Bergische Universität Wuppertal
Fakultät für
Mathematik und Naturwissenschaften

An FPGA-based Data Aggregator for the New ATLAS ITK Pixel DCS

Dissertation zur Erlangung des Doktorgrades

vorgelegt von
Ahmed Qamesh

Erstgutachter: Prof. Dr. Christian Zeitnitz

Zweitgutachter: Prof. Dr. Karl-Heinz Kampert

Wuppertal, December 2024

Abstract

The planned High Luminosity upgrade of the Large Hadron Collider (HL-LHC) at CERN will increase the collider's luminosity by a factor of ten compared to the LHC's current luminosity of 10^{34} $\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$. As part of the ATLAS Phase II upgrade, the existing tracking system will be replaced with the all-silicon Inner Tracker (ITK), which features a pixel detector as its core element. Monitoring data from the ITK system will be aggregated by an on-detector Application Specific Integrated Circuit (ASIC), known as the Monitoring Of Pixel System (MOPS), and relayed to the Detector Control System (DCS) via a newly developed FPGA-based interface known as MOPS-Hub.

The MOPS chip utilizes Controller Area Network (CAN) and CANopen protocols with a non-standard low voltage physical layer for communication, powered from the MOPS-Hub. The CAN interface is implemented in the MOPS-Hub firmware within its FPGA, while the low voltage physical layer is integrated into hardware known as the CAN Interface Card (CIC). All components are contained within a single housing, referred to as the MOPS-Hub crate. This thesis details the implementation and experimental results of the MOPS-Hub and its hardware components. Additionally, strategies for mitigating Single-Event Upsets (SEUs) and results from irradiation tests, including Total Ionizing Dose (TID) and Non-Ionizing Energy Loss (NIEL) assessments, are presented to evaluate the system's radiation tolerance.

Contents

Chapter 1

Introduction

After several years of operation, the Large Hadron Collider (LHC) will undergo a major upgrade, known as the High Luminosity LHC (HL-LHC)[1]. This upgrade is scheduled to be commissioned during the upcoming long shutdown around 2026. The HL-LHC aims to increase the LHC luminosity by a factor of 5 to 7, enhancing it from the current level of approximately 10^{34} cm^{-2}s^{-1}. The increased luminosity will facilitate the accumulation of rare physics events, enabling more precise measurements of the Standard Model parameters and the exploration of new physics beyond the Standard Model. However, this boost in luminosity will also result in higher radiation levels and increased particle flux per bunch crossing. The pixel detector, located just a few centimeters from the proton beam axis, will be most affected by these changes. Therefore, new detector technologies are needed to cope with these challenges. During the LHC Phase-II shutdown, the entire tracking system of the ATLAS experiment will be replaced by an all-silicon pixel detector called the Inner Tracker (ITk) [2]. The monitoring data of the new system will be aggregated from an on-detector Application Specific Integrated Circuit (ASIC), referred to as the Monitoring Of Pixel System (MOPS) chip, and channeled to the Detector Control System (DCS) via a newly developed FPGA-based interface known as MOPS-Hub [3].

The central work in this thesis is the development and evaluation of the MOPS-Hub. The MOPS-Hub system integrates advanced features for efficient data handling of the MOPS chip and other hardware components.

The thesis begins with Chapter 2, introducing the LHC and its High Luminosity Upgrade (HL-LHC), focusing on its impact on the ATLAS experiment and the need for advanced detector technologies. Chapter 3 details the DCS for the ITk and introduces the new Serial Power (SP) scheme. The hardware requirements and components of the MOPS-Hub crate are covered in Chapter 4. Chapters 5 and 6 build the fundamental and theoretical background for the work done in the thesis by explaining radiation effects on semiconductor devices and focusing on FPGA technology and its radiation tolerance. Chapter 7 provides a functional description of the firmware for the core FPGA component of MOPS-Hub, known as the PP3-FPGA. Chapter 8 discusses the testing and validation of the hardware components in the MOPS-Hub, and Chapter 9 presents irradiation testing for several components in MOPS-Hub. Finally, the conclusion and the outlook for this study are given in Chapter 10.

Chapter 2

The LHC and its Upgrade

This chapter explores various aspects of the LHC and its pivotal role in particle physics, while also highlighting the ATLAS experiment and its internal structure.

2.1 The Large Hadron Collider

The Standard Model is a theoretical framework that describes elementary particles and their interactions. Over the course of more than 30 years, experimental observations have consistently confirmed its predictions. Despite its success, certain phenomena remain unexplained by the Standard Model. To probe the boundaries of this model and explore new physics, scientists utilize particle accelerators. Among these, the LHC at CERN stands as the most powerful. Capable of accelerating two proton beams to energies of 7 TeV each, collisions at designated interaction points yield a center-of-mass energy of up to 14 TeV. Additionally, the LHC can accelerate beams of lead nuclei (Pb), resulting in collisions with a collision energy of 1150 TeV[4–6].

There are four interaction points in the LHC, where the beams are colliding. An experiment is located at each point performing different physics analysis. A Large Ion Collider Experiment (ALICE) is measuring the properties of the strongly interacting matter created in nucleus-nucleus collisions at the LHC energies[7]. The Large Hadron Collider Beauty (LHCb), focuses on b-quark physics and CP-violation[8]. ATLAS and Compact Muon Solenoid (CMS) experiments are designed to reconstruct events created at the interaction region to allow physicists testing the predictions of different theories of particle physics, observe new phenomena or search for evidence of theories of particle physics beyond the Standard Model [9, 10].

2.2 The High Luminosity Upgrade

Following the successful data-taking periods at the LHC by the major physics experiments since 2009, a long-term plan is already in place to fully exploit the vast physics potential of the LHC within the next two decades. After the third long shutdown of the LHC around 2026, a new Phase-II upgrade to the so-called HL-LHC is planned [1]. With this upgrade, the luminosity will increase by a factor of 10 compared to the LHC's current luminosity of 10^{34} $\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$[11].

The new machine will accumulate rare physics events crucial for more precise measurements of the Standard Model parameters and further explorations for physics beyond the Standard Model. However, this increase in luminosity comes with the challenge of managing a significantly larger amount of data delivered to all experiments. Consequently, the experiments must contend with very high detector occupancies and operate within the harsh radiation environment resulting from a vast multiplicity of particles produced in each beam crossing.

2.3 The ATLAS Experiment

ATLAS is a general-purpose detector designed for probing proton-proton and heavy ion collisions [10]. One of the most significant achievements of the ATLAS detector, together with the CMS was its collaboration in the discovery of the Higgs boson in 2012. This discovery confirmed the existence of the last missing particle predicted by the Standard Model of particle physics[12].

Figure 2.1 presents a cut-away view of the detector, with its sub-detectors clearly labeled.

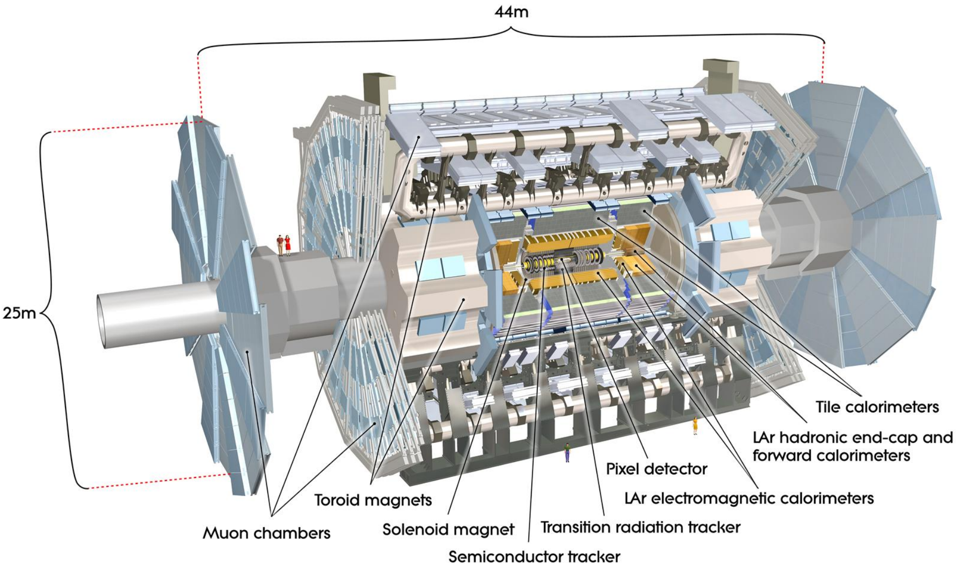

44m
25m
Tile calorimeters
LAr hadronic end-cap and forward calorimeters
Pixel detector
LAr electromagnetic calorimeters
Toroid magnets
Muon chambers
Solenoid magnet
Transition radiation tracker
Semiconductor tracker

Figure 2.1: Layout of the ATLAS detector [13]. The dimensions of the detector are 25 m in height and 44 m in length.

2.3.1 The ATLAS Structure

The ATLAS detector is constructed with a series of concentric cylinders surrounding the interaction point and endcaps. These functional components are organized into layers of sub-detectors, forming an onion-like layout with specific tasks.

Figure 2.2 depicts a slice of the ATLAS detector as well as a visualization of different particles detected inside the sub-detectors of the ATLAS experiment. The detector is constructed with forward-backward symmetry with respect to the interaction point.

Each sub-detector shown is only sensitive to certain particles and measures different properties (e.g. particle tracking, energy, and momentum). The combination of all information provided by the individual sub-systems of the ATLAS detector enables a precise reconstruction of the inelastic events which took place in the proton-proton collisions.

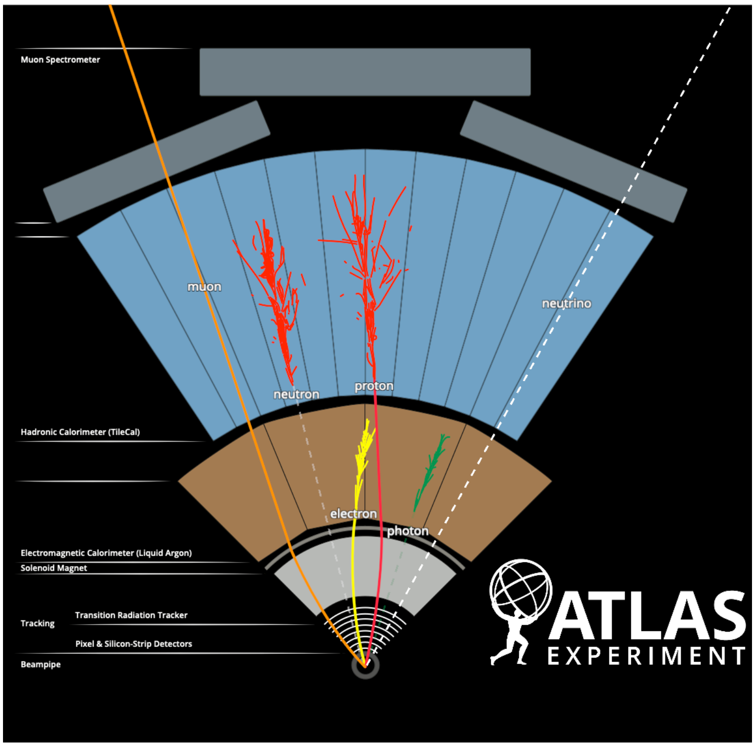

Muon Spectrometer
muon
neutrino
neutron
proton
Hadronic Calorimeter (TileCal)
electron
photon
Electromagnetic Calorimeter (Liquid Argon)
Solenoid Magnet
Tracking
Transition Radiation Tracker
Pixel & Silicon-Strip Detectors
Beampipe
ATLAS
EXPERIMENT

Figure 2.2: Visualization of particles inside the ATLAS's sub detectors[14].

At the core of this design is a superconducting solenoid magnet that surrounds the Inner detector cavity. This magnet configuration is augmented by three large superconducting toroids, one barrel and two end-caps which are strategically positioned with an eight-fold azimuthal symmetry around the calorimeters.

Operating within a 2 T solenoidal field, the Inner detector performs pattern recognition, momentum and vertex measurements, and electron identification tasks. This functionality is achieved through a combination of discrete, high-resolution semiconductor Pixel and strip detectors in the inner part of the tracking volume, complemented by the outer sub-detectors.

The outermost system of the detector is the **Muon Spectrometer**, located on the outermost layer of the detector, comprises three large superconducting toroidal magnets (two endcaps and one barrel), generating a magnetic field of approximately

(0.5 to 1) T [10]. It features multiple layers of precision chambers for tracking and measuring muon trajectories.

Following the Muon detector, the next crucial components in the ATLAS detector are the calorimeters, including the **Electromagnetic Calorimeter (ECAL)** and the **Hadronic Calorimeter (HCAL)**. The ECAL is tasked with measuring the energy of electrons and photons generated in collisions, aiding in the identification and measurement of electromagnetic particles[15]. On the other hand, the HCAL serves to measure the energy of hadrons, such as protons, neutrons, and baryons. It complements the ECAL by quantifying the energy of particles that interact via the strong nuclear force [10].

Situated at the heart of the ATLAS detector, the **Inner Detector (ID)** serves as the innermost component dedicated to accurately tracking charged particles generated in collisions. Its primary purpose lies in determining the trajectory, momentum, and charge of these charged particles with precision.

The ID features multiple layers, including the **Pixel detector**, the **Semiconductor Tracker (SCT)**, and the **Transition Radiation Tracker (TRT)** arranged cylindrically around the beam pipe. This arrangement enables the ID to capture and track the paths of charged particles, providing crucial insights into the fundamental properties of particles produced in collisions within the ATLAS detector. Since the Pixel detector is located a few centimeters from the proton beam axis, it will be most affected by the increased hit rates and radiation exposure after Phase-II upgrade. Therefore new detector technologies are needed to cope with these changes.

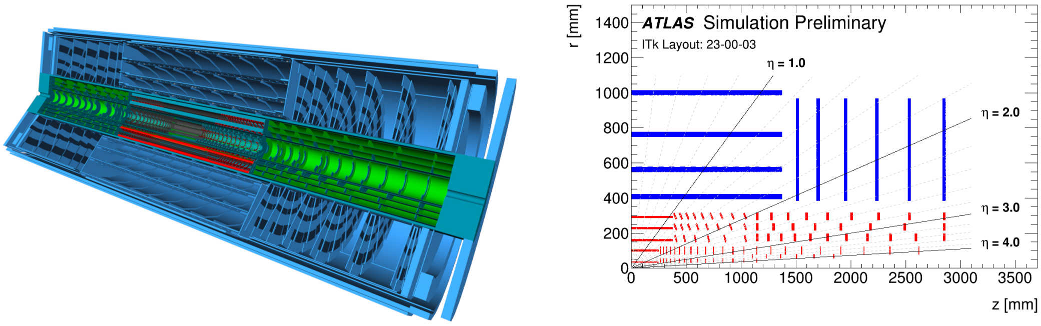
2.3.2 The Inner Tracker Detector

During the Phase-II shutdown of the LHC, the entire tracking system of the ID experiment will be replaced by an all-silicon detector known as the ITk [2]. This new detector serves a critical role in precisely determining the trajectory, momentum, and charge of charged particles resulting from collisions within the ATLAS detector. The ITk consists of several layers of silicon Pixel detectors and semiconductor microstrips arranged cylindrically around the beam pipe. Specifically, it features four layers of double-sided strip detectors and six double-sided Endcap disks. Furthermore, the ITk houses a Pixel detector consisting of five layers in the barrel section and multiple rings in the forward direction, as depicted in Figure 2.3(b).

Figure 2.3 illustrates the layout of the ATLAS ITk as detailed in the technical design report for the Pixel detector [16].

Due to its proximity to the proton beam axis, being only 34 mm away, the Pixel detector is particularly susceptible to increased hit rates and radiation exposure, corresponding to an ionizing dose of 1 Grad. Consequently, it is anticipated that the two innermost layers will require replacement after reaching half of their operational lifetime [16].

The Front-End (FE) readout chip, known as ITkPix, is developed by the RD53 collaboration [18]. This new readout chip features a matrix of 400×384 Pixels [19], providing the necessary resolution for precision track reconstruction across the Pixel



(a) Display of the ATLAS Phase-II ITk layout [16].

(b) Layout of one quadrant from the point of interaction [17].

Figure 2.3: The ATLAS ITk layout. The blue lines in Figure 2.3(b) represent the Strip part, while the red lines depict the Pixel part of the detector.

part of the ITk. Each readout chip is equipped with 4 channels, each operating at a speed of 1.28 Gbps [16].

2.4 Serial Powering for the ITk Pixel Detector

Compared to the current ATLAS Pixel detector, the area of the ID will increase from $2\,\mathrm{m}^2$ to $13\,\mathrm{m}^2$ [16], with more than 8000 Pixel modules in the new upgrade. A Pixel module in this context is a type of hybrid Pixel detector where a sensor and readout chip are separately manufactured and then electrically connected. Each Pixel in the sensor is connected to its corresponding readout cell via bump bonding [20].

To reduce the material budget in the detector volume and decrease cable power losses, the ITk implements a novel powering scheme where the modules are powered serially, while the chips inside the modules are powered in parallel [21, 22]. With the new powering scheme, the total power loss in the cable should be less than 30 % of the total required power [16].

Figure 2.4 illustrates the SP scheme of the new ITk. As depicted a constant current source, labeled as Low Voltage(LV), is used to power M modules hosting the readout chips in series. This generates a current (I_{SP}) equal to the maximum current consumption of one module. Each module has a regulator for each supply voltage, generating the supply voltage out of the constant current.

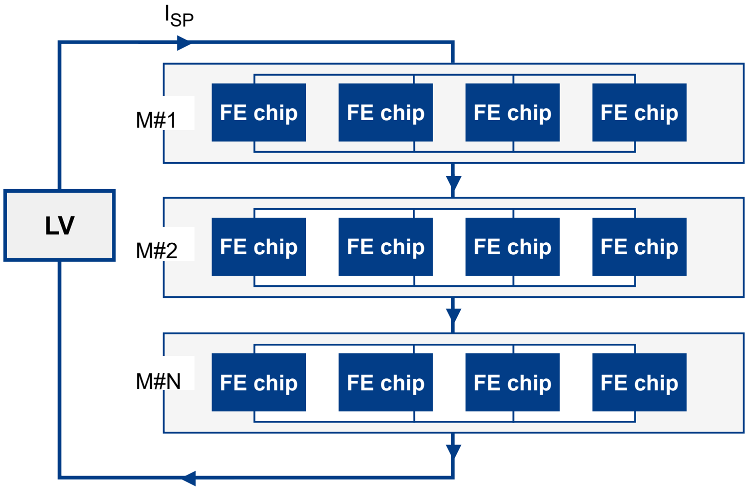


Figure 2.4: Serial powering chain for the ATLAS ITk Pixel detector.

According to the Pixel TDR [16], a single chain can have a maximum of sixteen modules but this is reduced to thirteen in the subsequent system planning. Each module can have a maximum of four FE chips bump bonded to a single sensor[23].

A SP chain is the smallest unit, which can be controlled individually from the power supplies. Over-temperature and over-voltage conditions will be reported by an independent monitoring chip called MOPS which will be detailed in Section 3.4.

Chapter 3

Detector Control System for the ITk Pixel Detector

This chapter outlines the DCS for the ITk Pixel detector in the ATLAS experiment, detailing its structure and purpose. It also introduces a new monitoring approach through an on-detector ASIC, referred to as MOPS, and its integration plan using MOPS-Hub as the core.

3.1 Scope of the Detector Control System

The DCS is designed to enable coherent and safe operation of the ATLAS detector and serves as a homogeneous interface to all sub-detectors [24]. Additionally, it supervises the experimental setup's hardware including detector services such as High Voltage (HV) and Low Voltage(LV) systems, cooling mechanisms, and overall infrastructure, including racks and environmental conditions. Furthermore, the DCS connects with external entities, including CERN's Technical Services (e.g. electricity, ventilation) and most notably to the LHC accelerator (e.g. for beam conditions and backgrounds). It is responsible for bringing the detector into any desired operational state, to continuously monitor and archive the operational parameters, to signal any abnormal behaviour to the operator, and to allow manual or automatic actions to be taken.

3.1.1 DCS Architecture

The DCS employs a highly distributed system, hierarchically organized for the detector supervision, during both operational and maintenance periods. The implemented architecture for this hierarchically is depicted in Figure 3.1.

The DCS is segmented into FE components and Back-End (BE) systems. Communication between these segments primarily occurs via CAN using the CANopen protocol (see Section 4.3.1), or through Ethernet using the Open Platform Communications Unified Architecture (OPC-UA) protocol [26, 27]. On the BE, there are the DCS computers which run a distributed system of SIMATIC WinCC OA[1] as the Supervisory Control And Data Acquisition (SCADA) software to collect status information from each sub-detector independently [24].

[1] https://www.winccoa.com/, developed by ETM professional control GmbH

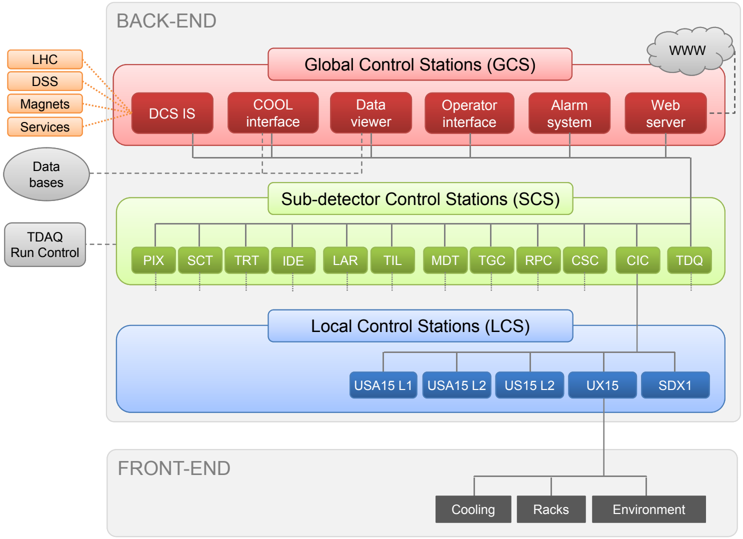


Figure 3.1: Schema of DCS architecture [25].

WinCC OA is a software package designed for the use in automation technology to monitor and control a system. It provides a scalable graphical user interface and can support redundant and distributed systems as depicted in Figure 3.2.

WinCC OA integrates a built-in OPC-UA client driver, facilitating connections with hardware components. The system visualizes the status of each element through color-coded sub-detectors, offering granular details for each. Data management is provided through the Para module, which organizes data points for SCADA systems [28]. The comprehensive DCS concept for the ATLAS ITk Pixel detector is elaborated in a specific Technical Design Report [16].

3.2 The New Pixel DCS

The concept of a new ITk Pixel DCS is driven by several requirements so that it can be used not only for the nominal operation but also during the commissioning of the new sub-systems. The DCS must have independent power and communication links/protocols to make sure that it is available at all times to allow monitoring, control, and safety of detector independent of the operational mode of the detector.

Figure 3.3 shows an overview of the new Pixel DCS system, based on [16].

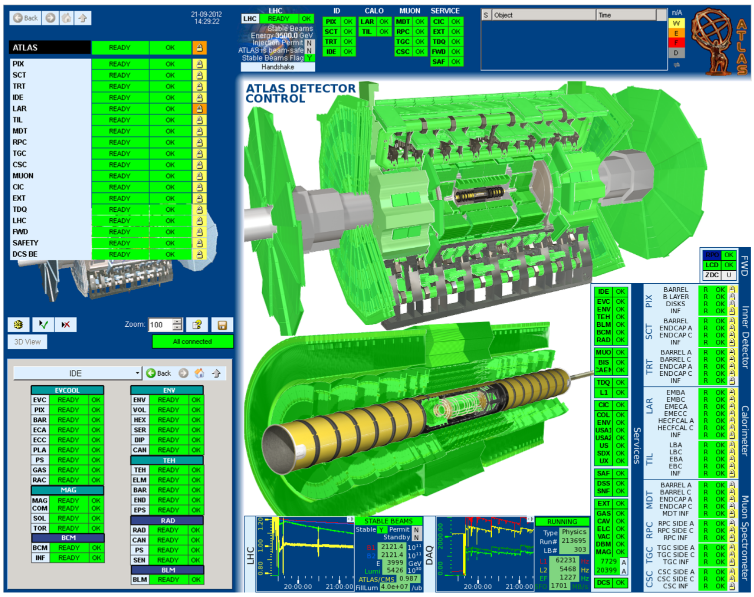


Figure 3.2: The ATLAS DCS user interface during a regular LHC fill for luminosity production with p-p collisions [25].

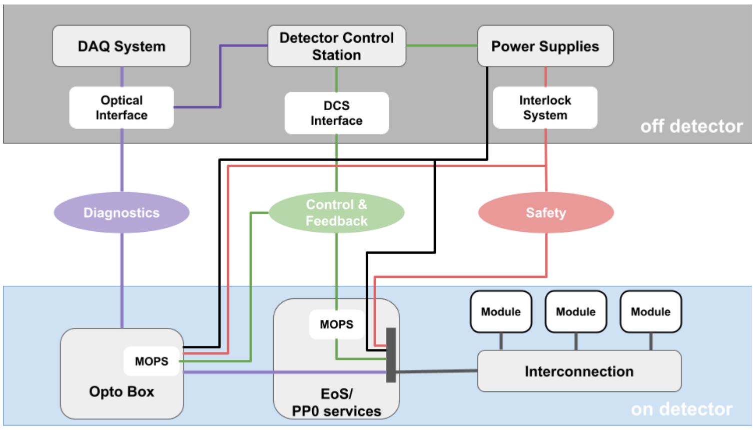


Figure 3.3: Overview of the new Pixel DCS with its paths, based on [16].

The DCS of the ATLAS detector is structured around three main pathways: **Diagnostics**, **Control & Feedback**, and **Safety**. These paths differ in granularity, availability, and reliability level.

3.2.1 Safety Path

The safety path of the DCS is primarily based on an interlock system, essentially serving as a critical safeguard for the detector against any critical malfunction [29]. The interlock system is a hardwired safety system that is designed to act on the interlock-controlled units, such as power supplies or other external devices, when potential risks arise that could endanger the experiment or operators. It gathers signals, which include temperature readings from Negative Temperature Coefficient Thermistors (NTCs) to monitor the temperature of specific interlock-protected devices, such as pixel modules in the SP chain. Additionally, it collects external signals from the ATLAS-Detector Safety System (DSS) or the ATLAS-Beam Interface System (BIS) to ensure safe operation. The ATLAS-DSS informs about failures in the cooling systems, high humidity, smoke in the experimental cavern or other global risks. Further, a signal from the ATLAS-BIS reports about dangerous beam conditions.

Figure 3.4 shows the topology of the Interlock system. The interlock system operates through a modular design, where each module serves a specific purpose and interfaces with other components, allowing sub-detectors to customize their configurations as needed. The system is housed within a 3U 19-inch crate known as Local Interlock & Safety System (LISSY). This crate contains various modules, including IO-modules, the Interlock FPGA (ILK-FPGA), and the Monitoring FPGA [30].

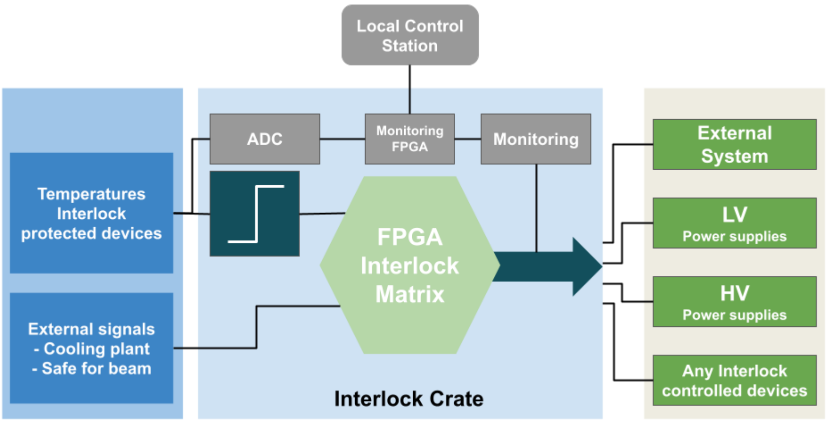


Figure 3.4: Concept of the interlock system of the ITk DCS [30].

Within the interlock crate, input from temperature sensors on the serial powering chain undergoes processing by two distinct instances. The ILK-FPGA instance handles temperature sensor data using hardwired logic, processing it after being translated into a binary signal by means of a discriminator. Simultaneously, the Analog Digital Converter (ADC) instance samples these signals and forwards them to the Monitoring FPGA, which monitors the analogue temperature signals. Furthermore, the Monitoring FPGA facilitates debugging of the interlock system by aggregating test signals to the local control station. These test signals serve to verify the functionality of LISSY and the implemented interlock matrix [16, 29].

This Safety path acts as the last line of defense, providing the highest reliability and availability in the DCS.

3.2.2 Control and Feedback Path

The Control and Feedback path within the DCS is responsible for controlling and monitoring the individual detector modules [16]. All control functionality is provided by the power supplies, as the power distribution is done per SP-chain.

The Control of the SP-chain includes one LV channel from a current source that powers the FE chips and HV channels for depletion of the sensors. The HV is supplied for a group of modules, there are at least two HV channels per SP-chain. Within the Opto-box, all Opto-boards associated to one SP-chain (up to eight Opto-boards) are collectively controlled.

The feedback of the detector modules and Opto-boards within the DCS is provided by an active chip called MOPS which is developed to perform these tasks under high radiation exposure. The MOPS ASIC will provide the monitoring of temperatures and voltages of modules of up to 16 FE detector modules in a SP-chain as well as temperatures and voltages inside the Opto-box [31].

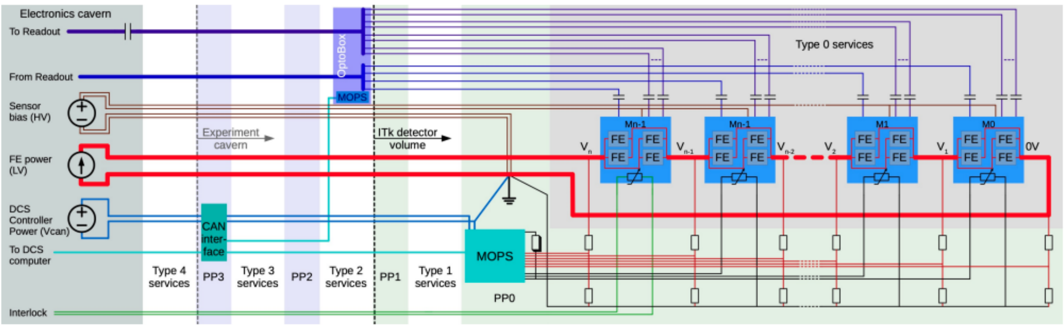


Figure 3.5: Schematic overview of the services in the ITk [32]. The red lines represent the serial powering chain where a MOPS chip is attached for monitoring.

The topology between the MOPS chip and the modules in the SP-chain shown in Figure 3.3 is elaborated in Figure 3.5. A MOPS chip requires a connection to the LV port and the module ground port of every single module in the SP chain. Additionally, the MOPS chip needs to be connected to the NTC lines from all modules of the SP chain except for the one connected to the interlock system [33].

The Control and Feedback path is a highly reliable system required during all operational phases, including commissioning, calibration, and data-taking [34]. Even when the detector is not actively running, the Control and Feedback system continues to monitor the state of the Pixel detector.

3.2.3 Diagnostic Path

The Diagnostic path of the DCS facilitates tuning the detector's performance by offering additional monitoring and debug information about the FE. Each pixel FE chip is equipped with an ADC to collect monitoring data. This data is then transmitted to the main DCS server via optical links connected to the Opto-box, as depicted in

Figure 3.3. The Opto-box is a custom mini-crate for housing the on detector part of the opto-electrical transceivers (Opto-boards) including ASICs for data handling [35].

The Opto-system, pivoted on the Opto-boards, plays a crucial role in this process, responsible for converting electrical monitoring signals from the pixel modules into optical signals. These optical signals are then transmitted to the off-detector Data Acquisition (DAQ) system, with the Front-End Link eXchange (FELIX) card serving as its central unit [36, 37].

The Diagnostic path operates with the highest granularity at the level of individual FE chips. During calibration periods, additional scans are performed to re-tune the FE for optimal data acquisition [16].

3.3 The Existing Monitoring Chip and its Limitation

The current DCS of the ATLAS experiment utilizes the CAN industrial field bus along with the CANopen protocol for its FE I/O whenever feasible and suitable [38]. CAN was specifically chosen by the CERN fieldbus working group due to its low cost, flexibility, and notably, its insensitivity to magnetic fields [39]. This configuration enables a variety of functionalities, ranging from monitoring environmental parameters like temperature and magnetic field to configuring and overseeing detector FE electronics and power supply control [40].

To facilitate slow-control and monitoring tasks for the LHC detector FE electronics, a multipurpose I/O and processing device called Embedded Local Monitor Board (ELMB) was developed [24]. The ELMB is a 50×67 mm^2 unit that can either be integrated into the electronics of detector elements or directly interface with sensors in a stand-alone manner. Using a built-in CAN controller, the ELMB can send data asynchronously to a FE system for environmental slow monitoring. Concerning the operation of the device in the harsh environment, the ELMB board provides a Total Ionizing Dose (TID) level of up 14 krad and a neutron fluency of $5 \times 10^{12}\,\mathrm{N_{eq}/cm^2}$ [40]. Moreover, the device contains components sensitive to a magnetic field up to 1.5 T [41].

While the ELMB has performed satisfactorily over the years, the need for a replacement has emerged due to several factors. The size of the board (50×67 mm^2) is too large to be integrated into the electronics of the new detector to monitor modules per serial power chain. This necessitates an increase in the number of services inside the detector volume.

Additionally, the new innermost layer of the new Pixel detector demands higher radiation tolerance levels of up to 500 Mrad. Despite the introduction of the more radiation tolerant ELMB2 [42], it falls short of fulfilling the radiation requirements set for the new Pixel detector. This makes ELMB very unsuitable for the monitoring granularity in the DCS of the new ITk, highlighting the need for further advancements in radiation tolerance technologies; therefore, a new DCS ASIC, called MOPS, was developed at the University of Wuppertal to fulfill the control and monitoring requirements of the new Pixel detector [43].

3.4 The new Monitoring Of Pixel System (MOPS)

The MOPS chip is an ASIC fabricated in 65 nm CMOS technology. It is foreseen in the DCS to monitor the voltages and temperatures of the detector modules and other sub-detector components independently. Using a built-in CAN controller that implement a CAN-open protocol, the chip can transfer the monitoring data of the detector modules. Due to restrictions of the utilized 65 nm technology of the MOPS ASIC, the voltage level of the physical layer of the CAN bus deviates from the CAN standard and is only 1.2 V [44].

Selected specifications of the MOPS chip are listed in Table 3.1.

Table 3.1: Specifications of the MOPS chip [45, 46].

Parameter	Specification
Technology	65 nm technology
chip size	2×2 mm^2
Package size	9×9 mm^2 QFN package
Communication protocol	CAN and CANopen protocol
Chip powering	(1.4 to 2) V
Voltage monitoring precision	15 mV
Temperature monitoring precision	1 K
Radiation hardness	500 Mrad
Operational temperature	(−40 to 60) °C

The small size of the chip (2×2 mm^2) allows reducing the number of services in the ATLAS detector by allowing local digitalization of the monitoring data. Specifically, the MOPS chip will be installed at two locations. Firstly, within the ITk volume itself in a location referred to as PP0/EOS to keep tabs on voltage and temperature for up to 16 FE modules in a single SP chain. Here, up to two MOPS chips will be connected per CAN bus. This requires 840 CAN bus connections to cover the whole new Pixel detector. The topology between the MOPS chip and the modules in the SP-chain is elaborated in Figure 3.5.

Secondly, MOPS chips will be allocated on the Opto-System to regulate temperatures and oversee the DC/DC converters placed on the power-board within the Opto-box. In this setting, up to four MOPS chips will be connected per CAN bus. This requires 64 CAN bus connections to cover the whole Pixel detector.

The powering and communication for this sub-system are completely independent of the SP-chains or the Opto-System, provided by an off-detector system called **MOPS-Hub**.

3.5 Integration of the MOPS Chip (MOPS-Hub)

MOPS-Hub is an FPGA-based interface designed to aggregate monitoring data between the MOPS chips and the DCS computers. The DCS will integrate up to 1200

MOPS chips to fulfill the functionality detailed in Section 3.4. To achieve this, MOPS chips will be connected via CAN bus interfaces and provided with the necessary power from the MOPS-Hub side. This topology ensures that monitoring capabilities persist even when the detector is not in standard operation mode. The actual CAN interface is implemented in MOPS-Hub firmware within the FPGA, while the non-standard 1.2 V physical layer is implemented in a hardware called CAN Interface Card (CIC). All hardware components are housed in what is called the MOPS-Hub crate. Each MOPS-Hub crate contains the necessary hardware modules to connect 32 CAN buses, with one FPGA handling 16 of these buses as depicted in Figure 3.6.

In addition to the actual monitoring data of the MOPS, MOPS-Hub provides monitoring information per CAN bus (voltage/current) and sends it to the DCS computer as part of the data stream. The voltage of each CAN bus is provided on the CIC and can be controlled separately from the MOPS-Hub FPGA logic.

The MOPS-Hub is designed modularly to fit into 19-inch racks, with each MOPS-Hub crate being 3U high (refer to Figure 3.6). These crates will be placed in racks on the walls of the ATLAS cavern, referred to as Patch Panel 3 (PP3), outside the vicinity of the detector modules. The required powering of the CAN busses and the FPGA relies on remote power supplies strategically placed in radiation-safe areas called USA15 and US15 in the counting room. The MOPS-Hub will distribute the CAN power supplied by the main power supply (VCAN-PSU) onto the individual MOPS. Additionally, it will supply the needed voltage for the FPGA, referred to as VPP3, which is completely independent of any CAN bus powering [47]. Data collected from the CAN buses are forwarded through low voltage differential signals called Electrical Links (eLinks) to a module called Embedded Monitoring and Control Interface (EMCI) [48], which is also located in each of the MOPS-Hub racks (refer to Figure 3.7). The EMCI combines all the received data into a single bidirectional channel and interfaces with an optical transceiver, which then transmits the data through an optical link to another FPGA based system called Embedded Monitoring Processor (EMP) board [49]. The EMP device will be placed in the counting room

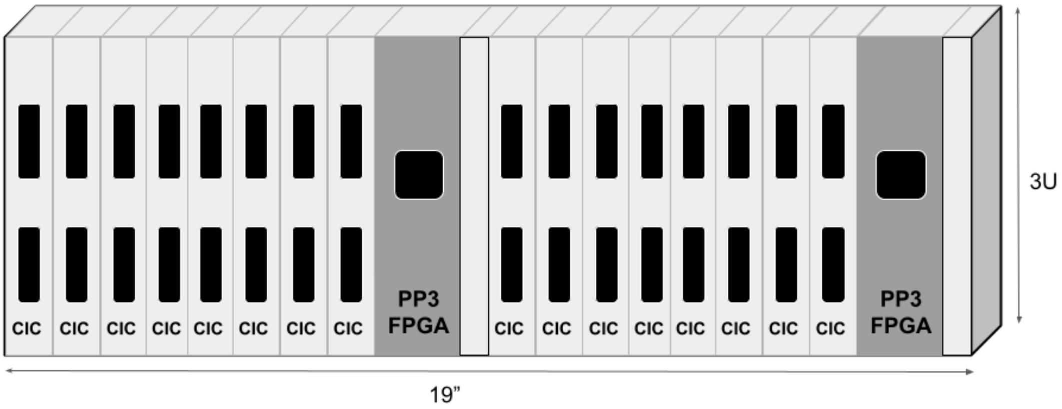


Figure 3.6: The MOPS-Hub crate [front view](not to scale).

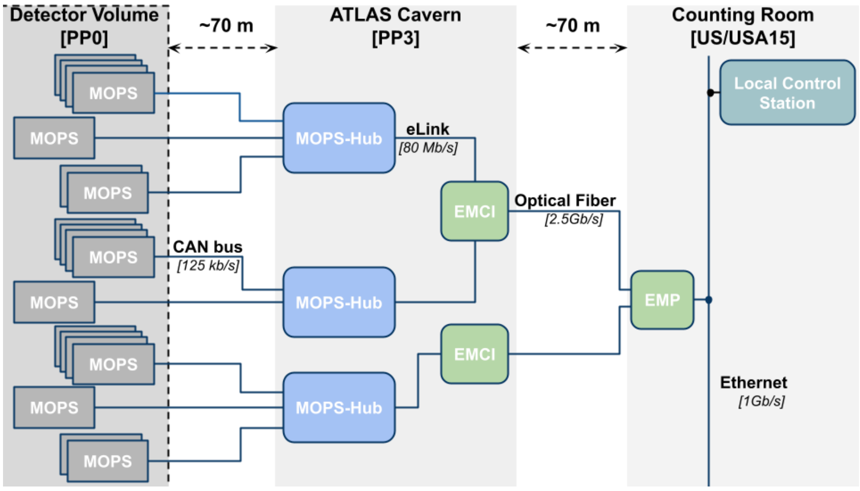


Figure 3.7: The complete MOPS-Hub network.

to drive up to 12 EMCIs and interface them to the DCS Ethernet network. Both EMCIs and the EMP are standard components of the ATLAS DCS.

Chapter 4

Hardware Requirements and Description of MOPS-Hub

This chapter examines the internal components of the MOPS-Hub crate, covering an overview of the ITk Pixel services and their requirements. It then explores the various modules within the MOPS-Hub crate, including the power module, key functions of the CIC, the PP3-FPGA module, and the EMCI/EMP chain.

4.1 ITK Pixel Services and Patch Panels

The operational and data collection equipment essential for the ITk Pixel detector, such as power supplies, readout systems, and the DCS, are connected to various detector modules and auxiliary components through the services provided by the ITk Pixel detector.

Figure 4.1 illustrates the routing of cables for data transmission, commands, monitoring, and power out of the detector volume, gathered at specific breakpoints known as Patch Panels (PPs). These PPs play a crucial role in facilitating the interface between various service types. This involves tasks like signal regrouping, cable mapping, and ensuring compliance with essential performance criteria, including grounding, shielding policies, and material safety requirements. Between the power supplies located in the service caverns US15/USA15 and the actual detector volume, there are as many as five patch panels in the power chain but not all of them are used by every system. The names applied to the patch panels used by the ITk system are as follows:

- Patch Panel 0 (PP0) (or in some cases called End Of Structure (EOS)): Several panels located at the detector, installed inside the primary Faraday Cage at the detector volume.
- Patch Panel 1 (PP1): located at the A and C sides of the primary Faraday Cage wall.
- Patch Panel 2 (PP2): located at several locations inside the Muon spectrometer.
- Patch Panel 3 (PP3): located at several locations inside the collider hall, on the walls of the ATLAS cavern.
- Patch Panel 4 (PP4): located at the power supply racks in US15/USA15.

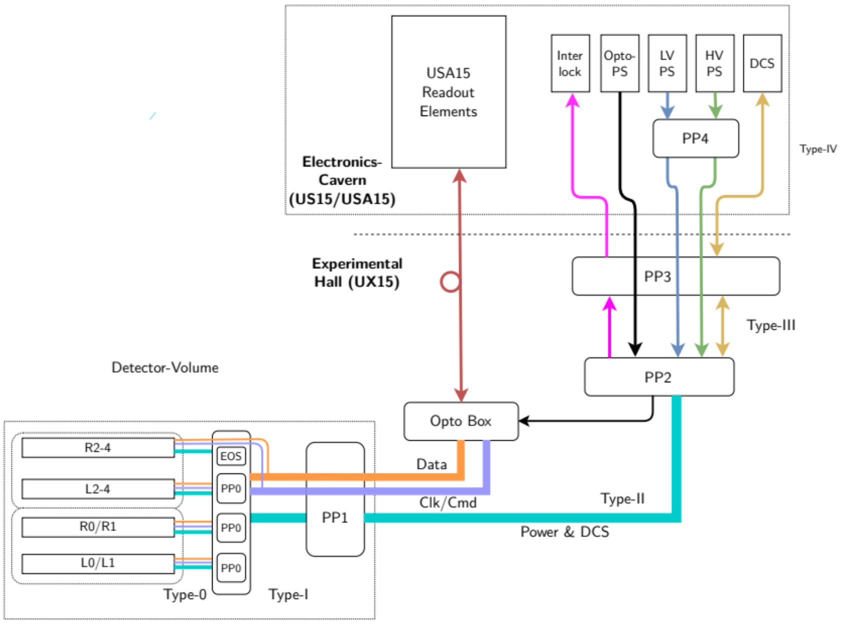


Figure 4.1: Overview of the ITk Pixel services and their breakpoints [50].

4.2 PP3 Requirements

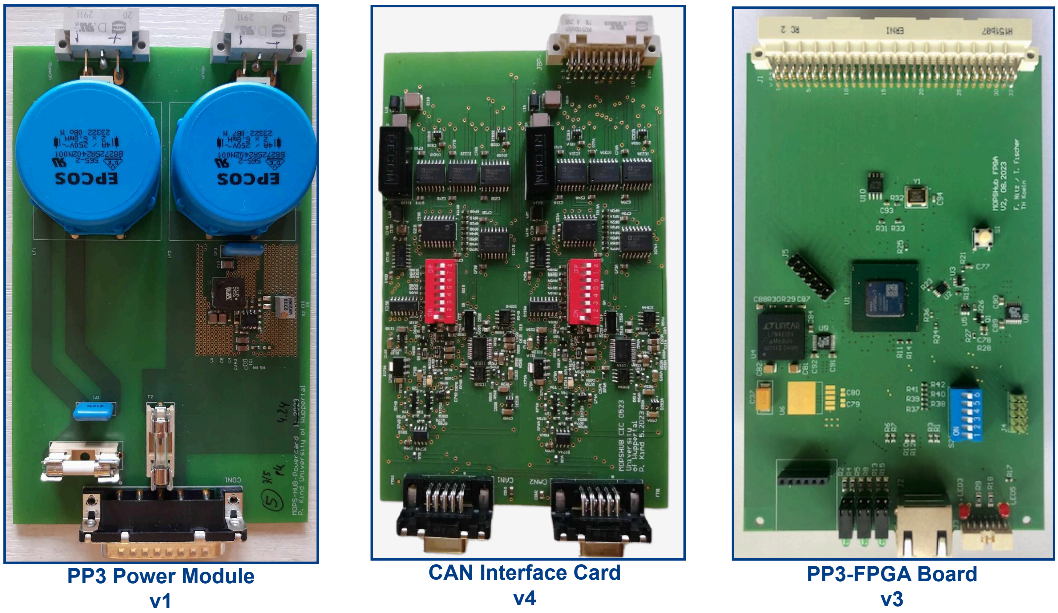
The MOPS-Hub will be positioned at an intermediary point between PP0 and the US15/USA15 locations, specifically at PP3, situated on the walls of the ATLAS cavern just outside the Muon spectrometer. This poses special requirements for the hardware components and firmware of the MOPS-Hub.

Figure 4.2 depicts the actual hardware components of each board discussed in the following sections. The PP3-Power module is detailed in Section 4.4, the CIC in Section 4.5, and the PP3-FPGA board in Section 4.6.

4.2.1 Power Requirements

The power supplies located at US15/USA15 transmit power to the FE systems via shielded bundles of twisted pair cables each has different thickness according to its purpose as detailed in Table 4.2. Breaking at a specific PP is necessary to transition between cable bundles or PCBs to cable bundles with larger wire diameters or to isolate other active components, such as the MOPS chip. All services between these PPs must comply with the ITk power requirements [51] as well as the grounding and shielding policy [52] discussed in Sections 4.2.2 and 4.2.3.

Table 4.1 presents the power requirements for various components at PP3 location.



PP3 Power Module v1

CAN Interface Card v4

PP3-FPGA Board v3

Figure 4.2: The MOPS-Hub hardware components (not to scale). Each component measures 100 × 160 mm^2.

Table 4.1: Power requirements at PP3 location[47].

	Max/Min Voltage	**Max Current**
VCAN on-detector	(3.6 to 3.0) V	70 mA
VCAN opto-box	(3.4 to 2.9) V	140 mA
VCANPSU	35 V/30 V	1 A
VPP3	35 V/30 V	2 A

The voltage supplied by the MOPS-Hub to individual CAN buses is represented by VCAN. Conversely, VCAN-PSU and VPP3 denote the voltages at the input of the MOPS-Hub crate.

4.2.2 Cable Resistances

The ITk Pixel detector's services are categorized into several types based on their function and location, such as Type-0, Type-I, Type-II, Type-III, and Type-IV, each connecting different parts of the detector and the service caverns. Services running along the local PPs are listed in Table 4.2. It's important to note that cable resistance directly impacts voltage drops along the cable length, which can significantly affect the performance of hardware components located at different PPs. Higher cable resistance results in greater voltage drops, potentially leading to reduced voltage levels at the endpoints of the cables. Therefore, VCAN comprises the voltage required at the input of the MOPS chips plus the voltage drop on the services connecting the MOPS chips

Table 4.2: Expected cable lengths and diameters for ITk pixel services [33, 53, 54].

Type	Min Length [m]	Max Length [m]	AWG	Ω/km	Rmin [Ω]	Rmax [Ω]
Type-I	2	6	32	556	1.112	3.336
Type-II	9.3	15.1	26	129	1.200	1.948
Type-III	25.6	69.3	24	86.8	2.222	6.015
Round trip, incl. "twisted pair factor" of 1.2					10.88	27.12
OB VCAN Lines						
Type-II/Type-III	21.6	62.2	24	86.8	1.875	5.399
Round trip, incl. "twisted pair factor" of 1.2					4.500	12.96
VCAN-PSU and VPP3 Lines						
Type-IV	20.5	80.4	18	21.6	0.443	1.737
Round trip, incl. "twisted pair factor" of 1.2					1.063	4.17

to the MOPS-Hub crate. Similarly, when applying power from the power supply in US15/USA15 to the components at PP3, it's essential to consider the voltage drop on the Type-IV services. This information is crucial for planning the layout of hardware components and ensuring that the voltage requirements are met throughout the system.

4.2.3 ITk Grounding and Shielding

The ITk grounding and shielding strategy is designed to protect the sensitive signals within the ITk sensors and readout electronics. It also aims to shield transmission signals from Electromagnetic Interference (EMI) originating externally or generated internally due to undesired coupling between signal sources [52]. This significantly influences the hardware designs and power supply strategy for the components of the MOPS-Hub as the active components of the MOPS-Hub crate, positioned at PP3, has a reference ground known as **PP3 GND**. Additionally, it powers the MOPS chips, located in a separate area, PP0, tied to the ITk reference ground, known as **ATLAS GND** [55]. This dual grounding arrangement is regulated within the MOPS-Hub system using galvanic isolation.

Figure 4.3 depicts the connection diagram for the MOPS chip situated at PP0. As depicted, the chip reference is positioned as close as feasible to the ITk reference, which is established by the Faraday cage. This connection serves as the sole earth connection for the entire ITk [52].

The system design entails one MOPS chip per SP, as detailed in Section 3.2.2. Each MOPS chip requires a power line (Power lines for individual CAN bus (VCAN)) and a pair of data lines (CAN High (CANH) and CAN Low (CANL)). The CANH and CANL lines, responsible for CAN communication, are directed to the MOPS-Hub situated at PP3 using TwinAx Type-II cables. These cables are AC coupled to the ITk reference through their shields to diminish emission in the service cable[1]. The VCAN return line is also connected to the SP ground at PP0 which is tied to the ITk reference at PP1.

[1] A 10 nF filtering capacitor is added between the power return and the shield [52].

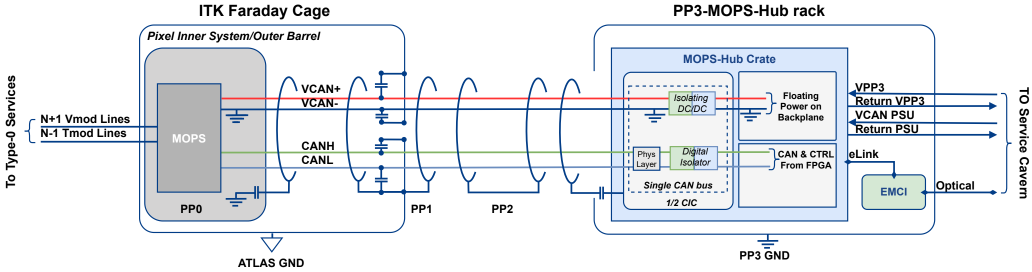


Figure 4.3: Connection diagram for the MOPS chip located at PP0 to PP3.

The Type-II cables are passively mapped and regrouped at Patch Panel 2 (PP2) into Type-III cables to PP3. At PP3, CANH/CANL lines undergo galvanic isolation from other CAN buses using digital isolation integrated into the CIC[2]. This isolation is crucial for safeguarding against voltage differences between the CAN signals and the FPGA, a concept elaborated in Section 4.5.1.

Furthermore, the VCAN power lines are equipped with separate isolated DC/DC converters integrated into the CIC[3]. These DC/DC converters serve the primary function of distributing power from the VCAN-PSU to the individual CAN buses according to the requirements detailed in Table 4.1.

The PP3-FPGA of the MOPS-Hub has a separate power line, VPP3, independent of the CAN bus power scheme. Where, both VCAN-PSU and VPP3 draw power directly from the sources located in US15/USA15 and have their own return line from PP3 [51].

To reduce the number of Type-IV services, each PP3- FPGA in MOPS-Hub will merge the data of at least 16 CAN buses onto one eLink data line routed to the EMCI, also located in the MOPS-Hub rack, as detailed in Section 4.7. The endcap and Optopanel topologies may exhibit slight variations from the ones depicted in Figures 3.5 and 4.3. However, the structural layout and isolation configurations of MOPS-Hub remain consistent across all setups.

4.2.4 Radiation Background

Despite the location of PP3 outside the ATLAS detector, positioned on the walls of the ATLAS cavern, it still experiences radiation levels.

Table 4.3 summarizes the expected radiation level at the walls based on the simulation results detailed in Section A.1 [58, 59]. All numbers were taken for the area of r between (1100 to 1200) cm and z between (0 to 1300) cm, where the PP3 racks will be installed.

In the original ATLAS design studies, 'safety factors' were introduced to reflect the uncertainties in simulating radiation backgrounds. Here, a SF of 3 is applied to the simulated radiation levels for TID, $\Phi_{\mathrm{eq}}^{\mathrm{Si}}$ and Φ_{20}^{Had} .

[2] The ADUM3402 digital isolator is utilized [56].

[3] The RSO-2405SZ DC/DC converter is utilized [57].

Table 4.3: Expected radiation on the walls of the ATLAS cavern as extracted from Figures A.1 and A.1(a) [58, 59]. A safety factor of 3 is applied to the simulated radiation levels.

Parameter	Expected Dose	Expected Dose with SF
TID	($\approx 30\,\mathrm{Gy}$)	($\approx 90\,\mathrm{Gy}$)
Neutron fluence ($\Phi_{\mathrm{eq}}^{\mathrm{Si}}$)	$5 \times 10^{11}\,\mathrm{N_{eq}/cm^2}$	$1.5 \times 10^{12}\,\mathrm{N_{eq}/cm^2}$
Hadron fluence >20 MeV (Φ_{20}^{Had})	$2 \times 10^{-7}\,\mathrm{cm^2/pp}$	$6 \times 10^{-7}\,\mathrm{cm^2/pp}$

4.2.5 Magnetic Field

The ATLAS detector features a hybrid system of four superconducting magnets: a Central Solenoid surrounded by 2 End-cap Toroids and a Barrel Toroid. The Inner tracker itself is partially enclosed by a superconducting solenoid which is 5.5 t in weight, 2.5 m in diameter and 5.3 m in length [60].

The ATLAS solenoid contributes with the highest magnetic field strength of ≈ 2 T near the central tracking volume of the ATLAS detector for measurement of charged track momentum [61]. This high magnetic field can influence devices that have magnetic cores, even if they are shielded. For example, a coil with a ferrite or iron core might experience a shift in its behavior due to the surrounding magnetic field. Depending on the strength and proximity of the magnetic field, this shift can push the device into or further into saturation of the B-H Curve, thereby altering its intended performance.

The PP3 racks positioned on the walls of the ATLAS cavern with radial distances between (11 to 14) m. Based on the simulation map detailed in Section A.2, this location will experience a magnetic field of ≈ 0.1 T that necessitate special requirements for the hardware components at PP3.

4.3 Communication Interfaces

The PP3-FPGA employs several communication protocols to manage tasks across different interfaces within PP3. Firstly, it uses a well-defined CAN protocol to communicate with the MOPS chips over CAN buses. Secondly, data collected from various CAN interfaces within the PP3-FPGA is transmitted to the EMCI via low-voltage differential signals known as eLinks. Lastly, the MOPS-Hub utilizes an SPI interface to handle monitoring information[4] for each CAN module of the CIC and facilitates control of individual power lines within the system. This section will discuss two of these communications, CAN and eLinks, explaining their main purposes and features of usage.

[4] Monitoring information includes voltage, current and temperature

4.3.1 Controller Area Network (CAN)

The CAN bus is a serial communication bus where all the nodes are connected to a single bus terminated by a termination resistor at both ends without a master. This topology allows all nodes to listen to every message on the bus and determine whether or not it is relevant to them.

The CAN protocol is defined only for the physical and data link layers of the Open Systems Interconnection model (OSI) model. After being created initially for automotive applications, CAN is being utilized in numerous different settings, including industrial automation.

This section gives a summary of the fundamentals of the CAN protocol as well as the current industry standard CANopen.

CAN Bus Communication Protocol

A CAN bus requires only four lines in total. Two lines are needed for data transmission, and two more are needed for powering. As depicted in Figure 4.4, the data transmission lines use a differential AND-signal, CANH and CANL, which are driven to either a dominant (logical ‘0’, typical 2 V differential voltage) or a recessive (logical ‘1’, typical 0 V differential voltage) state [62]. In order to prevent signal reflections, the two data wires are utilized in twisted pairs with a particular termination resistor, usually 120 Ω, at both ends.

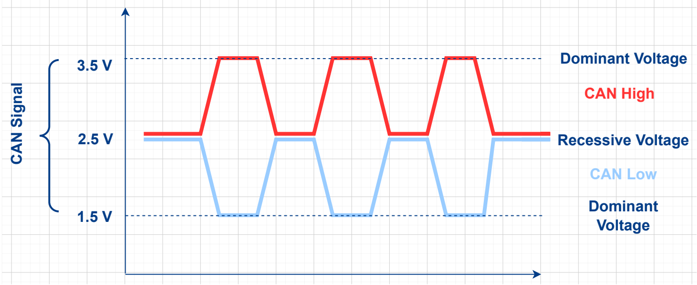


Figure 4.4: Levels of the differential AND-signal with CANH (red) & CANL (light blue) either driven to a dominant or recessive CAN state.

The CAN protocol itself does not explicitly limit the number of nodes that can be connected to a CAN network. However, several practical factors determine the maximum number of nodes that can be reliably connected. These factors include bus length, bit rate, and electrical characteristics of the transceivers and cables used.

CAN Bus Parametrization

Bus parameterization in a CAN network involves configuring several critical parameters that define the controller’s behavior and performance. As depicted in Figure 4.5, every bit is split into four time-based segments for this purpose.

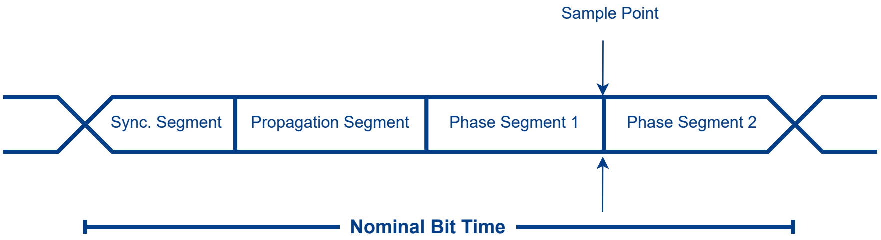


Figure 4.5: CAN communication protocol: Bit Timing Segments.

The proper configuration of CAN parameters ensures reliable communication within the CAN network. Key elements include:

- **Bit Rate:** The number of bits transmitted per second (bps). Typical values are 125 kbit/s, 250 kbit/s, 500 kbit/s, and 1 Mbit/s.
- **Time Quanta (TQ):** The smallest time unit in a bit time, derived from the CAN controller clock. The bit time is divided into multiple time quanta. The typical number of TQs per bit time is around 16 TQs.
- **Bit Timing Segments:**
  - **Synchronization Segment:** Used for bit synchronization to ensure that all nodes start bit transmission at the same clock edge.
  - **Propagation Segment:** Compensates for propagation delays. It allows nodes to adjust for varying signal transmission times across the network.
  - **Phase Buffer Segment 1 (Phase-Seg1):** Compensates for edge phase errors.
  - **Phase Buffer Segment 2 (Phase-Seg2):** Further compensates for edge phase errors.
- **Sample Point:** The point in time at which the bus level is read and interpreted as the value of the bit[5].
- **Synchronization Jump Width (SJW)**: Number of time quanta by which the sample point is shifted to compensate for the frequency mismatch between different nodes on the bus. It can shorten or lengthen the total bit time. Typically 1-4 TQ.

These parameters must be configured properly to enable effective and dependable communication within the CAN network. In the PP3-FPGA, for example, the instantiated CAN controllers within the firmware are configured to match the MOPS chip Bit Rate, nominal 125 kbit/s (refer to Section 7.3.2).

[5] The sample Point is defined as a percentage of the bit time

CAN Frame

There are two versions of the CAN frame structure: the standard and the extended. Since there is not much of a difference between the two types, just the standard version shown in Figure 4.6 is covered with its specific fields in the following.

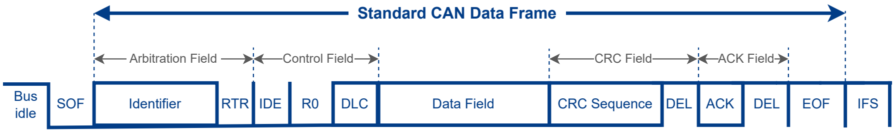


Figure 4.6: Standard CAN data frame with the various protocol-specific fields.

- **SOF**: This bit is always transmitted as a dominant bit. It serves the synchronization of every receiver to indicate that the bus is used.
- **Identifier (ID):** 11-bit address. The transmitter does not actually address the receiver, but the type of message that is sent. CAN 2.0B implements 29-bit addresses but as this field size is not modified, the remaining address bits are transmitted within the IDE field.
- **Remote Transmission Request (RTR):** Request to send messages. An Remote Transmission Request (RTR) packet does not send data but requests one participant of the bus to do so. The data field is therefore empty, but the DLC field transmits the length of the requested data.
- **Identifier Extension (IDE):** This field is used for the remaining address information and marks distinction between the standard and the extended data format.
- **R0:** Reserved for possible future use.
- **Data Length Code (DLC):** This field transmits the length of the upcoming data.
- **Data Field:** Within this field, the application data is transmitted. The field can have a flexible size between 0 bytes and 8 bytes.
- **Cyclic Redundancy Check (CRC):** 15-bit checksum to verify the correctness of the received data.
- **DEL (CRC):** also called Cyclic Redundancy Check (CRC) Delimiter bit. Its purpose is to separate the Cyclic Redundancy Check (CRC) field from the next field.
- **Acknowledgement (ACK):** Acknowledgment of the receiver of the message. This bit is first transmitted recessively and then overwritten by the receiver with a dominant bit if the frame is received correctly.

- **DEL (ACK):** also called Acknowledgement (ACK) Delimiter bit. Its purpose is to separate the Acknowledgement (ACK) field from the next field.
- **End-Of-Frame (EOF):** Eleven recessive bits. This sequence intentionally violates the rule related to the bit stuffing to indicate the end of a frame.
- **Inter Frame Spacing (IFS):** The bus is left empty for 3 µs until the next transmission can start.

MOPS Data Frame Structure

The MOPS chip utilizes the expedited transfer mode of CANopen standard's Service Data Objectss (SDOs), allowing up to four bytes of data to be transmitted in a single CAN message. The additional four bytes in the message carry vital information such as index, sub-index, and SDO operation specifics. Different type of SDOs frames used for communication with the MOPS chip are outlined in [44]. A typical SDO frame, as depicted in Figure 4.7, includes Bytes 4-7 for ADC data, with Byte 0, the Command Byte, containing command bits as per CANopen standards [62].

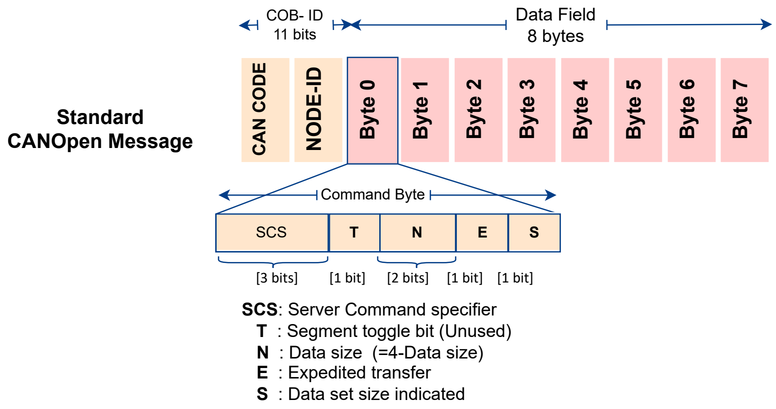


Figure 4.7: SDO data frame structure according to CANopen standards.

Table 4.4 describes the command byte configurations for various operations.

Command Byte bits				Command Byte	Description
SCS	N	**E**	**S**		
000_b	0	0	0	$0x40_h$	SDO read request to the MOPS chip.
010_b	0	1	1	$0x43_h$	MOPS response to a SDO read request.
001_b	0	1	1	$0x23_h$	SDO write request to the MOPS (4 bytes sent).
011_b	0	0	0	$0x60_h$	MOPS response to a SDO write request.
100_b	0	0	0	$0x80_h$	SDO abort message in case of an error.

Table 4.4: Command code bits description according to CANopen standard as specified for the MOPS chip [44].

Understanding the CAN frame structure for the MOPS chip is key to reconstructing CAN messages within the PP3-FPGA, as detailed in Section 7.3.4

4.3.2 eLinks

Interconnections between the PP3-FPGA and the EMCI device is made through differential transmission lines, called eLinks. The eLink is a low-power electrical line designed by CERN for chip-to-chip communication specifically made for high radiation environments. The signaling adopted through eLinks is defined by an ad-hoc "standard" called the CERN Low Power Signaling (CLPS) [63]. Depending on the data rate and transmission media used, eLinks allow connections that can extend up to a few meters. This standard defines a nominal common mode voltage (V_{cm}) of 600 mV, with a differential voltage amplitude (V_{PP}) ranging from 200 mV to 800 mV and calculated according to Equation 4.1.

$$V_{\text{PP}} = \max(V_{\text{out}+} - V_{\text{out-}}) - \min(V_{\text{out}+} - V_{\text{out-}}) \tag{4.1}$$

Additionally, the standard includes a 100 Ω termination resistor at the receiving end of a connection and recommends the use of AC coupling for DC common mode voltage and noise rejection [64]. More details about the differential standard for eLink communication adopted in the PP3-FPGA can be found in [65].

The actual physical connector for the eLink on the PP3-FPGA is defined using a RJ-45 connector, which is commonly used for Ethernet cables and systems. The interface consists of three individual signals: two unidirectional data lines (Tx and Rx) and a clock (Clk) line for timing. From the perspective of the PP3-FPGA, the Clk and Rx signals are inputs provided from the EMCI interface, while the Tx signal is an output.

8B10B Coding

The 8B10B encoding/decoding scheme is crucial for maintaining DC balance and ensuring bounded disparity within the eLink data stream. This scheme converts each 8 bit word of data into a 10 bit symbol, with an equal number of ones and zeros and a maximum run length[6] of 5 [66].

Figure 4.8 depicts the bidirectional conversion process.

In the bidirectional conversion process depicted in Figure 4.8, the eight input bits (A to H) are divided into two groups: a 5 bit group (A to E) and a 3 bit group (F to H). Similarly, the coded bits (a to j) are split into a 6 bit group (a to i) and a 4 bit group (f to j).

To achieve a DC code, the encoder keeps track of the difference between the number of ones and zeros in the encoded word, known as the Running Disparity (RD). The code restricts the RD so that it is always 1 or −1 at the end of each code word. This ensures neutral average disparity during encoding [66].

[6] The number of consecutive zeroes or ones in the encoded data stream

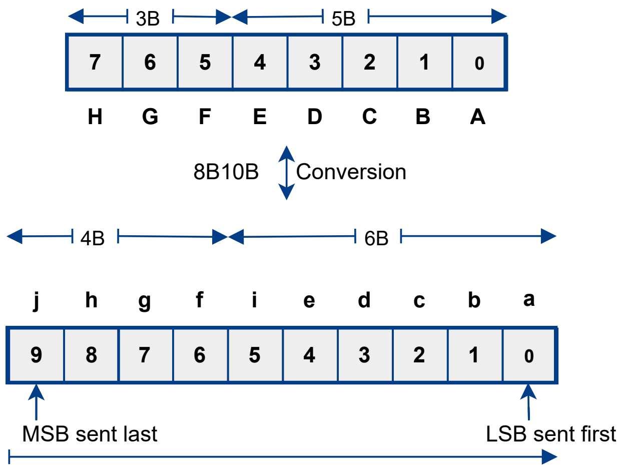


Figure 4.8: 8B10B Conversion.

- Neutral disparity indicates an equal number of ones and zeros.
- Positive disparity indicates more ones than zeros.
- Negative disparity indicates more zeros than ones.

To maintain neutral average disparity, a positive RD must always be followed by neutral or negative disparity, and a negative RD must be followed by neutral or positive disparity.

In addition to 2^8 data characters, the 8B10B code defines twelve special control words called *K-characters*. The 2^8 data characters are named $D_{x.y}$, and the special control characters are named $K_{x.y}$. The x value corresponds to the 5 bit group, and the y value to the 3 bit group [66]. The special control characters indicate, for example, whether the data is idle, data, or data delimiters. More detailed information regarding the *K-characters* can be found in the reference [66].

4.4 PP3-Power Module

The power distribution of the MOPS-Hub crate relies on two 100 × 160 mm^2 PP3-Power modules, situated at the back side of the MOPS-Hub crate, depicted in Figure 3.6.

Each PP3-Power module is tasked with receiving power from the VCAN-PSU and VPP3 supply lines and delivering the necessary power to the PP3-FPGA (5 V) and the CICs (9 to 36) V.

The module uses a DC/DC converter[7] to bring down the input voltage VPP3 (up to 40 V) into an adjustable output voltage V_{FPGA}(5 V). It suppress short noise signals and filter out high frequency noise using a large filter[8] at its outputs.

The output voltage of the DC/DC converter is adjustable within the range of (0.8 to 5.5) V, with a maximum continuous output current of 5 A.

The selection of the DC/DC converter circuit was driven by the specifications required for the PP3 location with a robust power supply capabilities, as outlined in the specification document of the power supply system of the ATLAS ITk Pixel detector [51].

In the quest to understand and evaluate the performance characteristics of the PP3-Power module, Section 8.2 is dedicated to outlining the comprehensive procedures undertaken to ensure the reliability and compliance of this module.

4.5 CAN Interface Card

The CAN Interface Card (CIC) is an interface unit within the MOPS-Hub crate, facilitating communication, power distribution, and monitoring functions crucial for the overall operation of the system. It interfaces the CAN-RX and CAN-TX signals of the FPGA card into a CANH and CANL signal with 1.2 V level (low level CAN-Bus), compatible to the CAN interface of the MOPS [46]. Additionally, it provides the power lines (VCAN) for powering the MOPS, controls the power (ON/OFF, voltage setting), and monitors crucial parameters such as the supply voltage and current for the MOPS, along with the surface temperature of the CIC.

Each MOPS-Hub crate is equipped with two groups of 8 CICs, each group is handled independently by a separate PP3-FPGA. The CIC, depicted in Figure 4.2, measures 100×160 mm^2 and is designed as a PCB with two layers, employing halogen-free materials .

To ensure robustness against radiation, semiconductor devices on the CIC are carefully selected for their radiation tolerance. Many of these components have been extensively tested and have demonstrated reliability in other experiments, as documented in the MOPS-Hub construction manual [47]. The following sub-sections provide an in-depth look into the internal components of the module and its workings principle.

4.5.1 CAN Interface

The CIC provides the CAN physical layer to the CAN interface and transmits the signals CAN-RX and CAN-TX of the FPGA. Two bus systems are implemented on one CIC, which are completely independent from each other except for the common supply by the backplane. Each system therefore has a separate isolated DC/DC-

[7] AP64500SP-13 DC/DC converter is used [67].

[8] EPCOS-B82722A power line choke is used [68].

conversion of the supply voltage, separate physical layers for CAN communication and separate bus lines (A and B).

As depicted in Figure 4.9, the CIC uses a galvanic isolation[9] between the CAN signals and the FPGA. This isolation is essential to fulfill the ITk grounding and shielding specifications at PP3 as detailed in Section 4.2.3.

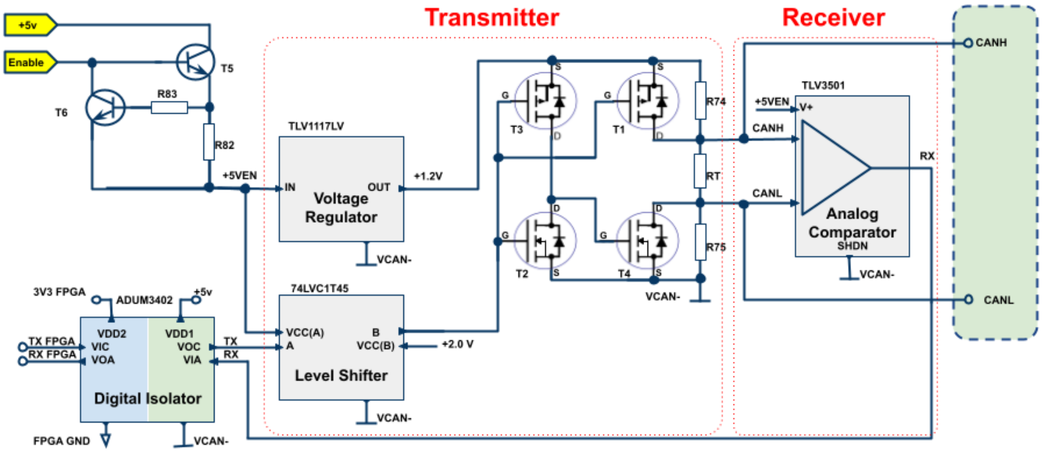


Figure 4.9: Simplified block diagram of the CIC modulation of a bus system.

Two NPN transistors (T5 and T6) act as power switches for the transmitter section of the transceiver circuit. One of those transistors (T6), along with two resistors (R82 and R83), forms a current limiter circuit. Together with T5, it ensures that, if the MOPS power supply is off, both CANH and CANL are forced to 0 V for system safety. The mechanism of this circuit is detailed in [47]. In order to adapt CANH and CANL signals from the standard voltage levels (5 V or 3.3 V) to the operation voltage of the MOPS (1.2 V), a LDO regulator[10] is used. The enable signal *+5VEN* is used again to supply voltage to the LDO which as a result gives an output of 1.2 V.

All these components represent the physical layer for CAN communication, which is responsible for providing the electrical, mechanical, and procedural interface to the transmission medium.

The Transmitter part

The transmitter part of the physical layer utilizes two NMOS and two PMOS transistors , along with a level shifter[11] to convert the TX signal from the 5 V level to a lower voltage for the MOS transistor gates.

In the case of a recessive state (high) on TX, then PMOS transistor T1 is turned off. Transistors T2 and T3 form a CMOS inverter circuit, resulting in an inverted

[9] ADUM3402 digital isolator is used[56].
[10] TLV1117LV LDO regulator is used[69].
[11] 74LVC1T45 or 74LXC1T45 level shifter is used [70].

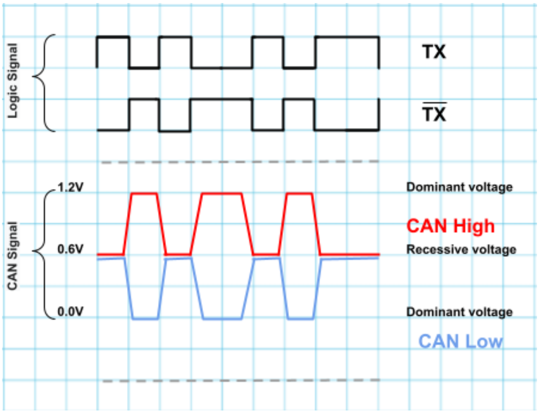


Figure 4.10: CAN bus signal vs TX transitions.

TX signal that pulls the base of NMOS transistor T4 low, also turning it off (refer to Figure 4.10). Consequently, CANH and CANL reach a recessive voltage of approximately 0.6 V, determined by the voltage divider consisting of resistors R74, RT, T75 (where RT = 100 Ω line termination, R74, R75 = 10 kΩ).

When TX transitions to a dominant state (low), PMOS transistor T1 turns on, pulling CANH to 1.2 V. Simultaneously, NMOS transistor T4 is turned on by the inverted TX signal, pulling CANL to 0 V, This configuration represents the dominant state on the CAN bus.

The Receiving part

The receiving part of the physical layer employs a high speed comparator[12] to monitor the voltage difference between CANH and CANL signals. This comparator, along with an additional resistor network (not shown in Figure 4.9), determines the state of the RX output.

If both CANH and CANL are at nearly the same voltage (e.g., approximately 0.6 V representing the recessive state), the comparator output RX is high. Otherwise, if the voltage difference (CANH- CANL) exceeds approximately 470 mV (e.g., CANH = 835 mV, CANL = 365 mV), RX transitions to a low state (representing the dominant state).

4.5.2 CAN Bus Control

The main VCAN-PSU from the PP3-Power module discussed in section 4.4, is responsible for providing supply voltages ranging from (9 to 36) V to the isolating DC/DC converter[13] on the CIC module. This DC/DC converter transforms the input voltage to the required output voltage, V_{out}, set at 5 V. Subsequently, the V_{out} signal serves

[12] TLV3501 comparator is used[71].

[13] RSO-2405SZ DC/DC is used [57].

as the supply voltage to the LDO[14] depicted in Figure 4.11. To enable control over the individual VCAN buses on the CIC module, an 8 bit register chip is employed to manage the enable signal (SHDN) of the LDO. The register chip[15] can be controlled via SPI communication from the digital logic of the FPGA or any other system that can provide a SPI interface. The register chip serves to preserve the status of VCAN, ensuring that even if the external system, such as the FPGA, undergoes a power cycle for any reason, the status of VCAN remains unchanged.

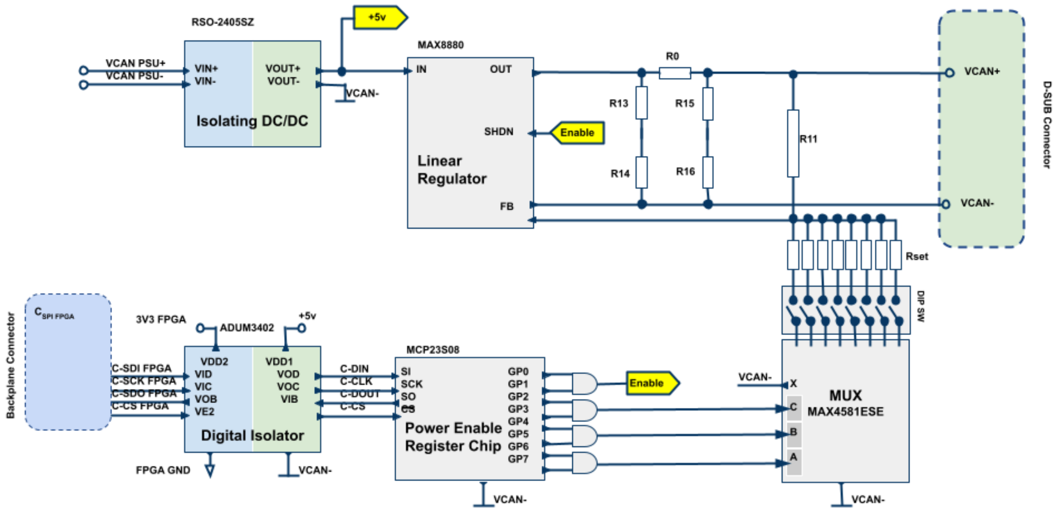


Figure 4.11: Simplified block diagram of the CAN-bus control schematics at the CIC.

VCAN level control

The control over the output voltage level VCAN is achieved using eight fixed reference resistors, allowing for the selection of eight fixed voltages based on the reference resistor set at the output (R_{set}). The selection of these voltages is managed by an 8-channel multiplexer [16], controlled by configuration bits A, B, and C provided by the register chip. Together, with the help of **R11** resistor ($= 47\,\text{k}\Omega$), the value of VCAN is set based on equation 4.2.

$$VCAN = V_{\text{FB}} \times \left(\frac{R_{\text{set}} + R11}{R_{\text{set}}} \right) \tag{4.2}$$

[14] MAX8880/MAX8881 LDO is used[72]

[15] MCP23S08 register chip is used [73].

[16] MAX4581ESE 8-channel multiplexer is used[74].

Where V_{FB} represents the feedback voltage compared in the LDO to its internal reference voltage V_{ref} (=1.257 V). The LDO regulates its output voltage to achieve $V_{\mathrm{FB}} = V_{\mathrm{ref}}$.

The 8-channel multiplexer utilizes 3 address bits (A, B and C), each of which comes from as a dual AND-logic. Additionally, the *SHDN* signal for the LDO, responsible for output voltage on/off, is generated as the AND (signal P (Enable)) of two register bits. (refer to Figure 4.11). This configuration enhances system robustness against SEUs, as any bit-flip results in a lower (and thus harmless) VCAN value.

To ensure safe powering over varying cable lengths with different voltage drops to the connected MOPS, an extra 8 bit DIP switch is integrated into the CIC. This switch allows the user to disable specific resistors R_{set} to adjust the delivered VCAN. For instance, when using very short cables where VCAN must never exceed 2 V, the uppermost 6 switches are set to off. In the event of an error where a high voltage R_{set} is selected for long cable voltage drops, the disabled R_{set} results in a VCAN value equivalent to V_{ref}=1.257 V.

4.5.3 CAN bus Monitoring

The monitoring of supply voltage and current for each CAN bus is facilitated by a dedicated ADC[17] with 16-bit resolution and 4 differential voltage inputs.

Control of the ADC is achieved through SPI communication, managed either by the digital logic of the FPGA or any other external system equipped with an SPI interface. Notably, the SPI bus of the ADC for monitoring purposes (M-SPI) operates independently from the SPI bus of the Power control register (C-SPI).

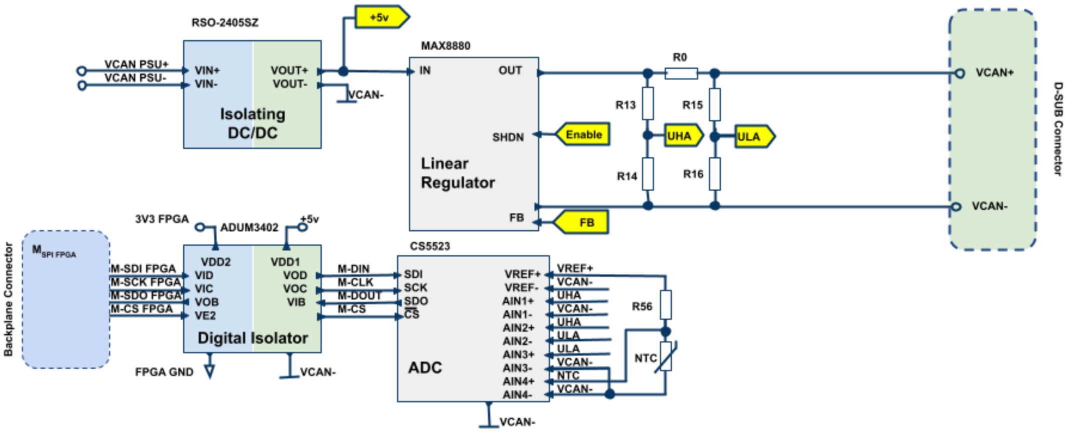


Figure 4.12: Simplified block diagram of the CAN-bus monitoring schematics at the CIC.

In Figure 4.12, the ADC utilizes a voltage of 2.5 V between its *VREF+* and *VREF-* pins. To measure VCAN voltages (VCAN+ to VCAN-) up to 5 V, the voltages

[17] CS5523 ADC is used [75].

UHA and UHL measured by the ADC are obtained across two voltage dividers, each consisting of two $10\,\mathrm{k}\Omega$ resistors (R13, R14) and (R15, R16), respectively, which scale them down by a factor of 2. Consequently, VCAN is calculated as $\mathrm{VCAN} = 2 \times \mathrm{ULA}$. The supply current provided to each CAN bus is determined by measuring the voltage drop across R0 ($1\,\Omega$). With the scaling factor of 2 taken into account, the current (IMON) is calculated using Equation 4.3:

$$\mathrm{I_{MON}} = \frac{(\mathrm{UHA} - \mathrm{UHL})}{1\,\Omega} \times 2 \tag{4.3}$$

4.6 PP3-FPGA Module

The PP3-FPGA board, depicted in Figure 4.2 measures $100 \times 160\ \mathrm{mm}^2$ and designed as a PCB with six layers, employing halogen-free materials. The board is designed by a team in the Technische Hochschule Köln [65]. More detailed information on the PP3-FPGA board can be found in [47].

The primary function of the PP3-FPGA board is to facilitate communication with the EMCI/EMP chain via the eLink connection, enabling control and read-back of up to 16 CAN buses. Additionally, it provides two separate SPI buses for monitoring and controlling functions of each CIC.

The PP3-FPGA interfaces with up to eight CICs (16 CAN buses) via the backplane. Each CIC comprises two TX/RX signal pairs for the CAN buses and four chip select signals for the SPI buses.

Additionally, the PP3-FPGA board provides multiple auxiliary I/O ports and a 6-position switch for lab testing.

The board utilizes a supervisory Watchdog IC[18] to hold the FPGA in reset as long as the supply voltage is insufficient [76]. After a normal power-up sequence of the PP3-FPGA board, the FPGA is monitored by the watchdog. If the FPGA does not send a heartbeat within 0.8 s, the watchdog resets the FPGA.

The semiconductor devices on the PP3-FPGA are chosen for their radiation tolerance, with many having a proven track record in other experiments [47].

Several studies [77–81] have provided comprehensive reviews of radiation-induced effects in modern FPGAs and associated mitigation strategies. These reviews serve as guidelines for selecting the Artix-7 family FPGA (**XC7A200T**) for the PP3-FPGA board.

Additionally, performance studies on the chosen FPGA (**XC7A200T**) have been conducted at two neutron facilities: LANSCE, USA (beam energies up to 800 MeV, maximum fluence $3.15 \times 10^{11}\ \mathrm{n/cm^2}$) and NCSR,Greece (maximum energy 20 MeV, maximum fluence $2.83 \times 10^{10}\ \mathrm{n/cm^2}$) [82, 83]. All data are summarized in Table 4.5.

[18] TPS3306 Watchdog/Supervisory IC is used [76]

Table 4.5: Experimental beam testing and real-time SER for CRAM-FPGA.

Resource	Cross-section	Facility (Beam)
Artix-7 [81]	$6.99 \times 10^{-15}\,\mathrm{cm}^2/\mathrm{bit}$	LANSCE (Neutron)
XC7A200TFFG1156 [82]	$2.53 \times 10^{-11}\,\mathrm{cm}^2/\mathrm{n}$	LANSCE(Neutron)
XC7A200TFFG1156	$3.018 \times 10^{-10}\,\mathrm{cm}^2/\mathrm{n}$	NCSR(Neutron)

4.7 EMCI/EMP Chain

The EMCI/EMP interface serves as a system aggregator, facilitating communication between PP3, where the MOPS-Hub is located, and the US15/USA15. The decision to utilize this system is driven by various factors, including the necessity for hardware components with radiation tolerance and resilience to magnetic fields, as detailed in [49].

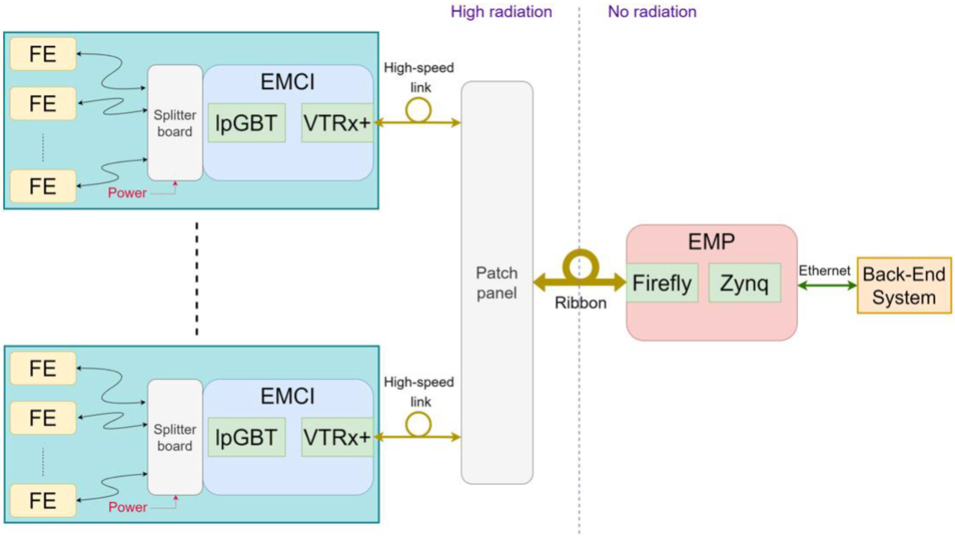


Figure 4.13: The complete EMCI/EMP network [49].

The EMCI, typically deployed in radiation environments, serves as an intermediary for controlling and monitoring data signals exchanged between multiple FE (e.g., MOPS-Hubs) and the DCS system. The EMCI has been designed to withstand high doses of radiation by integrating radiation hard components (lpGBT, VTRx++ or FEASTMP) as a part of its design [84–86]. Leveraging Low-power GigaBit Transceiver (lpGBT) technology, the EMCI consolidates all eLink signals into a single bidirectional channel and interfaces with the VTRx++ optical transceiver. This transceiver then transmits the data via a high-speed optical link, with transmission rates of 10.24 Gbit/s (or 5.12 Gbit/s)towards the back-end (uplink) and 2.56 Gbit/s towards the MOPS-Hub (downlink).

Using a single FMC connector, the EMCI integrates all eLink differential signals to the MOPS-Hubs, alongside additional digital and analog interfaces and power input (refer to Figure 4.13).

On the other hand, the EMP acts as a bridge between the EMCI and the distributed back-end of the experiment control system, typically within a commodity Local Area Network (LAN). It functions as an optical link transceiver module in non-radiation areas such as counting rooms, supporting multiple VL+ compatible optical fibers towards the detector front-end and an Ethernet interface towards the back-end. Utilizing a flexible System-On-Chip module (Zynq Ultrascale+), the EMP offers digital and analog interfaces. Furthermore, its embedded processing system allows running Linux-based software applications and, coupled with a standard Ethernet-compatible network interface, facilitates seamless integration of the EMP within the control system back-end.

Chapter 5

Radiation Effects on Semiconductor Devices

This chapter provides a comprehensive overview of radiation interactions with matter and their impact on semiconductor devices. It delves into various radiation effects on semiconductors, explaining how these effects impact the performance and longevity of semiconductor devices.

5.1 Particle Interactions with Matter

In high-energy physics experiments, ionizing radiation is any type of particle or electromagnetic wave that carries enough energy to ionize electrons from an atom. Ionizing radiation is crucial not only for identifying particles in silicon tracking detectors but also for its potential to irreversibly alter the characteristics of silicon sensors, as discussed in Sections 5.2. For transistors within SRAM cells in FPGAs, ionizing radiation can disrupt the transistor state by causing current surges or preventing current flow in electronics, as detailed in Section 6.4.

5.1.1 Heavy Charged Particles

The primary process of energy deposition of charged particles in matter is by excitation and ionization of the atoms. The mean energy loss per path of a charged particle by excitation and ionization is described by Bethe-Bloch formula shown in Equation 5.1.

$$\langle -dE/dx\rangle = 2\pi N_A r_e^2 m_e c^2 z^2 \frac{Z}{A}\frac{1}{\beta^2}\Big[\ln\Big(\frac{2m_e c^2\beta^2\gamma^2 W_{\max}}{I^2}\Big) - \beta^2 - \frac{\delta(\beta\gamma)}{2} - \frac{C(\beta\gamma, I)}{Z}\Big], \tag{5.1}$$

Figure 5.1 shows the curve of the mass stopping power in copper for positive muons in dependence of $\beta\gamma = p/Mc$. In the region $0.1 < \beta\gamma < 1000$ particles interact by ionization where the energy loss is proportional to $1/\beta^2$ until it reaches a broad minimum at $\beta\gamma \approx 3$, at which point particles are referred to as Minimum Ionizing Particles (MIPs). The particle behavior in this region is described by the *Bethe-Bloch function* [87].

with:

dE/dx	Energy loss per unit path length.
c, v	Speed of light and Particle speed.
m_e, r_e	Electron mass and the classical electron radius: $r_e = \frac{e^2}{4\pi\varepsilon_0 m_e c_0^2}$,
z	Charge number of the particle.
Z,A	Atomic number and Mass number of the absorber.
γ	Lorentz factor $\gamma = \sqrt{1-\beta^2}$
$W_{\max}$	Maximum energy transferred to an electron in a single collision, $W_{\max} \approx 2m_e c_0^2(\beta\gamma)^2$
β	Speed of incident particle given as the ratio to the speed of light: $\beta = \frac{v_0}{c}$.
I	Mean ionization energy of the material.
$\delta/2$	Density correction for high energies.
C/Z	Shell correction for low energies.

Table 5.1: Variables and constants used to describe the energy loss of particles in silicon using Bethe-Bloch formula[87].

As the particle energy increases, its electric field flattens so that the distant-collision contribution to Equation 5.1 increases as $ln(\beta\gamma)$. However, matter becomes polarized, limiting the field extension and effectively truncating this part of the logarithmic rise. For this reason the density effect term $\delta(\beta\gamma)$ is included in Equation 5.1. Another correction term $C(\beta\gamma, I)/Z$ is added to include the influence of shell correction at low energy where the velocity of the traversing particle is comparable to or smaller than the "orbital velocity" of the bound electrons in the matter.

High energy charged particles can be decelerated in the Coulomb field of the atomic nucleus releasing energy in the form of a photon, called *Bremsstrahlung*. The energy loss due to Bremsstrahlung is inversely proportional to the incident particle mass [88]. For that reason, low mass charged particles such as electrons and positrons are described differently with some modification in Equation 5.1. More details about passage of particles through matter and its interactions are reviewed in [87].

5.1.2 Electrons

Electrons and positrons lose energy through mechanisms similar to those of heavy charged particles. However, due to their lower mass, Bremsstrahlung becomes much more significant, and their interactions with shell electrons are governed by different kinematic constraints. At lower energies, ionization is the dominant process. However, once the energy exceeds a critical value, approximately $\mathrm{E_c} \approx 800/Z, \mathrm{MeV}$, where Z is the atomic number of the material, Bremsstrahlung becomes the primary mechanism for energy loss [20].

For high-energy electrons and photons, the electromagnetic interactions are characterized by the radiation length (X_0), which is the mean distance over which the energy is reduced by a factor of $(1/e)$ as it transverses the target material.

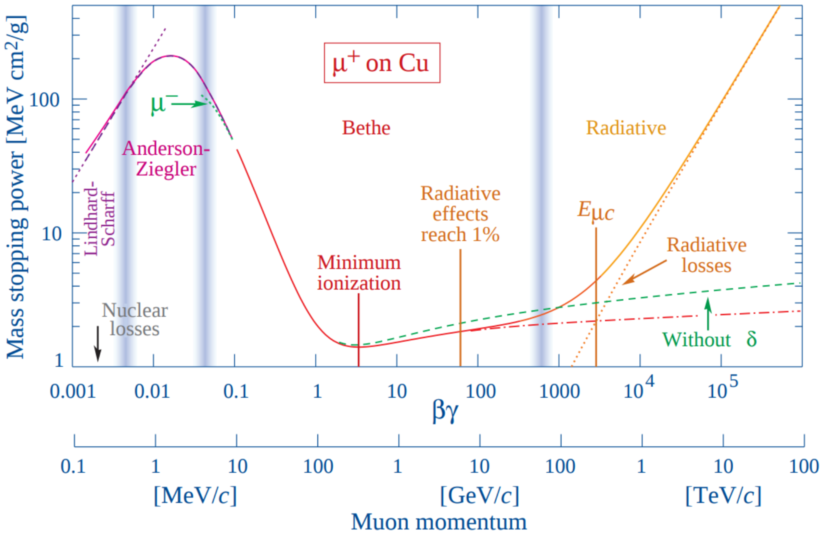


Figure 5.1: Mass stopping power = $\langle -dE/dx \rangle$ for positive muons in copper as a function of $\beta\gamma$ over nine orders of magnitude in momentum [87]. The curve of the total energy loss (solid) is red in the Bethe region. The dashed green line illustrates the rest of the Bethe function without the density effect term $\delta(\beta\gamma)$. Above the critical energy $E_{\mu c}$ losses due to Bremsstrahlung (dotted) dominate.

5.1.3 Photons

Compared to charged particles, photons in matter behave entirely differently. Since photons do not have an electric charge, coulomb interactions with atomic electrons do not cause them to lose energy. This makes photons significantly more penetrating than charged particles of comparable energy because, when passing through matter, a photon experiences no activity until it interacts with a single atom, depositing its energy via three major processes:

- The *photoelectric effect* is predominant at low energies. In this process, an atom absorbs an incident photon which transfers its entire energy E_γ to an electron from an inner shell with binding energy E_{bind} .

- *Compton scattering*, also called incoherent scattering, is the dominant interaction at energies from 50 keV to 1.5 MeV [89]. The incoming photon scatters off a quasi-free electron from the atomic shell.

- *Pair production* can only occur when the energy of a photon exceeds 1.02 MeV. It describes the conversion of a high energetic photon into an electron- positron pair in the Coulomb field of an atomic nucleus.

5.2 Radiation Effects on Semiconductors

Radiation effects in semiconductors can be broadly classified into two categories: Accumulated Effects and Single Event Effect (SEE).

5.2.1 Accumulated effects

Accumulated effects stem from the creation or activation of microscopic defects within the device. The gradual accumulation of these defects over time can result in measurable effects that affect the device's functionality and can eventually lead to complete device failure [90]. In more details, the category of the cumulative effects can be divided into two subgroups that identify the two micro-scale mechanisms on the basis of the incident particle type: Total Ionizing Dose (TID) and Displacement Damage (DD) which are detailed in the following Sections.

Total Ionizing Dose (TID)

The TID is a measure of the cumulative energy released into the material due to ionizing radiation [91]. This energy generates charges that can be collected in some sensitive part of the device, causing the degradation of the electronics (or sensors). In accordance with the International System of Units (SI), the Gray (Gy) is the unit of measure for the ionizing radiation dose ($1\,\mathrm{Gy} = 1\,\mathrm{J\,kg^{-1}}$) but it is often measured in rad ($1\,\mathrm{Gy} = 100\ 1\,\mathrm{rad}$).

When Metall-Oxid-Semiconductor (MOS) structures are exposed to ionizing particles, the energy of the particles is deposited in the SiO_2 insulator of the gate terminal, primarily through an ionization process. Figure 5.2 depicts a schematic energy band diagram of a MOS structure with a positive applied gate bias. As illustrated, the ionization in the material leads to the generation of electron-hole pairs, which can be separated by the local electric field.

In SiO_2, the generated electrons are much more mobile than the holes [93], and they drift out of the oxide towards the gate within Picoseconds [94]. However, in the first picosecond, some fraction of the electrons and holes will recombine. The holes, which escape initial recombination will move towards the Si/SiO_2 interface. As the holes approach the interface, some fraction will be trapped, forming a positive oxide-trap charge causing a negative threshold voltage shift in the MOS transistor [95]. The holes transported to the Si/SiO_2 interface will cause a short-term recovery of the threshold voltage. This process is normally over in much less than 1 s at room temperature, but it can be many orders of magnitude slower at low temperature [92]. Once the holes reach the Si interface, some fraction of the transporting holes fall into relatively deep long-lived trap states. These trapped positive charges cause a remnant negative voltage shift, which can persist for hours or even for years.

Another major mechanism happens right at the Si/SiO_2 interface building up radiation induced traps [96]. These traps are localized states with energy levels in the Si band-gap. Their occupancy is determined by the Fermi level (or by the applied

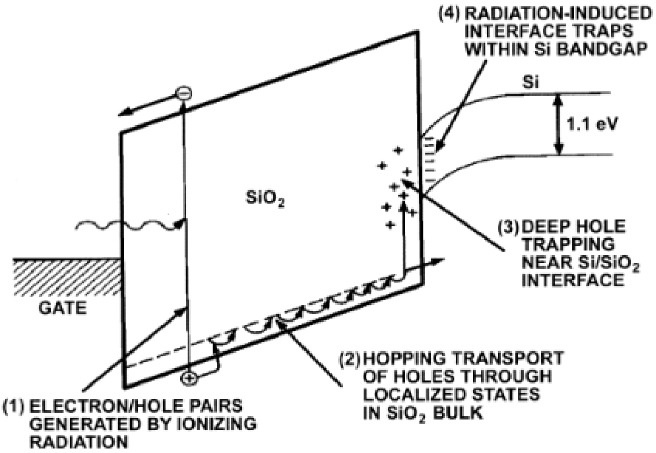

(4) RADIATION-INDUCED INTERFACE TRAPS WITHIN Si BANDGAP
Si
1.1 eV
SiO_2
GATE
(3) DEEP HOLE TRAPPING NEAR Si/SiO_2 INTERFACE
(2) HOPPING TRANSPORT OF HOLES THROUGH LOCALIZED STATES IN SiO_2 BULK
(1) ELECTRON/HOLE PAIRS GENERATED BY IONIZING RADIATION

Figure 5.2: Schematic energy band diagram for MOS structures, indicating major physical processes underlying radiation response [92].

voltage), giving rise to the threshold voltage (that is, a change in the voltage which must be applied to turn the device on).

The accumulation of trapped holes influences key parameters in the MOS transistors, such as the threshold voltage, charge mobility, and leakage current, ultimately altering the electric fields within the device and impacting its electrical characteristics.

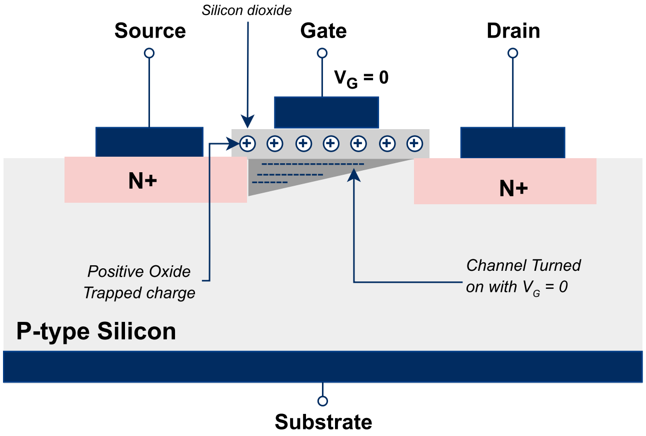


Figure 5.3: The effect of ionizing radiation on the gate oxide in an N-channel MOSFET with N-type implants in a P-type body, creating two PN junctions.

Figure 5.3 illustrates the radiation-induced trapped charge in a n-channel Metal-Oxide-Semiconductor Field-Effect Transistor (MOSFET) post-irradiation, the positive charge, trapped at the Si/SiO_2 interface, causes a shift in the threshold voltage.

If this shift is large enough, the device cannot be turned off, even at zero volts applied, and the device is said to have failed by going into depletion mode [97].

Displacement Damage (DD)

DD in an electronic device is caused by a longterm Non-Ionizing Energy Loss (NIEL), and it occurs when an incident particle has enough energy to displace atoms from their normal lattice site in the semiconductors. NIELs in silicon cause atoms to be displaced from their normal lattice sites, creating a vacancy where the atom had been. NIEL primarily occurs due to the elastic scattering of primary particles (electrons, protons, alpha particles, neutrons), as well as the fragments generated in nuclear reactions (inelastic nuclear scattering) involving incident protons or neutrons and device nuclei.

The number of particles/cm^2 transversing a material over some amount of time is expressed in terms of the radiation fluence, which is is defined as:

$$\Phi(E) = \int \psi(E,t)dt = \frac{dN}{dA} \tag{5.2}$$

where $\psi(E,t)$ is the particle flux or the fluence rate, given in particle/cm^2/s [88]. N is the number of incident particles over an area A.

Conventionally, to simplify complex radiation environments into equivalent mono-energetic forms, the NIEL damage is normalized to the damage level caused by 1 MeV neutrons [90]. This approach allows for the comparison of damage induced by different particle energies. As a result equation 5.2 can be expressed as

$$\Phi_n(E) = \int_{E_{\min}} \phi(E)dE \tag{5.3}$$

Given the spectral fluence $\phi(E)$, the 1 MeV equivalent neutron fluence is [98]:

$$\Phi_{\text{eq}}^{\text{1MeV}}(E) = \kappa \int_{E_{\min}}^{\infty} \phi(E)dE = \kappa \Phi_n(E). \tag{5.4}$$

Where κ is a parameter characterizing the displacement damage, known as the hardness parameter [99]. Here, $\Phi_{\text{eq}}^{\text{1MeV}}$ is the total radiation fluence quoted in 1 MeV neutron equivalents $\text{N}_{\text{eq}}/\text{cm}^2$. 1 MeV equivalent neutron fluence is the fluence of 1 MeV neutrons producing the same damage in a detector material as induced by an arbitrary fluence with a specific energy distribution.

5.2.2 Single Event Effects (SEE)

Single Event Effect (SEE) occur whenever an ionizing particle passes through a sensitive region of the electronic circuit. Depending on various factors, Single Event Effect (SEE) may cause no observable effect, a transient misbehaviour, a change of logic state, or permanent damage. In comparison to TID effects, SEE are caused by a single high energy particle hitting a device where it creates an ionization track.

A high energy deposition in a small volume of the electronics chip can be collected inside the reverse-biased p-n junctions by the present electric field through drift processes as depicted in Figure 5.4. This process causes a transient current at the junction contact and the alteration of the electrostatic potential, called field funnel. This funnelling effect will continue until it reaches to the bulk where most of the created electron hole pairs start to recombine. This process can increase the collection of charges at the node. The charge released along the ionizing particle path is collected at one of the microcircuit nodes, and the resulting current transient might generate a Single Event Upset (SEU). Two conditions are required for a SEE to happen, the highly energized particle must hit a specific region in a device defined as sensitive volume as well as the charge deposited by a particle hit must be larger than the required critical charge to create an upset [90]. In the LHC, the charged hadrons and the neutrons representing the particle environment do not directly deposit enough energy to generate a SEE. Nevertheless, they might induce a SEE through nuclear interaction in the semiconductor device or in its close proximity [90].

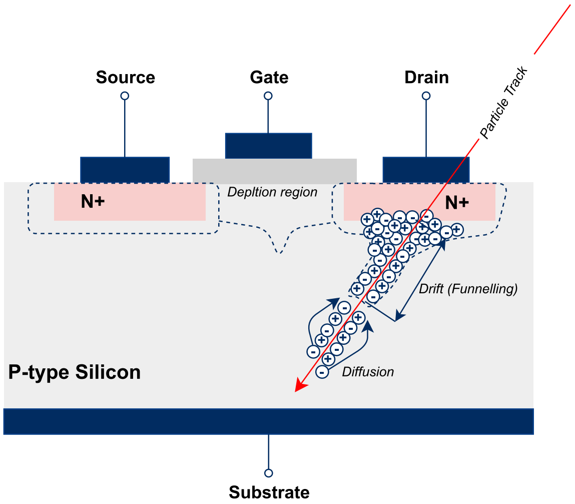


Figure 5.4: Example of SEE. Based on [100].

Figure 5.5 depicts the classifications of the SEE effects. As illustrated, SEEs can be classified as non-destructive SEE (soft errors) [101], which can be usually removed by resetting the device or applying the correct signal, and destructive SEE (hard errors), which may permanently damage the device [102].

The SEE sensitivity of a device to radiation is defined by the cross-section, or the ratio between the events and the particle integrated flux triggering them, that is measured in an irradiation facility with the appropriate particle type and energy.

At the LHC, the SEE cross-section is dominated by the hadron fluence rate $\leq$ 20 MeV [103]. For the planned upgrade of the LHC, the hadron fluence after an assumed 10-years running time is illustrated in Figure A.1(a) based on the simulation results [104].

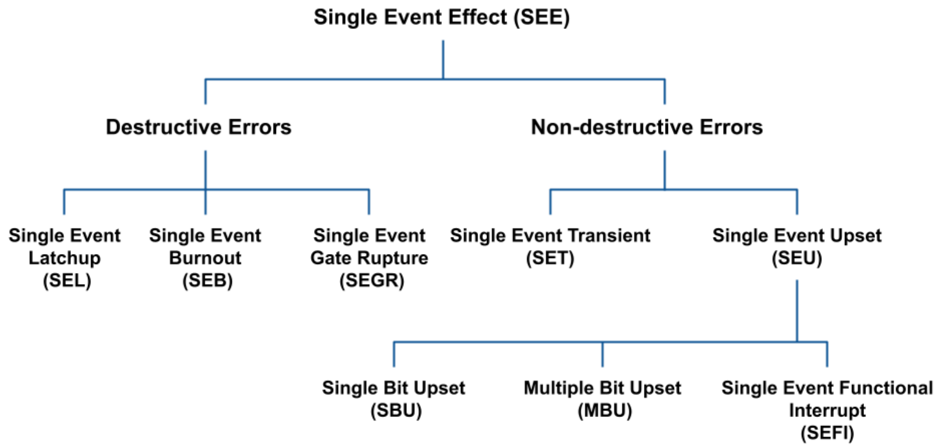

Single Event Effect (SEE)
Destructive Errors
Non-destructive Errors
Single Event Latchup (SEL)
Single Event Burnout (SEB)
Single Event Gate Rupture (SEGR)
Single Event Transient (SET)
Single Event Upset (SEU)
Single Bit Upset (SBU)
Multiple Bit Upset (MBU)
Single Event Functional Interrupt (SEFI)

Figure 5.5: Classification of SEEs.

Destructive SEEs (Hard errors)

As the name implies, failures induced by destructive effects are catastrophic and the devices are permanently damaged. Such situations can be due to one of the following events:

- **Single Event LatchUp (SEL):** It is a potentially destructive event that occurs when a low-resistance path between the power supply and ground of a device is created. This increases the current drawn by the device until power is removed, or the device fails catastrophically.

- **Single Event Burnout (SEB):** It occurs when the passage of a heavy-ion causes the MOSFET to enter a second breakdown that, induces a high-current leading the device into a thermal runaway until the failure.

- **Single Event Gate Rupture (SEGR):** It occurs when the passage of a heavy-ion through the neck region of the MOSFET creates a conducting path in the gate oxide. The charges created propagate up to the insulator interface, making the electric field across the dielectric very large. If this exceeds a certain value, a gate rupture can occur leading the device to fail.

Non-destructive SEEs (Soft errors)

- **Single Event Upset (SEU):** Is triggered by the generated transient current and the charge collected at the electric node when a charged particle passes through a reverse-biased junction [105].

- **Single Event Transient (SET):** It is a temporary voltage spike that can be triggered by a short-term current caused by the generated electron-hole pairs, which may change the logic state of a circuit [106]. It can last between picoseconds and nanoseconds.
- **Single Event Functional Interrupt (SEFI):** It is a temporary failing state that occurs when a single ion strike triggers an abnormal mode, such as test mode, or reset mode, which can cause Integrated Circuits (ICs) to lose their intended functionalities [107].

5.3 SEE Characterisation

SEEs are caused by a very high-energy deposition in a small volume of the electronics chip. The charge released along the ionizing particle path, or at least a fraction of it, is collected at one of the microcircuit nodes, and the resulting current transient might generate SEUs if the pulse has a certain *critical charge* Q_{crit} [108]. The critical charge corresponds to a *critical Energy* E_{crit} that has to be deposited in the silicon by the ionising particle to provoke SEU [108].

There are two different types of ionization: direct ionization, which occurs due to primary ionizing particles, and indirect ionization, which arises from secondary particles generated when non-ionizing or weakly ionizing particles interact with the device material.

5.3.1 Direct Ionization

Direct ionization occurs when a charged particle, such as a heavy ion, passes through a semiconductor and loses energy primarily through interactions with the electrons in the material, leading to ionization and creating a dense track of electron-hole pairs.

The SEU sensitivity of a circuit tested with heavy-ion irradiation facilities follows a different approach than indirect radiation, as the energy deposition of each heavy ion is commonly expressed in Linear Energy Transfer (LET), which represents the energy lost per unit path length, given in [MeVcm^2/mg].

$$LET = \frac{1}{\rho}\frac{dE}{dx} \tag{5.5}$$

Where ρ is the density of the material.

The amount of energy deposited (and charge created) in a sensitive volume of a circuit is proportional to LET as a function of path-length in this region.

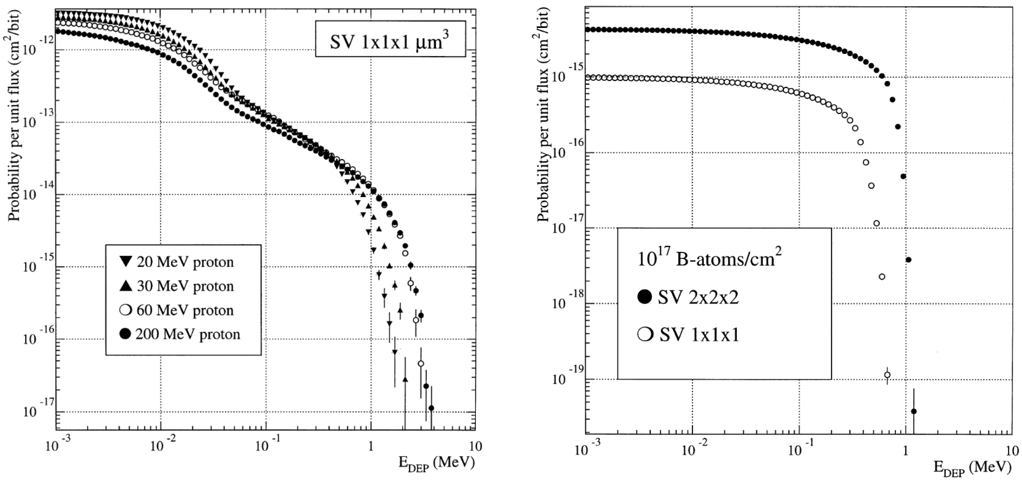
5.3.2 Indirect Ionization

Indirect ionization happens when particles like protons and neutrons interact with the semiconductor material and do not directly create enough charge to cause SEE.

Instead, these particles can cause ionization through interactions that generate secondary particles such as alpha particles and recoil nuclei. These secondary particles, being much heavier than the original striking particle, can then create tracks of electron-hole pairs along their paths, depositing sufficient charge into the sensitive volume of the circuit creating a SEE.

Figure 5.6 depicts the simulation done using FLUKA Monte Carlo code[109] for energy deposition probabilities in a sensitive volume $1 \times 1 \times 1$ µm^3. Where SEE characterization is influenced by the nature of ionization responsible for the upset.

Figure 5.6(a) depicts the energy deposition probabilities for four proton energies. As illustrated, at very low energies the probability increases with decreasing energy. This behaviour is consistent with the energy dependence of the proton-silicon cross section, as low-energy protons exhibit a higher LET. At E_{dep} of about 500 keV, the simulations predict the same SEU rate for all studied proton energies. This effect decreases with increasing E_{dep}.



(a) Energy deposition probabilities for protons. (b) Energy deposition probabilities for neutrons .

Figure 5.6: Energy deposition probabilities for protons and neutrons $^{10}B(n, \alpha)^7$Li of different energies using FLUKA simulations in a sensitive volume of $1 \times 1 \times 1$ µm^3 [103].

For neutrons, the SEU rate decreases gradually towards higher energies as depicted in Figure 5.6(b). This is also valid for the two sensitive volume ($1 \times 1 \times 1 \times 1$ µm^3 and $2 \times 2 \times 2 \times 2$µm^3) used in the simulation.

The major difference between proton and neutron irradiation is the coulomb repulsion, which mainly decreases the inelastic cross section of the proton at low energies. Neutrons and protons of the same energy produce almost identical upset rates if the incident energy exceeds 20 MeV [103, 110].

Equation 5.7 gives a general formula to calculate this SEU cross-section σ_{SEU} of a device.

$$\sigma_{SEU}(device) = \frac{N_{SEU}}{\phi} = \frac{N_{SEU}}{\psi \times \Delta t} \qquad [\mathrm{cm}^2/\mathrm{device}] \tag{5.6}$$

where N_{SEU} is the number of single event upsets that occur, ϕ is the total particle Fluence[1], ψ is the particle flux[2] and Δt is the time interval in seconds.

The cross section per bit can then be obtained by dividing the device cross section by the the number of bits in the used memory N_{bits}

$$\sigma_{SEU}(bit) = \frac{\sigma_{SEU}(device)}{N_{bits}} = \frac{N_{SEU}}{\phi \times N_{bits}} \qquad [\mathrm{cm}^2/\mathrm{bit}] \tag{5.7}$$

5.4 Radiation level in the ATLAS Hall

Many components of the detection systems of high-energy experiments are exposed to the adverse radiation environment that results from the interactions with surrounding materials of particles produced by high-rate collisions of the incoming beam with the target or head-on collisions of particle beams at high luminosity.

The radiation environment within the ATLAS inner detector is complex, encompassing a broad spectrum of particles such as pions, protons, neutrons, photons, and more. These particles vary in energy levels, ranging from TeV energies down to thermal energies for neutrons.

Close to the interaction point, the predominant particles originate directly from proton-proton collisions with high hadron fluxes, mainly induced by protons, pions and neutrons [111]. However, at larger distances, secondary neutrons resulting from high-energy hadron and electromagnetic cascades in the calorimeters can significantly influence the environment [98].

Such interactions typically will produce a shower of secondary particles and one nuclear recoil in some cases there can be several fragments. These nuclear recoils have low energies $\leq 10\,\mathrm{MeV}$ which can produce a SEU if it is trapped in the electronic chip. The upset rates will be dominated by the interaction of all hadrons $\leq 20\,\mathrm{MeV}$ with silicon nuclei in the ICs.

The radiation background simulations for areas inside the LHC and the corresponding radiation levels of the upcoming HL-LHC are primarily utilize the FLUKA Monte Carlo code to examine the hadronic and electromagnetic cascades initiated by high-energy particles [104, 109, 112]. Comprehensive details about FLUKA's physics models and capabilities are available in references [104, 113]. PYTHIA8 is employed to generate inelastic proton-proton collisions, which are then inputted into the FLUKA simulation for precise simulation of radiation environments [59].

[1] **Particle Fluence** ϕ is the number of particles per area over some amount of time (usually the assumed lifetime of the detector), given in [particle/cm^2] [88].

[2] **Particle Flux** ψ is the fluence rate, given in [particle/cm^2/s].

Simulation results detailed in Section A.1 showed that the two main contributions to SEE inside the LHC are the High Energy Hadrons (HEH) and neutrons. Recent simulation work addressing upset rates in the LHC radiation environment showed that the upset rates will be dominated by the interaction of HEH with silicon nuclei in the ICs [103].

Chapter 6

FPGAs and its Radiation Tolerance

The FPGA achieves its applications through hardware logic design that is uniquely re-configurable, acting as a balance between the fixed functionality of ASICs and general-purpose processors [114]. This chapter outlines the main concepts of FPGA technology. This will provide a groundwork for Chapter 9 in this thesis.

6.1 FPGA Architecture and Technologies

FPGAs are complex integrated circuits that house extensive digital logic circuitry known as Configurable Logic Blocks (CLBs), which are interconnected via programmable routing matrices, commonly referred to as *switch blocks*, and I/O cells [115].

Figure 6.1 depicts a generic FPGA architecture, illustrating how these routing matrices enable customizable interconnections between CLBs. This flexibility facilitates routing connections among logic blocks and I/O blocks, thereby enabling the creation of fully functional circuits. I/O cells also play a crucial role by interfacing with external peripherals.

Each CLB typically includes Static RAM (SRAM) cells configured as Look-Up Tables (LUTs) for combinational logic, along with multiplexers, carry logic, and storage elements such as Flip-Flops (FFs) for sequential logic and pipelines. The exact internal composition of a CLB can vary among different FPGA manufacturers. The overall functionality of the FPGA is determined by the configuration of these CLBs and their interconnections, which are stored in a memory region known as the Configuration Memory (CRAM). This memory, distributed across the FPGA, typically represents the largest number of bits within the device.

A typical 7-series Xilinx FPGA includes 8 LUTs per CLB, divided into two logic slices containing 4 LUTs each. Modern FPGAs incorporate additional efficient blocks like Block RAMs (BRAMs), Digital Signal Processing (DSP) blocks, and various I/O controllers such as Double Data-Rate (DDR) registers, which help enhance general-purpose computing capabilities [116–118].

FPGAs are categorized based on their process technology into three main types:

- *Antifuse-based FPGAs* are known for their one-time programmability and offer low power consumption and high radiation tolerance but are not re-configurable [119, 120].

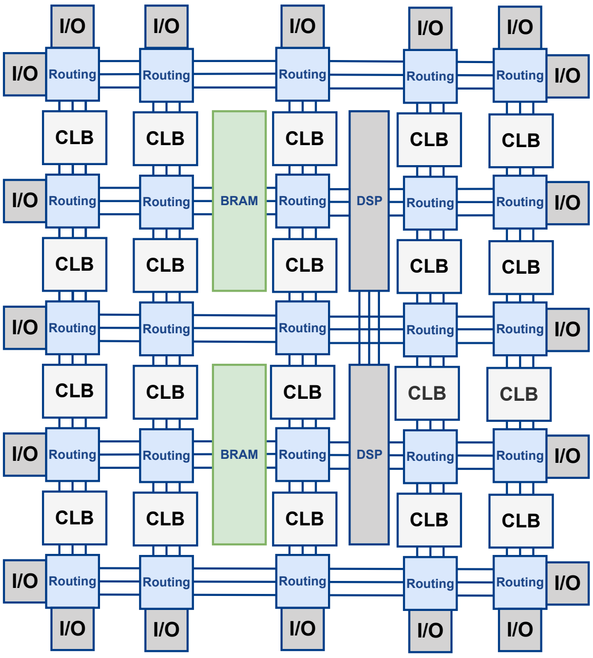


Figure 6.1: Generic FPGA architecture, illustrating the primary components and interconnects. Based on [115].

- *Flash-based FPGAs*, which maintain their configuration across power cycles due to their non-volatile flash memory, offer reconfigurability and low power consumption but generally do not match the performance of SRAM-based FPGAs.
- *SRAM-based FPGAs*, requiring external non-volatile memory for configuration data, are noted for their high power consumption but provide significant advantages in terms of resource availability, capacity, and speed. However, they are susceptible to radiation effects and require protective measures in adverse environments [121–123].

The Artix-7 model, utilized in this thesis, uses the SRAM-based FPGA category and is discussed in detail in Section 6.3.

6.2 Development Workflow

Figure 6.2 depicts the multi-step development flow for FPGA configuration, providing a visual overview of the processes involved from initial design to final implementation.

The initial step in designing a digital circuit in an FPGA involves describing the intended functionality using a Register Transfer Level (RTL) approach through a Hardware Description Language (HDL). Common languages used for this purpose include VHDL [124] and Verilog [125]. The essence of RTL design lies in specifying

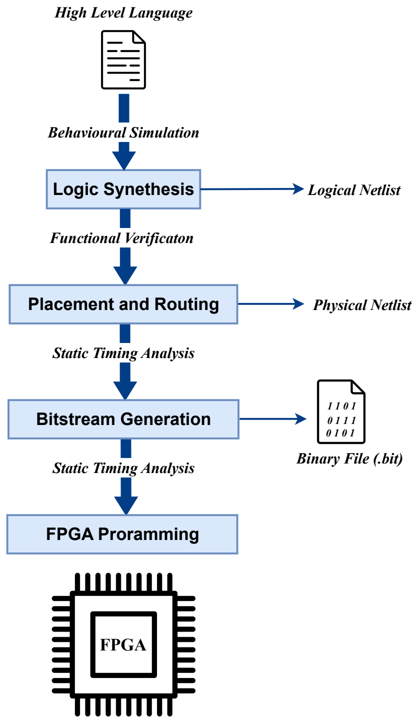


Figure 6.2: Development workflow for FPGA configuration.

the behavior of a circuit by detailing the signals transferred between registers and the logical operations applied to these signals. This detailed specification helps in configuring the LUTs, interconnecting them along with the FFs within the CLBs through multiplexers, and effectively linking CLBs via routing matrices [115]. The result of this design phase is the firmware; a comprehensive and detailed blueprint of the FPGA's configuration.

To ensure the correctness of the RTL code and its operation, Behavioral Simulation is essential. This simulation process is thoroughly discussed in Section 6.2.1. Following successful simulation and validation of the RTL code through various RTL checks, the process advances to Firmware Synthesis as described in Section 6.2.2. During synthesis, the RTL code is translated into a gate-level netlist, which is a schematic representation of the design that maps the logic described in RTL to the physical gates available in the FPGA.

6.2.1 Firmware Simulation

Firmware simulation is an essential step in the FPGA development process, typically performed using the same HDL that describes the firmware. This simulation employs a test unit known as a *testbench*, which is designed to emulate a physical lab bench that tests the circuit under real-world conditions.

The testbench functions as a comprehensive testing environment where the Device Under Test (DUT) is evaluated. It simulates inputs to the DUT and monitors its outputs to verify correct behavior according to the design specifications.

6.2.2 Synthesis and Implementation

Firmware synthesis is a critical phase in FPGA development, where the RTL code describing design modules is transformed into a gate-level netlist. This process encompasses several optimization steps targeting logic, area, and power efficiency, as well as the integration of scan chains for testing purposes. Various tools provided by FPGA manufacturers interpret the RTL description and generate a corresponding logic circuit that accurately reflects the intended functionality.

The process begins with **logical synthesis**, where the design is refined to implement the desired logic and functionality using the minimum number of gates. This includes optimizing the timing to meet specified constraints. The output of this stage is a logic circuit constructed from the basic building blocks of the FPGA, known as *primitives*[1].

Following logical synthesis, the workflow progresses to **physical synthesis**, which optimizes the placement and routing of the circuit within the FPGA's architecture to improve area utilization and power consumption. This stage is critical for ensuring that the circuit meets all physical constraints imposed by the FPGA's layout and operating conditions. The result of these efforts is the generation of a binary configuration file, or **bitstream**, which configures the FPGA's CRAM to replicate the designed circuit. This bitstream is a sequence of bits loaded into the FPGA through various methods, with Joint Test Action Group (JTAG) being a common choice during development phases.

The effectiveness of the synthesis and implementation process is highly dependent on the specific FPGA architecture targeted. Each stage of this development flow can be tailored and controlled through the use of constraints and directives set by the designers.

6.3 SRAM-based FPGA Technology

SRAM-based FPGAs are susceptible to radiation effects, which can cause transient or cumulative damage [126]. It is imperative to employ robust fault mitigation strategies to counteract these effects, particularly in radiation-rich environments like space. The space electronics community has devoted considerable efforts to develop and refine such methods to leverage the benefits of FPGAs safely in these challenging conditions [127].

An additional consideration is the volatility of SRAM cells, which lose data when power is discontinued. As a result, SRAM-based FPGAs must be paired with external

[1] Examples of primitives include FFs, Multiplexers, BRAM, and multipliers. These elements are the smallest configurable logic units within an FPGA.

memory sources that store the configuration bitstream, ensuring it is loaded into the FPGA's CRAM at each power-up.

6.4 Radiation Effects on SRAM-based FPGAs

Components manufactured in CMOS technologies are generally sensitive to TID and SEEs [90].

New generations of SRAMs, using a 6T cell design (six MOSFETs transistors), are expected to have an improved TID and SEU behaviour [128, 129]. In the following sub-sections, the two effects are discussed in details for SRAMs.

6.4.1 TID Effect in SRAM

For SRAM technologies, the main reliability issue lies with their high sensitivity to SEEs [123, 130]. Regarding the accumulated effects such as TIDs, SRAM cells are generally considered as incorruptible by TID as most modern SRAM devices can reach TID tolerance over 1 Mrad(Si) [131].

Beyond the intrinsic sensitivity of the CRAM, the rest of the FPGA fabric also exhibits TID induced effects. At the FPGA level, this can lead to an increase of the power consumption and a degradation of the propagation delay of the logic gates [132, 133]. Nevertheless, some studies have demonstrated that the accumulated TID dose on FPGA can cause an increase of their SEU sensitivity and increase in its power consumption [134, 135].

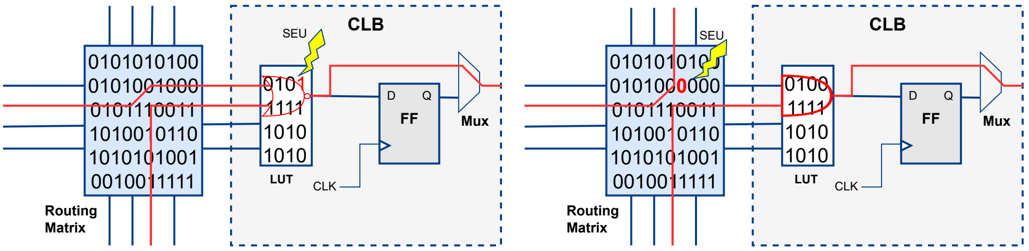
6.4.2 SEE in SRAM

Single Event Effect (SEE)s in SRAM-FPGA can occur in logic modules, I/Os, routing resources, and BRAMs. This effect may happen due to the charge deposition following the ionization process as detailed in Section 5.2.2. Each primitive resource that composes the SRAM-based FPGAs fabric has its own intrinsic sensitivity to SEE as discussed in the following sections.

SEU in SRAM

SEUs can corrupt the configuration bits that define the behaviour of the FPGA. This affect the logic implemented whether it is sequential or combinational. However, The impact on the sequential logic (e.g. FF) is often more noticeable. These kind of soft errors can propagate throughout the circuit potentially leading to a system failure. The mechanism by which the value of an SRAM-based cell corrupted by incident radiation is depicted in Figure 6.3.

Figure 6.3(a) illustrates functionality of the whole CLB can change by flipping one bit in the LUT due to SEE. Figure 6.3(b) depicts the same effect in the routing matrix. This type of upsets can disconnect routes, create new routes, or even bridge two routes together [136]. Consequently, SEUs will potentially modify the design and



(a) A SEU in a LUT module.

(b) An SEU in the routing matrix.

Figure 6.3: An abstraction of an FPGA under the effect of SEU. The red line represents the routing and functions implemented. Based on [137]

can cause any number of issues, the nature of which is difficult to predict. An upset in one LUT can corrupt the whole data stream, or even turn off an entire FPGAs effectively if it hits the clocking logic.

Another, scenario can happen when a particle traversing the FPGA can affect multiple memory cells, in that case the SEU will create Multi Cell Upset (MCU). This can occur when the particle is crossing the device at an angle, or with a normal incidence at the border between memory cells as depicted in Figure 6.4. The organization of

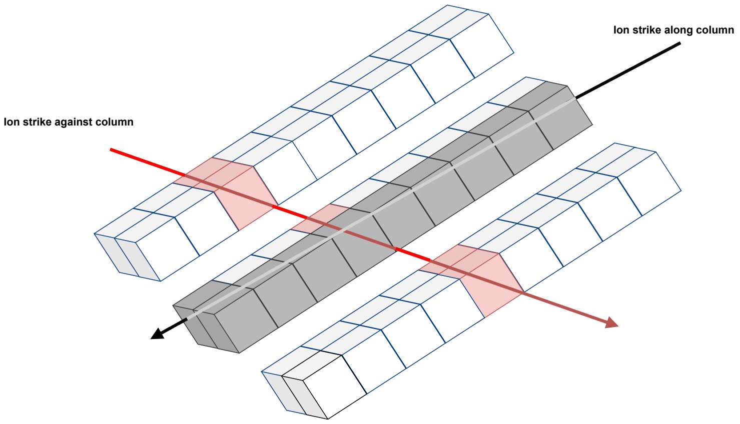


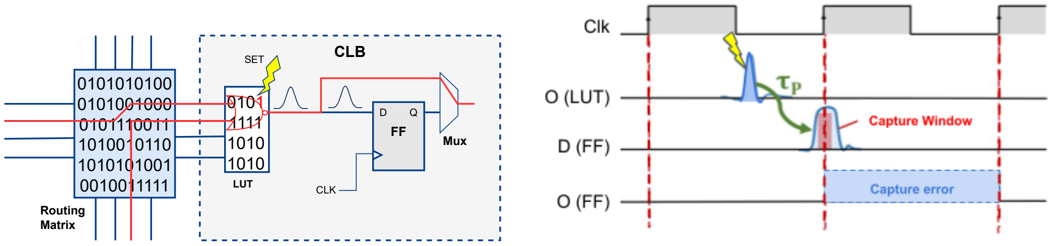
Figure 6.4: MCU in SRAM cells [138].

the SRAM cells in the Figure 6.4 significantly higher number of cells when ions strike along columns, rather than against them [138].

If an SEU produces a single bit-flip, then it will be referred to as Single Bit Upset (SBU). MCUs affecting bits within the same memory frame and ignores other MCU that affects other words is referred to as Multi Bit Upsets (MBUs) [139].

SET in SRAM

Single Event Transient (SET)s in the FPGA are mainly generated and propagated in the combinatorial logic. It is more complex to consider since they can be generated in any logic gate or pass transistor in the device and their propagation cannot be predicted accurately without proprietary information of the electrical properties of the FPGA primitives or extensive SET propagation tests. Furthermore, in logic circuits, SEUs can occur when SETs propagates through a combinational logic, and is then captured by a latch or a FF as depicted in Figure 6.5.



(a) Propagation of SETs generated in a LUTs.

(b) Captured signal due to a SET.

Figure 6.5: Propagation and capture of SETs generated in a LUTs. The SET must reach the input of the FF with sufficient amplitude during the capture window to be captured [140].

SEL in SRAM

Single Event LatchUp (SEL) are mainly due to a passage of a single energetic particle, typically heavy ions or protons, through sensitive regions of the device structure. This typically occurs in the circuitry where PNP[2] and a NPN[3] parasitic transistor are stacked next to each other in PNPN structures, which are also known as thyristors or Silicon Controlled Rectifiers (SCRs) [141]. In this structure the two PNP and NPN transistors are in a feedback loop, such that the output (collector) of each transistor is connected to the input (base) of the other, as shown in the overlaid equivalent circuit in Figure 6.6.

When the particle deposits enough energy in this thyristor, it produces a low-impedance path in the feedback loop between the power supply rail and the ground [142]. The result is an abnormal high-current state in the device resulting in the loss of device functionality. This effect must be corrected by power-cycling the device to avoid the risk of damaging the component permanently and restore the normal operation [143].

To mitigate the effects of SELs on an IC, the basic solution is to monitor externally the supply current of the circuit and to perform a power cycle when a significant current peak is detected.

[2] PNP is formed by the p+ source/N-well/P-substrate

[3] NPN is formed by the n+ source/P-substrate/N-well

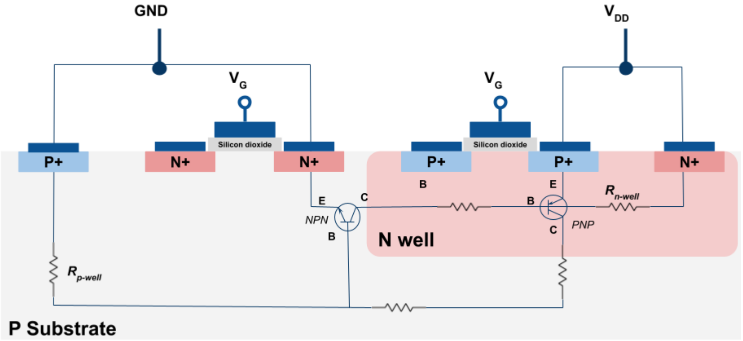


Figure 6.6: Cross-section of an N-well CMOS technology showing parasitic Parasitic thyristor. Based on [141]

6.4.3 Radiation Hardness Studies of SRAM-FPGA

This section briefly introduces several studies showing the radiation hardness of SRAM-based FPGAs, which will be the base criteria for the selection of the FPGA in this thesis.

SEU Cross-section in CRAM

The evaluation of SEU sensitivity is carried out by placing the device under a beam of particles and measuring its cross-section (σ_{SEU}) as defined in Equation 5.7.

Several studies, particularly focusing on Xilinx 7 series FPGAs, have been conducted by various groups, predominantly covering moderate or high-performance families. Notable works include those by [77–81]. The results of these studies give the cross-section data for all their FPGA families using irradiation facilities with wide-spectrum neutrons, protons, heavy-ions, and mixed high-energy hadron environments.

As an example, the SEU cross-section calculated for the CRAM of the Artix-7 FPGA from Xilinx using a neutron facility is found to be $6.99 \times 10^{-15}\,\text{cm}^2/\text{bit}$ [81].

TID Studies in SRAM-FPGA

Concerning the SRAM-FPGA, many FPGA manufacturers propose specific products for radiation applications, for which they guarantee TID tolerance up to (10 to 100) krad. Previous TID irradiation campaigns for TID on the Artix-7 have reached values of 550 krad [83, 144].

6.5 SEU Mitigation in SRAM-FPGAs

SEUs can be detected and corrected through a variety of SEU mitigation techniques. These mitigation approaches typically involve some form of redundancy and coupled with a repair process which restores the original configuration of the FPGA after an SEU occurs.

This section gives a brief overview of the different types redundancy techniques used in this thesis for the PP3-FPGA. The most popular of these techniques is Triple Modular Redundancy (TMR), discussed in Section 6.5.1, and the configuration scrubbing discussed in Section 6.5.2.

6.5.1 Triple Modular Redundancy (TMR)

TMR is an established SEU mitigation technique for improving hardware fault tolerance. The conceptual idea is to implement three identical instances of the logic with a voting module at the output. In this manner, if an SEU affects one of the three aforementioned logic instances, the output will be based on the majority voting of all three outputs. Consequently, the error will not propagate throughout the design. As depicted in Figure 6.7, the feedback of the output allows for the voting mechanism to restore the proper state on the next clock cycle.

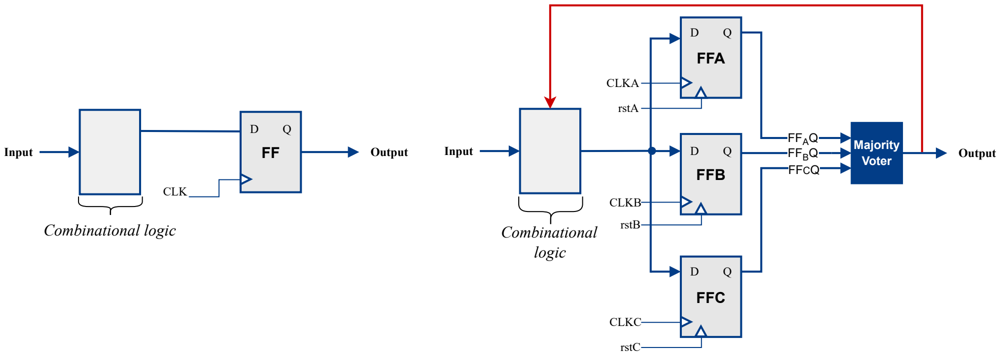


Figure 6.7: Illustration of the TMR implementation.

By the nature of triplication, two errors cannot be fixed by the voting mechanism but is signalized by an error signal. Three errors will result in a failure. The likelihood of such a failure depends on how the modular redundancy is implemented and the SEU rate in the environment.

Although TMR in addition with voting logic offers great protection, it also comes at a cost of requiring more than three times the FPGA resources compared to a non-TMR design. An increase in the triplication factor is associated with a higher resource usage, which leads to higher cost and power consumption, and a more complex majority voter, which will also be more vulnerable because of the increase in signal routing (routing matrices are a major source of SEU-related errors [145]. Therefore, the work shown in the firmware level of the PP3-FPGA in this thesis implements only Partial Triplication, means that only memory elements are triplicated. This implementation was essential to stay within the hardware constraints of the chosen Artix-7 FPGA.

6.5.2 Configuration Scrubbing

To obtain maximum system reliability, TMR is often coupled in FPGAs with configuration scrubbing as the repair process. The repair process fixes any existing faults in the system, particularly those affecting the CRAM. Otherwise, SEUs would build up over time, eventually overwhelming even the most robust mitigation technique. As a result,

The basic principle of configuration scrubbing is to use the FPGA configuration interface for repairing SEUs in the CRAM to recover the original state of the FPGA CRAM. This operation requires some basic components as depicted in Figure 6.8.

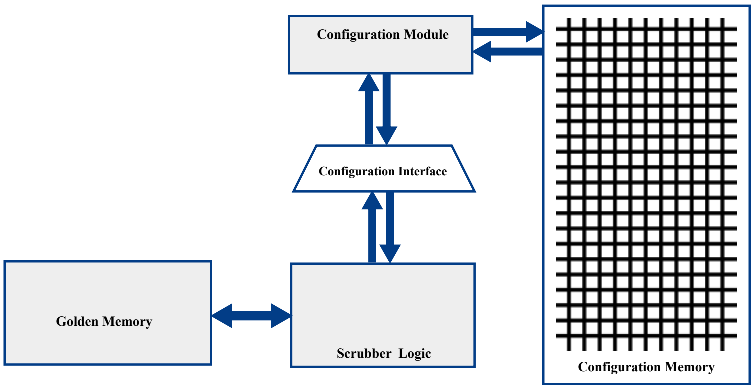


Figure 6.8: Basic Scrubbing components.

First, the *Scrubber Logic* needs a *Configuration Interface* to access the CRAM. Secondly, a *Configuration Module* in the form of processing unit is required to manage data exchanges with the CRAM, and interpret returned data to make informed decisions about scrubbing. This unit is often realized through a dedicated processor or a Finite State Machine (FSM) embedded within the FPGA's logic. Lastly, for enhanced error correction approaches, a *Memory* component is necessary to store the golden bitstream, which is used to overwrite upsets. This memory is typically a BRAM, external DDR or flash memory.

Depending on which FPGA configuration interface is used to reconfigure the device, scrubbing techniques are classified as either external or internal. External scrubbing techniques use external configuration ports (i.e., JTAG, SelectMap) and require an external radiation-hardened scrubbing controller as well as a golden copy of the configuration bits. On the other hand internal scrubbing techniques use an internal configuration interface (i.e., Internal Configuration Access Port (ICAP)) to access the CRAM of the FPGA. In this case, the scrubbing is controlled internally, and the controller usually consists of an embedded hardware processing unit. This section presents general overviews of two prevalent configuration scrubbing strategies, *Blind Scrubbing* and *Readback Scrubbing*.

Blind Scrubbing

The *Blind Scrubbing* simply re-writes the entire configuration of the FPGA at a chosen interval. It blindly overwrites the existing configuration bits with a pre-specified rate [146]. The re-write is done whether an upset exists in the configuration or not. This approach requires the use of a *golden copy* of the firmware, stored in an accessible location (e.g., a rad-hard memory near the FPGA). This process should be maintained so that, the scrubbing period is shorter than the estimated mean time between two SEUs.

One of the main disadvantages of blind scrubbing is its inability to detect upsets. Additionally, the power usage in this method is high as many frames are scrubbed and a large amount of memory is required to store the initial states of the configuration bitstream.

Readback Scrubbing

The *Readback Scrubbing* strategy, as opposed to blind scrubbing, systematically checks every frame of CRAM for errors and will repair/overwrite the contents of the frame with the correct data if needed. The checking process sequentially progresses through the configuration frames and cycles back to the device's start upon completion, an upset in a frame immediately after its inspection will remain undetected until the scrubber has traversed the entire memory. Upon reading a single frame, the detection of errors within the frame is accomplished by one of two methods [147]:

1. *Golden Data based* in which an external memory is used to store the golden data, which is a copy of the initial configuration data loaded onto the device. After reading back a frame, the data can be compared word by word to its corresponding location in the golden memory. If a discrepancy exists, an upset has been detected. One could also use an error-detection code on the memory frames (e.g. Xilinx built-in Error-Correction Code (ECC)) to avoid fetching every frame of the golden copy.

2. *ECC based* in which an ECC calculation is done whenever a readback of a frame is performed. This method requires some sort of data redundancy in the form of ECC as discussed in [148]. The ECCs are stored either in a separate memory or in the frame itself using an ECC hardware primitive as in 7 series FPGAs from Xilinx, refer to as *FRAME_ECC*.

Once an upset has been detected, the frame with the upset can be corrected by either overwriting the upset using the contents of the golden data, or by inverting the single bit at the location given using a 13 bit register, referred to as *Syndrom*, in the readback data with the error and then writing that data back into the CRAM.

Readback scrubbing is more complex than blind scrubbing, but it provides much more information about the upsets that occur. They offer the same robustness as blind scrubbers in upset correction which includes MBU correction if using golden data. Unfortunately, using the ECC values for correction only achieves a Single Error

Correction and Double Error Detection (SECDED) capability since ECCs cannot correct MBUs.

6.6 SEU Protection for 7-Series FPGAs

Since the FPGA of the PP3-FPGA is from Xilinx's 7 Series family, this section delves into various features and mechanisms specifically designed for the Xilinx's 7 Series family to safeguard the FPGA's CRAM against upsets.

6.6.1 7-Series Frame Layout

In the Xilinx 7-Series, each frame in the CRAM is comprised of 32 bit words [149]. Generally, frames can be either dynamic or static, and they are categorized into four different types (Type 0-3). The majority of frames (more than 70%) are static frames (Type 0, Type 2 and Type 3), corresponding to the circuit parts whose functions remain unchanged after a configuration. Other frame types, such as Type 1, including BRAMs, are dynamic frames changing during FPGA operation.

A frame is comprised of 101 words and with a built-in ECC word to achieve SECDED functionality through Readback Cyclic Redundancy Check (CRC) mechanism [150]. The 51^{st} word in a frame of Type 0, 2, and 3 includes the ECC.

6.6.2 Readback CRC

The Readback CRC in 7-series FPGAs is a dedicated hardware engine. It initiates the computation of ECC for each frame, and computes the 13 bit register (Syndrome) using all words included in a frame.

With the 13 bit syndrome there are $2^{13} = 8,192$ possible syndrome values used to indicate the precise location of a SBU in a frame.

Because each frame has its own ECC, the Readback CRC considers ECC upset detection and correction on an individual frame basis. During readback, the syndrome bits are calculated for every frame. If a SBU is detected, the readback is stopped immediately. The frame in error is read back again, and using the syndrome information, the bit in error is fixed and written back to the frame.

When MBUs occur in different frames by the same SEU or by different MBUs at nearly the same time, the Readback CRC will at best be unable to correct MBUs, and at worst be unable to detect them at all.

In 7 series FPGAs families, the Readback CRC capability is coupled with an ECC hardware primitive called *FRAME_ECC2* to correct SBUs per frame or MBUs distributed across frames, with a maximum of one per frame.

The *FRAME_ECC2* primitive plays an essential role in various scrubbing architectures by enabling the scrubber to access and analyze the built-in Readback CRC. More information about this primitive is described in the 7 Series FPGA Libraries Guide [151].

6.6.3 Soft Error Mitigation IP (SEM IP)

Xilinx created the Soft Error Mitigation IP (SEM IP) as a hybrid scrubber tool. Hybrid in this context means combining the capability of the built-in Readback CRC hardware to detect and correct SBUs and the MBUs correction capability that Readback CRC cannot deal with.

The SEM IP utilizes the ICAP to have a direct access to the configuration registers. The error-recovery mechanism uses the ICAP to read and write a configuration frame as opposed to letting the Readback CRC perform the correction. The readback and reconfiguration operations are performed using an appropriate sequence of 32 bit configuration commands sent to the ICAP input. These commands are predefined and stored in the internal memory of the FPGA BRAM.

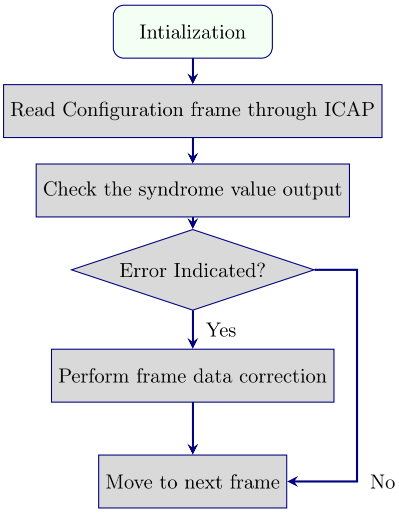

Intialization
Read Configuration frame through ICAP
Check the syndrome value output
Error Indicated?
Yes
Perform frame data correction
Move to next frame
No

Figure 6.9: Frame Error Correction using *FRAME_ECC* output.

The SEM IP utilizes the internal Readback CRC hardware to perform error detection using the built-in *FRAME_ECC* primitive and CRC codes [152]. The *FRAME_ECC* primitive works in parallel with the ICAP device. While the ICAP primitive reads the particular frame, the *FRAME_ECC* acts as an internal scrubber that uses the frame data to compute the syndrome value [151]. This gives the SEM IP the advantage of the built-in Readback CRC hardware to detecting and correcting both SBUs and MBUs.

As depicted in Figure 6.9, upon initialization, the SEM IP controller will bring itself into a known state by accessing control registers in the FPGA configuration system through the ICAP primitive [151]. The controller is then set up a Readback CRCs process to detect errors using *FRAME_ECC* primitive. This typical error detection has a time latency of 25 ms [152].

Xilinx SEM IP implements five main functions: optional capability to classify CRAM errors as either ‘essential’ or “non-essential”. Additionally, it has three error correction modes.

- *Repair*: Repairs single-bit errors using an ECC algorithm.
- *Enhanced Repair*: ECC and CRC algorithms correct single- and double-bit errors. This mode is the one used during the SEU testing for the PP3-FPGA.
- *Replace*: Using an external flash memory to store the design configuration an arbitrary number of upsets can be corrected.

If a CRC-Only error is detected (referring to a CRC error with no ECC information), the SEM IP considers this error "uncorrectable" and requires a full reconfiguration to return to proper functionality.

Chapter 7

Firmware Description and Verification

This chapter delves into the firmware development for the PP3-FPGA, with each section dedicated to a different aspect of the firmware's design and implementation.

7.1 Firmware Architecture

Fig. 7.1 depicts the firmware architecture for the PP3-FPGA, showcasing a structured design comprised of various components. Each component is a functional block of firmware that serves a dedicated purpose and interfaces with other blocks. The architecture also integrates additional FPGA resources, including clock management, dedicated memory and I/O ports.

7.1.1 Firmware Components

Detailed descriptions of each block are provided in the sections that follow.

The firmware architecture encompasses the following components:

1. **CAN Interface**: Aggregates input signals for subsequent processing, encompassing necessary blocks for CAN communication (refer to Section 7.3.2).

2. **Bus Monitoring Interface**: Conveys monitoring data, collected from external ADCs through SPI, into the data stream through eLink (refer to Section 7.4.2).

3. **Bus Control Interface**: Manages the VCAN for connected buses through SPI communication (refer to Section 7.4.3).

4. **eLink Receiver**: Houses downstream memory blocks and the 8B10B decoder module, ensuring data received from the eLink side is synchronized, deserialized, and aligned to 8B10B symbols (refer to Section 7.5.2).

5. **eLink Transmitter**: Houses upstream memory blocks and the 8B10B encoder module, with data from the encoder being multiplexed for transmission through the eLink port (refer to Section 7.5.1).

6. **eLink Serializer**: Supplies the necessary components for data transmission and serialization over the eLink (refer to Section 7.5.3).

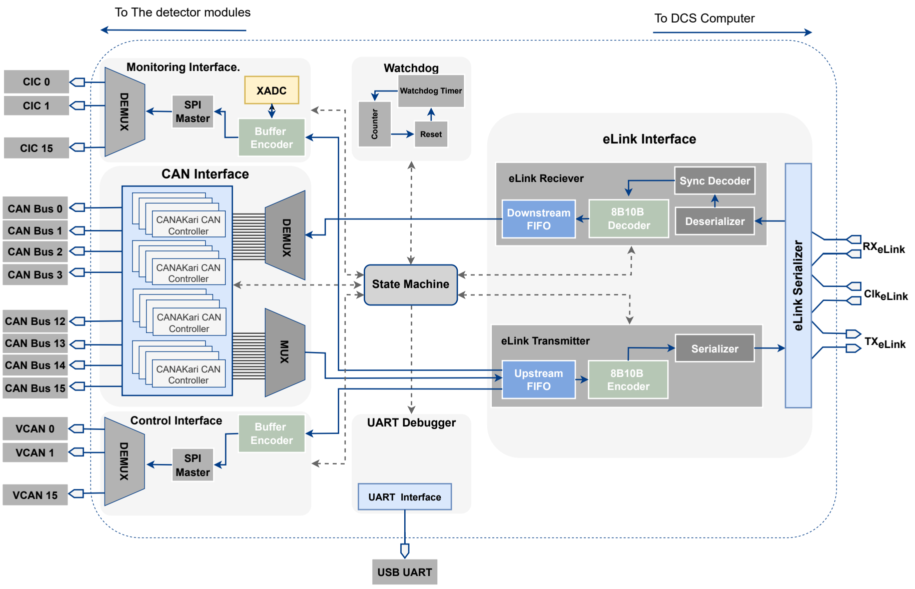


Figure 7.1: Firmware design of the PP3-FPGA. The blue bold lines represent data lines while the dashed lines represent the control signal from/to the top FSM.

7. **Central State Machine**: Initializes the system, synchronizes CAN signals, and coordinates interactions among all components (refer to Section 7.6).

8. **Watchdog Timer**: Oversees the operation of all FSMs post-power-up, ensuring the system reverts to a safe state in case of errors (refer to Section 7.7).

9. **UART Debugger**: Facilitates access to internal registers for debugging and monitors transmission status (refer to Section 7.8).

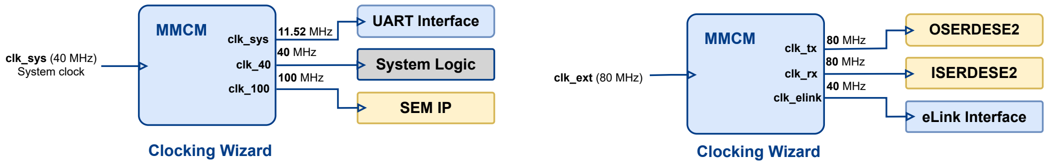
7.1.2 Clock Distribution

The PP3-FPGA manages two distinct clocks within the design: the system clock and the external eLink clock.

Figure 7.2(a) depicts how are these clocks routed from the system clock to other blocks in the design using the Clocking Wizard IP.

The system clock, depicted in Figure 7.2(a), operates at 40 MHz clock generated by a Phase-Locked Loop (PLL) derived from the FPGA's reference clock[1]. The PLL in the FPGA is a subset of the MMCM functionality, serving as a frequency synthesizer for a wide range of frequencies [154, 155]. From this clock, additional clocks are provided to the CAN and SPI interfaces after being scaled down or divided to meet their respective timing requirements.

[1] The reference clock is provided by a 40 MHz clock oscillator [153]



(a) Clocks routed from the system clock, denoted by *clk_sys*.

(b) Clocks routed from the external clock, denoted by *clk_ext*.

Figure 7.2: Overview of the clock distribution in the PP3-FPGA firmware

The external eLink clock, depicted in Figure 7.2(b), is an 80 MHz clock provided by the eLink differential clock. It provides clocking signals to the *eLink interface* and all eLink relevant components OSERDESE2 and the ISERDESE2). The Clocking Wizard IP is used in this domain to scale down the 80 MHz clock into a 40 MHz clock needed for data processing within the interface. While the clock scaled for serialization is kept the same at 80 MHz.

7.2 Development Workflow

As detailed in Section 6.2, the firmware development flow encompasses several critical stages, including behavioral simulation and firmware synthesis. This section delves into the implementation of these steps in the firmware design of the PP3-FPGA.

7.2.1 Firmware Design and Simulation

During the firmware development, the functionality for most of the firmware blocks has been verified with simulations. A SystemVerilog testbench is designed to verify the functionality of the DUT by generating predefined input sequences, capturing outputs, and comparing them to expected outcomes. The testbench consists of various components like generators, drivers, and a monitor class, each performing specific operations like stimulus generation, driving, and monitoring. The simulation environment is based on the HDL simulation tool Mentor Graphics Questasim [156, 157].

Figure 7.3 depicts an abstract view of the MOPS-Hub testbench architecture as detailed in Section 6.2.1. Since the design communicate externally through different protocols. Several agents were instantiated within the testbench environments to covers all possible input combinations of the DUT through its interface. These Agents are listed as follows:

- **CAN Agent** groups the classes specific to CAN communication. Within this agent, the *CAN Frame* generator acts like a master node which generates frames according to CAN standard to the *CAN driver*. The *CAN driver* is a network of MOPS logic connected to the MOPS-Hub CAN buses. It receives the stimulus from the driver and acts on it and respond back to the *CAN Interface*.

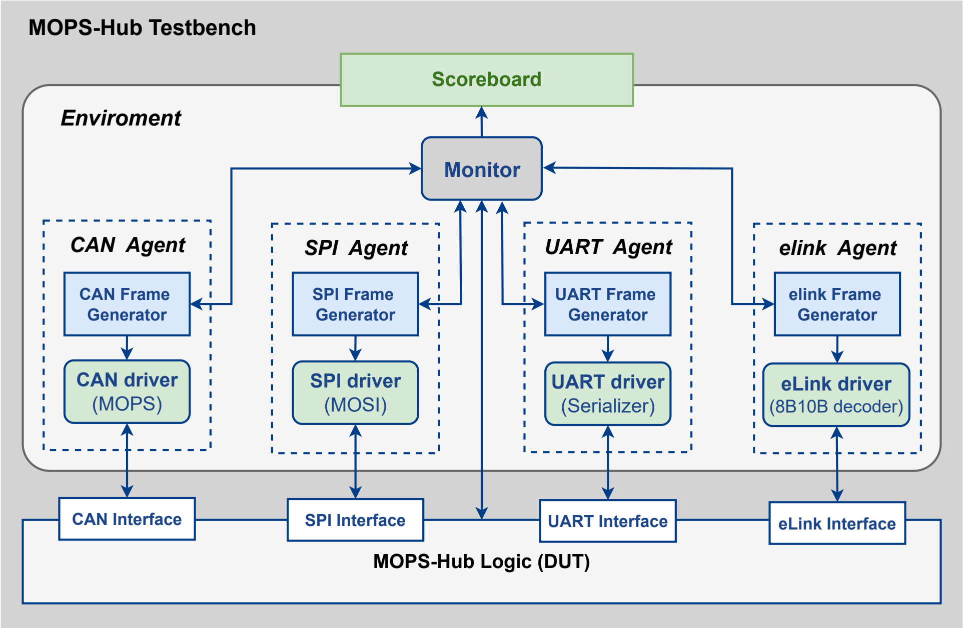


Figure 7.3: Abstract view of the full MOPS-Hub testbench.

- **SPI Agent** groups the classes specific to SPI communication. Within this agent, the *SPI Frame* generator produces predefined stimulus frames to the *SPI driver*. The *SPI driver* acts as an aggregator which serialize the data from/to the *SPI Interface* of the DUT.

- **UART Agent** groups the classes specific to UART communication. The *UART Frame* generator produces predefined stimulus frames to the *UART driver*. within this agent, the *UART driver* acts as a serializer which serialize the data with specific frequency to the *UART Interface* of the DUT.

- **eLink Agent** groups the classes specific to eLink interface. The *eLink Frame* generator produces produces predefined stimulus frames to the *eLink driver*. The *eLink driver* acts as an 8B10B encoder/decoder that aggregate the data with the *eLink Interface*.

All the four agents are Monitored by an advanced Moore FSM, meaning that the output is defined by the current state only. The monitored signals are sent to a *scoreboard* that contains checkers to compare the data with golden reference values and classify the result as Passed/Failed process.

7.2.2 Synthesis and Implementation

The **synthesis process** is achieved using the Xilinx' ISE[2] design tool. The process followed these steps are detailed in Section 6.2.2. As a result of the synthesis flow, a specific bitstream for the PP3-FPGA (**XC7A200T**) is generated.

Table 7.1 lists the resources utilization estimate based on post-implementation during the firmware prototyping on **XC7A200T**-FPGA with/without TMR implemented.

Table 7.1: Resources utilization estimate during prototyping with the **XC7A200T**-FPGA with/without TMR implemented.

Resources	**Available**	**Non-Triplicated**		**Triplicated**	
		Utilization	**%**	**Utilization**	**%**
LUT	133800	22624	16.91	80216	59.95
LUTRAM	46200	261	0.56	267	0.58
FF	267600	15126	5.65	46517	17.38
BRAM	365	10	2.74	9	2.47
IO	285	93	32.63	93	32.63
BUFG	32	11	34.38	11	34.38
MMCM	10	3	30.00	3	30.00

7.2.3 Design Verification

Design triplication is partially automated using the TMRG Toolset, which facilitates the triplication process[158]. However, it requires modification of the RTL source code, potentially leading to unintended changes in the desired design behavior. Therefore, detailed verification becomes essential to ensure that the design remains functional after triplication. Two stages of verification, namely *Static Verification* and *Dynamic Verification*, have been implemented by Lucas Schreiter at Fachhochschule Dortmund - University of Applied Sciences and Arts [159].

Static Verification

The primary objective of *Static Verification* is to ensure that each FF's output is connected to a Voting Cell, thereby ensuring proper triplication of clock and reset domains. A Tool Command Language (TCL) script is deployed for this task within the synthesis tool (e.g., Vivado). The script iterates through the netlist, checking the output connectivity of every FF primitive. It provides insights into critical voting mechanism issues by identifying any bugs or FFs that are not triplicated.

This verification method can detect critical issues at earlier stages without the need for extensive testbench simulations. It is applicable to RTL, post-synthesis, and post-implementation stages, providing insights into the triplication status across various design phases. The script's output can be parsed with a Python tool to automate the

[2] Integrated Synthesis Environment

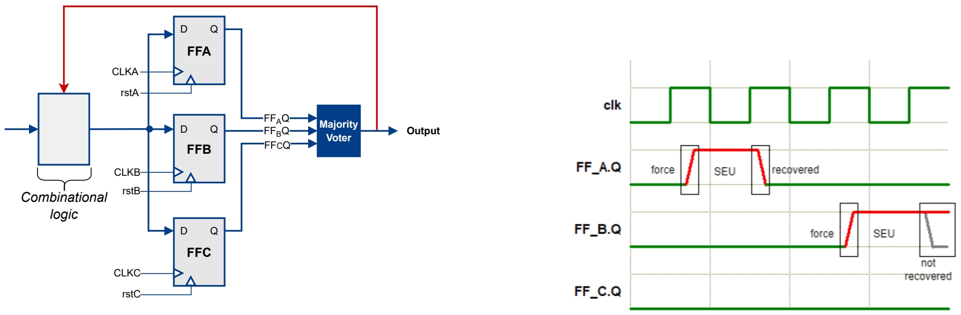
insertion of necessary directives for Triple Modular Redundancy Generator (TMRG) in each Verilog source file, thereby reducing manual effort and potential errors.

Dynamic Verification

The *Dynamic Verification* has two simulation perspectives, namely *Functional simulation* and *SEUs and SETs simulation*.

Functional simulation is conducted to verify that the triplicated design produces the same output as the original design when subjected to the same stimuli. In this stage, assertions are utilized using *assert* command on the triplicated and the non-triplicated design simultaneously to ensure that triplication does not influence the design behavior. If an assertion fails during simulation, it indicates a potential bug or error in the design as shown in Figure 7.4(b).

SEUs and SETs simulation verifies the design's resilience against SEUs and ensures that the triplicated design remains robust under radiation-induced faults. It is conducted on both RTL code and synthesized Netlist. In this stage, the *force* command is used to set a value directly on a signal and override the normal behavior of the design temporarily. This simulates the effect of SEUs in the digital logic. Assertions are employed to observe the state of FFs and check the design behaviour.



(a) Illustration of the partial TMR implementation.

(b) Triplicated signals under *force* and *assert* commands.

Figure 7.4: Generic example for Dynamic Verification.

7.3 The CAN Interface

The CAN communication with the MOPS chips is accomplished through the *CAN Interface* block, which integrates multiple sub-modules to enable simultaneous communication across various CAN buses. It aggregates signals from all connected CAN buses and forwards them to the other blocks for further processing.

7.3.1 Overview

The *CAN Interface* functions as a transceiver, facilitating the transfer of CAN signals between 16 CAN buses and other design components. This interface incorporates the *CANakari* IP Core as the primary CAN-Controller [160].

The data exchange between the CAN buses is managed by the *CAN State Machine*, which initiates CAN communications in response to various interrupt signals from each CAN controller. This ensures that reading and writing operations are executed independently within the design.

7.3.2 CANakari CAN-Controller

The *CANakari* is an ISO 11898 Controller implementation, developed in VHDL and Verilog by teams from FH Köln and FH Dortmund [160]. This IP is also utilized as part of the MOPS digital logic for CAN communication [44]. The *CANakari* IP is controlled through its 16 bit register interface (refer to Table C.2), allowing configuration for sending and receiving messages in accordance with CAN standards [62]. A direct access to the *CANakari* controller is facilitated by a simple parallel interface called *Avalon interface*.

Additionally, the controller issues interrupt acknowledge signals upon successful message transmission or reception, which triggers the firmware to check the controller status.

The *Avalon interface* consists of the control signals: *can_cs*, *read_n* and *write_n* and the data signals: *address*, *readdata*, and*writedata*. In order to read or write to a specific register the related data are set to the correct values before pulling up the *can_cs* signal. Pulling up the *can_cs* enables the controller reading in the provided data ports depending on the process read or write (refer to Table C.3).

Each of the 16 CAN controllers within the firmware is assigned a unique *CAN bus ID* for bus identification. A 16 to 1 Multiplexer identifies the bus ID for each controller, storing the relevant information in a register for subsequent top-level design processing. This synchronization is overseen by a FSM, known as the *CAN State Machine*, which manages the Multiplexer's select lines to ensure the correct bus ID is chosen. The FSM is designed to queue incoming information, accommodating scenarios where multiple CAN controllers receive valid CAN messages simultaneously.

7.3.3 CAN State Machine

The reading/writing process within the CAN interface is facilitated by the *CAN State Machine*. Which has a control over the *Avalon interface* of the *CANakari* controller through its control signals.

Figure 7.5 presents an abstract view of the *CAN State Machine*, with the *IDLE* state as the default post-interrupt state.

Following initialization by the *Central State Machine*, the *CAN State Machine* enters an *IDLE* state, awaiting an interrupt signal from the *Central State Machine* (detailed in Section 7.6). These interrupts indicate either a write access request to a

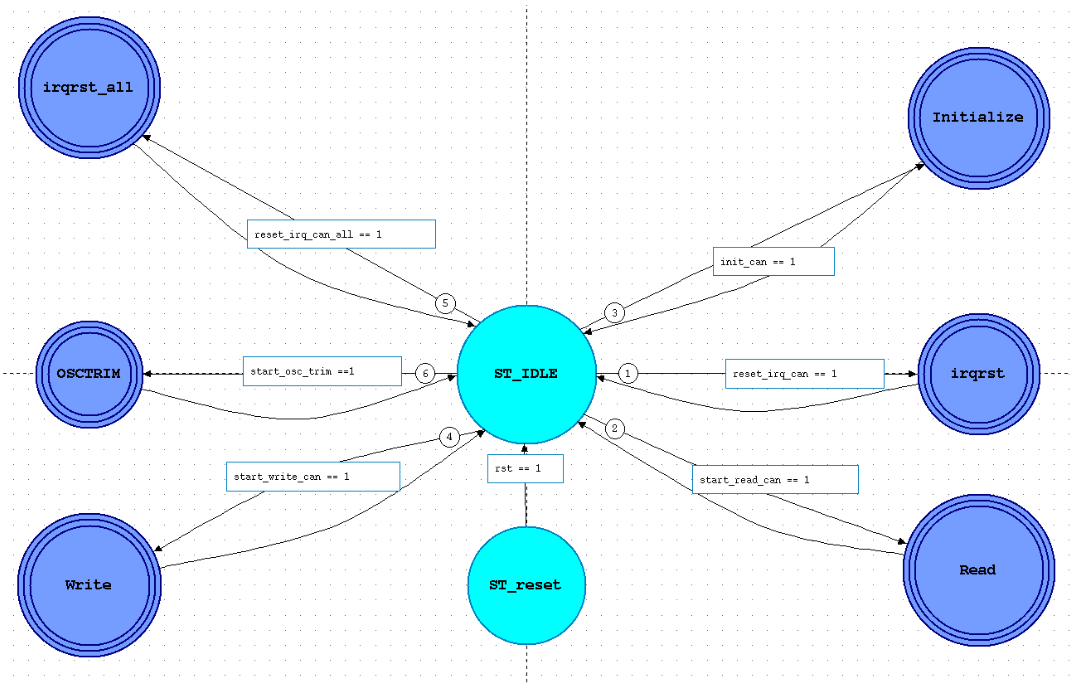


Figure 7.5: CAN State Machine. The dark blue states are hierarchical states which includes other sub-states. All interrupt signals related to CAN communication from the *CAN State Machine* are listed in Table C.4.

CAN bus or a read request from a CAN controller, identified by a 5 bit *address* signal that specifies the *CAN bus ID*. Upon Successful reception or transmission , the FSM reverts to the *IDLE* state.

Figure 7.6 depicts a scenario in which the *CAN State Machine* uses a handshaking mechanism with the help of the *CANakari* control signals to transfer data from a 76 bit *data_tra_downlink* port to the CAN bus.

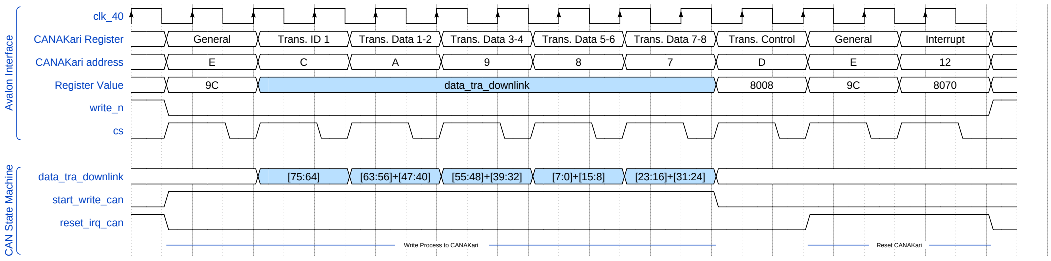


Figure 7.6: Wave form illustrating a write process to the *CANakari* controller. The *start_write_can* signal is asserted to write data to the internal registers detailed in Table C.2.

Figure 7.7 depicts a scenario in which the *CAN State Machine* employs a handshaking mechanism with the *CANakari* control signals to read data from the *readdata* port of the *CANakari*. Depending on the address register, the data from the *readdata* port is allocated to a designated bit slice in the 76 bit *data_rec_uplink* port to construct a complete CAN message.

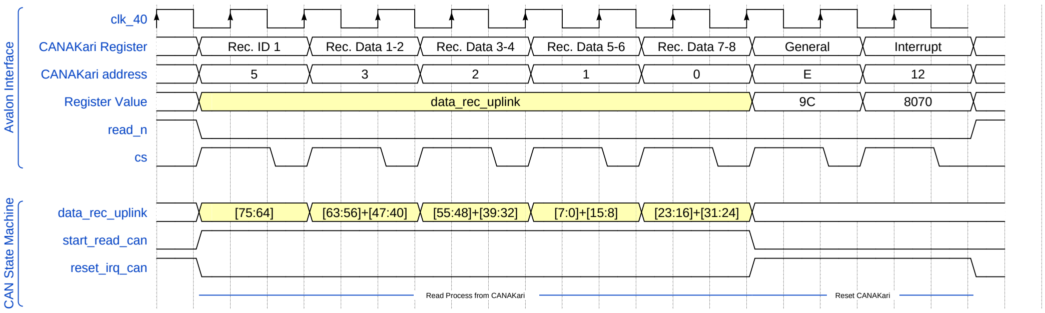


Figure 7.7: Wave form illustrating a read process from the *CANakari* controller. The *start_read_can* signal is asserted to read data to the internal registers detailed in Table C.2.

7.3.4 Data Frame Structure

The CAN frame structure defined for the MOPS chip, depicted in Figure 4.7, is essential for reconstructing the CAN messages within the PP3-FPGA.

The *CAN bus ID* incorporated in the frame is vital for pinpointing the location of each MOPS chip within the detector based on the *CAN bus ID* associated with

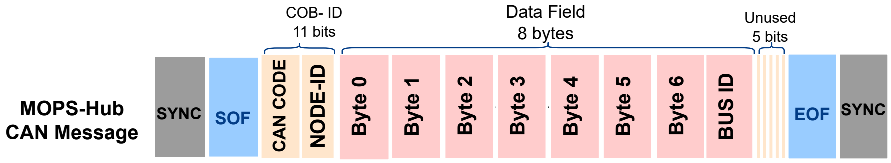


Figure 7.8: eLink data frame structure identifying a CAN Message.

each message. At the DCS level, identifying a specific MOPS chip connected to a particular bus is necessary to request data from that MOPS.

The PP3-FPGA firmware utilizes Byte 7, which is available according to MOPS specifications, to incorporate the *CAN bus ID*. This addition is depicted in Figure 7.8, demonstrating the final data frame structure.

The last 4 bits in the data field are designated as *Unused Bits*. These bits are intentionally included to extend the data stream to an 80 bit boundary, facilitating subsequent processing and ensure proper alignment for 8B10B encoding. The data frame is organized with delimiters for SOF, EOF, and SYNC, conforming to the requirements for eLink transmission as outlined in Section 7.5.

7.4 SPI Interfaces

The SPI communication protocol plays an essential role in the firmware design of the PP3-FPGA due to its dual functionality. Firstly, SPI is instrumental in handling monitoring information per CAN bus with the help of the ADCs on the CIC as detailed in Section 4.5.3. Secondly, SPI facilitates the control of individual power lines, particularly the VCAN within the system. Hence, the PP3-FPGA needs to manage all these incoming SPI signals and integrate them into the eLink data stream. These tasks are accomplished effectively using two interfaces: the *Monitoring Interface* and the *Bus control Interface* detailed in Sections 7.4.2 and 7.4.3 respectively.

7.4.1 Overview

As depicted in Figure 7.1, both the *Monitoring Interface* and the *Bus control Interface* incorporate similar block structure for SPI communication. Consequently, they feature common components such as the *SPI master*, the *De-multiplexer*, and the buffering system (denoted by the *Buffer Encoder*).

At the core of each SPI interface lies the *SPI Master* module, which controls the SPI communication process between the FPGA and the external SPI slaves.

Additionally, the *SPI Master* provides the clock signal to all the SPI slaves. This clock signal is essential for synchronizing data transmission on both the Master Out Slave In (MOSI) and Master In Slave Out (MISO) lines.

In order to manage the SPI data traffic effectively, each SPI line in the interface (16 VCAN or 16 ADCs) is assigned a unique *SPI bus ID*. This ID enables data addressing

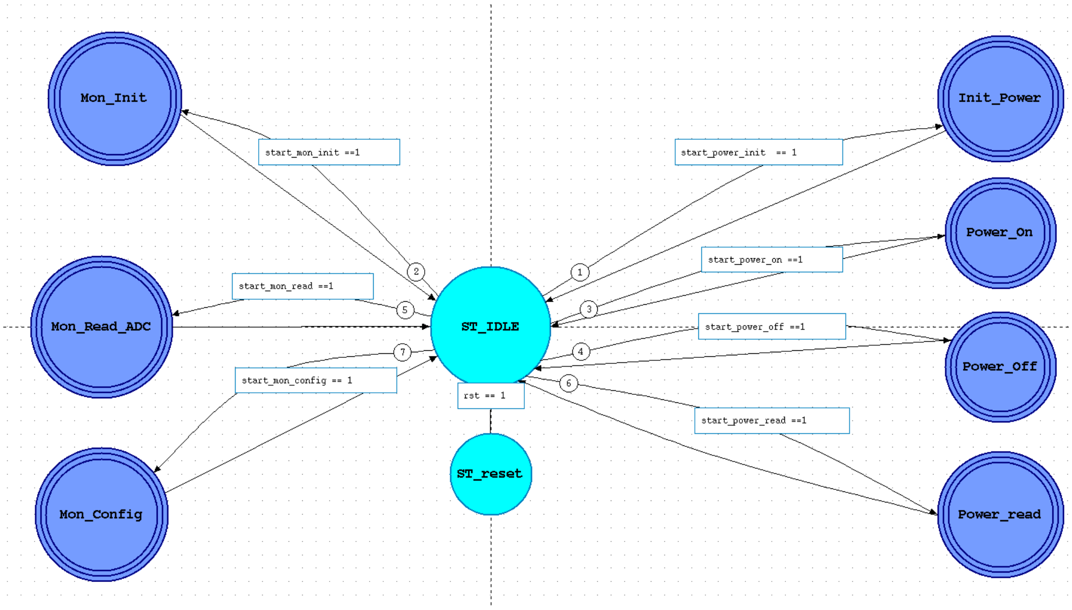


Figure 7.9: SPI State Machine. The dark blue states are hierarchical states which includes other sub-states. All interrupt signals related to SPI communication from the *SPI State Machine* are listed in Table C.5

during transmission. Through the chip select signal, denoted as $\overline{CS}$, the 16-to-1 Demultiplexer in each interface establish communication with the designated SPI slave once the transmission process initiates.

The *SPI State Machine* depicted in Figure 7.9 synchronizes the communication with SPI lines based on the selected *SPI bus ID* and the command received from the eLink side.

7.4.2 The Monitoring Interface

As outlined in Section 4.5.3, monitoring information per CAN bus, such as voltage and current readings of the VCAN, is obtained through the SPI communication protocol from the ADCs situated on the CIC. This function is managed by the *Bus Monitoring Interface* within the PP3-FPGA.

Monitoring Mechanism

The SPI monitoring process is managed by the *SPI State Machine.*

Upon the initialization step from the top level State Machine, the *SPI State Machine* starts configuring each individual ADC on the CIC as recommended by the manufacturer[75]. The FSM subsequently goes into an *IDLE* state, where it remains until an interrupt signal received from the top level indicating a write access request.

Transmission and reception of SPI messages within the module is handled using handshaking mechanism in which several interrupt signals are triggered by the *SPI State Machine*. The specific SPI line is selected based on its *SPI bus ID*. Upon each successful process (reception or transmission), the FSM reverts to the *IDLE* state. The data aggregated from the *SPI master* is stored in an appropriate buffer location before being sent to the *eLink Interface*.

Data Frame Structure

Each 16 bit ADC situated on the CIC has four 24 bit Channel Setup Registers (CSRs). Each CSR contains two 12-bit setups, programmable through SPI communication, to contain data conversion or calibration information [75].

Figure 7.10 depicts the serial sequence required to write to, or read from the serial port's registers implemented in the CS5523 ADC. Reading data from one ADC, requires transmitting the appropriate write command which accesses one of the four CSRs followed by 24 bits of data. In response, the ADC provides the monitoring information of that specific CSR, which is read by the SPI master [75].

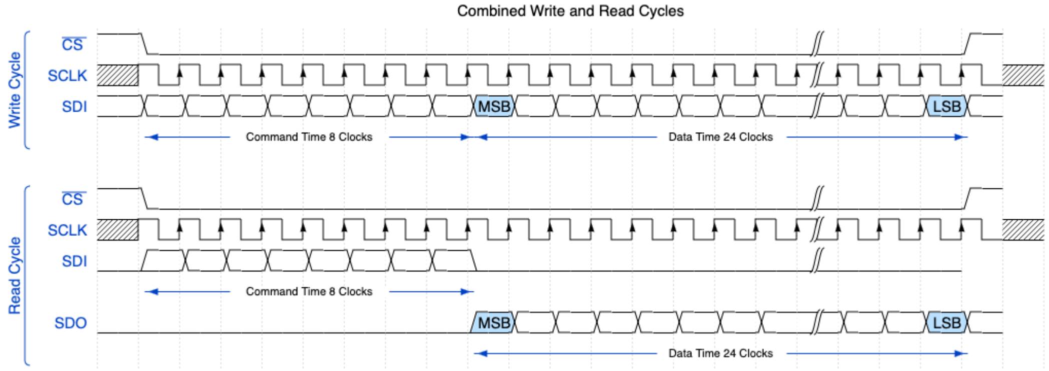


Figure 7.10: Command and data word timing of the CS5523 ADC, based on [75].

This reading/writing process is managed within the *SPI Interface* using a buffer management mechanism through the *Buffer Encoder* module. This mechanism comprises two buffering stages. The first stage of buffering includes storing the requested message delivered from the eLink to read the ADC data. The request message includes the *SPI bus ID*, the ADC register (*SPI register*), and a unique identifier (*SPI CODE*) to distinguish SPI requests within the design and assign the collected SPI data as detailed in Table 7.3. Subsequently, in the second stage of buffering, the responses from each ADC based on their respective addresses are stored in the buffer's data field. Once the reading process concludes, the eLink interface retrieves the data stored in the 80 bit SPI buffer register for further processing.

Figure 7.11 depicts the monitoring data structure received from the *Buffer Encoder* module. The 12 bit field (*XADC*), depicted in the figure, contains additional information about the local monitoring data collected by the XADC IP. This data

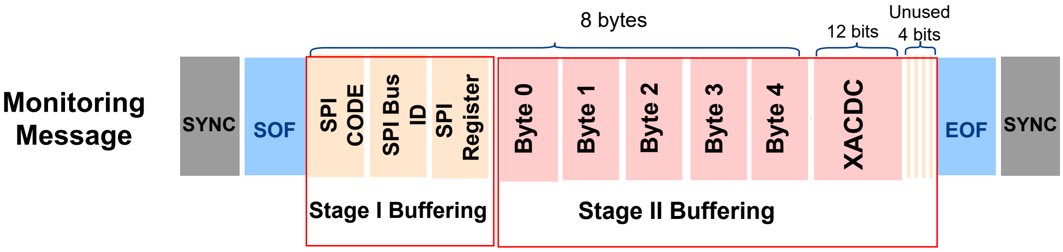


Figure 7.11: eLink data frame structure identifying an SPI monitoring message.

is independent from the SPI interface and is sampled from on-chip sensors for temperature and power monitoring [161]. The last 4 bits in the data field are *Unused Bits* added intentionally to align the data stream to the 80 bit boundary for further processing using 8B10B encoding. The frame is structured with delimiters for SOF, EOF, and SYNC, as specified for eLink transmission in Section 7.5.

7.4.3 The Bus Control Interface

As outlined in Section 4.5.2, the management of individual power lines, particularly the VCAN, is facilitated through an 8 bit register chip[3] mounted on the CIC. Controlling the chip via SPI communication allows for the activation or deactivation of specific power lines.

The *Control Interface* module is designed to manage the power of connected CAN buses through SPI communication. Controlling the 8 bit register chip is exclusively through SPI communication. Consequently, the status of VCAN remains unaffected by the operational states of VCAN-PSU and VPP3, and it remains unchanged until it receives specific activation commands from the PP3-FPGA. This mechanism ensures that even if the PP3-FPGA undergoes a power cycle for any reason, the status of VCAN remains unaffected.

Bus Control Mechanism

Similar to the *Monitoring Interface* module discussed in Section 7.4.2, the synchronization of the VCAN SPI lines is managed by the *SPI State Machine*. Which configures the 8 bit register chip in an early stage of initialization before moving to its *IDLE* state.

Upon reception of a power enabling message from the top level, the FSM makes a transition to a state where it selects the SPI line based to its *SPI bus ID* address. The FSM activates the slave by enabling the $\overline{CS}\_control$ signal and initiates communication by sending the 10 kHz clock signal, denoted to the slave.

Transmission and reception of SPI messages within the module is handled using handshaking mechanism in which several interrupt signals are triggered by the *SPI*

[3] MCP2308 chip

State Machine. Upon each successful process (reception or transmission), the FSM reverts to the *IDLE* state.

Data Frame Structure

The 8 bit register chip situated on the CIC consists of multiple 8 bit configuration registers for input, output and polarity selection [73].
Figure 7.12 depicts the serial sequence required to write to, or read from the configuration registers implemented in the MCP23S08 register chip. The SPI write/read operation includes transferring a control Byte, denoted by *device opcode*, followed by the register address and at least one data byte. The control Byte contains five fixed bits and two user-defined hardware address bits (A1 and A0), with the R/W bit to define the write/read process. During communication, the appropriate write command which accesses these registers should be transferred while lowering $\overline{CS}$ signal [73].

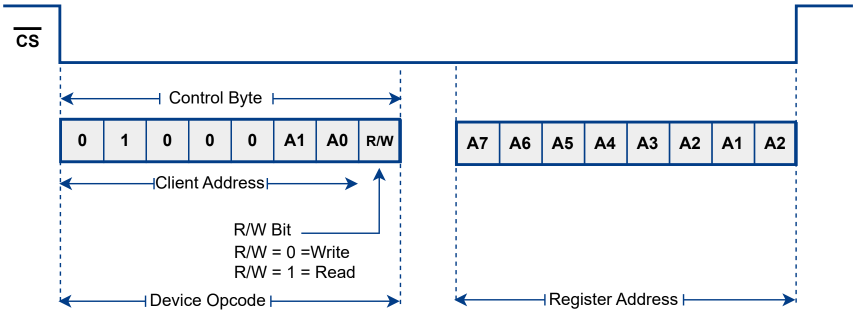


Figure 7.12: SPI addressing registers in the MCP23S08 register chip, based on [73].

This reading and writing process is managed within the *Bus Control Interface* using the same buffer mechanism described in Section 7.4.2.

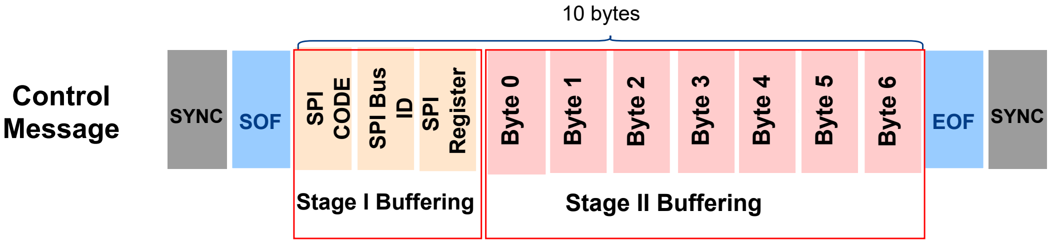


Figure 7.13: eLink data frame structure identifying an SPI bus control message.

Once the control process concludes, the eLink Interface retrieves the data stored in the 80 bit frame, depicted in Figure 7.13, as received from the *Buffer Encoder* module. The frame is structured with delimiters for SOF, EOF, and SYNC, as specified for eLink transmission in Section 7.5.

7.5 The eLink Interface

Data collected from various modules within the PP3-FPGA is transmitted across eLink differential lines to the EMCI at a frequency of 80 MHz as detailed in Section 4.7.

To ensure direct DC balance and maintain bounded disparity within the eLink data stream, the 8B10B encoding/decoding scheme is utilized for data transmission across eLink lines. This technique converts each 8 bit word of the data into 10 bit transmission symbols through serialization [66], as detailed in Section 4.3.2. This data undergoes encoding prior to transmission and requires decoding upon receiption at the EMP end. The EMP will be tasked with managing this encoding/decoding process before the data is ultimately delivered to the DCS.

The *eLink Interface* bridges the gap between the eLink lines and other PP3-FPGA modules, utilizing the *eLink Transmitter* and *eLink Receiver*, as elaborated in Sections 7.5.1 and 7.5.2.

7.5.1 The eLink Transmitter

The *eLink Transmitter* plays a crucial role in the communication path within the PP3-FPGA system. It collects the data from various interfaces, such as the CAN controller and SPI interface, and serialize it via the eLink transceiver.

Figure 7.14 depicts a block diagram illustrating the data path of the transmitter within the PP3-FPGA system.

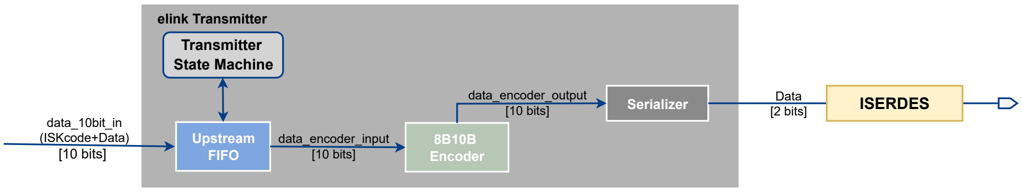

elink Transmitter
Transmitter State Machine
data_10bit_in (ISKcode+Data) [10 bits]
Upstream FIFO
data_encoder_input [10 bits]
8B10B Encoder
data_encoder_output [10 bits]
Serializer
Data [2 bits]
ISERDES

Figure 7.14: eLink Transmitter, encoding stage: The data coming from the 8B10B encoder is multiplexed out on 2 bit port for serialization.

As depicted in Figure 7.14, the data received from various interfaces is stored in internal registers within the system. These registers act as temporary storage for the incoming data (discussed further in Section 7.5.5). Each piece of data stored in these internal registers is assigned a unique ID which helps in identifying the data as it progresses through the processing pipeline as detailed in Table 7.3. In this stage, 2 bit data flags, called *Internal Synchronization Keys (ISKs)*, are assigned to each 8 bit word as defined in Table 7.2 to delineate the boundaries of each data frame. Subsequently, the data is encapsulated between delimiters, indicating the SOF and EOF before being buffered in the *Upstream FIFO* for later processing.

The 8B10B Encoder

The *8B10B encoder* transforms each 8 bit byte of data stored in the FIFO into a 10 bit transmission code. This encoding process occurs periodically as long as the FIFO contains data.

The 10 bit encoded data is then serialized in packs of 2 bit using a multiplexer over 5 clock cycles of a 40 MHz clock before being sent to serialization.

The data delimiters defining the SOF and EOF are chosen out of twelve *K-characters* used in 8B10B coding with a unique bit sequence as defined in Table 7.2. These symbols are also recognizable by the receiver to ensure accurate frame alignment and to signal the data boundaries.

Function	**K-character**	**Decoded Representation**		**Encoded Representation**	
		ISK code		RD = -1	RD = +1
SYNC	K.28.5	11	10111100	110000 0101	001111 1010
SOF	K.28.1	10	00111100	110000 0110	001111 1001
EOF	K.28.6	01	11011100	110000 1001	001111 0110

Table 7.2: K-words of special interest as used in the 8B10B encoding/decoding.

The special character (K.28.5) serves as a SYNC idle symbol to aid alignment during transmission. This SYNC symbol is periodically sent over the eLink bus in the absence of data to facilitate byte and word synchronization at the receiver side, as discussed in Section 7.5.2.

7.5.2 The eLink Receiver

The *eLink Receiver* module plays a crucial role in the data reception process, particularly in decoding and processing the asynchronous data stream received from the eLink lines. As depicted in Figure 7.15, the data received at the input of the serializer (discussed in Section 7.5.3) is sent automatically to the *deserializer* block.

In the *deserializer* block, the incoming data stream is shifted by two bits on the rising edge of a 40 MHz clock. Though each 8B10B symbol has a width of 10 bits, the output of the *deserializer* has a width of 12 bits. The *sync detector* module will search for the SYNC in 12 bits data and correctly align the subsequent operations to the 8B10B symbols on the pipeline. Subsequently, an acknowledge signal will go high to enable the *8B10B decoder* in the subsequent step.

The 8B10B Decoder

The *8B10B Decoder* is responsible for decoding the synchronized data stream and converting the encoded symbols back into their original 8 bit format. Upon receiving an acknowledge signal from the *deserializer*, the 8B10B decoder begins decoding each 10 bit symbol received.

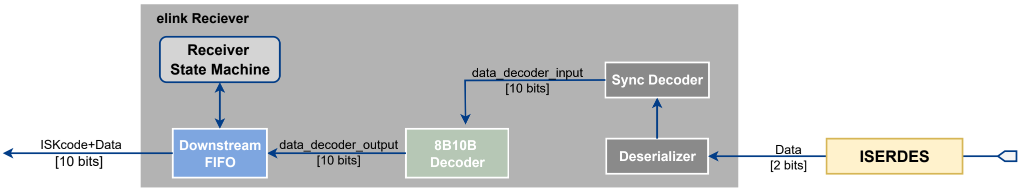


Figure 7.15: Receiver, decoding stage: The data is synchronized, deserialized and aligned to 8B10B symbols.

To ensure accurate decoding, the decoder has to start at the same RD as the encoder. In case of an error, it's crucial to recover RD received. This is facilitated by the sync detector, which utilizes the disparity of the SYNC symbol to adjust the *8B10B Decoder* RD to the adjust its value every time such a symbol is received.

In a subsequent stage, the synchronized data is sent to the *downstream FIFO* which in return raise an acknowledge signal indicating a reception of an error free data frame. The information regarding the edges of the data frame based on the *K characters*, summarized in Table 7.2, will be then extracted. Once a SOF has been read, all following data bytes will be written to the targeted Interface based on the unique ID defined in it, while the EOF code will indicate the end of the data frame.

7.5.3 The eLink Serializer

The *Elink Serializer* module in the FPGA provides all the essential building blocks required for serializing data over the eLink. The module utilizes the OSERDESE2 and ISERDESE2 to interface with the high speed SSTL, which are commonly used in high-speed memory interfaces. Detailed descriptions of these IPs can be found in [117].

As depicted in Figure 7.14, the OSERDESE2 serializes the parallel data received from the *eLink Transmitter* into a serial stream for transmission over a differential pair. The serialization rate is determined by the externally provided clock frequency from the eLink side, typically set at 80 MHz (refer to Section 7.1.2). The sampling clock is a 40 MHz clock that is derived from the 80 MHz clock to guarantee a perfect phase alignment during serialization. During bit serial transmission, the LSB is transmitted first.

On the receiving end, the ISERDESE2 de-serializes the serial stream received over the differential pair from the eLink into parallel data. This deserialized data is subsequently forwarded to the *eLink Receiver*. The de-serialization process also operates at the 80 MHz clock provided by the eLink side.

7.5.4 eLink State Machines

The reading/writing process within the eLink interface is facilitated by two FSMs, the *eLink receiver State Machine* and the *eLink Transmitter State Machine* respectively.

Following initialization by the *Central State Machine*, the *eLink State Machines* enter an *IDLE* state, awaiting an interrupt signal from the other FSMs based on the *bus ID*. These interrupts indicate either a write access request to the *eLink Transmitter* or a read request from the *eLink Receiver*. Table C.6 lists all interrupt signals controlling the *eLink State Machines*.

7.5.5 Data Frame Structure

To ensure perfect synchronization and frame alignment, the DC-balanced encoded data is framed with unique delimiters, denoted by SOF, EOF and SYNC symbols. The structure of the final eLink data frame in the PP3-FPGA firmware, incorporating these delimiters, as depicted in Figure 7.16.

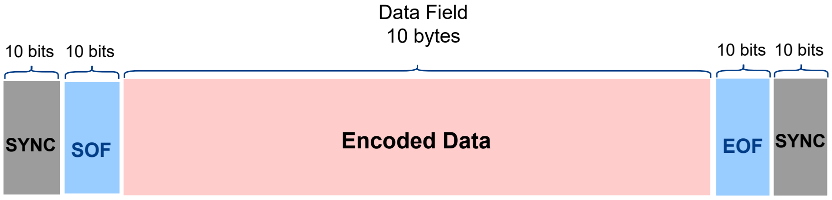


Figure 7.16: eLink data frame structure based on 8B10B encoding. The typical frame structure within the PP3-FPGA firmware carries 10 bytes of payload data.

The typical frame structure within the PP3-FPGA firmware carries 10 bytes of payload data, representing the data collected from different interfaces embedded within the design. As discussed in previous sections, MOPS-Hub interacts with a range of data from different sources, utilizing various communication protocols such as CAN and SPI. To effectively manage this diversity, data within the *eLink Interface* is categorized based on Unique Identifiers (IDs). This categorization allows the system to process different message types efficiently. The assignment of IDs to their respective message types, along with the distinct data frames for communication across interfaces, is depicted in the table 7.3.

Table 7.3: Data Frame Dictionary in PP3-FPGA firmware. Each data frame is assigned with a Unique Identifier(ID).

Message Type	**HEX ID Code**	**Bit Size**	**Description**
CAN Message	600_h+Node ID	12 bits	SDO write CAN request.
CAN Message	580_h+Node ID	12 bits	SDO read CAN response.
CAN Message	700_h+Node ID	12 bits	MOPS Sign-In message.
CAN Message	000_h	12 bits	Restart MOPS command
MOPS-Hub Sign-In	40_h	8 bits	MOPS-Hub Sign-In message.
SPI Control	31_h	8 bits	Powering ON CAN bus .
SPI Control	30_h	8 bits	Powering OFF CAN bus.
SPI Monitoring	20_h	8 bits	Request PP3 Monitoring Information.

7.5.6 Simulation

Figure 7.17 illustrates a simulated waveform of the encoding/decoding process within the PP3-FPGA firmware. The data received on an internal 76 bit *data_rec_uplink* port,collected from the CAN controller , is packed into 8 bit frames after assigning ISK code to it as referred in Section 7.5.1.

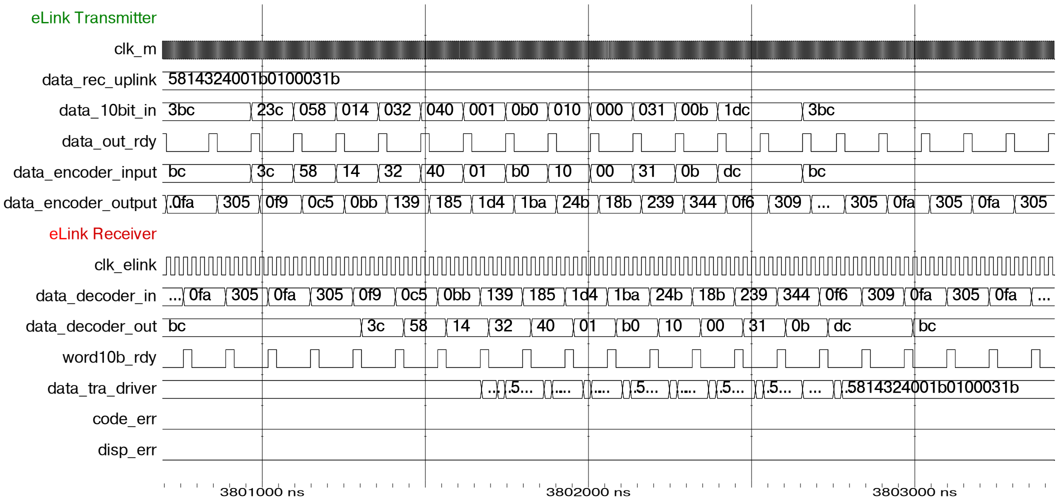


Figure 7.17: Simulation results for encoding/decoding process within the PP3-FPGA firmware based on the testbench detailed in Figure 7.3.

The encoding process requires 1.692 µs as simulation shows.

To maintain a neutral average disparity, the encoding scheme ensures that a positive running disparity is always followed by negative disparity, and conversely, any negative running disparity is always followed by positive disparity. If these conditions are not met, the decoder flags an error by asserting its *disp_err* output.

Upon reception, the data received by the eLink Receiver undergoes scanning by the *sync_decoder*, as referred to in Section 7.5.2. Similarly, the decoding process requires 2 µs as simulation shows.

7.6 Central Finite State Machine

All embedded FSMs in the design embody the characteristics of the Moore FSM concept, where outputs are state-dependent. The *Central Finite State Machine* is the principal governor of the PP3-FPGA firmware design, managing its overall operation and distributing tasks to other state machines within the logic using a handshaking mechanism. In the following sections, a detailed description of the functionality of FSM is discussed.

7.6.1 Overview

The *Central Finite State Machine* operates in three distinct phases depicted in Figure 7.18. The first phase, the *Initialization Phase*, involves initializing all other state machines. This includes configuring various interfaces and initializing different blocks within the system. This early initialization step occurs immediately upon system startup.

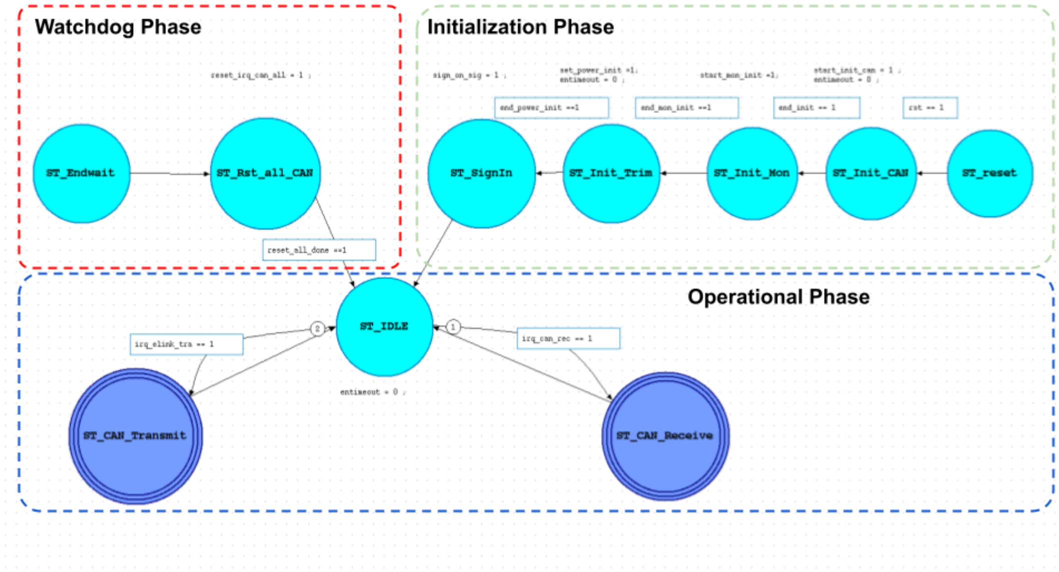


Figure 7.18: The Central Finite State Machine.

The second phase, the *Operational Phase*, continues throughout the system's operation. In this phase, the FSM acts as a bridge between the CAN interfaces and the eLink interface.

The *Watchdog Phase*, governed by the *Watchdog Timer*, serves as an additional safeguard. In this phase, the FSM is compelled to revert to the *IDLE* state if it becomes non-responsive due to a design malfunction. This scenario is discussed in detail in Section 7.7.

7.6.2 Initialization Phase

Figure 7.19 depicts the initialization phase of the *Central Finite State Machine* after power upon the system. Up on startup, the central FSM configures the 16 CAN controllers by issuing specific configuration commands to it. This process is managed in collaboration with the *CAN State Machine* by applying a predefined timing information for pre-scaling each *CANakari* controller. This information involves pre-scalar configurations and other general registers for the time scaling and the bus parameters introduced in Section 4.3.1. This step is essential to define the CAN bus parameters like frequency and time stamps for later communication with the MOPS chips [160].

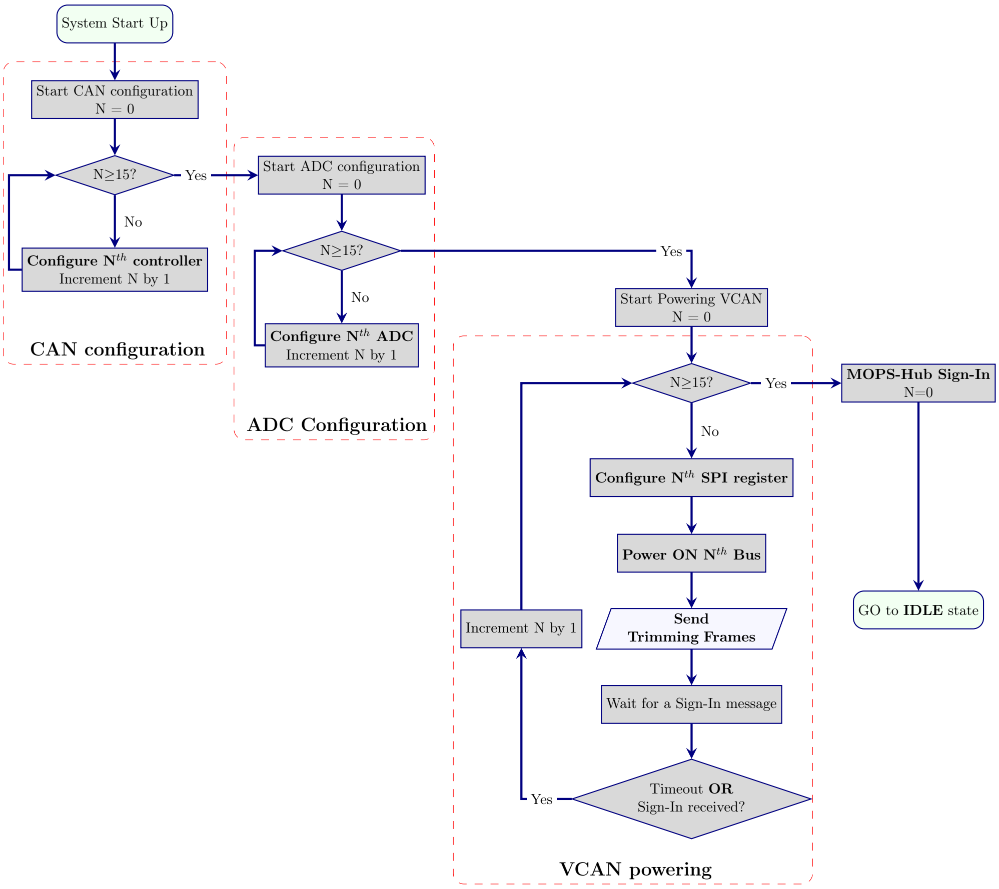


Figure 7.19: Initialization phase of the **Central State Machine**.

Following CAN configuration, the FSM activates the CSRs of the monitoring ADCs situated on each CIC. This process is managed in collaboration with the *SPI State Machine*. The *Central Finite State Machine* then proceeds to power the individual VCAN lines as described in Section 7.4.3.

Upon power activation, the PP3-FPGA starts transmitting several trimming messages to the bus in order to fine-tune the oscillator frequencies of the MOPS chips. This process is required directly after configuring the SPI control registers on the CIC and powering the VCAN lines before the connected MOPS chips are out of the trimming mode [44].

Post-trimming, each MOPS chip connected to the bus should load its configuration and broadcasts a sign-in message at the nominated bus frequency 125 kbit/s. The MOPS-Hub aggregates these sign-in messages and forwards them through the eLink interface. Absence of a sign-in message triggers the *Watchdog Timer*, preventing FSM lock-up by resetting the state to *IDLE* and proceeding to the next CAN bus. This cycle repeats until all CAN buses are active and the *SPI State Machine* is back to its *IDLE* state.

Finally, the *Central Finite State Machine* sends an acknowledge signal to the *eLink Transmitter* block. Consequently, the *eLink Transmitter State Machine* sends a sign in message carries the MOPS-Hub-ID through eLink to the DCS indicating the system's readiness.

7.6.3 Operational Phase

After the initialization phase discussed in Section 7.6.2, the *Central Finite State Machine* transitions to an *IDLE* state. Here, it remains responsive, awaiting interrupt signals from either the CAN side or the eLink side. This stage continues throughout the system's operation as depicted in Figure 7.20.

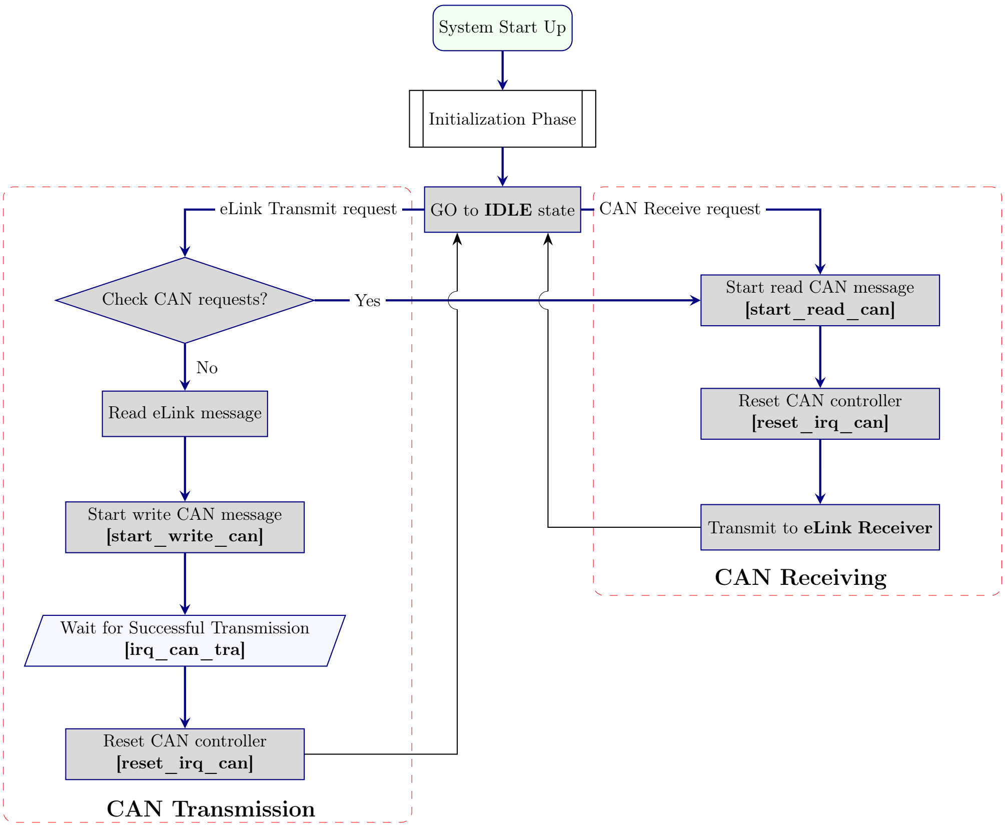


Figure 7.20: Operational phase of the *Central State Machine*.

Upon receiving a transmit request from the eLink side, the FSM will check if the CAN side has any messages to read, and if not, it makes transition to a state where it reads the eLink message and selects the CAN controller according to its *CAN bus ID*. Subsequently, the FSM proceeds to write the data information to the *Avalon interface* with the help of the *CAN State Machine* (detailed in Section 7.3.3). Once the data is successfully transmitted to the bus, an interrupt acknowledge signal, denoted by (*irq_can_tra*), goes high successfully. As a response, the FSM will reset the controller to bring the interrupt signal back to low then revert to the *IDLE* state.

Upon reception of a CAN message on one of the CAN buses, an interrupt acknowledge signal (*irq_can_rec*) will be triggered by the FSM. The FSM then proceeds to read out the identifier and data associated to it. Finally, the FSM resets the controller to bring the interrupt signals to low before reverting back to the *IDLE* state.

When it comes to message prioritization, the *Central Finite State Machine* aggregate messages received from the CAN interface irrespective of their Node ID. This approach is taken because the prioritization of one CAN message over another is managed efficiently by the *CAN controller* itself.

However, the FSM is designed to prioritize receiving requests from the CAN side over transmission requests from the eLink side. This means that in instances where both requests are active simultaneously, the state machine will prioritize the receiving request from the CAN side first. The *Downstream FIFO* implemented in the *eLink Receiver* block assist in buffering the messages received from the eLink side for a certain amount of time before they reach capacity, depending on the data rate. This guarantees that no message requests from the DCS are lost while processing others from the CAN side.

7.6.4 Watchdog Phase

The initialization of the *Watchdog Phase* is mainly managed by the *Watchdog Timer* discussed in Section 7.7.

7.7 Watchdog Timer

The *Watchdog Timer* monitors the occurrence of timeout events from different FSMs, each denoted by a unique signal. The signals are OR'ed together so that If any become active (high), a timeout signal is activated. A timeout condition, denoted by *rst_timeout*, is activated if a predetermined timeout period (2 s) elapses.

7.7.1 Watchdog Timer Mechanism

The Watchdog timeout mechanism is depicted in Figure 7.21.

After Power cycling, all the FSMs enter the *IDLE* state. The *Watchdog Timer* is initialized at this stage. If any of the FSMs remains in a state other than *IDLE*, the watchdog counter increments with every clock edge until the counter reaches a predefined limit (2 s sampled at 40 MHz clock). Once the predefined limit is reached, the watchdog sends a soft reset signal which resets the counter and automatically recovers the FSMs by bringing it back into the *IDLE* state. This mechanism effectively resolves most reversible errors without requiring an external reset or power-cycle.

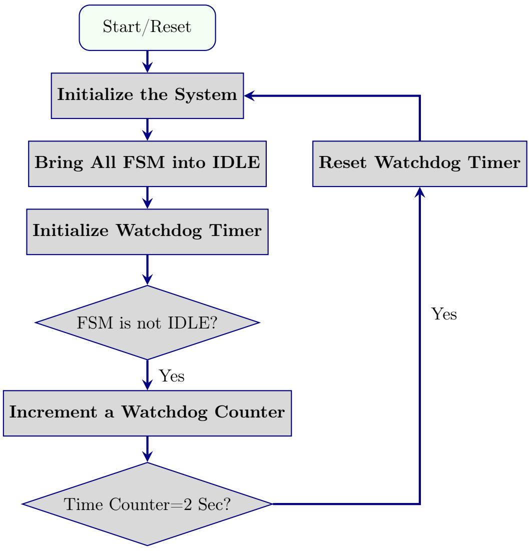


Figure 7.21: *Watchdog* timeout mechanism.

7.7.2 Simulation

Figure 7.22 depicts the simulation of the *Watchdog* timeout mechanism. Three timeout signals are tracked different FSMs during CAN communication. During simulation, an interrupt signal from the *CAN Interface* was deactivated. Consequently, the timeout signals from two FSMs (*enable_timeout1* and *enable_timeout0*) were activated around ≈ 2.08 s. The *rst_timeout* signal is then generated to trigger a soft reset signal, denoted by *recovery_signal*, for all the state machines. Finally, the Watchdog counter is reset once the FSM reverts to the *IDLE* state.

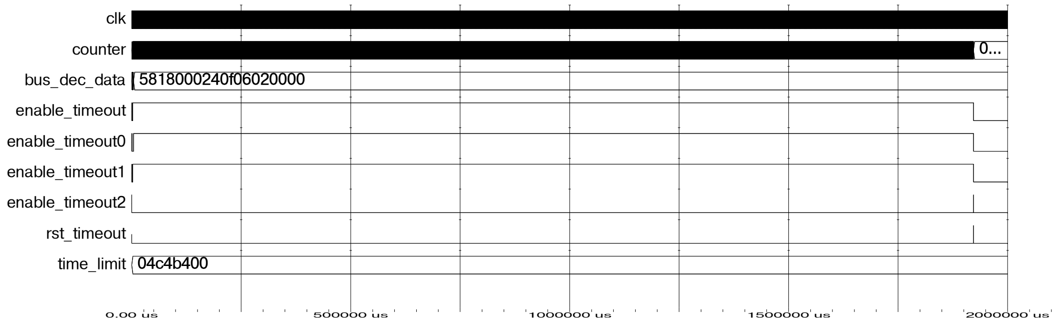


Figure 7.22: Simulation of timeout signals in the *Watchdog Timer* module based on the testbench detailed in Figure 7.3.

7.8 UART Debugger

In addition to the essential firmware modules for PP3 monitoring, power control and CAN communication, the firmware design incorporates a specialized UART block independent of the eLink/CAN interfaces. This module aims to monitor the system's status during integration or testing phases for debugging purposes.

7.8.1 Overview

The *UART Debugger Module* depicted in Figure 7.23 is designed to establish a UART communication protocol between the PP3-FPGA and any external debugging interface. This module enables the observation of internal registers crucial for data transmission. Additionally, it tracks the status of FSMs.

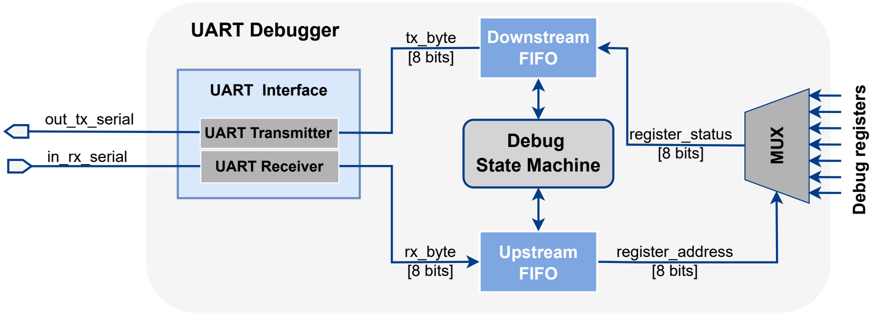


Figure 7.23: *UART Debugger* Module.

The main functionality of the *UART Debugger* is achieved through the instantiating of several sub-module instances. These sub-modules include the *UART Receiver*, *UART Transmitter*, *Debug State Machine*, *FIFO Buffers*, and the *Multiplexer*. Each sub-module contributes to specific aspects of UART communication, such as data reception, state management, and data transmission.

7.8.2 Debugging Mechanism

Debugging mechanism in the *UART Debugger module* depends on the requests received from the user on the UART port, denoted by *in_rx_serial*. The request/response process is handled within the module using the *Debug State Machine*. Which is always in an *IDLE* state waiting for an interrupt signal from the *Upstream FIFO* indicating the reception of a debug request.

7.8.3 Simulation

Figure 7.24 depicts the simulation of the debugging request/response process. A debug request is an 8 bit data, denoted by *rx_ byte* received from UART interface carries the address of the internal registers under request.

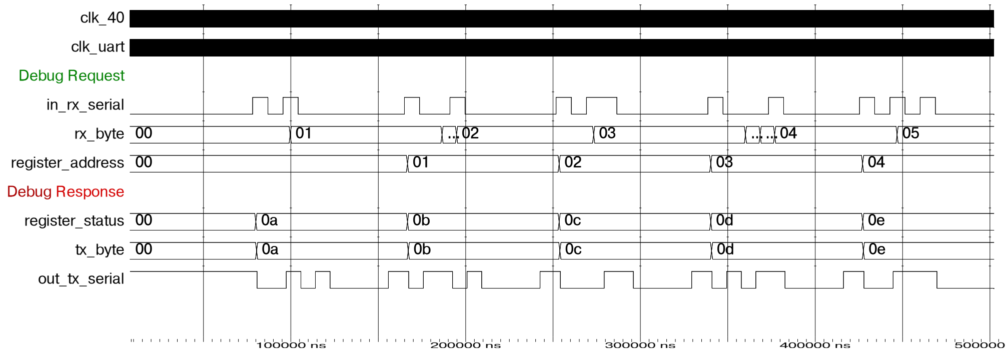


Figure 7.24: Simulation of the request/response of the debug signals in the *Debugger Module* based on the testbench detailed in Figure 7.3.

Subsequently, the FSM will make transition to a state where it selects the register according to its address, denoted by *register_ address*. In response, the *Multiplixer* will push the register status on its output, denoted by *register_status*. Finally, the FSM writes this data to the *Downstream FIFO* which serializes it to the *out_ tx_ serial* port using the *UART Transmitter*.

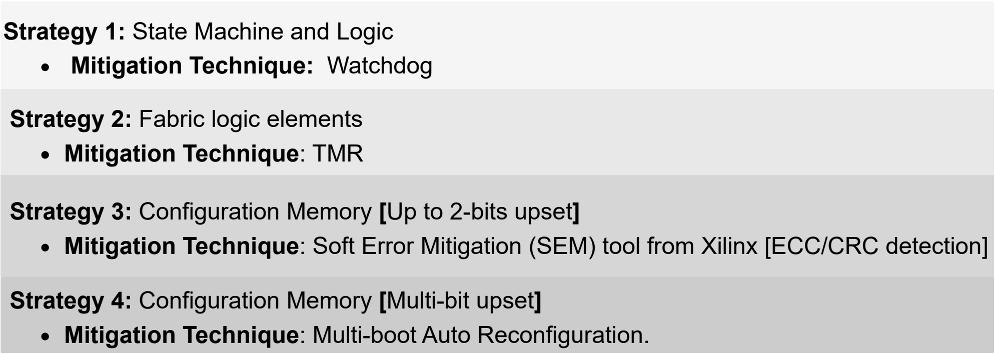
7.9 SEU Mitigation in MOPS-Hub

The complex nature of FPGAs, with multiple vulnerable sections, presents a challenge for SEU mitigation strategies. To address this challenge, a multi-level SEU mitigation technique has been implemented within the PP3-FPGA firmware design to enhance the robustness of the PP3-FPGA in the demanding environmental conditions of the PP3 location.

Figure 7.25 illustrates the radiation-hardening strategies categorized by their specific applications. Each of these strategies will be discussed in detail in the following sections.

7.9.1 Strategy 1: State Machine and Logic

This part has already been discussed in Section 7.7, in which the *Watchdog Timer* module continuously monitors the status of all the FSMs and sends a soft reset signal to recover the FSMs whenever any of them is stuck in a locked state for more than a predefined time. This mechanism effectively resolves most reversible errors without requiring an external reset or power-cycle.



Strategy 1: State Machine and Logic

- **Mitigation Technique:** Watchdog

Strategy 2: Fabric logic elements

- **Mitigation Technique**: TMR

Strategy 3: Configuration Memory **[**Up to 2-bits upset**]**

- **Mitigation Technique**: Soft Error Mitigation (SEM) tool from Xilinx [ECC/CRC detection]

Strategy 4: Configuration Memory **[**Multi-bit upset**]**

- **Mitigation Technique**: Multi-boot Auto Reconfiguration.

Figure 7.25: Radiation hardening strategies for the PP3-FPGA.

7.9.2 Strategy 2 : Fabric Logic Elements Protection

This level directly employs TMR to mitigate SEU in the fabric logic elements. TMR implements three identical copies of each critical element along with a voting mechanism to mask errors caused by SEUs affecting individual bits (see Section 6.5.1). While some SEUs might induce errors in the data stream, the redundant data allows the voting process to accurately determine the correct value, enhancing overall system reliability. To aid in the implementation of TMR, the TMRG tool from CERN is used [162]. The TMR implementation is done by Lucas Schreiter at FH Dortmund - University of Applied Sciences and Arts.

Implementing TMR increases resource utilization on the FPGA. As detailed in Table 7.1, triplicating storage elements, combinational logic, clock, and reset nets leads to a higher footprint compared to non-TMR approaches.

7.9.3 Strategy 3: CRAM Protection

Triplicating the main design functionality effectively minimizes unsafe behavior during SEU events. However, it's still possible for multiple SEUs to occur in critical CRAM bits, potentially altering the design's behavior (as explained in Section 6.1). While this approach minimizes risk, this level adds a further mitigation layer.

Two strategies were considered: either CRC and ECC are detected and corrected, or the entire configuration is reconfigured. These strategies will be covered in the following sections.

2-bits upset

The *Enhanced Repair* mode of the SEM IP described in Section 6.6.3 is used in the design to correct single- and double-bit errors. With Readback CRC feature enabled, the configuration logic continuously reads back the CRAM contents in the background, calculating the CRC per frame. The CRC value is compared with the expected value. Subsequently, the instantiated *ICAP* primitive within the SEM IP automatically corrects the error and rewrites the corrected data back to the affected frame. This ensures continuous operation and data integrity even with occasional bit flips in the CRAM. The Readback CRC settings specified in the Vivado constraints File during synthesis are listed in Sections C.2.1.

Multi-bit upset

Even with TMR and SEM IP control, there is still a possibility that MBU affecting more than two bits per frame simultaneously, potentially disrupting protection mechanisms from other strategies. Additionally, certain components of the FPGA, such as hard IP cores, either cannot be triplicated by the designer or lack resources for direct SEU mitigation. The accumulation of upsets in these areas can lead to functional failures.

To address SEU effects in these scenarios, an additional Multi-boot Auto Reconfiguration (mBAR) strategy was implemented on the PP3-FPGA. This strategy involves mapping multiple copies of the PP3-FPGA firmware design to different addresses within the external flash memory [163, 164].

In this strategy, the golden image of the bitstream, stored at address 0x0, contains all the necessary registers for multi-boot triggering, such as the *IPROG* command and the *WBSTAR* register. Fallback images are stored at higher addresses in memory and can be dynamically triggered by the FPGA. All settings specified in the Vivado constraints file during synthesis for MBU are detailed in Section C.2.2.

After successful configuration, the PP3-FPGA was configured with the triplicated design. Each instance of the bitstream file used for configuration occupies approximately 5.5 MB. This facilitated storing three copies of the design at different locations in the 16 MB flash memory (S25LP128-JBLE [165, 166]). These results are summarized in Table 7.4.

Table 7.4: Multi-boot Configuration Results

Property	**Specifications**
Size of the Golden image	5.5 MB
Size of each Fallback image	5.5 MB
Number of copies on the flash memory	3 copies
Memory programming time via JTAG	6 mins

The stored copies in the flash are independent of each other and are queued in sequence to be loaded into the FPGA once a mBAR scenario is activated as discussed in Section 7.9.3.

Chapter 8

Test Results and Validation

The testing and validation phase of the MOPS-Hub includes strict assessments of both hardware and firmware components. This chapter aims to ensure the reliability, functionality, and performance of the MOPS-Hub crate across various scenarios.

8.1 The MOPS-Hub Readout Board

In order to make a comprehensive crate testing independent from the EMCI/EMP chain as discussed in Section 3.5 , a specialized test setup has been developed utilizing an FPGA-based interface known as **MOPS-Hub Readout board**.

Figure 8.1 illustrates the concept of the main test setup deployed to do full crate testing. Notably, the MOPS-Hub Readout board consolidates the entire functionality of the EMCI/ EMP chain into a single FPGA-based system[1]. This system serves as a data aggregator between MOPS-Hub and any local computer.

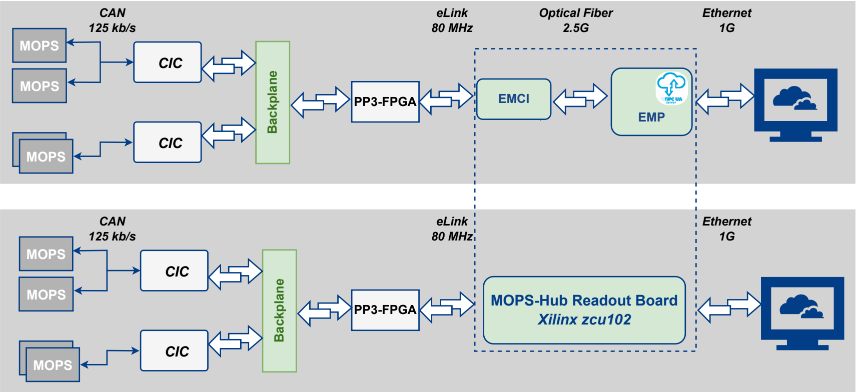


Figure 8.1: Block diagram of the main test setup used to test the MOPS-Hub crate(bottom) compared to the main chain structure in the final system (top).

In the downstream direction (to MOPS-Hub), the Readout board interfaces with MOPS-Hub through the differential eLinks signals. In the upstream direction (toward

[1] Zynq UltraScale+ MPSoC ZCU102 Evaluation Kit

the local computer), communication is established through a bidirectional Ethernet interface at 1 Gbit/s via a TCP/IP connection.

8.1.1 Firmware Architecture

The firmware for the MOPS-Hub Readout board is primarily written in Verilog. However, certain Intellectual Propertys (IPs) blocks instantiated within the design, such as IPBus, are implemented in VHDL.

Figure 8.2 depicts the block diagram of the firmware design for the MOPS-Hub Readout board. Each block is further elaborated upon in dedicated sections below.

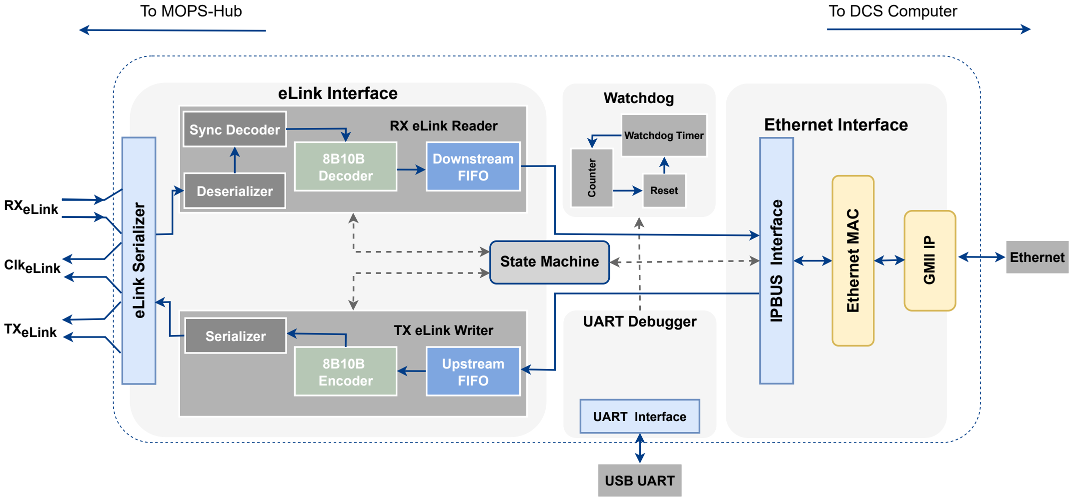


Figure 8.2: Firmware design of the MOPS-Hub Readout board. The bold blue lines represent data lines, while the light gray lines represent control signals from/to the top FSM.

The firmware consists of six main components:

1. **The TX Writer**: Contains upstream memory blocks and the 8B10B encoder module. The module structure is identical to the eLink Transmitter described in Section 7.5.1.

2. **The RX Reader**: Contains downstream memory blocks and the 8B10B decoder module. The module structure is identical to the eLink Receiver detailed in Section 7.5.2.

3. **eLink Serializer**: Provides the necessary building blocks for transmission and serialization over the eLinks. This component includes Output Double Data Rate (ODDRE1) and Input Double Data Rate (IDDRE1) primitives , both supported IPs by the manufacturer (Xilinx) [167].

4. **Watchdog**: Monitors the status of the top FSM after powering up. It generates a timeout signal if the FSM hangs for approximately 2 seconds due to

errors. The module structure is identical to the Watchdog module detailed in Section 7.7.

5. **Ethernet Core**: Translates Ethernet signals to the IP-based protocol (IPBus) interface and manages the data link layer (also referred to as the Media Access Control (MAC)) as well as the physical layer, which is implemented through the Gigabit Media-Independent Interface (GMII) interface.

6. **UART Interface**: Observes some internal registers or FSMs status either during operation. The module offers an independent communication protocol from the other interfaces for debugging.

8.1.2 Clock Distribution

The clock distribution within the readout firmware is depicted in Figure 8.3.

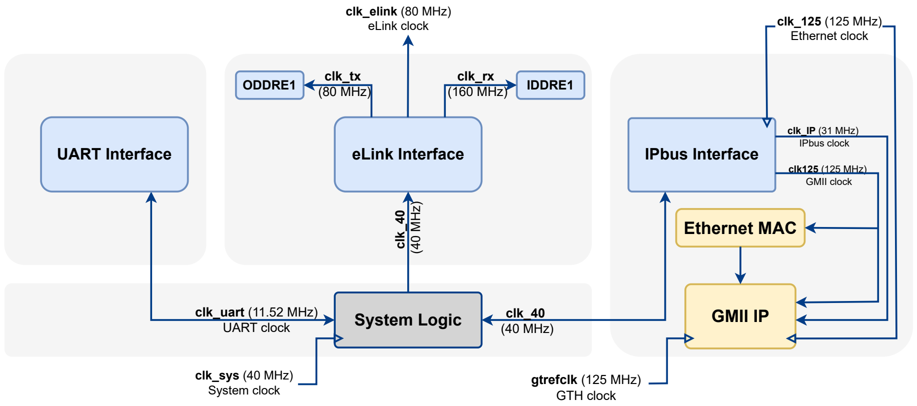


Figure 8.3: Overview of the four clock domains used in the MOPS-Hub readout firmware.

The readout firmware operates within four distinct clock domains:

1. **System clock (*clk_sys*)** (40 MHz): Used by the system logic, and all relevant components. Additionally, the clock is used to derive the UART clock (*clk_uart*) after rescaling.

2. **eLink Clock (*clk_elink*)** (80 MHz): Facilitates the eLink core and all eLink-relevant components. Additionally, the clock serves as the master eLink clock source for the PP3-FPGA logic.

3. **Ethernet Clock (*clk_125*)** (125 MHz): Generates the Ethernet clock necessary for communication between the FPGA (via IPBus) and the GMII layer.

4. **GTH Clock (*gtrefclk*)** (125 MHz): Utilized for the GMII transceiver in the Zynq device.

8.1.3 eLink Components

All components utilized for communication through the eLink are identical to those employed for MOPS-Hub, as extensively detailed in Section 7.5 except the eLink Serializer.

The eLink Serializer of the MOPS-Hub readout utilizes ODDRE1 and IDDRE1 primitives for data serialization and deserialization, respectively [167].

The Output Double Data Rate (ODDRE1) primitive functions as a parallel-to-serial converter, transforming the parallel data received from the **eLink Transmitter** into a serial stream for transmission over a differential wire pair. Meanwhile, the Input Double Data Rate (IDDRE1) primitive acts as a serial-to-parallel converter, generating a 2-bit-wide parallel word derived from the differential input pair of the eLink lines into the MOPS-Hub Readout board. More details about the module are available in the firmware specification document of the MOPS-Hub [168].

8.1.4 Ethernet MAC

The Ethernet Core provides the network interface required for Ethernet communication within the firmware. It integrates the Ethernet MAC layer, which leverage the Xilinx Tri-Mode Ethernet MAC as specified in the documentation [169]. This MAC layer organises communication between the IPBus interface and the Ethernet Physical Layer (PHY). The MAC layer employs the Serial Gigabit Media Independent Interface (SGMII) to establish connectivity with the Ethernet PHY, as outlined in [170]. Additionally, communication with the Programmable Logic (PL) side of the Gigabit Transceiver-High Speed (GTH) on the board is facilitated through the quad Small Form-factor Pluggable (SFP)+ interface.

8.1.5 IPBus Interface

The IPBus interface serves as the backbone for implementing address-based read and write interfaces for master and slave components. It relies on the IPBus controller, a core component provided by the IPBus community [171]. To facilitate master-slave communication, a set of interface types is defined to manage signal directions:

- *ipb_wbus*: signals from master to slave to request read and write operations.
- *ipb_rbus*: signals from slave to master to reply to read and write operations.

All the sub signals between master and slave are detailed in Section C.3.

Bus Protocol

As detailed in Section 7.5.5, the typical eLink frame structure within the PP3-FPGA firmware, as well as the MOPS-Hub readout firmware, carries 10 bytes of payload data. These information is buffered in the raeadout firmware through 96 bit registers, referred to as *data_rec_elink* and *data_tra_elink* for received and transmitted eLink data respectively as indicated in Table C.8.

The bus transaction within the *IPBus interface* is initiated by the bus-master using interrupt signals such as *start_write_elink* and *start_read_elink*, which manage data buffering between the IPBus data registers and the internal First In First Outs (FIFOs) within the Ethernet Interface.

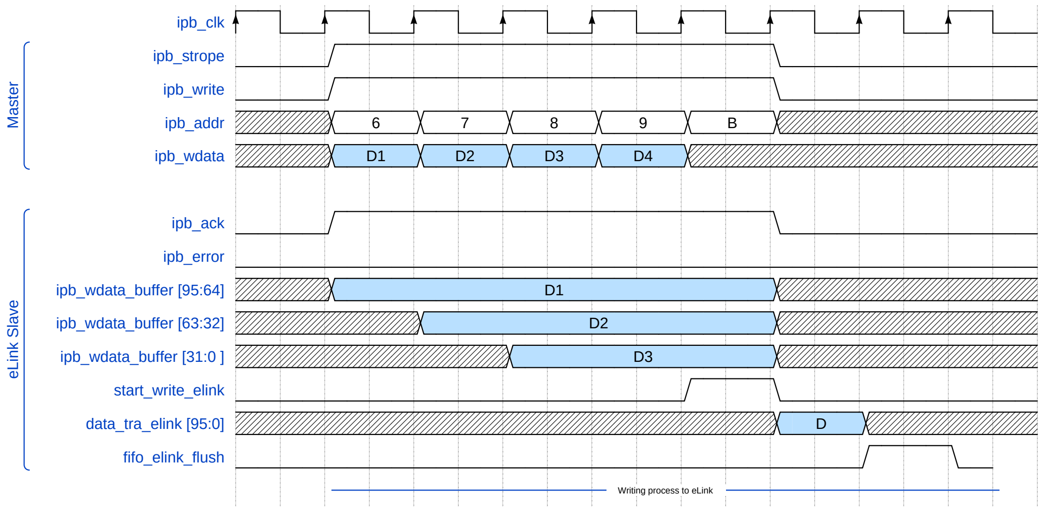


Figure 8.4: Wave form illustrating a write process to the eLink slave.

Figure 8.4 depicts a scenario involving a write process to the eLink slave, where the master asserts *ipb_wdata* along with the data to be written. Subsequently, the write operation is initiated by asserting *ipb_strope* and *ipb_write*. The data carried by *ipb_wdata* are collected based on the corresponding address *ipb_addr* in 4 clock cycles of the *ipb_clk*, followed by the assertion of *start_write_elink* to write data to the internal FIFOs.

Figure 8.5 depicts a scenario involving a read process from the eLink interface, where the master asserts *ipb_strobe* and *ipb_addr* signals. The slave then responds with either *ipb_ack* for valid data or *ipb_err* for errors. The data is then requested from the address-based registers on the *ipb_rdata* signal. Finally, the payload of the *data_rec_elink* register is transferred to the *ipb_rdata* in 32 bit packets in 3 clock cycles.

8.1.6 Software Implementation

The software controlling the MOPS-Hub Readout board operates on a Python-based framework running on a local computer. With a focus on modularity and rapid development, this framework is structured into distinct, self-contained classes, each assigned to specific functions. Figure 8.6 provides a comprehensive overview of the class structure and relationships within the MOPS-Hub software design.

The cornerstone of the software lies in the *UHALWrapper* class, which serves as a central component for managing various operations necessary for configuring and communicating with the hardware platform.

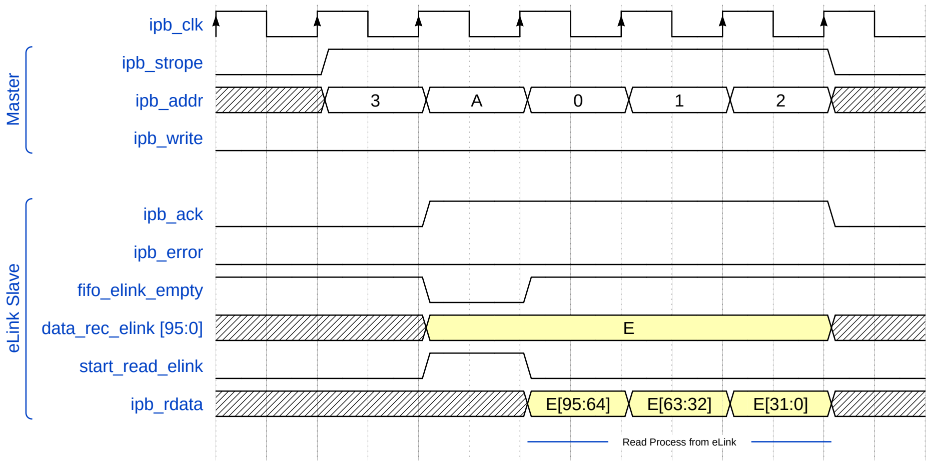


Figure 8.5: Wave form illustrating a read process from the eLink slave.

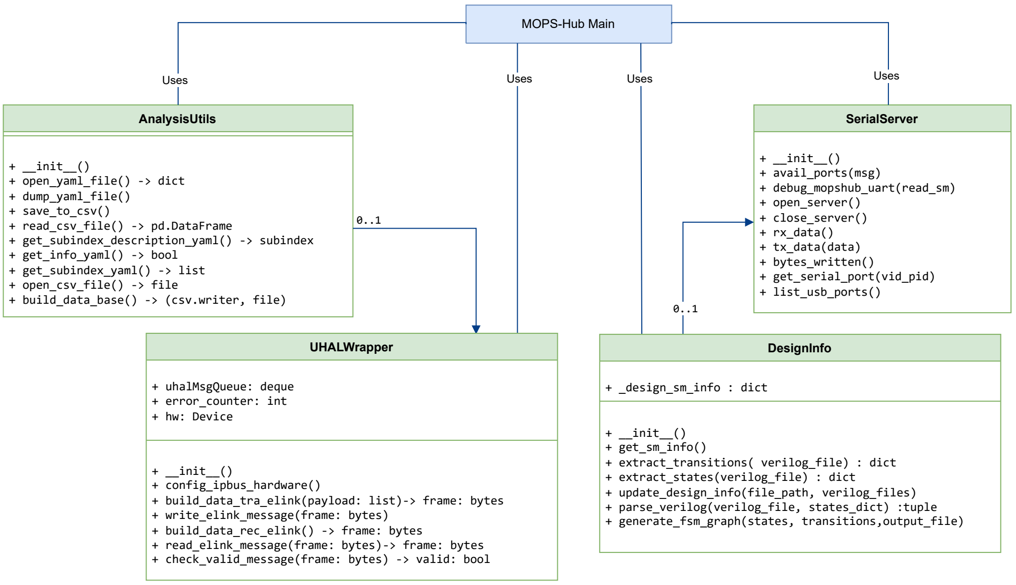

MOPS-Hub Main
Uses
Uses
Uses
Uses
AnalysisUtils
+ __init__()
+ open_yaml_file() -> dict
+ dump_yaml_file()
+ save_to_csv()
+ read_csv_file() -> pd.DataFrame
+ get_subindex_description_yaml() -> subindex
+ get_info_yaml() -> bool
+ get_subindex_yaml() -> list
+ open_csv_file() -> file
+ build_data_base() -> (csv.writer, file)
0..1
SerialServer
+ __init__()
+ avail_ports(msg)
+ debug_mopshub_uart(read_sm)
+ open_server()
+ close_server()
+ rx_data()
+ tx_data(data)
+ bytes_written()
+ get_serial_port(vid_pid)
+ list_usb_ports()
0..1
UHALWrapper
+ uhalMsgQueue: deque
+ error_counter: int
+ hw: Device
+ __init__()
+ config_ipbus_hardware()
+ build_data_tra_elink(payload: list)-> frame: bytes
+ write_elink_message(frame: bytes)
+ build_data_rec_elink() -> frame: bytes
+ read_elink_message(frame: bytes)-> frame: bytes
+ check_valid_message(frame: bytes) -> valid: bool
DesignInfo
+ _design_sm_info : dict
+ __init__()
+ get_sm_info()
+ extract_transitions(verilog_file) : dict
+ extract_states(verilog_file) : dict
+ update_design_info(file_path, verilog_files)
+ parse_verilog(verilog_file, states_dict) :tuple
+ generate_fsm_graph(states, transitions,output_file)

Figure 8.6: UML diagram depicting the class structure in the MOPS-Hub software design.

Leveraging the capabilities of the uHAL package, this class provides a Python API for accessing the IPBus registers embedded within the firmware [171]. Through this API, the software can efficiently read out the board via a TCP/IP connection. Furthermore, the software implementation encompasses additional classes dedicated to analysis and data visualization tasks.

Notably, the MOPS-Hub Readout board has demonstrated its reliability during MOPS-Hub crate testing and quality control. The modular architecture adopted in both the software and firmware levels enables the replacement or updating of individual components without disrupting the entire system, thereby increasing its functionality to evolving requirements over time.

Hardware Interface Configuration

The essential part of the software is the configuration function withing the *UHAL-Wrapper*, refer to as *config_ipbus_hardware*. This function is crafted to establish a connection to the hardware device using the designated IP address of the control IP-Bus master on the Readout board. Additionally, the function loads the address table of the design, which defines the address layout of the IPBus endpoints, providing a mapping of addresses to specific registers within the hardware. This address table is referenced in an XML file, the structure of which is depicted in Code 8.1. After initialization, the software is ready for data acquisition between the computer and the IPBus master, as elaborated in the subsequent sections.

Code 8.1: An example of the address table structure in the XML file. The registers for writing processes are indicated.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<node>
    </node>
    <node id="IPb_addr6"  address="0x12000006" description="R/W reg." fwinfo="endpoint;width=31"/>
    <node id="IPb_addr7"  address="0x12000007" description="R/W reg." fwinfo="endpoint;width=31"/>
    <node id="IPb_addr8"  address="0x12000008" description="R/W reg." fwinfo="endpoint;width=31"/>
    <node id="IPb_addr9"  address="0x12000009" description="R/W reg." fwinfo="endpoint;width=31"/>
    <node id="IPb_addr10" address="0x1200000A" description="R/W reg." fwinfo="endpoint;width=31"/>
    </node>
</node>
```

Data Transmission

Data transmission within the *UHALWrapper* class is managed using two methods referred to as *build_data_tra_elink* and *write_elink_message*. These methods facilitate the construction and transmission of data frames to the IPBus master.

As illustrated in Figure 8.4, writing data to the *data_tra_elink* register requires targeting the mapped address buses from the software level. This task is handled within the software using the function *build_data_tra_elink*, which constructs a frame containing the payload values for registers 6, 7, and 8.

Figure 8.7 illustrates the frame structure in Python to request a message to the MOPS-Hub with the help of the Readout board using the uHAL package.

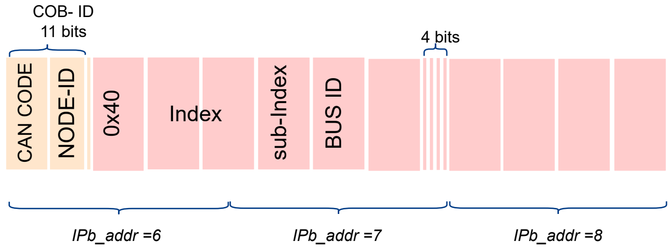


Figure 8.7: The frame structure in Python to transmit a message to the MOPS-Hub with the help of the MOPS-Hub Readout board.

As depicted, the parameters needed for a CANOpen SDO (i.e., COBE-ID, index, sub-index, and Bus-ID) are structured within the frame itself before being written to the IPBus master via the function *write_elink_message*.

Data Reception

Data Reception within *UHALWrapper* class is managed using two methods refer to as *build_data_rec_elink* and *read_elink_message*. These methods handle the reception and processing of data from the IPBus master. The status of the internal FIFO of the Readout board is triggered by reading back the data accompanied by "*ipb_addr* = 3", indicating the availability of data on the register *data_rec_elink*. Once data is available to be read, the reading process will be enabled. Subsequently, the function *build_data_rec_elink* starts building the received frame out of the registers 0, 1, and 2. The mapping of each byte in the frame depends on the message received from the eLink on the readout side, whether it is a CAN message or a message received from one of the SPI interfaces on the MOPS-Hub.

Figure 8.8 illustrates the frame structure of the data extracted from registers 0, 1, and 2 during data reception of a CAN message.

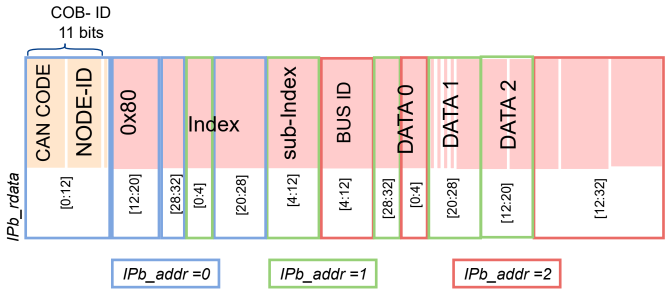


Figure 8.8: The frame structure in Python of a received CAN message from the MOPS-Hub with the help of the MOPS-Hub Readout board.

As depicted, the 96 bit frame received from the hardware system carries the payload received from the MOPS-Hub, including all the information regarding the MOPS chip as detailed in Section 7.3.4.

To verify the validity of the received messages during communication, another function called *check_ valid_ message* is dedicated to ensuring that the frames received from the hardware are correct and complete.

Serial interfaces

The *SerialServer* class is designed to facilitate serial communication for debugging and interaction with embedded systems. Additionally, it provides specialized debugging capabilities for the MOPS-Hub UART, allowing for the transmission of data to specified IDs and subsequent reading of responses.

Data Handling and Analysis

Once data reception is complete, raw frames received via TCP/IP from the readout are saved to disk in CSV format. This format enables the storage of metadata alongside the raw data, including information such as Bus ID, timestamps, and more, all within a single file.

Any interpretation of the data is conducted offline by an analysis class, referred to as *AnalysisUtils*, after the scan is finished (refer to Figure 8.6). This analysis can also be manually invoked at any time after the readout process, without requiring any hardware to be connected to the PC, as long as the raw data file is available.

The debugging information extracted from the MOPS-Hub received via UART is managed autonomously using the *DesignInfo* class (refer to Figure 8.6). This class integrates various functionalities to extract FSM information from the Verilog files and compiles it into a dictionary. This information is then stored for further comparison with data received via the UART interface interface through the *SerialServer* class.

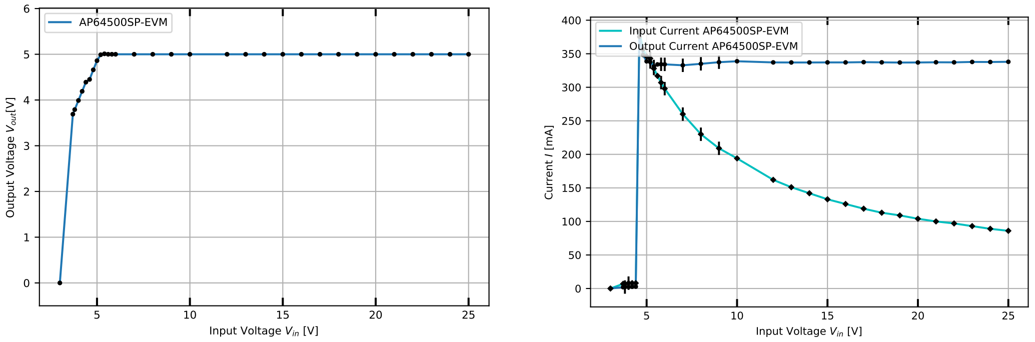
8.2 PP3-Power Module Testing

This Section outlines various steps involved in ensuring the reliability and compliance of the PP3-Power module detailed in Section 4.4.

8.2.1 DC/DC Measurements

The DC/DC converter testing involved capturing four critical parameters: input voltage (V_{in}), input current (I_{in}), output current (I_{out}), and output voltage (V_{out}) at different V_{in} points.

Figure 8.9(a) demonstrates the behavior of V_{out} in response to V_{in} varying levels. The DC/DC converter consistently maintains the required output voltage for the FPGA (5 V) when the input voltage exceeds 5 V, indicating stability in voltage regulation.



(a) Voltage behavior of the DC/DC converter. (b) Current handling by the DC/DC converter.

Figure 8.9: DC/DC measurements for the step down DC/DC converter on the PP3-Power module.

Figure 8.9(b) showcases the DC/DC converter's capability to manage different current demands as V_{in} varies. Several Modules have been tested to verify the adaptability of the PP3-Power module to changing power requirements.

The stability and efficiency of the DC/DC converter's can be inferred from these curves. Figure B.1 illustrates the DC/DC converter's performance in terms of efficiency across various input voltages, peaking at approximately 90% during nominal operation (with $V_{in} = 5\,\mathrm{V}$), aligning with manufacturer specifications [67].

8.2.2 Startup Behavior Study

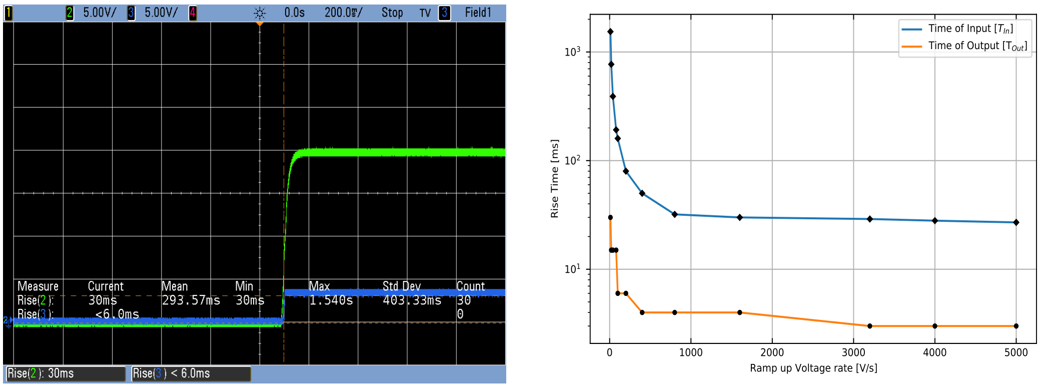
The specifications for MOPS-Hub power supply indicate the range of the ramp up rate in the range of 10 V/s to 40 kV/s [51]. To verify that the system stability under these specifications, the rise time of the input/output signal at various input voltages was measured (refer to Figure 8.10(a)).

The investigation into the startup behavior focuses on the rise time of input and output signals under various ramp-up rates. To assess compliance with the power specifications, the rise time was measured with a nominal load of 3.4 A, tracking the transition from 10% to 90% of the signal's amplitude for both V_{in} and V_{out}.

The findings, highlighted in Figure 8.10(b), indicate a consistent rise time for the output voltage at approximately 3 ms, regardless of variations in the input voltage or the ramp-up speed. This consistency ensures the module adherence to the specified ramp-up rates.

8.2.3 Magnetic Field Test

In this section, the testing of the PP3-Power module under varying magnetic field strengths is detailed, with the objective of verifying its functionality at the PP3 location.



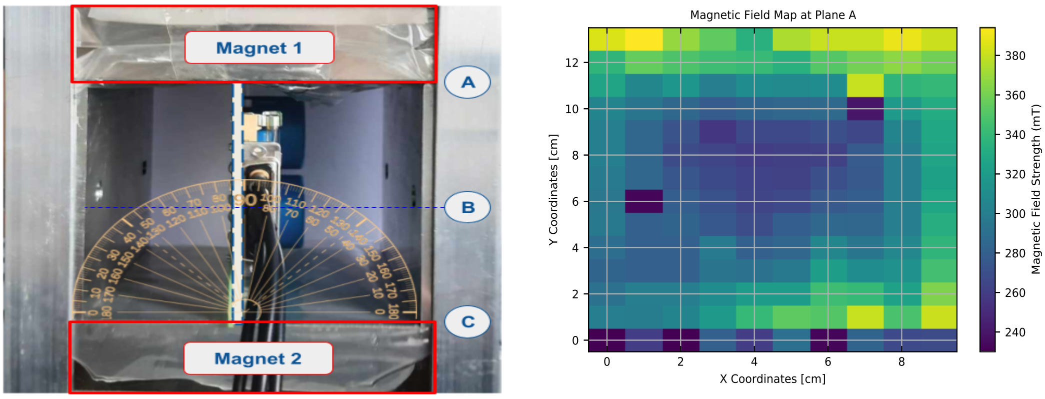
(a) Voltage signals for V_{in}= 20 V (Green) and V_{out}= 5 V (blue) at a ramp up speed 5 kV/s.

(b) Rise time analysis for different ramp-up speeds with V_{in} set at 20 V.

Figure 8.10: Analysis of the startup behavior of the DC/DC converter on the PP3-Power module.

Test Procedure: Setup and Description

As depicted in Figure 8.11(a), the test setup involves subjecting the DUT to a continuous magnetic field ranging from (220 to 380) mT. This magnetic field is generated using two magnets (Magnet 1 and Magnet 2), each with dimensions of 100 × 160 mm^2. The magnets are positioned opposite each other, creating a region in which the DUT is placed.



(a) Magnetic field test setup.

(b) Magnetic field strength at plan A.

Figure 8.11: The controlled test setup during the magnetic field evaluation.

Figure 8.11(b) depicts the measured magnetic field strength in the horizontal plane (A) between the two magnets. The magnetic field strength at plane (B) is approximately 50% less than that at plane (A).

Figure 8.12 depicts the schematics of the setup used to evaluate the PP3-Power module (ID = V2.4)[2] with an Arduino-based test under controlled conditions. This setup allowed the Arduino to read the parameters listed in Table 8.1.

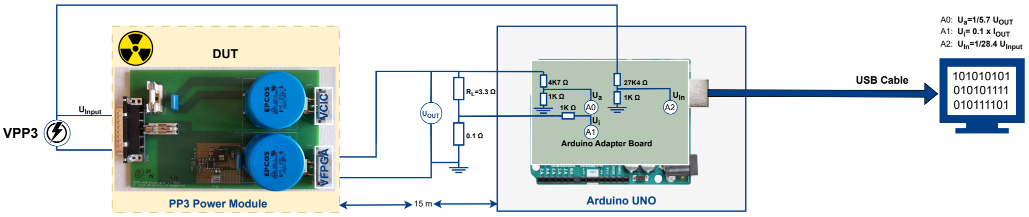


Figure 8.12: The test setup designed to evaluate the PP3-Power module under controlled conditions.

During testing, the input power source was connected to the module via cables. An electronic load of 3.3 Ω was connected to the module's output port to simulate the behavior of the power consumed by the PP3-FPGA.

Table 8.1: Tracked PP3-Power module parameters using the setup depicted in Figure 8.12.

Item	**Range**	**Definition**
$\mathbf{U_{Supply}}$	(0 to 25) V	Supply voltage (≈ 18 V)
$\mathbf{I_{Supply}}$	(0 to 2.5) A	Supply current (≈ 0.6 A)
VPP3	(3.8 to 40) V	Input voltage for the PP3-Power module (≈ 16 V)
$\mathbf{V_{FPGA}}$	(4.5 to 26.5) V[67]	Output voltage for the FPGA (≈ 5 V)
$\mathbf{I_{FPGA}}$	(0 to 5) A[67]	Output current consumption

To ensure comprehensive exposure to the magnetic field, the DUT was rotated at various angles.

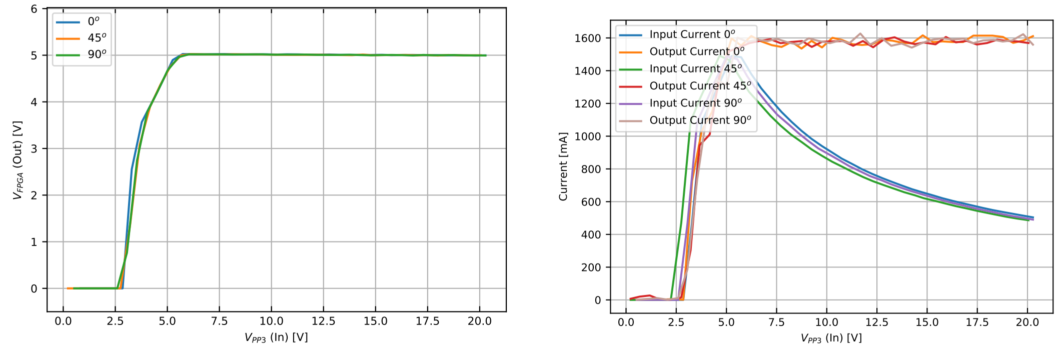
Summary of the Magnetic Field Test Results

Figure 8.13(a) demonstrates the behavior of V_{FPGA} in response to VPP3 varying levels. The DC/DC converter consistently maintains the required output voltage for the FPGA (5 V) when the input voltage exceeds 5 V, indicating stability in voltage regulation at various angles.

Figure 8.13(b) showcases the converter's capability to manage different current demands as VPP3 varies, illustrating its adaptability to changing power requirements at various angles at the magnetic field described in Figure 8.11.

The stability and efficiency of the module can be inferred from these curves. Figure B.2 illustrates the module's performance in terms of efficiency across various input

[2] Va.b refers to "a" as the version and "b" indicates the board ID.



(a) Voltage behavior of the PP3-Power module. (b) Current handling by the PP3-Power module.

Figure 8.13: Monitored parameters for the PP3-Power module (ID = V2.4) under several angles at the magnetic field described in Figure 8.11.

voltages, peaking at approximately 90% during nominal operation (with VPP3 = 6 V) irrespective of the magnetic field direction.

The noise levels on the V_{FPGA} of the PP3-Power module (ID = V2.4) were measured before and after exposure to the magnetic field, as detailed in Table 8.2. During exposure, the noise level increased significantly to approximately (360 to 380) $\mathrm{mV_{pp}}$ and did not return to its initial level. However, this change did not affect the powering of the regulator on the PP3-FPGA board.

Table 8.2: Noise level on the $V_{\mathrm{FPGA}}(out)$ of the PP3-Power module (ID = V2.4) before/during magnetic field testing.

Status	**Noise level** V_{FPGA}	**Comment**
Before magnetic field	≈(155 to 170) $\mathrm{mV_{pp}}$	Noise from the DC/DC converter
During magnetic field	≈(360 to 380) $\mathrm{mV_{pp}}$	A change of ≈180 $\mathrm{mV_{pp}}$

8.3 CAN Interface Card Testing

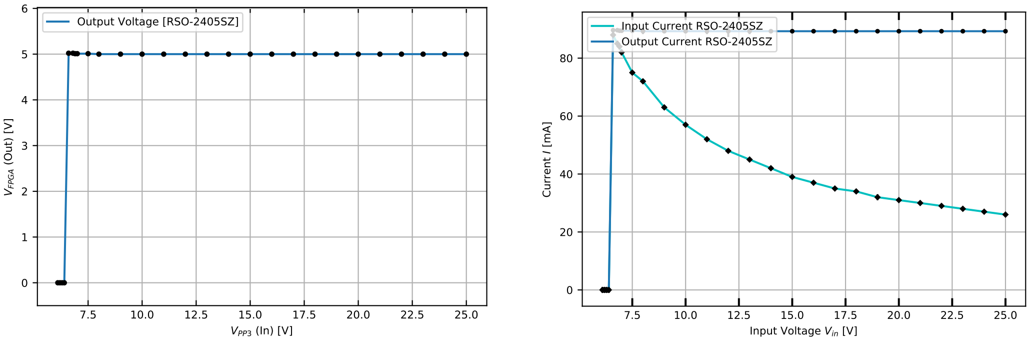
This Section outlines various steps involved in ensuring the reliability and compliance of the CIC detailed in Section 4.5.

8.3.1 DC/DC Measurements

The DC/DC measurements involved capturing four critical parameters: input voltage (V_{in}), input current (I_{in}), output current (I_{out}), and output voltage (V_{out}) at the point when V_{in} reached the circuit's operational point. In this test, V_{out} was connected to a nominal load, to mimic the card behaviour.

Figure 8.14(a) demonstrates the behavior of V_{out} in response to V_{in} varying levels. The DC/DC converter consistently maintains the required output voltage for the LDO (5 V) when the input voltage exceeds 6 V, indicating stability in voltage regulation.

Figure 8.14(b) showcases the DC/DC converter's capability to manage different current demands as V_{in} varies, illustrating its adaptability to changing power requirements. The stability and efficiency of the DC/DC converter can be inferred from these curves. Figure B.3 illustrates the DC/DC converter's performance in terms of efficiency across various input voltages, peaking at approximately 80% during nominal operation (with $V_{in} = 7.5\,\text{V}$), aligning with manufacturer specifications [172].



(a) Voltage behavior of the DC/DC converter.

(b) Current handling by the DC/DC converter.

Figure 8.14: DC/DC measurements for the isolating DC/DC converter on the CIC.

8.3.2 Voltages Level of the Communication Channels

In this test, the CIC channels are powered, with both channels A and B connected (CANH(A/B) and CANL(A/B)). Consequently, any CAN message transmitted from A is received at B, and vice versa. During each transmission, the communication signals CANH/L and VCANA/B signals are observed on the oscilloscope, as illustrated in Figure 8.15. The test shows no voltage drop or misbehavior during communication in any of the channels. Additionally, the voltage level of the CANH and CANL signals is adapted to the operation voltage of the MOPS (1.2 V) as detailed in Section 4.5.1.

8.3.3 Cross Talk Check

A cross-talk check is an essential test to ensure that as a signal traverses any of the traces, it does not induce capacitive or inductive coupling in adjacent traces. This section details the methodology employed to assess the potential for cross-talk between power and communication channels on the CIC.

The cross-talk susceptibility was evaluated under four distinct electrical conditions, designed to simulate various operational scenarios, depicted in Section B.2.4. In this

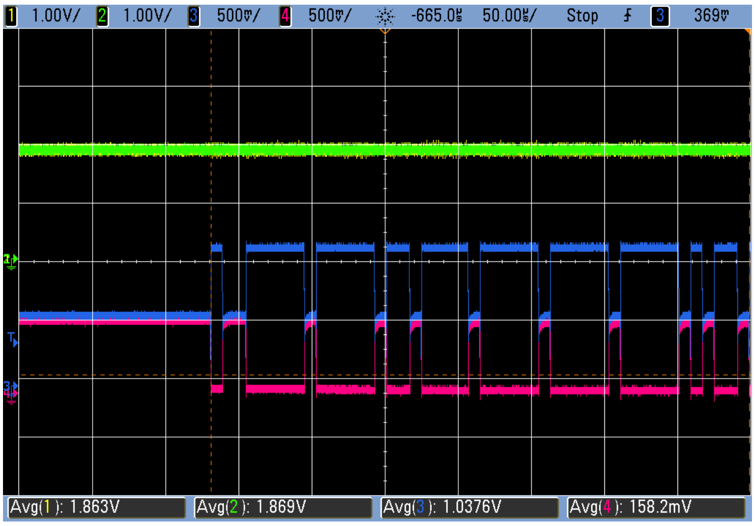


Figure 8.15: CANH/L (A/B) signals during communication. The channels are defined as follow, CH1: VCANA, CH2: VCANB, CH3: CANL(A/B) and CH4: CANH(A/B). The oscilloscope Settings:[(All The channels are DC coupled), (V/div: CH1/2=1 V/div, CH3/4=0.5 V/div), (Time/div=50 µs/div), (Bandwidth limit=20 MHz)].

test, no load was connected at the output pin of any of the channels, and the rise time [3] (or the discharge time) of the input/output signal at various input voltages was recorded.

The outcomes, tabulated in Table 8.3, indicate the absence of detrimental cross-talk between the evaluated channels (VCANA and VCANB). This confirms the efficiency of the PCB design in minimizing electromagnetic interference during operation.

Table 8.3: Cross talk check on the power channels (VCANA and VCANB). No load is connected. All the data are extracted from Figure B.5

Test Settings		**Time Results for VCANA**		**Conclusion**
VCANA	VCANB	Rise Time	Discharge Time	
ON	OFF	≈ 247 µs	-	No cross-talk detected
OFF	OFF	-	≈ 137 ms	No cross-talk detected
ON	ON	≈ 243 µs	-	No cross-talk detected
OFF	ON	-	≈ 139 ms	No cross-talk detected

8.3.4 Noise Measurements

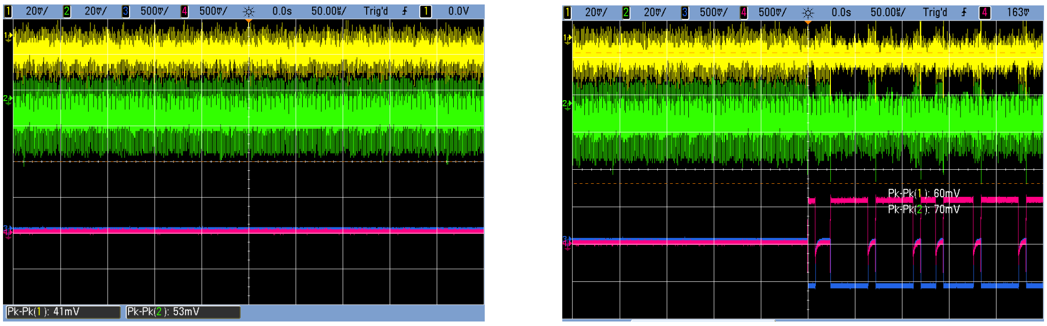
Maintaining high signal integrity against various forms of noise is critical for ensuring reliable data transmission during CAN communication. Random noise spikes can lead

[3] The rise time is calculated during transition from 10% to 90% of the output signal.

to bit errors where bits are incorrectly read as high (1) or low (0), potentially altering the data being transmitted. Additionally, excessive noise can lead to error frames in the bus or even complete signal loss in severe cases. Two primary sources of noise are considered. The first is noise due to communication channels. The second source is noise originating from the PP3-Power module.

Noise due to Communication

This test was conducted under two conditions: during communication and without communication, as depicted in Figure 8.16(b). The noise level in both channels extracted from the oscilloscope signal is illustrated in Table 8.4.



(a) **Without** CAN communication on the CAN side (Time/div=50 µs/div).

(b) **During** CAN communication on the CAN side (Time/div=50 µs/div).

Figure 8.16: Noise measurements for VCANA and VCANB, CANA and CANB. The channels are defined as follow, CH1: VCANA, CH2: VCANB, CH3: CANH(A/B) and CH4: CANL(A/B). Oscilloscope Settings:[(CANH/L(A/B) are DC coupled while VCAN+(A/B) are AC coupled), (Bandwidth limit=20 MHz), (V/div(CH1&CH2) =20 mV/div) and (V/-div(CH3&CH4) =500 mV/div)].

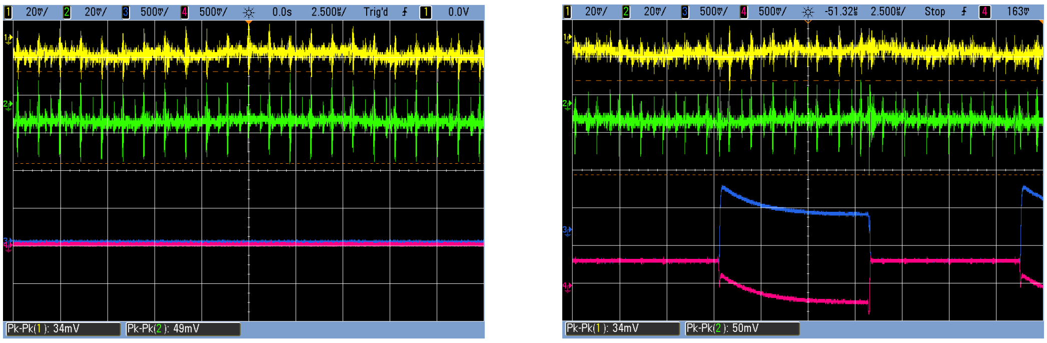
Table 8.4: Noise measurements on the CIC (V4.1) due to communication. All the data are extracted from Figure 8.16.

Test Conditions	**Noise level**	**Conclusion**
	$\text{VCAN}_{\text{Noise}}$	
Without CAN communication	$\approx$(40 to 55) mV_{pp}	Noise from the DC/DC converter
During CAN communication	$\approx$(60 to 70) mV_{pp}	No Effect on CAN communication

In these tests, a noticeable increase in noise levels on the VCAN channels during active CAN communication were observed. Despite this increase, results indicate that there is no adverse effect on the quality of CAN communication. Notably, the DC/DC converter is identified as the primary noise source, confirmed by specifications from the manufacturer's datasheet [172]. More measurements were also performed during VCAN are depicted in Section B.2.3.

Noise due to the PP3-Power Module

The influence of the PP3-Power module (VPP3) on the communication channels is investigated to determine how fluctuations in power supply and switching noise impact data integrity. For this test, the PP3-Power module was powered with VPP3 at 6 V (0.191 A).


1 20mV/ 2 20mV/ 3 500mV/ 4 500mV/ 0.0s 2.500µs/ Trig'd 1 0.0V
Pk-Pk(1): 34mV
Pk-Pk(2): 49mV
1 20mV/ 2 20mV/ 3 500mV/ 4 500mV/ -51.32µs 2.500µs/ Stop 4 163mV
Pk-Pk(1): 34mV
Pk-Pk(2): 50mV

(a) **Without** CAN communication on the CAN side (Time/div=50 µs/div).

(b) **During** CAN communication on the CAN side (Time/div=2.5 µs/div).

Figure 8.17: Noise measurements for VCANA and VCANB, CANA and CANB. The channels are defined as follow, CH1: VCANA, CH2: VCANB, CH3: CANH(A/B) and CH4: CANL(A/B). Oscilloscope Settings:[(CANH/L(A/B) are DC coupled while VCAN+(A/B) are AC coupled), (Bandwidth limit=20 MHz), (V/div(CH1&CH2) =20 mV/div) and (V/-div(CH3&CH4) =500 mV/div)].

The noise level due to VPP3 powering as extracted from the oscilloscope signal is illustrated in Table 8.5.

Table 8.5: Noise measurements on the CIC (V4.1) due to VPP3 powering. All the data are extracted from Figure 8.17.

Test Conditions	**Noise level** $\mathrm{VCAN}_{\mathrm{Noise}}$	**Conclusion**
Without CAN communication	≈(30 to 50) $\mathrm{mV}_{\mathrm{pp}}$	No Effect on CANH/L.
During CAN communication	≈(30 to 50) $\mathrm{mV}_{\mathrm{pp}}$	No Effect on CANH/L.

Notably, the noise levels in both tested scenarios (without and during CAN communication) remained consistent, ranging ≈(30 to 50) $\mathrm{mV}_{\mathrm{pp}}$. These findings suggest that the PP3-Power module's noise output does not adversely affect the communication integrity of the CAN system.

8.3.5 Magnetic Field Test

In this section, the testing of the CIC under varying magnetic field strengths is detailed. The CIC was exposed to a continuous magnetic field ranging from (220 to 380) mT, as depicted in Figure 8.11.

Test procedure: Setup and Description

Figure 8.18 depicts the setup used to evaluate the CIC module for the magnetic field testing. This setup allowed the Arduino to read the parameters listed in Table 8.6. A Python script running on a PC directed the Arduino to transmit the recently acquired parameters.

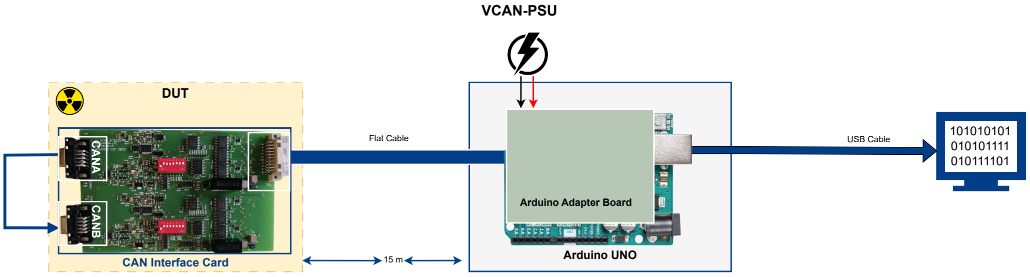


Figure 8.18: The test setup designed to evaluate the CIC under controlled conditions.

Table 8.6: Tracked CIC parameters using the setup depicted in Figure 8.18

Item	Details
U_CIC_VCAN (A/B)	Voltage readings of CAN buses using the onboard ADC.
I_CIC_VCAN (A/B)	Current readings of CAN buses using the onboard ADC.
U_DCoupler	Input voltage of the digital isolator.[4]
I_DCoupler	Supply current of the digital isolator.
V_CAN (A/B)	Voltage readings for CAN buses using the Arduino ADC.
U_CIC_PSU	Voltage supplied by the Power Supply Unit (PSU).
I_CIC_PSU	Supply current from the PSU.

To ensure comprehensive exposure to the magnetic field, the DUT was rotated at various angles.

Summary of the Magnetic Field Test Results

Table 8.7 presents the parameters of the CIC (ID = V4.2) during the magnetic field test at different orientations. Voltage readings for VCAN channels (*V_CANA* and *V_CANB*) as well as other electrical parameters, have only minor fluctuations.

Table 8.7: Average values of the CIC (ID = V4.2) parameters during Magnetic Field Test at different Orientations. Based on the data depicted in Section B.2.5.

Parameter	0°	45°	90°
U-CIC_PSU (V)	9.34±0.02	9.34±0.02	9.34±0.02
I-CIC_PSU (mA)	91.81±11.11	92.44±11.91	104.85±12.22
U_DCoupler(V)	5.17±0.01	5.16±0.01	5.17±0.012
I_DCoupler (mA)	11.91±3.46	11.91±6.26	11.86±6.45
V_CANA(V)	1.91±0.001	1.91±0.03	1.91±0.004
V_CANB(V)	1.90±0.001	1.90±0.03	1.91±0.004
U_CIC_VCANA(V)	0.96±0.01	0.95±0.001	0.96±0.01
U_CIC_VCANB(V)	0.96±0.001	0.95±0.001	0.96±0.01

The noise levels on the VCAN channel of the CIC (ID = V4.2) were measured before and after exposure to a magnetic field, as detailed in Table 8.8. After exposure, the noise level increased significantly to approximately (280 to 300) $\mathrm{mV_{pp}}$. Despite this substantial increase in noise, communication through the VCAN channel remained unaffected, indicating that the system's communication integrity was maintained even under higher noise conditions.

Table 8.8: Noise level on the VCAN channel of the CIC (ID = V4.2) before/during Magnetic Field.

Status	**Noise level** $\mathrm{VCAN_{Noise}}$	**Comment**
Before magnetic field	≈(40 to 60) $\mathrm{mV_{pp}}$	Noise from the DC/DC converter
During magnetic field	≈(280 to 300) $\mathrm{mV_{pp}}$	High noise level observed

8.4 Full System Testing

8.4.1 Test Setup

The test setup utilized for comprehensive testing of the full MOPS-Hub crate is detailed in Figure 8.19. The power specifications for the test setup include a stabilized voltage of 16 V supplied to VCAN-PSU and a voltage of 8 V for VPP3.

The setup detailed in Figure 8.19 integrates several key components, each serving a specific function within the testing framework:

- **MOPS-Hub Readout Board**: Facilitates data aggregation to an external PC via a Python script running on a local computer, as discussed in Section 8.1.
- **PP3-FPGA Board**: Further described in Section 4.6.

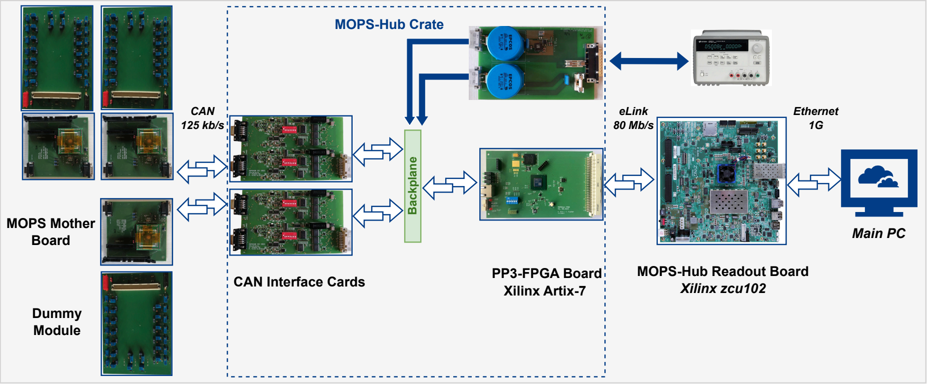


Figure 8.19: Block diagram of the main test boards used in the MOPS-Hub crate testing, complementing the setup shown in Figure 8.1.

- **PP3-Power Module**: Further described in Section 4.4.
- **CAN Interface Card (CIC)**: Further described in Section 4.5.
- **MOPS Motherboard**: Provides essential interface connections for the MOPS Carrier board, ensuring proper communication and integration of the MOPS chip into the system. Refer to Figure B.13(a) for more details.
- **Dummy Chain Board**: Simulates the actual serial power chain construction using variable resistors to represent detector modules and NTC sensors for temperature measurements, mimicking real module detector conditions, Refer to Figure B.13(b) for more details.

8.4.2 Test Procedure and Results

The performance of the MOPS-Hub interface was evaluated utilizing the detailed test system depicted in Figure 8.1. Each channel of the MOPS-ADC received a designated voltage from the **dummy chain module** to simulate operational conditions. The ADC readings were systematically initiated through commands from an external computer via the uHAL software package (detailed in Section 8.1.6). This procedure was repeated across all available buses to ensure comprehensive coverage.

During the testing period, real-time data extraction was performed on the local computer, capturing information such as the *Node ID* and the *CAN bus ID* continuously over a 72 h span. The collected data were subsequently stored for offline analysis, as summarized in Table 8.9. Remarkably, the dataset was found to be free of any errors, indicating a high level of reliability in the testing setup and execution.

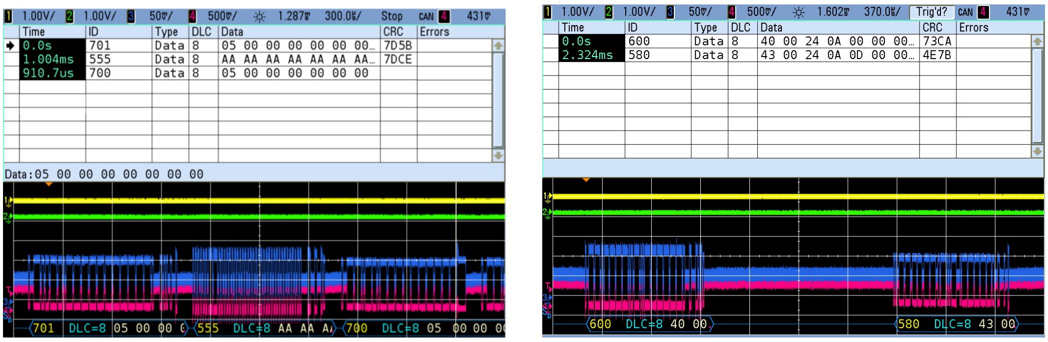
CAN Communication Results

Figure 8.20(a) depicts the behaviour of the MOPS-Hub upon powering up. once the system enters the **Initialization Phase**, a power enable signal is sent to the CIC,

Table 8.9: Summary of data aggregation performance between MOPS-Hub and MOPS-Hub Readout board

Parameter	Results
Requested CAN Messages	550,407 messages
Responded CAN Messages	550,407 messages
Failed Messages	0 messages
Test Duration	72 hours
Number of CICs	4
Activated Buses	3
Connected MOPS per Bus	2

causing the VCAN+/- to go high. This triggers the connected MOPS chips on the bus to enter configuration mode. Within approximately 5 seconds, the MOPS-Hub transmits a trimming pattern to adjust the oscillator frequency of the MOPS to the desired 10 MHz, as outlined in the MOPS specifications [46]. The two connected MOPS chips respond with a sign-in message acknowledging the receipt of the trimming pattern.



Time	ID	Type	DLC	Data	CRC	Errors
0.0s	701	Data	8	05 00 00 00 00 00 00…	7D5B	
1.004ms	555	Data	8	AA AA AA AA AA AA AA…	7DCE	
910.7us	700	Data	8	05 00 00 00 00 00 00		

Data:05 00 00 00 00 00 00 00

(a) Automated trimming after powering up.

Time	ID	Type	DLC	Data	CRC	Errors
0.0s	600	Data	8	40 00 24 0A 00 00 00…	73CA	
2.324ms	580	Data	8	43 00 24 0A 0D 00 00…	4E7B	

(b) ADC measurements on the MOPS chip

Figure 8.20: Overview of the CAN testing processes during MOPS-Hub testing. Channel configuration is as follows: CH1 is VCAN-, CH2 is VCAN+, CH3 is CANH, and CH4 is CANL.

Figure 8.20(b) depicts the ADC measurements from a MOPS chip (Node-ID = 0). The illustrated CAN message details the reading of Channel 10 (index$_h$ = 0x2400 and subindex$_h$ = 0xA). The transmitted message has a *COB-ID* of 600 (600+Node-ID), and the chip's response message with the target ADC info, with a *COB-ID* of 580 (580+Node-ID), includes the data in the last four bytes.

SPI Communication Testing

For SPI testing, several messages were sent from the PC to control the power enabling signals and read monitoring data from the CIC via SPI protocol at a frequency of 100 kHz. Several signals were monitored using the oscilloscope as illustrated in Figure 8.21.

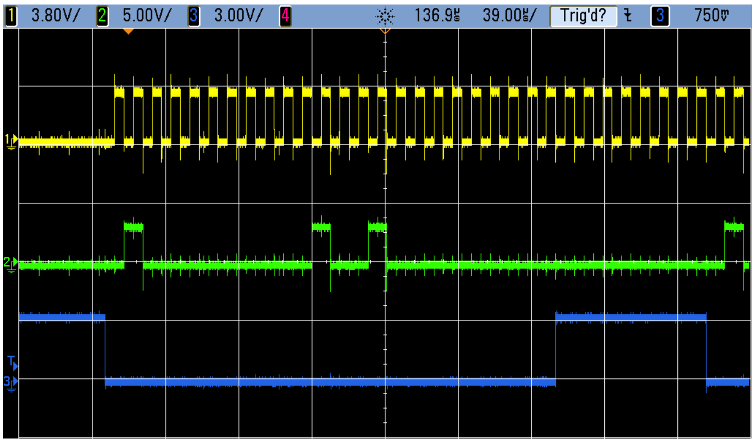


Figure 8.21: An example of the SPI wave form during communication on the CIC. The channels are defined as follows: CH1:**M-clk**, CH2:**M-SDO** and CH3:**M-CS**.

Monitored data from the CIC were saved on the **Host Computer** for later analysis.

VCAN voltages under different Configurations

As detailed in Section 4.5.2, the voltage level of the individual VCAN buses on the CIC can be adjusted via the 8 bit register chip.

Table B.1 in Section B.2.2 shows the truth table of these different configuration and the corresponding measured output VCAN according to equation 4.2.

Figure 8.22 illustrates the relation between each bit configuration and the output voltage $\mathrm{VCAN}_{\mathrm{Exp}}$. As depicted, the VCAN never exceeds 1.2 V as long as the connected resistor R_{set} is not enabled through the 8 bit DIP switch (DIP switch OFF). Once the corresponding bit is enabled the delivered VCAN will reach the target value within the accepted error margin [± 50 mV].

UART Communication Testing

The communication via UART is done with the help of an external PmodUSBUART circuitry which provides a USB to UART interface[173].

The write/read data process to the interface was done using the software package detailed in Section 8.1.6, with the help of the *SerialServer* class for serial communication. The baud rate is set to 115 200 bit s^{-1} to match the baud rate set on PP3-FPGA side.

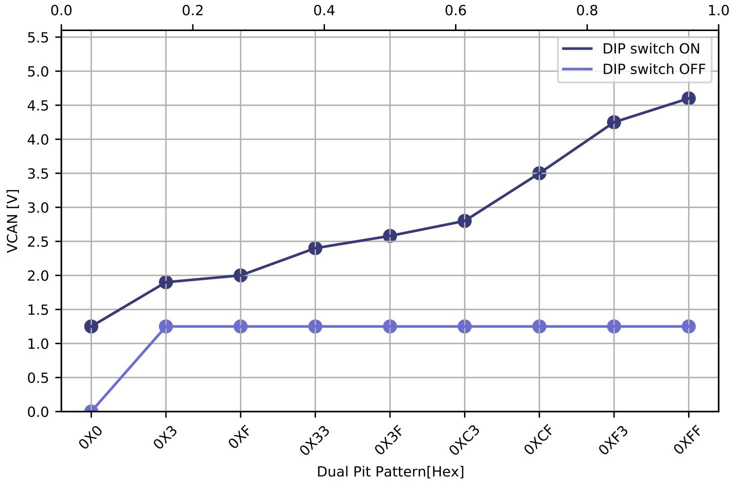


Figure 8.22: The output voltage level VCAN_{Exp} for channel A of the CIC (V4.1) under different configurations as illustrated in Table B.1.

Figure 8.23 shows an example of the UART wave form during debugging. The PC constructs serial communication requesting debug information from in the PP3-FPGA design through **in_rx_serial_0** (Ch2) and the PP3-FPGA replies back with its data **out_tx_serial_0**(Ch1) as a response with the same baud rate (115 200 bit s^{-1}).

The response data on the UART interface is only for debugging purposes and gives information about the status of different critical signals (e.g. FSMs) within the design.

8.4.3 Power Consumption of the MOPS-Hub Crate

Understanding the power consumption of the MOPS-Hub crate is essential for ensuring that the system operates within the electrical parameters defined at PP3 location. This section details the power requirements calculated for each component within the crate under maximum operational load conditions. This includes 2 PP3- FPGAs, 16 CICs, and 4 MOPS chips connected per CAN bus, all configured to draw maximum power simultaneously.

The power consumption for one CIC, without any external MOPS chips connected, is approximately 0.5 W. Given that each MOPS-Hub crate can contain up to 16 CICs, the total power consumption for all CICs per crate (P_{CIC}) is calculated as follows:

$$P_{\text{CIC}} = 16 \times 0.5\,\text{W} = 8\,\text{W} \tag{8.1}$$

Additionally, considering maximum utilization, the power consumption for all MOPS chips per crate (P_{MOPS}) is estimated using the equation:

$$P_{\text{MOPS}} = 16 \times 2 \times 4 \times (35\,\text{mA} \times 4\,\text{V}) = 17.92\,\text{W} \tag{8.2}$$

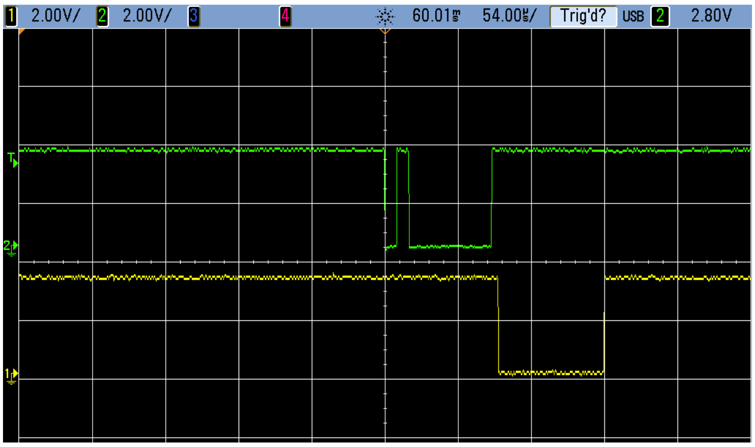


Figure 8.23: An example of the UART wave form during debugging. The PC constructs serial communication requesting debug information from in the PP3-FPGA design through **in_rx_serial_0** (Ch2) and the PP3-FPGA sends its data **out_tx_serial_0** (Ch1) as a response with the same baud rate ($115\,200\,\mathrm{bit\,s^{-1}}$).

This assumes that each CAN bus powers four MOPS chips at 4 V over approximately 70 meters, with each chip consuming a maximum of 35 mA during active communication, as specified in the MOPS manual [46].

The FPGA module, configured with triplicated firmware, exhibits an estimated power consumption of 3.2 W per module, with a conservative estimate including additional components on the PP3-FPGA board rounding up to 5 W per module. With two such modules per crate, the total power needed for the FPGA components ($P_{\mathrm{PP3-FPGA}}$) is:

$$P_{\mathrm{PP3-FPGA}} = 2 \times 5\,\mathrm{W} = 10\,\mathrm{W} \tag{8.3}$$

Summing up these components, the overall power consumption for each MOPS-Hub crate is calculated as:

$$P_{\mathrm{Crate}} = P_{\mathrm{PP3-FPGA}} + P_{\mathrm{CIC}} + P_{\mathrm{MOPS}} = 35.92\,\mathrm{W} \tag{8.4}$$

Chapter 9

Irradiation Tests of the MOPS-Hub

This chapter provides a comprehensive overview of the irradiation tests conducted on the MOPS-Hub, including TID, proton irradiation, and neutron irradiation campaigns. The objective of each test was to evaluate the performance of the MOPS-Hub without significant performance degradation.

9.1 TID Tests

The TID irradiation aimed to verify that MOPS-Hub hardware components can tolerate a steady-state TID exceeding 90 Gy. The TID test was conducted at the Gamma Irradiation Facility (GIF++) at CERN [174]. The facility is equipped with a chamber containing a ^{137}Cs gamma source with an activity of 14 TBq, delivering a dose rate of approximately 2 Gy/h. Details about the irradiation campaigns can be found in Table 9.1.

Table 9.1: Detailed information about the TID campaigns for the MOPS-Hub hardware components at GIF++.

Item	PP3-Power module	CAN Interface Card (CIC)	PP3-FPGA module
Test Date	May 2024	March 2023	May 2024
Focus of Interest	Electrical parameters	Physical layer Electrical parameters	Electrical parameters Temperature
Total Exposure Time	84 hours	72 hours	84 hours
Radiation Level	168 Gy	91 Gy	168 Gy

9.1.1 PP3-Power Module

The setup depicted in Figure 8.12 was used to evaluate the PP3-Power module (ID = V2.1) at GIF++. This setup allowed the Arduino to read the parameters listed in Table 8.1 via the Arduino's ADC ports, with real-time logging managed by a Python script on a PC. An electronic load of 3.3 Ω was connected to the module's output port to simulate the behavior of the power consumed by the PP3-FPGA.

Summary of the TID Test Results

During the TID irradiation, all parameters of the PP3-Power module were monitored through the Arduino's ADC ports, with real-time logging facilitated by a Python script on a PC. The PP3-Power module operated under a constant voltage of VPP3 = 16 V.

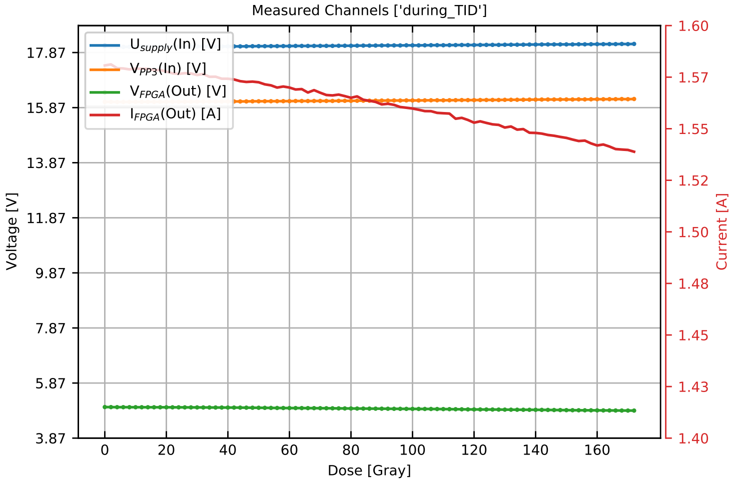


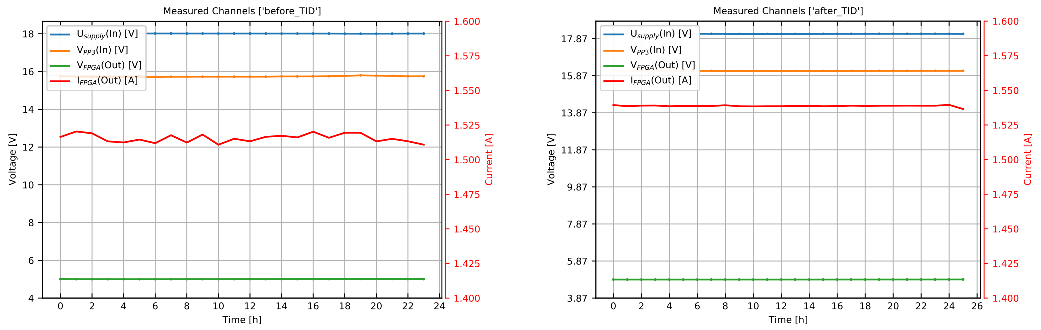
Figure 9.1: Monitored parameters for the PP3-Power module (ID = V2.1) during TID irradiation.

As depicted in Figure 9.1, the PP3-Power module's output voltage (V_{FPGA}) exhibited a slight downward trend from its nominal value of 5 V. Similarly, the output current (I_{FPGA}) showed a decrease of 49 mA after reaching a TID of 166 Gy. This reduction, although observable, remained within the acceptable range specified by the electrical characteristics required for the DC/DC module used in the PP3-FPGA, as specified in the manufacturer's datasheet [175]. Additionally, a precise voltage regulation on the PP3-Power module can be maintained under varying conditions using an analog potentiometer that allows adjustment of the output voltage (V_{FPGA}).

Post Irradiation Analysis

To evaluate the module's performance following irradiation, a long-term test was conducted using the same setup as detailed in Figure 8.12. The results, illustrated in Figure 9.2, indicate stable operation with no observed misbehavior from the module post-exposure.

Figure 9.3(a) depicts that the output voltage (V_{FPGA}) is stabilized around 4.8 V after the irradiation campaign, with a slight decrease from its nominal 5 V. This stabilization suggests that the module maintained adequate voltage regulation despite the radiation exposure (refer to Figure 9.2). Conversely, the output current consumption

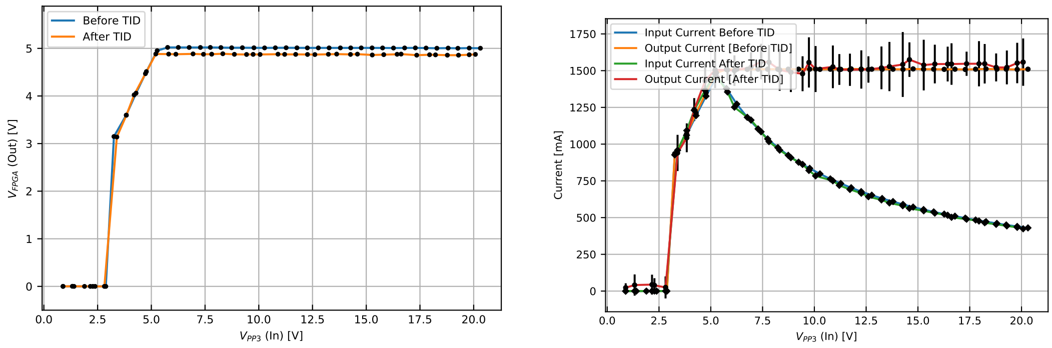


(a) Measured channels before TID.

(b) Measured channels after TID.

Figure 9.2: Long term test for the PP3-Power module (ID =V2.1) before and after TID measurements.

(I_{FPGA}), defined as the total current absorbed by the 3.3 Ω resistance load, exhibited higher fluctuations post-irradiation, indicating reduced stability.



(a) $\mathrm{V_{FPGA}}$ before and after irradiation.

(b) $\mathrm{I_{FPGA}}$ before and after irradiation.

Figure 9.3: Data results for the PP3-Power module (ID = V2.1) before and after TID irradiation.

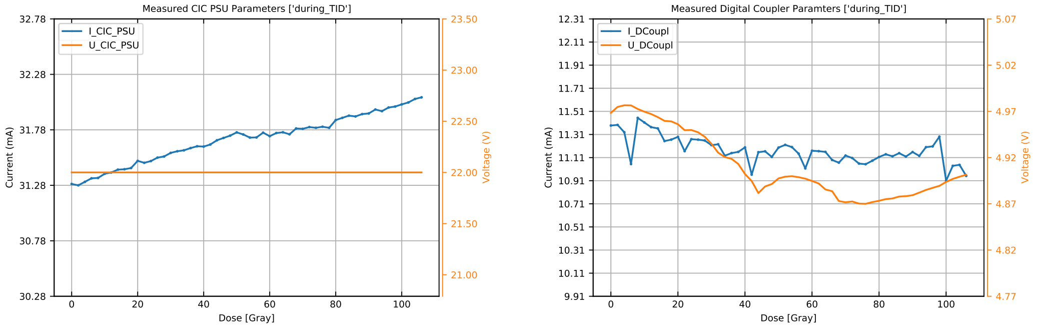
The noise levels on the V_{FPGA} of the PP3-Power module (ID = V2.1) were measured before and after the TID test. The noise level increased by approximately 50 $\mathrm{mV_{pp}}$, which did not significantly affect the communication with the PP3-FPGA.

9.1.2 CAN Interface Card

The TID test for the CIC has been implemented by Felix Nitz at TH Köln–University of Applied Sciences using an older prototype of the CIC (ID = V2.1). The setup depicted in Figure 8.18 was used to evaluate the CIC module at GIF++. This setup allowed the Arduino to read the parameters listed in Table 8.6. The CIC operated under a constant voltage (U_CIC_PSU = 22 V).

Summary of the TID Test Results

The TID irradiation campaign conducted on the CIC reached a total exposure of approximately 90 Gy without interruption to the power supply. The TID campaign resulted in a 2.5% increase in the PSU current by the end of the campaign, as shown in Figure 9.4(a).



(a) PSU parameters during irradiation.

(b) The digital coupler parameters during irradiation.

Figure 9.4: Monitored parameters on the CIC (ID =V2.1) during the TID irradiation.

Post Irradiation Analysis

A long-term test was conducted before and after the TID irradiation to assess the impact of accumulated radiation exposure on the device's performance over time (refer to Table 9.2).

Table 9.2: Average values of the CIC (ID =V2.1) parameters before and after the TID irradiation. Based on the data illustrated in Figure B.9.

Parameter	**Before Irradiation**	**After Irradiation**
U-CIC_PSU (V)	22.0±0.01	22.0±0.01
I-CIC_PSU (mA)	30.1±4.13	31.0±1.19
U_DCoupler(V)	4.91±0.03	5.04±0.07
I_DCoupler (mA)	11.2±1.03	11.4±0.92
V_CANA(V)	2.12±0.04	2.14±0.21
V_CANB(V)	2.32±0.06	2.23±0.25
U_CIC_VCANA(V)	2.11±0.01	2.12±0.01
U_CIC_VCANB(V)	2.30±0.01	2.30±0.01

The monitored parameters exhibited no abnormal behavior. Communication with the physical layer was smooth, and power readings were successfully obtained.

9.1.3 PP3-FPGA Module

The TID test for the PP3-FPGA has been implemented by Mark Leyer at TH Köln–University of Applied Sciences using the Tester-based setup illustrated in Figure 9.5 [176].

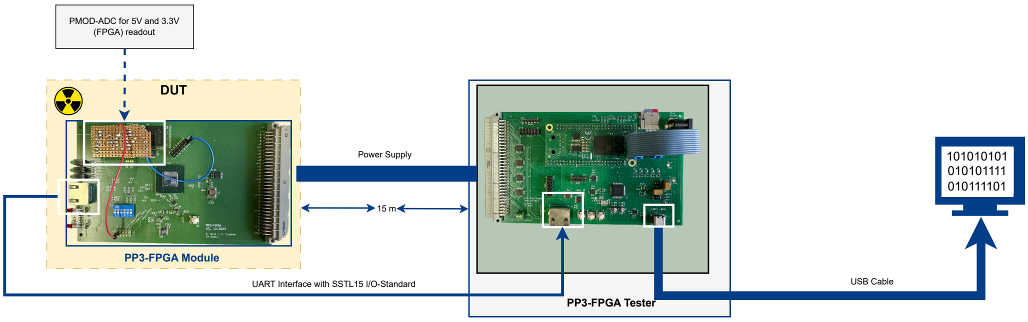


Figure 9.5: Test setup for the PP3-FPGA module during TID irradiation.

This setup allowed the tester to power-cycle the PP3-FPGA and monitor the internal FPGA parameters listed in Table 9.3. Input power has been connected to the module by cable and placed with sufficient distance from the radiation. No heating or cooling was provided.

Table 9.3: Monitored PP3-FPGA module parameters during TID campaign.

Item	**Range**	**Definition**
$\mathbf{V_{Supply}}$	(4.7 to 16) V	Supply Voltage from the PSU (≈ 5 V)
$\mathbf{V_{+3V3\_FPGA}}$	(1.14 to 3.465) V[177]	Supply voltage for the PP3-FPGA (≈ 3.3 V)
Temperature	(0 to 100) °C[177]	Temperature of the PP3-FPGA
VCCINT	(0.95 to 1.05) V[177]	PP3-FPGA V_{ccint} (≈ 1.00 V)
VCCAUX	(1.71 to 1.89) V[177]	PP3-FPGA V_{ccaux} (≈ 1.80 V)
VCCBRAM	(0.95 to 1.05) V[177]	PP3-FPGA V_{ccbram} (≈ 1.00 V)

Summary of the TID Test Results

The PP3-FPGA module demonstrated strong performance during the TID irradiation campaign. As detailed in Table 9.4, most monitored voltages showed no significant changes compared to pre-irradiation measurements. A slight decline in the 1 V supply voltages (**VCCBRAM**/**VCCINT**) was detected, dropping below 1 V after a dose of 130 Gy.

Post Irradiation Analysis

Post-campaign testing for the PP3-FPGA confirmed that the voltage did not recover and remained below 1 V. This decrease, starting only after 130 Gy, significantly

Table 9.4: Average values of the monitored supply voltages of the PP3-FPGA module before and during TID irradiation.

Parameter	Before Irradiation	During Irradiation
$V_{\textbf{Supply}}$ (V)	5.054±0.02	5.048±0.02
$V_{\textbf{+3V3\_FPGA}}$ (V)	3.38±0.005	3.37±0.003
VCCINT (V)	1.00±0.001	1.00±0.001
VCCAUX (V)	1.80±0.001	1.80±0.001
VCCBRAM (V)	1.00±0.001	1.00±0.001

exceeds the expected dose for the PP3 location—even with a safety factor of 3 applied—thus it can be considered non-critical. Additionally, the final supply voltage of 0.99 V remains within the operational range of the PP3-FPGA module's FPGA [177]. More information regarding the test is described in [176].

9.2 Neutron Campaign

The primary objective of the neutron irradiation campaign was to evaluate the NIEL effects on electronic components. The campaign aimed to achieve a specific neutron fluence of 10^{12} n/cm^2 to replicate conditions that electronics might encounter in the PP3 location as listed in Table 4.3. Details about the irradiation campaign can be found in Table 9.5.

Table 9.5: Detailed information about the neutron irradiation campaign for the MOPS-Hub hardware components.

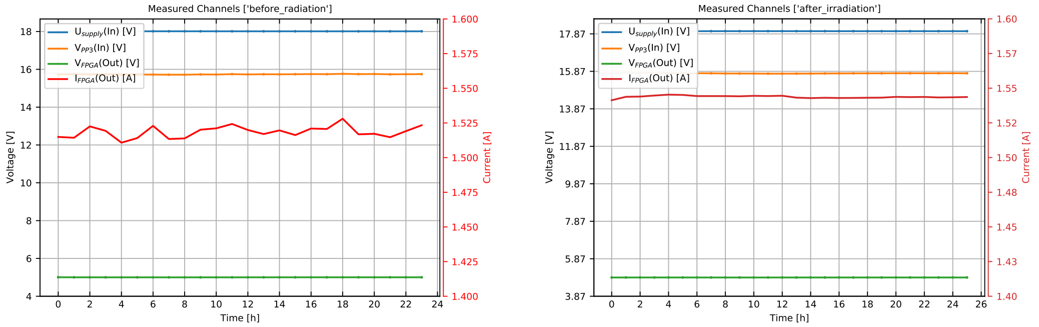
Item	Details
Test Date	June 2024
Focus of Interest	NIEL effect
Facility	TRIGA neutron reactor at JSI
Total Exposure Duration	0.6 hours
Neutron Fluence	10^{12} n/cm^2±10%

The neutron irradiation test for the MOPS-Hub hardware modules was conducted in the tangential channel [178] in the TRIGA Reactor at JSI[1] as part of the EURO-LABS project [179, 180].

[1] Jozef Stefan Institute

9.2.1 PP3-Power Module

The PP3-Power module was tested both before and after irradiation using the setup illustrated in Figure 8.12. During the test, the parameters listed in Table 8.1 were continuously monitored.



(a) Measured channels before neutron irradiation.

(b) Measured channels after neutron irradiation.

Figure 9.6: Long term test for the PP3-Power module (ID =V2.3) before and after neutron irradiation.

Table 9.6 provides a summary of the average values for the PP3-Power module (ID = V2.3) parameters before and after neutron irradiation, based on the data illustrated in Figure 9.6.

Table 9.6: Average values of the PP3-Power module (ID = V2.3) parameters before and after neutron irradiation. Based on the data illustrated in Figure 9.6.

Parameter	**Before Irradiation**	**After Irradiation**
U_{Supply} (V)	18.0±0.01	18.0±0.01
VPP3 (V)	15.7±0.06	15.7±0.03
V_{FPGA} (V)	5.01±0.01	4.87±0.01
I_{FPGA} (A)	1.52±0.04	1.54±0.01

The observations in Table 9.6 suggest no abnormal behavior due to the irradiation, indicating that the module parameters remained stable throughout the testing process.

9.2.2 CAN Interface Card

The CIC was tested both before and after irradiation using the setup illustrated in Figure 8.18. During the test, the parameters listed in Table 8.6 were continuously monitored.

Table 9.7 provides a summary of the average values for the CIC (ID = V4.1) parameters before and after neutron irradiation, based on the data illustrated in Figure B.10.

Table 9.7: Average values of the CIC (ID = V4.1) parameters before and after neutron irradiation. Based on the data illustrated in Figure B.10.

Parameter	Before Irradiation	After Irradiation
U-CIC_PSU (V)	9.36±0.01	9.31±0.01
I-CIC_PSU (mA)	62.2±0.46	195±8.55
U_DCoupler(V)	4.94±0.01	4.92±0.04
I_DCoupler (mA)	11.4±1.54	11.8±1.57
V_CANA(V)	1.92±0.01	1.92±0.01
V_CANB(V)	1.91±0.01	1.94±0.01
U_CIC_VCANA(V)	0.96±0.01	0.99±0.01
U_CIC_VCANB(V)	0.95±0.01	0.97±0.01

The observations in Table 9.7 show a increase in the PSU current consumption after irradiation.

The electrical measurements of the digital coupler, as illustrated in Figures B.10(c) and B.10(d), show only minor changes in voltage before and after neutron irradiation.

In order to test the stability of the power lines, the VCAN is monitored using both the onboard ADC and the 10 bit ADC on the Arduino board. Any mismatch between the two indicates a problem in the regulator system within the CIC. As mentioned in [47], this implies that the *V_CAN (A/B)* values measured directly by the Arduino board are twice the *U_CIC_VCAN (A/B)* measured by the ADC on the CIC due to the effect of the voltage divider circuit on the CIC. This can already be seen in Table 9.7.

Communication with the physical layer was unproblematic, and power readings were successfully obtained. No functional performance issues were identified.

9.2.3 PP3-FPGA Module

The PP3-FPGA was tested both before and after irradiation. During the test, essential local monitoring data, such as Internal supply voltage (VCCINT), Supply voltage for the block RAM (VCCBRAM), Auxiliary supply voltage (VCCAUX), and the FPGA temperature, were collected after each run using the XADC IP provided by Xilinx.

Table 9.8 provides a summary of the average values for the PP3-FPGA (ID = V2.1) parameters before and after neutron irradiation, based on the data illustrated in Figure B.11.

Table 9.8: Average values of the internal PP3-FPGA (ID = V2.1) voltages before and after neutron irradiation. Based on the data illustrated in Figure B.11.

Parameter	**Before Irradiation**	**After Irradiation**
Temperature (°C)	38.0 ± 2.08	36.9 ± 1.56
VCCAUX (V)	1.79 ± 0.01	1.79 ± 0.01
VCCBRAM (V)	1.03 ± 0.02	1.02 ± 0.017
VCCINT (V)	1.03 ± 0.02	1.02 ± 0.02

As shown, the internal FPGA voltages remained stable after neutron irradiation. All values stayed within the acceptable range, as specified by the electrical characteristics of the FPGAs outlined in the Artix-7 data sheet [181].

The functionality of the CRAM of the PP3-FPGA remained stable, and the configuration could be loaded from the memory without difficulty. No severe or abnormal behaviour was observed during configuration.

9.3 Proton Campaign

This section investigates the robustness of the PP3-FPGA against proton beam fluences. The proton beam testing campaign was carried out at Heidelberg Ion Beam Therapy Center (HIT) at different beam settings.

9.3.1 HIT Facility

The HIT facility located at Heidelberg University Hospitals is equipped with heavy ion and proton beams dedicated for cancer treatment. The proton beam line is capable of accelerating protons up to 221 MeV [182]. Table 9.9 illustrates the technical parameters of interest for the proton beam at HIT.

Table 9.9: Proton beam technical parameters at HIT.

Item	**Details**
Energy range	(48 to 221) MeV
Beam intensity range	$(8 \times 10^7$ to $3.2 \times 10^9)$ protons/s
Energy-dependent beam spot size	(32.9 to 8.1) mm (FWHM)

In addition, the maximum beam intensity is 3.2×10^9 protons/s. This intensity is sufficient to mimic the expected hadron fluence, listed in Table 4.3, at PP3 within a short amount of time.

9.3.2 Test Procedure: Setup and Description

Figure 9.7 depicts the schematics of the test setup during proton beam Testing at HIT facility.

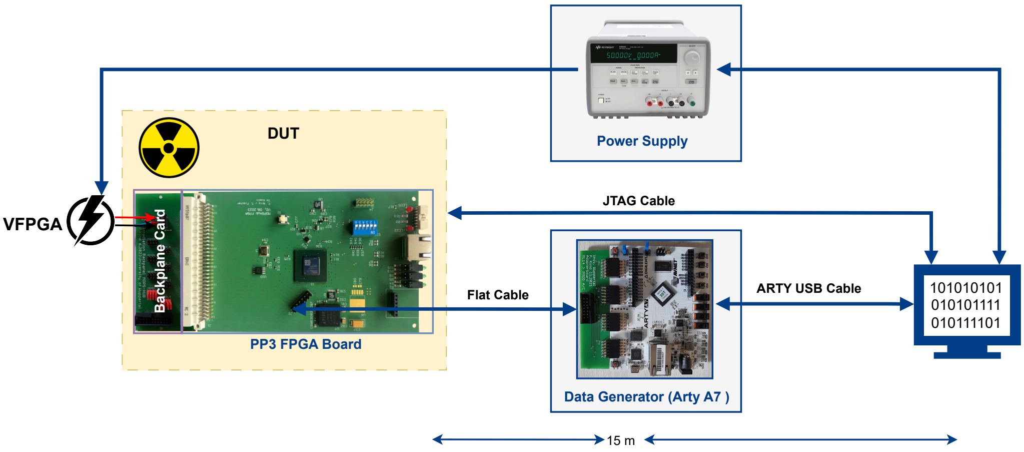


Figure 9.7: Test setup for the PP3-FPGA during proton beam testing at HIT facility.

The setup consists of the PP3-FPGA module (DUT), connected to an **ARTY board**[2] [183]. In order to estimate the SEU cross-section in the CRAM, a test logic consisting of a 3000 bit-long shift register was implemented in the PP3-FPGA firmware. The **ARTY board** acts as a control unit, writing to and reading data from the 3000 bit shift register to check any mismatch.

During operation, two scenarios were considered. The first involved testing the mBAR mechanism, as described in Section 7.9.3, with the external watchdog enabled. Under this condition, the DUT was powered while the SEM IP monitored the CRAM and corrected any detected errors, facilitating the testing of the mBAR feature in the design. In the second scenario, the external watchdog was disabled to avoid unnecessary resets during the reading/writing process to the implemented shift register, allowing for the calculation of the SEU cross-section. Details regarding the irradiation campaign are provided in Table 9.10. During the campaign, the DUT was aligned at the centre of the beam to ensure the entire FPGA on the module was irradiated. The alignment was facilitated by two laser beams, which were used to centre the x-y position of the beam onto the DUT. The beam spot size during the campaign was chosen to cover the **XC7A200T-FPGA** package on the module (23 × 23 mm), without affecting other electronic components on the PP3-FPGA module.

Firmware reloading via JTAG was only considered during the radiation campaign in cases where all copies in the flash memory were corrupted. Additionally, local FPGA parameters such as VCCINT, VCCBRAM, VCCAUX, and the FPGA temperature were collected after each run by the XADC IP provided by Xilinx. This

[2] ARTIX-7 FPGA.

Table 9.10: Detailed information about the proton irradiation campaign for the PP3-FPGA module at HIT.

Item	Details
Test date	August 2024
Facility	Proton beam facility at HIT
Focus of interest	CRAM behaviour and SEU estimation
Total test duration	4 hours
Accumulated proton fluence	7.5×10^{12} protons/cm^2

data is independent of the main data stream and is sampled from on-chip sensors for temperature and power monitoring [161].

To avoid the destruction of the DUT due to SELs, the DUT power supply current was continuously monitored from the **Host Computer** using a python script.

9.3.3 Summary of the Proton Campaign Results

The PP3-FPGA operated at a constant voltage ($V_{\text{FPGA}} = 5\,\text{V}$) for 4 hs, ensuring stable power during radiation exposure as depicted in Figure 9.9(a).

CRAM Behavior

The proton irradiation of the PP3-FPGA with the watchdog enabled was performed with varying fluences, achieving a maximum fluence of 7.5×10^{12} protons/cm^2 during the campaign as depicted in Table 9.11.

Table 9.11: Run details summary for the proton irradiation at HIT for the PP3-FPGA module (ID = V3.3), with the watchdog enabled.

Run	**Energy**	**FWHM**	**Intensity**	**Duration**	**Fluence**	**Rest rate**
	[MeV]	[mm]	[protons/s]	[min]	[protons/cm^2]	[resets/s]
1	48.2	66	3.2×10^9	23	4.42×10^{12}	1
2	102.61	66	3.2×10^9	10	1.92×10^{12}	2
3	145.46	23	3.2×10^9	5	9.60×10^{11}	1
4	145.46	23	8.0×10^7	12	5.76×10^{10}	1
5	145.46	23	4.0×10^8	5	1.20×10^{11}	1
6	145.46	23	1.2×10^8	10	7.20×10^{10}	1

Figure 9.8, depicts the behaviour of the CRAM throughout the campaign.

During irradiation, the average reset rate was observed to be up to 2 resets/s. This elevated reset rate is attributed to a high upset rate in the CRAM. While some upsets are correctable by the SEM IP, as indicated by the *status_correction* signal, there remains a risk of MBU affecting more than two bits per frame simultaneously. Such events can overwhelm the protection mechanisms of the SEM IP. This issue is

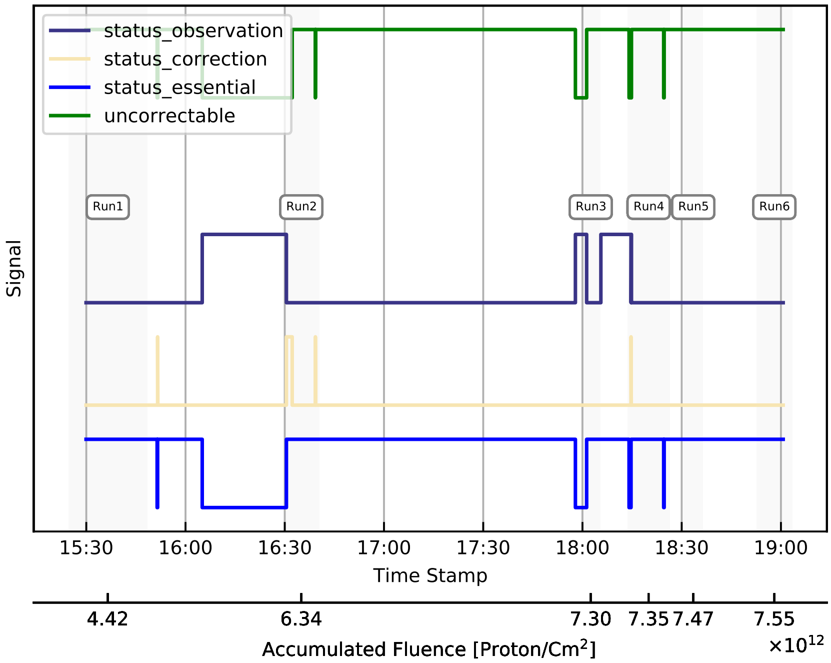


Figure 9.8: CRAM behavior of the PP3-FPGA module using SEM IP (ID = V3.3), with the watchdog enabled, during the proton campaign.

highlighted by the *uncorrectable* and *status_essential* outputs, which provide insights into the nature of the detected errors and their impact on system functionality.

Accumulation of upsets, particularly in critical areas in critical circuits such as clock generators, can disrupt the system's operation. At a certain point, the clock network stopped functioning within the FPGA. Unrecoverable errors cannot be repaired without re-downloading the FPGA firmware using the mBAR mechanism.

SEU Estimation

SEU estimation was performed under varying fluences, with the watchdog disabled, reaching a maximum of 2×10^{12} protons/cm^2, approximately 10^4 times the expected fluence at the PP3 location.

With the help of the **ARTY Board** depicted in Figure 9.7, a specific pattern was continuously written to the shift register in the PP3-FPGA. Data was shifted through the register at a 1 kHz clock rate, with a hold time of 1 s and then read back. When a mismatch was found, the ARTIX-7 FPGA reported an SEU failure, incremented the failure counter, and sent the information to the Host computer for diagnosis. The calculated SEU cross-sections for the PP3-FPGA during the campaign are listed in Table 9.12.

The SEU cross-section in Table 9.12 is calculated based on Equation 5.7. A total of 20 SEUs were observed during the campaign and considered for the analysis. The

Table 9.12: Run details summary for the proton irradiation at HIT for the PP3-FPGA module (ID = V3.1), with the watchdog disabled. The SEU cross-section is calculated based on Equation 5.7.

Run	Energy	FWHM	Intensity	Duration	Fluence	SEUs	σ
	[MeV]	[mm]	[protons/s]	[min]	[protons/cm^2]	[N]	[cm^2/bit]
7	145.46	11.5	8×10^8	37	8×10^{11}	7	$2.92{\times}10^{-15}$
8	155.82	10.8	8×10^8	39	8×10^{11}	9	$3.75{\times}10^{-15}$
9	165.89	10.2	8×10^8	17	4×10^{11}	4	$3.33{\times}10^{-15}$

results agree with other measurements done by Xilinx on the same **Artix-7** family [81] and other experiments using different strategies as listed in Table 4.3.

As detailed in Table 9.12, the cross-section is energy independent within the uncertainty. These results agree with the simulation studies for FPGA devices for the upset rates at different energies discussed in Section 5.3.2.

The estimated rate of SEUs in the CRAM of the **XC7A200T-FPGA**[3] at PP3-location, based on Equation 5.7 can reach up to 25349 SEUs in 10 years of the LHC operation, assuming the maximum SEUs cross-section estimated in Table 9.12(See Equation 9.1).

$$\begin{aligned} N_{\mathrm{SEU}} = 3.75 \times 10^{-15}\,[\mathrm{cm}^2/\mathrm{bit}] \times 2 \times 10^{-7}\,[/\mathrm{cm}^2/\mathrm{pp}] \\ \times 40 \times 10^{6}\,[\mathrm{pp/s}] \times 200\,[\mathrm{Collisions}] \\ \times 3.14 \times 10^{7}\,[\mathrm{s}] \times 10\,[\mathrm{years}] \\ \times 13\,455\,360\,[\mathrm{bits}] \\ = 25349\ \mathrm{SEUs} \end{aligned} \tag{9.1}$$

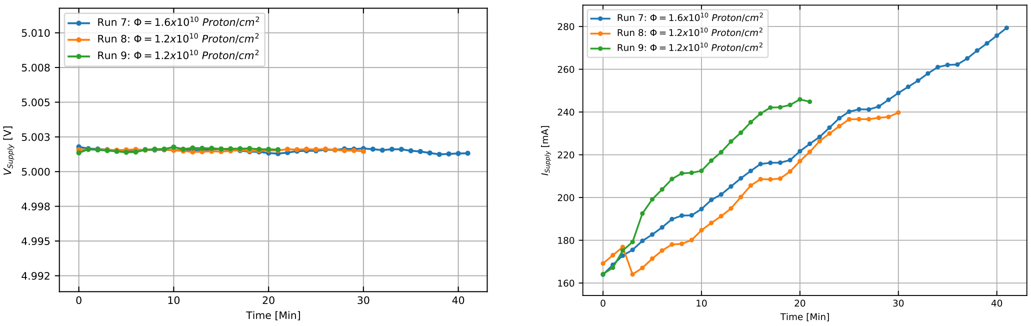
Where the value of the particle fluence is the expected HEH fluence at PP3 location ($\phi = 2 \times 10^{-7}\,\mathrm{cm}^2/\mathrm{pp}$), as listed in Table 4.3. The number of proton collisions is driven by the ultimate goal of the HL-LHC to manage an average of at least 140 pile-up events per bunch crossing, with a target of reaching 200 events [1].

Current and Voltage Behavior

The static current consumed by the **XC7A200T-FPGA** when powered on without any switching activities, according to the datasheet, gives a total maximum current of 462 mA when all the 10 I/O banks are considered [181].

A preset limit was established during the irradiation campaign at 510 mA, which is 10% above the calculated static current to safeguard the FPGA against potential over-current scenarios caused by SELs.

[3] Total RAM Bits in XC7A200T-FPGA = 13455360 bits



(a) Supply voltage during proton irradiation.

(b) Supply current during proton irradiation.

Figure 9.9: Supply power to the PP3-FPGA module (ID = V3.1), with the watchdog disabled, during the proton campaign.

Throughout the irradiation process, no sudden increases in the power supply current were observed. This stability suggests that no latchups occurred, which typically cause abrupt current spikes. However, it was noted that as SEUs accumulated, there was a gradual increase in the current consumption of the FPGA, as depicted in Figure 9.9(b). When reconfiguration is enabled, the FPGA's configuration memory is refreshed, clearing these errors and restoring the FPGA to its initial, error-free state. This results in the current consumption returning to its original level, as the FPGA resumes normal operation. This phenomenon is consistent with observations reported in other SEU studies [184–186].

Chapter 10

Conclusion

In the course of this thesis, a new FPGA-based system called MOPS-Hub was developed as part of the upgrade to the Detector Control System (DCS) for the ATLAS ITk Pixel detector, addressing the challenges presented by the HL-LHC. MOPS-Hub is designed to aggregate monitoring data between an on-detector ASIC, referred to as Monitoring Of Pixel System (MOPS), and the DCS.

10.1 Status and Summary

The MOPS-Hub serves as a critical interface, enabling communication between the MOPS chips via CAN bus interfaces and delivering the necessary power through components housed within the MOPS-Hub crate. This crate will operate in racks on the walls of the ATLAS cavern, referred to as Patch Panel 3 (PP3).

Each MOPS-Hub crate interfaces with 32 CAN buses via CAN Interface Card (CIC)s, with one PP3-FPGA managing 16 of these buses. Power is supplied by two independent PP3-Power modules. Data collected from the CAN buses is forwarded through eLink signals to external components, including EMCI and EMP, as part of the DCS.

The firmware design in the PP3-FPGA, detailed in Chapter 7, was driven by the requirements of the MOPS chip, which communicates via CAN and eLink specifications defined by CERN for chip-to-chip communication in high radiation environments.

Through extensive testing described in Chapter 8, the design and functionality of the MOPS-Hub crate were validated using automated test setups. The reliability of the system was confirmed, demonstrating that it meets the requirements for operation in the PP3 environment, including power delivery and communication protocols. The validation setups are intended to be repurposed for quality control during the pre-production phase of MOPS-Hub.

Given the radiation nature of the PP3 environment, special attention was paid to the hardware components. Radiation campaigns confirmed that key components, such as the CIC, PP3-Power modules, and PP3-FPGA, can withstand Total Ionizing Dose (TID) up to 168 Gy, with tolerance to NIEL effects and neutron fluences of $10^{12}\,\mathrm{n/cm^2}$.

The PP3-FPGA was rigorously evaluated for its SEU resilience through the proton beam campaign at the HIT facility. System robustness, including Multi-boot Auto Reconfiguration (mBAR) and bit correction, were assessed during this campaign.

The comprehensive radiation characterization demonstrated that the FPGA could operate reliably in a radiation environment, with HEH fluences reaching a maximum of 2×10^{12} protons/cm^2. The estimated SEU cross section in the CRAM of the **XC7A200T-FPGA** was found to be 3.75×10^{-15} cm^2, consistent with measurements reported in other studies.

In conclusion, this thesis represents a significant advancement in the development of the MOPS-Hub data aggregator system for the ATLAS Phase II upgrade. The successful design, validation, and testing of the MOPS-Hub demonstrate its capability to meet the requirements of the DCS, including its performance in the challenging radiation environment of the PP3 location.

10.2 Outlook

The basic functionality of the MOPS-Hub has been successfully implemented in the first prototype and has been demonstrated to be operational. However, the full system testing, as detailed in Section 8.4, was limited to a few components due to hardware constraints. Additionally, while several enhancements were proposed to optimize the system further, unfortunately, time constraints did not allow for their full implementation. Recommended improvements for this project can be classified into the following categories

System Level Improvements

In the immediate future, the pre-production phase will involve up to 9 fully equipped MOPS-Hub crates. During this phase, the readout setup will be scaled to its full capacity and will perform additional measurements. Future work should involve repeating the previous measurements on the complete setup and potentially improving them by testing under varying conditions. For example, testing the FPGA's performance under varying temperatures, and subjecting the FPGA to simulated errors is recommended.

Preliminary studies in a magnetic field with strengths of up to (280 to 380) mT indicate that the power consumption of the PP3-FPGA board increases, causing a voltage drop of 1 V. This drop is primarily due to the LTM4619 voltage regulator mounted on the PP3-FPGA board, which was unable to handle the magnetic field strength, suggesting potential issues with the system's performance under these conditions. This observation has also been noted by other teams, indicating that the efficiency of the LTM4619 dropped significantly at around 200 mT [187]. Therefore, testing the system's performance with switched voltages in a lower magnetic field of 100 mT, as anticipated at PP3, is recommended.

At the time of writing this thesis, the full EMCI/EMP chain described in Section 4.7 was not available. Instead, an FPGA-based setup was developed to consolidate the entire functionality of the EMCI/EMP chain into a single system, referred to as the MOPS-Hub Readout Board. Although this system has proven its reliability, it does not fully reflect the behavior of the final system.

Firmware Level Improvements

The modular architecture adopted in the PP3-FPGA firmware, as described in Chapter 7, enables the replacement or updating of individual components without disrupting the entire system. One potential area for future development is the implementation of partial reconfiguration capability, which is currently not yet in place. However, the firmware still allows for the integration of this feature, enabling later updates without requiring physical access to the FPGA. This could be achieved with the help of the ICAP controller through eLink or other interfaces, allowing for changes to specific portions of the FPGA's configuration while the rest of the device remains operational.

Additionally, although the functionality of the watchdog was verified during the irradiation campaign, the high reset rate observed suggests that there may be a better way to trigger the external watchdog.

Acknowledgments

The work I have done during my time as a PhD student would not have been possible without the help of several people. I would like to thank everyone who supported and encouraged me through this phase of my studies.

First, my deep thanks go to Prof. Dr. Christian Zeitnitz, who trusted in my skills and abilities, allowing me the opportunity to work on this topic under his supervision in the physics department. His knowledge and experience in developing electronics for particle physics experiments were invaluable. His ability to explain problems in a way that made them feel manageable was a tremendous help.

I also want to express my gratitude to Prof. Dr. Michael Karagounis for his technical supervision, particularly at the firmware level, and for his collaboration alongside his team on the project.

Many thanks go to my colleagues at the University of Wuppertal. Susanne Kersten and Dr. Tobias Flick, with their extensive experience in the ATLAS ITk project, were instrumental in helping to schedule and manage timelines, proofreading documents, and providing invaluable suggestions. Peter Kind was a tremendous help in developing test boards for the chip and sharing his experience in electronics. Dr. Marius Wensing was always there to save the day when it came to FPGA challenges, and his advice was immensely valuable. A special thanks to my colleague Rizwan Ahmad, who assisted me in understanding the MOPS chip at the logic level and supported me with his knowledge of system testing and validation. Our discussions, both professional and personal, have been incredibly valuable to me.

Finally, my deep and sincere gratitude goes to my family for their continuous love, support, and encouragement. I am forever indebted to my parents for the opportunities and experiences that have shaped who I am today. A special thanks to my father, who inspired me to pursue my dreams and follow my own path. My sincere gratitude goes out to my wife Dina; without her perseverance, patience, and support, this journey would not have been possible.

Bibliography

[1]G. Apollinari et al., *High-Luminosity Large Hadron Collider (HL-LHC): Preliminary Design Report*, tech. rep. (CERN, Geneva, 2015) (cit. on pp. 1, 3, 129).

[2]T. Flick, "The phase II ATLAS Pixel upgrade: the Inner Tracker (ITk)", Journal of Instrumentation **12**, C01098–C01098 (2017) (cit. on pp. 1, 6).

[3]A. Qamesh and R. Ahmad and D. Ecker and T. Fischer and M. Karagounis and P. Kind and S. Kersten and T. Krawutschke and L. Schreiter and C. Zeitnitz and on behalf of the ATLAS ITK collaboration, "System integration of ATLAS ITK Pixel DCS ASICs", Journal of Instrumentation **18**, C05003 (2023) (cit. on p. 1).

[4]M. Drewes et al., "New long-lived particle searches in heavy-ion collisions at the LHC", Phys. Rev. D **101**, 055002 (2020) (cit. on p. 3).

[5]L. Evans and P. Bryant, "LHC Machine", Journal of Instrumentation **3**, S08001 (2008) (cit. on p. 3).

[6]J. Ellis, "Beyond the standard model with the LHC", Nature **448**, 297–301 (2007) (cit. on p. 3).

[7]I. Kraus (ALICE), "ALICE pp physics programme", in 47th International Winter Meeting on Nuclear Physics (July 2009) (cit. on p. 3).

[8]R. Aaij et al. (LHCb), "Implications of LHCb measurements and future prospects", Eur. Phys. J. C **73**, 2373 (2013) (cit. on p. 3).

[9]A. M. Sirunyan et al. (CMS), "Electron and photon reconstruction and identification with the CMS experiment at the CERN LHC", JINST **16**, P05014 (2021) (cit. on p. 3).

[10]G. Aad et al. (ATLAS), "The ATLAS Experiment at the CERN Large Hadron Collider", JINST **3**, S08003 (2008) (cit. on pp. 3, 4, 6).

[11]P. L. Rocca and F. Riggi, "The upgrade programme of the major experiments at the Large Hadron Collider", Journal of Physics: Conference Series **515**, 012012 (2014) (cit. on p. 3).

[12]G. Aad et al., "Observation of a new particle in the search for the Standard Model Higgs boson with the ATLAS detector at the LHC", Physics Letters B **716**, 1–29 (2012) (cit. on p. 4).

[13]O. Brandt, "Search for Supersymmetry in Trilepton Final States with the ATLAS Detector and the Alignment of the ATLAS Silicon Tracker", PhD thesis (Oxford U., 2009) (cit. on p. 4).

[14] S. Mehlhase (ATLAS), "ATLAS detector slice (and particle visualisations)", CERN Document Server (2021) (cit. on p. 5).

[15] B. Aubert, B. Beaugiraud, and J. Colas, "Construction, assembly and tests of the ATLAS electromagnetic barrel calorimeter", Nuclear Instruments and Methods in Physics Research Section A **558**, 388–418 (2006) (cit. on p. 6).

[16] *Technical Design Report for the ATLAS Inner Tracker Pixel Detector*, tech. rep. (CERN, Geneva, Sept. 2017) (cit. on pp. 6–8, 10–14).

[17] *Expected tracking and related performance with the updated ATLAS Inner Tracker layout at the High-Luminosity LHC*, tech. rep., Accessed on Jan 1, 2024 (CERN, Geneva, 2021) (cit. on p. 7).

[18] J. Chistiansen and M. Garcia-Sciveres, *RD Collaboration Proposal: Development of pixel readout integrated circuits for extreme rate and radiation*, tech. rep. CERN-LHCC-2013-008. LHCC-P-006, Accessed on Jan 1, 2024 (CERN, Geneva, June 2013) (cit. on p. 6).

[19] S. Terzo et al., "Novel 3D Pixel Sensors for the Upgrade of the ATLAS Inner Tracker", Frontiers in Physics **9**, 624668 (2021) (cit. on p. 6).

[20] T. R. L. Rossi P. Fischer and N. Wermes, *Pixel Detectors: From Fundamentals to Applications* (Springer Science and Business Media, 2006) (cit. on pp. 7, 40).

[21] T. Stockmanns et al., "Realisation of serial powering of ATLAS pixel modules", in Ieee symposium conference record nuclear science 2004. Vol. 2 (2004), 894–898 Vol. 2 (cit. on p. 7).

[22] T. Stockmanns et al., "Serial powering of pixel modules", Nuclear Instruments and Methods in Physics Research Section A **511**, Proceedings of the 11th International Workshop on Vertex Detectors, 174–179 (2003) (cit. on p. 7).

[23] M. Standke, "Hybrid Pixel Readout Chip Verification, Characterization and Wafer Level Testing for the ATLAS-ITK Upgrade at the HL-LHC", Dissertation (University of Bonn, Bonn, Germany, 2023) (cit. on p. 8).

[24] A. B. Poy et al., "The detector control system of the ATLAS experiment", Journal of Instrumentation **3**, P05006 (2008) (cit. on pp. 9, 14).

[25] ATLAS Collaboration, *Approved Detector Control System (DCS) Figures*, Accessed on Nov 1, 2023, 2014 (cit. on pp. 10, 11).

[26] W. Mahnke, S.-H. Leitner, and M. Damm, *Opc unified architecture* (Springer-Verlag Berlin Heidelberg, Mar. 2009), p. 339 (cit. on p. 9).

[27] P. P. Nikiel et al., "OPC Unified Architecture within the Control System of the ATLAS Experiment", (2014) (cit. on p. 9).

[28] *SIMATIC WinCC Open Architecture: Version 3.19 Documentation*, Siemens AG (2022) (cit. on p. 10).

[29] S. Kersten et al., "The ITk interlock hardware protection system", Nuclear Instruments and Methods in Physics Research Section A **1045**, 167613 (2023) (cit. on pp. 11, 12).

[30] S. Kersten et al., "The ITk Common Monitoring and Interlock System", 10.18429/JACoW-ICALEPCS2019-THCPR06 (2020) (cit. on p. 12).

[31] A. Walsemann et al., "A CANopen based prototype chip for the Detector Control System of the ATLAS ITk Pixel Detector", PoS **TWEPP2019**, 013. 5 p (2020) (cit. on p. 13).

[32] A. Heggelund and on behalf of the ATLAS collaboration, "Overview of the ATLAS ITk pixel detector", Journal of Instrumentation **18**, C02014 (2023) (cit. on p. 13).

[33] *Specification of ITk Pixel Services*, tech. rep. AT2-IP-CD-0028, Accessed on Jan 7, 2024 (2022) (cit. on pp. 13, 22).

[34] *ATLAS DCS Requirement Document for HL-LHC*, tech. rep. ATU-GE-ES-0004 (CERN, Geneva, June 2020) (cit. on p. 13).

[35] D. Bertsche, *Opto-Box*, tech. rep. (CERN, Geneva, 2016) (cit. on p. 14).

[36] L. Franconi, "The Opto-electrical conversion system for the data transmission chain of the ATLAS ITk Pixel detector upgrade for the HL-LHC", 10.1088/1742-6596/2374/1/012105 (2022) (cit. on p. 14).

[37] S. Ryu and on behalf of the ATLAS TDAQ Collaboration, "FELIX: The new detector readout system for the ATLAS experiment", Journal of Physics: Conference Series **898**, 032057 (2017) (cit. on p. 14).

[38] B. I. Hallgren et al., "The Embedded Local Monitor Board (ELMB) in the LHC Front-end I/O Control System", 10.5170/CERN-2001-005.325 (2001) (cit. on p. 14).

[39] G. Baribaud et al., "Recommendations for the use of fieldbuses at CERN in the LHC era", (1997) (cit. on p. 14).

[40] H. B. et al., "Design and Implementation of the ATLAS Detector Control System", IEEE Transactions on Nuclear Science **51**, https://doi.org/10.1109/tns.2004.828523 (2004) (cit. on p. 14).

[41] T. A. C. et al., "The ATLAS Experiment at the CERN Large Hadron Collider", Journal of Instrumentation **3**, S08003 (2008) (cit. on p. 14).

[42] K. S. Nicpon et al., "The Embedded Local Monitor Board upgrade proposals", PoS **TWEPP2018**, 034 (2019) (cit. on p. 14).

[43] R. Ahmad, "The Monitoring of Pixel System (MOPS) chip for the Detector Control System of the ATLAS ITk Pixel Detector", Journal of Physics: Conference Series **2374**, 10.1088/1742-6596/2374/1/012105 (2022) (cit. on p. 14).

[44] R. Ahmad, "Development and characterization of the monitoring of pixel system (mops) chip to monitor the atlas itk pixel detector" (Bergische Universität Wuppertal, 2023) (cit. on pp. 15, 28, 71, 85).

[45] *DCS: Requirements document for HL-LHC*, tech. rep. ATU-GE-ES-0004, Accessed on Jan 7, 2022 (2019) (cit. on p. 15).

[46] *Specification for the Pixel DCS ASIC: Monitoring Of Pixel System*, tech. rep. AT2-IP-ES-0001, version v3.2, Accessed on Nov 7, 2023 () (cit. on pp. 15, 31, 113, 116).

[47] *Construction of MOPS-Hub*, tech. rep. AT2-IP-EP-0021, Accessed on Jan 7, 2022 (2021) (cit. on pp. 16, 21, 31, 32, 36, 124).

[48] *EMCI specification document*, tech. rep. (CERN, Geneva, Apr. 2021) (cit. on p. 16).

[49] D. B. Serrano et al., "Description and status of the EMCI-EMP interface", Journal of Instrumentation **17**, C06012 (2022) (cit. on pp. 16, 37).

[50] D. Giugni et al., *Specifications for the ATLAS ITk Pixel Services, Technical Requirements*, tech. rep. AT2-IP-EP-0007, version 2, version v.3, Accessed on April 7, 2024 (ATLAS Collaboration, 2020) (cit. on p. 20).

[51] *Specifications for the Power Supplies*, tech. rep. AT2-IP-ES-0014 v.2, Accessed on Jan 7, 2022 (2019) (cit. on pp. 20, 23, 31, 102).

[52] *ITk grounding and shielding requirements*, tech. rep. AT2-I-EP-0001, Accessed on Jan 7, 2024 (2021) (cit. on pp. 20, 22).

[53] *ITK Pixel Type III and Type IV Services*, tech. rep. AT2-IP-ES-0019, Accessed on April 7, 2024 (2024) (cit. on p. 22).

[54] *Pixel Electrical Services*, tech. rep. ATL-IP-ES-0069, Accessed on Jan 7, 2024 (2004) (cit. on p. 22).

[55] *Specifications for the ATLAS ITk Pixel Services*, tech. rep. AT2-IP-EP-0007, Accessed on Jan 7, 2024 (2023) (cit. on p. 22).

[56] A. Devices, *ADUM3402 Digital Quad-Channel, Digital Isolator, Enhanced System-Level ESD Reliability*, Accessed on Nov 7, 2023 (2017) (cit. on pp. 23, 32).

[57] R. P. GmbH, *RSO-S(D)(Z) DC/DC Converter*, Accessed on Oct 7, 2024 (2019) (cit. on pp. 23, 33).

[58] ATLAS, *Radiation Background Simulations*, https://twiki.cern.ch/twiki/bin/viewauth/Atlas/RadiationBackgroundSimulationsStep3X#S3_1Q6_Full_hall_region (visited on 09/10/2019) (cit. on pp. 23, 24, 166, 167).

[59] I. Dawson, ed., *Radiation effects in the LHC experiments: Impact on detector performance and operation*, Vol. 1/2021, CERN Yellow Reports: Monographs (CERN, Geneva, 2021) (cit. on pp. 23, 24, 49).

[60] M. Aleksa et al., "Results of the ATLAS solenoid magnetic field map", J. Phys. Conf. Ser. **110**, edited by R. Barlow, 092018 (2008) (cit. on p. 24).

[61] A. Yamamoto et al., "Progress in ATLAS central solenoid magnet", IEEE Trans. Appl. Supercond. **10**, 353–6 (2000) (cit. on p. 24).

[62] N. Sawyer, *CAN Specification 2.0*, version v2.0, Accessed on Jan 17, 2022, Robert Bosch GmbH (1991) (cit. on pp. 25, 28, 71).

[63] S. Bonacini, K. Kloukinas, and P. Moreira, "E-link: A Radiation-Hard Low-Power Electrical Link for Chip-to-Chip Communication", 10.5170/CERN-2009-006.422 (2009) (cit. on p. 29).

[64] *lpGBT Documentation*, tech. rep. (lpGBT Design Team, Dec. 2020) (cit. on p. 29).

[65] T. Fischer, "Design and Test of a radiation tolerant FPGA system for the use in a particle accelerator experiment at CERN" (2022) (cit. on pp. 29, 36).

[66] A. X. Widmer and P. A. Franaszek, "A dc-balanced, partitioned-block, 8b/10b transmission code.", IBM J. Res. Dev. **27**, 440–451 (1983) (cit. on pp. 29, 30, 79).

[67] D. Incorporated, *AP64500 synchronous buck converter*, Accessed on Nov 7, 2023 (2011) (cit. on pp. 31, 102, 104).

[68] TDK, *Current-compensated ring core double chokes*, Accessed on Nov 7, 2023 (2016) (cit. on p. 31).

[69] T. Instrument, *TLV1117LV 1-A, Positive Fixed-Voltage, Low-Dropout Regulator*, Accessed on Nov 7, 2023 (2011) (cit. on p. 32).

[70] N. Semiconductors, *74LVC1T45 Level Shifter Datasheet*, Accessed on Nov 7, 2023 (2012) (cit. on p. 32).

[71] T. Instrument, *TLV350x 4.5-ns, High-Speed Comparator*, Accessed on Nov 7, 2023 (2016) (cit. on p. 33).

[72] M. Integrated, *MAX8880/MAX8881 Low-Dropout Linear Regulator*, Accessed on Nov 7, 2023 (2007) (cit. on p. 34).

[73] M. T. Inc., *MCP23S08/MCP23008 Data Sheet*, Accessed on Nov 7, 2023 (2007) (cit. on pp. 34, 78).

[74] M. Integrated, *MAX4581 Low-Voltage, CMOS Analog Multiplexers/Switches*, Accessed on Nov 7, 2023 (2012) (cit. on p. 34).

[75] C. Logic, *CS5523 16-bit 4-channel ADCs*, Accessed on Nov 7, 2023 (2009) (cit. on pp. 35, 75, 76).

[76] T. Instruments, *Dual-processor voltage supervisor with power-fail & watchdog timer*, Accessed on Nov 7, 2023 (2000) (cit. on p. 36).

[77] D. Lee, G. Swift, and M. Wirthlin, "An Analysis of High-Current Events Observed on Xilinx 7-Series and Ultrascale Field-Programmable Gate Arrays", IEEE Transactions on Nuclear Science (2016) (cit. on pp. 36, 58).

[78] D. Lee, M. Wirthlin, and G. Swift, "Single-Event Characterization of the 28 nm Xilinx Kintex-7 Field-Programmable Gate Array under Heavy Ion Irradiation", IEEE Transactions on Nuclear Science, 10.1109/REDW.2014.7004595 (2014) (cit. on pp. 36, 58).

[79] M. Cannon et al., "Evaluating Xilinx 7 series GTX transceivers for use in high energy physics experiments through proton irradiation", IEEE Transactions on Nuclear Science **62**, 2695–2702 (2015) (cit. on pp. 36, 58).

[80] M. J. Wirthlin, H. Takai, and A. Harding, "Soft error rate estimations of the Kintex-7 FPGA within the ATLAS Liquid Argon (LAr) Calorimeter", Journal of Instrumentation **9**, C01025 (2014) (cit. on pp. 36, 58).

[81]I. Xilinx, *Device Reliability Report*, UG116, User Guide (2018), p. 106 (cit. on pp. 36, 37, 58, 129).

[82]X. Hu et al., "A Multi-Layer SEU Mitigation Strategy to Improve FPGA Design Robustness for the ATLAS Muon Spectrometer Upgrade", Nucl. Instrum. Meth. A **939**, 30–35 (2019) (cit. on pp. 36, 37).

[83]X. Hu et al., "Radiation characterization of COTS components for a signal packet router in the upgrade of the ATLAS muon spectrometer", Journal of Instrumentation **14**, P10023–P10023 (2019) (cit. on pp. 36, 58).

[84]J. Troska, *Versatile Link+ Transceiver (VTRx+)*, Technical Specification part 2.1, Accessed on Feb 7, 2024 (CERN) (cit. on p. 37).

[85]D. converters team, *FEASTMP Datasheet*, Accessed on Feb 7, 2024 (cit. on p. 37).

[86]P. Moreira, "The lpGBT: a radiation tolerant ASIC for Data, Timing, Trigger and Control Applications in HL-LHC", in Topical workshop on electronics for particle physics (Sept. 2019) (cit. on p. 37).

[87]C. e. a. Patrignani (Particle Data Group), "Review of particle physics", Chin. Phys. **C40**, 100001 (2016) (cit. on pp. 39–41).

[88]H. Kolanoski and N. Wermes, *Particle Detectors: Fundamentals and Applications* (Oxford University Press, June 2020) (cit. on pp. 40, 44, 49).

[89]E. Hussein, *Handbook on Radiation Probing, Gauging, Imaging and Analysis: Volume I:Basics and Techniques* (Springer Netherlands, 2003) (cit. on p. 41).

[90]F. Faccio, "COTS for the LHC radiation environmentain: the rules of the game", in 6th Workshop on Electronics for LHC Experiments (2000) (cit. on pp. 42, 44, 45, 55).

[91]G. Messenger and M. Ash, *The Effects of Radiation on Electronic Systems*, 2nd (Van Nostrand Reinhold, New York, 1992) (cit. on p. 42).

[92]T. Oldham and F. McLean, "Total ionizing dose effects in MOS oxides and devices", IEEE Transactions on Nuclear Science **50**, 483–499 (2003) (cit. on pp. 42, 43).

[93]R. C. Hughes, "Charge-Carrier Transport Phenomena in Amorphous SiO_2: Direct Measurement of the Drift Mobility and Lifetime", Phys. Rev. Lett. **30**, 1333–1336 (1973) (cit. on p. 42).

[94]T. P. Ma and P. V. Dressendorfer, *Ionizing Radiation Effects in MOS Devices and Circuits* (John Wiley and Sons, 1989) (cit. on p. 42).

[95]J. Schwank et al., "Radiation effects in MOS oxides", IEEE Transactions on Nuclear Science **55**, 1833–1853 (2008) (cit. on p. 42).

[96]P. C. Adell and L. Z. Scheick, "Radiation Effects in Power Systems", IEEE Transactions on Nuclear Science **60**, 1929–1952 (2013) (cit. on p. 42).

[97]J. R. Srour and J. M. McGarrity, "Radiation effects on microelectronics in space", Proc. IEEE; (United States) **76:11**, 10.1109/5.90114 (1988) (cit. on p. 44).

[98] C. Leroy and P. G. Rancoita, "Particle interaction and displacement damage in silicon devices operated in radiation environments", Reports on Progress in Physics **70**, 493 (2007) (cit. on pp. 44, 49).

[99] T. A. et al., "A neutron irradiation facility for damage studies", Nuclear Instruments and Methods in Physics Research Section A **345**, 303–307 (1994) (cit. on p. 44).

[100] D. Rybka et al., "Development of a Digital Photometer for the Evaluation of COTS LED Fast Neutron Dosimeter; Integrated Measurement Systems for Electronic Devices Operating in Radiation Environment", Proceedings of SPIE, 10.13140/RG.2.2.16619.31522 (2005) (cit. on p. 45).

[101] P. Dodd and L. Massengill, "Basic mechanisms and modeling of single-event upset in digital microelectronics", IEEE Transactions on Nuclear Science **50**, 583–602 (2003) (cit. on p. 45).

[102] F. Sexton, "Destructive single-event effects in semiconductor devices and ICs", IEEE Transactions on Nuclear Science **50**, 603–621 (2003) (cit. on p. 45).

[103] M. Huhtinen and F. Faccio, "Computational method to estimate Single Event Upset rates in an accelerator environment", Nuclear Instruments and Methods in Physics Research Section A **450**, 155–172 (1999) (cit. on pp. 45, 48, 50).

[104] T. Böhlen et al., "The FLUKA Code: Developments and Challenges for High Energy and Medical Applications", Nuclear Data Sheets **120**, 211–214 (2014) (cit. on pp. 45, 49, 166, 167).

[105] P. E. Dodd and L. W. Massengill, "Basic mechanisms and modeling of single-event upset in digital microelectronics", IEEE Transactions on Nuclear Science **50 III**, 583–602 (2003) (cit. on p. 46).

[106] V. Ferlet-Cavrois, L. W. Massengill, and P. Gouker, "Single event transients in digital CMOS - A review", IEEE Transactions on Nuclear Science **60**, 1767–1790 (2013) (cit. on p. 47).

[107] R. Koga et al., "Single event functional interrupt (SEFI) sensitivity in microcircuits", in Radecs 97. fourth european conference on radiation and its effects on components and systems (cat. no.97th8294) (1997), pp. 311–318 (cit. on p. 47).

[108] G. Torrens et al., "A 65-nm Reliable 6T CMOS SRAM Cell with Minimum Size Transistors", IEEE Transactions on Emerging Topics in Computing **7**, 447–455 (2019) (cit. on p. 47).

[109] *The FLUKA Code: Description And Benchmarking* (Sept. 2007), pp. 31–49 (cit. on pp. 48, 49).

[110] G. e. a. Hofman, "Light-hadron induced SER and scaling relations for 16- and 64-Mb DRAMS", IEEE Transactions on Nuclear Science **47**, 403–407 (2000) (cit. on p. 48).

[111] *ATLAS: Technical proposal for a general-purpose pp experiment at the Large Hadron Collider at CERN*, tech. rep., Accessed on Sep 1, 2023 (Geneva, 1994) (cit. on p. 49).

[112] A. Fasso et al., “The FLUKA code: Present applications and future developments”, eConf **C0303241**, MOMT004 (2003) (cit. on p. 49).

[113] *FLUKA Web Site*, https://fluka.cern/, CERN, Geneva, July 2020 (cit. on p. 49).

[114] H. R. Zohouri et al., “Evaluating and Optimizing OpenCL Kernels for High Performance Computing with FPGA”, in Proceedings of the international conference for high performance computing, networking, storage and analysis, 16 (2016), 35:1–35:12 (cit. on p. 51).

[115] C. e. a. Zhang, “High-Performance and Energy-Efficient FPGA-GPU-CPU Heterogeneous System Implementation”, 477–492 (2021) (cit. on pp. 51–53).

[116] C. Maxfield, *The Design Warrior’s Guide to FPGAs Devices, Tools, and Flows*, 1st (Newnes, Amsterdam, 2004) (cit. on p. 51).

[117] I. Xilinx, *7 Series FPGAs SelectIO Resources*, https://www.xilinx.com/support/documentation/user_guides/ug471_7Series_SelectIO.pdf (visited on 08/05/2018) (cit. on pp. 51, 81).

[118] I. Kuon, R. Tessier, and J. Rose, “FPGA Architecture: Surveyand Challenges”, Foundations and Trends® in Electronic Design Automation **2**, http://dx.doi.org/10.1561/1000000005 (2008) (cit. on p. 51).

[119] J. M. Benedetto and C. C. Hafer, “Ionizing radiation response of an amorphous silicon based antifuse”, in 1997 ieee radiation effects data workshop nsrec snowmass (1997), pp. 101–104 (cit. on p. 51).

[120] V. Placinta et al., “Measurements of radiation effects in an antifuse FPGA”, Nuclear Instruments and Methods in Physics Research Section A **1055**, 168551 (2023) (cit. on p. 51).

[121] J. L. Nunes, J. C. Cunha, and M. Zenha-Rela, “On the Effects of Cumulative SEUs in FPGA-Based Systems”, in 2016 12th european dependable computing conference (edcc) (2016), pp. 89–96 (cit. on p. 52).

[122] K. Røed et al., “Radiation tolerance of an SRAM based FPGA used in a large tracking detector”, PoS, 043 (2010) (cit. on p. 52).

[123] A. Lesea et al., “The Rosetta experiment: atmospheric soft error rate testing in differing technology FPGAs”, IEEE Transactions on Device and Materials Reliability **5**, 317–328 (2005) (cit. on pp. 52, 55).

[124] P. J. Ashenden, *The Designer’s Guide to VHDL, Third Edition (Systems on Silicon)* (Elsevier Inc., 2008) (cit. on p. 52).

[125] “IEEE Standard for Verilog Hardware Description Language”, IEEE Std 1364-2005 (Revision of IEEE Std 1364-2001), 1–590 (2006) (cit. on p. 52).

[126] B. Todd and S. Uznanski, “Radiation Risks and Mitigation in Electronic Systems”, in CAS - CERN Accelerator School: Power Converters (2015) (cit. on p. 54).

[127] M. J. Wirthlin, “FPGAs operating in a radiation environment: Lessons learned from FPGAs in space”, JINST **8**, C02020 (2013) (cit. on p. 54).

[128] A. Lelis et al., "Radiation response of advanced commercial SRAMs", IEEE Transactions on Nuclear Science **43**, 3103–3108 (1996) (cit. on p. 55).

[129] C. Poivey et al., "Radiation Characterisation of Commercially Available 1Mbit/4Mbit SRAMs for Space Applications", in 1998 nsrec data workshop (1998), p. 68 (cit. on p. 55).

[130] C. R. Julien, B. J. LaMeres, and R. J. Weber, "An FPGA-based radiation tolerant SmallSat Computer System", in 2017 ieee aerospace conference (2017), pp. 1–13 (cit. on p. 55).

[131] X. Yao et al., "The impact of Total Ionizing Dose on unhardened SRAM cell margins", English (US), IEEE Transactions on Nuclear Science **55**, 3280–3287 (2008) (cit. on p. 55).

[132] S. Vartanian, G. Allen, and D. Thorbourn, "SRAM-Based FPGA: High Dose Test Methods Using Evaluation Boards", in 19th european conference on radiation and its effects on components and systems (radecs) (Sept. 2019), pp. 1–6 (cit. on p. 55).

[133] L. Ding et al., "Analysis of TID Failure Modes in SRAM-Based FPGA Under Gamma-Ray and Focused Synchrotron X-Ray Irradiation", IEEE Transactions on Nuclear Science **61**, 1777–1784 (2014) (cit. on p. 55).

[134] L. A. Tambara et al., "Soft error rate in SRAM-based FPGAs under neutron-induced and TID effects", in 2014 15th latin american test workshop - latw (2014), pp. 1–6 (cit. on p. 55).

[135] J. Benfica et al., "Analysis of SRAM-Based FPGA SEU Sensitivity to Combined EMI and TID-Imprinted Effects", IEEE Transactions on Nuclear Science **63**, 1294–1300 (2016) (cit. on p. 55).

[136] M. Bellato et al., "Evaluating the effects of SEUs affecting the configuration memory of an SRAM-based FPGA", in Proceedings design, automation and test in europe conference and exhibition, Vol. 1 (Jan. 2004), pp. 584–589 (cit. on p. 55).

[137] B. H. Pratt, "Analysis and Mitigation of SEU-induced Noise in FPGA-based DSP Systems" (Brigham Young University - Provo, 2011) (cit. on p. 56).

[138] D. S. Lee et al., "Addressing Angular Single-Event Effects in the Estimation of On-Orbit Error Rates", IEEE Transactions on Nuclear Science **62**, 2563–2569 (2015) (cit. on p. 56).

[139] M. Wirthlin et al., "A Method and Case Study on Identifying Physically Adjacent Multiple-Cell Upsets Using 28-nm, Interleaved and SECDED-Protected Arrays", IEEE Transactions on Nuclear Science **61**, 3080–3087 (2014) (cit. on p. 56).

[140] J. R. Azambuja, F. Kastensmidt, and J. Becker, *Fault Tolerance Techniques for Processors* (Springer International Publishing, Cham, 2014), pp. 19–28 (cit. on p. 57).

[141] A. Johnston, "The influence of VLSI technology evolution on radiation-induced latchup in space systems", IEEE Transactions on Nuclear Science **43**, 505–521 (1996) (cit. on pp. 57, 58).

[142] N. J. Pieper et al., “Single-Event Latchup Vulnerability at the 7-nm FinFET Node”, in 2022 ieee international reliability physics symposium (irps) (2022), 5C.2–1–5C.2–6 (cit. on p. 57).

[143] H. Becker, T. Miyahira, and A. Johnston, “Latent damage in CMOS devices from single-event latchup”, IEEE Transactions on Nuclear Science **49**, 3009–3015 (2002) (cit. on p. 57).

[144] X. Wang et al., “Design and testing of an stgc asic interface board for the atlas new small wheel upgrade”, Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **1028**, 166326 (2022) (cit. on p. 58).

[145] H. Quinn et al., “Domain crossing errors: Limitations on single device triple-modular redundancy circuits in Xilinx FPGAs”, eng, IEEE Transactions on Nuclear Science **54**, 7 (2007) (cit. on p. 59).

[146] A. Ahmed, “New FPGA blind Scrubbing Technique”, in 2016 ieee aerospace conference (2016), pp. 1–9 (cit. on p. 61).

[147] J. Heiner et al., “FPGA Partial Reconfiguration via Configuration Scrubbing”, 99–104 (2009) (cit. on p. 61).

[148] U. Legat, A. Biasizzo, and F. Novak, “SEU Recovery Mechanism for SRAM-Based FPGAs”, IEEE Transactions on Nuclear Science **59**, 2562–2571 (2012) (cit. on p. 61).

[149] M. Moraitis and E. Dubrova, “Interconnect-Aware Bitstream Modification”, IACR Cryptol. ePrint Arch. (2020) (cit. on p. 62).

[150] M. R. Naeemi Khaledi, M. Ebrahimi, and Z. Navabi, “Compensating Detection Latency of FPGA Scrubbers with a Collaborative Functional Hardware Duplication”, in 2021 ieee microelectronics design & test symposium (mdts) (2021), pp. 1–6 (cit. on p. 62).

[151] *Vivado Design Suite 7 Series FPGA and Zynq-7000 SoC*, UG953, Libraries Guide, Xilinx Inc. (Oct. 2023) (cit. on pp. 62, 63).

[152] I. Xilinx, *Soft Error Mitigation Controller*, PG036, Product Guide, Xilinx (Apr. 2017) (cit. on p. 63).

[153] IQD, *Frequency Products LFSPXO018042*, Accessed on Feb 7, 2024 (2016) (cit. on p. 66).

[154] I. Xilinx, *7 Series FPGAs Clocking Resources*, UG472, User Guide (2018), p. 114 (cit. on p. 66).

[155] I. Xilinx, *Clocking Wizard LogiCORE IP*, Product Guide, PG065 (2022) (cit. on p. 66).

[156] *ModelSim® User’s Manual*, Software Version 10.1c, Mentor Graphics Corporation (2012) (cit. on p. 67).

[157] M. Mano, *Digital Logic & Computer Design* (Pearson Education, 1979) (cit. on p. 67).

[158] S. Kulis, “Single Event Effects mitigation with TMRG tool”, Journal of Instrumentation **12**, C01082 (2017) (cit. on p. 69).

[159] L. Schreiter, “Konzepte zur Steigerung der Resilienz gegenüber strahleninduzierten Logikfehlern des MOPS-HUB FPGA Entwurfs im Kontrollsystem des ATLAS Pixeldetektors”, de, Bachelor Thesis (), p. 80 (cit. on p. 69).

[160] M. Karagounis, “Design eines CAN Controllers mit VHDL und SpecCharts” (Fachhochschule Köln, 2000) (cit. on pp. 71, 84, 182).

[161] *7 Series FPGAs and Zynq-7000 SoC XADC Dual 12-Bit 1 MSPS Analog-to-Digital Converter*, UG480, User Guide, Xilinx, Inc. (June 2022) (cit. on pp. 77, 127).

[162] S. Kulis, “Single Event Effects mitigation with TMRG tool”, Journal of Instrumentation **12**, C01082 (2017) (cit. on p. 91).

[163] I. Xilinx, *7 Series FPGAs Configuration*, UG470, Reference Guide (Nov. 2017) (cit. on pp. 92, 184, 185).

[164] B. Li and K. K. Gakhal, *MultiBoot with 7 Series FPGAs and SPI*, version v1.1 (Feb. 2017) (cit. on pp. 92, 184).

[165] I. Integrated Silicon Solution, *IS25LP032/064/128 Serial Flash Memory*, Rev. D, Integrated Silicon Solution, Inc. (2014) (cit. on p. 92).

[166] *Radiation Test Report, IS25LP128-JBLE*, tech. rep. EDMS Document No. 2005861 (July 2018) (cit. on p. 92).

[167] I. Xilinx, *UltraScale Architecture SelectIO Resources*, UG571, User Guide (Aug. 2023) (cit. on pp. 94, 96).

[168] A. Qamesh and R. Ahmad and T. Fischer and M. Karagounis and P. Kind and S. Kersten and T. Krawutschke and L. Schreiter and C. Zeitnitz and on behalf of the ATLAS ITK collaboration, *Specifications of the MOPS-HUB FPGA*, tech. rep. AT2-IP-ES-0025, Accessed on Jan 7, 2024 (2022) (cit. on p. 96).

[169] *Tri-Mode Ethernet MAC*, PG051, Accessed on Jan 7, 2024, Xilinx, Inc. (Nov. 2023) (cit. on p. 96).

[170] *1G/2.5G Ethernet PCS/PMA or SGMII v16.0*, PG047, Accessed on Jan 7, 2024, Xilinx, Inc. (Apr. 2017) (cit. on p. 96).

[171] C. Ghabrous Larrea et al., “IPbus: a flexible Ethernet-based control system for xTCA hardware”, JINST **10**, C02019 (2015) (cit. on pp. 96, 99).

[172] R. P. GmbH, *Econoline DC/DC Converter*, Accessed on Nov 7, 2023 (2019) (cit. on pp. 106, 108).

[173] I. Digilent, *PmodUSBUART™ Reference Manual*, Accessed on Nov 17, 2023 (2016) (cit. on p. 114).

[174] D. Pfeiffer et al., “The radiation field in the Gamma Irradiation Facility GIF++ at CERN”, Nuclear Instrument **866**, 91–103 (2017) (cit. on p. 117).

[175] A. Devices, *LTM4619 Series Data Sheet*, Accessed on Nov 7, 2023 (2005) (cit. on p. 118).

[176] M. Leyer, "Development of a multifunctional FPGA-based hardware tester PCB for the ATLAS ITK particle experiment used to monitor a TID campaign at the GIF++ facility at CERN" (Institute of Computer and Communication Technology, Sept. 2024) (cit. on pp. 121, 122).

[177] Xilinx, *Artix-7 FPGAs Data Sheet: DC and AC Switching Characteristics*, https://www.xilinx.com/support/documentation/data_sheets/ds181_Artix_7_Data_Sheet.pdf (visited on 02/22/2022) (cit. on pp. 121, 122).

[178] V. e. a. Radulovic, "Large Object Irradiation Facility In The Tangential Channel Of The JSI TRIGA Reactor", Advanced European Infrastructures for Detectors at Accelerators (2017) (cit. on p. 122).

[179] A. Kolsek et al., "Using TRIGA Mark II Research Reactor for Irradiation with Thermal Neutrons", Nuclear Engineering and Design **283**, SI:NENE 2013, 155–161 (2015) (cit. on p. 122).

[180] B. Smodiš and L. Snoj, "Utilization and Application of the Slovenian TRIGA Reactor", in International conference on research reactors: safe management and effective utilization (International Atomic Energy Agency (IAEA), Nov. 14–18, 2011) (cit. on p. 122).

[181] *Artix-7 FPGAs Data Sheet: DC and AC Switching Characteristics*, DS181 (v1.25), Xilinx, Inc. (June 2018) (cit. on pp. 125, 129, 179).

[182] H. U. Hospital, *Heidelberg Ion Beam Therapy Center (HIT)*, Accessed on 2024-01-01, https://www.klinikum.uni-heidelberg.de/interdisziplinaere-zentren/heidelberger-ionenstrahl-therapiezentrum-hit (cit. on p. 125).

[183] Digilent, *ARTY Reference*, Accessed on Sep 1, 2023, 2017 (cit. on p. 126).

[184] E. e. a. Fuller, "Radiation test results of Virtex FPGA and ZBT SRAM for space-based reconfigurable computing", in (Sept. 1999) (cit. on p. 130).

[185] K. Røed, "Single Event Upsets in SRAM FPGA based readout electronics for the Time Projection Chamber in the ALICE experiment" (Bergen University, 2009) (cit. on p. 130).

[186] C. Detemmerman, "Experimental Study on SEU Mitigation in a Commercial-Off-the-Shelf SRAM FPGA for the LHC Phase-2 Upgrade" (Université Libre de Bruxelles (ULB), 2019) (cit. on p. 130).

[187] S. Fiore et al., "Radiation and magnetic field effects on commercial DC-DC converters for HL-LHC experiments", in 2014 IEEE Nuclear Science Symposium and Medical Imaging Conference (Mar. 2016) (cit. on p. 132).

[188] T. Sjöstrand, S. Mrenna, and P. Skands, "A brief introduction to PYTHIA 8.1", Computer Physics Communications **178**, 852–867 (2008) (cit. on pp. 166, 167).

[189] The ATLAS Collaboration, *Magnetic Field*, Accessed on Sep 1, 2023, (2014) http://atlas.web.cern.ch/Atlas/GROUPS/MUON/magfield/mag_page1.html (cit. on p. 167).

[190] I. Xilinx, *Vivado Design Suite Properties*, UG912, Reference Guide (Nov. 2023) (cit. on p. 184).

Acronyms

ACK Acknowledgement. 27, 28

ADC Analog Digital Converter. 12, 13, 28, 35, 36, 65, 74–76, 85, 110, 112, 113, 124, 159, 183

ALICE A Large Ion Collider Experiment. 3

API Application Programming Interface. 99

ASIC Application Specific Integrated Circuit. 1, 9, 13–15, 51

ATLAS A Toroidal LHC Apparatus. 1, 3–7, 9–12, 14, 15, 19, 20, 23, 24, 49, 157, 161, 166, 167

BE Back-End. 9

BIS Beam Interface System. 12

BRAM Block RAM. 51, 54, 55, 60, 62, 63, 69

BUFG Global Clock Buffer. 69

CANH CAN High. 22, 23, 25, 31–33, 106–109, 113, 157, 160–162, 172, 173

CANL CAN Low. 22, 23, 25, 31–33, 106–109, 113, 157, 160–162, 172, 173

CLB Configurable Logic Block. 51, 53, 55

CLPS CERN Low Power Signaling. 29

CMOS Complementary metal-oxide-semiconductor. 15, 32, 55, 58, 158

CMS Compact Muon Solenoid. 3, 4

CRAM Configuration Memory. 37, 51, 54, 55, 58, 60–63, 91, 92, 125–129, 132, 161, 163, 184–186

CRC Cyclic Redundancy Check. 27, 62–64, 91, 92, 184, 185

CSR Channel Setup Register. 76, 85

DAQ Data Acquisition. 14

DC Direct Current. 29

DD Displacement Damage. 42, 44

DDR Double Data-Rate. 51, 60

DIP Dual In – Line Package. 35, 114, 171

DLC Data Length Code. 27

DSP Digital Signal Processing. 51

DSS Detector Safety System. 12

DUT Device Under Test. 54, 67, 68, 103, 104, 110, 126, 127

ECAL Electromagnetic Calorimeter. 6

ECC Error-Correction Code. 61, 62, 64, 91, 184

eLink Electrical Link. 16, 23, 24, 29, 36–38, 65–68, 74–87, 89, 93–98, 100, 131, 133, 159, 164, 183

ELMB Embedded Local Monitor Board. 14

EMI Electromagnetic Interference. 22

EOF End-Of-Frame. 28, 74, 77–82

EOS End Of Structure. 15, 19

FE Front-End. 6, 8, 9, 13–15, 20, 37

FELIX Front-End Link eXchange. 14

FF Flip-Flop. 51, 53–55, 57, 69, 70, 158

FIFO First In First Out. 79–81, 87, 89, 90, 97, 100, 183, 186

FSM Finite State Machine. 60, 66, 68, 71, 73, 75–78, 81–90, 94, 95, 101, 115, 158, 159

GIF++ Gamma Irradiation Facility. 117, 119

GMII Gigabit Media-Independent Interface. 95

GTH Gigabit Transceiver-High Speed. 95, 96

HCAL Hadronic Calorimeter. 6

HDL Hardware Description Language. 52, 53, 67

HEH High Energy Hadrons. 50, 129, 132

HIT Heidelberg Ion Beam Therapy Center. 125–127, 129, 131, 161, 164

HV High Voltage. 9, 13

IC Integrated Circuit. 47, 49, 50, 57

ICAP Internal Configuration Access Port. 60, 63, 91, 133

ID Inner Detector. 6, 7

IDE Identifier Extension. 27

IFS Inter Frame Spacing. 28

ILK-FPGA Interlock FPGA. 12

IP Intellectual Property. 66, 67, 71, 76, 94, 124, 126

IPBus IP-based protocol. 94–97, 99, 100, 164, 186

IPROG internal PROGRAM-B. 92

ISERDESE2 Input Serial-to-Parallel Logic Resources. 67, 81

ISK Internal Synchronization Key. 79, 80, 83

ITk Inner Tracker. 1, 6, 7, 9, 10, 12, 14, 15, 19–22, 31, 32, 131, 157, 163

JTAG Joint Test Action Group. 54, 60, 92, 126

LAN Local Area Network. 38

LANSCE Los Alamos Neutron Science Center. 36, 37

LDO Low Dropout Linear Regulator. 32, 34, 35, 106

LHCb Large Hadron Collider Beauty. 3

LISSY Local Interlock & Safety System. 12

lpGBT Low-power GigaBit Transceiver. 37

LSB Least Significant Bit. 81, 181

LUT Look-Up Table. 51, 53, 55–57, 69, 158

MAC Media Access Control. 95, 96

MBU Multi Bit Upset. 56, 61–63, 92, 127, 184

MCU Multi Cell Upset. 56, 158

MIP Minimum Ionizing Particle. 39

MISO Master In Slave Out. 74

MMCM Mixed-Mode Clock Manager. 66, 69

MOS Metall-Oxid-Semiconductor. 32, 42, 43, 158

MOSFET Metal-Oxide-Semiconductor Field-Effect Transistor. 43, 46, 55, 158

MOSI Master Out Slave In. 74

MSB Most Significant Bit. 181

NCSR National Centre of Scientific Research. 36, 37

NIEL Non-Ionizing Energy Loss. 44, 122, 131

NMOS N-channel metal–oxide–semiconductor. 32, 33

NPN Negative-Positive-Negative. 32

NTC Negative Temperature Coefficient Thermistor. 12, 13, 112, 162, 180

OA Open Architecture. 9, 10

OB Outer Barrel. 22

OPC-UA Open Platform Communications Unified Architecture. 9, 10

OSERDESE2 Output Parallel-to-Serial Logic Resources. 67, 81

OSI Open Systems Interconnection model. 25

PHY Physical Layer. 96

PL Programmable Logic. 96

PLL Phase-Locked Loop. 66

PMOS p-channel metal–oxide–semiconductor. 32, 33

PP Patch Panel. 19–21

PSU Power Supply Unit. 110, 120, 121, 124, 174–178

RD Running Disparity. 29, 30, 80, 81

RTL Register Transfer Level. 52–54, 69, 70

RTR Remote Transmission Request. 27

SBU Single Bit Upset. 56, 62, 63, 184

SCADA Supervisory Control And Data Acquisition. 9, 10

SCR Silicon Controlled Rectifier. 57

SCT Semiconductor Tracker. 6

SDO Service Data Objects. 28, 82, 100, 157

SEB Single Event Burnout. 46

SECDED Single Error Correction and Double Error Detection. 61, 62

SEFI Single Event Functional Interrupt. 47

SEGR Single Event Gate Rupture. 46

SEM IP Soft Error Mitigation IP. 63, 64, 91, 92, 126–128, 161, 184

SER Soft Error Rate. 37, 163

SF Safety Factor. 23, 24

SFP Small Form-factor Pluggable. 96

SGMII Serial Gigabit Media Independent Interface. 96

SI International System of Units. 42

SJW Synchronization Jump Width. 26

SOF Start-Of-Frame. 27, 74, 77–82, 183

SP Serial Power. 1, 7, 8, 12, 13, 15, 22

SRAM Static RAM. 39, 51, 52, 54–56, 58, 158

SSTL Stub Series Terminated Logic. 81

SYNC Synchronization Symbol. 74, 77, 78, 80–82

TCL Tool Command Language. 69

TCP/IP Transmission Control Protocol/Internet Protocol. 94, 99, 101

TDR Technical Design Report. 8

TMR Triple Modular Redundancy. 59, 60, 69, 70, 91, 92, 158, 163

TMRG Triple Modular Redundancy Generator. 69, 70, 91

TQ Time Quanta. 26

TRIGA Training, Research, Isotopes, General Atomics. 122

TRT Transition Radiation Tracker. 6

UART universal asynchronous receiver / transmitter protocol. 68, 89, 90, 95, 101, 114–116, 159, 161

uHAL Hardware Access Library. 99, 112

UML Unified Modeling Language. 98, 159

VCAN Power lines for individual CAN bus. 21–23, 31, 34–36, 65, 74, 75, 77, 85, 106–111, 113–115, 124, 160–162, 171–178

VCAN-PSU CAN power supplied by the main power supply. 16, 21–23, 30, 33, 77, 111

VCCAUX Auxiliary supply voltage. 121, 122, 124, 126

VCCBRAM Supply voltage for the block RAM. 121, 122, 124, 126

VCCINT Internal supply voltage. 121, 122, 124, 126

VL Versatile Link. 38

VPP3 Power lines for the FPGA of MOPS-HUB. 16, 21–23, 30, 31, 77, 104, 105, 109, 111, 118, 123

VTRx+ Versatile Link+ Transceiver. 37

WBSTAR Warm Boot Start Address. 92, 184

XML Extensible Markup Language. 99

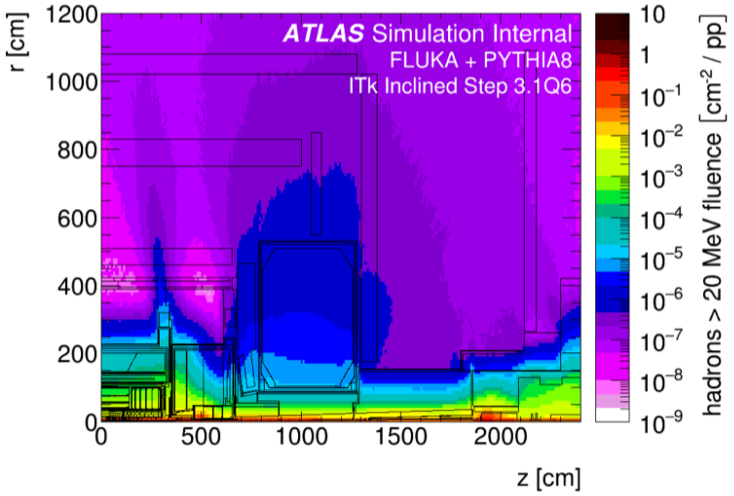
List of Figures

List of Tables

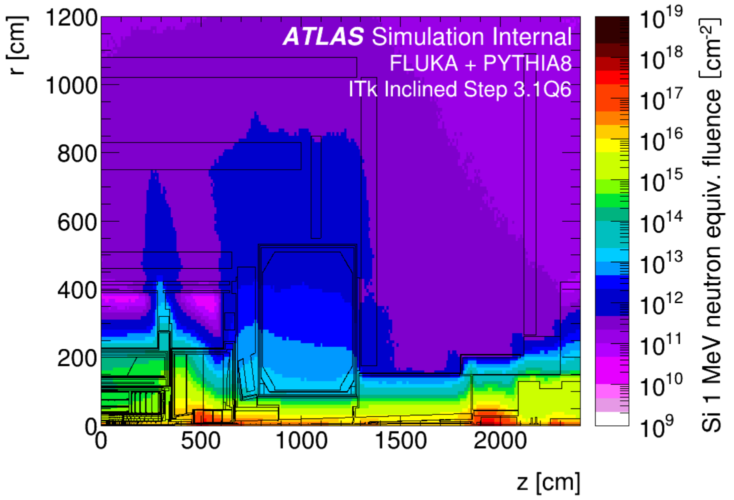
Appendix A

Requirements in the ATLAS Hall

A.1 Radiation level in the ATLAS Hall



(a) Simulation of the Hadron fluence (> 20 MeV).



(b) Simulation of the 1 MeV Neutron equivalent fluence ($\Phi_{\mathrm{eq}}^{\mathrm{Si}}$).

Figure A.1: Expected Hadron fluence (a) and Neutron fluence (b) using FLUKA [104] and PYTHIA8 [188] in the ATLAS hall. The 1 MeV fluence is normalized to 4000 fb^{-1}. No safety factors have been applied [58]. The radius r is measured from the beam line, while z is measured from the interaction point.

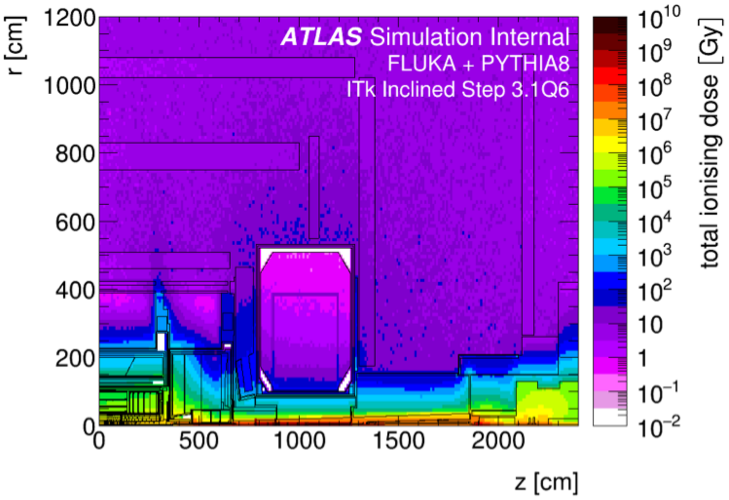


Figure A.2: Simulation results of the TID using FLUKA [104] and PYTHIA8 [188] in the ATLAS hall. No safety factors have been applied [58].

A.2 Magnetic Field in the ATLAS Hall

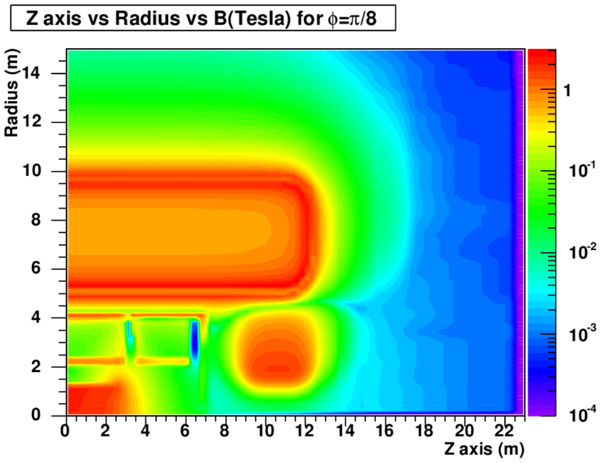


Figure A.3: Expected magnetic field strength in the ATLAS hall [189].

Appendix B

Hardware

B.1 PP3-Power Module Testing

B.1.1 DC/DC Measurements

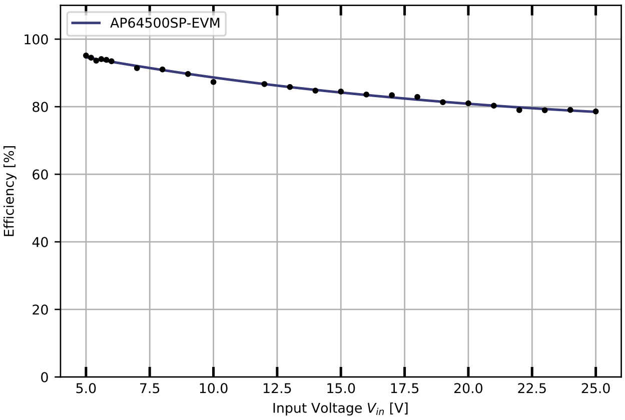


Figure B.1: Efficiency measurements for the DC/DC converter on the PP3-Power module

B.1.2 Magnetic Field Test

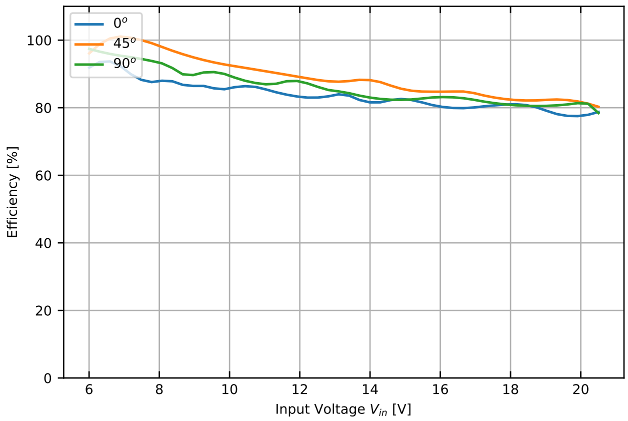


Figure B.2: Efficiency measurements for the PP3-Power module (ID = V2.4) under several angles at the magnetic field described in Figure 8.11.

B.2 CIC Testing

B.2.1 DC/DC Measurements

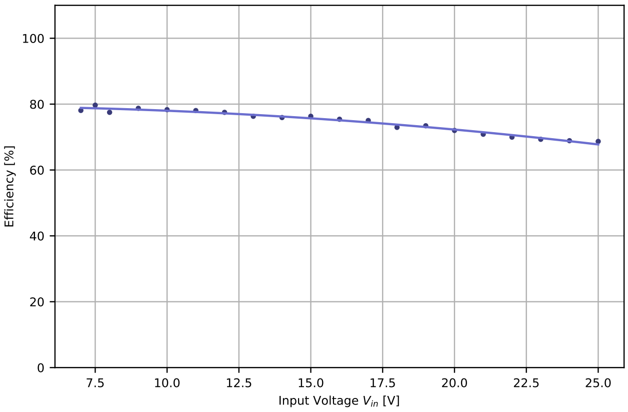


Figure B.3: Efficiency measurements for the isolated DC/DC converter on the CIC.

B.2.2 Powering Measurements for Communication Channels

Table B.1 shows the truth table of different configurations in the CIC and the corresponding measured output voltage VCAN. $\text{VCAN}_{\text{Theo}}$ in the table is the theoretical value according to Equation 4.2, while $\text{VCANA}_{\text{Exp}}$ and $\text{VCANB}_{\text{Exp}}$ are the measured values based on the test described in Section 8.3.2 after setting the configuration.

Table B.1: Powering test results for different communication channels on the CIC (VCANA and VCANB) under different configurations.

Dual bit patterns				**Value**	**Status**	R_{set}	$\textbf{VCAN}_{\textbf{Theo}}$	$\textbf{VCANA}_{\textbf{Exp}}$	$\textbf{VCANB}_{\textbf{Exp}}$
A	B	C	P	[Hex]	[ON/OFF]	[kΩ]	[V]	[V]	[V]
bit[7-6]	bit[5-4]	bit[3-2]	bit[1-0]						
0	0	0	0	0x00	OFF	-	0	0	0
0	0	0	1	0x03	ON (DIP OFF)	∞	1.25	1.25	1.25
0	0	0	1	0x03	ON	90.9	1.90	1.89	1.89
0	0	1	1	0x0F	ON	78.7	2.00	1.99	1.99
0	1	0	1	0x33	ON	51.1	2.40	2.39	2.39
0	1	1	1	0x3F	ON	44.2	2.58	2.57	2.57
1	0	0	1	0xC3	ON	38.3	2.80	2.77	2.77
1	0	1	1	0xCF	ON	26.1	3.50	3.48	3.48
1	1	0	1	0xF3	ON	19.6	4.25	4.22	4.22
1	1	1	1	0xFF	ON	16.9	4.60	4.64	4.56

B.2.3 Noise measurements

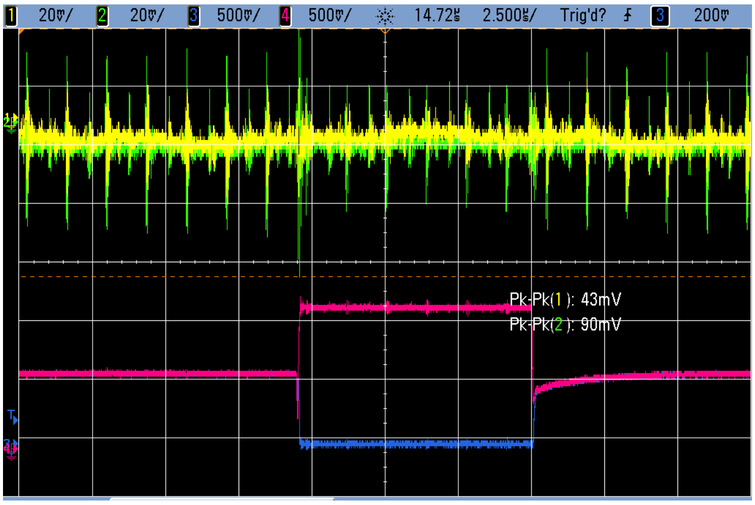
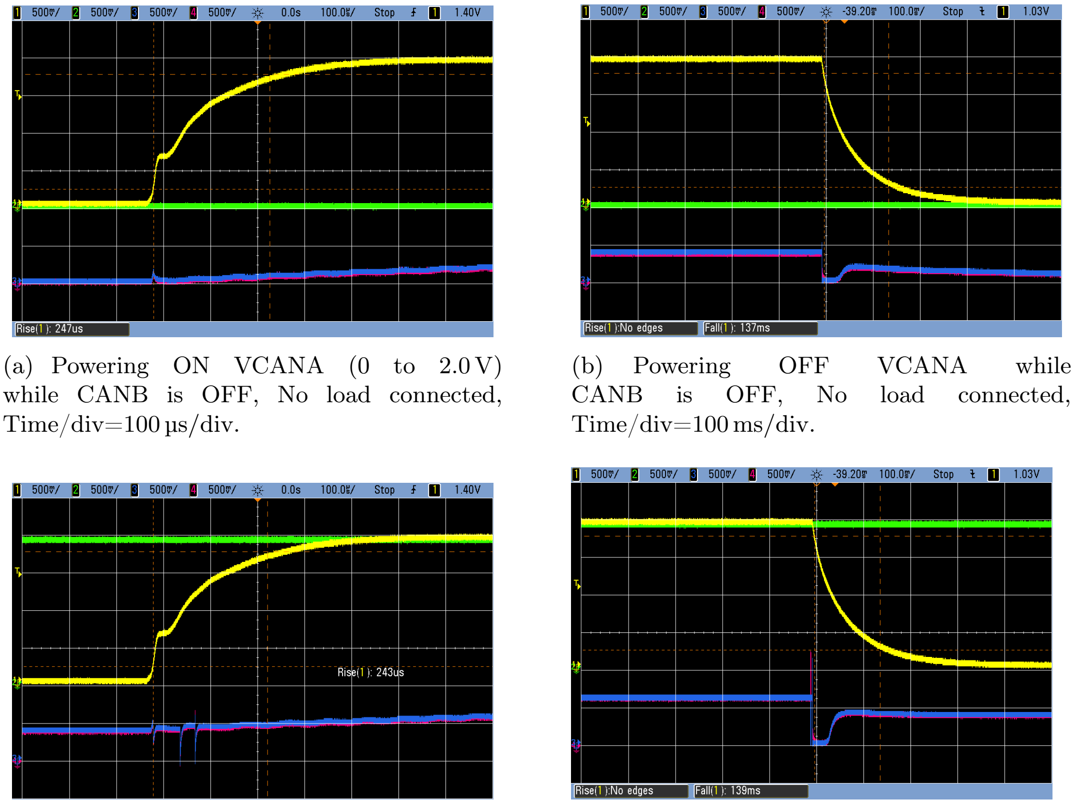


Figure B.4: Noise measurements for VCAN channels during CAN transitions. The channels are defined as follows, CH1:VCANA, CH2:VCANB, CH3:CANH(A/B) and CH4:CANL(A/B). Oscilloscope settings:(All the channels are AC coupled), (Time/div=2.5 µs/div), (Bandwidth limit=20 MHz), (V/div(CH1&CH2) =20 mV/div) and (V/div(CH3&CH4) =500 mV/div).

B.2.4 Cross Talk Check



(a) Powering ON VCANA (0 to 2.0 V) while CANB is OFF, No load connected, Time/div=100 µs/div.

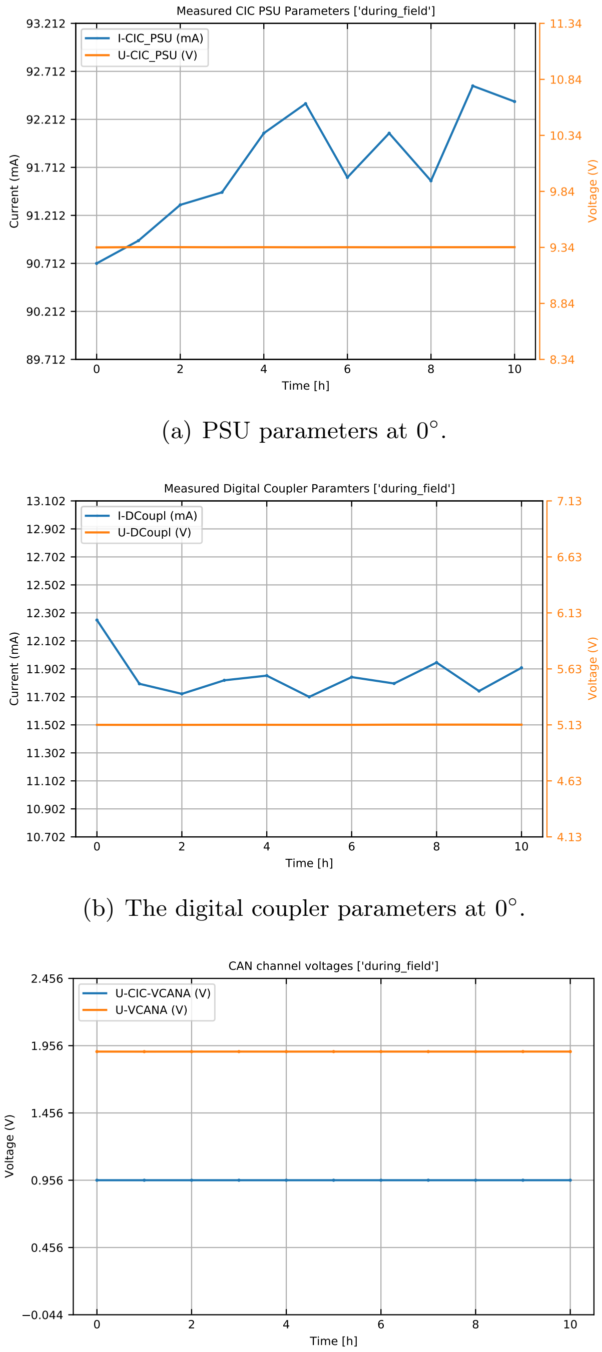
(b) Powering OFF VCANA while CANB is OFF, No load connected, Time/div=100 ms/div.

(c) Powering ON VCANA (0 to 2.0 V) while CANB is ON, No load connected, Time/div=100 µs/div.

(d) Powering OFF VCANA while CANB is ON, No load connected, Time/div=100 ms/div.

Figure B.5: Cross-talk assessment for VCANA and VCANB. CANA and CANB. The channels are defined as follows: CH1: VCANA, CH2: VCANB, CH3: CANH(A/B), and CH4: CANL(A/B). Oscilloscope settings: (All the channels are DC coupled), (V/div=500 mV/div), (Bandwidth limit=20 MHz).

B.2.5 Magnetic Field Test

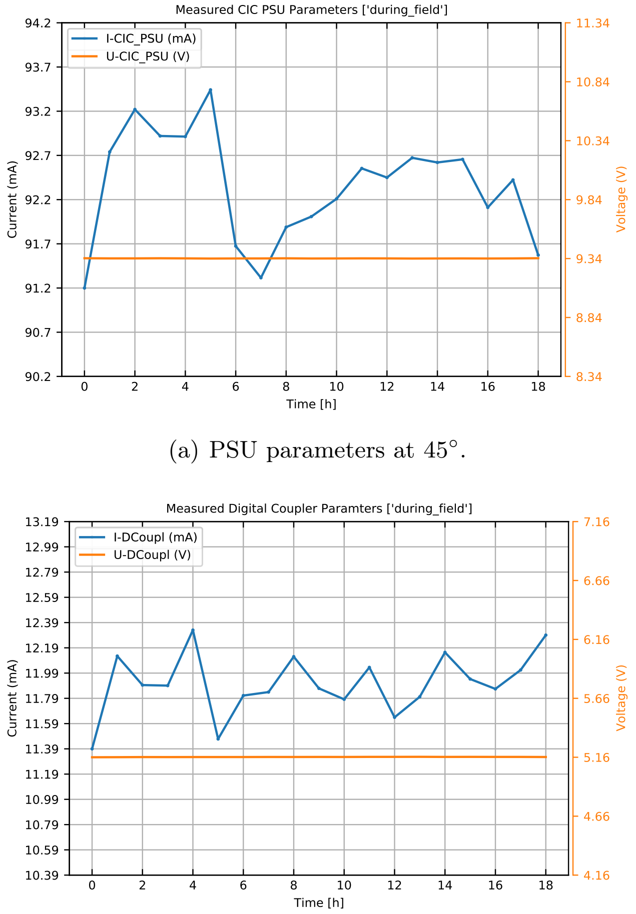


(a) PSU parameters at 0°.

(b) The digital coupler parameters at 0°.

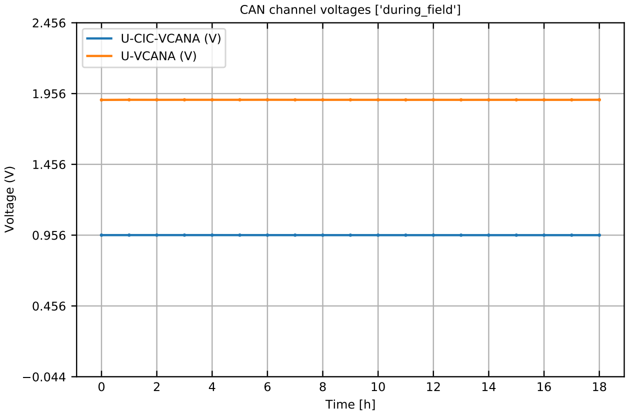
(c) The VCAN at 0°.

Figure B.6: Test results for CIC (ID =V4.2) under magnetic field at 0°.



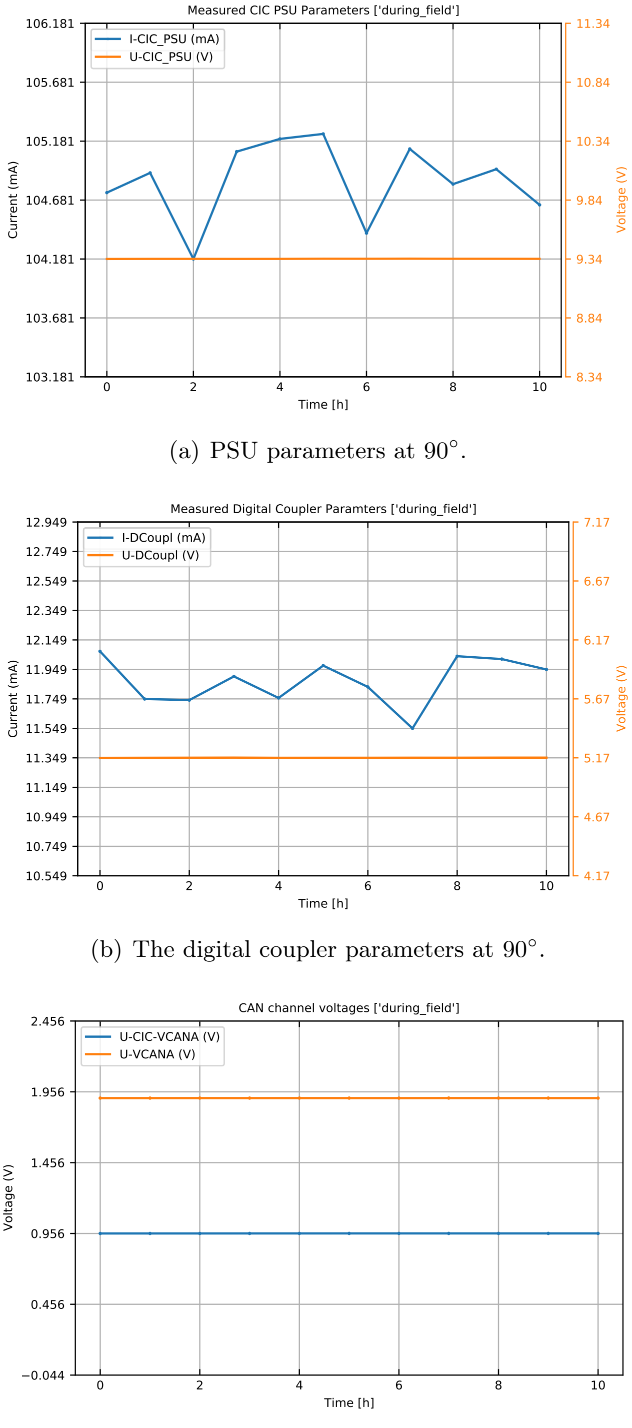
(a) PSU parameters at 45°.

(b) The digital coupler parameters at 45°.



(c) The VCAN at 45°.

Figure B.7: Test results for CIC (ID =V4.2) under magnetic field at 45°.



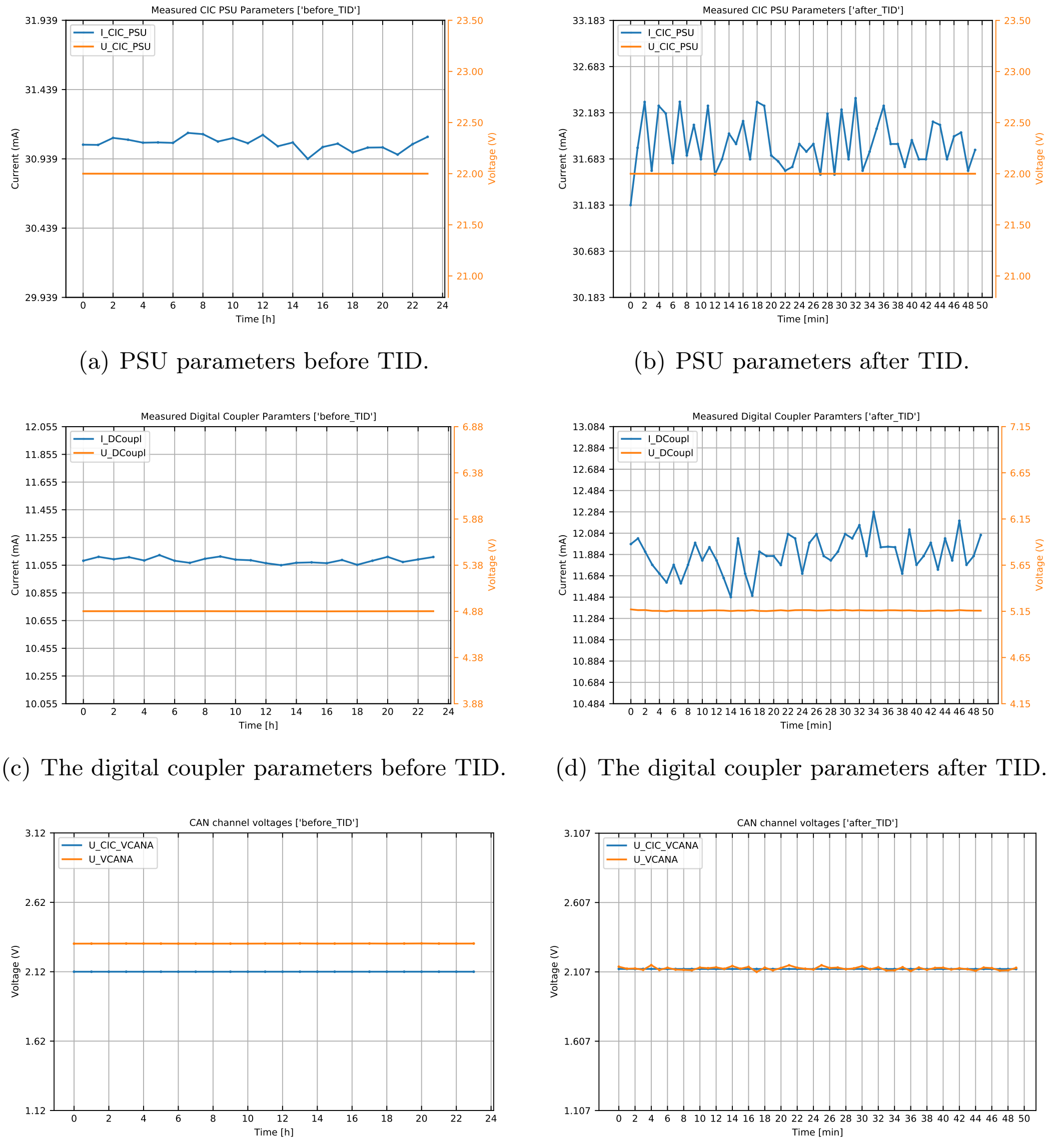
(a) PSU parameters at 90°.

(b) The digital coupler parameters at 90°.

(c) The VCAN at 90°.

Figure B.8: Test results for CIC (ID =V4.2) under magnetic field at 90°.

B.2.6 TID Irradiation Test



(a) PSU parameters before TID.

(b) PSU parameters after TID.

(c) The digital coupler parameters before TID.

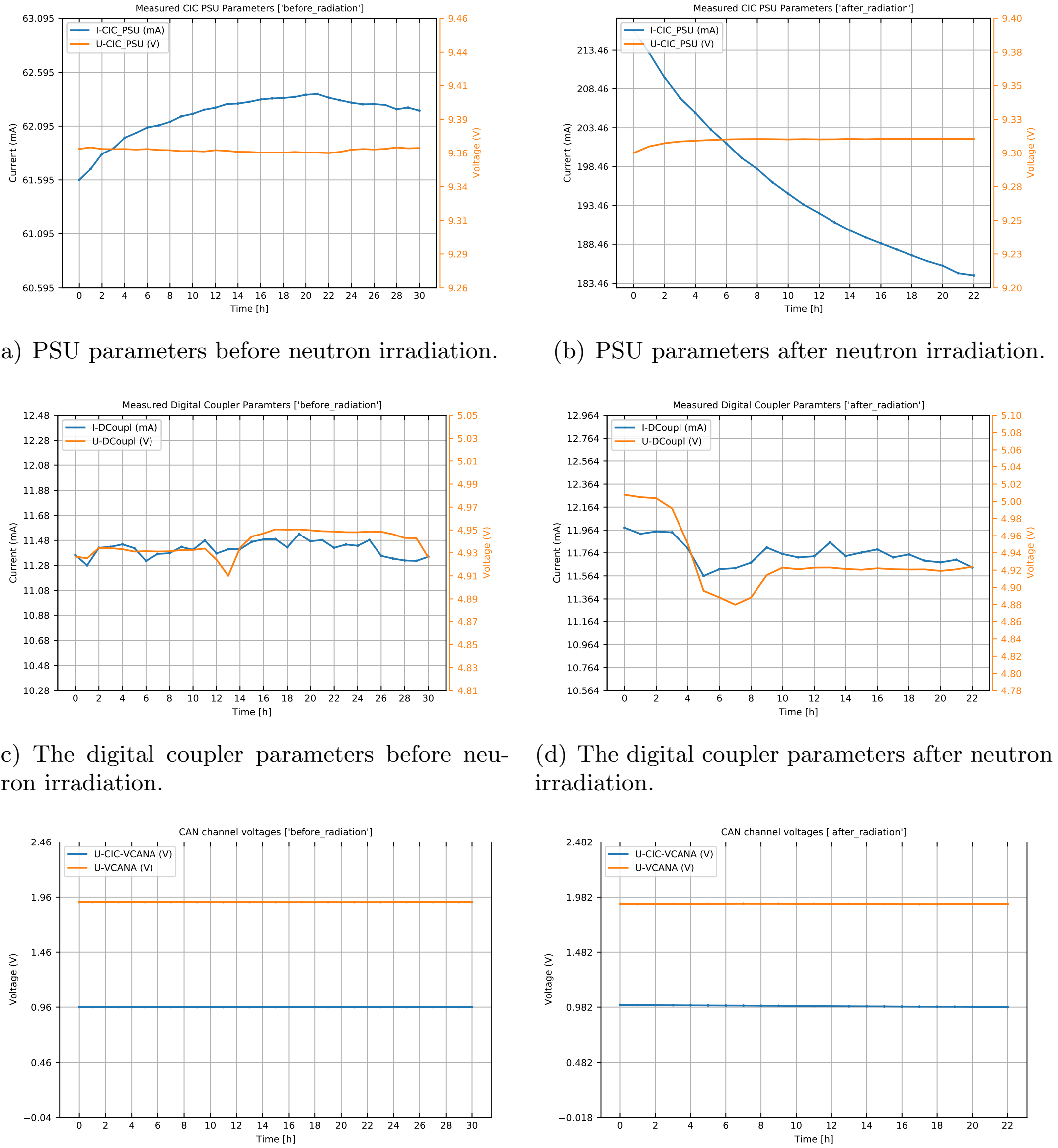
(d) The digital coupler parameters after TID.

(e) The VCAN before TID.

(f) The VCAN after TID.

Figure B.9: Long term test for the CIC (ID =V2.1) before and after TID.

B.2.7 Neutron Irradiation Test



(a) PSU parameters before neutron irradiation.

(b) PSU parameters after neutron irradiation.

(c) The digital coupler parameters before neutron irradiation.

(d) The digital coupler parameters after neutron irradiation.

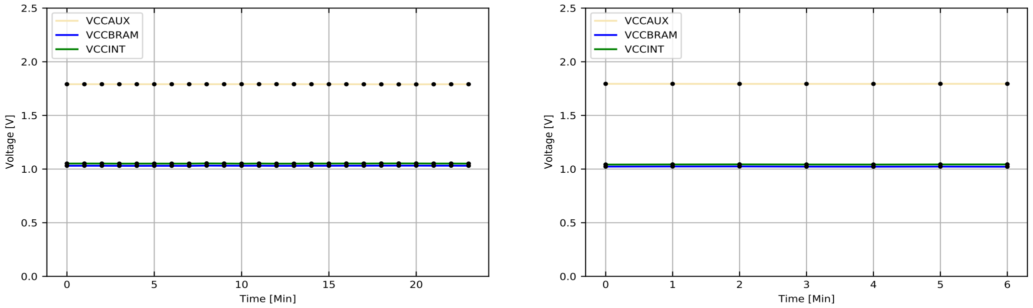
(e) The VCAN before neutron irradiation.

(f) The VCAN after neutron irradiation.

Figure B.10: Long term test for the CIC (ID =V4.1) before and after neutron irradiation.

B.3 PP3 FPGA Testing

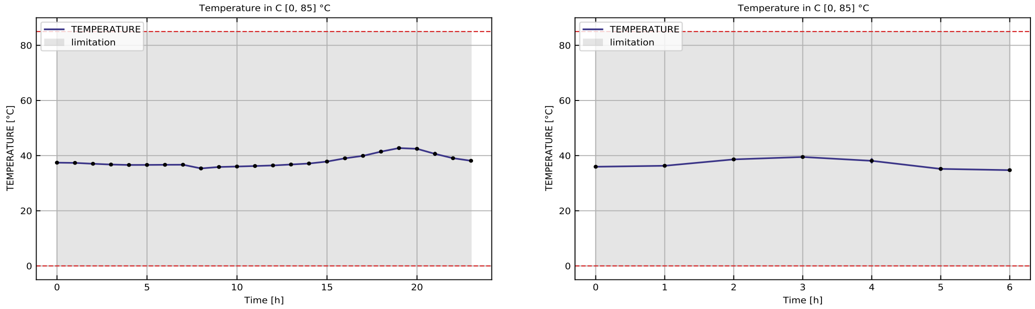
B.3.1 Neutron Irradiation Test



(a) Measured channels before neutron irradiation test.

(b) Measured channels after neutron irradiation test.

Figure B.11: Long term test for the PP3-FPGA module (ID $=V2.1$) before and after neutron irradiation test.

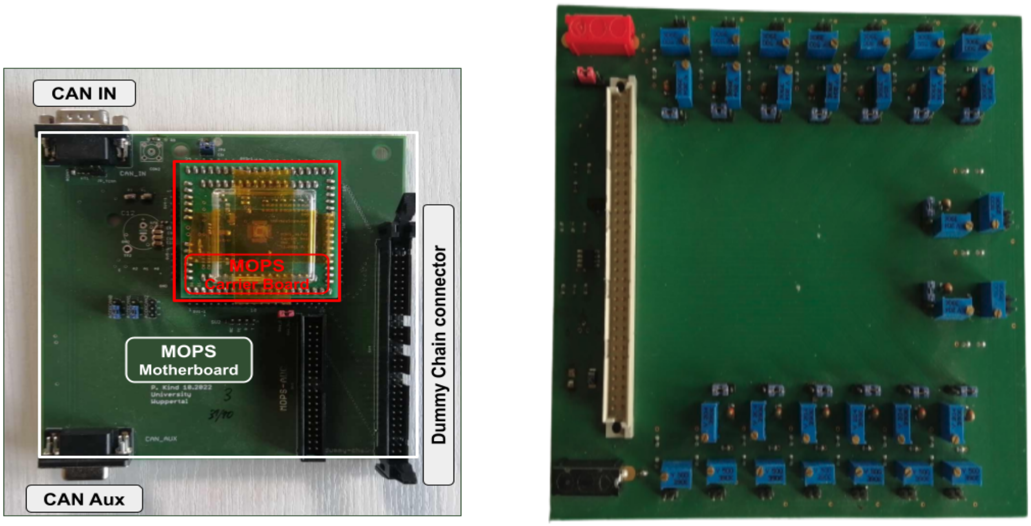


(a) Internal Temperature before neutron irradiation Test.

(b) Internal temperature after neutron irradiation test.

Figure B.12: Internal temperature for the PP3-FPGA module (ID $=V2.1$) before and after neutron irradiation test. The limitation shown are based on the FPGA specifications described in the Artix-7 FPGAs data sheet [181]. Data collected in a room temperature of 20 °C.

B.4 Full Crate Setup



(a) The Motherboard board and Carrier board.

(b) The Dummy chain board.

Figure B.13: Test boards to host the MOPS chip and provide dummy values. The MOPS carrier Board designed to hosts the actual MOPS chip, while the Motherboard board provides the interface to the MOPS carrier board(a). The Dummy chain board mimics the construction of an actual serial power chain using variable resistors to depict detector modules and NTCs (b). Both Figures are not to scale.

Appendix C

Firmware

C.1 Internal Signals in PP3-FPGA

C.1.1 CANakari Signals

CANakari Address Values

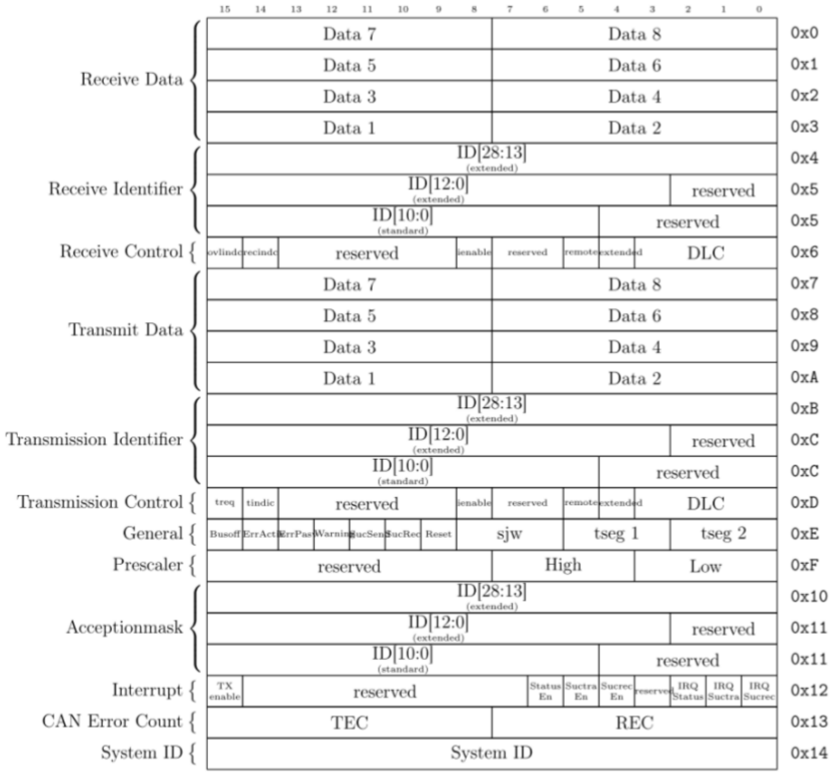
Table C.1 gives a short overview of the register structure for every single register of the **CANakari** module.

Table C.1: Address values of the CANakari registers.

Address Binary	Hex Address Hex	Register Name	Function
10010	0x12	Interrupt	Interrupt Status & Configuration.
10001	0x11	Acceptionmask 1	Receive address mask MSBs 28-13
10000	0x10	Acceptionmask 2	Receive address mask LSBs 12-0
01111	0xF	Prescaler	Pre-scalar configuration.
01110	0xE	General	Sets sjw[1], tseg1[2] and tseg2[3].
01101	0xD	Transmission Control	Transmission control register.
01100	0xC	Transmission Identifier 1	Transmission address MSBs 28-13
01011	0xB	Transmission Identifier 2	Transmission address LSBs 12-0
01010	0xA	Transmission Data 1-2	Transmission register Byte 1-2
01001	0x9	Transmission Data 3-4	Transmission register Byte 3-4
01000	0x8	Transmission Data 5-6	Transmission register Byte 5-6
00111	0x7	Transmission Data 7-8	Transmission register Byte 7-8
00110	0x6	Receive Control	Receive control register.
00101	0x5	Receive Identifier 1	Receive address MSBs 28-13
00100	0x4	Receive Identifier 2	Receive address LSBs 12-0
00011	0x3	Receive Data 1-2	Receive register Byte 1-2
00010	0x2	Receive Data 3-4	Receive register Byte 3-4
00001	0x1	Receive Data 5-6	Receive register Byte 5-6
00000	0x0	Receive Data 7-8	Receive register Byte 7-8

CANakari Registers

Table C.2: Registers of the **CANakari** controller [160].



	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Receive Data	Data 7								Data 8								0x0
	Data 5								Data 6								0x1
	Data 3								Data 4								0x2
	Data 1								Data 2								0x3
Receive Identifier	ID[28:13] (extended)																0x4
	ID[12:0] (extended)													reserved			0x5
	ID[10:0] (standard)											reserved					0x5
Receive Control	ovlindc	recindc	reserved					ienable	reserved		remote	extended	DLC				0x6
Transmit Data	Data 7								Data 8								0x7
	Data 5								Data 6								0x8
	Data 3								Data 4								0x9
	Data 1								Data 2								0xA
Transmission Identifier	ID[28:13] (extended)																0xB
	ID[12:0] (extended)													reserved			0xC
	ID[10:0] (standard)											reserved					0xC
Transmission Control	treq	tindic	reserved					ienable	reserved		remote	extended	DLC				0xD
General	Busoff	ErrActi	ErrPas	Warning	SucSen	SucRec	Reset	sjw			tseg 1			tseg 2			0xE
Prescaler	reserved								High				Low				0xF
Acceptionmask	ID[28:13] (extended)																0x10
	ID[12:0] (extended)													reserved			0x11
	ID[10:0] (standard)											reserved					0x11
Interrupt	TX enable	reserved								Status En	Suctra En	Sucrec En	reserved	IRQ Status	IRQ Suctra	IRQ Sucrec	0x12
CAN Error Count	TEC								REC								0x13
System ID	System ID																0x14

Avalon Interface Ports

Table C.3: **Avalon Interface** ports of the **CANakari** controller.

Name	Type	Bits	Description
cs	input	1	Chip Select. Set to 1 (high active).
write_n	input	1	Write command set to 0 to place data to **writedata** port.
read_n	input	1	Read command set to 0 to register data on **readdata** port.
address	input	5	Register address of the target register.
writedata	input	16	Data to be written to the target register.
readdata	output	16	Requested data of the target register.

C.1.2 CAN FSM Signals

Table C.4: Interrupt signals related to CAN communication.

Signal	Description
start_ write_ can	Write request to a CAN bus.
start_ read_ can	Read request from a CAN bus.
reset_ irq_ can	Reset request for a specific CAN controller.
reset_ irq_ can_ all	Reset request for all CAN controllers.
init_ can	Initialize a specific CAN controller.
start_ osc_ trim	Send trimming messages to a specific CAN bus.

C.1.3 SPI FSM Signals

Table C.5: Interrupt signals related to SPI communication.

Interface	Signal	Description
Control	*start_ power_ init*	Initialize a 8 bit register chip[4] on a specific CIC.
	start_ power_ on	Power on a specific SPI bus.
	start_ power_ off	Power off a specific SPI bus.
	start_ power_ read	Request information from the 8 bit register chip.
Monitor	*start_ mon_ init*	Initialize a 16 bit ADC chip[5] on a specific CIC.
	start_ mon_ config	Send configuration to the ADC[6].
	start_ mon_ read	Request information from the ADC.

C.1.4 eLink FSM Signals

Table C.6: Interrupt signals related to the eLink interface.

Interface	Signal	Description
Receiver	*fifo_ elink_ rdy*	Indicates data availability in the **downstream FIFO**.
	fifo_ data_ sop	Check the SOF word in the data stream.
	can_ id	Check the *CAN bus ID*.
	spi_ mon_ id	Check the *SPI bus ID* for monitoring.
	spi_ power_ off_ id	Check the *SPI bus ID* for control signal (switching off request).
	spi_ power_ on_ id	Check the *SPI bus ID* for control signal (switching on request).
Transmitter	*sign_ in_ signal*	A signal indicating MOPS-Hub Sign-In message.
	start_ write_ elink_ can	Write request to the eLink from the CAN interface.
	start_ write_ elink_ spi	Write request to the eLink from the SPI interface.

C.2 CRAM Protection in MOPS-Hub

The implementation of the CRC and ECC detection and correction technique in MOPS-Hub is covered in Sections C.2.2 and C.2.2.

C.2.1 2-bits Upset

The Readback CRC settings specified in the Vivado constraints File during MOPS-Hub synthesis are listed in C.1.

Code C.1: Constraint for Readback CRC in Vivado

```
#Enables the Post CRC checking
set_property POST_CRC ENABLE [current_design]
#Determines an expected CRC value from the bitstream
set_property POST_CRC_SOURCE PRE_COMPUTED [current_design]
#Even if a CRC error is detected, continue CRC checking.
set_property POST_CRC_ACTION CORRECT_AND_CONTINUE [current_design]
#The INIT_B pin enabled as a source of the CRC error signal.
set_property POST_CRC_INIT_FLAG ENABLE [current_design]
```

The **POST_CRC_ACTION** constraint in Vivado controls the mode of the Readback CRC mechanism [190]. This setting determines the action that the device takes upon detecting a CRC mismatch. The *CORRECT_AND_ CONTINUE* mode is utilized by the scrubbing architectures in the PP3-FPGA to correct SBU within a frame using ECC and then proceed with scanning subsequent frames. Conversely, all other **POST_CRC_ACTION** modes assert the *CRCERROR* pin in the **FRAME_ECCE2** primitive within the SEM IP, regardless of whether the error is a SBU or a MBU) [190]. Given that *ECCERROR* can identify SBUs efficiently, it is desirable to limit *CRCERROR* assertion to the MBUs.

The **POST_CRC_INIT_FLAG** option activates the *INIT_B* pin[7]. This option also flags the occurrence of a CRC error condition when it occurs.

The **POST_CRC_SOURCE**option specifies the source of the CRC value upon detection of a CRC. In this configuration, the CRC value is *PRE_ COMPUTED* from the bitstream CRC.

C.2.2 Multi-bits Upset

Figure C.1 illustrates the flow diagram of the Multi-boot mechanism implemented in Artix 7 family FPGAs [163, 164], which is also integrated into the PP3-FPGA firmware as an advanced level for CRAM protection. When the power is turned on, the golden image initiates a Multi-boot process. Next, using the command **IPROG**, the update bitstream is loaded from the address kept in the **WBSTAR**. After then, the design triggers Fallback feature's booting that ensures a known error free design can be loaded into the device [163]. If an error occurs during the loading the Multi-boot image, the Fallback circuitry will trigger the golden image again to be loaded

[7] *INIT_B* pin indicates when the FPGA is ready to start the configuration process. Pulling down the *INIT_B* signal indicates that the configuration logic fails.

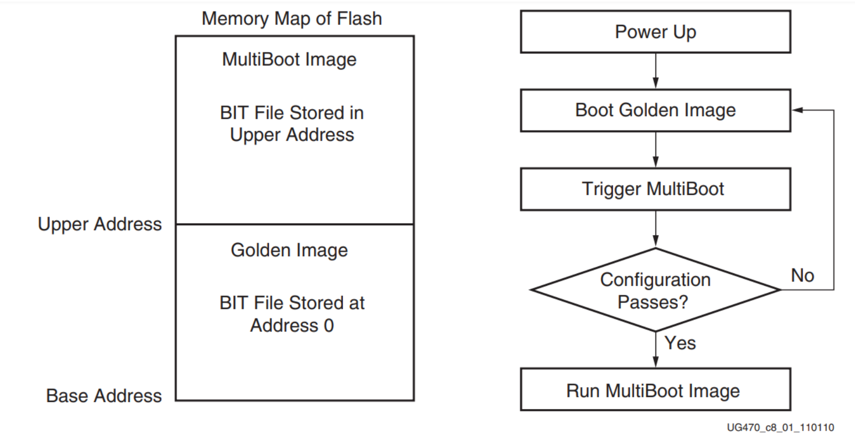


Figure C.1: Initial Golden Image Flow Diagram [163].

from address 0. A **DONE** signal will go high to indicate that the configuration process has finished.

Under various error scenarios, such as a CRC error in the CRAM due to SEU, an **IDCODE** error (where the **DONE** signal remains low), or a watchdog timer timeout (approximately 0.8 s), the FPGA can initiate the Fallback feature for mBAR. This process ensures that an error-free design is loaded into the device by restarting the entire configuration process, as illustrated in Figure C.1.

The Multi-boot settings specified in the Vivado constraints File during MOPS-Hub synthesis are listed in C.2.

Code C.2: Constraint for Multi-boot mechanism in Vivado

```
#Enables the Fallback option
set_property BITSTREAM.CONFIG.CONFIGFallback ENABLE [current_design]
# Allows internal configuration mode
set_property BITSTREAM.CONFIG.PROG_MODE INTERNAL [current_design]
set_property BITSTREAM.CONFIG.NEXT_CONFIG_REBOOT ENABLE [current_design]
# Define the location of Fallback images
set_property BITSTREAM.CONFIG.NEXT_CONFIG_ADDR 0x0049FD5E [current_design]
set_property BITSTREAM.CONFIG.NEXT_CONFIG_ADDR 0x0093FABC [current_design]
# Reduces the size of the configuration file
set_property BITSTREAM.GENERAL.COMPRESS TRUE [current_design]
# Identifies critical configuration bits
set_property BITSTREAM.SEU.ESSENTIALBITS YES [current_design]
```

Activating the **CONFIG.CONFIGFallback** option enables the Fallback option when a configuration attempt fails. The location of the Fallback image is defined by the **CONFIG.NEXT_CONFIG_ADDR** option, which sets the start address for the next configuration image. The **PROG_MODE INTERNAL** option allows the FPGA to use an internal configuration mode for configuration. The bitstream compression **GENERAL.COMPRESS** option is utilized to reduce the size of the configuration file. This option works by writing identical configuration frames once instead of writing each frame individually. While theoretically any alteration in the CRAM could potentially disrupt the functionality of the FPGA, it's important to note that not all configuration bits in the CRAM are equally critical. Xilinx provides an option, **SEU.ESSENTIALBITS**, which facilitates the identification of these critical configuration bits. When an essential bit experiences an upset, it can indeed alter the design circuitry.

C.3 Internal Signals in MOPS-Hub Readout

The Master-to-Slave IPBus Interface

Table C.7: Sub-signals of the IPBus master-to-slave interface.

Interface	**Sub-signal**	**Width**	**Description**
IPBus	*ipb_ addr*	32	The 32 bit wide address bus.
	ipb_ wdata	32	The 32 bit wide write data bus.
	ipb_ strobe	1	Strobe signal to indicate a transfer.
	ipb_ write	1	Write control signal, if 0: read transfer, if 1: write transfer.

C.3.1 The Slave-to-Master IPBus Interface

Table C.8: Sub-signals of the IPBus slave-to-master interface.

Interface	**Sub-signal**	**Width**	**Description**
IPBus	*ipb_ rdata*	32	The 32 bit wide read data bus.
	ipb_ ack	1	Indicates valid data on the read data bus.
	ipb_ err	1	Indicates an error during the read/write transaction.
eLink	*data_ rec_ elink*	96	The 96 bit wide eLink received data register.
	data_ tra_ elink	96	The 96 bit wide eLink transmitted data register.
	start_ write_ elink	1	Write data to the internal FIFO.
	start_ read_ elink	1	Read data from the internal FIFO.
	fifo_ elink_ empty	1	Status port indicates an empty FIFO memory.
	fifo_ elink_flush	1	Interrupt reset signal to empty the FIFO memory.