as crucial interfaces for the lpGBT FPGA core, facilitating communication with the VL+ ecosystem across various user fields of the lpGBT frame. Equally significant is the ability to communicate with these modules, ensuring that users can effectively monitor and control various aspects of the system operation. This communication is facilitated by the implementation of the IPbus protocol[32], a protocol developed to meet specific communication and control requirements of experiments conducted at particle accelerators and other scientific research facilities. It offers the capability for remote configuration, monitoring, and control of hardware components, such as data acquisition systems, FPGAs, and other instrumentation devices. Additionally, IPbus is versatile across a variety of hardware platforms and scales for complex setups with numerous components. Its ability to scale effectively is complemented by the inclusion of a hierarchical addressing system, which streamlines the organization and management of these diverse components.

The IPbus suite integrates software and firmware components to establish a reliable and high-performance control link, utilizing standard internet protocol, with User Datagram Protocol (UDP)[33] as a primary communication method. Comprising three core elements, it encompasses IPbus firmware, responsible for implementing the protocol within end-user hardware, μHal, which serves as an intermediary control software interface to communicate with the underlying firmware, and the ControlHub, functioning as a centralized access point for multiple μHal entities on the software side.

4.5.1 IPbus Firmware

An IPbus firmware repository[34] provides a reference for system-on-chip implementation of modules designed to interpret the IPbus protocol within FPGAs. These modules encompass interfaces for both receiving and transmitting IPbus transactions via Gigabit Ethernet and PCIe connections. Among the key components is the transactor, responsible for decoding transactions and serving as the master of the on-chip bus. Additionally, a bus fabric, coupled with address decoding capabilities, allows for the attachment of multiple slaves to the bus, enhancing system scalability. To integrate this firmware with the test system modules, the key requirement is the design and implementation of slaves that can effectively translate transactions between the IPbus protocol and the respective modules.

The slaves are components designed to interact with the IPbus master and interpret IPbus transactions. Much like the serial control communication utilized for the lpGBT, data sent from the slave to the master is a direct response to user-initiated operations initiated by data sent from the

master to the slave. For the master to facilitate communication with multiple slaves, a set of distinct signals is utilized, including a 32-bit address signal (*ipb_addr*), a 32-bit payload data signal (*ipb_wdata*), a write enable signal (*ipb_write*), and a strobe signal (*ipb_strobe*). All signals, except for the strobe signal, are broadcast to all the connected slaves. The strobe signal is selectively asserted by the fabric for the specific slave to which the data is addressed. This strobe signal serves as a marker, indicating the initiation of a new transactions directed at the designated slave. When the strobe signal is low, the slave recognizes that the data on the bus is not intended for it and disregards it. In response to new transactions initiated by the master, the signals transmitted from the slave to the master consists of a 32-bit payload data signal (*ipb_rdata*), an acknowledge signal to inform the master that the data has been received (*ipb_ack*), and an error signal to indicate to the master that an issue occurred during data reception (*ipb_err*).

4.5.2 IPbus Slave for the IC Module

To integrate the IPbus firmware into the test system, the implementation of dedicated IPbus slaves, which act as intermediaries connecting the master device with specific modules is needed. In the case of the IC module, the IPbus slave is identified as the "*IC slave*", implemented as a register block. This register block serves as a bridge between the master and the IC module, enabling indirect control of the module via the IPbus protocol. Both the IC module and the register block are implemented together as a part of a top-level entity known as "*IC Module Top*". This entity also includes two 32-bit FIFOs, one for storing payload data words intended for downlink transmission and another for storing serial control frames received by the IC module on the uplink. Figure 4.12 illustrates a block diagram depicting the structure and connections of this top entity.

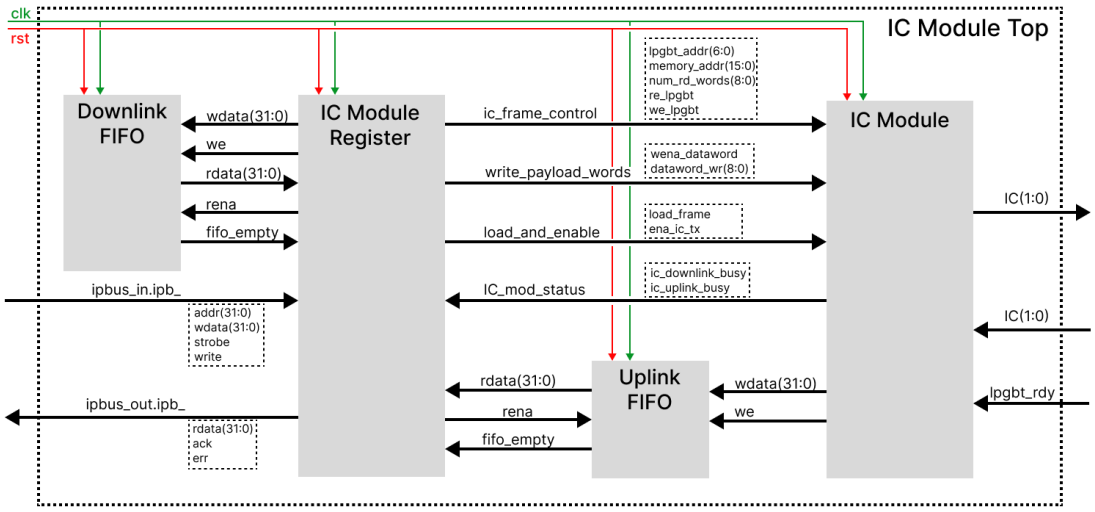


Figure 4.12: Block Diagram of the IC module top entity.

The IPbus address signal are important for system communication, distinguishing between slave addresses and the internal registers within those slaves. A register map for the IC slave is provided in Table 4.2. The address used to engage this specific slave is 0x01, this slave address is retrieved from *ipb_addr(15:8)*. Meanwhile, *ipb_addr(7:0)* is employed to access the internal registers of the slave. Much like the internal registers in the lpGBT, the registers of the IC slave exhibit specific access types. Some registers are categorized as Read/Write (RW), enabling users to both write and subsequently read the values stored within them. Others are exclusively designated as Write-Only (W) or Read-Only (R), restricting them to either writing or reading operations. Additionally, there are Write Trigger (WT) registers, which do not exchange data but instead trigger specific operations within the slave.

Name	Register	Access	Description
IC_SLAVE (slave address 0x01)			
STATUS	0x00	R	[6:6] Uplink FIFO empty [4:4] IC uplink busy [2:2] Downlink FIFO empty [0:0] IC downlink busy
DATA_TX	0x01	W	[31:24] Dataword 4 [23:16] Dataword 3 [15:8] Dataword 2 [7:0] Dataword 1
DATA_RX	0x02	R	[31:24] Dataword 4 [23:16] Dataword 3 [15:8] Dataword 2 [7:0] Dataword 1
FRAME_ADDR	0x03	RW	[22:7] Memory address [6:0] lpGBT address
FRAME_CTRL	0x04	RW	[17:10] Command [9:9] Read/Write bit [8:0] Payload size
LOAD_AND_EX	0x05	WT	Loads datawords from FRAME registers to the downlink frame. Will also load n(payload size) words from the Uplink FIFO for lpGBT write operations (Read/Write bit = 0).
RESET	0x06	WT	Resets

Table 4.2: Register map for IPbus IC slave.

Within the IC slave, the Read/Write registers are employed for storing lpGBT frame control words, offering users flexibility to write values and later verify the content stored within them. There exists one registers that is write-only. This particular register is responsible for receiving payload data words to be transmitted within the serial control frame during a write operation. This restriction is

in place because these words are written to a FIFO for storage, ensuring that the 8-bit words are loaded into the IC module in the correct order. Conversely, there are two registers that are read-only. These registers include one that stores status values originating from components within the IC module system and another register used to read the uplink payload data FIFO. Lastly, two registers possess WT access, responsible for initiating specific actions. These actions include initiating a system reset, or loading a serial control frame into the IC module and subsequently triggering its transmission.

IC Slave Operation

Before the IC Module can initiate any operation, it must be presented with all the required data for the serial control frame. Consequently, the initial user task involves writing this data to the IC slave. In the IPbus protocol, data transfers are structured into transactions with a payload size of 32 bits. To optimize the efficiency of providing the complete data needed for a serial control frame transmission, specific segments of the serial control frame are logically grouped within the same register transactions to the IC slave. The organization of data for these transactions are referenced in Table 4.2, with the design seeking a balanced compromise between efficiency and comprehensibility.

The data associated with the addresses within the serial control frame are consolidated into the **FRAME_ADDR** register transaction. This design choice serves to optimize the communication process by reducing the number of IPbus transactions required when performing a read operation on the same lpGBT register after a previous write operation. The remaining overhead for the frame are grouped within the **FRAME_CTRL** register transaction. For IC read operations, executing these two IPbus transactions is sufficient before initiating serial control frame transmission. However, in the case of an IC write operation, additional steps are necessary as payload data must also be loaded into the IC module. To fully utilize the IPbus data transfer capacity, up to four 8-bit payload data words are consolidated into the **DATA_TX** register transaction, which is loaded to the downlink FIFO by the slave. Given that a single serial control frame can accommodate up to 511 8-bit payload words, the FIFO can receive a total of 128 32-bit words before being full. The specific order in which these 8-bit words are organized within the serial control frame is detailed in the description field of Table 4.2.

Once all the necessary data for a serial control frame has been loaded into the respective registers of the IC slave, IC module operations can be initiated. This initiation is carried out by performing a **LOAD_AND_EXECUTE** register transaction. When this WT register is accessed, it triggers a FSM to transition, moving from the *idle* state to the *operation* state, provided that the downlink portion of the IC module is not currently busy. A visual representation of the state transitions in the

FSM can be found in Figure 4.13. In the *operation* state, the FSM determines the type of operation to be performed. If the Read/Write bit that was written to the slave in the last **FRAME_CTRL** register transaction is in a high state, the FSM proceeds directly to the *load frame* state. Otherwise, it advances to the *write 8b words* state.

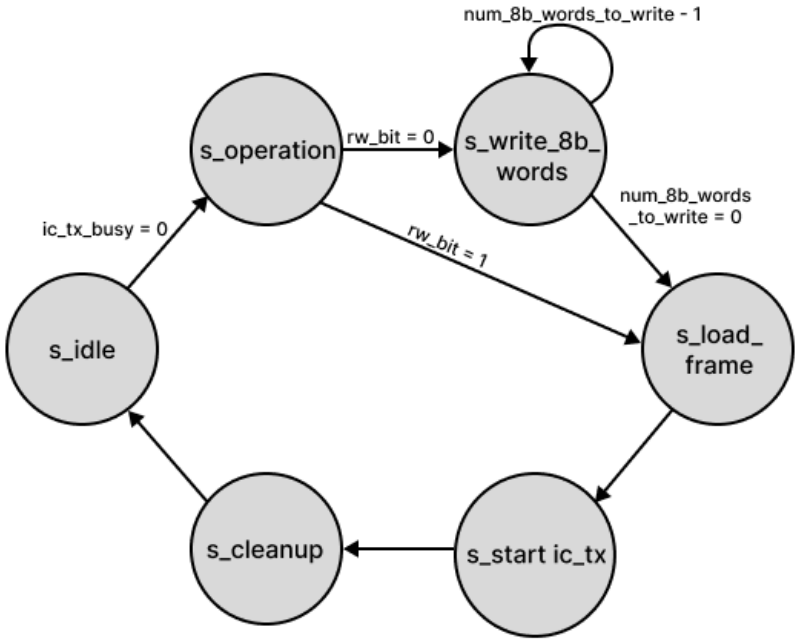


Figure 4.13: State diagram of the final state machine in the IPbus IC slave for LOAD_AND_EXECUTE transaction.

In the *write 8b words* state, the FSM retrieves 32-bit data words from the downlink FIFO. From these 32-bit words, it extracts 8-bit words starting from the LSB and loads them into the IC module. The number of words loaded into the IC module is defined by the payload size written to the slave in the last **FRAME_CTRL** register transaction. If four 8-bit words have been extracted from the same 32-bit word, and there are more 8-bit words to be loaded, a new 32-bit word is fetched from the FIFO. This process continues until the number of words loaded into the IC module matches the quantity defined by the user, at which point the FSM proceeds to the *load frame* state.

In the *load frame* state, the FSM delivers all the necessary control words required for the serial control frame to the IC module and activates the IC module load enable signal. Following this, the FSM proceeds to the *start IC transmission* state, where it deactivates the load enable signal, activates the IC transmit enable signal, and moves on to the final *cleanup* state. The FSM deactivates all enable signals, resets internal signals and variables, and returns to the *idle* state. The automation provided by the implementation of this FSM enhances user-friendliness, making it easier and more efficient to work with the IC module for serial control frame transmission.

4.6 Pixel Data Transfer Format

The data transfer formats for the pixel layers closely resemble the ITS2 data transfer format, with the primary distinction lying in the front-end to back-end transmission. In the case of ITS2, GBTx chips are employed to manage this transmission, where each frame transmitted consists of a GBT word structured from 80 bits of ALPIDE payload data. In the back-end, the CRU facilitates the transmission of this data to the FLPs, achieving a parallel transfer of 128 bits of data in each data transfer cycle. The frame received by the CRU is smaller than the frame subsequently sent to the FLPs, allowing the CRU to efficiently pack the GBT words, enhancing overall transmission efficiency as it waits for data to fill up the 128-bit words.

However, when the CRU receives data from lpGBT chips, each lpGBT word necessitates one and a half transfer cycles to be transmitted from the CRU to the FLP within the 128-bit words, as illustrated in Figure 4.14. This difference in transfer configuration means that achieving further transmission efficiency improvements at this stage of the readout path is no longer possible.

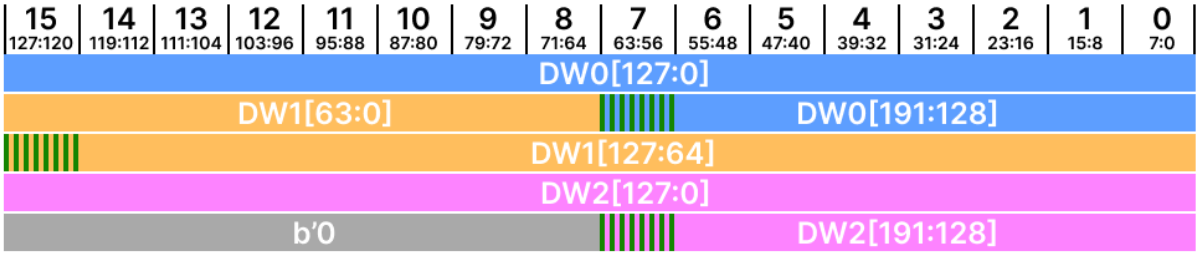


Figure 4.14: Pixel Data Format

5 lpGBT Test-Suite Verification and Testing

This chapter provides details of the efforts made to ensure the proper operation of the IC module and its dedicated IPbus slave. It outlines measures taken to verify functional correctness, adherence to specifications, and the identification and mitigation of potential design flaws. Tools and methodologies used to enhance these efforts are also introduced.

FPGAs are incredibly versatile devices known for their flexibility and adaptability, allowing designers to implement complex digital circuits for countless applications. However, these very attributes introduces an inherent level of complexity and the potential for errors and defects. To mitigate risks and ensure that an FPGA-based system performs as intended, it is essential to implement rigorous verification and testing methodologies.

Verification primarily concentrates on confirming the correctness and functionality of the FPGA design during the early phase of development through simulation and virtual environments. This process ensures that the design operates according to its specifications before any physical implementation. On the other hand, testing focuses on evaluating the actual hardware implementation of the FPGA in its intended real-world environment to verify its practical functionality and performance. Both verification and testing plays crucial roles in delivering FPGA-based solutions that are reliable and capable of achieving the specific requirements.

5.1 Verification

The process of verification is a critical step in ensuring the reliability and functionality of FPGA designs. It involves a comprehensive assessment of the design at various levels, ranging from individual modules to complete systems. This evaluation aims to confirm the accuracy, functional integrity and adherence to design specifications as the system evolves throughout its development cycle. A fundamental aspect of effective FPGA verification is the creation and utilization of testbenches. These testbenches serve as simulated environment that mimic the behavior of the FPGA design under diverse conditions and inputs. Developing these testbenches concurrently with the design phase is essential, as it facilitates the early detection and resolution of issues in the design. Additionally, it enables regression testing, which involves systematically retesting a previously verified system after incorporating changes. This iterative process helps ensure that any modifications introduced do not unintentionally introduce new problems.

To maintain thorough and up-to-date verification, it is essential to continuously improve and expand testbenches as the design evolves. When these testbenches are integrated with verification tools and methodologies like Unviersal VHDL Verification Methodology (UVVM), the process of enhancing and refining the testbench becomes notably more efficient and straightforward. UVVM introduces a standardized framework for crafting testbenches in VHSIC Hardware Description Language (VHDL), delivering advantages such as enhanced readability, maintainability, scalability, and reusability. By integrating testbench development with the capabilities of UVVM, developers can achieve a streamlined and highly effective verification process, ultimately ensuring the reliability and functionality of evolving designs.

5.1.1 Universal VHDL Verification Methodology

UVVM is a robust verification framework specifically designed for VHDL-based projects. This framework promotes the creation of modular and reusable testbench components, streamlining the process of adapting and reusing verification setups across different projects. Users have the flexibility to design their own testbench components, but UVVM also offers a wide array of built-in utilities and methodologies that can be easily integrated into the design. These utilities, available in the library[35], can be customized to suit the specific requirements of each design. By embracing this modular approach, UVVM provides a structured and scalable solution to testbench development. An approach that significantly reduces the time and effort needed to establish comprehensive test environments.

Among other advanced verification techniques, the UVVM framework offers notable utilities that significantly enhance the verification process. One such utility optimizes the implementation of transaction-based testing, allowing designers to efficiently validate interactions within the FPGA design. Additionally, the framework provides support for constrained random testing, enabling automatic generation of diverse test scenarios to uncover corner-case bugs, enhancing the overall test coverage. Furthermore, UVVM empowers designers to construct self-checking testbenches that autonomously verify the correctness of FPGA responses, reducing the reliance on manual inspection. The construction of these testbenches are facilitated by the inclusion of built-in debug utilities that simplify the identification and resolution of issues during the verification process.

The frameworks final noteworthy feature is its significant emphasis on the effective utilization of logging and alerts. These elements play an important role in enhancing the overall comprehensibility and clarity of the testbench. Logs serve as a continuous source of information during the simulation process, offering insights into the current stage of the test sequence, the ongoing tests, and those that have already been executed. Moreover, the framework incorporates a verbosity control mechanism, granting users the flexibility to manage the level of detail in the logs. This feature

allows users to deactivate redundant logs, ensuring that the generated log messages remain relevant and concise. By identifying and categorizing log messages, users gain granular control over the detail recorded, enabling a more efficient and tailored approach to monitoring and debugging during the simulation process.

The logging system also includes alerts to promptly inform users of any issues encountered during simulation. These alerts should contain essential details, including the specific test being conducted, the expected outcome, and the actual result. Users can configure the framework to set the threshold for the number of alerts that trigger a halt in the simulation, granting control over handling unexpected situations. This approach to logging and alerts ensures effective management of complex simulations.

5.1.2 IC Module Testbench

The verification process commenced in the early stages of the IC module design, starting with the creation of an independent testbench for the downlink portion. This testbench is developed in tandem with the design of the module to ensure a systematic verification approach. Initially, the primary focus is on validating the 2-bit serial shift transmission, confirming that the base for the serial control frame is transmitted as expected and in the correct sequence. Once this initial verification is successfully completed, the testbench is expanded to encompass the uplink portion. In this expanded setup, the serial stream transmitted from the downlink portion is connected to the serial port of the uplink portion.

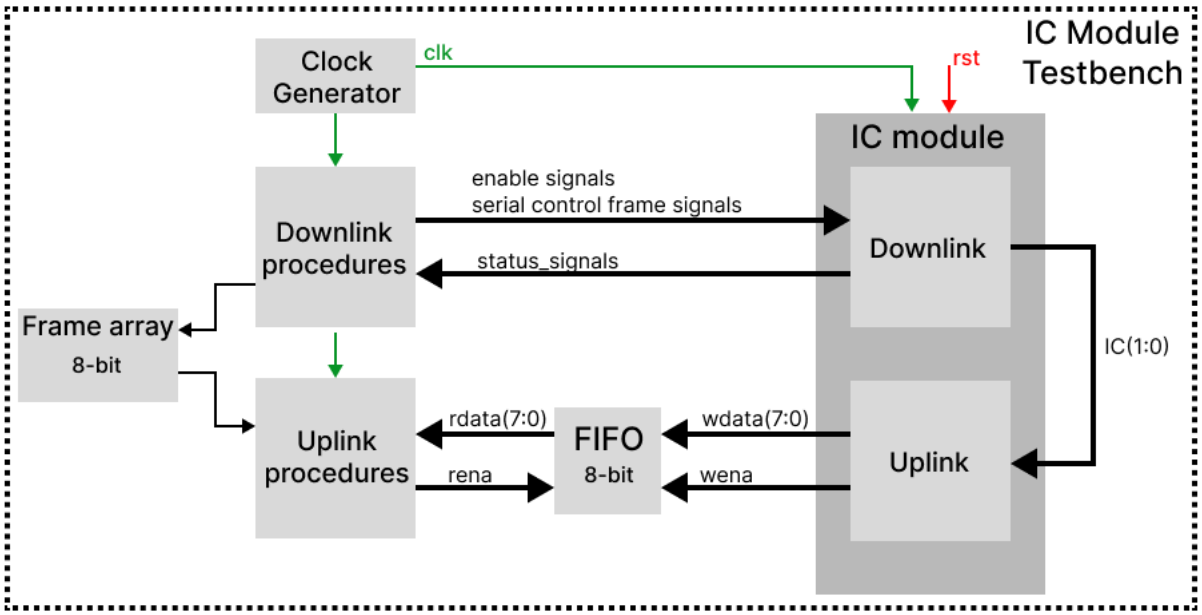


Figure 5.1: Block diagram of the testbench setup for the IC Module.

This integration is done to confirm that the data applied by the testbench to the downlink portion and subsequently serial shifted out, perfectly matches the reconstructed data in the uplink portion, confirming functional alignment between the two elements of the IC module. Connecting the serial stream between the uplink and downlink portions is sufficient for this verification phase, given that a CDC mechanism will be implemented to manage serial streams both entering and exiting the IC module. By simulating the operation of both portions with the same 40 MHz clock, the behaviour of the module is the same as when integrated into the full system with a CDC element present. Figure 5.1 illustrates a block diagram depicting the testbench setup designed for the IC module.

During the construction of the testbench for the initial verification phase, a systematic approach is taken to streamline the process and facilitate future enhancements. Specifically, a set of general procedures is developed with a focus on creating a self-checking testbench, thereby enabling efficient regression testing. One key procedure involves automating the writing of payload data to the downlink portion of the IC module. Users are only required to provide the payload data and specify the number of 8-bit words it consists of. The procedure then takes care of writing the data according to the specifications of the IC module, while simultaneously storing these 8-bit words within the testbench for subsequent verification. Another procedure is designed to load control words for the frame and initiate the transmission of either a write or read operation frame, this data is also stored within the testbench. Lastly, a procedure is implemented to read 8-bit words from the uplink FIFO until it is empty, while concurrently logging the data read. It also compares the read data to the data stored by the other procedures responsible for constructing the downlink serial control frame, enabling verification of the IC module functionality.

Verification Related to the Parity Word

Once the fundamental functionality of the IC module is verified, the focus turns to implementing and verifying additional features. The first of these enhancements is the calculation of the parity word for the serial control frame in the downlink portion, along with the progressive calculation of the parity word in the uplink portion. The verification process for this new feature involves four steps:

1. The initial step is to confirm that the downlink parity word is calculated correctly, adhering to the specified protocol. This involves calculations of the parity word for various serial control frames outside of the testbench, using both manual calculations and external computing tools, and then comparing the results to the parity word computed by the downlink portion.
2. The next verification step aims to ensure that the parity word is transmitted and subsequently reconstructed accurately by the uplink portion. Since test procedures for confirming the

correct reconstruction and storage of 8-bit words following the frame protocol are already established for the initial verification phase, these test are repurposed for verifying the parity word.

3. The third verification step is to validate that the uplink portion correctly calculates its own parity word. Given that this calculation is progressive, it introduces unique challenges different from those encountered in the downlink portion. Since the previous verification steps have already confirmed the accuracy of the downlink parity word calculations, a straightforward test is implemented to compare the parity words computed by each portion.

4. Finally, the last verification step addresses scenarios where the reconstructed parity word do not match the one calculated by the uplink portion. This is achieved by intentionally introducing errors into the transmitted parity word to asses the error-handling capabilities of the uplink portion.

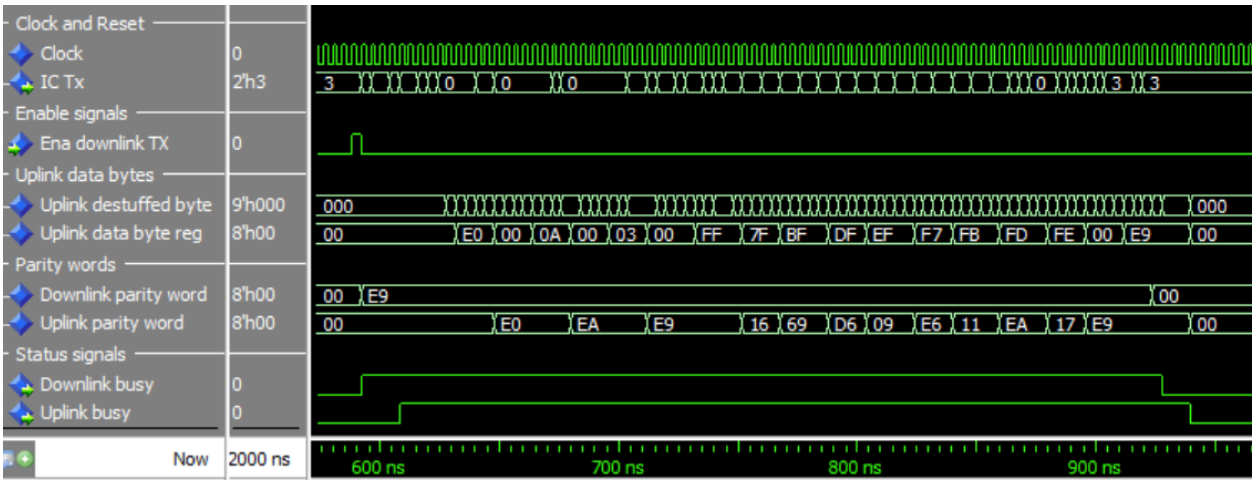


Figure 5.2: Waveguide segment extracted from IC module testbench simulation.

In Figure 5.2, a segment of the waveguide extracted from the IC module testbench simulation is presented, focusing on signals related to the parity word calculation for both the downlink and uplink sections. The waveguide illustrates that the computation of the downlink parity word occurs immediately after the activation of downlink transmission. Conversely, the uplink parity word computation is depicted as an incremental process, derived from 8-bit words that have undergone serialization and destuffing within the uplink portion. This 8-bit word used for the uplink parity word calculation is updated each time the destuffing vector has accumulated a new 8-bit word. The stored 8-bit word is employed for calculations just prior to updating, rather than immediately after updating, this is done to ensure that the received parity word is not included in the calculations, an approach necessary since the uplink portion relies on the presence of the end delimiter to distinguish the received parity word.

Verification Related to Bit-stuffing and Destuffing

The next feature incorporated into the IC module design is the implementation of bit-stuffing in the downlink portion and its reverse process, destuffing, in the uplink portion. Verifying this functionality involves combining various simulation configurations with manual verification. While confirming that the transmitted serial control frames matches their reconstructed counterparts with both bit-stuffing and destuffing functions in place indicates successful reversal by the uplink portion, it is insufficient to guarantee proper functionality. In a worst-case scenario, both processes can malfunction in a way that they mask each others errors, resulting in seemingly successful checks. Therefore, a thorough manual inspection of the serial stream is necessary to validate proper bit-stuffing. This manual inspection involves examining different patterns and lengths of serial control frames to ensure that bit-stuffing occurs where it should and that no issues are introduced. Examining various patterns helps verify that bit stuffing within different parts of the 8-bit words, as well as transitions between 8-bit words, do not disrupt the systems operation. Similarly, testing different lengths of serial control frames is essential to confirm that consecutive bit-stuffing actions do not introduce any unexpected issues.

Corner Case Verification

Verification of the IC module's operation is crucial to confirm that it functions according to its design specifications, ensuring that all its features perform as intended under typical operating scenarios. However, to truly asses the resilience and reliability of a module, it is essential to go beyond standard testing and delve into the verification of corner cases. These corner cases encompass rare and often unforeseen scenarios that could potentially exert a substantial influence on the performance or safety of the module. While the specific testing methods for these corner cases may vary, the goal remains to fortify the module's ability to handle an extensive range of inputs and conditions. How the verification is performed will not be detailed in this section, but some of the uncommon scenarios that have been subjected to verification are listed below.

- The outcome when a user attempts an action while another operation is already in progress.
- The response when only the uplink portion undergoes a reset during the transmission of a serial control frame.
  - How does the uplink portion reacts upon encountering the end delimiter?
- The response when only the downlink portion undergoes a reset during the transmission of a serial control frame.
  - How does the uplink portion react?

- The behavior when a user writes data to the downlink module and then initiates a read operation.
- The behavior when a user initiates a write operation without having previously written any data to the downlink portion.

Figure A.1 in Appendix A presents a code snippet detailing the parameters required to initiate a complete self-test procedure. This procedure involves loading data to the downlink portion, starting transmission, and concurrently reading and verifying the transmitted data. Additionally, Figure A.2 displays the log of test case headers for the IC module simulation, providing insights into simulated scenarios.

5.1.3 IC IPbus Slave Testbench

The primary objective of the IC slave testbench is to verify communication between the slave and the previously validated IC module. To facilitate this verification, a testbench is established, encompassing both the slave, the module, and auxiliary components. Given the confidence in the functionality of the IC module, any difficulties or unusual behaviors encountered during simulation are anticipated to be linked to the behavior of the IC slave, as the two components interact to achieve the intended functionality. This approach allows for a more targeted and efficient debugging process during the development of the IPbus IC slave. Figure 5.3 presents a simplified block diagram depicting the testbench setup for the IPbus IC slave.

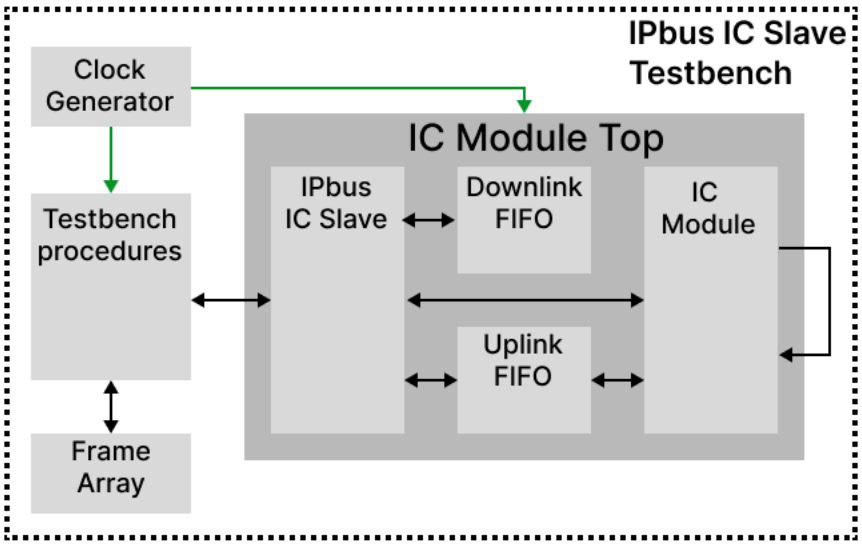


Figure 5.3: Block diagram of the testbench setup for the IPbus IC slave.

Similar to the approach taken with the IC module, a series of standardized procedures are constructed to streamline the testing process for the IC slave. These procedures are designed to enable

the simulation of various IPbus transactions with the slave, simplifying the process of initiating transactions to just a single line of code within the testbench. For a write access transaction, the procedures requires only the payload data intended for transmission to the slave. This data needs to follow the configuration detailed in Table 4.2 of section 4.5, in accordance to the specific register transaction. In contrast, for write trigger or read access transactions, no additional data is necessary, and the procedure should be called without any arguments. All other essential information, including internal slave register addresses and enable signals are automatically managed by these procedures.

Additionally, each of these procedures are equipped with logging capabilities, providing the users with transaction details. This approach ensures that users have access to a range of information regarding every transaction executed, enhancing transparency and facilitating effective testing and debugging. As illustrated in Figure 5.4, a sample of these log messages reveals transaction details for both write and subsequent read operations on the address and control registers. Figure 5.5 complements this by showcasing a portion of the waveguide extracted from the IPbus IC slave testbench simulation, offering a visual representation of the signal behaviors of the logged transactions.

```
WRITE to ipbus IC register: (x"03"): lpGBT address = (x"55"), and memory address = (x"3333")
WRITE to ipbus IC register: (x"04") : command = (x"FF"), read/write bit = (1), and payload size
= (x"00A")
READ from IPbus register: (x"03"): lpGBT address = (x"55"), and memory address = (x"3333")
READ from IPbus register: (x"04"), command: (x"FF"), read/write bit: (1) and payload size:
(x"00A")
```

Figure 5.4: Transaction log messages for write and read operations on address and control registers.

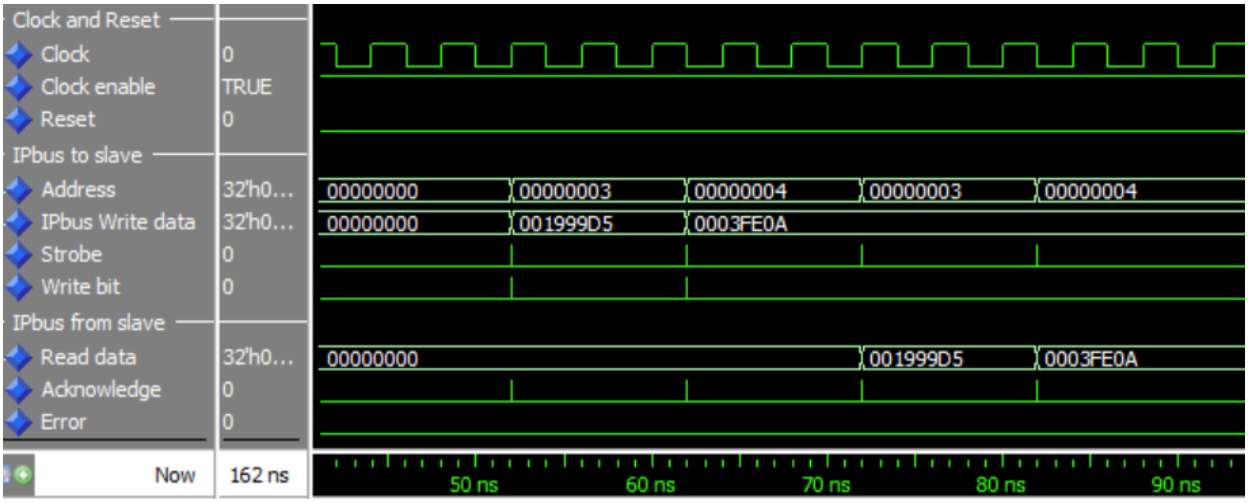


Figure 5.5: Waveguide segment extracted from IPbus IC slave testbench simulation.

The testbench designed for the IPbus IC slave exhibits a more streamlined approach compared to the testbench for the IC module. This streamlined approach is primarily due to the absence of complex features within the slave. The testbench ensures compatibility with the IPbus firmware and protocol by incorporating the IPbus Bus Functional Model (BFM)[36] developed by *Ola Slettevoll Grøttevik*, alongside the IPbus firmware[34]. The testing involves checking all access types of each register within the IC slave, using various payload data configurations. This approach guarantees that the IC slave operates seamlessly with the IC module and adheres to the specified requirements of the IPbus firmware and protocol. In this phase of development, the primary focus is on confirming the correct operation of the slave, including addressing some minor corner cases. Given the relatively straightforward nature of the slave's functionality, this level of verification is deemed sufficient to meet the requirements. In Appendix A, Figure A.6 and A.7 present the log of test case headers for the IPbus IC slave simulation and the coverage report of the same simulation, respectively.

5.1.4 lpGBT Simulation

To confirm the IC module's compatibility with the lpGBT ASIC and additional verification step is performed. To facilitate this verification, the IC module is integrated into a lpGBT simulation testbench designed by the lpGBT team. This testbench includes all the necessary components for simulating bidirectional communication between the back-end and front-end. For the back-end, it incorporates the lpGBT FPGA core, CDC components for both the downlink and uplink, and utilizes an ideal Serializer/Deserializer (SerDes) to emulate an FPGA MGT receiving a reference clock matching the upstream line-rate. For the front-end, the testbench incorporates an lpGBT model to emulate lpGBT ASIC operations. By combining the IC module with the lpGBT simulation testbench, it is possible to verify that the lpGBT model recognizes the serial control frames transmitted from the IC module, and conversely, that the IC module accurately reconstructs the frame received from the lpGBT model. Figure 5.6 provides a block diagram of the lpGBT simulation testbench with the IC module incorporated.

Integrating the IC module into the lpGBT simulation testbench is a straightforward process. It involves adding the IC module to the *lpgbtfpga_sim* entity, mapping the *BUNCH_CLOCK* to the IC module clock port, and the CDC IC user fields to the appropriate serial ports on the IC module. The test procedures developed for the IC module testbench are simplified and adapted for use without UVVM. Instead of implementing check procedures, additional signals are introduced to facilitate manual verification. This approach is considered sufficient due to the prior comprehensive verification of the IC module's functionality. To validate the compatibility between the IC module and the lpGBT model, manual verification is used to check that the expected serial control frame

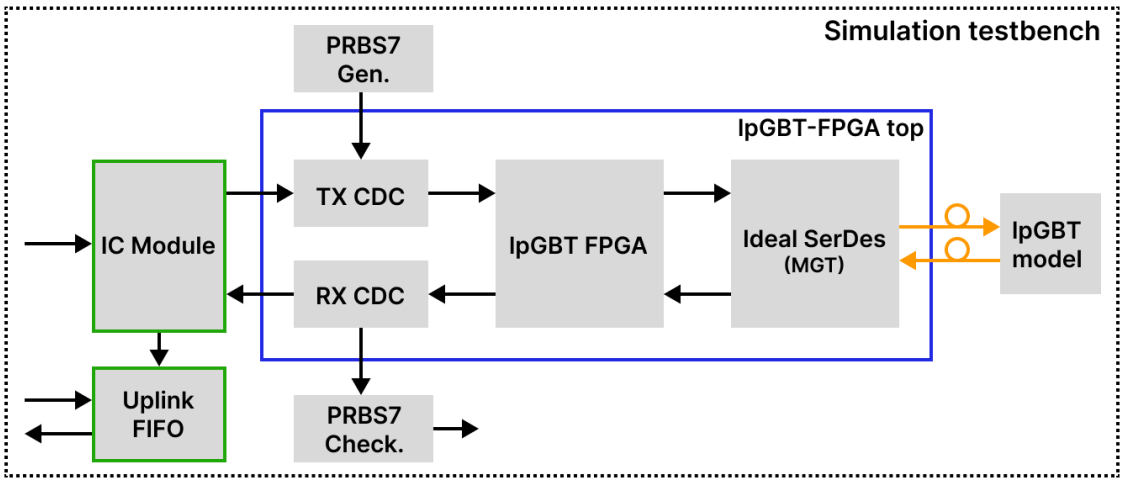


Figure 5.6: Block diagram of lpGBT model testbench simulation with integrated IC module.

is received from the lpGBT model, in response to a serial control frame transmitted from the IC module. Various serial control frames are transmitted to confirm the correct response of the lpGBT model. These frames encompass independent read operations, write operations followed by read operations, and frames that require bit-stuffing, all of which serve to validate the interaction between the IC module and the lpGBT model.

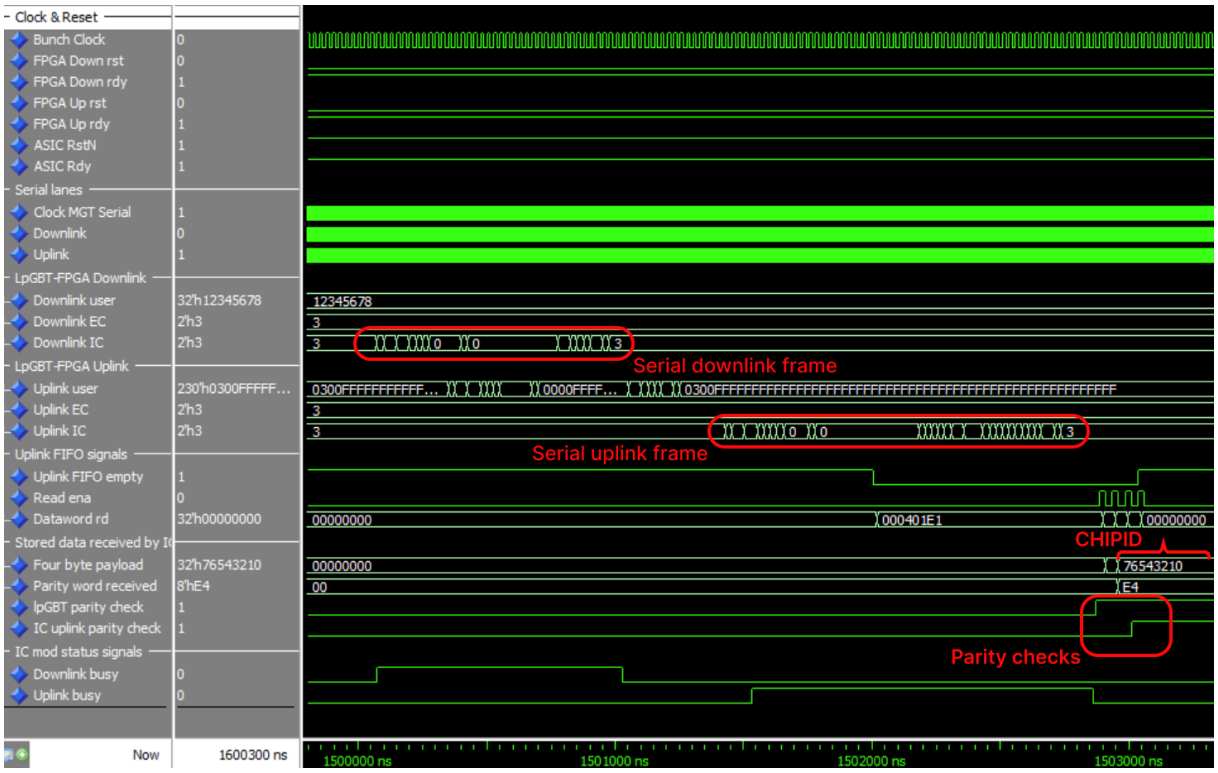


Figure 5.7: Waveguide segment extracted from lpGBT model testbench simulation.

In Figure 5.7, a segment of the waveguide extracted from the simulation is depicted. To enhance clarity, numerous irrelevant signals have been removed, while all signals below the "*Uplink FIFO signals*" have been added to facilitate the verification process. The waveguide represents the transmission of a serial control frame via the downlink channel. This frame involves a read operation targeting the four **CHIPID** registers in the lpGBT model, register address [0x000] to [0x003]. In the lpGBT model code, the **CHIPID** is defined as *x"76543210"*, and the lpGBT address is defined as the standard *b7"1110000*. Following the completion of the serial frame transmission, initiation of IC data reception on the uplink channel can be observed. This uplink frame is longer than the downlink frame due to the request for four data payload words. When the reception of this frame concludes, the IC module's uplink busy signal drops, triggering the testbench to read data from the uplink FIFO. The relevant data is stored in additional signals and displayed in the waveguide. Specifically, the four data payload words are stored, and it can be observed that this data matches the expected **CHIPID** value. Additionally, the parity check bit applied to the frame by the lpGBT model, along with the parity check bit written to the FIFO by the IC module uplink portion, is displayed. Both of these values are set to high, indicating a successful parity check. In Figure A.3, A.4, and A.4 of Appendix A, additional waveguides representing serial control operations performed on the lpGBT model is depicted.

5.2 Testing

Once the correctness and functionality of the FPGA design have been verified through simulations, the next step is to implement it on actual hardware for testing and validation. This step is needed to confirm that the design not only operates correctly in physical applications but also interfaces correctly with the external components it is integrated with. To ensure efficient and reliable testing, the testing process is typically divided into phases.

The initial phase involves validating the performance of the FPGA design in isolation, using a physical FPGA board. This phase ensures that the design functions as expected within the hardware environment, taking into account the unique characteristics of the FPGA chip. Subsequently, the design is integrated into the larger system where it will be deployed. This integration phase verifies the designs ability to collaborate with other system components and fulfill its intended role within the broader context. It is important to recognize that FPGA design is an iterative process, and designers may need to revisit various stages of the design as issues or optimization opportunities arise during testing and validation. This iterative approach allows for continuous refinement and improvement to ensure the final FPGA implementation meets all requirements and performance targets.

In the context of testing, this thesis primarily focuses on initial testing, which serves the purpose of validating the startup procedures of the VLDB+ and establishing communication with the board through the PiGBT toolkit. This testing phase encompasses tasks such as gaining access to the lpGBT I2C controller using a *RaspberryPi4* and utilizing the PiGBT web application for control. The PiGBT web application simplifies the setup, monitoring and, configuration of operational parameters for the lpGBT ASIC on the VLDB+. The application provides the capability to both read and write the internal registers of the lpGBT, making it a useful resource for ensuring proper communication between the IC module and the lpGBT. The Graphical User Interface (GUI) for the internal register I2C master is illustrated in Figure 5.8, with additional screenshots of the PiGBT web application GUI presented in Appendix B for reference.

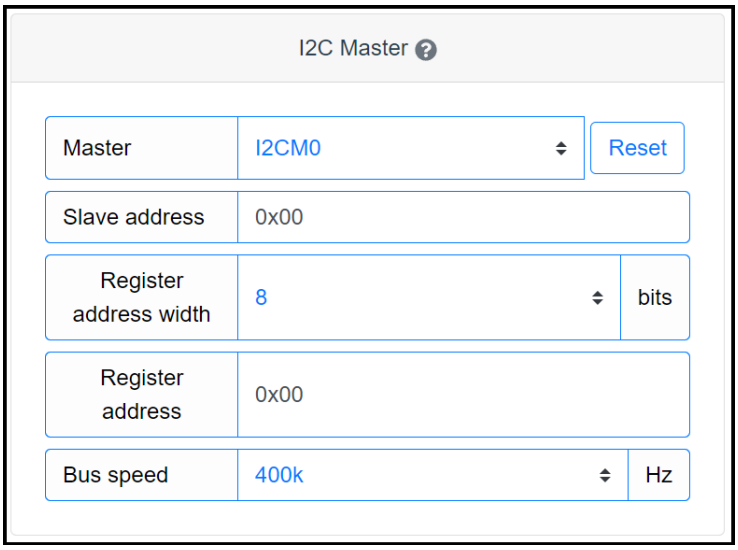


Figure 5.8: PiGBT web application - I2C Master register control.

Moving forward the next testing phase should delve into validating the practical implementation of the design on hardware, with particular emphasis on confirming the fundamental functionality of the design. This includes ensuring effective communication with the IPbus IC slave registers and testing the basic operations of the IC module. To perform an initial test of the IC design, a setup similar to that used in the testbench simulations can be employed, connecting the serial ports of both the downlink and uplink portions.

As testing progresses, this setup can be further enhanced to incorporate the CDC and lpGBT FPGA core components. This setup can still take advantage of the loop-back of the serial lane, as the data rate for the IC field remains consistent between the uplink and downlink sections. Once operational functionality is validated for the back-end components, the system can be integrated with the VLDB+ to test the overall system operations in conjunction with the lpGBT ASIC, thus completing the testing and validation process of the IC system design. In Figure 5.9, the complete physical

setup of the test-suite is depicted, showcasing the *KCU105 devboard*, the VLDB+ featuring the lpGBT ASIC, and the *Raspberry Pi 4* coupled with a level translator board to ensure compatibility with the lpGBT IOs. The pigtail originating from the VTRx+ is intended to connect with fiber optic cables through an MT-MT connection to the fiber optic transceiver on the KCU105. However, the establishment of this MT-MT connection necessitates the insertion of guide pins, which are currently unavailable. The absence of these guide pins has halted the hardware testing process.

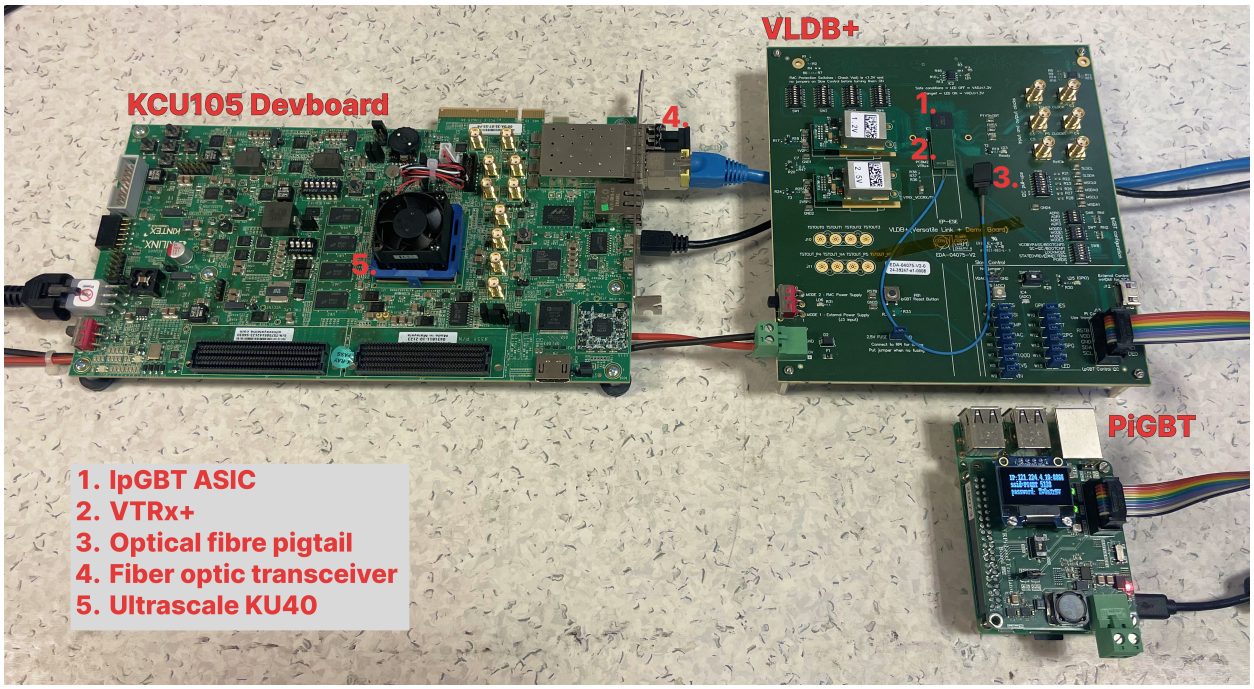


Figure 5.9: lpGBT Test-suite physical setup.

6 Discussion and Conclusion

This chapter begins by summarizing the work described in this thesis. It then moves on to a discussion and evaluation of this work. Additionally, it covers the future aspects of the project, outlining the remaining work required to complete the lpGBT test-suite.

6.1 Summary

The primary objective of this thesis was to evaluate the Versatile Link Plus (VL+) ecosystem, with a particular focus on gaining an understanding of the lpGBT ASIC's functionality and operation. Special attention was given to the integration of the lpGBT within the pixel readout chain, focusing on back-end integration. To achieve this understanding, the development of an emulation system mimicking a CRU was envisioned. Extensive verification was anticipated during the development of the emulation system to ensure its intended functionality before integration and testing with the lpGBT chipset on the VLDB+.

In order to test and validate communication with the lpGBT ASIC, the decision was made to initiate the development of the emulation system by creating components dedicated to managing the uplink and downlink IC channels. This approach was driven by the recognition that overseeing this data flow offered the ability to both write and read internal registers within the lpGBT ASIC, enabling bidirectional back-end communication. The reasoning behind this decision was the expectation that this method would serve as the most straightforward means to verify established communication with the lpGBT, as it involved only back-end components operating with similar interfaces. Suggestions for the design of the complete test-suite have been presented, with a more detailed exploration of the back-end interface for the payload data.

The design of the IC module prioritizes user-friendliness, ensuring simplicity in its operation. The downlink portion organizes data from independent ports in accordance with the serial control frame protocol. The uplink portion reconstructs the frame into 32-bit words, storing them in a FIFO after removing delimiters. The choice of 32-bit words facilitates compatibility with the IPbus protocol, enabling user communication with the module. Both the uplink and downlink portions automatically calculates parity words for the serial control frames they receive. Additionally, the module incorporates bit-stuffing for the downlink and destuffing for the uplink. A dedicated IPbus slave for the IC module was created to integrate IPbus firmware into the system. The design focused in enhancing simplicity and user-friendliness, aiming to minimize the necessary user inputs for

initiating IC module operations.

The IC module has undergone verification through independent testbench simulations to confirm its intended functionality. Additionally, a lpGBT model simulation testbench was employed to asses the module's compatibility with the lpGBT ASIC ensuring the accuracy of the design specifications. The IPbus slave for the IC module underwent a general verification process, confirming its proper operation with the IPbus protocol and the IC modules.

6.2 Discussion

The developed and verified components of the test-suite are primarily focused on the IC channel, aiming to facilitate back-end serial communication with the lpGBT ASIC. The decision to prioritize these elements was driven by their potential to validate communication with the lpGBT and their perceived simplicity as a starting point. However, as the project has advanced, unforeseen complexities and challenges have surfaced, turning what initially seemed like the most straightforward starting point into one of the more challenging aspects of the project. Initiating the project by setting up data flow management for payload data between the front-end and back-end presents a more straightforward process. This is attributed to the fact that this specific field of the lpGBT frame lacks additional protocols, requiring no extra processing for incoming or outgoing data. All necessary operations are efficiently handled by the lpGBT ASIC and the lpGBT FPGA core. Furthermore, the e-port configuration in relation to frame allocation adds to the simplicity of testing, allowing a focused examination of as little as 8-bits per lpGBT frame.

In the design of the IC module, the primary challenge surfaced during the determination of the transmission protocol. Gathering essential information proved challenging, with some details being difficult to comprehend or entirely left out. Initially intended as a straightforward component for organizing the serial control frame structure and shifting it out two bits at a time, the complexity grew unexpectedly. The realization that bit-stuffing and destuffing were not handled by the lpGBT FPGA core added complexity to the module's design. This was particularly notable due to the aspect of shifting out two bits simultaneously, introducing several factors that needed careful consideration in the overall design process.

The next challenge concerning the transmission protocol was determining the order of data transmission and reception. While the use of the 8-bit frame structure was evident, details regarding the order in which these eight bits were transmitted remained unclear. Given that all other transmission orders in the lpGBT system adhered to the MSB-first convention, it was assumed that this also applied for the 8-bit words in the serial control frame. However, receiving no response from the lpGBT simulation testbench, the assumption was reconsidered, leading to the conclusion that

the transmission order must be LSB-first. Despite this adjustment, the expected response was not achieved prompting an extensive investigation to identify the root cause. Each simulation run of the lpGBT model testbench demanded 10-15 minutes due to the high transmission rate of the ideal SerDes, making debugging a time-consuming endeavor. Eventually it was discovered that while the 8-bit words were transmitted with the LSB-first convention, the two IC bits within the serial control frame followed the MSB-first convention. This meant that the bits were initially shifted out and then required flipping. This discovery highlighted the complexity in understanding and navigating the transmission conventions during the development process.

Due to the fragile nature of the VTRx+ pigtail, it was decided to use only the original guide pins and avoid substitutes that might pose a risk of damaging the components. Acquiring the original guide pins turned out to take more time than expected. Consequently, the hardware testing for the system has been delayed, as the absence of the original guide pins prevents the establishment of a back-end link between the KCU105 Devboard and the VLDB+. Despite this setback, it is worth noting that the IC module has undergone verification in simulation with an lpGBT model, and the IC IPbus slave has been successfully verified in simulation with the IPbus firmware and BFM. As a result, there is an assumption that the system will be operational with the lpGBT ASIC on the VLDB+ wit only minor modifications.

6.3 Further Work

While the IC module and its associated slave have undergone development and verification, the next crucial phase involves their implementation and testing on physical hardware. The ultimate goal is to asses their performance in conjunction with the lpGBT on the VLDB+. However, as the IC module has been validate through simulations using a lpGBT model, it is anticipated that only minor adjustments may be necessary to ensure operation with the physical ASIC.

Moving forward with the test-suite development, the focus should shift towards enabling interaction with both the front-end interface of the e-Links, and the back-end interface of the lpGBT frame's payload data field. This integration is important for establishing data flow management within the testing framework. Furthermore, the incorporation of data generators and checkers dedicated to the payload data is suggested, providing a means to conduct thorough testing of the lpGBT. To enhance testing flexibility and functionality, the implementation of an IPbus API abstraction is recommended. This abstraction layer will play an important role in facilitating testing across various aspects of the lpGBT, utilizing the components developed for the test-suite.

6.4 Conclusion

Within the context of the objective, achieving the set goal was ambitious. Although the test-suite remains incomplete, the efforts undertaken in this thesis have established a foundation for future development. The construction of the IC module provides a platform for initial testing of the back-end communication with the lpGBT. Furthermore, the thesis compiles a range of information about the systems involved and various specifications and protocols that are important for the completion of the test-suite.

References

[1] S. Braibant, G. Giacomelli, and M. Spurio, *Particles and Fundamental Interactions: An Introduction to Particle Physics*. Bologna: Springer, 2012, pp. 1–3.

[2] J. Rafelski, Ed., *Melting Hadrons, Boiling Quarks - From Hagedorn Temperature to Ultra-Relativistiv Heavy-Ion Collisions at CERN*. Tucson: Springer, 2015, pp. 271–274.

[3] CERN, *ALICE schematics as during RUN3 (after upgrade)*, CERN Document Server, Accessed: 14.08.23. [Online]. Available: https://cds.cern.ch/images/ALICE-PHO-SKE-2017-001-5.

[4] CERN, *Longer term LHC schedule*, Accessed: 14.08.23. [Online]. Available: https://lhc-commissioning.web.cern.ch/schedule/LHC-long-term.htm.

[5] ALICE Collaboration, "Letter of Intent: A Forward Calorimeter (FoCal) in the ALICE experiment," CERN, Geneva, Tech. Rep., 2020, Accessed : 20.08.23. [Online]. Available: https://cds.cern.ch/record/2719928.

[6] P. Moreira, S. Baron, S. Biereigel, *et al.*, *lpGBT documentation: release*. 2022, Accessed : 22.08.23. [Online]. Available: https://cds.cern.ch/record/2809058.

[7] J. Troska, C. Soos, and L. Olanterä, "The Versatile Link+ Application Note," 2021, Accessed: 24.08.23. [Online]. Available: https://edms.cern.ch/document/2149674/1.

[8] B. Abelev, J. Adam, D. Adamová, *et al.*, "Technical Design Report for the Upgrade of the ALICE Inner Tracking System," Tech. Rep., 2014, Accessed : 29.08.23. DOI: 10.1088/0954-3899/41/8/087002. [Online]. Available: https://cds.cern.ch/record/1625842.

[9] ALICE Collaboration, "Technical Design Report of the ALICE Forward Calorimeter (FoCal)," Unpublished.

[10] C. V. Scheel, "The Spaghetti Calorimeter, Reasearch, Development, Application," Accessed : 31.08.23, Ph.D. dissertation, Faculty of Phys. and Astron., Univ. of Amsterdam, Amsterdam, Netherlands, 1994. [Online]. Available: https://inis.iaea.org/.

[11] T. Chujo, "Towards a FoCal for the ALICE experiment," *Newsletter of the EP department*, 2022, Accessed : 31.08.23. [Online]. Available: https://ep-news.web.cern.ch/content/towards-focal-alice-experiment#.

[12] A. Honma, "Silicon Detectors," *NATO Sci. Ser. II*, vol. 123, H. B. Prosper and M. Danilov, Eds., pp. 245–271, 2003. DOI: 10.1007/978-94-010-0076-5_6.

[13] H. A.Bethe and J. Ashkin, "Passage of radiations through matter," *Experimental Nuclear Physics*, vol. 1, E. Segrè, Ed., pp. 166–357, 1953, Accessed : 06.09.23. [Online]. Available: archive.org/details/dli.ernet.242816.

[14] D. Green, "Electromagnetic calorimetry," in *The physics of particle detectors*. Cambridge, UK: Cambridge University Press, 2000, vol. 12, pp. 251–256.

[15] G. Zomer, "Photon and pi0 separation by shower shape analysis in the forward calorimeter," Univ. of Utrecht, Utrecht, NL, 2013, Accessed : 12.09.23. [Online]. Available: https://studenttheses.uu.nl/handle/20.500.12932/15299.

[16] S. V. Nesbø, "Readout Electronics for the Upgraded ITS Detector in the ALICE Experiment," Ph.D. dissertation, Dept. of Phys. and Tech., Univ. of Bergen, Bergen, NO, 2022.

[17] ALICE ITS ALPIDE development team, "ALPIDE Operations Manual," Tech. Rep., version 0.3, 2016, Accessed : 20.09.23. [Online]. Available: https://usermanual.wiki/Document/ALPIDEoperationsmanualversion03.952786419/view.

[18] B. G. Taylor, "Timing distribution at the LHC," 2002, Accessed : 07.10.23. DOI: 10.5170/CERN-2002-003.63. [Online]. Available: https://cds.cern.ch/record/592719.

[19] The ALICE Collaboration, "Technical Design Report for the Upgrade of the ALICE Inner Tracking System," Tech. Rep., 2014, Accessed : 04.10.23. [Online]. Available: https://cds.cern.ch/record/1625842.

[20] AMD, *Kintex UltraScale FPGAs Data Sheet: DC and AC Switching Characteristics*, xilinx.com, 2020.

[21] K. W. P.Moreira J. Christiansen, Ed., *Gbtx manual*, version 0.18 DRAFT, 2021.

[22] W. Millen, "In space no one can hear electornics scream," *Mouser*, 2015, Accessed : 06.10.23. [Online]. Available: https://www.eeweb.com/in-space-no-one-can-hear-electronics-scream/.

[23] J. Schambach, "ITS Upgrade Data Format," Unpublished internal document.

[24] J. Schambach, "ITS-UL Specification," Unpublished internal document.

[25] O. Bourrion, J. Bouvier, F. Costa, *et al.*, "Versatile firmware for the Common Readout Unit (CRU) of the ALICE experiment at the LHC," *Journal of Instrumentation*, vol. 16, no. 05, pp. 5–19, May 2021, Accessed : 10.10.23. DOI: 10.1088/1748-0221/16/05/P05019. [Online]. Available: https://dx.doi.org/10.1088/1748-0221/16/05/P05019.

[26] J. Cachemiche, P. Duval, F. Hachon, R. L. Gac, and F. Réthoré, "The PCIe-based readout system for the LHCb experiment," *Journal of Instrumentation*, vol. 11, no. 02, pp. 2–13, Feb. 2016, Accessed : 10.10.23. DOI: 10.1088/1748-0221/11/02/P02013. [Online]. Available: https://dx.doi.org/10.1088/1748-0221/11/02/P02013.

[27] ISO, *ISO/IEC 13239:2002 Information technology — Telecommunications and information exchange between systems — High-level data link control (HDLC) procedures*. 2002, Accessed : 15.10.23. [Online]. Available: https://www.iso.org/standard/37010.html.

[28] I. S. Reed and G. Solomon, "Polynomial codes over certain finite fields," *Journal of the Society for Industrial and Applied Mathematics*, vol. 8, no. 2, pp. 300–304, 1960. DOI: 10.1137/0108018. eprint: https://doi.org/10.1137/0108018. [Online]. Available: https://doi.org/10.1137/0108018.

[29] J.Troska, "Versatile link+ transceiver (vtrx)," CERN, Tech. Rep., 2023, Accessed: 20.10.23. [Online]. Available: https://edms.cern.ch/document/1719329/1.

[30] VLDB+ Team, *VLDB+ manual*, Accessed : 25.10.23, 2023. [Online]. Available: https://vldbplus.web.cern.ch/manual/.

[31] VLDB+ Team, *VLDB+ control toolkit manual*, Accessed : 25.10.23, 2023. [Online]. Available: https://pigbt.web.cern.ch/manual/.

[32] C. Ghabrous Larrea, K. Harder, D. Newbold, *et al.*, "IPbus: a flexible Ethernet-based control system for xTCA hardware," *JINST*, vol. 10, no. 02, p. C02019, 2015. DOI: 10.1088/1748-0221/10/02/C02019.

[33] "User datagram protocol," *RFC Editor*, vol. 768, pp. 1–3, 1980.

[34] C. Ghabrous Larrea, K. Harder, D. Newbold, *et al.*, *Ipbus-firmware*, github.com, Accessed : 25.11.23.

[35] Bitvis., *UVVM*, github.com, Accessed : 26.11.23.

[36] O. S. Grøttevik, *IPbus BFM*, github.com, Accessed : 10.12.23.

A Verification

A.1 IC module testbench simulation

In the verification process of the IC module, the testbench is executed by accessing the *scripts* folder and running the "*0_compile_and_sim_ic_mod.do*" file. This testbench includes scenarios with both user-defined values and random values. To evaluate the serial control frame transmission and reception, the "*load_read_check()*" procedure is employed, utilizing the parameters illustrated in Figure A.1. Figure A.2 shows the log of test case headers in the simulation.

```
procedure load_read_check(
    signal clk                      : in std_logic;
    constant lpgbt_addr_i           : in std_logic_vector; --7 bit
    constant num_payload_words_i    : in std_logic_vector; --9 bit
    constant memory_addr_i          : in std_logic_vector; --16 bit
    constant rw_bit_i               : in std_logic;
    constant data_i                 : in std_logic_vector; -- 8 to 4088 bits
    constant scope                  : in string := C_SCOPE
    )is
```

Figure A.1: IC module simulation - Self-test procedure code snippet.

ID_LOG_HDR	0.0 ns	TB seq.(uvvm)	Start simulation of tb for IC module
ID_LOG_HDR	50.0 ns	TB seq.(uvvm)	Check transmission and reception of single word read frame
ID_LOG_HDR	232.0 ns	TB seq.(uvvm)	Check transmission and reception of single word write frame, with all low payload data
ID_LOG_HDR	444.0 ns	TB seq.(uvvm)	Check transmission and reception of single word write frame, with all high payload data
ID_LOG_HDR	668.0 ns	TB seq.(uvvm)	Check transmission and recpetion of write frame with two payload words, payload data is delimiter
ID_LOG_HDR	908.0 ns	TB seq.(uvvm)	Check transmission and reception of write frame with 2-10 payload words, payload data all high
ID_LOG_HDR	3960.0 ns	TB seq.(uvvm)	Check transmission and reception of write frame with 2-10 payload words, payload data all low
ID_LOG_HDR	6828.0 ns	TB seq.(uvvm)	Check transmission and reception of write frame with 2-10 payload words, payload data is random
ID_LOG_HDR	9712.0 ns	TB seq.(uvvm)	Check transmission and reception of write frame with 10 payload words, various payload data with high concentration of 1
ID_LOG_HDR	10140.0 ns	TB seq.(uvvm)	Check transmission and reception of write frame with 33 payload words, payload data is random
ID_LOG_HDR	11008.0 ns	TB seq.(uvvm)	Check transmission and reception of write frame with 511 payload words, payload data is random
ID_LOG_HDR	22060.0 ns	TB seq.(uvvm)	Check corner cases

Figure A.2: IC module simulation - Log of test case headers.

A.2 lpGBT model testbench simulation

The following three figures displays segments of the waveguide extracted from the lpGBT model testbench simulation. These waveguides depict serial control operations carried out on the **USERID** internal registers of the lpGBT model (register addresses [0x004] to [0x007]). In Figure A.3, the initial serial control read operation on the registers are displayed, revealing that all registers contain zero values. In Figure A.4, a serial control write operation performed on the same registers are shown. The four 8-bit words written to the four **USERID** registers are [0xFF], [0x7E], [0xFB] and [0xF8] respectively. This data is transmitted to verify that the serial control frames for both downlink and uplink are recognized when bit-stuffing and destuffing are employed. Figure A.5 presents a serial control read operation on the **USERID** registers, following the write operation, confirming that the data read back matches the originally written data.

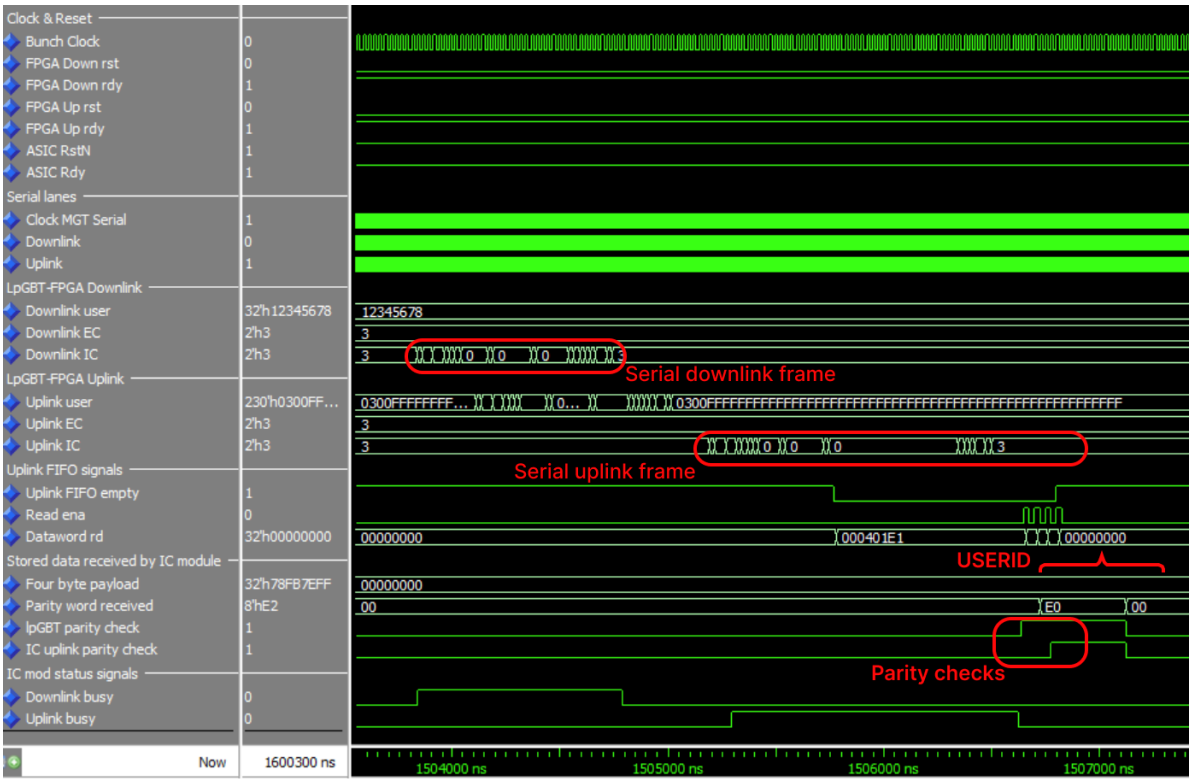


Figure A.3: lpGBT model simulation - Waveguide segment of serial control read operation of USERID registers.

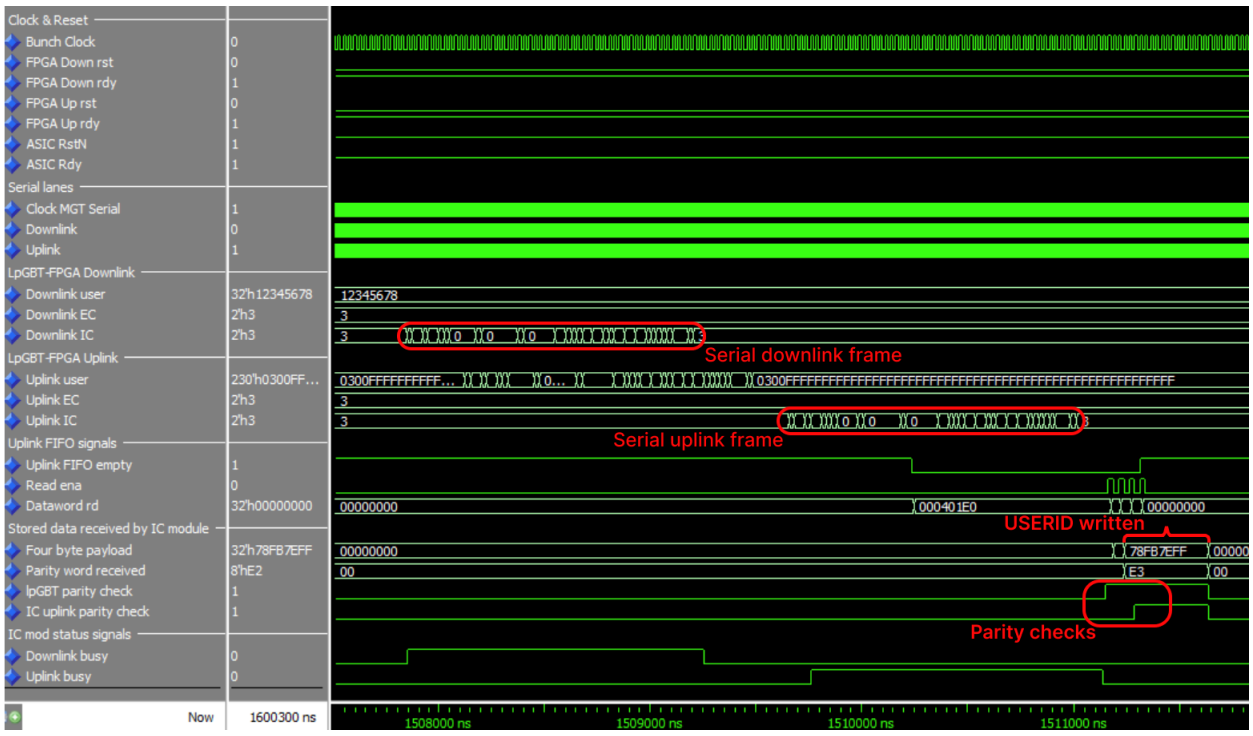


Figure A.4: lpGBT model simulation - Waveguide segment of serial control write operation of USERID registers.

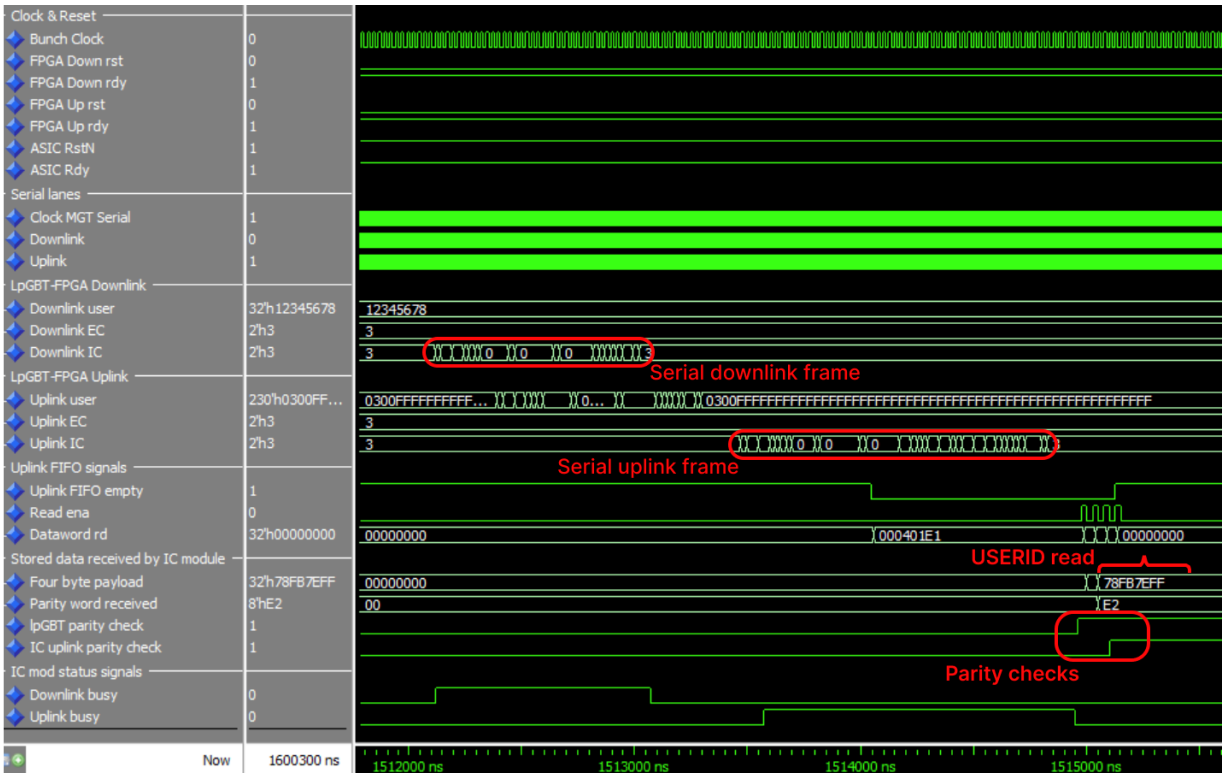


Figure A.5: lpGBT model simulation - Waveguide segment of serial control frame operation of USERID registers.

A.3 IPbus IC slave testbench simulation

```
================================================================================================================================
ID_LOG_HDR_LARGE                500.0 ns  TB seq.                       Checking Status Register (reg0) - RO fields
================================================================================


================================================================================================================================
ID_LOG_HDR_LARGE                701.0 ns  TB seq.                       Checking TX FIFO register (reg1) - Pulse fields
================================================================================


================================================================================================================================
ID_LOG_HDR_LARGE               1690.0 ns  TB seq.                       Check RX FIFO register (reg2) - RO fields and Pulse field
================================================================================


================================================================================================================================
ID_LOG_HDR_LARGE               3040.0 ns  TB seq.                       Check Addr register (reg3) - RW fields
================================================================================


================================================================================================================================
ID_LOG_HDR_LARGE               4921.0 ns  TB seq.                       Check Control register (reg4) - RW fields
================================================================================


================================================================================================================================
ID_LOG_HDR_LARGE               6401.0 ns  TB seq.                       Check Load_and_Ex register (reg5) - Pulse std_logic
================================================================================


================================================================================================================================
ID_LOG_HDR_LARGE               6430.0 ns  TB seq.                       Check Reset register (reg6) - Pulse std_logic
================================================================================


================================================================================================================================
ID_LOG_HDR_LARGE               6460.0 ns  TB seq.                       Checking that invalid register returns ERR
================================================================================
```

Figure A.6: IPbus IC slave simulation - Log of test case headers for IPbus IC slave simulation.

```
# Coverage Report Summary Data by instance
#
# ================================================================================
# === Instance: /ic_ipbus_reg_tb/DUT
# === Design Unit: work.ic_ipbus_reg(behavior)
# ================================================================================
#     Enabled Coverage              Bins      Hits    Misses  Coverage
#     ----------------              ----      ----    ------  --------
#     Branches                        46        40         6    86.95%
#     Statements                      86        79         7    91.86%
#     Toggles                        880       800        80    90.90%
#
# ================================================================================
# === Instance: /string_methods_pkg
# === Design Unit: uvvm_util.string_methods_pkg
# ================================================================================
#     Enabled Coverage              Bins      Hits    Misses  Coverage
#     ----------------              ----      ----    ------  --------
#     Assertions                       1         1         0   100.00%
#
#
# TOTAL ASSERTION COVERAGE: 100.00%  ASSERTIONS: 1
#
# Total Coverage By Instance (filtered view): 92.43%
```

Figure A.7: IPbus IC slave simulation - Coverage report.

B Testing

B.1 PiGBT

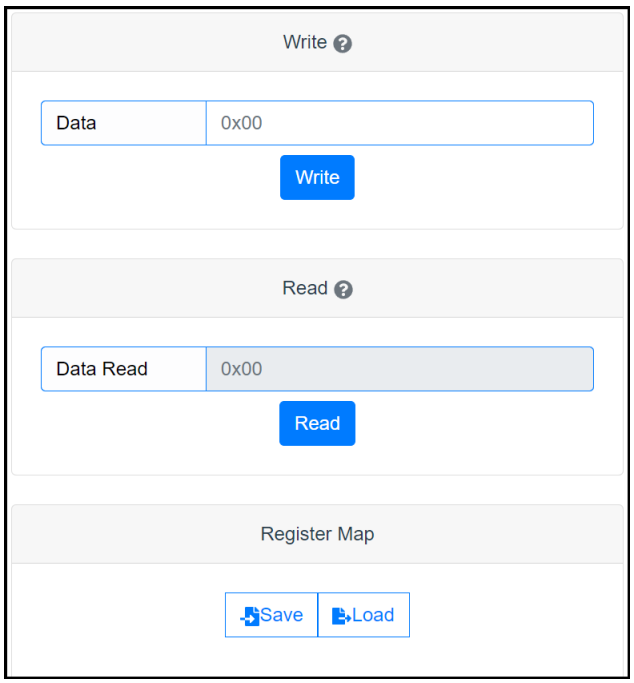


Figure B.1: PiGBT web application - I2C Master write and read fields.

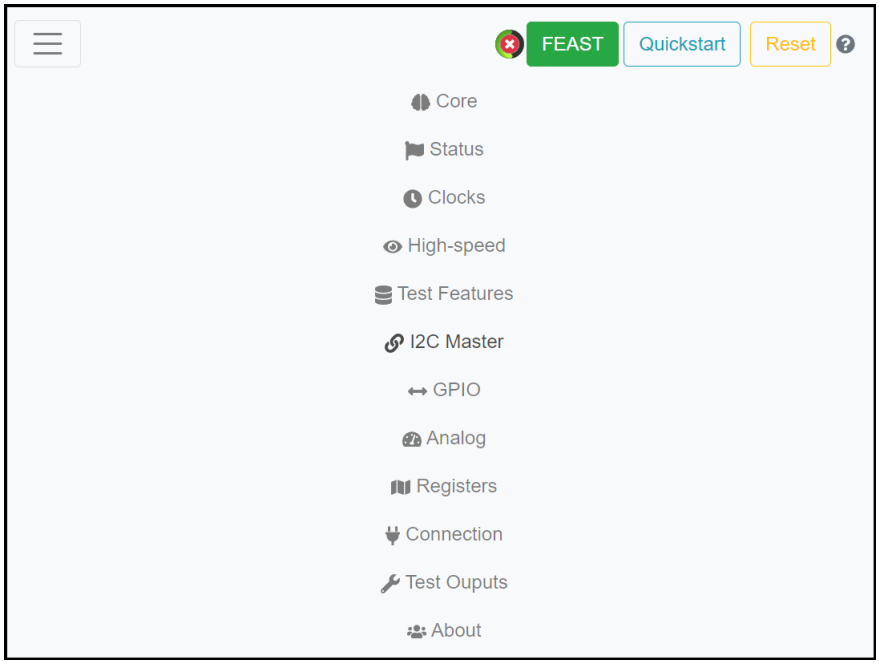


Figure B.2: PiGBT web application - I2C Master access tabs.

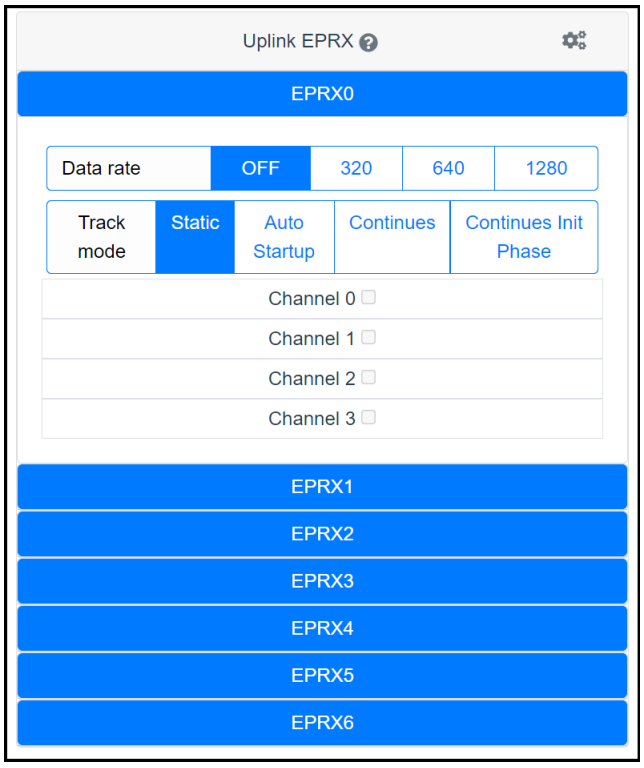


Figure B.3: PiGBT web application - Uplink e-Port configuration.

C lpGBT Frame structure

C.1 Downlink

Frame	Function	I/O Group
FRMDWN[23:0]	FEC[23:0]	
FRMDWN[31:24]	Data[7:0]	0
FRMDWN[39:32]	Data[15:8]	1
FRMDWN[47:40]	Data[23:16]	2
FRMDWN[55:48]	Data[31:24]	3
FRMDWN[56]	EC[0]	EC
FRMDWN[57]	H[0]	
FRMDWN[58]	EC[1]	EC
FRMDWN[59]	H[1]	
FRMDWN[60]	IC[0]	
FRMDWN[61]	H[2]	
FRMDWN[62]	IC[1]	
FRMDWN[63]	H[3]	H[3:0]=4'b1001

Table C.1: Downlink frame structure before interleaving[6].