WARSAW UNIVERSITY OF TECHNOLOGY

FACULTY OF ELECTRONICS
AND INFORMATION TECHNOLOGY

Ph.D. Thesis

Krzysztof Marek Sielewicz, M.Sc.

Mitigation Methods Increasing Radiation Hardness of the FPGA-Based Readout of the ALICE Inner Tracking System

Supervisor
Krzysztof Poźniak, Ph.D., D.Sc.

WARSAW 2018

Acknowledgements

First and foremost, I want to wholeheartedly thank Professor Krzysztof Poźniak, my PhD supervisor at Warsaw University of Technology, for his invaluable support and guidance, as well as his never-ending optimism.

Secondly, I am grateful to Dr. Gianluca Aglieri Rinella, my supervisor at CERN, for all the help, friendly advice and many fruitful discussions about this thesis. I would also like to thank Dr. Piero Giubilato (CERN) for his helpful comments on the earlier drafts of this thesis, all our discussions about the radiation effects and radiation testing, and his positive energy.

I am also thankful to Dr. Luciano Musa (CERN) for the opportunity to become part of the ALICE ITS team. I would like to extend thanks to my colleagues from the ALICE ITS group: Matthias Bonora, Andres Sanchez Gonzalez, Jacobus van Hoorne, Matteo Lupi, Felix Reidt, Tomáš Vaňát, Mónika Varga-Kőfaragó, Arild Velure, and everyone whom I failed to mention, for their smile and great support.

Next, I would like to thank the mentors I met outside of CERN. I am grateful to Melanie Berg (ASRC Federal Space & Defense in support of NASA Goddard Space Flight Center) for her valuable comments, advice and guidance within the vast topic of radiation effects on FPGA devices and testing methodologies. I also would like to thank Professor Mike Wirthlin (Brigham Young University) for his support and numerous discussions about radiation mitigation techniques. I am also thankful to Gary Swift for discussions about SRAM-based FPGAs radiation testing techniques and sharing his vast experience in the field.

I am very grateful, as well, to Dr. Krzysztof Kasiński (AGH University of Science and Technology) for providing valuable comments to an earlier draft of this thesis.

Last but not least, I want to thank my family and friends, especially my wonderful fiancée Izabela, for their unconditional support and patience.

„Trzeba działać!”

Jan Sielewicz

Dla Izabeli

Abstract

Mitigation methods increasing radiation hardness of the FPGA-based Readout of the ALICE Inner Tracking System

Over the years 2019–2020, the ALICE experimental apparatus will undergo a major upgrade. A key element of the modernization is the construction of a new Inner Tracking System (ITS) consisting of 24,120 monolithic active pixel sensors (MAPSs). The sensors' constrained power budgets limit their data line driving capability, forcing the readout electronics to be located as close as possible to the detector in a hostile radiation environment. The readout system serves many various functions. It configures, monitors and reads out data from the detector and interfaces with the trigger and power systems. It also preprocesses and packetizes the received data, and streams the packets out to the data acquisition system. Because of the tasks that it performs, it requires powerful and flexible processing units. Also, as new functionalities are usually implemented over time, it is required that the readout system can be upgraded. Modern FPGA devices meet all of the aforementioned requirements. Unfortunately, dedicated radiation-hard by design FPGAs cannot be utilized, as they are either limited in resources or too expensive. Conversely, commercial-grade SRAM-based FPGAs offer a large amount of logic resources and transceivers, and their price is low enough to be used in large quantities. Although they are susceptible to ionizing and non-ionizing radiation, there exist radiation mitigation methods that can be employed to increase the radiation hardness.

Mitigation methods that increase the radiation hardness of non-radiation-hardened SRAM-based FPGAs are the subject of this dissertation. The author verifies the hypothesis that it is possible to design and deploy the readout system employing commercial, non-radiation-hardened SRAM-based FPGAs for the upgraded ALICE Inner Tracking System. The design functional error rate estimation methodology is proposed, which is followed by the explanation of concepts of various radiation mitigation methods based on: spatial redundancy, refreshing of configuration memory of SRAM-based FPGAs, and triplication of inputs and outputs. Basic building blocks of an SRAM-based FPGA design (combinational and sequential circuits, Block RAM, Finite State Machines) were imple-

mented with various mitigation methods employed, and experimentally evaluated via fault injection and irradiation tests. Finally, implementation guidelines based on the obtained experimental results were formulated for the application in the FPGA-based Readout of the upgraded ALICE Inner Tracking System.

In the dissertation the author also includes the description of the prototype readout electronics that were utilized as the main platform for radiation testing and various development activities while designing the ITS Readout System.

Key words: CERN, LHC, ALICE, Inner Tracking System (ITS), MAPS, readout system, SRAM-based FPGA, radiation, SEE, SEU, SET, SEFI, fault injection, irradiation.

Streszczenie

Metody zwiększania odporności radiacyjnej dla wewnętrznego systemu śladowego opartego na układach FPGA w eksperymencie ALICE

W latach 2019–2020 eksperyment ALICE w CERN przejdzie gruntowną modernizację, podczas której zostanie zainstalowany nowy wewnętrzny system śladowy (Inner Tracking System - ITS). Będzie się on składał z 24,120 sensorów wykonanych w technologii Monolithic Active Pixel Sensor (MAPS). Ze względu na fakt, że sensory pikselowe zostały zaprojektowane z uwzględnieniem jak najniższego poboru energii elektrycznej, nie są w stanie transmitować danych na dużą odległość. W związku z tym system odczytu danych do detektora ITS musi zostać zlokalizowany możliwie blisko niego - w środowisku gdzie występuje promieniowanie jonizujące i niejonizujące. System odczytu danych pełni wiele funkcji dotyczących sterowania, kontroli i odczytywania danych z detektora, wstępnego przetwarzania oraz komunikacji z systemami zasilania, wyzwalania i akwizycji danych. Wykorzystanie reprogramowalnych układów FPGA pozwala na implementację skomplikowanych funkcjonalności oraz umożliwienie modyfikacji pracy systemu wraz z czasem użytkowania. Odporne na wpływ promieniowania układy FPGA nie mogą zostać wykorzystane ze względu na niewystarczającą ilość zasobów logicznych lub zbyt wysoki koszt. Natomiast układy FPGA oparte o pamięć konfiguracyjną typu SRAM, które oferują wystarczają ilość zasobów logicznych i szybkich peryferii do transmisji danych, nie są odporne na promieniowanie jonizujące i niejonizujące. Istnieją jednak metody, które pozwalają na zwiększenie odporności radiacyjnej układów SRAM FPGA.

Niniejsza praca doktorska dotyczy metod zwiększania odporności radiacyjnej struktur logicznych implementowanych w komercyjnych układach SRAM FPGA, które nie zostały zaprojektowane do pracy w środowisku, gdzie występuje promieniowanie jonizujące i niejonizujące. Autor poddał weryfikacji tezę o tym, że system odczytu danych do detektora ITS może zostać skonstruowany w oparciu o komercyjne układy SRAM FPGA, które nie zostały przystosowane do pracy w środowisku radiacyjnym. W tym celu została zaproponowana nowatorska metodologia pozwalająca na szacowanie częstości występowania błędów funkcjonalnych w strukturach logicznych zaimplementowanych w SRAM

FPGA. Przedstawione i opisane zostały metody zabezpieczania struktur logicznych oparte na redundancji przestrzennej, wykorzystaniu kodów korekcji błędów, odświeżaniu pamięci konfiguracyjnej układów SRAM FPGA oraz potrajaniu układów wejścia-wyjścia. Zaprezentowane zostały również metody testowania polegające na sztucznym wstrzykiwaniu błędów do pamięci konfiguracyjnej układów SRAM FPGA oraz napromieniowywaniu za pomocą testowych wiązek cząstek. Opracowana metodologia pozwala na testowanie odporności radiacyjnej struktur implementowanych w układach SRAM FPGA w laboratorium, bez konieczności wykorzystywania drogiego i czasochłonnego napromieniowywania skracając tym samym czas i obniżając koszt testowania. Autor eksperymentalnie testuje podstawowe typy układów logicznych implementowanych w SRAM FPGA (układy kombinacyjne i sekwencyjne, pamięć blokowa, maszyny stanów) z zastosowaniem różnych technik zabezpieczania przed promieniowaniem. Następnie przeprowadzona zostaje analiza danych eksperymentalnych dla wielu różnych struktur testowych i na podstawie wyników zostają sformułowane zalecenia dotyczące implementacji finalnego projektu dla układu SRAM FPGA, który będzie pracował w systemie odczytu danych dla nowego detektora ITS.

W rozprawie doktorskiej autor zamieszcza również opis zaprojektowanego i uruchomionego przez siebie prototypowego systemu odczytu danych dla detektora ITS. Był on wykorzystywany jako główne narzędzie służące do prowadzenia testów radiacyjnych oraz prac rozwojowych podczas opracowywania systemu odczytu danych do detektora ITS.

Słowa kluczowe: CERN, LHC, ALICE, wewnętrzny system śladowy (ITS), MAPS, system odczytu danych, SRAM-FPGA, promieniowanie, SEE, SEU, SET, SEFI, wstrzykiwanie błędów, napromieniowywanie.

Contents

Preface

Motivation, subject and scope of the thesis

The development of particle detectors and electronic systems for high-energy physics (HEP) experiments at accelerator facilities (e.g. CERN Large Hadron Collider [1]) is driven by their dedicated physics research programs. Analyzing rare particle collision events with high recurrence and precision pushes towards constant modernization of the experimental apparatus. Following new research objectives, the continuous development of novel solid-state particle sensors for high-energy physics experiments allows for tracking more complex collision events at higher rates and with better precision. However, the aforementioned technology advancements pose many demanding design challenges. First of all, an increase in beam luminosity results in higher radiation levels in the experimental areas where specialized electronic equipment operates. What is more, the mentioned increase in beam luminosity and a higher detection resolution lead to a significant increase of data throughput that have to be captured and processed.

Over the years 2019–2020, the ALICE experimental apparatus will undergo a major upgrade. A key element of the modernization is the construction of a new Inner Tracking System (ITS) consisting of 24,120 monolithic active pixel sensors. The sensors' constrained power budgets limit their data line driving capability, forcing the readout electronics to be located as close as possible to the detector in a hostile radiation environment. The ITS Readout System serves various functions. First of all, it configures, monitors and reads data out from the detector. Furthermore, it interfaces with the trigger and detector power systems, preprocesses and packetizes the received data, and streams the packets out to the data acquisition system for storage and detailed analysis. Because of the abundance of tasks that it performs, it requires powerful and flexible processing units. To allow for the implementation of new functionalities over time, the Readout System is required to be upgraded. Modern FPGA devices meet all of the aforementioned requirements. They offer a large amount of logic resources, high-speed signal transceivers, and re-configurability to modify or extend functionalities during the lifetime of an experiment. Also, developing an FPGA design is faster and cheaper than designing a custom ASIC. As it has been

already mentioned, the ITS Readout System has to be in close proximity of the detector in radiation environment. Unfortunately, dedicated radiation-hard by design FPGA devices cannot be utilized, as they are either limited in resources or too expensive. Conversely, commercial-grade SRAM-based FPGAs offer a large amount of logic resources and transceivers, and their price is low enough to be used in large quantities. However, they are susceptible to ionizing and non-ionizing radiation effects, which have limited their employment so far. Thus, as commercial-grade non-radiation-hardened SRAM-based FPGAs can significantly contribute to the development of the Readout Systems, there is a strong interest in evaluating the possibility of using these devices in radiation environments similar to the one in which the ALICE ITS Readout will operate.

Mitigation methods increasing radiation hardness of non-radiation-hardened SRAM-based FPGAs are the subject of this dissertation. In order to characterize the operation of different FPGA design logic modules in the radiation environment, the design functional error rate estimation methodology is proposed. It is followed by the explanation of concepts of various radiation mitigation methods based on: spatial redundancy, refreshment of configuration memory of SRAM-based FPGAs, and triplication of inputs and outputs. Later, basic building blocks of an SRAM-based FPGA design (combinational and sequential circuits, Block RAM, Finite State Machines) are implemented with various mitigation methods employed. Then, they are experimentally evaluated via fault injection and irradiation tests. Finally, implementation guidelines based on experimental results are formulated for the application in the FPGA-based Readout of the upgraded ALICE Inner Tracking System. However, the presented design functional error rate estimation methodology and following radiation mitigation methods can also be applied to other HEP systems, space instrumentation, or to other electronic systems that operate in a radiation environment.

Structure of the thesis

The doctoral thesis consists of 8 chapters and 3 additional appendices. In Chapter 1 the Large Hadron Collider at CERN is briefly presented. Then, the ALICE experiment is shown and its main electronic systems are explained. The description is followed by the introduction of the ALICE Inner Tracking System (ITS) Upgrade Project. Later, a construction of the new detector is explained. Then, the design of the Readout System for the ALICE Inner Tracking System is presented and its main design constraints are emphasized. Finally, the environment where the ITS Readout System will operate is discussed. Chapter 2 describes radiation effects on FPGA devices. First, FPGA architectures and their application areas are presented. Then, radiation-induced effects are

discussed. Finally, the operation of SRAM-based FPGA devices in different radiation environments is shown and potential risks are indicated. Chapter 3 contains the motivation for this dissertation, which is followed by a definition of the hypothesis. Then, thesis objectives are formulated and discussed. Chapter 4 presents the methodology of design functional error rate estimation. Later, the designed Emulator and experimental testing techniques are shown and discussed. Chapter 5 describes different Single Event Upset mitigation methods based on spatial redundancy, configuration memory scrubbing, and inputs-outputs triplication. Chapter 6 describes the design of the hardware prototype platform. First, the motivation and design objectives are formulated. Later, architecture and implementation details are depicted. Chapter 7 describes in detail testing methodologies that were employed while evaluating basic building blocks of an SRAM-based FPGA design. The description is followed by a presentation of the experimental setup that was employed during fault injection and irradiation tests. Later, various testing designs, their evaluation and experimental results are shown and discussed. Chapter 8 summarizes the experimental results and discusses the main contributions to the Readout System of the ALICE ITS. Additional information that are not presented in the main body of the dissertation are included in the Appendices.

Chapter 1

Introduction

1.1 The Large Hadron Collider

The Large Hadron Collider (LHC) [1] is a circular synchrotron. It was installed in an existing 26.7 km long tunnel, located from 50 m to 75 m under ground level in the Geneva area across the Swiss-French border, which had been previously built for the CERN Large Electron-Positron Collider machine. The LHC became operational in 2008 and is the latest upgrade of the CERN's accelerator complex presented in Fig. 1.1.

The LHC accelerates and collides two counter rotating beams of protons or heavy ions. Greatly simplifying, it consists of two vacuum beam pipes integrated into numerous superconducting magnets, and accelerating cavities, which respectively control and accelerate the trajectory of the particles. There are in total 1232 custom-designed, state-of-the-art bending dipoles. Each of them is 14.3 m long, weighs 35 t, and contains two independent vacuum beam pipes. The magnetic field of 8.4 T, required to bend the accelerated particles, is generated by an electric current of 11,700 A flowing though niobium-titanium superconductive coils cooled down to 1.9 K. The cooling is provided by the liquid helium cryogenic installation. Before particles are collided, they go through a series of steps in which their energy is gradually increased. First, they are accelerated to 50 MeV by a linear accelerator LINAC 2. Next, they are injected to the circular Proton Synchrotron Booster where their energy is further increased to 1.4 GeV. Then, they are transferred to the Proton Synchrotron Booster. When they reach a 25 GeV energy, they are injected to the Super Proton Synchrotron, which further accelerates them to 450 GeV before the final injection to the LHC, where the final energy of 6.5 TeV per beam is reached. Each proton beam consists of bunches of $1.15 \cdot 10^{11}$ particles. Each bunch is 30 cm long, and the spacing between two bunches in the vacuum pipe equals 7.5 m. The bunch crossing interval equals 25 ns (40 MHz).

At four Interaction Points positioned along the LHC at regular intervals, the special-

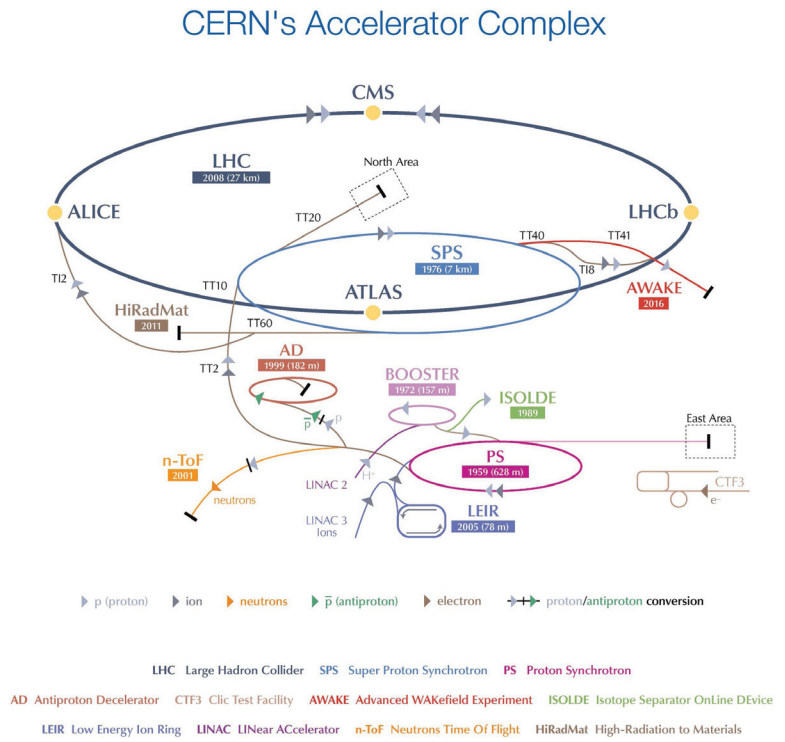


Figure 1.1: Accelerator complex at CERN (CERN copyright).

ized groups of three quadropole magnets (called the inner triplets) focus the two counter-rotating beams so that they can collide in well-defined positions, where the particle detectors are installed. The particles are collided with a maximum centre-of-mass energy of up to 13 TeV in the Interaction Points in the center of the detectors. The products of the particle interactions are recorded by the detectors, and then data analysis algorithms are employed to analyze collisions. A Toroidal LHC Apparatus (ATLAS) [2] and Compact Muon Solenoid (CMS) [3] detectors are dedicated to carrying out experimental tests of supersymmetry [4] and the Higgs mechanism [5, 6]. The Large Hadron Collider beauty (LHCb) [7] is designed to study the CP violation and physics of the B meson. A Large Ion Collider Experiment (ALICE) [8] is described in detail in Section 1.2.

1.2 The ALICE Experiment

A Large Ion Collider Experiment (ALICE) [8], whose general layout is shown in Fig. 1.2, is a general-purpose, heavy ion collision detector. The physical dimensions of the experimental apparatus are 16 m × 16 m × 26 m, and it weighs approximately 10,000 t. It has

been specifically designed to investigate the physics of strongly interacting matter and the Quark-Gluon Plasma [9] at high values of energy density and temperature in relativistic nucleus-nucleus collisions. Data taken during proton-proton runs at the top LHC energy provides reference data for the heavy ion programme and addresses a number of physics observables for which ALICE is complementary to the other LHC detectors. The ALICE experiment consists of 18 different sub-detector systems grouped into a central barrel and forward section. Their design was driven by the unique physics requirements (the extreme particle multiplicity anticipated in central Pb–Pb collisions) and the experimental conditions expected at the LHC. Each one was customized in terms of design requirements and construction technology. The different subsystems were optimized both to provide high-momentum resolution and particle identification over a broad range in momentum, up to the highest expected multiplicities.

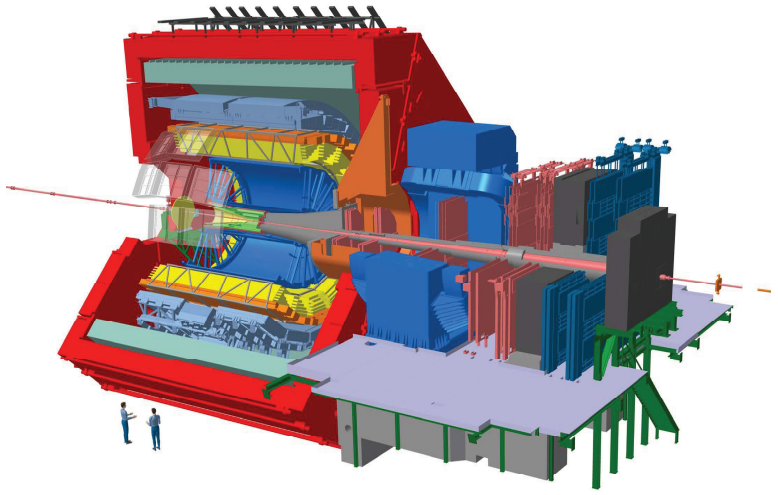


Figure 1.2: ALICE experimental apparatus after LS2 (ALICE Collaboration copyright).

There are eight central sub-detectors within the central barrel (Inner Tracking System, Time-Projection Chamber, Transition Radiation Detector, Time-Of-Flight Detector, High-Momentum Particle Identification Detector, Photon Spectrometer, Electromagnetic Calorimeter, ALICE Cosmic Ray Detector) which measure hadrons, electrons, and photons. Five other detectors cover the forward direction (a muon spectrometer, Zero Degree Calorimeter, Photon Multiplicity Detector, Forward Multiplicity Detector, V0 detector, T0 detector) and are used for detecting muons, global event characterization, and triggering.

The ALICE is equipped with a dedicated multilevel trigger system. The Central Trigger Processor (CTP) triggers the detectors. Later, the High-Level Trigger (HLT)

filters and compresses the data received from the detectors, and transmits them to the Data Acquisition System (DAQ). Selected events are permanently stored for later analysis by the Offline computing system. The Detector Control System (DCS) provides control and online monitoring of all the detectors in the experiment.

The operational conditions for the front-end and readout electronics systems installed near the experimental apparatus are demanding, especially because of ionizing and non-ionizing radiation which may lead to a Single Event Effect disrupting a correct operation, or to destructive latch-up conditions [10]. Also, the presence of a magnetic field imposes many design limitations for electronic systems.

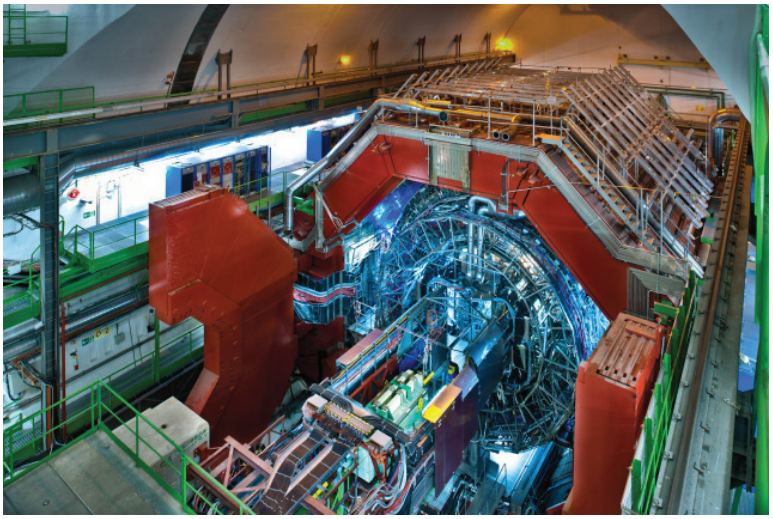


Figure 1.3: ALICE experimental apparatus with solenoid magnet door opened (ALICE Collaboration copyright).

1.3 The Upgraded ALICE Inner Tracking System

During the Second Long LHC Shutdown (LS2) [11] in the years 2019–2020, the ALICE experimental apparatus will undergo a major upgrade. A key element of the modernization is the construction of a new ultra-light, high-resolution Inner Tracking System (ITS) [12]. It is the innermost barrel detector of the ALICE apparatus (Fig. 1.2), closest to the Interaction Point. Its main functions are tracking charged particles produced in collisions, and reconstructing primary vertices of primary beam collisions and secondary vertices from the decay of short-lived particles produced in the primary collisions. To meet the physics requirements in terms of spatial resolution and tracking speed, the upgraded

ITS shown in Fig. 1.4, has been designed as a barrel detector arranged in 7 concentric layers of different lengths to optimize solid angle coverage. The Inner Barrel (IB) consists of the three innermost layers, while the Outer Barrel (OB) contains the other four layers. They are azimuthally segmented into units called staves, which are electrically and mechanically independent. The staves within each group of layers share a common design and are fixed to the end-wheels, which serve as precision support structures. Cooling pipes and cabling enter only from one side of the detector, simplifying its maintenance.

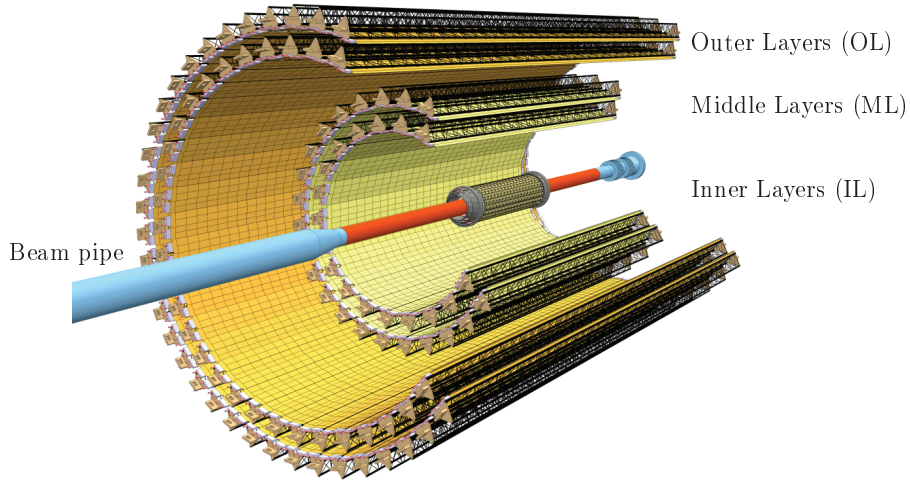


Figure 1.4: Layout of upgraded Inner Tracking System detector [12].

One of the most innovative aspects of the new ITS is the use of Monolithic Active Pixel Sensor ASICs to operate within its entire surface [13, 14]. The very same sensor design is used across the entire ITS, both in the IB and OB, even though they clearly differ in terms of the expected tracks density. It has been specifically designed for the ALICE experiment and it is implemented in 0.18 µm TowerJazz CMOS Imaging Technology [15]. The sensor size is 30 mm × 15 mm with about 28 µm pixel pitch, and it is thinned down either to 50 µm on the Inner Layers or to 100 µm on the Outer ones. It embeds s high-speed serial data interface and a bidirectional control bus for configuration and monitoring. There are a total of 24,120 sensors in the ITS, creating a detection surface of about 10 m^2, which is segmented into 12.6 billion pixels.

The Inner Barrel (IB) layers consist of staves which incorporate Hybrid Integrated Circuit (HIC), a space frame, and a cold plate (Fig. 1.5). The IB HIC (Fig. 1.6) is equipped with 9 pixel sensors which are glued and wire-bonded to the Flex Printed Circuit (FPC). The HIC is glued to the carbon fibre space frame, which provides mechanical fixation and cooling. There are 48 staves within layers 0, 1 and 2 which utilize 432 pixel chips in total. The length of the IB HIC is 30 cm. The pixel sensors within the IB HIC share common

clock and control signals, and they have their own high-speed serial outputs running at 1.2 Gbps. Block diagram of the Inner Barrel Module is shown in Fig. 1.7.

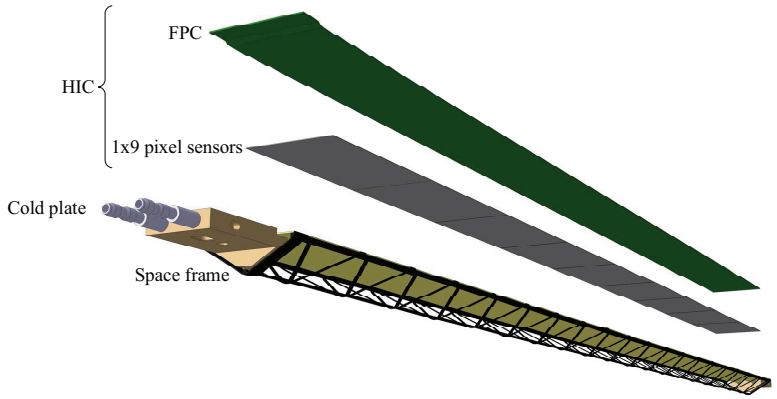


Figure 1.5: Inner Barrel stave design.

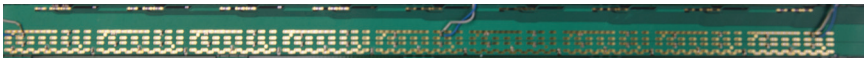


Figure 1.6: Inner Barrel Module physical design.

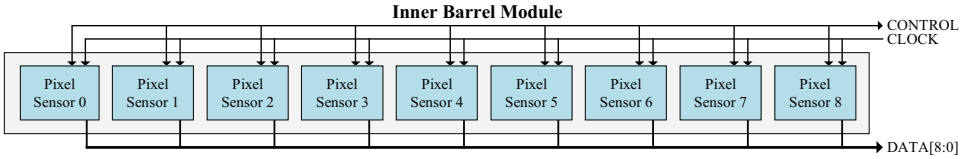


Figure 1.7: Block diagram of Inner Barrel Module.

The Outer Barrel (OB) layers also consist of staves. Each OB stave incorporates 2 half-staves installed on a common carbon fibre space frame. Each OB half-stave incorporates OB HICs, a power bus, a bias bus, and a cold plate (Fig. 1.8). The OB HIC (Fig. 1.9) contains 2 rows of 7 pixel sensors which are glued and wire-bonded to the FPC. To minimize the number of connections in the direction of the Readout System, one sensor (per row) operates in master mode, which propagates clock and control signals to the other sensors in the row and collects their data via a shared inter-chip bus. The HICs in the half-stave share clock and control signals. Power to the OB HICs within the same half-stave is delivered by a shared power bus. Each OB HIC has two data outputs (one for each master chip) running at 400 Mbps. Block diagram of the Outer Barrel (OB) Module is shown in Fig. 1.10. One Middle Layers half-stave consists of 4 HICs, and the

Outer Layers one of 7 HICs. There are 23,688 pixel sensors utilised in the Outer Barrel layers. There are 144 staves in total (54 for Middle Layers and 90 for Outer Layers). The corresponding lengths are 84 cm and 150 cm.

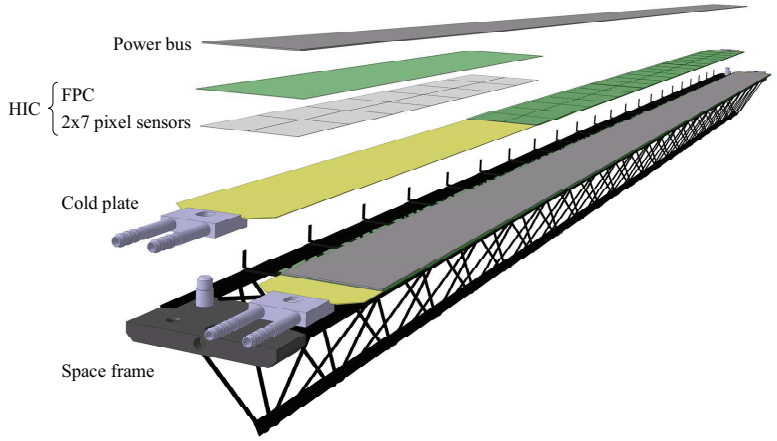


Figure 1.8: Outer Barrel stave design.

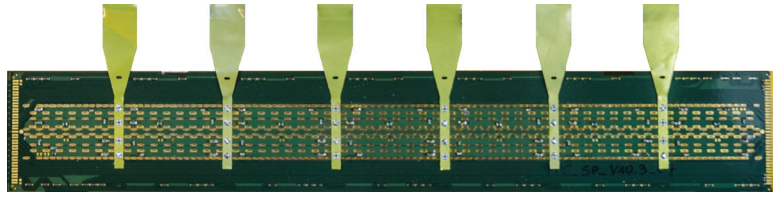


Figure 1.9: Outer Barrel Module physical design.

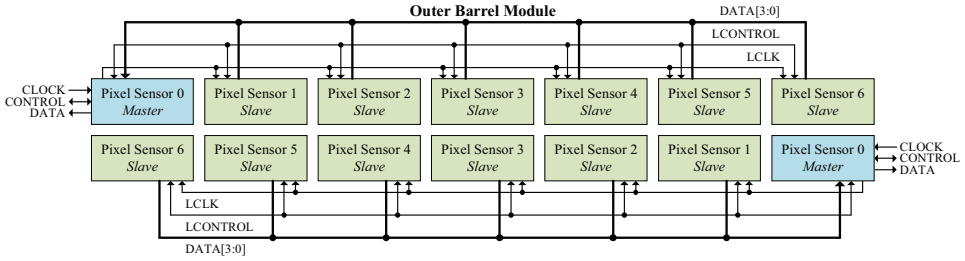


Figure 1.10: Block diagram of Outer Barrel Module.

1.4 Readout of the Upgraded ALICE Inner Tracking System

The ITS Readout System (RS), illustrated in Fig. 1.11, is composed of 192 Readout Units (RUs) and 168 Power Boards (PBs). They are all located about 5 m from the detector's end-wheels. The RUs configure and control the pixel sensors, receive and assemble data, and manage the PBs. The pixel sensors are controlled and monitored via 624 bidirectional, differential lines, and the clock signals are also distributed via 624 lines. Their data are read out via 3816 differential high-speed lines. These are made of segments of microstrip tracks on the FPCs and of micro-twinax cables [16] from the staves' edges to the RS. Between the detector and the RS, the interconnection lines are grouped into the micro-twinax ribbon cables, each carrying 12 differential pairs. Expected data rates generated by the ITS for the two baseline modes of operation, 50 kHz Pb–Pb and 200 kHz pp respectively, are detailed in [17]. The Readout System interfaces with the ALICE Online and Offline computing system (ALICE O^2) [18] and the Central Trigger Processor (CTP) via the Gigabit Transceiver (GBT) optical links [19]. Each RU has one downstream optical control link, one downstream trigger link, and three upstream data links. The PBs deliver and monitor power to the detector. A summary with the number of resources used per layer is presented in Table 1.1.

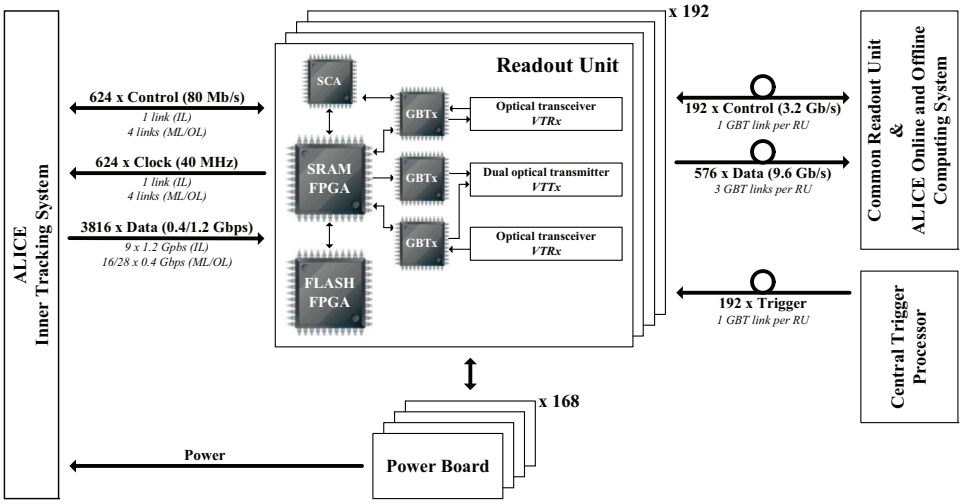


Figure 1.11: Architecture of ITS Readout System.

	Layout				Optical fibers		
Layer	Staves	Micro-twinax ribbon cables	RUs	PBs	Trigger	Data	Control
0	12	12	12	6	12	36	12
1	16	16	16	8	16	48	16
2	20	20	20	10	20	60	20
3	24	96	24	24	24	72	24
4	30	120	30	30	30	90	30
5	42	168	42	42	42	126	42
6	48	192	48	48	48	144	48
Total			**192**	**168**	**192**	**576**	**192**

Table 1.1: Usage of micro-twinax ribbon cables, Readout Units (RUs), Power Boards (PBs), and optical fibers by ITS Readout System.

1.5 Operational environment

The ITS Readout System will be installed in the underground ALICE experimental cavern. Like all experiments installed at the LHC, the entire ALICE apparatus operates in the harsh radiation environment resulting from the beams interactions [20]. The expected radiation levels at the position where racks with the electronic Readout Units and Power Boards will be located are as follows: a Total Ionising Dose (TID) of 10 krad (including a safety factor of 10) and a 1 MeV n_{eq} fluence of $1.6 \cdot 10^{11}\,cm^{-2}$. The main concern for the readout electronics comes from the Single Event Effects induced by charged hadrons of energy greater than 20 MeV. The high-energy hadron flux is expected to be approximately $10^3\,cm^{-2}\,s^{-1}$ [21]. The radiation field expected in the experiment is shown in Fig. 1.12. Another requirement for the readout electronics is to operate in a constant magnetic field of 0.5 T.

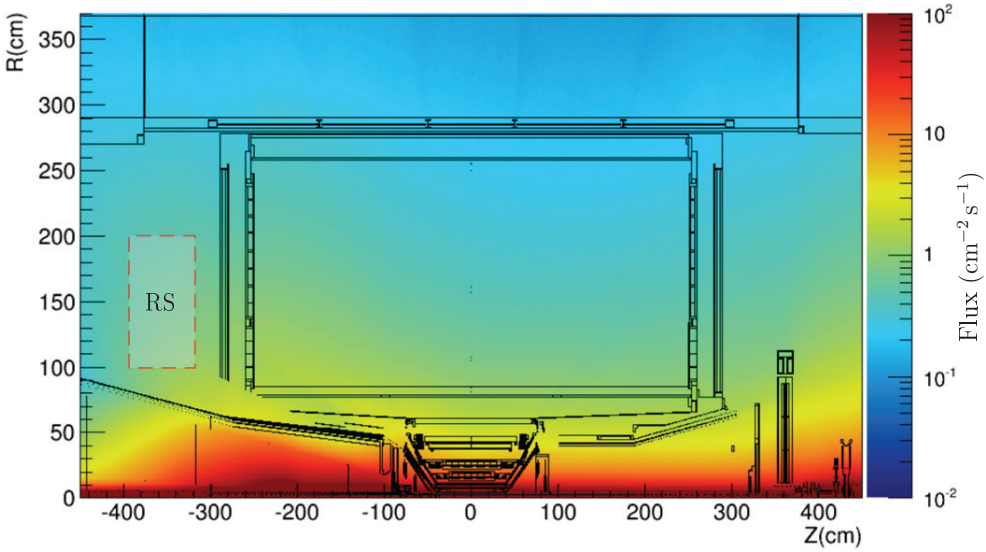


Figure 1.12: Flux of high-energetic hadrons ($E_{\mathrm{kin}} > 20\,\mathrm{MeV}$) in ALICE experimental cavern after LS2 [21]. Position of ITS Readout System (RS) is marked.

Chapter 2

Radiation effects on FPGA devices

2.1 FPGA devices and areas of their application

2.1.1 FPGA devices architecture

A Field-Programmable Gate Array (FPGA) is an advanced, configurable semiconductor device whose final functionality is defined after its manufacturing [22]. Its simplified architecture is presented in Fig. 2.1. It consists of a logic fabric, global routing network, a

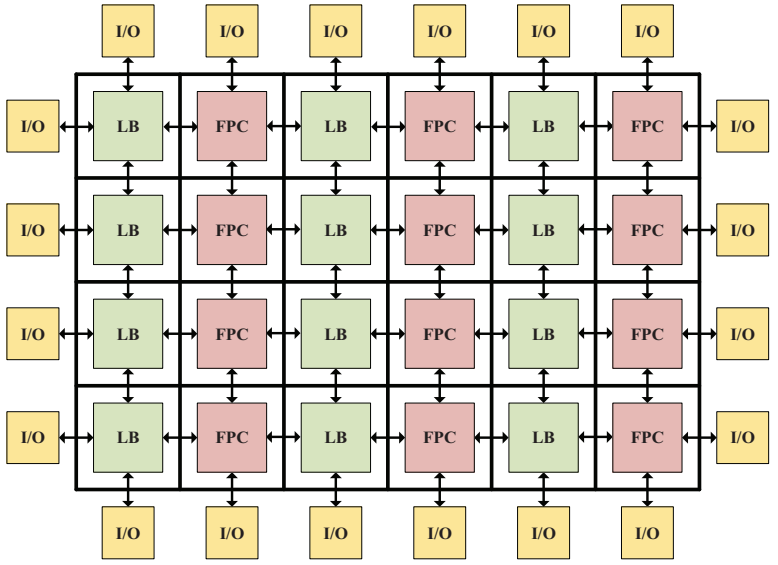


Figure 2.1: Simplified FPGA architecture.

ring of I/O blocks, and various functionally predefined components (FPC) [23]. A logic fabric contains a matrix of logic blocks (LB) that can be configured to execute combinational functions of different levels of complexity. A global routing network connects

various FPGA components together, and it usually occupies about 80 % of the total device area. There are two different global routing network architectures: hierarchical [24] or island-style [25, 26]. The latter one is the most commonly used among academic and commercial FPGAs, where logic blocks are arranged on a 2D grid and are interconnected by a programmable routing network [27]. The I/O blocks interface off-chip signals operating in different I/O standards to the internal FPGA routing network. A functionally predefined component can be: a built-in memory block or external memory controller; an embedded DSP block, multiplier or processor; clock Phase-locked Loop (PLL) or Delay-locked loop (DLL); high-speed serializer-deserializer, data transceiver; analog-to-digital converter. Usually these components are optimized for speed, performance, power consumption, and occupied area on the die.

2.1.2 Technology choice for FPGA devices

SRAM-based FPGAs

Presently, SRAM, flash and antifuse technologies are employed to implement programmable switches that configure all FPGA components and interconnections. Because of re-programmability and usage of standard CMOS manufacturing process, SRAM programming technology has become the most common approach for modern FPGA devices. As the implementation of SRAM cells does not require any special integrated circuit processing steps beyond standard CMOS process, Xilinx, Intel (previously Altera Corp.), and Lattice Semiconductors offer devices exploiting the latest semiconductor technology with smaller geometries, which brings significant advantages, such as increased integration level and operational frequency, and lower dynamic power consumption [23,27]. Fig. 2.2 shows a reference 6-transistor CMOS SRAM memory cell which is distributed all over the SRAM-based FPGA die. These cells configure logic fabric and multiplexers steering interconnect signals. In comparison to other programming technologies SRAM-based FPGAs can be configured an indefinite number of times, even during circuit operation [28, 29]. SRAM memory is volatile, which means that at every power-up all configuration cells have to be programmed. For some applications this requirement can constitute a drawback because storing configuration data requires an external flash or EEPROM memory component, increasing both price and system complexity. What is more, since the configuration data must be loaded into the device at power-up, there is a risk that the configuration information might be intercepted. However, FPGA vendors managed to address this issue by employing encryption mechanisms to protect the configuration process for the applications requiring increased security level. Also, for the applications requiring reduced complexity, there are SRAM-based FPGAs that embed on-chip banks of flash memory to hold the

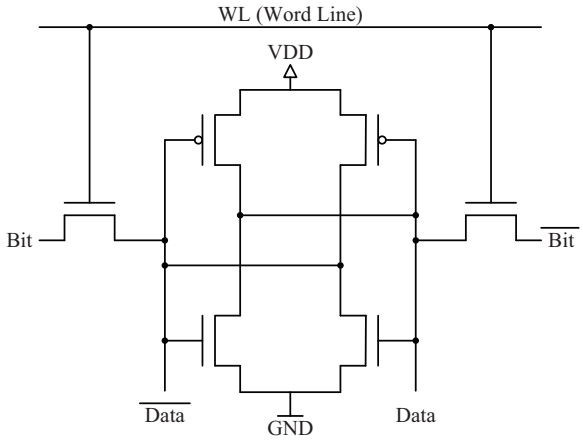


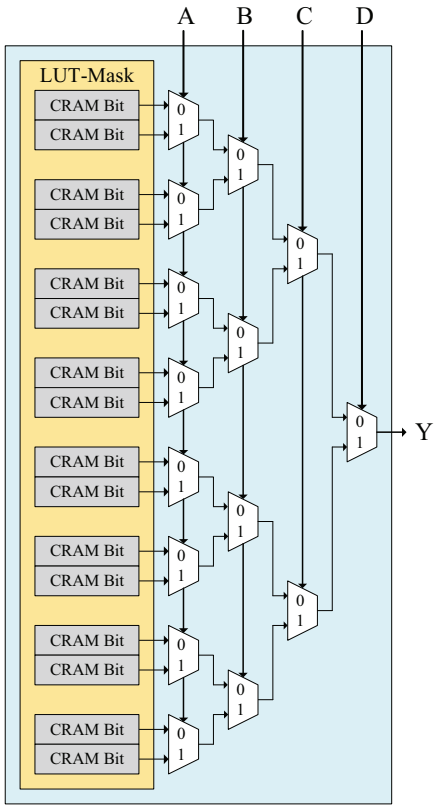
Figure 2.2: Static memory cell used in SRAM-based FPGA devices [27].

configuration data.

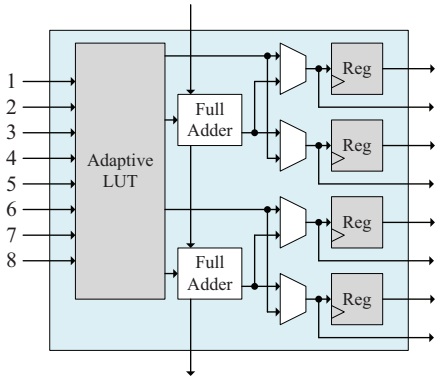
Vendors of SRAM-based FPGA devices proposed different naming and architectures of their logic blocks. Xilinx employs Configuration Logic Blocks (CLB) [30], while Intel uses Adaptive Logic Modules (ALM) [31]. Both logic blocks contain a Look-up Table (LUT), whose simplified 4-input architecture is presented in Fig. 2.3a. Some key components of an LUT are an SRAM-based LUT-mask and a set of multiplexers which select the CRAM bits that drive the output [32]. In general, a k-input LUT consists of 2^k memory configuration bits and can implement any k-input boolean function. The shown LUT (Fig. 2.3a) incorporates 16 SRAM configuration bits to implement any 4-input boolean function. Fig. 2.3b shows simplified architecture of the ALM employed in the Intel Arria V, which consists of an 8-input LUT, two adders, four multiplexers, and two registers. To optimally utilize available 8-input LUTs, the ALM can implement, for example, two independent 4-input functions, or a 5-input function and a 3-input function with independent inputs. However, to reduce the number of required configuration SRAM bits, 8-input LUT can implement at most 6-input function [31]. One or many adjacent ALMs implement a logic function, depending on its complexity. In Xilinx 7-series FPGAs, each CLB is divided into two slices. All slices contain four 6-input LUTs, wide multiplexers, carry logic chain and eight flip-flops. The 6-input LUT can implement a 6-input function or two 5-input functions with common or independent inputs [30].

Flash-based FPGAs

Another type of FPGA employs non-volatile flash-based programming technology. In those devices, flash cells are distributed throughout the die to connect signal lines to



(a) Look-up Table (LUT) [32].



(b) Adaptive Logic Block (ALM) employed in Intel Arria V family devices [31].

Figure 2.3: Simplified architectures of Look-up Table (a) and Adaptive Logic Block (b).

inputs and outputs of logic blocks. Fig. 2.4 shows an example of a flash-based switch [33]. It consists of two transistors sharing a common floating gate which stores the programming information. The first transistor writes and verifies the floating gate voltage, whereas the second one (the switching transistor that connects or separates routing nets) configures logic fabric and erases the floating gate. Flash-based FPGAs are non-volatile, which

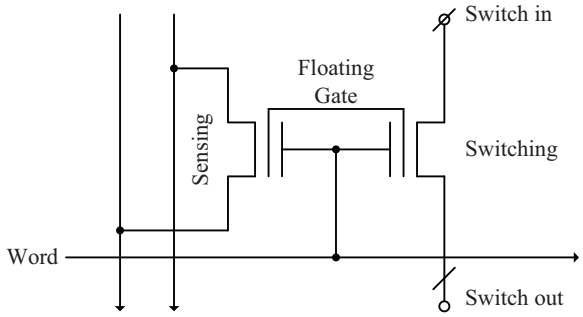


Figure 2.4: Flash-based switch used in flash-based FPGA devices [33].

means that they do not require any external memory device to keep their configuration data. They also operate immediately after power-up. Although flash-based cells can be reprogrammed, this process can be repeated only a limited number of times due to a charge buildup in the oxide that eventually prevents a flash-based device from being properly erased and programmed [34]. Fig. 2.5 shows the architecture of the VersaTile logic block employed in Microsemi ProASIC3 flash-based FPGA. The VersaTile can implement a 3-input logic function, a single D-type flip-flop with enable, clear or set inputs, or a latch with clear or set inputs. Compared to an SRAM-based logic block, a flash-based one embeds much simpler logic capabilities.

Antifuse FPGAs

An alternative to SRAM and flash-based programming technologies is antifuse technology. Instead of reprogrammable switches, it is based on one-time programmable connections. Activated by a high-current pulse, they create a permanent, passive, low-impedance path that connects different components together. Fig. 2.6 shows the interconnection elements employed in Microsemi RTAX antifuse FPGAs [35]. Their basic logic block is called the SuperCluster (Fig. 2.7c), and it consists of two clusters, each containing two combinational cells (C-cell), a single register cell (R-cell), two Transmit (TX) and two Receive (RX) routing buffers (Fig. 2.7). The C-cell (Fig. 2.7a) can implement up to 5-input logic function, whereas the R-cell (Fig. 2.7b) is a physically triplicated register, which to the user, appears as a single D-type flip-flop.

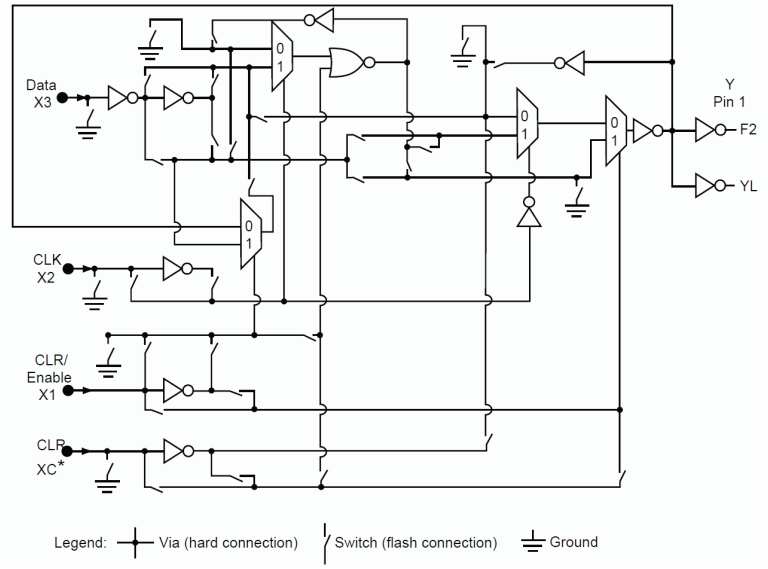


Figure 2.5: Architecture of VersaTile logic block in Microsemi ProASIC 3 [33].

2.1.3 FPGA application areas

FPGAs are available in various hardware configurations: different number of logic elements, built-in memory cells, clocking resources, physical inputs and outputs, and high-speed serial data transceivers. The variety of devices well suit the needs of both customer and professional applications. The main areas of their usage are:

- aerospace and defense: avionics; communication systems; guidance systems for aircraft, ship, missile, munitions, rocket and satellite; radars and sonars;
- ASIC prototyping;
- audio: professional audio equipment; Software Defined Radio applications;
- audio and video broadcasting systems;
- automotive: driver assistance systems; in-vehicle infotainment systems; high-resolution video and image processing systems; vehicle connectivity and networking systems;
- consumer electronics;
- data centers, data storage, high-performance computing: high-bandwidth, low-latency servers; data storage systems; high-speed trading applications; cryptocurrency mining applications;
- industrial: imaging and surveillance systems; automation systems; motor control applications;
- medical: diagnostic applications (ultrasound, computer-tomography, MRI, PET, X-ray); surgical video systems;

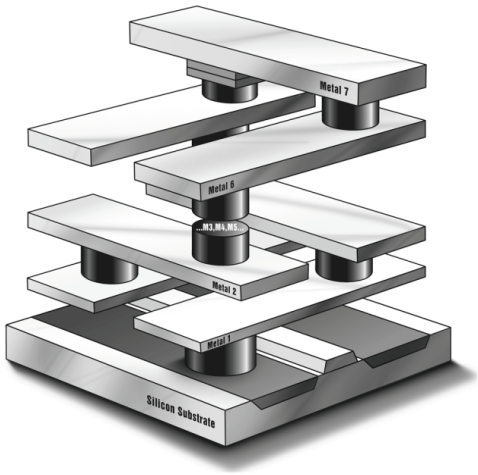


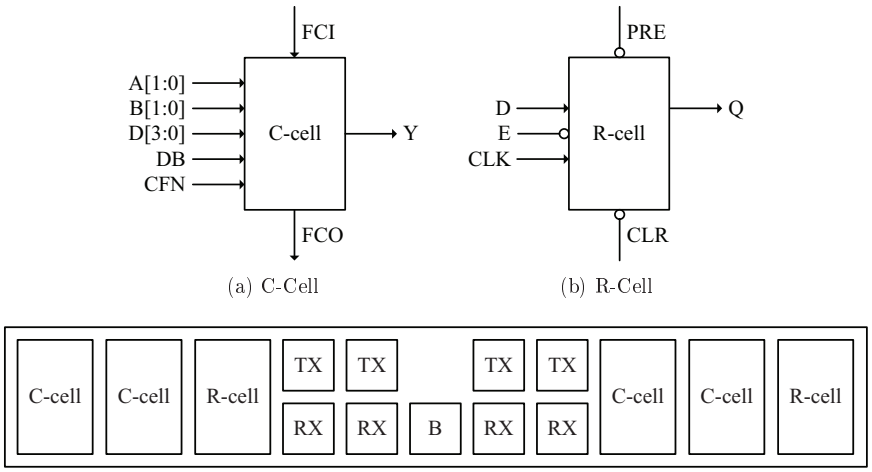
Figure 2.6: RTAX interconnect elements [35].

- security: access control and safety systems;
- video and image processing;
- wired and wireless communications: networking equipment (routers, switches, gateways, firewalls, base station RF transceivers).

	SRAM	**Flash**	**Antifuse**
Manufacturing process	Standard CMOS	Flash	Antifuse
Configuration	Volatile	Non-volatile	Non-volatile
Reprogrammability	Yes	Yes	No
Operable on power-up	No	Yes	Yes
In-system programmable	Yes	Yes	No
Area of basic storage element	High 6 transistors	Moderate 1 transistor	Low 0 transistors
Amount of logic resources	High	Medium	Low

Table 2.1: Comparison of different FPGA technologies [23].

Table 2.1 compares different FPGA technologies [23]. Although the SRAM cell employs up to 6 transistors, the utilization of modern CMOS manufacturing process with the smallest geometries allow SRAM-based FPGAs to offer the highest density of logic resources and functionally predefined components. Abundance of logic resources, reconfigurability, and high performance make them a compelling solution for parallel data acquisition and processing systems for high-energy physics experiments. However, very



(a) C-Cell

(b) R-Cell

(c) SuperCluster

Figure 2.7: RTAX C-Cell (a), R-Cell (b) and SuperCluster (c) [35].

often electronic systems at those experiments must operate in a hostile, radiation environment. Sections 2.2 and 2.4 describe radiation-induced effects on FPGA devices and their operation in different radiation environments.

2.2 Radiation-induced effects

Integrated circuits operating in hostile radiation environments are susceptible to effects caused by interaction of ionizing and non-ionizing particles with their silicon lattice. There are two main categories of radiation-induced effects for electronic components: cumulative effects and Single Event Effects (SEEs) [36–41]. Cumulative effects lead to a progressive degradation of the component electrical parameters, while SEEs include a variety of different types of events induced by a single, energetic particle.

2.2.1 Cumulative effects

Cumulative effects include relatively stable, long-term changes in the electrical characteristics of electronic components, which are caused by ionization and interaction with atoms of the irradiated material. Usually the changes are irreversible and eventually lead to a functional failure. Cumulative effects can be further classified into Total Ionising Dose (TID) and Displacement Damage, also known as Total Non-Ionizing Dose (TNID) [39].

Total Ionizing Dose

Total Ionising Dose (TID) effects result from the energy deposited through ionization by particles such as electrons, protons, or heavy ions [42]. Ionizing radiation creates in material electron-hole pairs. Holes are transported and trapped in the bulk of the dielectric or in its interface layers. This localized charge buildup changes device electric parameters like flatband or threshold voltage, leakage current, or timing skew [39, 43]. Fig. 2.8 illustrates charge buildup changes due to the irradiation. As the holes' mobility is reduced compared to electrons [44], they tend to accumulate in the gate oxide, which gradually increases its overall positive charge. It results in the progressive shift of the threshold voltage. At first, TID effects manifest themselves as the degradation of device electrical parameters and can result in a functional failure [45]. The unit of TID is gray

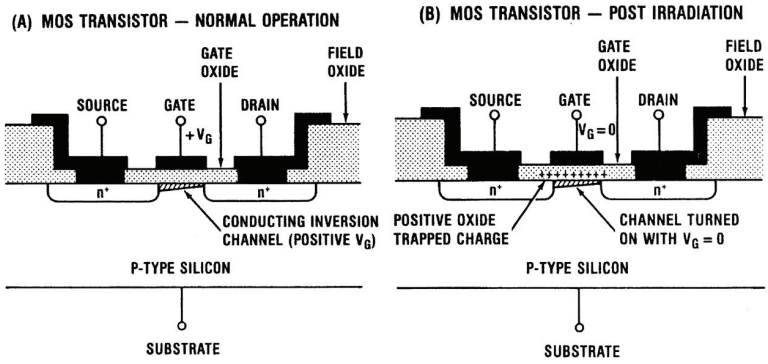


Figure 2.8: Radiation-induced charging of gate oxide in N-channel MOSFET: (a) normal operation and (b) post-irradiation [43].

[Gy], and another non-SI unit often used in the field of radiation hardness community is rad [rad] (1 Gy = 100 rad).

Total Non-Ionizing Dose

Total Non-Ionizing Dose effects are induced by non-ionizing transfer of energy. Particles penetrating a semiconductor crystal lattice displace the atoms from their original positions. The resulting empty spaces and surplus atoms in the crystal lattice eventually lead to stable and electrically active defects [46]. Usually, ASICs and FPGAs are not susceptible to TNID [39].

2.2.2 Single Event Effects

Single Event Effects (SEEs) are induced by the passage of charged particles through a device or a sensitive region of a microcircuit (Fig. 2.9). They can be generated either by

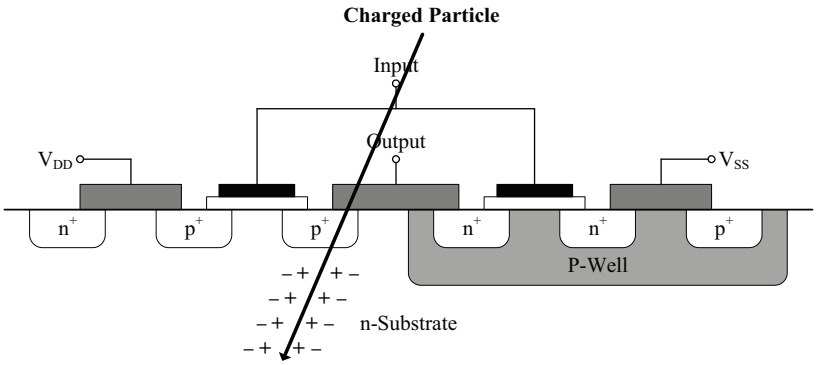


Figure 2.9: Single Event Upset in FPGA [47].

direct ionization or secondary particles issued from nuclear reactions or elastic collisions, either with charged or electrically neutral primary particles like neutrons [39]. SEEs can be classified into two main categories: soft recoverable errors or hard non-recoverable errors [48]. Fig. 2.10 presents different types of SEEs. Comprehensive definitions of soft errors are presented in the JEDEC Standard No. 89A [49]. Modern FPGA devices are in general sensitive to Single Event Transient, Single Event Upset, Single Event Functional Interrupt, and Single Event Latch-Up. The following paragraphs describe the most relevant types of SEEs for modern FPGAs operating in a radiation environment similar to the one of the ALICE experiment.

Single Event Transient

A Single Event Transient (SET) is a temporary voltage or current spike at a node of an integrated circuit, which is induced by a single charged particle ionizing a semiconductor material while traversing through or next to an electrically sensitive junction. If the width and amplitude of the pulse is adequate, the glitch can propagate through the circuit [50]. It becomes a Single Event Upset if latched into a storage cell when arriving at data input during clock edge. An SET is a non-destructive and recoverable error, which may develop into a Single Event Upset or Single Event Functional Interrupt [51].

Single Event Upset

A Single Event Upset (SEU) is a change of state of a storage element like: flip-flop, latch, or SRAM memory cell. A change of state in an SRAM memory cell is illustrated in

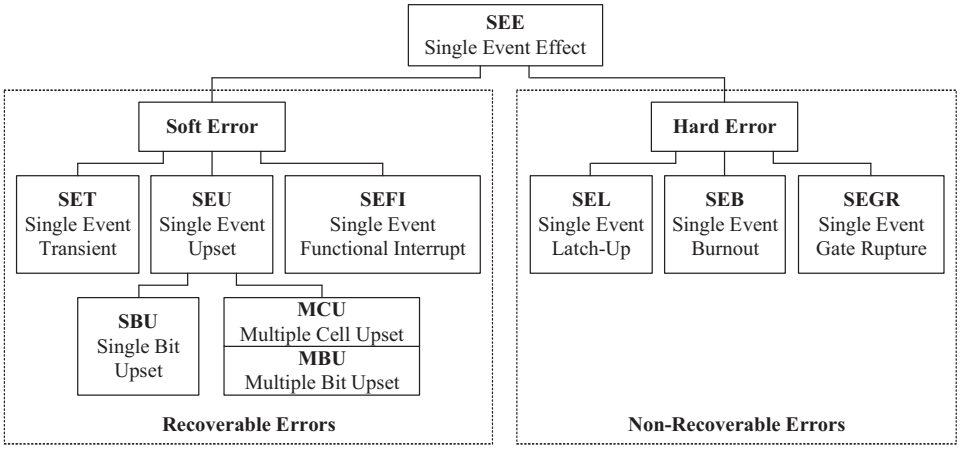


Figure 2.10: Types of Single Event Effects [48].

Fig. 2.11. The studies in [36, 37, 52, 53] describe in detail the SEU mechanism. An SEU

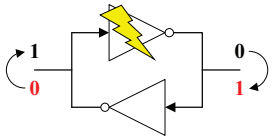


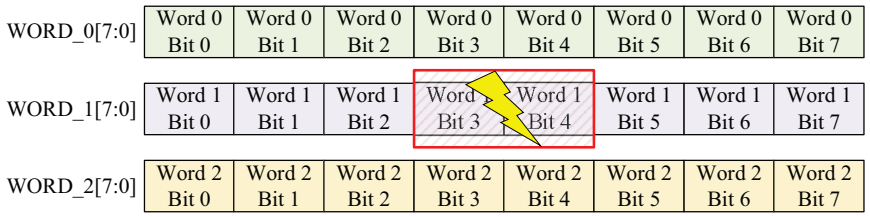
Figure 2.11: SEU in SRAM memory cell.

is a non-destructive and recoverable error, which can lead to a Single Event Functional Interrupt. The change of state of only one bit is called a Single Bit Upset (SBU). Whereas, if two or more bits within the same memory word are altered (Fig. 2.12a), the event is classified as a Multiple Bit Upset (MBU). Furthermore, if two or more bits are flipped within two adjacent memory words (Fig. 2.12b), then it is called a Multiple Cell Upset (MCU).

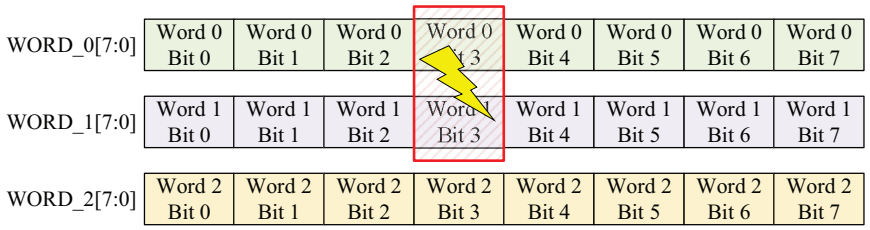
Single Event Functional Interrupt

A Single Event Functional Interrupt (SEFI) is a non-destructive error that results in the interference of normal operation of a complex digital device or system.

In the case of FPGA devices, a SEFI is often associated with an upset in a control bit or register [49, 54]. It is usually related to a failure in the embedded circuits of memory configuration or re-configuration, or power-on-reset [47]. To restore correct operation after a SEFI, a full reconfiguration or power-up cycle is required.



(a) Multiple Bit Upset



(b) Multiple Cell Upset

Figure 2.12: Visualization of Multiple Bit Upset (a) and Multiple Cell Upset (b) [39].

Single Event Latch-Up

A Single Event Latch-Up (SEL) is a non-recoverable hard error, which results from triggering parasitic P-N-P-N or N-P-N-P structures widely existing in CMOS circuits [39, 41]. When triggered, the structure remains in a conductive, low-resistance state, which results in a high current flow. If not turned off quickly enough, the device can be permanently damaged due to the resulting thermal runaway. Usually, a SEL can be detected by a sudden increase in current draw.

2.3 Radiation environments

Due to the advantages that modern FPGA devices offer, they are widely used in many different applications, as explained in Section 2.1.3. Various environments in which those devices are incorporated also include radiation environments, in particular:

- particle accelerators and high-energy physics experiments;
- space environment: Low Earth Orbit, Geostationary Orbit, and deep space;
- high-altitude environment;
- nuclear plants and nuclear research facilities;
- nuclear medicine and imaging apparatuses;
- weapons and military applications.

Space radiation environment

Because of the constant progress in space exploration and high-altitude Earth observation, the space radiation environment has become a field of intense research over recent decades [55–58]. The natural radiation sources in space can be grouped into two main categories:

- charged particles (protons, electrons, and heavy ions) trapped by the planetary magnetic field inside the inner and outer belts surrounding the planet (Van Allen belts for Earth);
- transient particles generated by solar events (solar flares, solar wind, coronal mass ejections), or coming from outside the solar system, called galactic events (galactic cosmic rays).

Trapped radiations show a predictable behavior, while solar events are mostly random, even if there is a strong correlation to the solar cycle. Galactic events also have random distribution, including unpredictable high-energy events. They are however many orders of magnitude less frequent than solar events [59].

High-energy accelerators radiation environment

The radiation environment of high-energy hadron accelerators is different than the one found in space [60, 61]. There are three main sources of radiation at high-energy accelerators:

- the particle collisions at experimental areas and the residual collision products;
- beam interception and collimation, which is performed to avoid distributed losses in other critical machine locations (e.g. in case of the LHC protecting the superconducting magnets so that they do not heat up and go out of superconducting state);
- beam interactions with residual gas inside the beam pipe and the X-ray radiation produced at beam bendings.

The resulting radiation comprises a wide range of particles and energies, and it is therefore called a mixed-field radiation environment. It is composed of charged and neutral hadrons (protons, pions, kaons and neutrons), photons, electrons, positrons and muons [20, 58, 60, 62]. In mixed-field radiation environments of high-energy accelerators SEEs are usually induced by [60, 63]:

- indirect energy deposition events from nuclear interactions between radiation environment hadrons of energy greater than 20 MeV and components nuclei in their sensitive volumes;
- neutrons of energies from 0.2 to 20 MeV, which may generate high-LET secondaries

through elastic and inelastic interactions with nuclei in or in close proximity of device sensitive volume.

Fig. 2.13 presents parameters of mixed-field radiation environment at various sites at the CERN accelerator complex. The LHC machine electronics is installed in protected and shielded locations, or in the tunnel. Depending on the position it has to withstand the annual High-Energy Hadron (HEH) fluence from 10^6 to $10^{12}\,\mathrm{cm}^{-2}$.

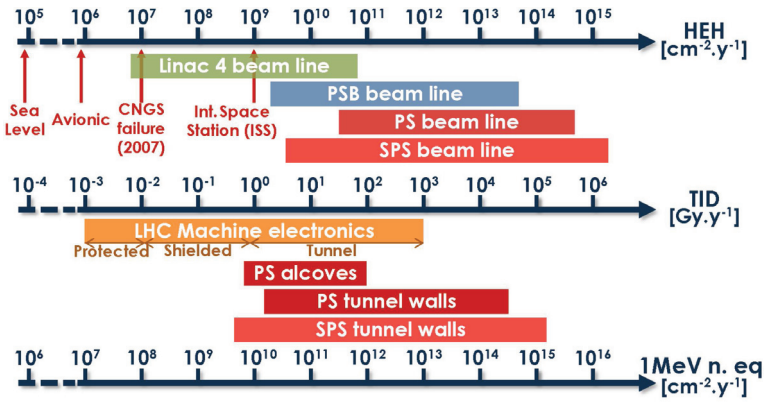


Figure 2.13: TID, HEH flux and 1 MeV n_{eq} fluence at different sites at CERN accelerator complex compared to aerospace environments [62].

Terrestrial radiation environment

Although the Earth's magnetic field provides a natural radiation shield, some solar particles (high-energy protons, products of coronal mass ejections) and high-energy cosmic rays enter upper atmosphere, where they interact with atoms of nitrogen and oxygen [58, 64–67]. These interactions create a shower of secondary particles that further interact with the atmosphere particles at lower altitudes. Protons, electrons, neutrons, heavy ions, muons, and pions are generated; some of them are long-lived and find their way down to the Earth's surface, as shown in Fig. 2.14. Particles reaching the Earth's surface include high-energy neutrons and low altitude-dependent percentage of protons. As already described in Section 2.2, high-energy neutrons may indirectly interact with sensitive regions of electronic active components and create alpha particles, or other charged nuclei that generate electron-hole pairs. If the charge is high enough, it may lead to an SEU.

Electronic components are evaluated for susceptibility to neutrons via accelerated beam testing [64, 68–71] or via exposition to natural radiation sources like at the Rosetta Experiment [72]. Another concern is alpha particles emitted by isotopes present in plastic molding compounds used in semiconductor packaging. Package-originated effects of alpha

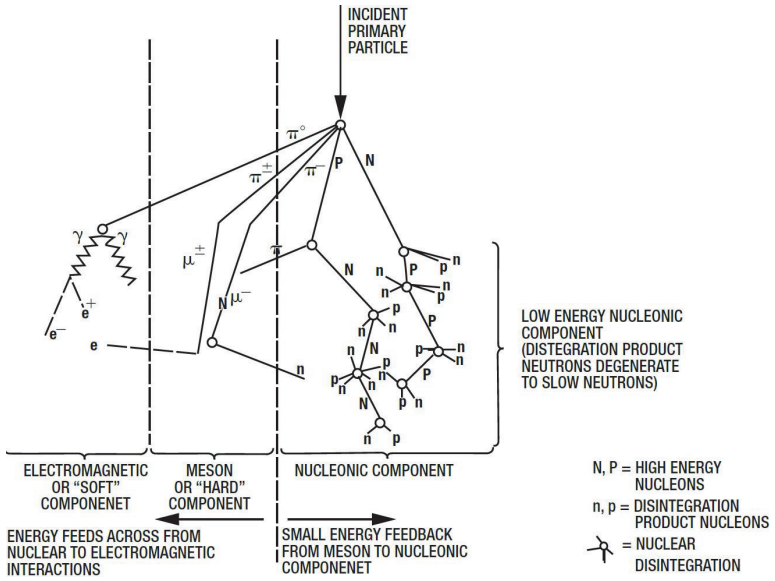


Figure 2.14: Schematic diagram of Cosmic Ray Shower [64].

particles are impossible to avoid because these materials are in close proximity to the semiconductor die [73].

Parameters describing radiation environment

For the purpose of this dissertation, the following radiation environment parameters are described, as they are later intensively used while describing experimental testing with particle beams:

- particle fluence Φ is a quantity describing how many particles ($\mathrm{d}n$) passed through a given area $\mathrm{d}a$:

$$\Phi = \frac{\mathrm{d}n}{\mathrm{d}a} \ [\mathrm{cm}^{-2}] \tag{2.1}$$

- particle flux F is a fluence per time unit $\mathrm{d}t$:

$$F = \frac{\mathrm{d}\Phi}{\mathrm{d}t} \ [\mathrm{cm}^{-2}\,\mathrm{s}^{-1}] \tag{2.2}$$

2.4 SRAM-based FPGA operation in radiation environment

FPGA devices are employed in various electronic systems because of their numerous advantages like short development time of a logic circuit, re-configurability, and abundance of logic resources and functionally predefined components. Although new generations of SRAM-based FPGA devices show a gradual decrease in susceptibility to ionizing and non-ionizing radiation [74,75], their operation can still be compromised by such SEE phenomena like [76,77]: SET, SEU, SEFI, or SEL. SEUs in configuration memory (CRAM) can modify the routing, logic, clocking, or other aspects of a user design [78–81]. Fig. 2.15 shows how a logic function f_{L1} that is programmed by a configurable LUT-mask gets affected by an SEU, and in consequence is transformed into f_{L2}.

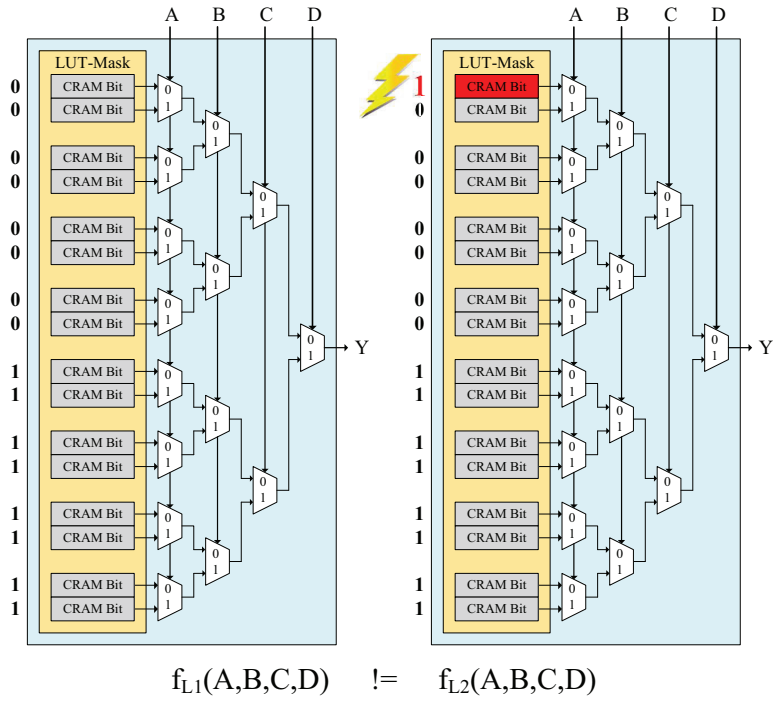


Figure 2.15: SEU affecting LUT-Mask content in SRAM-based FPGA.

FPGA devices based on different switch technologies have been successfully deployed in various radiation environments [82,83]. There is a well documented history of FPGAs deployment in space systems [84]. As in many other missions, SRAM-based FPGAs were utilized in Mars Exploration Rover's Spirit and Opportunity landers [38]. Also, SRAM-based FPGAs were incorporated in the numerous satellite instruments [84,85].

FPGA devices are also widely employed in data readout systems in HEP experiments. However, so far FPGA-based systems were only located in radiation-free areas, while front-end systems operating in a radiation environment were based on custom-designed ASICs. To the author's knowledge, the ALICE Time-Projection Chamber Readout was the first large-scale FPGA-based system to continuously operate in a radiation environment. The system employs 216 FPGA-based Readout Control Unit (RCUs). The first version of the RCU (named RCU1) utilized SRAM-based Xilinx Virtex-II [86]. Then, it was replaced by the RCU2 employing flash-based System-on-Chip Microsemi SmartFusion2 [87].

Tab. 2.2 lists parameters (TID, HEH flux, 1 MeV n_{eq} fluence) of various radiation environments in which existing or planned data readout systems operate. To the author's knowledge within all of the Readout Systems listed in the Tab. 2.2, the development of the ALICE ITS Readout is the most advanced, and in the case of the other Readout Systems the final decision, about which FPGAs (SRAM-based, flash-based, antifuse) will be employed, has still not been made. The ALICE ITS Readout System will be the first one to continuously operate in the HEH flux 5 times higher than the already existing applications. Tab. 2.3 presents the resulting mean time between SEUs in CRAM of SRAM-based FPGAs operating at different locations. The data for the Readout Systems other than the ALICE ITS are shown only as examples so that a reader can visualize how different HEH fluxes impact the mean time between SEUs. The results are presented for the Xilinx XCKU060 which will be finally used in the ITS Readout. Also, reference data for the Xilinx XC7K325T are shown, which was employed as the main processing unit in the prototype readout electronics for the ITS presented in Chapter 6. In the case of the ALICE ITS Readout, the mean time between SEUs for the single Xilinx XCKU060 equals 2 611.2 s. As already described in Section 1.4, ALICE ITS Readout consists of 192 Readout Units, each utilizing Xilinx XCKU060. Tab. 2.4 presents mean time between SEUs in CRAM of 192 FPGAs. On average in the entire ALICE ITS Readout System there will be an SEU in the CRAM of the SRAM FPGAs every 13.60 s.

Thus, as every SEU might potentially lead to a logic design functional error, radiation mitigation methods must be employed to provide a reliable operation of the final ALICE ITS Readout System. Radiation mitigation methods will be discussed and experimentally evaluated in the following part of this dissertation.

Readout System	TID (krad)	HEH flux[a] ($cm^{-2}\,s^{-1}$)	1 MeV n_{eq} fluence (cm^{-2})	Reference
Existing				
ALICE TPC	0.3 [b]	$2 \cdot 10^2$	10^{11}	[88]
Planned				
LHCb Outer Tracker	30	$4.3 \cdot 10^2$	$4 \cdot 10^{12}$	[89]
CMS HCAL FEE [c]	10	$8.19 \cdot 10^2$	$2 \cdot 10^{12}$	[90]
ALICE ITS	**10**	$\mathbf{10^3}$	$\mathbf{1.6 \cdot 10^{11}}$	[21]
ATLAS Liquid Argon (LAr) Calorimeter	2.07 [b,d] 13.8 [b,e]	$6.4 \cdot 10^3$ [b,d] $5.2 \cdot 10^3$ [b,e]	$1.04 \cdot 10^{12}$ [b,d] $7.14 \cdot 10^{12}$ [b,e]	[71]
LHCb RICH sub-detector	200	10^4 [f]	$3 \cdot 10^{12}$	[91–93]
CBM ToF	0.1 to 10 [g]	10^4	10^{10} to 10^{11} [g]	[94]

[a] $E_{kin} > 20\,MeV$ (kinetic energy)
[b] Integrated over 10 years
[c] Front-end electronics
[d] Endcap electronics
[e] Barrel electronics
[f] Calculated for 5 year operation
[g] Integrated over 1 year

Table 2.2: Comparison of different radiation environments in which FPGA-based readout systems operate.

Readout System	Mean time between SEUs	
	XC7K325T (s)	**XCKU060** (s)
Existing		
ALICE TPC	14 720.9	13 056.0
Planned		
LHCb Outer Tracker	6 846.9	6 072.6
CMS HCAL FEE [a]	3 594.9	3 188.3
ALICE ITS	2 944.2	**2 611.2**
ATLAS Liquid Argon (LAr) Calorimeter	4 600.3 [b] 566.2 [c]	4 080.0 [b] 502.2 [c]
LHCb RICH sub-detector	294.4	261.1
CBM ToF	294.4	261.1

[a] Front-end electronics

[b] Endcap electronics

[c] Barrel electronics

Table 2.3: Mean time between SEUs in CRAM of SRAM-based FPGAs operating in different readout systems locations in HEP experiments. Calculations are depicted in Appendix A.

Readout System	Mean time between SEUs	
	XC7K325T (s)	**XCKU060** (s)
ALICE ITS	15.33	**13.60**

Table 2.4: Mean time between SEUs in CRAM of 192 SRAM-based FPGAs operating within ITS Readout System. Calculations are depicted in Appendix A.

2.5 Validation methods of SRAM-based FPGAs

Systems employing SRAM-based FPGAs that are intended to operate in radiation environments must be first validated to see if they meet the imposed constraints in terms of reliability and availability in the target hostile environment. The susceptibility to radiation of an SRAM-based FPGA must be characterized. Also, Single Event Latch-Up (SEL) occurrences in the target radiation environment and a TID limit must be measured [39, 41, 95–97]. Furthermore, SEU cross-sections of configuration (CRAM) and block (BRAM) memories must be calculated. Finally, after the logic design is implemented in the target SRAM-based FPGA, its susceptibility to radiation must be investigated. As not every SEU leads to a design functional error, a detailed experimental validation is required to characterize the anticipated design functional error rate. To the author's knowledge a unified and complex methodology for a design functional error rate estimation has not been developed yet.

Calculation of an SEU cross-section requires counting the number of upsets that occur while exposing a Device Under Test (DUT) to a given number of ionizing particles [98,99]. A DUT can be a memory, an FPGA logic design, or a device. An SEU cross-section σ_{SEU} unit is in respect to area, and depending on a DUT it is expressed in $\text{cm}^2\,\text{bit}^{-1}$, $\text{cm}^2\,\text{design}^{-1}$, or $\text{cm}^2\,\text{device}^{-1}$. In order to calculate the σ_{SEU} the number of error events during a DUT irradiation must be counted and divided by the number of ionizing particles per unit area (fluence) of exposure. Eq. 2.3 shows the simplest form of the equation used for calculating a σ_{SEU} (where $N_{\text{upset\_events}}$ is the number of error events during irradiation, Φ is the fluence, F is the average flux, and T is the time of exposure to the beam).

$$\sigma_{\text{SEU}} = \frac{N_{\text{upset\_events}}}{\Phi} = \frac{N_{\text{upset\_events}}}{FT} \left[\text{cm}^2\right] \tag{2.3}$$

Fig. 2.16 presents the available SRAM-based FPGA validation methods. One can distinguish analytical and experimental validation methods.

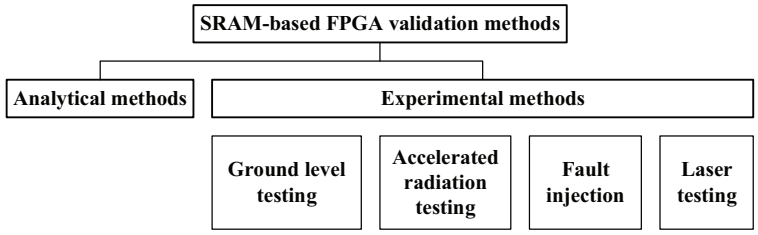


Figure 2.16: SRAM-based FPGA validation methods [39, 95].

Analytical methods are based on the analysis of a topology of a logic circuit implemented in the SRAM-based FPGA and identifying the paths sensitive to soft errors. The

analysis is coupled with the target radiation environment parameters and the anticipated failure rate is estimated [39, 100].

Experimental methods include ground level testing, accelerated radiation testing, fault injection, and laser testing. Ground level testing is based on the long-term exposure to natural radiation which can reveal valuable information about the sensitivity of a device to TID and SEE effects. An example of the ground level testing is the Rosetta experiment by Xilinx [72]. Several hundreds of SRAM-based FPGAs are installed at different altitudes and in different locations around the world, and their configuration memories are monitored to detect upsets.

Accelerated radiation testing employs artificial radiation environments with different kind of particles which are usually generated in facilities such as synchrotrons, cyclotrons, or linear accelerators [101–104]. Particle fluxes used during accelerated radiation testing are several orders of magnitude higher than in the natural environment, which allows for collecting statistically meaningful data in much shorter time with respect to ground level testing [39, 41, 95, 96]. Accelerated radiation testing is often used in SRAM-based FPGAs characterization to measure [95]:

- SEU cross-section of the configuration memory (CRAM),
- SEU cross-section of the embedded Block RAM (BRAM),
- SET cross-section of events affecting global clock and reset lines, or any other components susceptible to SETs,
- SEFI cross-section of events disturbing the operation of the functionally predefined components such as configuration control logic or power-on reset circuit.

What is more, accelerated radiation testing is used to experimentally measure the parameters important from the point of view of deploying the SRAM-based FPGA device in a radiation environment:

- TID limit within which the device does not show any degradation of parameters,
- SEL threshold that specifies the minimum LET value that the particle must have to induce a SEL.

Another experimental method of SRAM-based FPGA evaluation is fault injection [39, 41, 95, 96]. This method relies on an injection of faults into the FPGA configuration memory (CRAM) where bit flips emulate SEUs. Two approaches are possible within this method:

- faults are injected to a bitfile offline, and then an FPGA is programmed with the corrupted data,
- an FPGA is programmed with a golden bitfile and faults are injected to the CRAM while the FPGA operates.

Multiple fault injection test systems were developed [105–108]. Also, more advanced tools, such as JTAG Configuration Manager [109, 110] or Flipper [111], which allow for

the execution of various operations on the CRAM - like fault injection or scrubbing, are available. There is also a documented history of fault injection methodology employment during research and development of data readout systems for high-energy physics detectors [112, 113] and space instrumentation [114].

The last experimental validation method is laser testing. It is often used in conjunction with an accelerated radiation testing. It allows for the induction of SEUs with great precision so that sensitive locations on the die can be mapped and investigated [115, 116].

Chapter 3

Motivation and hypothesis

3.1 Motivation

The upgraded Inner Tracking System (ITS) Readout, as presented in Section 1.5, will continuously operate in the radiation environment resulting from particle collisions at the interaction point of the ALICE experiment. The employment of reconfigurable FPGA devices, described in detail in Section 2.1, facilitates the realization of the Readout System. Although dedicated radiation-hard by design FPGAs are available on the market, they cannot be utilized because they are either limited in resources or too expensive. Conversely, commercial-grade non-radiation-hardened SRAM-based FPGAs offer a large amount of logic resources and required high-speed data transceivers, and their price is low enough to be used in large quantity. Although they are susceptible to ionizing and non-ionizing radiation, there exist radiation mitigation methods which can increase the radiation resiliency of the implemented design so that operational conditions in terms of reliability and availability are met.

As explained in detail in Section 2.2, SRAM-based FPGAs deployed in radiation environments experience Single Event Effects, which influence operation of the implemented digital designs. According to Table 2.2, the ITS Readout System will operate in the HEH flux 5 times higher than any other readout system deployed earlier in a high-energy physics experiment. Within the designed Readout System, as calculated in Section 2.4, on average there will be one SEU in CRAM of a single SRAM-based FPGA every 2611.2 s, which corresponds to an SEU in the entire system every 13.6 s. The resulting SEU rate, which may lead to a functional failure of the implemented digital circuit, is not negligible, so in order to provide stable system operation during a data taking run, radiation mitigation methods must be incorporated into the final FPGA design.

The literature review and the experimental results in Chapter 7 show that a correct implementation of radiation mitigation methods is not a straightforward task. If incor-

porated incorrectly, they may lead to:

- unnecessary resource utilization,
- timing degradation,
- increase of power consumption,
- increase of radiation susceptibility.

Thus, in order to correctly choose and incorporate radiation mitigation techniques into the final FPGA design, it becomes obligatory to characterize and quantitatively compare them before the final implementation.

3.2 Hypothesis and goals

Taking into account the motivation explained in Section 3.1, the following hypothesis is formulated:

> It is possible to design and deploy the readout system employing commercial, non-radiation-hardened SRAM-based FPGAs for the upgraded ALICE Inner Tracking System, provided that FPGA radiation mitigation methods are first characterized and then incorporated into the final FPGA design.

To support the above-mentioned hypothesis and demonstrate the practical utility of the proposed solution, two partial objectives are established:

- elaboration of a methodology allowing for the estimation of functional error rates of SRAM-based FPGA design modules;
- evaluation and quantitative comparison of main SRAM-based FPGA mitigation methods and proposition of a guideline for the final FPGA design implementation.

The successful realization of the partial objectives will confirm the hypothesis and demonstrate the advantages of the developed methodology. The presented dissertation is of a highly practical nature and strongly contributes to the subject of utilization of SRAM-based FPGA devices in radiation environments.

Chapter 4

Design functional error rate estimation methodology

4.1 Methodology

As explained in Chapter 2, SRAM-based FPGAs operating in radiation environment experience Single Event Effects. However, not every Single Event Upset (SEU) occurrence in either CRAM or BRAM leads to a digital circuit functional error. Thus, if an SRAM-based FPGA is intended to operate in a radiation environment, it becomes essential to estimate the susceptibility of the implemented digital circuit to SEUs, so that system designers can verify its performance and estimate its final reliability. The final FPGA design radiation susceptibility can only be estimated after the digital circuit is entirely implemented. However, testing and characterizing separate design modules directs and shortens the main development. The literature review shows that a generic methodology and testing framework for functional error rate estimation of SRAM-based FPGA design modules is unavailable. Thus, a custom methodology, shown in Fig. 4.1, has been proposed to address the above-mentioned issue. The methodology consists of the following elements:

- *Radiation environment parameters (particle flux and fluence, TID)* where an SRAM-based FPGA device will operate, are explained in detail in Section 2.3;
- *SRAM-based FPGA radiation parameters and qualification* describe device performance (CRAM, BRAM, SEFI, and SEL cross-sections) and general device qualification for operation in radiation environment;
- *SRAM-based FPGA* is a target device which is also used for the Emulator implementation and experimental testing;
- *Emulator* is an FPGA design where the tested design module is implemented. It is described in detail in Section 4.2;
- *FPGA design module* and *SRAM-based FPGA mitigation methods* is the evaluated

module with or without mitigation methods employed;

- *FPGA design module functional error rate* is the experimentally-derived functional error rate.

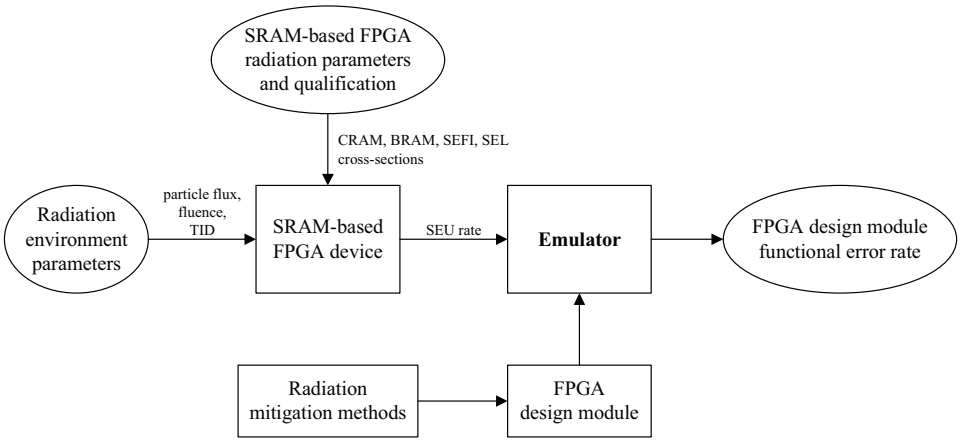


Figure 4.1: Design module functional error rate estimation methodology for SRAM-based FPGAs.

The generic methodology flow is presented in Fig. 4.2. The starting point is a definition of the radiation environment parameters, in which an SRAM-based FPGA device will operate. Secondly, it is required to calculate the SEU rate resulting from the particle flux and experimentally measured CRAM and BRAM cross-sections of the target SRAM-based FPGA. The SEU rate gives an overview of how many faults on average will occur in both CRAM and BRAM over a given period of time. It is also an important parameter while experimentally testing via fault injection or beam irradiation. Also, SRAM-based FPGA device has to be verified in terms of SEFI rate and Single Event Latch-Up (SEL) occurrence before being deployed in the intended radiation environment. Both SEL-free operation and TID tolerance within the specified LET and TID range must be investigated. Then, the user has to estimate the minimum Mean Time Between Failures of the FPGA design module, which will be required to define the testing parameters. The next step is an integration of the investigated module into the Emulator design, which is the developed testing framework. Later, a set of experimental tests is carried out. During a single test execution, an operation of the implemented digital circuit is monitored, while particle irradiation or fault injection dynamically modifies the content of both CRAM and BRAM. The last step is an analysis of the received error counters data and estimation of the resulting functional error rate of a tested design module. The practical utilization of the developed methodology is presented in Chapter 7 in an example of the basic building blocks of the SRAM-based FPGA design for the ALICE Inner Tracking System Readout.

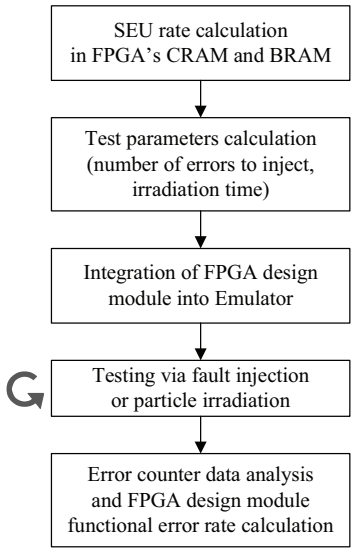


Figure 4.2: Design functional error rate estimation methodology flow.

4.2 Emulator

The Emulator, presented in Fig. 4.3, is an FPGA design which is used for FPGA design modules testing. It has a generic and modular architecture to facilitate implementation of various design modules and instantiation in different SRAM-based FPGAs. It contains many identical modules, called lanes, and an error readout module. The parameterizable error readout module consists of n-bit error counters. Their size is set depending on the investigated FPGA design module and the selected testing parameters. As many lanes as possible are instantiated to maximize the available logic resources utilization in the FPGA. Each lane consists of a triplicated pattern generator, a chain of the Logic Test Structures (LTSs), and a triplicated pattern checker. The chain of the LTSs may comprise one or more replicated test structures. The chain with more than only one LTS is instantiated to increase the ratio between the utilized resources by the investigated test structures and both the pattern generator and pattern checker. The higher the ratio the greater the probability that the investigated test structures are influenced by particle irradiation or fault injection rather than the circuitry used for carrying out the testing.

After the reset signal is deasserted, the pattern generator starts outputting test vectors which are pushed through the chain of the LTSs. After the circuit initialization, the pattern checker either verifies the data at the output from the chain of the LTSs or compares them to the reference data outputted by the pattern generator via an optional connection shown in Fig. 4.3. When the pattern checker finds an error in the operation of

the chain of the LTSs, or there is a discrepancy while comparing by the pattern checker, a pattern error signal is asserted on the *PAT_ERR* output. Furthermore, when a discrepancy is found during voting in the triplicated pattern generator, or the internal voter of the triplicated pattern checker, a lane warning signal is asserted on the *LANE_WARN* output. Both signals increment corresponding error counters in the error readout module. During particle irradiation or fault injection testing, the values stored in the error counters are periodically read out via a USB interface and saved. Depending on the available hardware platform, other interfaces like UART or Ethernet can be employed to stream out the data. During later data analysis, it is possible to recognize how many lanes were affected by SEUs or distinguish whether the error was in the error readout module. Tests where the error readout module failed due to SEUs or injected faults are excluded from the analysis.

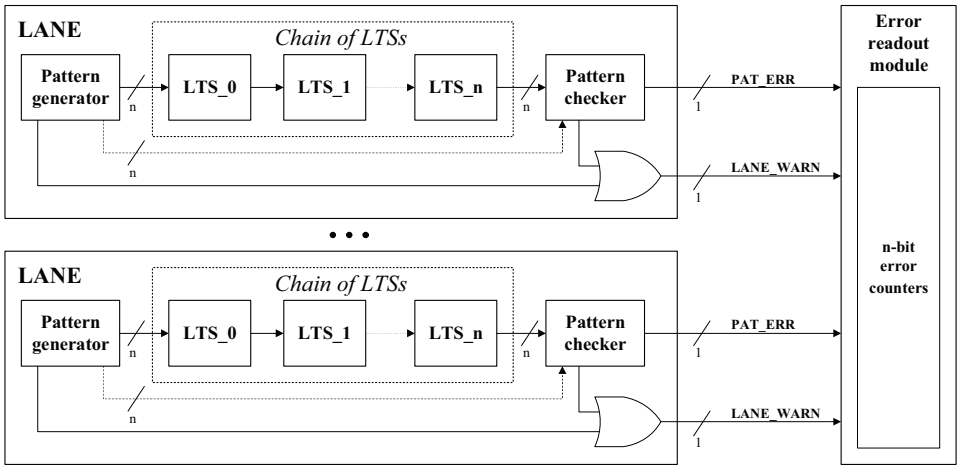


Figure 4.3: Emulator design architecture.

4.3 Testing and functional error rate estimation

After the Emulator containing investigated design module is implemented in the target SRAM-based FPGA, a set of experimental tests is executed. The testing flow is presented in Fig. 4.4. The first step is to define test parameters:

- time of exposure to a particle beam,
- number of faults to inject,
- and number of test runs to carry out.

The number of SEUs induced to the CRAM $N_{\mathrm{ERR\_CB}}$ during a single irradiation test can be estimated using Eq. 4.1 (where σ_{CRAM} is the CRAM cross-section, Φ is the integrated

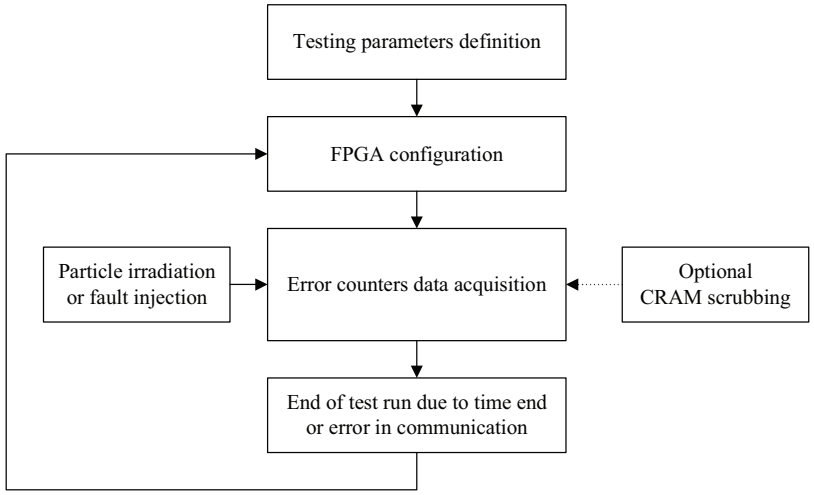


Figure 4.4: Testing execution flow.

beam fluence, F is the beam average flux, T is time of exposure to the particle beam, and N_{CB} is the number of CRAM bits).

$$N_{\mathrm{ERR\_CB}} = \sigma_{\mathrm{CRAM}} \Phi N_{\mathrm{CB}} = \sigma_{\mathrm{CRAM}} F T N_{\mathrm{CB}} \tag{4.1}$$

While the number of SEUs induced to the BRAM $N_{\mathrm{ERR\_BB}}$ during a single irradiation test can be estimated using Eq. 4.2 (where σ_{BRAM} is the BRAM cross-section, Φ is the integrated beam fluence, F is the beam average flux, T is time of exposure to the particle beam, and N_{BB} is the number of BRAM bits).

$$N_{\mathrm{ERR\_BB}} = \sigma_{\mathrm{BRAM}} \Phi N_{\mathrm{BB}} = \sigma_{\mathrm{BRAM}} F T N_{\mathrm{BB}} \tag{4.2}$$

After testing parameters are established, a set of test runs is executed and controlled by the developed software running on a PC. In each run, after the device is configured, the Emulator design starts operation, and SEUs are induced to the CRAM and BRAM by a particle beam or faults are injected to the CRAM. Testing with particle beams and via fault injection is explained in detail in Section 2.5. Each test run is terminated after either:

- a time of exposure to a particle beam expired,
- a specified number of faults was successfully injected,
- or there was an error in the communication between the Emulator and software reading the error counters running on a PC.

Also, while executing a test run it is possible to enable an external mitigation method called *memory scrubbing*, which is explained in detail in Section 5.1.2.

After a specified number of test runs have been successfully carried out, the number of lanes that showed an incorrect operation are checked. While beam testing or fault injection faults are randomly induced to the CRAM and BRAM, it is not possible to exclude the pattern generators, pattern checkers, and the error readout module from the testing. Thus, although those modules are implemented with the TMR, they can also be influenced by the induced SEUs or injected faults, and in consequence, stop operating reliably. In order to correctly estimate the design module functional error rate, the test runs in which modules other that the chain of the LTSs showed incorrect operation must be excluded from further analysis. A custom data analysis algorithm was developed that performs an iterative Poisson probability distribution fit and cuts off the outliers outside the 3 σ range. The lane is considered faulty after its pattern checker asserts the pattern error signal (PAT_ERR) and there are not any errors in the error readout module. A detailed analysis of the error counters behavior allows for the study of the operation of different mitigation methods in the tested design modules.

A cross-section of the lane σ_{L} is calculated using Eq. 4.3 (where N is the number of conducted irradiation or fault injection tests, E is the derived number of faulty lanes in a single test run, N_{L} is the number of lanes in the given Emulator design, Φ_{i} is the fluence, F_{i} is the average flux, and T_{i} is the time of exposure to the beam during a single test).

$$\sigma_{\mathrm{L}} = \frac{\sum_{\mathrm{i}=1}^{\mathrm{N}} E_{\mathrm{i}}}{N_{\mathrm{L}} \sum_{\mathrm{i}=1}^{\mathrm{N}} \Phi_{\mathrm{i}}} = \frac{\sum_{\mathrm{i}=1}^{\mathrm{N}} E_{\mathrm{i}}}{N_{\mathrm{L}} \sum_{\mathrm{i}=1}^{\mathrm{N}} F_{\mathrm{i}} T_{\mathrm{i}}} \left[\mathrm{cm}^2\,\mathrm{lane}^{-1}\right] \tag{4.3}$$

From the lane cross-section, it is possible to derive the lane functional error rate f_{fer} at a given particle flux using Eq. 4.4 (where σ_{L} is the lane cross-section, F is the beam flux).

$$f_{\mathrm{fer}} = \sigma_{\mathrm{L}} F \left[\mathrm{s}^{-1}\right] \tag{4.4}$$

In the Emulator design, it is impossible to distinguish multiple errors occurring in the same lane. To keep the occurrence of this condition negligible, the total number of injected errors must be kept low[1].

[1] The number of actual errors in respect to the number of lanes that demonstrated faulty operation (because of one or more injected errors to that lane) may be underestimated and it can be calculated using the following formula ("m balls in n bins" probability problem): $E = \frac{\ln\left(\frac{N_{\mathrm{L}} - E_{\mathrm{L}}}{N_{\mathrm{L}}}\right)}{\ln\left(1 - \frac{1}{N_{\mathrm{L}}}\right)}$ (where E is the number of actual errors, N_{L} is the number of lanes, and E_{L} is the number of faulty lanes).

4.4 Discussion

In this section, advantages and limitations of the developed testing methodology are discussed. The main novelty brought by this methodology is a step by step approach of estimating a radiation hardness of an FPGA design module. Secondly, the employment of only one FPGA substantially simplifies the experimental testing. What is more, the utilization of the methodology directs and speeds up the main SRAM-based FPGA design development. However, as the final FPGA design is implemented differently depending on the target FPGA device, synthesizer and placer settings, the developed methodology has also some limitations. First of all, for every investigated FPGA module a user has to develop a pattern generator, pattern checker, and the Emulator. However, it is possible to reuse available FPGA design service modules to shorten the development time. Secondly, the derived functional error rate is related to the entire lane structure and corresponding part of the error readout module, not only to the investigated FPGA module. However, as it is shown in Chapter 7, analysis of the cross-sections of the lanes containing chains of the LTSs with different mitigation methods employed allows for the drawing of conclusions about radiation protection effectiveness and guiding the development.

Chapter 5

Single Event Upset mitigation methods

As already discussed, commercial SRAM-based FPGA devices offer an abundance of logic resources and functionally predefined components, which make them suitable candidates for processing units in multi-channel data readout systems in high-energy physics experiments. However, as these systems must often be located close to particle detectors, they operate in radiation environments. As described in Chapter 2, the operation of SRAM-based FPGAs, which do not contain radiation-hard by design circuits, can be compromised by radiation-induced effects. This chapter presents methods and techniques that can be employed to mitigate Single Event Upsets, and in consequence allow for the utilization of SRAM-based FPGAs in systems operating in radiation environments.

5.1 Mitigation methods

An FPGA design without radiation mitigation methods employed, when implemented in an SRAM-based FPGA operating in radiation environment, will have a substantial soft error rate disrupting its correct work and may not satisfy the requirements of some critical applications [98]. However, if radiation mitigation methods based on logical masking and correction circuits are incorporated, then the functional error rate resulting from operation in radiation environment can be decreased, and a commercial SRAM-based FPGA may be a candidate for system use [117, 118]. Various radiation mitigation methods have both advantages and penalties, so the best way to protect digital circuits implemented in SRAM-based FPGAs is to use a combination of complementary mitigation techniques [39]. Experimental results showing benefits brought by employment of different radiation methods are presented and discussed in detail in Chapter 7.

5.1.1 Spatial redundancy

Spatial redundancy is a mitigation method based on replicating sensitive functional design modules and comparing their outputs together in order to detect discrepancies [39]. This mitigation method is not intended to correct soft errors [95], such as SEUs in the CRAMs of the SRAM-based FPGAs. Instead it is used to:

- detect occurrencies of soft errors modifying the expected operation of the circuit,
- mask soft errors and their propagation through the circuit.

Spatial redundancy incorporates additional design module instances with other auxiliary components like comparators or voters. As the mismatch between results outputted by different module replicas is detected by either a comparator or voter, they become a critical part of these architectures.

Dual modular redundancy

The Dual Modular Redundancy (DMR) scheme, presented in Fig. 5.1, can be applied both to combinational and sequential logic. In the DMR two identical design blocks operate in parallel and the comparator detects discrepancies in the outputted results, which are caused by SEEs. As the comparator detects faults but does not correct them, the DMR is often called a fail-stop architecture [39, 119]. When both results are identical, then the comparator does not report any error. Conversely, if the outputted results are different, then the comparator asserts an error signal and it is up to the designer to foresee a circuit response to such an error. The potential solutions are either to mark the result as faulty and continue operation, or stop and repeat.

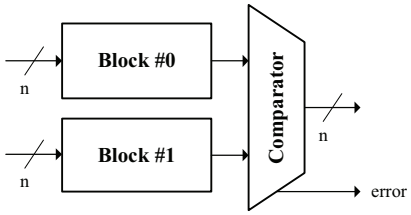


Figure 5.1: Dual Modular Redundancy architecture.

Triple modular redundancy

The Triple Modular Redundancy (TMR) is one of the most popular and most user-applied data path mitigation method [39, 98], that produces low-cost reliability [120]. In the TMR, a base design block is replicated three times and a majority voter votes for the correct output. In commercial-grade FPGAs a user can implement TMR using proper

HDL instantions [121]. Similarly as in the DMR, the TMR does not correct soft errors. Instead it masks them, so that the instantiated circuitry can operate without interruption. The optimal TMR implementation depends on the circuitry that the SRAM-based FPGA implements [122]. The various types of TMR techniques are differentiated by the portions of circuitry that are replicated and where the voters are placed [98, 119, 123–126]. There are several different TMR schemes:

- No-TMR, shown in Fig. 5.2, a basic block is implemented without TMR. No additional logic is added to the design pertaining to SEU mitigation.

Figure 5.2: Basic block implemented without TMR.

- Local-TMR (LTMR), shown in Fig. 5.3, is a scheme where only flip-flops are triplicated and data-paths stay singular. Voters are placed after the flip-flops. Combinational logic paths, clock and reset lines are shared, and they are single sources of failure. With this mitigation scheme only the effects of flip-flop SEUs are reduced, as they are masked. SETs in data-paths of combinational logic can be captured by endpoint flip-flops.

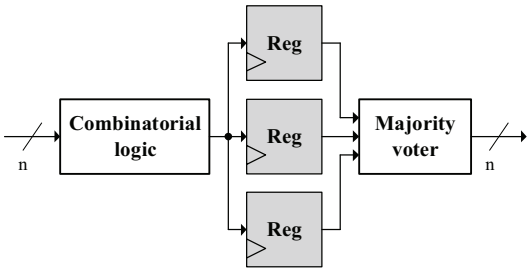


Figure 5.3: Local Triple Modular Redundancy architecture.

- Block-TMR (BTMR), shown in Fig. 5.4, is a scheme where complex functions containing combinational logic or flip-flops are triplicated, and majority voters are placed at the output of the triplets. In the BTMR it is not required that the inputs come from a common source. However, if the I/Os are not fanned out to the replicated blocks from a common source, then voting will be unreliable due to synchronization issues. The outputs of the replicated block are voted and the voter can have an additional warning output that informs the surrounding circuitry that not all the blocks output identical data. The BTMR only provides masking capability and does not correct soft errors. Thus, this technique is only practical for design

modules that can be regularly reset. In this case soft errors (SETs in data-paths and SEUs in flip-flops) are regularly flushed, and the design blocks can be forced to reach a deterministic state.

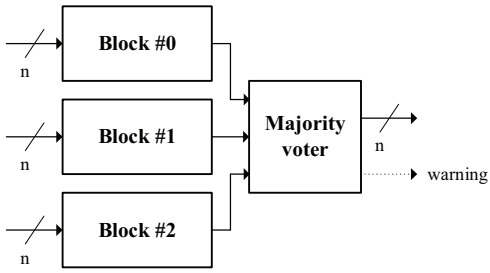


Figure 5.4: Block Triple Modular Redundancy architecture.

- Distributed-TMR (DTMR), shown in Fig. 5.5 and Fig. 5.6, is a scheme where all functional logic is triplicated except for global routes: clocks, resets, and high-fanout enables. This scheme reduces data path upsets, but since the global routes are not mitigated, SETs on global routes can still disrupt the system. In the DTMR voters are placed after the flip-flops.

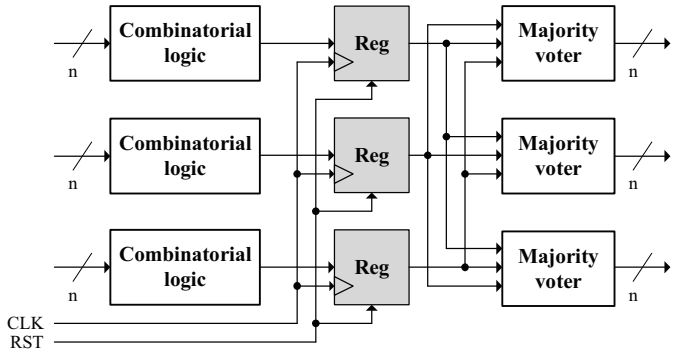


Figure 5.5: Distributed Triple Modular Redundancy architecture.

- Global-TMR (GTMR), shown in Fig. 5.7, is a scheme where the entire design is triplicated including all global routes: clocks, resets, and high-fanout enables. The GTMR reduces data path upsets. However, modern FPGA devices employ additional logic outside of the data-path that cannot be mitigated. Incorporating the GTMR reduces the upset rate, but still has some points of failure.

As the TMR only masks soft errors and does not correct them, the affected circuit replicas may remain faulty or unsynchronized with each other. For some applications, bare TMR implementation does not provide enough mitigation. Thus, to further increase the TMR

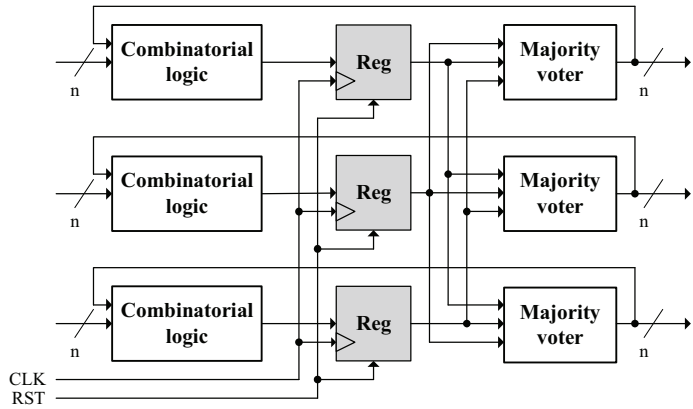


Figure 5.6: Architecture of Distributed Triple Modular Redundancy with feedback paths.

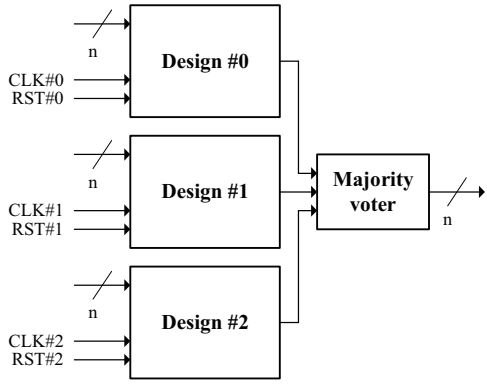


Figure 5.7: Global Triple Modular Redundancy architecture.

performance, it can be implemented with various synchronization mechanisms [78,127] and CRAM reconfiguration [128]. The TMR implementation is not a straightforward task, and its improper insertion can jeopardize the system radiation performance [124]. Placement of the voters can directly affect the radiation susceptibility of the TMR design [129]. Various difficulties with TMR insertion are explained in detail in [96, 130]. In Section 7.2 experimental testing results of different TMR topologies are presented and discussed.

5.1.2 Configuration memory scrubbing

Spatial redundancy methods, described in Section 5.1.1, only mask soft errors and do not correct SEUs in the CRAM of the SRAM-based FPGAs. In order to keep spatial redundancy methods working reliably, they must often be complemented with an additional mitigation method called scrubbing which avoids fault accumulation and restores the cor-

rect CRAM content. Scrubbing is a mitigation method that relies on periodic cycles of simultaneous writing to the CRAM with the intent of correcting bit errors, while the device's functional logic area is still operating [39, 41, 76, 119, 131]. Scrubbing prevents the accumulation of configuration upsets and significantly reduces the probability that two SEUs may overcome TMR. It is possible because modern SRAM-based FPGAs offer a functionality called Partial Reconfiguration [132–135], where part of the design can be independently reconfigured. An example of the configuration interfaces and circuitry inside the Xilinx 7 Series FPGAs is presented in Fig. 5.8. The configuration module manages the configuration memory consisting of frames [136, 137], which are the smallest addressable elements. The configuration memory could be accessed and modified via SelectMAP, JTAG, or ICAP interfaces. As explained in detail in [139], apart from CRAM scrubbing,

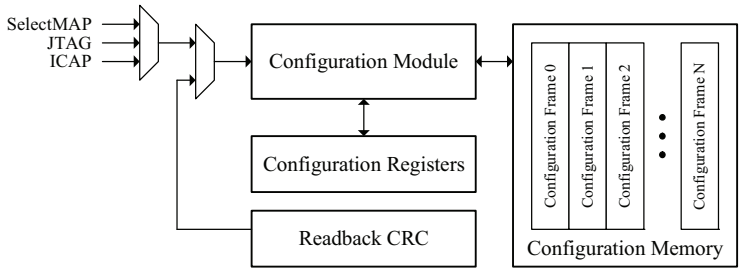


Figure 5.8: Configuration interfaces and circuitry [137, 138].

the content of the BRAM can be also scrubbed.

Scrubbers' classification is presented in Fig. 5.9 and it is explained in detail in [39, 41, 131, 140]. They can be categorized depending on the scrubbing system architecture or

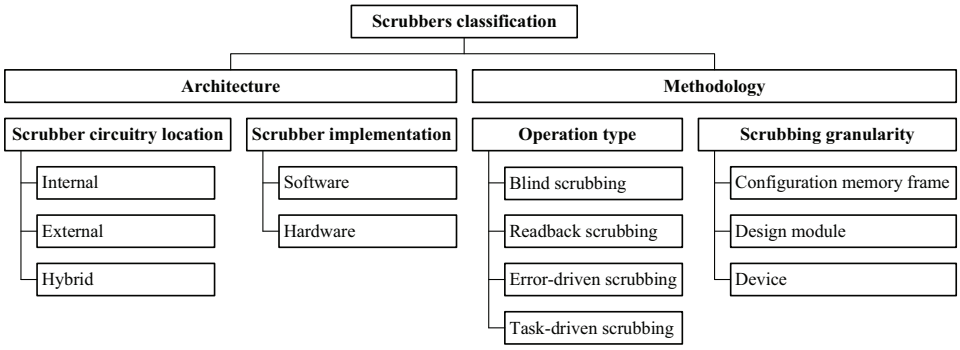


Figure 5.9: Scrubbers classification [140].

adopted scrubbing methodology. Depending on the selected scrubbing architecture, there are internal, external, and hybrid CRAM scrubbers. They can be implemented either in

software and run on a processor, or be instantiated in a programmable logic and operate in an FPGA. However, depending on the operation type that the scrubber executes, there are blind, readback, error and task driven scrubbers. The blind scrubber reads golden configuration data from an external non-volatile memory and writes it back to the CRAM of the SRAM-based FPGA. In this method it is impossible to gather information about how many SEUs were induced to the CRAM. On the other hand, the readback scrubber first reads back the configuration frame data from both the SRAM-based FPGA and the external non-volatile memory. Next, the scrubber controller compares the data using either a direct comparison or Error Correction Codes (ECCs). If a discrepancy is found, the faulty configuration frame of the SRAM-based FPGA is rewritten using golden data stored in the external memory.

While using the readback scrubber, it is possible to count the number of induced SEUs and get valuable information about the radiation environment where the target application operates. Also, it allows for verifying simulated radiation environment parameters with the real ones. Depending on the scrubbing granularity, scrubbers operate on a different sizes of memory blocks. In a single run, they can process either only one configuration frame, one design module, or the full device.

Internal Scrubber

The scrubbing controller in the internal CRAM scrubber is implemented either as an on-chip hard core or is created out of user fabric blocks located inside the target SRAM-based FPGA. The internal scrubber architecture is presented in Fig. 5.10. The scrubbing

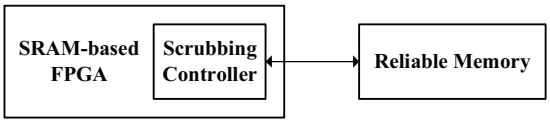


Figure 5.10: Architecture of internal scrubber.

controller is connected to the internal CRAM of the SRAM-based FPGA via a dedicated configuration interface. It can also optionally interface with an external non-volatile memory holding the golden copy of the configuration data. Usually, internal scrubbers use Single Error Correct Double Error Detect (SEC-DED) technique together with the ECC codes embedded in the configuration frames to detect and correct SEUs, and detect MBUs. The configuration frames containing MBUs are rewritten by fetching correct data from the external memory holding the golden bitstream.

An example of the internal scrubber that can be instantiated in modern Xilinx SRAM-based FPGAs is the Soft Error Mitigation Controller which delivers functionality of cor-

recting both SEUs and MBUs [141, 142]. There also exist other implementations of internal scrubbers. One example is the internal scrubber based on a coarse grain TMR FPGA design. Although each TMR branch is implemented in a different FPGA region, it uses the same configuration data. Thus, any configuration frame of the TMR branch can be repaired using the data from the other identical branches [140, 143, 144]. Also, as demonstrated in [145], the internal scrubber can be implemented in a system that requires Partial Reconfiguration functionality.

External Scrubber

The scrubbing controller in the external scrubber is located outside the target SRAM-based FPGA. External scrubber architecture is presented in Fig. 5.11. External scrubbers

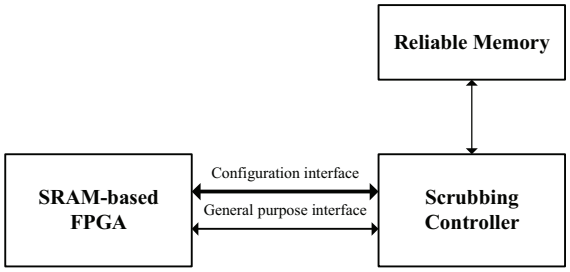


Figure 5.11: Architecture of external scrubber.

are created out of user fabric blocks located inside the antifuse or flash-based FPGAs [119], or are implemented as standalone radiation-hardened ASICs [146]. As demonstrated, the external scrubbers have better performance than the internal ones [147, 148].

External scrubbers usually operate in either blind or readback mode. An example of blind scrubber implementation in a high-energy physics experiment is the CBM Readout Controller board [149]. It is equipped with the Xilinx Virtex-4 SRAM-based FPGA [150], which operates as the core data processing device. It is continuously scrubbed by a flash-based Microsemi ProASIC3 A3P125 FPGA [151], which reads golden configuration data from an on-board, non-volatile flash memory. Another example of blind scrubber implementation is the Readout Board of the ALICE Inner Tracking System detector. Its architecture is presented in Fig. 5.12. The Xilinx Ultrascale XCKU060 [152] SRAM-based FPGA is continuously scrubbed by the flash-based Microsemi ProASIC3 A3PE600L [153].

Although the architecture of the readback scrubbers is more complicated than the design of the blind scrubbers, they are also implemented because they provide useful statistics on the number of induces SEUs. An example of the readback scrubber implementation is the Readout Control Unit [86, 154] utilized within the TPC detector at the

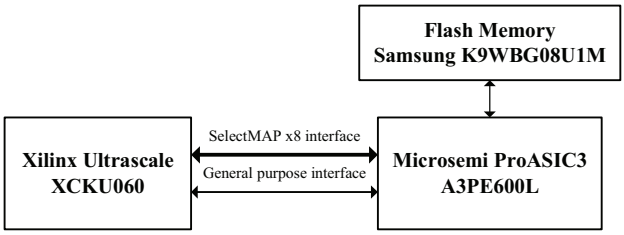


Figure 5.12: Scrubber architecture implemented in Readout Board of ITS Readout System.

ALICE experiment. The CRAM of the SRAM-based Xilinx Virtex-II [155] is monitored and refreshed by Microsemi ProASIC APA075 [156].

External flash memories utilized within scrubbing architectures are susceptible to radiation and they increase the complexity of the system. Because of this, external scrubbers that do not need access to the golden bitstream were implemented. The first example is the external scrubber for the system with triplicated SRAM-based FPGAs [157]. The external scrubber continuously compares the CRAMs of the three FPGAs, and if a discrepancy is found, then the faulty configuration frames of the affected device are reconfigured using the data retrieved from other FPGAs. Another example of the external scrubber is a solution based on the Error Detection and Correction mechanism, which does not require an access to the golden configuration data [158]. Yet another example is the JTAG Configuration Manager (JCM), which is the external scrubber that can operate as both blind and readback scrubber [110].

Hybrid Scrubber

The hybrid scrubber, whose architecture is presented in Fig. 5.13, has got two controllers. The internal controller is usually implemented as an on-chip hard core inside the target

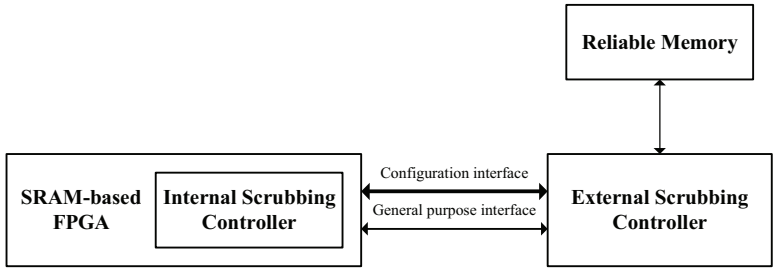


Figure 5.13: Architecture of hybrid scrubber.

SRAM-based FPGA. It provides the SEC-DED functionality, and by using the ECC

codes embedded in each configuration frame, it corrects SEUs and detect MBUs. When the internal controller finds a configuration frame containing an MBU, its address is sent to the external controller. It fetches the correct configuration frame from an external memory and rewrites the faulty data frame using either JTAG, SelectMAP, or PCAP interfaces [109, 110]. The external controller of the hybrid scrubber is usually:

- implemented with fabric blocks located inside the antifuse or flash-based FPGAs,
- is implemented in the processor of a System-on-a-chip [138] or in the external System-on-a-chip [109, 110],
- is realized as a dedicated software running on a PC [159, 160].

An example of the hybrid scrubber implementation in a high-energy physics experiment is the readout board of the TOF detector at CBM experiment [160]. The readout board is equipped with the SRAM-based Xilinx Artix-7 FPGA. The FPGA employs the Xilinx SEM Controller [141], which operates as the internal controller of the hybrid scrubber. The controller corrects SEUs and detects MBUs in the CRAM. While the MBUs are repaired by the dedicated software, which interfaces with the JTAG port of the SRAM-based FPGA via the radiation-tolerant GBT-SCA adapter [161]. The external controller of the hybrid scrubber is implemented in software, which runs on a PC located in the radiation-free area.

5.1.3 Inputs-outputs triplication

In order to increase the reliability of a system, all single points of failure should be removed. As explained in Section 5.1.1, the TMR methodology is known for increasing system reliability. When it is employed, all circuit replicas should have their own set of signals, so that a failure at the input is not propagated through all the redundant paths [122]. Thus, if one input is affected by a soft error, it will affect only one redundant block. The input triplication scheme is shown in Fig. 5.14. The single ended signal is brought to the three separate FPGA input pins. Then, each redundant logic is fed with the input signal coming from a separate input buffer IBUF [162].

As the TMR design generates critical logic paths in triplicate, there must be a method to bring triple logic paths back to a single path that does not create a single point of failure. This can be accomplished with the triplicated outputs. The output triplication scheme is shown in Fig. 5.15. The TMR output in the Xilinx FPGAs is designed using the generic 3-state output buffer OBUFT [162]. Each signal coming from the triplicated circuitry inside the FPGA has got its own output buffer, which is controlled by the minority voter that compares the primary signal to the two other signals. If the primary signal is in the majority, then the minority voter enables the corresponding output buffer, which allows the data on its primary path to be driven out through the OBUFT and onto

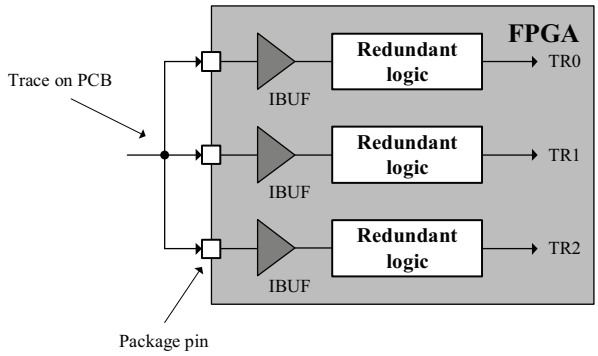


Figure 5.14: Triple redundant FPGA inputs [122].

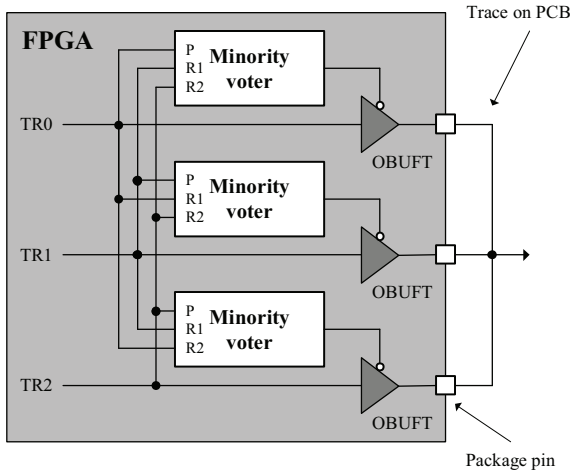


Figure 5.15: Minority voted TMR FPGA outputs [122].

the package pin. Otherwise, if the signal is in the minority, the minority voter disables the corresponding output buffer by placing its output in a high-impedance state allowing the redundant outputs to drive the correct data. Externally from the FPGA on the PCB, the three outputs are hardwired together. This scheme does not cause any contentious states because only paths that agree with others are actively driven.

5.2 Discussion

In this chapter, different radiation mitigation techniques were presented and discussed. It must be mentioned that the approach toward mitigation methods implementation in SRAM-based FPGAs is different than in ASICs. First of all, the designers must take into account that SRAM-based FPGAs contain the susceptible to radiation configuration

memory which configures all the device functionalities. On the other hand, while designing an ASIC designers can utilize radiation-hard CMOS technology such as the 65 nm CMOS by ST Microelectronics [163]. The selection of specific mitigation methods is up to the system designer, and depends both on the target radiation environment and the required system reliability. Often different mitigation methods are utilized together. The examples can be spatial redundancy and partial reconfiguration [164], spatial redundancy and scrubbing, or Error Correction Codes and scrubbing.

Chapter 6

Hardware prototype platform

6.1 Motivation and design objectives

The ITS Readout System, described in Section 1.4, is a fully customised electronic apparatus consisting of 192 Readout Units (RUs). Due to the unique functions that it serves and the necessity to operate in the radiation environment, a dedicated electronic solution has been developed. However, before designing and commissioning the final system, all of the functionalities must be tested and verified. Thus, to address many different research and development activities regarding the design of the ITS Readout System, the Prototype Readout Unit [165] has been designed. It is an electronic platform, whose versatile architecture allows for testing and verification of:

- the interface between a single pixel sensor installed on a dedicated carrier card, or Inner Barrel (IB) or Outer Barrel (OB) Hybrid Integrated Circuit (HIC) and readout electronics,
- the signal integrity and data transfer reliability over 5 m long twinax cables,
- the interface to the Power Boards and the ALICE Online and Offline computing system,
- the triggering capabilities,
- the FPGA's operation in a radiation environment,
- the methods for mitigation of radiation effects.

6.2 Architecture and implementation

The hardware prototype platform consists of the Prototype Readout Unit and the custom FMC mezzanine card. The architecture of the Prototype Readout Unit is presented in Fig. 6.1. The SRAM-based Xilinx Kintex-7 XC7K325T [166] FPGA is the main processing unit. It receives data from the single pixel sensor or the pixel sensor module (IB

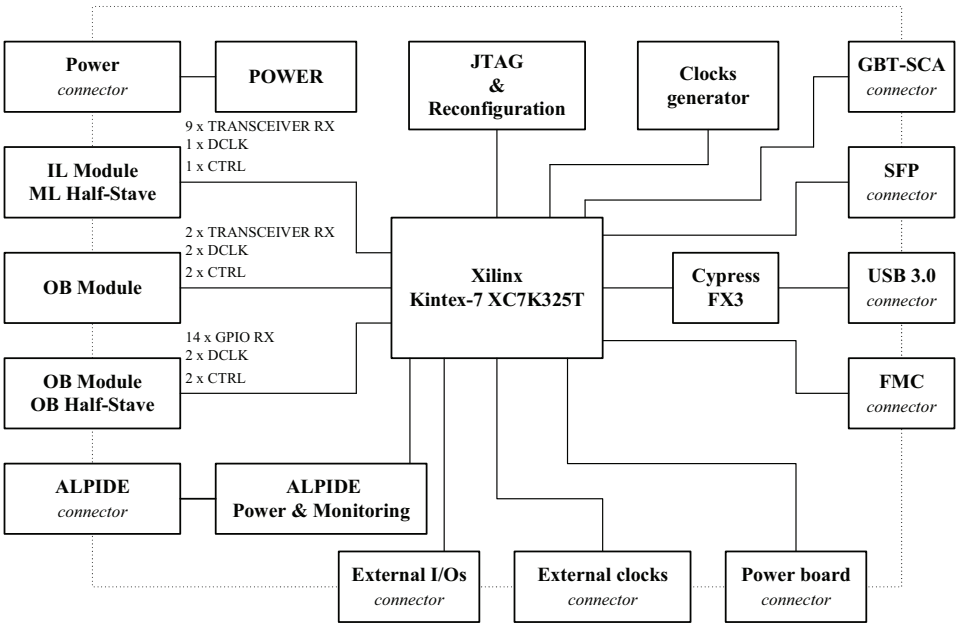


Figure 6.1: Architecture of ITS Prototype Readout Unit (ITS RU v0).

or OB HIC) either via high-speed tranceivers or differential GPIO inputs. The FPGA accesses the sensor's differential control interface either via an MLVDS buffer or a direct AC-coupled connection. The adjustable power supply section generates and monitors the voltages for the pixel sensor modules. In order to provide communication with a PC for debugging or sending out data, the board has been equipped with the Cypress FX3 USB 3.0 controller. An FMC slot is installed to interface with the GBTx-FMC board [167] that allows for communication with a DAQ or trigger system via the versatile GBT link [19]. Also, an SFP optical transceiver module has been installed to enable the implementation of the GBT-FPGA project [168]. Multiple FPGA configuration schemes have been foreseen to support both the initial programming and scrubbing. The FPGA can be configured either by a JTAG programmer, GBT-SCA JTAG master controller [161], or via a flash-based memory.

A versatile clock tree allows the FPGA to operate using either a local 160 MHz clock, an LHC 40 MHz machine clock received by the GBTx chip, or an externally provided clock. Shunt resistors at the outputs of the power converters with easily accessible terminals facilitate power monitoring and characterization. The Prototype Readout Unit provides a dedicated interface to control the ITS Power Boards. Most of the unused transceivers and clock inputs can be accessed from outside to further increase flexibility. The ITS

Prototype Readout Unit is a 6U[1] VME form factor, 10-layers electronic board. Its physical implementation is presented in Fig. 6.2.

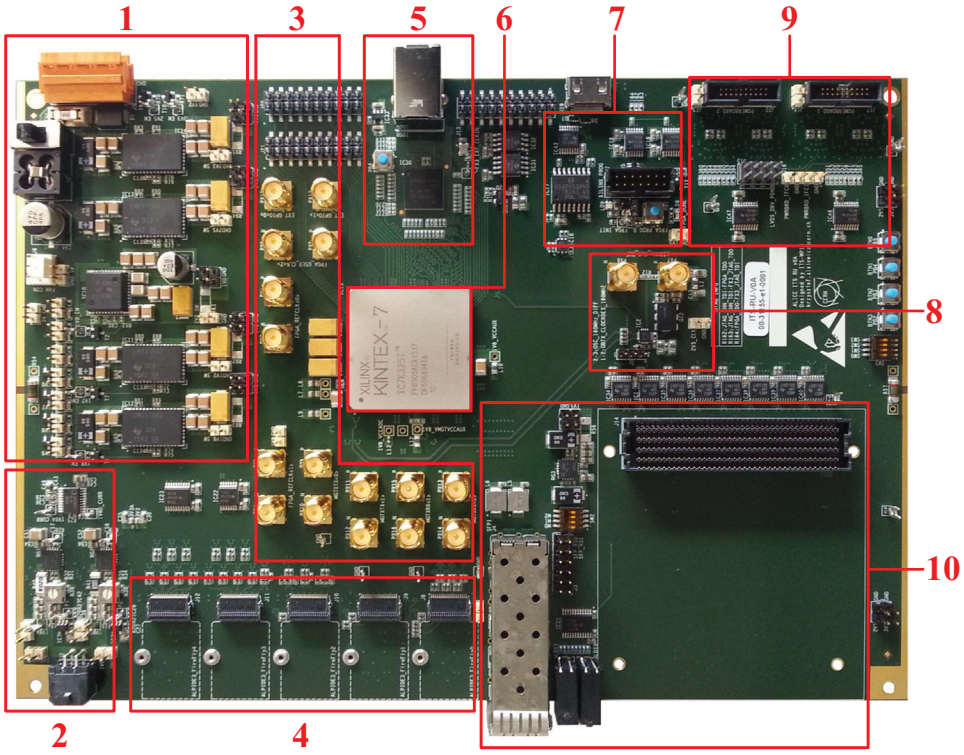


Figure 6.2: ITS Prototype Readout Unit (ITS RU v0). (1) Board power section, (2) Sensor powering section, (3) Transceivers and clock input ports, (4) Connectors for Samtec Twinax ribbon cables and termination networks, (5) USB 3.0 interface, (6) Xilinx Kintex-7 XC7K325T FPGA, (7) FPGA configuration interface, (8) Clock input for clocking tree, (9) Connectors for interface to ITS Power Board, (10) FMC slot for GBTx-FMC mezzanine card and optical SFP module.

The Prototype Readout Unit was used as a main hardware platform for characterizing the ALPIDE pixel sensor for qualification during the Production Readiness Review. It was also used as a primary prototyping base to test and verify different functionalities of the ITS Readout System during its Engineering Design Review.

The GBTx-FMC is shown in Fig. 6.3. Its architecture and connection with the Prototype Readout Unit v0 is presented in Fig. 6.4. The FMC hosts the GBTx and GBT-SCA chips. VTRx and VTTx optical modules provide optical communication with the Central

[1] 233 mm × 160 mm

Trigger Processor and ALICE Online and Offline computing system. The FMC interfaces with the SRAM-based Kintex-7 FPGA via FMC-HPC connector.

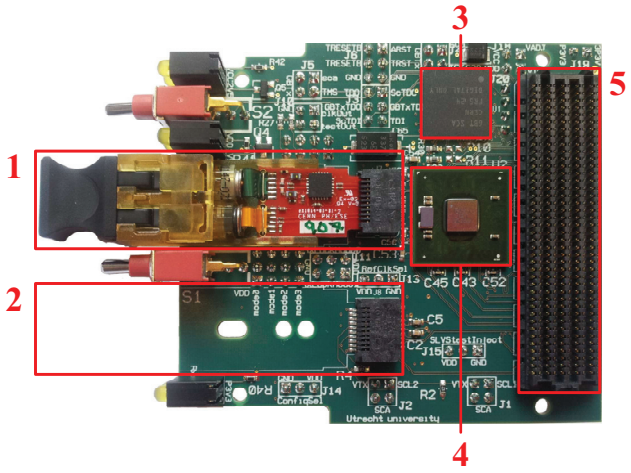


Figure 6.3: GBTx-FMC mezzanine card. (1) VTRx bidirectional optical communication module, (2) VTTx one-directional optical communication module (unmounted), (3) GBT-Slow Control Adapter, (4) GBTx chip, (5) FMC-HPC connector.

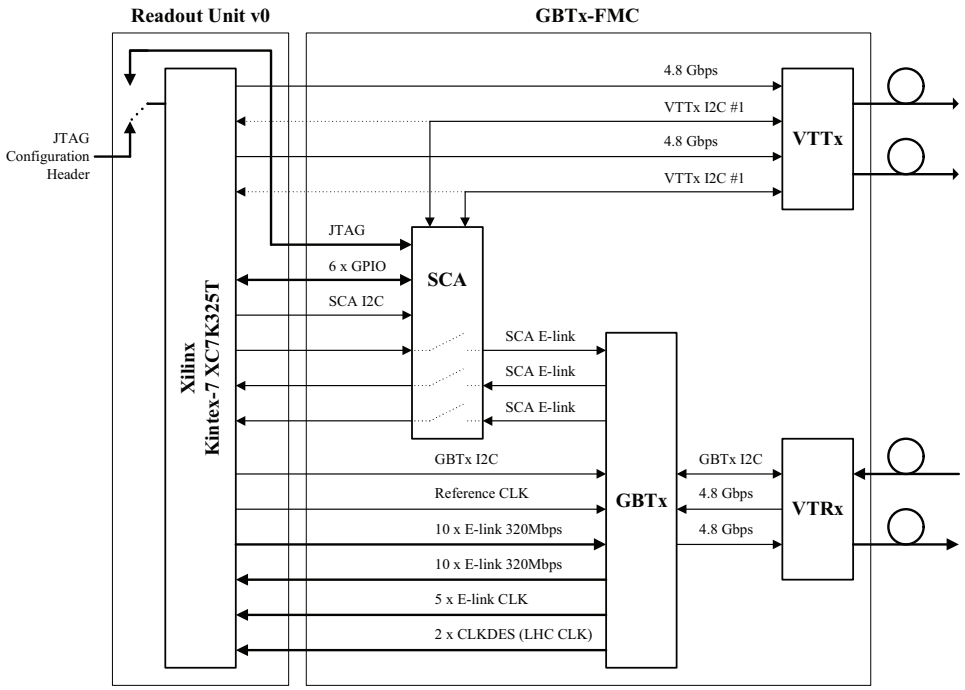


Figure 6.4: Architecture of GBTx-FMC mezzanine card and connection with Readout Unit v0.

Chapter 7

Experimental testing of the mitigation methods

As described in Section 1.4, the ALICE Inner Tracking System Readout will utilize SRAM-based FPGA devices, which are susceptible to ionizing and non-ionizing radiation. The final logic design operating simultaneously in 192 FPGAs will consist of:

- long combinational and sequential processing pipelines,
- memory components based on Block RAM,
- Finite State Machines (FSMs).

Thus, before deploying the Readout System in the target radiation environment of the ALICE experiment, it was essential to characterize the aforementioned logical structures using the design functional error rate estimation methodology described in Chapter 4, and experimentally evaluate and quantitatively compare effectiveness of main SRAM-based FPGA mitigation methods. The experimental results have been summarized into an implementation guideline for the final FPGA design. The following Chapter presents:

- the design of the setup used during experimental testing,
- the developed testing designs,
- the obtained experimental results from fault injection and irradiation tests,
- the discussion of the obtained experimental results.

7.1 Experimental setup

The following Section presents the hardware setup, which was employed in all of the experimental tests described in this Chapter. The irradiation experiments were conducted at the isochronous cyclotron [102] at the Nuclear Physics Institute of the Academy of Sciences of the Czech Republic in Řež, near Prague. The machine provides a proton beam with an energy range from 6 to 37 MeV. The available proton flux ranges from 10^4 to

$10^{14}\,\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$, over a uniform area of about $2.5 \times 2.5\,\mathrm{cm}^2$. All the experimental tests were carried out with a 30 MeV proton beam at the target's surface at Linear Energy Transfer (LET) of $1.469 \cdot 10^{-2}\,\mathrm{MeV\,cm^2\,mg^{-1}}$. In order to continuously monitor the beam intensity during the irradiation tests, a dedicated dosimetry system [101, 102] was used.

The experimental setup, presented in Fig. 7.1, was installed on a remotely controlled X-Y platform to allow for the precise beam alignment before each set of irradiation tests. The hardware platform [165, 169], described in detail in Chapter 6, was installed behind a 10 mm aluminum shield which stops all protons and protects the active electronic components on the board. Only the Xilinx Kintex-7 325T [166] SRAM-based FPGA can be exposed to the proton beam. FPGA's exposure to the beam is managed by a remotely controlled pneumatic shutter. The test setup connections are shown schematically in Fig. 7.2. All of the irradiation tests are automated by the custom software running on a PC located in a bunker 5 m away from the radiation zone. The software operates as described in Section 4.3. The status of the testing design is periodically read out over USB and UART interfaces. A back-up UART connection is used to read out the testing design status in case of error readout module failure. The FPGA can be programmed and reconfigured via a JTAG programmer [170, 171] or an external scrubbing device. The JTAG Configuration Manager (JCM) [110] is used as an external scrubbing device. The controlling software commands the power supply and the beam shutter controller. The supply current was continuously monitored, even though no latch-up events were expected with a 30 MeV proton beam.

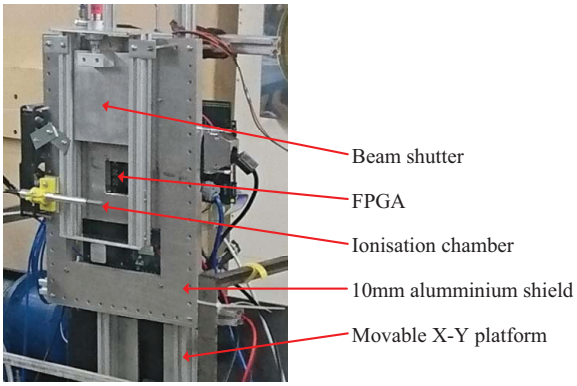


Figure 7.1: Test setup used during proton irradiation tests [169].

The fault injection tests were also carried out using the aforementioned hardware platform. The Prototype Readout Unit v0 was connected to the controllable power supply, and either to the JTAG programmer or JCM. All of the fault injection tests were automated by the custom software running on a PC.

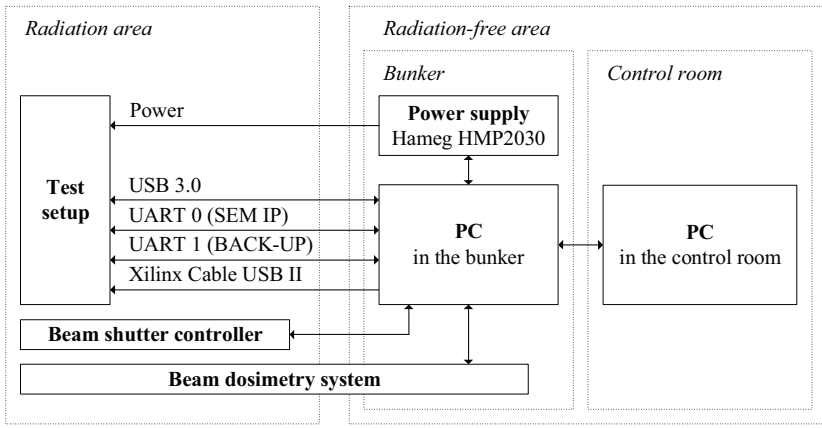


Figure 7.2: Diagram of beam test setup connections.

7.2 Combinational and sequential circuits testing

The testing design [169], based on the design functional error rate estimation methodology described in Chapter 4, has been developed to test for radiation susceptibility of combinational and sequential circuits implemented using FPGA's Look-up Tables (LUTs) and registers (FFs). It allows for the study of different topologies of Triple Modular Redundancy (TMR), and the refreshment of configuration memory (scrubbing).

7.2.1 Testing design architecture and theory of its operation

The architecture of the developed testing design is presented in Fig. 7.3. The design contains many identical modules, called lanes, and an error readout module. Each lane consists of a triplicated pattern generator, a chain of Logic Test Structures (LTSs), a triplicated pattern checker, and a triplicated discriminator. The pattern generator outputs test vectors that are periodically repeated 6 bits wide sequences from 0 to 63. A width of the test vector was set to 6 bits because the input of LUTs in the Xilinx Kintex-7 are also 6 bits wide [30]. The LTS is replicated 64 times, forming a chain that shifts the test vectors. The basic LTS, shown in Fig. 7.4, consists of a hard-coded LUT transfer function (Combinational Logic) and an output register (R). The combinational logic increments the input data by 1 and the output register holds the incremented value for one clock cycle. In order to test different spatial redundancy topologies in the LTS, it is possible to employ various TMR schemes described in detail in Section 5.1.1. The TMR can be employed either in the combinational logic (Fig. 7.6), the output register (Fig. 7.5), or both (Fig. 7.7). Also, the data voter between the combinational logic and the output register (Fig. 7.8), or the full chain of the basic LTSs (Fig. 7.9) can be implemented with TMR.

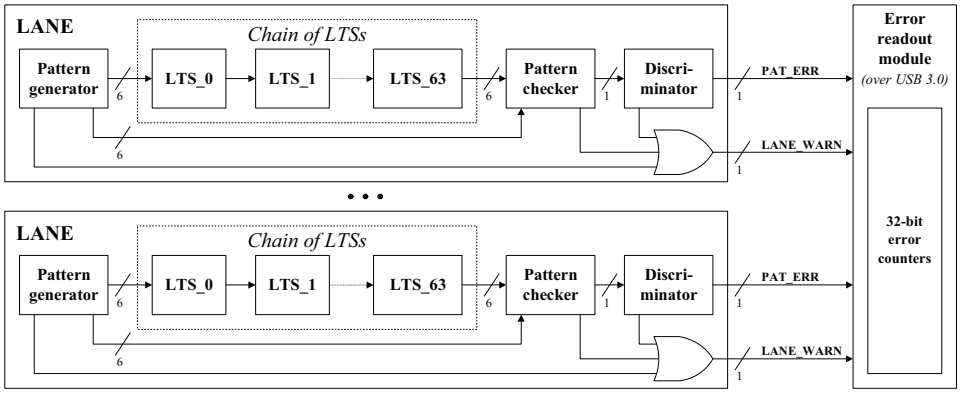


Figure 7.3: Architecture of combinational and sequential circuits testing design.

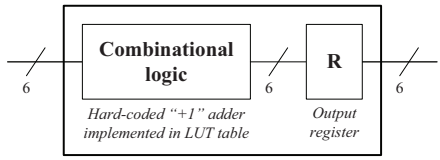


Figure 7.4: Basic Logic Test Structure design.

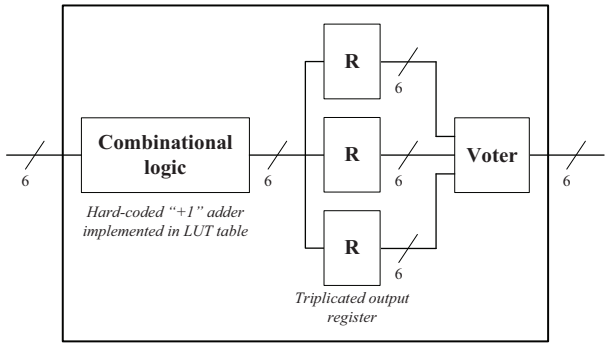


Figure 7.5: Logic Test Structure design in which only output register is triplicated.

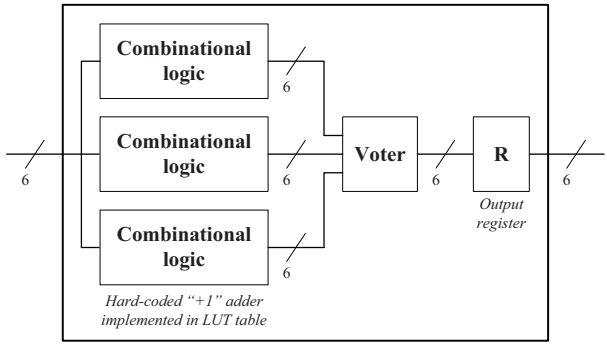


Figure 7.6: Logic Test Structure design in which only combinational logic is triplicated.

After the reset signal is deasserted, the pattern generator starts outputting test vectors which are pushed through the chain of the LTSs at a frequency of 100 MHz. 64 clock cycles later, data at the output from the chain of the LTSs are copies of the data outputted by the pattern generator. This condition remains true unless a soft error occurs either in the pattern generator, the chain of the LTSs, or the routing between those modules and the pattern checker. The pattern checker compares the output from the chain of the LTSs to the reference output from the pattern generator. If a discrepancy is found, a pattern error signal is asserted on the PAT_ERR output. The discriminator receives that signal and increments the corresponding error counter only once per the full pattern generator

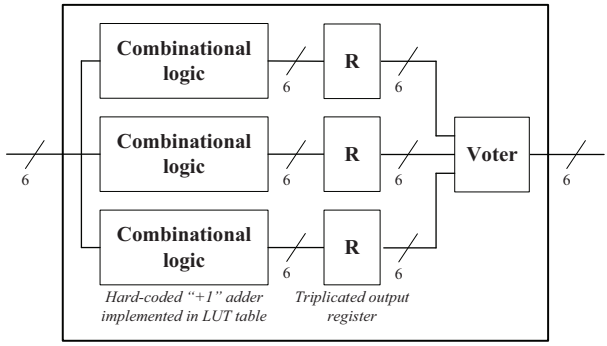


Figure 7.7: Logic Test Structure design in which both combinational logic and output register are triplicated.

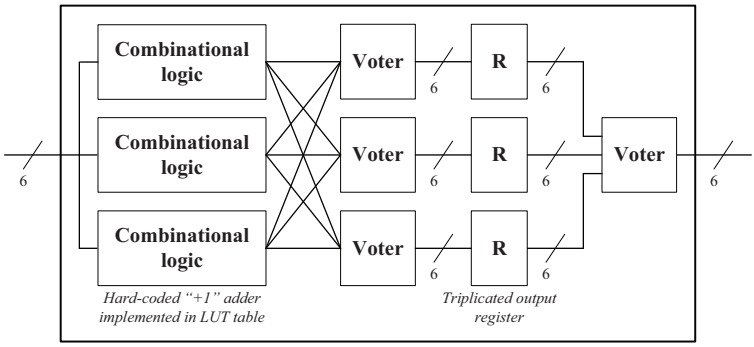


Figure 7.8: Logic Test Structure design in which both combinational logic and output register are triplicated. They are connected via 3 redundant voters.

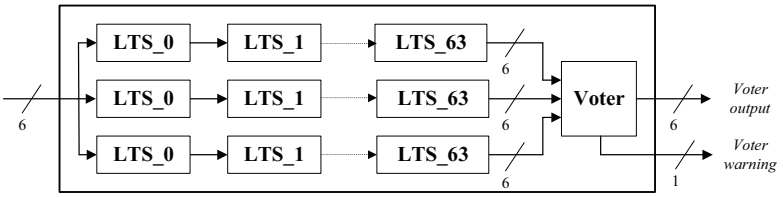


Figure 7.9: Triplicated chain of Logic Test Structures.

sequence. Also, when a discrepancy is found during voting in the pattern generator, pattern checker, or discriminator, a lane warning signal is asserted on the $LANE\_WARN$ output.

During beam or fault injection test, the values stored in the error counters are periodically read out via a USB interface and saved. During later analysis, it is possible to recognize how many lanes were affected by soft errors or whether the error was in the

error readout module. Tests where the error readout module failed are excluded from the further analysis.

The utilization of resources by different implementations of the logic test structures in the Kintex-7 FPGA is presented in Tab. 7.1. In order to replicate either the combinational logic or register, it is necessary to add a voter. Resources required to implement it and other components of the lane are shown in Tab. 7.2. It takes 6 LUTs to add a single 6 bits voter to any LTS design. The implementation of the basic LTS requires 4 LUTs and 6 FFs, while the most resource-demanding one utilizes 36 and 18, respectively. To implement the topology where the full chain of the basic LTSs is triplicated (Fig. 7.9), 774 LUTs and 1152 FFs are required. Depending on the selected TMR topology of the logic test structure, 6 different testing designs with 256, 160, 128, or 64 lanes were implemented to utilize as many FPGA resources as possible, thus maximizing the area susceptible to radiation. In Tab. 7.3 the utilization of resources by different implementations of the testing designs is presented. Also, the numbers of essential bits per different testing designs variants were extracted. Essential bits are utilized CRAM bits that are responsible for the correct configuration of the circuit implemented in the SRAM-based FPGA device [172]. In the testing FPGA design where the full chain of the LTSs was triplicated and the redundant voter was used, the utilization of essential bits per lane is the highest and equals 0.49 %. Meanwhile, in the testing design where nothing was triplicated, utilization of the essential bits per lane is the lowest and equals 0.1 %.

Logic Test Structure (LTS)	**Figure**	**Resources**	
		LUT	FF
Nothing triplicated	Fig. 7.4	4	6
Register triplicated	Fig. 7.5	10	18
Combinational logic triplicated	Fig. 7.6	18	6
Combinational logic and registers triplicated	Fig. 7.7	18	18
Combinational logic and registers triplicated, redundant voter	Fig. 7.8	36	18

Table 7.1: Utilization of resources, in Xilinx Kintex-7 325T, by logic test structures with different TMR topologies employed.

Block	Resources	
	LUT	FF
Triplicated pattern generator	40	24
Triplicated pattern checker	13	3
Triplicated discriminator	7	6
Voter 3×6 bits to 6 bits	6	0

Table 7.2: Utilization of resources by different functional blocks in Xilinx Kintex-7 325T.

Firmware	Lanes	Used resources per design				Essential bits	
		LUT	LUT per lane	FF	FF per lane	Testing design	Lane
Nothing triplicated	256	57 %	**0.22 %**	35 %	**0.14 %**	25.9 %	**0.1 %**
Registers triplicated	160	74 %	**0.46 %**	52 %	**0.33 %**	37 %	**0.23 %**
Combinational logic triplicated	128	80 %	**0.63 %**	18 %	**0.14 %**	32.2 %	**0.25 %**
Combinational logic and registers triplicated	128	81 %	**0.63 %**	42 %	**0.33 %**	32.2 %	**0.25 %**
Combinational logic and registers triplicated, redundant voter	64	80 %	**1.25 %**	21 %	**0.33 %**	31.1 %	**0.49 %**
Full chain of LTS triplicated	128	69 %	**0.54 %**	42 %	**0.33 %**	28.7 %	**0.22 %**

Table 7.3: Utilization of resources, in Xilinx Kintex-7 325T, by testing design with different schemes of replication (values presented as percentages of total amount of resources in device). Lowest and highest utilization of essential bits per lane within developed testing designs with radiation mitigation methods employed is marked.

7.2.2 Fault injection and proton irradiation testing results

Tests without CRAM scrubber

Tests without the CRAM scrubber were carried out both in proton irradiation and fault injection tests. The number of SEUs induced to the CRAM $N_{\mathrm{ERR\_CB}}$ during a single irradiation test was estimated using Eq. 7.1 (where σ_{CRAM} is the CRAM cross-section, Φ is the beam fluence, F is the beam flux, T is time of exposure to the beam, and N_{CB} is the number of CRAM bits).

$$N_{\mathrm{ERR\_CB}} = \sigma_{\mathrm{CRAM}} \Phi N_{\mathrm{CB}} = \sigma_{\mathrm{CRAM}} F T N_{\mathrm{CB}} \tag{7.1}$$

It was calculated that, during the 120 s long irradiation test at the flux of $10^7\,\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$ (fluence equals $1.2 \cdot 10^9\,\mathrm{cm}^{-2}$), on average 408 and 100 errors are induced to the CRAM and BRAM, respectively (on average an error is induced the the CRAM every 0.29 s). Then, during fault injection tests, the same number of faults were injected to the CRAM using the Xilinx Soft Error Mitigation Controller [141]. It is important to mention that during the irradiation tests, SEUs were induced both to the CRAM and BRAM, while during the fault injection tests, faults were injected only to the CRAM. To obtain the required statistics, fault injection and beam tests for each of the testing designs were repeated 500 and 50 times, respectively. Lower statistics of the irradiation measurements are a result of both availability issues and the high price of beam testing. In each test, the number of faulty lanes was monitored. The lane was considered faulty after its pattern checker asserted the pattern error signal (PAT_ERR). After carrying out fault injection and beam tests, the data were processed and the cross-section of the lane σ_{L} for different testing designs was calculated using Eq. 7.2 (where N is the number of conducted irradiation or fault injection tests, E is the number of faulty lanes in a single test, N_{L} is the number of lanes in the given testing design, Φ_{i} is the fluence, F_{i} is the average flux, and T_{i} is the time of exposure to the beam during a single test).

$$\sigma_{\mathrm{L}} = \frac{\sum_{\mathrm{i}=1}^{\mathrm{N}} E_{\mathrm{i}}}{N_{\mathrm{L}} \sum_{\mathrm{i}=1}^{\mathrm{N}} \Phi_{\mathrm{i}}} = \frac{\sum_{\mathrm{i}=1}^{\mathrm{N}} E_{\mathrm{i}}}{N_{\mathrm{L}} \sum_{\mathrm{i}=1}^{\mathrm{N}} F_{\mathrm{i}} T_{\mathrm{i}}} \tag{7.2}$$

The results are presented in Tab. 7.4. The largest lane cross-section was measured for the testing design where only the output register of the LTS is implemented with TMR. This result was expected because the 6-bit voter cross-section is larger than the single bit register one, thus increasing the total LTS cross-section. The lowest lane cross-section was measured for the testing design where the full chain of the LTSs is triplicated.

Fig. 7.10 visualises the operation of the testing design with triplicated chains of the basic LTSs. The upper plot shows the results of comparisons (PAT_ERR signals) of

the data voted from the triplicated chains of the LTSs and the reference outputs from the pattern generators. Lack of marks on the plot means that the comparison operation was correct. The bottom plot shows warnings while voting the outputs by the pattern generators, the chain voters, the pattern checkers, or the discriminators ($LANE\_WARN$ signals). Without scrubbing, errors in the CRAM are not corrected and the lanes affected by SEUs stay faulty until the end of the test.

Testing with CRAM scrubber

The testing design was also tested with two different CRAM scrubbers. During fault injection tests, the errors were injected to the CRAM using the Xilinx Soft Error Mitigation Controller. First, the controller was switched into the fault injection mode in order to inject a single error to a randomly selected position in the CRAM. Then, the controller was switched into observation and correction mode so that the injected error could be automatically localized and corrected. The number of repetitions of this sequence was calculated using Eq. 7.1.

During the proton irradiation tests, the JCM scrubber [110] was employed and configured to operate as a blind scrubber. As it performs a blind scrub cycle every 2 s, the duration of a single irradiation test and beam intensity were set to 480 s and $10^6\,\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$ respectively, so that on average an SEU was induced to the CRAM every 2.94 s. This means that the entire CRAM content was refreshed 1.5 times between occurrences of an SEU. The corresponding fluence of $4.8 \cdot 10^8\,\mathrm{cm}^{-2}$ was 2.5 times less than the fluence of the conducted irradiation tests without the scrubber employed.

Fig. 7.11 and Fig. 7.12 visualise the operation of the testing design with triplicated chains of the basic LTSs. The presented tests were carried out with the JCM scrubber employed at the flux of $10^6\,\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$ and $10^7\,\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$, respectively. The upper plots show when the lanes no longer generate the correct outputs (PAT_ERR signals). The bottom plots show errors while voting the outputs in the pattern generators, the chain voters, the pattern checkers, or discriminators of each lane ($LANE\_WARN$ signals). The striking difference in respect to Fig. 7.10 is that the warning outputs due to the errors in the lanes are continuously reset by the operation of the scrubbers (bottom plot). Without errors accumulating in the lanes (as it is in the case of the operation without any scrubber shown on Fig. 7.10) the TMR topology ensures that the voted lane outputs stay error-free for the whole irradiation period. It can be seen that the increase of the flux induces more errors to the CRAM, which are anyway corrected by the external scrubber. Although warnings in voting by different modules are detected, the pattern checkers do not report any errors.

The measurement results are summarized in Tab. 7.5. The highest increase in radiation

hardness can be observed for the testing design where the full chain of the basic LTSs was triplicated.

7.2.3 Discussion

Among all of the evaluated testing designs, the one with the full chains of the basic LTSs implemented with TMR is the least susceptible to radiation. Although the full chain of the basic LTSs is triplicated, the utilization of resources per lane is the lowest within the developed testing designs with TMR employed. The reason for this is the amount of implemented voters. In this particular design it is necessary to add only one voter per lane, while in other architectures, voters must be added at every local point of triplication. Furthermore, adding a voter requires not only additional LUTs, but also many routing resources which increase the total number of utilized essential bits and in consequence lead to the increase of radiation susceptibility. As for any other TMR topology, the employment of the scrubber is mandatory to keep the system operating correctly.

Analyzing the obtained lane cross-sections and resource utilization of the tested designs, a conclusion could be drawn that the higher the ratio between the amount of resources of the triplicated circuit and those of the voter, the higher the protection efficiency.

Fault injection	**Errors**	**Lanes**	**Faulty lanes**		**Lane σ**	**Error $\delta\sigma$**
			Mean	SD	$(\mathrm{cm}^2\,\mathrm{lane}^{-1})$	$(\mathrm{cm}^2\,\mathrm{lane}^{-1})$
Nothing triplicated	408	256	23.72	6.07	$7.99 \cdot 10^{-11}$	$1.03 \cdot 10^{-12}$
Register triplicated	408	160	19.83	4.84	$\mathbf{10.70 \cdot 10^{-11}}$	$1.29 \cdot 10^{-12}$
Combinational logic triplicated	408	128	6.00	2.53	$4.04 \cdot 10^{-11}$	$0.89 \cdot 10^{-12}$
Combinational logic and registers triplicated	408	128	5.59	3.00	$3.76 \cdot 10^{-11}$	$0.97 \cdot 10^{-12}$
Combinational logic and registers triplicated, redundant voter	408	64	2.80	1.70	$3.78 \cdot 10^{-11}$	$1.03 \cdot 10^{-12}$
Full chain of LTS triplicated	408	128	0.97	1.04	$\mathbf{0.65 \cdot 10^{-11}}$	$0.32 \cdot 10^{-12}$
Proton irradiation tests	**Fluence** (cm^{-2})					
Nothing triplicated	$1.2 \cdot 10^9$	256	28.1	6.85	$9.70 \cdot 10^{-11}$	$3.22 \cdot 10^{-12}$
Registers triplicated	$1.2 \cdot 10^9$	160	23.43	5.89	$\mathbf{12.20 \cdot 10^{-11}}$	$4.85 \cdot 10^{-12}$
Combinational logic triplicated	$1.2 \cdot 10^9$	128	5.55	2.87	$3.61 \cdot 10^{-11}$	$2.62 \cdot 10^{-12}$
Combinational logic and registers triplicated	$1.2 \cdot 10^9$	128	4.64	2.13	$3.11 \cdot 10^{-11}$	$1.85 \cdot 10^{-12}$
Combinational logic and registers triplicated, redundant voter	$1.2 \cdot 10^9$	64	2.32	1.71	$3.08 \cdot 10^{-11}$	$3.25 \cdot 10^{-12}$
Full chain of LTS triplicated	$1.2 \cdot 10^9$	128	0.95	0.86	$\mathbf{0.62 \cdot 10^{-11}}$	$0.65 \cdot 10^{-12}$

Table 7.4: Comparison of lane cross-sections obtained from proton irradiation and fault injection tests without CRAM scrubber.

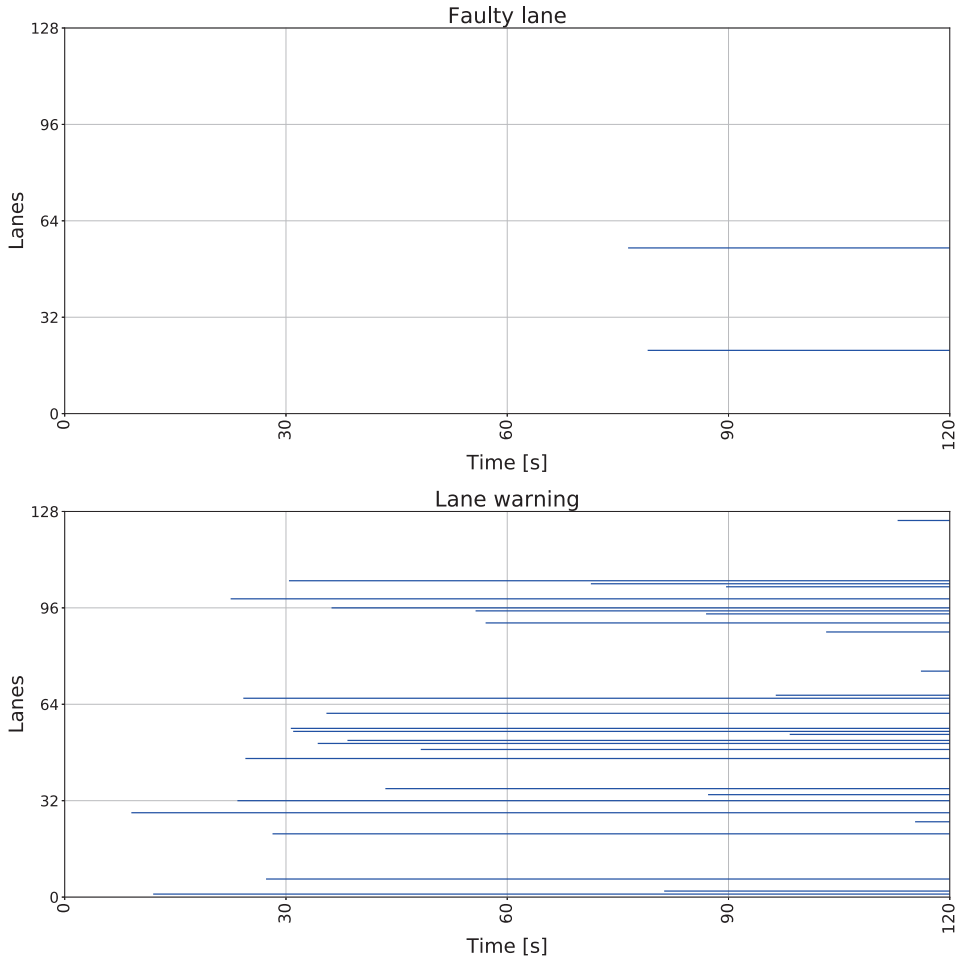


Figure 7.10: Visualization of testing firmware operation with full chains of basic LTSs triplicated (Fig. 7.9). Test parameters are: $T = 120\,\mathrm{s}$, $F = 10^7\,\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$, CRAM scrubber not employed. Upper plot shows when voters at end of lanes no longer generate correct outputs (*PAT_ERR* signals). Bottom plot shows warnings while voting outputs by pattern generators, chain voters, pattern checkers, or discriminators (*LANE_WARN* signals). As no CRAM scrubber is employed, functional errors caused by SEUs accumulate, causing more lanes to fail.

Fault injection (SEM IP scrubber)	**Errors**	**Lanes**	**Faulty lanes** Mean	SD	**Lane σ** $(\mathrm{cm^2\,lane^{-1}})$	**Error $\delta\sigma$** $(\mathrm{cm^2\,lane^{-1}})$
Nothing triplicated	408	256	22.58	5.19	$7.60 \cdot 10^{-11}$	$1.26 \cdot 10^{-12}$
Register triplicated	408	160	19.88	5.39	$\mathbf{10.70 \cdot 10^{-11}}$	$1.81 \cdot 10^{-12}$
Combinational logic triplicated	408	128	5.98	2.49	$4.03 \cdot 10^{-11}$	$1.19 \cdot 10^{-12}$
Combinational logic and registers triplicated	408	128	5.39	2.5	$3.63 \cdot 10^{-11}$	$1.09 \cdot 10^{-12}$
Combinational logic and registers triplicated, redundant voter	408	64	2.87	1.65	$3.86 \cdot 10^{-11}$	$0.56 \cdot 10^{-12}$
Full chain of LTS triplicated	408	128	0.41	0.78	$\mathbf{0.28 \cdot 10^{-11}}$	$0.30 \cdot 10^{-12}$
Proton irradiation tests (JCM scrubber)	**Fluence** $(\mathrm{cm^{-2}})$					
Nothing triplicated	$0.48 \cdot 10^9$	256	13.50	3.15	$10.60 \cdot 10^{-11}$	$6.04 \cdot 10^{-12}$
Registers triplicated	$0.48 \cdot 10^9$	160	9.50	3.32	$\mathbf{11.60 \cdot 10^{-11}}$	$9.67 \cdot 10^{-12}$
Combinational logic triplicated	$0.48 \cdot 10^9$	128	1.56	1.25	$2.57 \cdot 10^{-11}$	$4.80 \cdot 10^{-12}$
Combinational logic and registers triplicated	$0.48 \cdot 10^9$	128	2.47	1.39	$4.04 \cdot 10^{-11}$	$5.19 \cdot 10^{-12}$
Combinational logic and registers triplicated, redundant voter	$0.48 \cdot 10^9$	64	1.16	1.07	$3.77 \cdot 10^{-11}$	$7.99 \cdot 10^{-12}$
Full chain of LTS triplicated	$0.48 \cdot 10^9$	128	0.06	0.24	$\mathbf{0.09 \cdot 10^{-11}}$	$0.92 \cdot 10^{-12}$

Table 7.5: Comparison of lane cross-sections obtained from proton irradiation and fault injection tests with CRAM scrubbers.

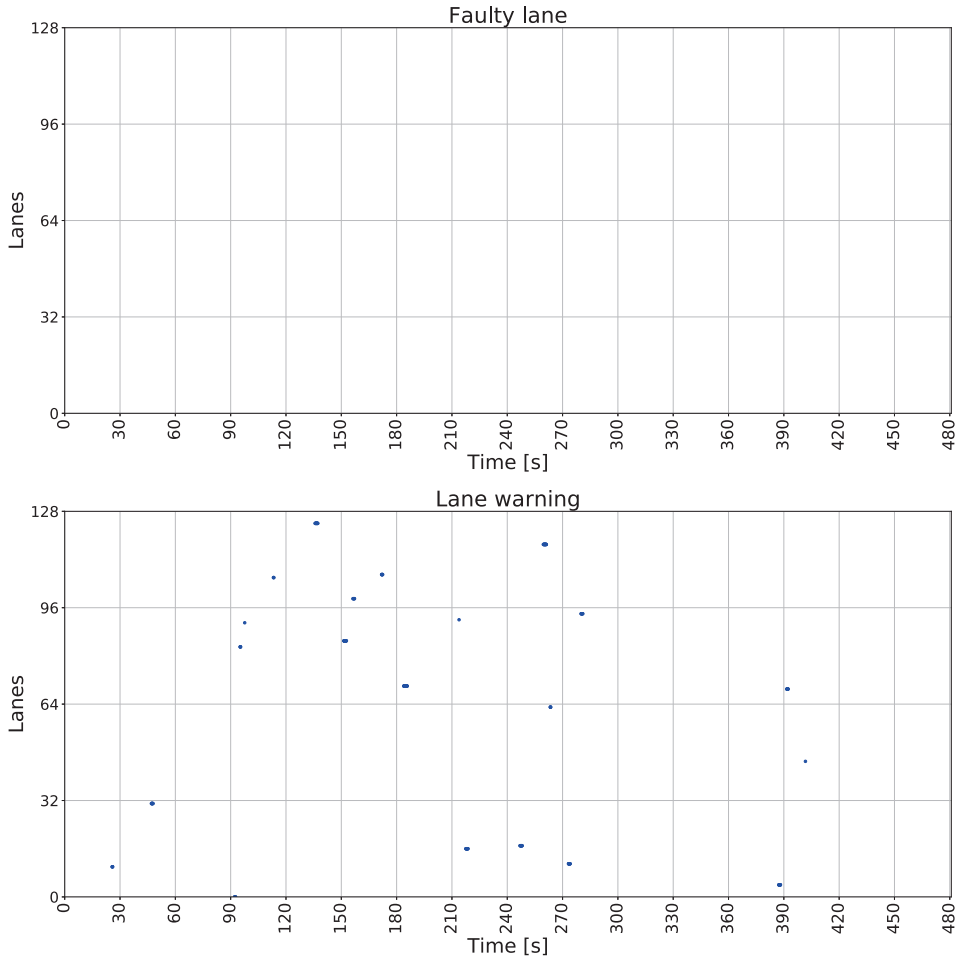


Figure 7.11: Visualization of testing design operation with full chains of basic LTSs triplicated (Fig. 7.9). Test parameters are: $T = 480\,\mathrm{s}$, $F = 10^6\,\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$, JCM scrubber employed. Upper plot shows when lanes no longer generate correct outputs (*PAT_ERR* signals) and how the malfunctions are corrected. Bottom plot shows warnings while voting outputs by pattern generators, chain voters, pattern checkers, or discriminators (*LANE_WARN* signals).

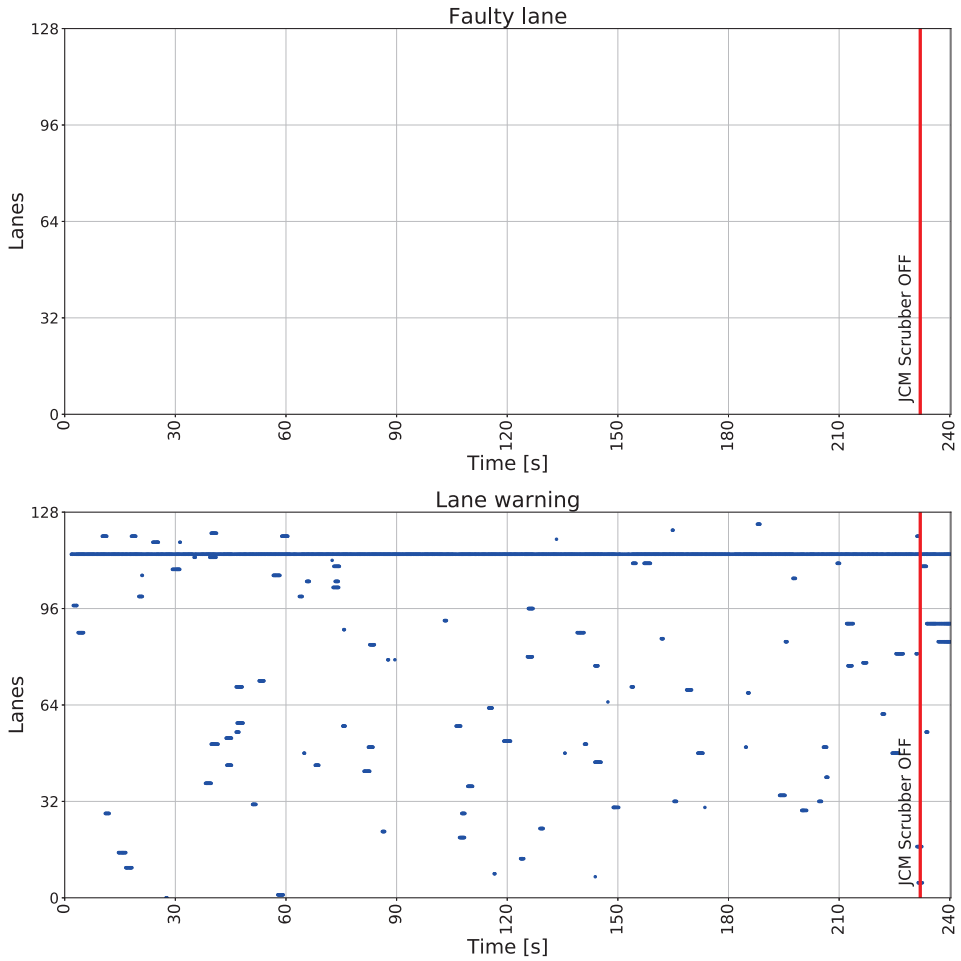


Figure 7.12: Visualization of testing design operation with full chains of basic LTSs triplicated (Fig. 7.9). Test parameters are: $T = 240\,\mathrm{s}$, $F = 10^7\,\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$, JCM scrubber employed. Upper plot shows when lanes no longer generate correct outputs (*PAT_ERR* signals) and how the malfunctions are corrected. Bottom plot shows warnings while voting outputs by pattern generators, chain voters, pattern checkers, or discriminators (*LANE_WARN* signals). After 228 s JCM scubber is turned off and it can be seen that errors are no longer repaired. In bottom plot, continuous warning is visible. Single branch in triplicated pattern generator was affected and then repaired by scrubber. However, as no synchronisation mechanism is implemented between branches of the pattern generator, voter will constantly report warning because of out-of-sync branch.

7.3 Built-in Xilinx Kintex-7 Block RAM testing

FIFO memories are essential elements of almost every single FPGA design. In modern SRAM-based FPGAs they are implemented using either hard built-in macros, on-chip Block RAM, shift registers, or distributed RAM. Usually deep depth FIFOs employ on-chip BRAM, which, similar as CRAM, is implemented using SRAM cells that are susceptible to ionizing and non-ionizing radiation.

The typical radiation mitigation techniques employed to protect BRAM are either spatial redundancy or Error Correction Codes (ECCs). In the case of Xilinx 7 Series FPGAs Hamming code, error correction [173] is employed, which is a Single Error Correct Double Error Detect (SEC-DED) mechanism. Although the BRAM is protected, the Hamming Error Injection and Correction Checking mechanism, consisting of encoder, decoder and correction units, is still susceptible to radiation. The BRAM of the Xilinx Kintex-7 has already been experimentally evaluated in [71,75,174,175] but the available studies did not directly test the performance of the Hamming Error Injection and Correction Checking mechanism. Thus, the main motivation for implementing the testing design and conducting the experimental testing is to characterize the radiation susceptibility of the FPGA's built-in Hamming ECC mechanism and its usefulness for a final FPGA design for the ALICE Inner Tracking System Readout.

7.3.1 Testing design and theory of its operation

The developed testing design is based on the methodology explained in Chapter 4. Its architecture is presented in Fig. 7.13. The design consists of 4 replicated lanes and the common error readout module. Each lane contains the triplicated 32-bit pattern generator, a chain of FIFOs, and the triplicated pattern checker.

After the reset signal is deasserted, the pattern generator starts outputting test vectors which are pushed through the chain of FIFOs at a frequency of 100 MHz. The chain of 48 FIFOs is presented in Fig. 7.14. Between separate FIFO instances, a dedicated glue logic module reads data and writes them to the next FIFO only when the previous one is almost full. It allows as much BRAM content as possible to be exposed to radiation. Each FIFO embeds two 36 kB blocks of BRAM protected with the Hamming code. The data checker reads data from the chain of the FIFOs. It increments the previous data word by 1 and compares it with the current one. The incremented data word and the current one are equal unless a soft error occurs either in the pattern generator, the chain of the FIFOs, or the routing between those modules and the pattern checker. If the incremented data word and the current one are different, then an error pulse is asserted on the $COMP\_ERR$ output and the corresponding error counter is incremented. Outputs

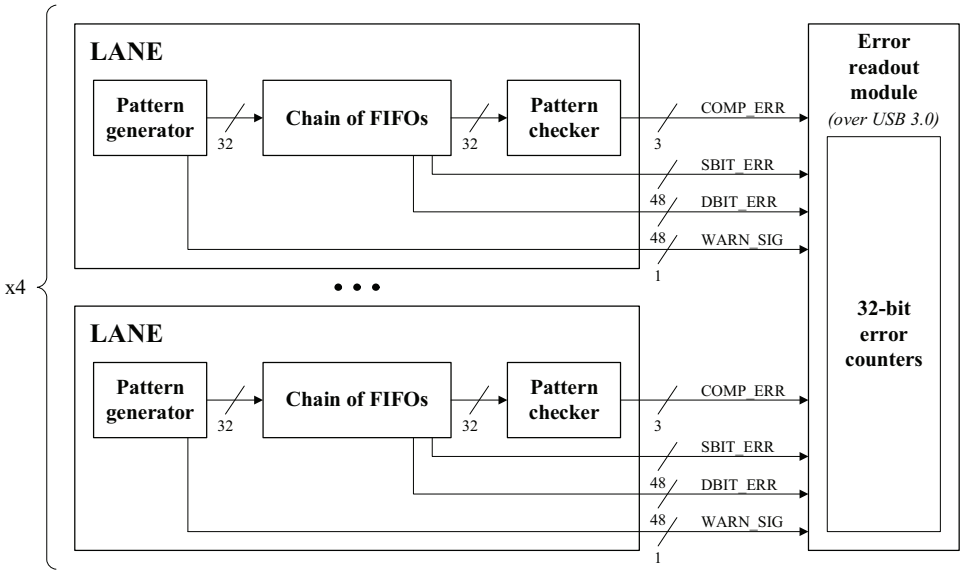


Figure 7.13: Architecture of BRAM testing design.

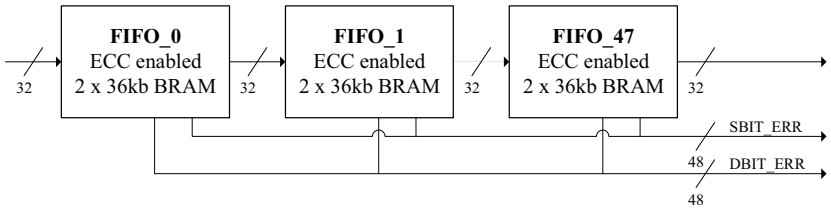


Figure 7.14: Architecture of chain of 48 FIFOs.

from the ECC mechanism informing of single bit upset ($SBIT\_ERR$) or double bit upset ($DBIT\_ERR$) are monitored. Also, the voter warning signals from the data generators are logged. Error counters are periodically read out over USB interface. The utilization of resources by the testing design is presented in Tab. 7.6. The implemented testing design utilizes 88 % of the available Block RAM in the Xilinx Kintex-7 325T.

Lanes	Used resources per design						Essential bits	
	LUT	LUT per lane	LUT RAM	FF	FF per lane	BRAM	Testing design	Lane
4	18 %	4.5 %	1 %	17 %	4.25 %	88 %	15.1 %	3.775 %

Table 7.6: Utilization of resources, in Xilinx Kintex-7 325T, by testing design (values presented as percentages of the total amount of resources in device).

7.3.2 Proton irradiation testing results

The testing design was evaluated in a series of irradiation experiments. Tests with and without a CRAM scrubber were conducted. Tests without a CRAM scrubber were conducted at the flux of $10^7\,\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$. It was calculated that, during the 120 s long irradiation test at the flux of $10^7\,\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$ (fluence equals $1.2 \cdot 10^9\,\mathrm{cm}^{-2}$), on average 408 and 100 errors are induced to the CRAM and BRAM, respectively. Data obtained during testing without a CRAM scrubber are presented in Tab. 7.7. In each test, the ECC mechanism reported the number of successfully corrected single-bit errors. The ECC mechanism did not report any multi-bit error in the data read from the chain of FIFOs.

Tests with CRAM scrubbers were conducted at the flux of $10^6\,\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$. The hybrid scrubber described in Appendix C and the JCM scrubber were employed. It was calculated that, during the 480 s long irradiation test at the flux of $10^6\,\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$ (fluence equals $0.48 \cdot 10^9\,\mathrm{cm}^{-2}$), on average 163.2 and 40 errors are induced to the CRAM and BRAM, respectively. Data obtained during testing with CRAM scrubbers are presented in Tab. 7.8 and Tab. 7.9.

7.3.3 Discussion

The testing design allowed for the experimental evaluation of the embedded Hamming Error Injection and Correction Checking mechanism. Also, BRAM cross-section was calculated (Appendix B.1) and compared with other measurements available in the literature.

The Hamming ECC circuitry managed to successfully correct all of the single-bit errors, and did not report any multi-bit errors in the data read from the chain of FIFOs. Thus, it was deemed robust enough to be employed in the final FPGA design. However, detailed analysis of errors reported by the pattern checker on the *COMP_ERR* output shows that input and output routing from the BRAM is susceptible to radiation and additional mitigation methods must be implemented. Detailed descriptions of available solutions that can be employed are presented in [176].

Test	Flux ($cm^{-2}\,s^{-1}$)	Time (s)	Errors I[a]	Errors II[b]
TEST_2, run_0	$1 \cdot 10^7$	120	91	0
TEST_2, run_1	$1 \cdot 10^7$	120	72	0
TEST_2, run_2	$1 \cdot 10^7$	120	77	0
TEST_2, run_3	$1 \cdot 10^7$	120	80	0
TEST_2, run_4	$1 \cdot 10^7$	120	82	0
TEST_2, run_8	$1 \cdot 10^7$	120	71	0
TEST_2, run_9	$1 \cdot 10^7$	120	68	0
TEST_2, run_10	$1 \cdot 10^7$	120	83	0
TEST_2, run_13	$1 \cdot 10^7$	120	65	0
TEST_2, run_14	$1 \cdot 10^7$	120	43	0
TEST_2, run_16	$1 \cdot 10^7$	120	79	0
TEST_2, run_17	$1 \cdot 10^7$	120	100	0
TEST_2, run_19	$1 \cdot 10^7$	120	96	0
TEST_2, run_20	$1 \cdot 10^7$	120	96	0
TEST_2, run_21	$1 \cdot 10^7$	120	88	0
TEST_2, run_22	$1 \cdot 10^7$	120	85	0
TEST_2, run_23	$1 \cdot 10^7$	120	74	0
TEST_2, run_24	$1 \cdot 10^7$	120	68	0

[a] Single-bit BRAM error repaired by the ECC mechanism.

[b] Multi-bit BRAM error not repaired by the ECC mechanism.

Table 7.7: Data obtained during beam test for testing design without CRAM scrubber employed.

Test	Flux	Time	Errors		Active scrubber			
			I[a]	II[b]	SBU[c]	MBU[d]	CRC[e]	Scrubber lifetime[f]
	($cm^{-2} s^{-1}$)	(s)						(s)
TEST_0, run_0	$1 \cdot 10^6$	480	39	0	154	7	0	480
TEST_0, run_1	$1 \cdot 10^6$	480	40	0	179	10	0	480
TEST_0, run_2	$1 \cdot 10^6$	480	40	0	111	6	1	337.54
TEST_0, run_3	$1 \cdot 10^6$	480	34	0	83	2	1	227.69
TEST_0, run_5	$1 \cdot 10^6$	480	37	0	21	3	1	41.87
TEST_0, run_6	$1 \cdot 10^6$	480	28	0	164	8	0	480
TEST_0, run_7	$1 \cdot 10^6$	480	45	0	141	7	0	480
TEST_0, run_8	$1 \cdot 10^6$	480	30	0	165	12	0	480
TEST_0, run_9	$1 \cdot 10^6$	480	38	0	156	4	0	480
TEST_0, run_10	$1 \cdot 10^6$	480	29	0	174	6	0	480
TEST_0, run_11	$1 \cdot 10^6$	480	34	0	156	0	0	480
TEST_0, run_12	$1 \cdot 10^6$	480	42	0	115	7	1	298.97
TEST_0, run_13	$1 \cdot 10^6$	480	41	0	40	0	0	480
TEST_0, run_14	$1 \cdot 10^6$	480	33	0	148	4	0	480

[a] Single-bit BRAM error repaired by ECC mechanism.

[b] Multi-bit BRAM error not repaired by ECC mechanism.

[c] Single Event Upset (number of SBUs before SEM IP reported CRC error).

[d] Multiple Bit Upset (number of MBUs before SEM IP reported CRC error).

[e] Cyclic Redundancy Check (CRC) error reported by SEM IP.

[f] Time after which SEM IP reported CRC error and suspended its operation.

Table 7.8: Data obtained during beam test of testing design with active scrubber employed.

Test	Flux ($cm^{-2}\,s^{-1}$)	Time (s)	Errors I[a]	Errors II[b]
TEST_1, run_0	$1 \cdot 10^6$	480	32	0
TEST_1, run_1	$1 \cdot 10^6$	480	45	0
TEST_1, run_2	$1 \cdot 10^6$	480	27	0
TEST_1, run_3	$1 \cdot 10^6$	480	35	0
TEST_1, run_4	$1 \cdot 10^6$	480	29	0
TEST_1, run_5	$1 \cdot 10^6$	480	33	0
TEST_1, run_6	$1 \cdot 10^6$	480	35	0
TEST_1, run_7	$1 \cdot 10^6$	480	27	0
TEST_1, run_8	$1 \cdot 10^6$	480	31	0
TEST_1, run_9	$1 \cdot 10^6$	480	42	0
TEST_2, run_0	$1 \cdot 10^7$	120	85	0
TEST_2, run_0	$1 \cdot 10^7$	120	75	0
TEST_2, run_0	$1 \cdot 10^7$	120	88	0
TEST_3, run_1	$1 \cdot 10^7$	120	116	0

[a] Single-bit BRAM error repaired by the ECC mechanism.
[b] Multi-bit BRAM error not repaired by the ECC mechanism.

Table 7.9: Data obtained during beam test for testing design with JCM scubber employed.

7.4 Finite State Machine testing

Finite State Machines (FSMs) are used to control operational flow in logic circuits. As they greatly simplify the design and verification process, they are widely implemented in synchronous designs [177]. Synchronous FSMs store their current state in DFFs and utilize combinational circuits to implement next state transition logic. As SRAM-based FPGAs are susceptible to ionizing and non-ionizing radiation, operation of FSMs implemented in those devices can be compromised by SEEs and may lead to an undesirable condition when an FSM transitions to a wrong state. FSM functional errors can occur from either data-path SETs, DFF SEUs, clock or reset SETs, SEEs in the next state transition circuit, or SEUs in a circuit routing [177].

FSM mitigation methods based on TMR and Error Detection and Correction (EDAC) have already been studied in [177–180]. However, the presented testing designs were not implemented in the target SRAM-based FPGA and not all of the available mitigation methods were investigated. Thus, the main motivation for implementing the testing FSM designs and conducting the experimental testing is to quantitatively compare FSM mitigation methods and evaluate their performance.

7.4.1 Testing design architecture and theory of its operation

The FSM testing design is based on the methodology explained in Chapter 4. Its architecture is presented in Fig. 7.15. The design consists of many identical modules, called lanes, and an error readout module. As many lanes as possible were implemented in each testing design to maximize the utilization of the available FPGA logic resources. The testing designs operate at a frequency of 100 MHz. In order to reduce the number of pattern generators, each of them simultaneously writes test vectors to 4 lanes. Fig. 7.16 shows the 12-state test sequence that is continuously outputted by the pattern generators. Each lane consists of a chain of 458 FSMs and a triplicated pattern checker. Four different FSM architectures were implemented to evaluate the following mitigation methods:

- no mitigation (Fig. 7.18),
- the Hamming code encoding (Fig. 7.19),
- TMR (Fig. 7.20),
- TMR with redundant voters (Fig. 7.21).

The implemented Hamming mechanism operates in a single error correct (SEC) mode. In the evaluated FSMs, a combinational circuit generating output signals is not implemented, because the current 3-bit state code is outputted. All of the investigated FSMs perform the same logic operation whose state diagram is shown in Fig. 7.17.

After the reset signal is deasserted, the pattern generators start outputting the defined

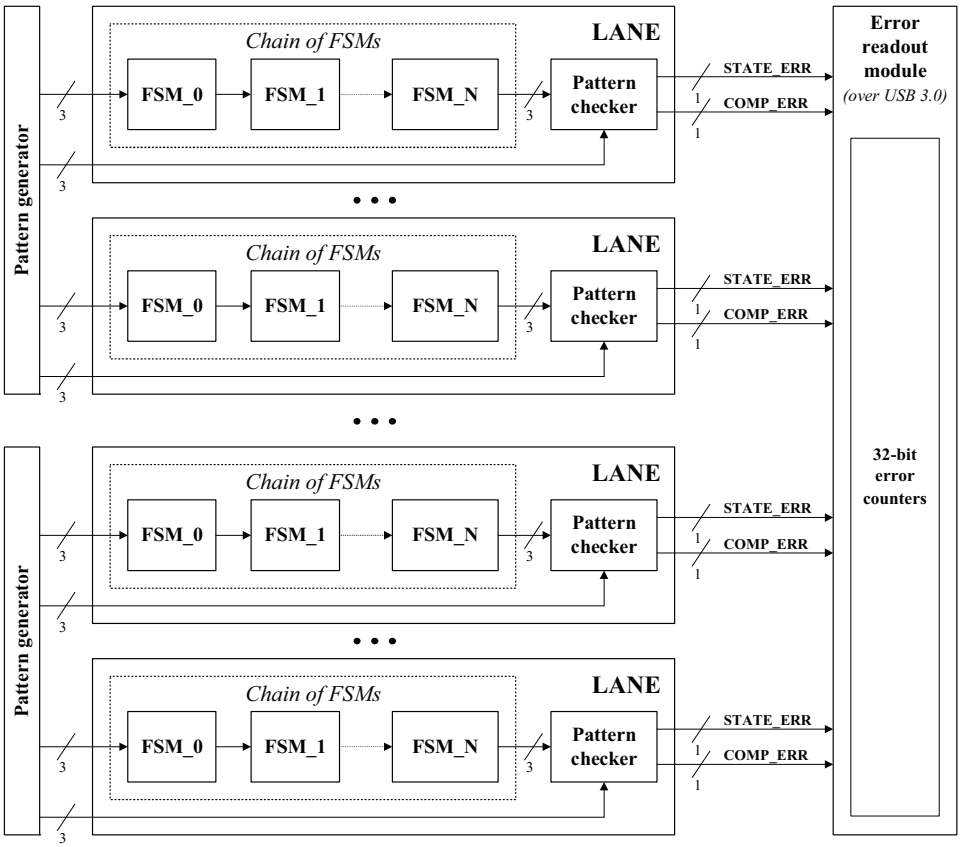


Figure 7.15: FSM testing design architecture.

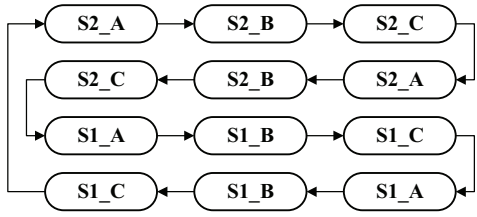


Figure 7.16: 12-state test sequence generated by pattern generator.

3-bit test sequences (Fig. 7.16). The first FSMs in the chains read the test sequences and transition to the next state according to the state flow diagram presented in Fig. 7.17. They simultaneously output their current state code which is read by the next FSMs in the chains. Then, the test sequences are propagated through the chains of the FSMs. As the generated test sequence has 12 states, and each chain consists of 458 FSMs, 38 repetitions of the test sequence are required so that the data outputted by the chains are the same

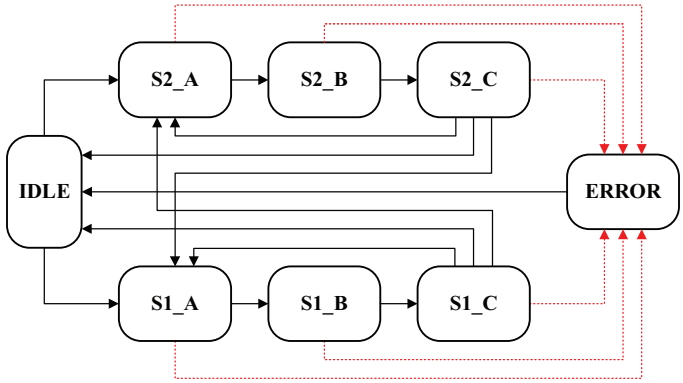


Figure 7.17: FSM state diagram.

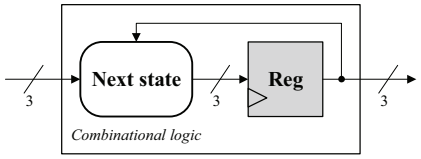


Figure 7.18: FSM architecture with no additional mitigation method employed.

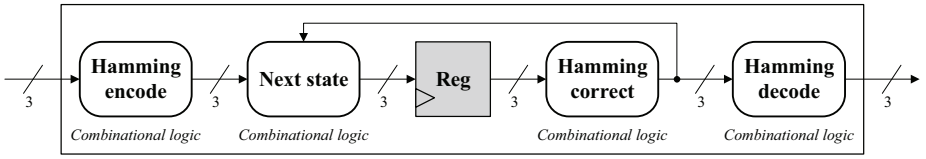


Figure 7.19: FSM architecture with Hamming state encoding.

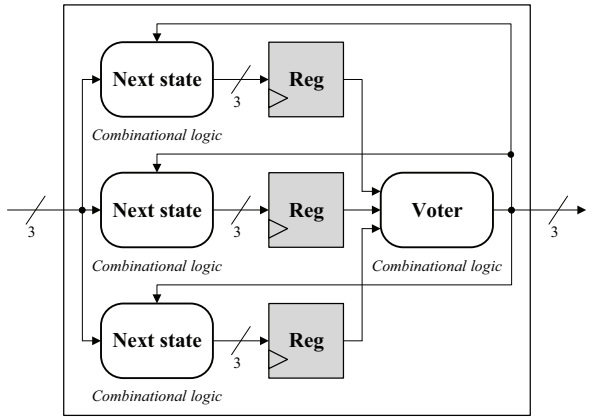


Figure 7.20: Triplicated FSM architecture.

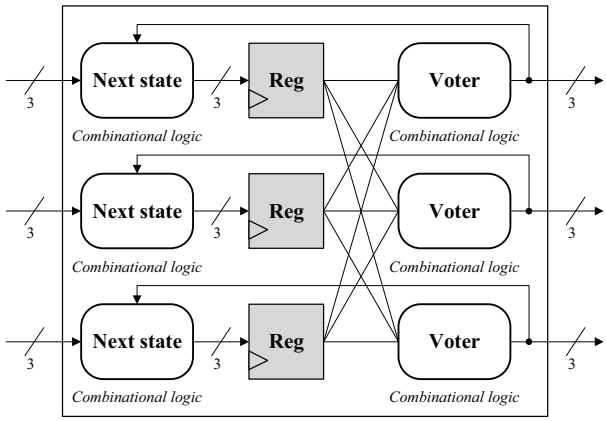


Figure 7.21: Triplicated FSM architecture with redundant voters.

as the test vectors outputted by the pattern generators. Unless a soft error occurs either in the pattern generator, the chain of FSMs, or the routing between those modules and the pattern checker, the data outputted by the chains are the same as the test vectors outputted by the pattern generators. If an FSM or the routing between FSMs are affected by a soft error or an invalid input data is read, the FSM transitions into the error state and outputs the error state code. Then, the next FSM in the chain reads the error state code (outputted by the previous FSM) and transitions into the error state and outputs the error state code. Such a mechanism allows for the propagation on the error state code by the remaining FSMs in the chain up to the pattern checker. The pattern checkers at the end of each lane compare the output from the chains to the reference outputs from the pattern generators. If a discrepancy is found, the pattern checker reports either a state or comparison error. The state error is asserted on the $STATE\_ERR$ output when the error state code is read from the chain of the FSMs. On the other hand, the comparison error is asserted on the $COMP\_ERR$ output when the inconsistency is found in the outputted state sequence. When error signals are asserted either on $STATE\_ERR$ or $COMP\_ERR$ outputs, the corresponding error counters are incremented. During beam or fault injection tests, the values stored in the error counters are periodically read out via a USB interface and saved. During later analysis, it is possible to recognize how many lanes were affected by soft errors or whether the error was in the pattern generator or error readout module. The test runs, in which the pattern generator or the error readout module fail, are excluded from the analysis.

The summaries of the utilized logic resources by different FSMs and testing designs are shown in Tab. 7.10 and Tab. 7.11, respectively. The FSM implemented with TMR with triplicated voters utilizes the most logic resources among the tested architectures.

Logic test structure	Figure	Resources	
		LUT	FF
No mitigation	Fig. 7.18	3	3
Hamming	Fig. 7.19	17	6
TMR	Fig. 7.20	12	9
TMR with triplicated voters	Fig. 7.21	18	9

Table 7.10: Utilization of resources, in Xilinx Kintex-7 325T, by FSMs with different mitigation schemes employed.

Firmware	Lanes	Used resources per design			
		LUT	LUT per lane	FF	FF per lane
No mitigation	128	92 %	**0.72 %**	48 %	**0.38 %**
Hamming	24	93 %	**3.88 %**	18 %	**0.75 %**
TMR	32	88 %	**2.75 %**	34 %	**1.06 %**
TMR with triplicated voters	24	99 %	**4.13 %**	26 %	**1.08 %**

Table 7.11: Utilization of resources, in Xilinx Kintex-7 325T, by FSM testing designs with different mitigation methods employed (values presented as percentages of total amount of resources in device).

7.4.2 Fault injection and proton irradiation testing results

The developed testing designs were evaluated in a series of fault injection and irradiation tests. Experiments with and without a CRAM scrubber were carried out. Tests without a CRAM scrubber were conducted at the flux of $10^7\,\mathrm{cm^{-2}\,s^{-1}}$. It was calculated that, during the 120 s long irradiation test at the flux of $10^7\,\mathrm{cm^{-2}\,s^{-1}}$ (fluence equals $1.2 \cdot 10^9\,\mathrm{cm^{-2}}$), on average 408 and 100 errors are induced to the CRAM and BRAM, respectively. Data obtained without a CRAM scrubber during testing are presented in Tab. 7.12.

Experimental tests with a CRAM scrubber were carried out, as well. It was calculated, that during the 480 s long irradiation test at the flux of $10^6\,\mathrm{cm^{-2}\,s^{-1}}$ (fluence equals $0.48 \cdot 10^9\,\mathrm{cm^{-2}}$), on average 163.2 and 40 errors are induced to the CRAM and BRAM, respectively. The JCM [110] scrubber was employed as a blind scrubber. As it performs a blind scrub cycle every 2 s, the duration of a single irradiation test and beam intensity were set to 480 s and $10^6\,\mathrm{cm^{-2}\,s^{-1}}$ respectively, so that on average an SEU was induced to the CRAM every 2.94 s. This means that the entire CRAM content was refreshed 1.5 times

between occurrences of an SEU. The corresponding fluence of $4.8 \cdot 10^8\,\mathrm{cm}^{-2}$ was 2.5 times less than the fluence of the conducted irradiation tests without the scrubber employed. Data obtained with CRAM scrubbers during testing are presented in Tab. 7.13.

Fault injection	**Errors**	**Lanes**	**Faulty lanes**		**Lane σ**	**Error $\delta\sigma$**
			Mean	SD	$(\mathrm{cm}^2\,\mathrm{lane}^{-1})$	$(\mathrm{cm}^2\,\mathrm{lane}^{-1})$
No mitigation	408	128	30.19	9.20	$\mathbf{1.96 \cdot 10^{-10}}$	$5.99 \cdot 10^{-11}$
Hamming	408	24	3.96	1.82	$1.37 \cdot 10^{-10}$	$6.31 \cdot 10^{-11}$
TMR	408	32	4.04	1.86	$1.05 \cdot 10^{-10}$	$4.85 \cdot 10^{-11}$
TMR with triplicated voters	408	24	0.23	0.42	$\mathbf{8.10 \cdot 10^{-12}}$	$1.47 \cdot 10^{-11}$
Proton irradiation tests	**Fluence** (cm^{-2})					
No mitigation	$1.2 \cdot 10^9$	128	39.92	11.26	$\mathbf{2.59 \cdot 10^{-10}}$	$7.33 \cdot 10^{-11}$
Hamming	$1.2 \cdot 10^9$	24	6.30	2.51	$2.19 \cdot 10^{-10}$	$8.70 \cdot 10^{-11}$
TMR	$1.2 \cdot 10^9$	32	5.17	3.12	$2.06 \cdot 10^{-10}$	$12.5 \cdot 10^{-11}$
TMR with triplicated voters	$1.2 \cdot 10^9$	24	0.21	0.42	$\mathbf{7.44 \cdot 10^{-12}}$	$1.45 \cdot 10^{-11}$

Table 7.12: Comparison of lane cross-sections obtained from fault injection and proton irradiation tests without CRAM scrubber.

7.4.3 Discussion

Four different FSM testing designs were experimentally evaluated via fault injection and proton irradiation tests. Analyzing the obtained experimental data in Tab. 7.12 and Tab. 7.13, a clear conclusion can be drawn that the FSM implemented with TMR with triplicated voters is the least susceptible to soft errors at the cost of the highest utilization of logic resources.

Obtained experimental results in Tab. 7.13 show that employment of the scrubber did not bring significant improvement. Such an outcome was anticipated. Once an FSM loses its synchronization, the scrubbing does not restore the correct operation state, because the feedback and synchronization paths were not implemented. Further increase of radiation hardness can be achieved by implementing a synchronization mechanism between the registers holding the state data [181]. Also, as scrubbing restores the correct content of the configuration memory, it is recommended to be employed.

Fault injection (SEM IP scrubber)	**Errors**	**Lanes**	**Faulty lanes** Mean	SD	**Lane σ** ($\mathrm{cm^2\,lane^{-1}}$)	**Error $\delta\sigma$** ($\mathrm{cm^2\,lane^{-1}}$)
No mitigation	408	128	30.52	5.16	$\mathbf{1.99 \cdot 10^{-10}}$	$3.35 \cdot 10^{-11}$
Hamming	408	24	2.91	1.80	$1.01 \cdot 10^{-10}$	$6.24 \cdot 10^{-11}$
TMR	408	32	3.82	2.16	$9.93 \cdot 10^{-11}$	$5.61 \cdot 10^{-11}$
TMR with triplicated voters	408	24	0.21	0.41	$\mathbf{7.15 \cdot 10^{-12}}$	$1.41 \cdot 10^{-11}$
Proton irradiation tests (JCM scrubber)	**Fluence** ($\mathrm{cm^{-2}}$)					
No mitigation	$0.48 \cdot 10^{9}$	128	16.07	5.90	$\mathbf{2.61 \cdot 10^{-10}}$	$6.57 \cdot 10^{-11}$
Hamming	$0.48 \cdot 10^{9}$	24	3.00	1.52	$2.60 \cdot 10^{-10}$	$1.32 \cdot 10^{-10}$
TMR	$0.48 \cdot 10^{9}$	32	2.15	1.28	$1.40 \cdot 10^{-10}$	$8.33 \cdot 10^{-11}$
TMR with triplicated voters	$0.48 \cdot 10^{9}$	24	0.21	0.43	$\mathbf{1.86 \cdot 10^{-11}}$	$3.69 \cdot 10^{-11}$

Table 7.13: Comparison of lane cross-sections obtained from fault injection proton irradiation tests with CRAM scrubbers.

7.5 Discussion

Testing designs based on the methodology presented in Chapter 4 were developed to evaluate the combinational and sequential circuits, Block RAM, and Finite State Machines. Next, they were experimentally tested via fault injection and irradiation, and cross-sections of the lanes employing different mitigation methods were calculated.

To quantitatively visualize the improvement brought by radiation mitigation methods in case of the ALICE ITS Readout System, mean times between failures (MTBFs) of the testing designs implemented in 192 Xilinx Kintex-7 FPGAs and operating in the target radiation environment were calculated. Tab. 7.14 shows the MTBFs of the testing designs embedding different logic structures that were implemented with and without mitigation methods.

Without mitigation methods, the MTBF of the testing designs containing combinational and sequential circuits equals 0.11 h. However, if mitigation methods are employed and the full chains of LTSs are triplicated, the MTBF draws out to 12.56 h. Whereas, if mitigation methods are employed in the testing designs containing FSMs, then the MTBF extends from 1.82 h to 50.59 h.

	MTBF	
Logic structure	**Without mitigation**	**With mitigation**
Combinational and sequential circuits (Section 7.2)	**0.11 h**	**12.56 h**
Finite State Machines (Section 7.4)	**1.82 h**	**50.59 h**

Table 7.14: Mean Time Between Failures (MTBF) for testing designs, described in Sections 7.2 and 7.4, operating in target radiation environment of ALICE Inner Tracking System Readout.

A final FPGA design implementation guideline based on the obtained experimental results is depicted in Tab. 7.15. First, it is recommended that combinational and sequential circuits are implemented with Block-TMR and CRAM scrubbing. What is more, it has been shown that TMR and CRAM scrubbing bring the greatest improvement in radiation hardness in case of an FSM implementation. Furthermore, it was experimentally evaluated that the embedded ECC mechanism can be employed in the final FPGA design. However, circuits that interface with the Block RAM must be implemented with mitigation methods.

Structure	**Recommendation**
Combinational and sequential circuits	Block-TMR & scrubbing
Finite State Machines	TMR with redundant voters & scrubbing
Block RAM	Embedded ECC protection mechanism is reliable, however writing and reading circuits must be implemented with mitigation methods

Table 7.15: Final FPGA design implementation guideline based on obtained experimental results.

Chapter 8

Summary

Mitigation methods increasing radiation hardness of non-radiation-hardened SRAM-based FPGAs are the subject of this doctoral thesis. The dissertation is of a highly practical and experimental nature, and it delivers key design information for the realization of the ALICE Inner Tracking System Readout.

The hypothesis stating that it is possible to design and deploy the readout system employing commercial, non-radiation-hardened SRAM-based FPGAs for the upgraded ALICE Inner Tracking System, has been proven by successful realization of the two partial objectives described in Section 3.2. First, the design functional error rate estimation methodology was elaborated and its practical utility was shown by the implementation and evaluation of various testing designs. Secondly, models of various radiation mitigation methods based on spatial redundancy, refreshing of configuration memory of SRAM-based FPGAs, and triplication of inputs and outputs were evaluated and quantitatively compared. The basic building blocks of an SRAM-based FPGA design (combinational and sequential circuits, Block RAM, Finite State Machines) were implemented with various mitigation methods employed. Then, they were experimentally evaluated via fault injection and irradiation tests. Finally, an implementation guideline based on experimental results was formulated for the application in the FPGA-based Readout of the upgraded ALICE Inner Tracking System.

The main novelty brought by this doctoral thesis is the design functional error rate estimation methodology. Improvements brought by utilization of the aforementioned methodology while designing the Readout System for the upgraded ALICE Inner Tracking System are presented in Tab. 8.1. First of all, a guideline for implementation of basic building blocks of SRAM-based FPGA design was prepared. It gives hints about the way the basic building blocks should be implemented to be the least radiation susceptible. Secondly, a guideline about the employment of radiation mitigation methods was proposed. It suggests what kind of methods should be employed in order not to unneces-

sarily complicate the system design. Finally, a consistent testing methodology and tools were provided so that radiation susceptibility of the FPGA design can be evaluated.

	Without methodology	**With methodology**
Implementation of basic logic structures of SRAM-based FPGA design	**No guideline** Improper implementation can decrease radiation resilience	**Guideline provided** together with experimental results
Employment of radiation mitigation methods	**No guideline** Improper implementation can unnecessarily complicate system design	**Guideline provided** together with experimental results
Testing methodology	**No consistent testing methodology**	**Testing methodology and tools provided**

Table 8.1: Improvements brought by utilization of design functional error rate estimation methodology.

The presented dissertation strongly contributes to the subject of utilization of SRAM-based FPGA devices in radiation environments. Although the target application was the Readout System of the particle detector at the HEP experiment, the developed methodology can be used at any system employing SRAM-based FPGAs that operates in a radiation environment.

The list of author's contributions to the ALICE Inner Tracking System Upgrade Project include the following achievements:

- design and evaluation of the Prototype Readout Unit,
- elaboration of the methodology allowing for estimation of functional error rates of SRAM-based FPGA design modules,
- evaluation and quantitative comparison of various SRAM-based FPGA mitigation methods,
- testing methodology via fault injection or particle irradiation,
- guideline for the final SRAM-based FPGA design implementation.

References

[1] L. Evans and P. Bryant, "LHC Machine," *Journal of Instrumentation*, vol. 3, no. 08, p. S08001, 2008.

[2] ATLAS Collaboration, "The ATLAS Experiment at the CERN Large Hadron Collider," *Journal of Instrumentation*, vol. 3, no. 08, p. S08003, 2008.

[3] CMS Collaboration, "The CMS experiment at the CERN LHC," *Journal of Instrumentation*, vol. 3, no. 08, p. S08004, 2008.

[4] S. Weinberg, *The Quantum Theory of Fields, Volume 3: Supersymmetry*. Cambridge University Press, 1999.

[5] P. W. Higgs, "Broken symmetries, massless particles and gauge fields.," *Phys. Lett.*, vol. 12, p. 132, 1964.

[6] P. W. Higgs, "Broken symmetries and the masses of gauge bosons.," *Phys. Rev. Lett.*, vol. 13, pp. 508–509, 1964.

[7] LHCb Collaboration, "The LHCb Detector at the LHC," *Journal of Instrumentation*, vol. 3, no. 08, p. S08005, 2008.

[8] ALICE Collaboration, "The ALICE experiment at the CERN LHC," *Journal of Instrumentation*, vol. 3, no. 08, p. S08002, 2008.

[9] J. Bartke, *Introduction to relativistic heavy ion physics*. Singapore: World Scientific, 2009.

[10] Eilhard Haseloff, *Latch-Up, ESD, and Other Phenomena*, May 2000.

[11] M. Bernardini and K. Foraz, "Long Shutdown 2 @ LHC," 2015.

[12] ALICE Collaboration, *Technical Design Report for the Upgrade of the ALICE Inner Tracking System*, Nov 2013.

[13] G. Aglieri Rinella, "The ALPIDE pixel sensor chip for the upgrade of the ALICE Inner Tracking System," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 2016.

[14] M. Mager, "ALPIDE, the Monolithic Active Pixel Sensor for the ALICE ITS upgrade," *Nuclear Instruments and Methods in Physics Research Section A: Acceler-*

ators, Spectrometers, Detectors and Associated Equipment, vol. 824, pp. 434 – 438, 2016.

[15] *TowerJazz CMOS Image Sensor Technology*, 2016 (accessed February 1, 2016).

[16] Samtec Inc., *FireFly Application Design Guide*.

[17] A. Szczepankiewicz, "Readout of the upgraded ALICE-ITS," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 824, pp. 465 – 469, 2016. Frontier Detectors for Frontier Physics: Proceedings of the 13th Pisa Meeting on Advanced Detectors.

[18] P. Buncic, M. Krzewicki, and P. V. Vyvre, *Technical Design Report for the Upgrade of the Online-Offline Computing System*, Apr 2015.

[19] P. Moreira, A. Marchioro, and K. Kloukinas, "The GBT, a proposed architecture for multi-Gb/s data transmission in high energy physics," *Proceedings of the Topical Workshop on Electronics for Particle Physics TWEPP-07, CERN-2007-07*, pp. 332–336, 2007.

[20] R. García Alía, *Radiation Fields in High Energy Accelerators and their impact on Single Event Effects*. PhD thesis, Oct 2014. Presented 15 Dec 2014.

[21] A. Alici, A. D. Mauro, W. Riegler, and A.Tauro, *Radiation Dose and Fluence in ALICE after LS2*, June 2015.

[22] R. Dubey, *FPGA Devices*. Springer London, 2009.

[23] I. Kuon, R. Tessier, and J. Rose, "FPGA Architecture: Survey and Challenges," *Foundations and Trends® in Electronic Design Automation*, vol. 2, no. 2, pp. 135–253, 2008.

[24] A. A. Aggarwal and D. M. Lewis, "Routing architectures for hierarchical field programmable gate arrays," *IEEE International Conference on Computer Design*, pp. 475–478, October 1994.

[25] V. Betz, J. Rose, and A. Marquardt, *Architecture and CAD for Deep-Submicron FPGAS*. The Springer International Series in Engineering and Computer Science, Springer US, 2012.

[26] S. Brown, R. Francis, J. Rose, and Z. Vranesic, *Field-Programmable Gate Arrays*. The Springer International Series in Engineering and Computer Science, Springer US, 1992.

[27] U. Farooq, Z. Marrakchi, and H. Mehrez, *Tree-based Heterogeneous FPGA Architectures: Application Specific Exploration and Optimization*. SpringerLink : Bücher, Springer New York, 2012.

[28] Xilinx Inc., *Partial Reconfiguration User Guide*, XAPP538 (v1.0) April 4, 2012.

[29] Intel Inc., *Partial Reconfiguration IP Core*, May 8, 2017.

[30] Xilinx Inc., *7 Series FPGAs Configurable Logic Block*, UG474 (v1.8) September 27, 2016.

[31] Intel Inc., *Arria V Device Overview*, AV-51001, December 21, 2015.

[32] Intel Inc., *FPGA Architecture*, WP-01003-1.0 (v1.0), July 2006.

[33] Microsemi Inc., *ProASIC3 FPGA Fabric User's Guide*, 2012.

[34] P. Pavan, R. Bez, P. Olivo, and E. Zanoni, "Flash memory cells-an overview," *Proceedings of the IEEE*, vol. 85, pp. 1248–1271, Aug 1997.

[35] Microsemi Inc., *RTAX-S/SL and RTAX-DSP Radiation-Tolerant FPGAs*, DS2169 Rev. 17, February, 2015.

[36] R. C. Baumann, "Radiation-induced soft errors in advanced semiconductor technologies," *IEEE Transactions on Device and Materials Reliability*, vol. 5, pp. 305–316, Sept 2005.

[37] F. Wang and V. D. Agrawal, "Single Event Upset: An Embedded Tutorial," in *21st International Conference on VLSI Design (VLSID 2008)*, pp. 429–434, Jan 2008.

[38] P. Adell and G. Allen, *Assessing and Mitigating Radiation Effects in Xilinx FPGAs*, February 2008.

[39] ESA Requirements and Standards Division, *Techniques for radiation effects mitigation in ASICs and FPGAs handbook*, 2016.

[40] F. Kastensmidt and P. Rech, *FPGAs and Parallel Architectures for Aerospace Applications*. Springer, 2016.

[41] F. Siegle, T. Vladimirova, J. Ilstad, and O. Emam, "Mitigation of Radiation Effects in SRAM-Based FPGAs for Space Applications," *ACM Comput. Surv.*, vol. 47, pp. 37:1–37:34, Jan. 2015.

[42] J. J. Wang, "Radiation Effects in FPGAs," Actel Corporation, Mountain View, CA 94043, USA.

[43] T. R. Oldham and F. B. McLean, "Total ionizing dose effects in mos oxides and devices," *IEEE Transactions on Nuclear Science*, vol. 50, pp. 483–499, June 2003.

[44] F. B. McLean, H. E. Boesch, Jr., and T. R. Oldham, *Ionizing Radiation Effects in MOS Devices and Circuits*, ch. Electron-Hole Generation, Transport, and Trapping in SiO_2. Wiley, 1989.

[45] F. Schmidt, M. I. of Technology. Department of Aeronautics, and Astronautics, *Fault Tolerant Design Implementation on Radiation Hardened by Design SRAM-Based FPGAs*. 2013.

[46] A. Holmes-Siedle and L. Adams, *Handbook of Radiation Effects*. Oxford science publications, OUP Oxford, 2002.

[47] Xilinx Inc., *Single-Event Upset Mitigation Selection Guide*, XAPP987 (v1.0) March 18, 2008.

[48] Xilinx Inc., *Considerations Surrounding Single Event Effects in FPGAs, ASICs, and Processors*, WP402 (v1.0.1) March 7, 2012.

[49] JEDEC Solid State Technology Association, *Measurement and Reporting of Alpha Particle and Terrestrial Cosmic Ray-Induced Soft Errors in Semiconductor Devices*, JESD89A, October 2006.

[50] M. Berg, *Single Event Upset (SEU) Analysis of Complex Circuits Implemented in Actel RTAX-S Field Programmable Gate Array (FPGA) Devices*.

[51] Xilinx Inc., *Mitigating Single-Event Upsets*, WP395 (v1.1) May 19, 2015.

[52] Y. Shiyanovskii, F. Wolff, and C. Papachristou, "Sram cell design protected from seu upsets," in *2008 14th IEEE International On-Line Testing Symposium*, pp. 169–170, July 2008.

[53] A. Paccagnella, *Single Event Effects: SRAM*, 29.3.2007.

[54] Intel Inc., *Understanding Single Event Functional Interrupts in FPGA Designs*, WP-01207-1.0, September 2013.

[55] National Semiconductor, "Radiation Owner's Manual," March 29, 1999.

[56] J. Barth, "The Radiation Environment," 1999.

[57] H. B. Garrett, *Spacecraft Environment Interactions*, 2011 IEEE NSREC Short Course.

[58] G. Santin, *Radiation environments: space, avionics, ground and below*, 2017 RADECS Short Course.

[59] P. Giubilato, *The novel Ion Electron Emission Microscope at SIRAD*. PhD Thesis, 2005.

[60] R. G. Alía, M. Brugger, S. Danzeca, F. Cerutti, J. P. de Carvalho Saraiva, R. Denz, A. Ferrari, L. L. Foro, P. Peronnard, K. Røed, R. Secondo, J. Steckert, Y. Thurel, I. Toccafondo, and S. Uznanski, "Single event effects in high-energy accelerators," *Semiconductor Science and Technology*, vol. 32, no. 3, p. 034003, 2017.

[61] F. Faccio, "Radiation issues in the new generation of high energy physics experiments," *International Journal of High Speed Electronics and Systems*, vol. 14, no. 02, pp. 379–399, 2004.

[62] M. Brugger, *Radiation Effect, calculation Methods and Radiation Test Challenges in Accelerator Mixed Beam Environments*, 2014 IEEE NSREC Short Course.

[63] 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: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 450, no. 1, pp. 155 – 172, 2000.

[64] J. Fabula, J. Moore, and A. Ware, *Understanding neutron single-event phenomena in FPGAs.*

[65] R. Silberberg, C. H. Tsao, and J. R. Letaw, "Neutron generated single-event upsets in the atmosphere," *IEEE Transactions on Nuclear Science*, vol. 31, pp. 1183–1185, Dec 1984.

[66] R. C. Baumann, *Landmarks in Terrestrial Single-Event Effects*, 2013 IEEE NSREC Short Course.

[67] R. Baumann, *Industrial challenges and trends in terrestrial single-event effects (SEE)*, 2014.

[68] M. Ohlsson, P. Dyreklev, K. Johansson, and P. Alfke, "Neutron single event upsets in sram-based fpgas," in *1998 IEEE Radiation Effects Data Workshop. NSREC 98. Workshop Record. Held in conjunction with IEEE Nuclear and Space Radiation Effects Conference (Cat. No.98TH8385)*, pp. 177–180, Jul 1998.

[69] P. Maillard, M. Hart, J. Barton, P. Jain, and J. Karp, "Neutron, 64 MeV Proton, Thermal Neutron and Alpha Single-Event Upset Characterization of Xilinx 20nm UltraScale Kintex FPGA," in *2015 IEEE Radiation Effects Data Workshop (REDW)*, pp. 1–5, July 2015.

[70] P. Maillard, M. Hart, J. Barton, P. Jain, and J. Karp, "Impact of Temperature and Vcc Variation on 20nm Kintex UltraScale FPGAs Neutron Soft Error Rate," in *2015 IEEE Radiation Effects Data Workshop (REDW)*, pp. 1–5, July 2015.

[71] 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*, vol. 9, no. 01, p. C01025, 2014.

[72] A. Lesea, S. Drimer, J. J. Fabula, C. Carmichael, and P. Alfke, "The rosetta experiment: atmospheric soft error rate testing in differing technology FPGAs," *IEEE Transactions on Device and Materials Reliability*, vol. 5, pp. 317–328, Sept 2005.

[73] Microsemi, *FPGA Reliability and the Sunspot Cycle*, 55900103-1/9.11, September 2011.

[74] J. Gebelein, *FPGA fault tolerance in radiation environments.* PhD Thesis, 2016.

[75] Xilinx Inc., *Device Reliability Report*, UG116 (v10.5.1) December 19, 2016.

[76] M. Stettler, M. Caffrey, P. Graham, and J. Krone, "Radiation effects and mitigation strategies for modern FPGAs," 2004.

[77] H. Quinn, "Radiation effects in reconfigurable FPGAs," *Semiconductor Science and Technology*, vol. 32, no. 4, p. 044001, 2017.

[78] J. M. Johnson and M. J. Wirthlin, "Voter Insertion Algorithms for FPGA Designs Using Triple Modular Redundancy," in *Proceedings of the 18th Annual ACM/SIGDA International Symposium on Field Programmable Gate Arrays*, FPGA '10, (New York, NY, USA), pp. 249–258, ACM, 2010.

[79] F. L. Kastensmidt, L. Carro, and R. A. da Luz Reis, *Fault-tolerance techniques for SRAM-based FPGAs*, vol. 1. Springer, 2006.

[80] F. G. de Lima, *Single Event Upset Mitigation Techniques for Programmable Devices*, 2000.

[81] F. G. de Lima Kastensmidt, G. Neuberger, R. F. Hentschke, L. Carro, and R. Reis, "Designing fault-tolerant techniques for SRAM-based FPGAs," *IEEE Design Test of Computers*, vol. 21, pp. 552–562, Nov 2004.

[82] M. Wirthlin, "High-Reliability FPGA-Based Systems: Space, High-Energy Physics, and Beyond," *Proceedings of the IEEE*, vol. 103, pp. 379–389, March 2015.

[83] M. J. Wirthlin, "FPGAs operating in a radiation environment: lessons learned from FPGAs in space," *Journal of Instrumentation*, vol. 8, no. 02, p. C02020, 2013.

[84] R. Roosta, "A Comparison of Radiation-Hard and Radiation Tolerant FPGAs for Space Applications," 2004.

[85] F. Rittner, R. Glein, T. Kolb, and B. Bernard, "Broadband FPGA payload processing in a harsh radiation environment," in *2014 NASA/ESA Conference on Adaptive Hardware and Systems (AHS)*, pp. 151–158, July 2014.

[86] K. Røed, J. Alme, E. Askeland, E. David, T. Gunji, H. Helstrup, T. Kiss, C. Lippmann, A. Rehman, D. Röhrich, K. Ullaland, A. Velure, and C. Zhao, "Upgrade of the ALICE TPC FEE online radiation monitoring system," *Journal of Instrumentation*, vol. 10, no. 12, p. P12019, 2015.

[87] J. Alme, T. Alt, L. Bratrud, P. Christiansen, F. Costa, E. David, T. Gunji, T. Kiss, R. Langøy, J. Lien, C. Lippmann, A. Oskarsson, A. U. Rehman, K. Røed, D. Röhrich, A. Tarantola, C. Torgersen, I. N. Torsvik, K. Ullaland, A. Velure, S. Yang, C. Zhao, H. Appelshäuser, and L. Österman, "RCU2 — The ALICE TPC readout electronics consolidation for Run2," *Journal of Instrumentation*, vol. 8, no. 12, p. C12032, 2013.

[88] A. Morsch and B. Pastircak, *Radiation in ALICE Detectors and Electronic Racks*, April 2002.

[89] C. Färber, U. Uwer, D. Wiedner, B. Leverington, and R. Ekelhof, "Radiation tolerance tests of SRAM-based FPGAs for the potential usage in the readout electronics

for the LHCb experiment," *Journal of Instrumentation*, vol. 9, no. 02, p. C02028, 2014.

[90] T. Grassi, F. Costanza, T. Karakaya, M. O. Sahin, D. Lincoln, N. Strobbe, D. T. A. Kaminskiy, Y. Wang, and J. F. Hirschauer, *Irradiation test of the HCAL Forward and Endcap upgrade electronics at the CHARM facility at CERN*. Geneva, Feb 2016.

[91] LHCb Collaboration, *LHCb PID Upgrade Technical Design Report*, Nov 2013.

[92] M. Adinolfi, J. Kariuki, K. Petridis, J. Rademacker, P. Garsed, S. Wotton, R. Cardinale, A. Petrolini, D. Clark, U. Egede, M. McCann, M. Patel, T. Savidge, D. Websdale, M. Brock, N. Harnew, M. Tacon, M. Benettoni, A. J. Brummitt, S. Easo, A. Papanestis, S. Ricciardi, F. F. Wilson, C. D'Ambrosio, C. Frei, J. He, and D. Piedigrossi, *LHCb Upgraded RICH 1 Engineering Design Review Report*. Geneva, Jun 2016.

[93] P. Garsed, S. Wotton, R. Cardinale, A. Petrolini, R. Cardinale, A. Petrolini, S. Easo, C. D'Ambrosio, C. Frei, J. He, and D. Piedigrossi, *LHCb Upgraded RICH 2 Engineering Design Review Report*. Geneva, Jun 2016.

[94] V. Friese and C. Sturm, eds., *CBM Progress Report 2011*. Darmstadt: GSI, 2012.

[95] N. Battezzati, L. Sterpone, and M. Violante, *Reconfigurable Field Programmable Gate Arrays for Mission-Critical Applications*. Springer Science & Business Media, 2010.

[96] A. Manuzzato, *Single Event Effects on FPGAs*. PhD Thesis, 2010.

[97] F. Kastensmidt and P. Rech, *Radiation Effects and Fault Tolerance Techniques for FPGAs and GPUs*, pp. 3–17. Springer International Publishing, 2016.

[98] M. Berg, *Field Programmable Gate Array (FPGA) Single Effect (SEE) Radiation Testing*, 2012.

[99] R. C. Baumann, *CMOS Single-Event Effects in Advanced CMOS Technology*, 2005 IEEE NSREC Short Course.

[100] L. Sterpone, M. Violante, R. H. Sorensen, D. Merodio, F. Sturesson, R. Weigand, and S. Mattsson, "Experimental Validation of a Tool for Predicting the Effects of Soft Errors in SRAM-Based FPGAs," *IEEE Transactions on Nuclear Science*, vol. 54, pp. 2576–2583, Dec 2007.

[101] T. Vaňát, J. Pospíšil, F. Křížek, J. Ferencei, and H. Kubátová, "A System for Radiation Testing and Physical Fault Injection into the FPGAs and Other Electronics," in *2015 Euromicro Conference on Digital System Design*, pp. 205–210, Aug 2015.

[102] T. Vaňát, *Physical Fault Injection and Monitoring Methods for Programmable Devices*. PhD thesis, Czech Technical University in Prague, 2017.

[103] R. G. Alía, B. Biskup, M. Brugger, M. Calviani, C. Poivey, K. Røed, F. Saigné, G. Spiezia, and F. Wrobel, "SEU Measurements and Simulations in a Mixed Field Environment," *IEEE Transactions on Nuclear Science*, vol. 60, pp. 2469–2476, Aug 2013.

[104] R. G. Alía, M. Brugger, S. Danzeca, V. Ferlet-Cavrois, C. Poivey, K. Røed, F. Saigné, G. Spiezia, S. Uznanski, and F. Wrobel, "SEE Measurements and Simulations Using Mono-Energetic GeV-Energy Hadron Beams," *IEEE Transactions on Nuclear Science*, vol. 60, pp. 4142–4149, Dec 2013.

[105] J. Tonfat, J. Tarrillo, L. Tambara, F. L. Kastensmidt, and R. Reis, *Multiple Fault Injection Platform for SRAM-Based FPGA Based on Ground-Level Radiation Experiments*, pp. 135–151. FPGAs and Parallel Architectures for Aerospace Applications: Soft Errors and Fault-Tolerant Design: Springer International Publishing, 2016.

[106] J. Tarrillo, J. Tonfat, L. Tambara, F. L. Kastensmidt, and R. Reis, "Multiple fault injection platform for SRAM-based FPGA based on ground-level radiation experiments," in *2015 16th Latin-American Test Symposium (LATS)*, pp. 1–6, March 2015.

[107] H. Guzmán-Miranda, J. Barrientos-Rojas, and M. A. Aguirre, *A Fault Injection technique oriented to SRAM-FPGAs*, pp. 49–59. Springer International Publishing, 2016.

[108] N. A. Harward, M. R. Gardiner, L. W. Hsiao, and M. J. Wirthlin, *A Fault Injection System for Measuring Soft Processor Design Sensitivity on Virtex-5 FPGAs*, pp. 61–74. Cham: Springer International Publishing, 2016.

[109] A. B. Gruwell, *"High-Speed Programmable FPGA Configuration Memory Access Using JTAG*, All Theses and Dissertations. 2015.

[110] A. Gruwell, P. Zabriskie, and M. Wirthlin, "High-speed FPGA configuration and testing through JTAG," in *2016 IEEE AUTOTESTCON*, pp. 1–8, Sept 2016.

[111] M. Alderighi, S. D'Angelo, F. Casini, G. Sorrenti, D. M. Codinachs, and S. Davin, "The FLIPPER Fault Injection Platform: Experiences and Knowledge from a Ten-year Project," in *ARCS 2017; 30th International Conference on Architecture of Computing Systems*, pp. 1–8, April 2017.

[112] K. Røed, K. Ullaland, H. Helstrup, and T. Natås, "Single Event Upsets in SRAM FPGA based readout electronics for the Time Projection Chamber in the ALICE experiment," 2009. Presented on 11 Dec 2009.

[113] K. Roed, D. Rohrich, M. Richter, D. Fehlker, H. Helstrup, J. Alme, and K. Ullaland, "Fault injection as a test method for an FPGA in charge of data readout for a large

tracking detector," *Nucl. Instrum. Methods Phys. Res., A*, vol. 629, no. 1, pp. 260–268, 2011.

[114] F. Brosser and E. Milh, "Seu mitigation techniques for advanced reprogrammable fpga in space," Master's thesis, 2014. 128.

[115] A. Bocquillon, G. Foucard, F. Miller, N. Buard, R. Leveugle, C. Daniel, S. Rakers, T. Carriere, V. Pouget, and R. Velazco, "Highlights of laser testing capabilities regarding the understanding of see in sram based fpgas," in *2007 9th European Conference on Radiation and Its Effects on Components and Systems*, pp. 1–6, Sept 2007.

[116] M. Cannon, A. Perez-Celis, and M. Wirthlin, "Targeting SRAM FPGA Components Using a Two-Photon Absorption Laser," 2014.

[117] F. L. Kastensmidt, *SEE Mitigation Strategies for Digital Circuit Design Applicable to ASIC and FPGAs*, 2007 IEEE NSREC Short Course.

[118] M. Berg, *Trading ASIC and FPGA Considerations for System Insertion*, 2009 IEEE NSREC Short Course.

[119] M. Berg, *New Developments in Error Detection and Correction Strategies for Critical Applications.*

[120] K. S. Morgan, D. L. McMurtrey, B. H. Pratt, and M. J. Wirthlin, "A Comparison of TMR With Alternative Fault-Tolerant Design Techniques for FPGAs," *IEEE Transactions on Nuclear Science*, vol. 54, pp. 2065–2072, Dec 2007.

[121] S. Habinc, *Functional Triple Modular Redundancy (FTMR). VHDL Design Methodology for Redundancy in Combinatorial and Sequential Logic*, Design and Assessment Report, Gaisler Research FPGA-003-01, Version 0.2, December 2002.

[122] Xilinx Inc., *Triple Module Redundancy Design Techniques for Virtex FPGAs*, XAPP197 (v1.0.1) July 6, 2006.

[123] M. Berg, *Actel ProASIC A3PE3000L-PQ208 Field Programmable Gate Array Single Event Effects (SEE) High-Speed Test Plan - Phase II*, 2011.

[124] M. Berg and K. LaBel, *Verification of Triple Modular Redundancy Insertion for Reliable and Trusted Systems*, 2016.

[125] M. Berg, *Single Event Upset Test and Qualification for FPGA Devices.*

[126] M. Berg, *Single Event Effects in FPGA Devices 2015-2016.*

[127] L. Rui and K. Yan-jia, "A method of synchronous-feedback based state machine with triple modular redundancy," in *Proceedings of 2014 IEEE Chinese Guidance, Navigation and Control Conference*, pp. 136–139, Aug 2014.

[128] J. R. Azambuja, F. Sousa, L. Rosa, and F. L. Kastensmidt, "Evaluating large grain TMR and selective partial reconfiguration for soft error mitigation in SRAM-based FPGAs," in *2009 15th IEEE International On-Line Testing Symposium*, pp. 101–106, June 2009.

[129] F. L. Kastensmidt, L. Sterpone, L. Carro, and M. S. Reorda, "On the optimal design of triple modular redundancy logic for SRAM-based FPGAs," in *Design, Automation and Test in Europe*, pp. 1290–1295 Vol. 2, March 2005.

[130] K. Iniewski, *Radiation Effects in Semiconductors.* Devices, Circuits, and Systems, CRC Press, 2010.

[131] I. Herrera-Alzu and M. Lopez-Vallejo, "Design Techniques for Xilinx Virtex FPGA Configuration Memory Scrubbers," *IEEE Transactions on Nuclear Science*, vol. 60, pp. 376–385, Feb 2013.

[132] Xilinx Inc., *Vivado Design Suite User Guide: Partial Reconfiguration*, UG909 (v2016.1) April 6, 2016.

[133] Intel Inc., *Increasing Design Functionality with Partial and Dynamic Reconfiguration in 28-nm FPGAs*, WP-01137-1.0, July 2010.

[134] Xilinx Inc., *Correcting Single-Event Upsets Through Virtex Partial Configuration*, XAPP216 (v1.0) June 1, 2000.

[135] Xilinx Inc., *Correcting Single-Event Upsets in Virtex-II Platform FPGA Configuration Memory*, XAPP779 (v1.1) February 19, 2007.

[136] Xilinx Inc., *UltraScale Architecture Configuration*, UG570 (v1.7) March 15, 2017.

[137] Xilinx Inc., *7 Series FPGAs Configuration User Guide*, UG470 (v1.11) September 27, 2016.

[138] A. G. Stoddard, *Configuration Scrubbing Architectures for High-Reliability FPGA Systems*, All Theses and Dissertations. 2015.

[139] Xilinx Inc., *Single-Event Upset Mitigation for Xilinx FPGA Block Memories*, XAPP962 (v1.1) March 14, 2008.

[140] J. L. T. Seclen, *Frame-Level Redundancy Scrubbing Technique for SRAM-based FPGAs.* PhD Thesis, 2015.

[141] Xilinx Inc., *Soft Error Mitigation Controller v4.1*, PG036 September 30, 2015.

[142] Xilinx Inc., *UltraScale Architecture Soft Error Mitigation Controller v3.1*, PG187 April 4, 2018.

[143] J. Tonfat, F. Kastensmidt, and R. Reis, "Energy efficient frame-level redundancy scrubbing technique for SRAM-based FPGAs," in *2015 NASA/ESA Conference on Adaptive Hardware and Systems (AHS)*, pp. 1–8, June 2015.

[144] J. Tonfat, F. L. Kastensmidt, P. Rech, R. Reis, and H. M. Quinn, "Analyzing the Effectiveness of a Frame-Level Redundancy Scrubbing Technique for SRAM-based FPGAs," *IEEE Transactions on Nuclear Science*, vol. 62, pp. 3080–3087, Dec 2015.

[145] J. Heiner, B. Sellers, M. Wirthlin, and J. Kalb, "FPGA partial reconfiguration via configuration scrubbing," in *2009 International Conference on Field Programmable Logic and Applications*, pp. 99–104, Aug 2009.

[146] D. R. Czajkowski, P. K. Samudrala, and M. P. Pagey, "Seu mitigation for reconfigurable FPGAs," in *2006 IEEE Aerospace Conference*, pp. 7 pp.–, 2006.

[147] M. Berg, C. Poivey, D. Petrick, D. Espinosa, A. Lesea, K. LaBel, M. Friendlich, H. Kim, and A. Phan, "Effectiveness of internal vs. external SEU scrubbing mitigation strategies in a Xilinx FPGA: Design, test, and analysis," in *2007 9th European Conference on Radiation and Its Effects on Components and Systems*, pp. 1–8, Sept 2007.

[148] M. Berg, C. Poivey, D. Petrick, D. Espinosa, A. Lesea, K. A. LaBel, M. Friendlich, H. Kim, and A. Phan, "Effectiveness of Internal Versus External SEU Scrubbing Mitigation Strategies in a Xilinx FPGA: Design, Test, and Analysis," *IEEE Transactions on Nuclear Science*, vol. 55, pp. 2259–2266, Aug 2008.

[149] S. Manz, J. Gebelein, A. Oancea, H. Engel, and U. Kebschull, "Radiation mitigation efficiency of scrubbing on the fpga based cbm-tof read-out controller," in *2013 23rd International Conference on Field programmable Logic and Applications*, pp. 1–6, Sept 2013.

[150] Xilinx Inc., *Virtex-4 Family Overview*, DS112 (v3.1) August 30, 2010.

[151] Microsemi Inc., *Automotive ProASIC3 Flash Family FPGAs*, Revision 5, January 2013.

[152] Xilinx Inc., *UltraScale Architecture and Product Data Sheet: Overview*, DS890 (v2.11) February 15, 2017.

[153] Microsemi Inc., *ProASIC3L Low Power Flash FPGAs with Flash Freeze Technology*, DS0100, June 2015.

[154] K. Røed, J. Alme, D. Fehlker, M. Richter, K. Ullaland, H. Helstrup, and D. Röhrich, "Radiation tolerance of an SRAM based FPGA used in a large tracking detector," in *Large Scale Applications and Radiation Hardness of Semiconductor Detectors*, p. 43, 2009.

[155] Xilinx Inc., *Virtex-II Pro and Virtex-II Pro X Platform FPGAs: Complete Data Sheet*, DS083 (v5.0) June 21, 2011.

[156] Microsemi Inc., *ProASIC Flash Family FPGAs*, v5.8, June 2009.

[157] I. Herrera-Alzu and M. López-Vallejo, "Self-reference Scrubber for TMR Systems Based on Xilinx Virtex FPGAs," in *Integrated Circuit and System Design. Power and Timing Modeling, Optimization, and Simulation* (J. L. Ayala, B. García-Cámara, M. Prieto, M. Ruggiero, and G. Sicard, eds.), pp. 133–142, Springer Berlin Heidelberg, 2011.

[158] G. Alonzo, S. H. Ardalan, and X. Yao, "Fast Local Scrubbing for FPGA's Configuration Memory," 2012.

[159] M. Wirthlin and A. Harding, *Hybrid Configuration Scrubbing for Xilinx 7-Series FPGAs*, pp. 91–101. FPGAs and Parallel Architectures for Aerospace Applications: Soft Errors and Fault-Tolerant Design: Springer International Publishing, 2016.

[160] A. D. Oancea, C. Stuellein, J. Gebelein, and U. Kebschull, "A resilient, flash-free soft error mitigation concept for the CBM-ToF read-out chain via GBT-SCA," in *2015 25th International Conference on Field Programmable Logic and Applications (FPL)*, pp. 1–4, Sept 2015.

[161] A. Caratelli, S. Bonacini, K. Kloukinas, A. Marchioro, P. Moreira, R. D. Oliveira, and C. Paillard, "The GBT-SCA, a radiation tolerant ASIC for detector control and monitoring applications in HEP experiments," *Journal of Instrumentation*, vol. 10, no. 03, p. C03034, 2015.

[162] Xilinx Inc., *7 Series FPGAs SelectIO Resources*, UG471 (v1.4) May 13, 2014.

[163] STMicroelectronics, "Rad hard 65nm CMOS technology platform for space applications," C65SPACE, 2015.

[164] R. Glein, B. Schmidt, F. Rittner, J. Teich, and D. Ziener, "A Self-Adaptive SEU Mitigation System for FPGAs with an Internal Block RAM Radiation Particle Sensor," in *2014 IEEE 22nd Annual International Symposium on Field-Programmable Custom Computing Machines*, pp. 251–258, May 2014.

[165] K. M. Sielewicz, G. Aglieri Rinella, M. Bonora, J. Ferencei, P. Giubilato, M. J. Rossewij, J. Schambach, and T. Vaňát, "Prototype readout electronics for the upgraded ALICE Inner Tracking System," *Journal of Instrumentation*, vol. 12, no. 01, p. C01008, 2017.

[166] Xilinx Inc., *7 Series FPGAs Data Sheet: Overview*, DS180 (v2.2) December 15, 2016.

[167] J. Schambach, M. J. Rossewij, K. M. Sielewicz, G. Aglieri Rinella, M. Bonora, J. Ferencei, P. Giubilato, and T. Vaňát, "ALICE inner tracking system readout electronics prototype testing with the CERN "Giga Bit Transceiver"," *Journal of Instrumentation*, vol. 11, no. 12, p. C12074, 2016.

[168] S. Baron, J. P. Cachemiche, F. Marin, P. Moreira, and C. Soos, "Implementing the GBT data transmission protocol in FPGAs," 2009.

[169] K. M. Sielewicz, G. A. Rinella, M. Bonora, P. Giubilato, M. Lupi, M. J. Rossewij, J. Schambach, and T. Vanat, "Experimental methods and results for the evaluation of triple modular redundancy SEU mitigation techniques with the Xilinx Kintex-7 FPGA," in *2017 IEEE Radiation Effects Data Workshop (REDW)*, pp. 1–7, July 2017.

[170] Digilent Inc., *JTAG-HS3 Programming Cable for Xilinx FPGAs*, 2015.

[171] Xilinx Inc., *Platform Cable USB II*, DS593 (v1.5) June 23, 2015.

[172] Xilinx Inc., *Soft Error Mitigation Using Prioritized Essential Bits*, XAPP538 (v1.0) April 4, 2012.

[173] Xilinx Inc., *7 Series FPGAs Memory Resources*, UG473 (v1.12) September 27, 2016.

[174] M. Cannon, M. Wirthlin, A. Camplani, M. Citterio, and C. Meroni, "Evaluating Xilinx 7 Series GTX Transceivers for Use in High Energy Physics Experiments Through Proton Irradiation," *IEEE Transactions on Nuclear Science*, vol. 62, pp. 2695–2702, Dec 2015.

[175] D. S. Lee, M. Wirthlin, G. Swift, and A. C. Le, "Single-Event Characterization of the 28 nm Xilinx Kintex-7 Field-Programmable Gate Array under Heavy Ion Irradiation," in *2014 IEEE Radiation Effects Data Workshop (REDW)*, pp. 1–5, July 2014.

[176] N. Rollins, M. Fuller, and M. J. Wirthlin, "A Comparison of fault-tolerant memories in SRAM-based FPGAs," in *2010 IEEE Aerospace Conference*, pp. 1–12, March 2010.

[177] M. Berg, *An Analysis of Heavy-Ion Single Event Effects for a Variety of Finite State-Machine Mitigation Strategies*, 2014.

[178] J. Gebelein, *Hamming FSM with Xilinx Blind Scrubbing - Trick or Treat*, 2012.

[179] A. Tiwari and K. A. Tomko, "Enhanced reliability of finite-state machines in FPGA through efficient fault detection and correction," *IEEE Transactions on Reliability*, vol. 54, pp. 459–467, Sept 2005.

[180] G. Burke and S. Taft, *Fault Tolerant State Machines*, 2004.

[181] J. M. Johnson and M. J. Wirthlin, "Voter Insertion Algorithms for FPGA Designs Using Triple Modular Redundancy," in *Proceedings of the 18th Annual ACM/SIGDA International Symposium on Field Programmable Gate Arrays*, FPGA '10, (New York, NY, USA), pp. 249–258, ACM, 2010.

[182] K. Chapman, *A Practical Look at SEU, Effects and Mitigation*, 2014.

[183] D. S. Lee, G. R. Allen, G. Swift, M. Cannon, M. Wirthlin, J. S. George, R. Koga, and K. Huey, "Single-Event Characterization of the 20 nm Xilinx Kintex UltraScale Field-Programmable Gate Array under Heavy Ion Irradiation," in *2015 IEEE Radiation Effects Data Workshop (REDW)*, pp. 1–6, July 2015.

[184] D. M. Hiemstra, V. Kirischian, and J. Brelski, "Single Event Upset Characterization of the Kintex UltraScale Field Programmable Gate Array Using Proton Irradiation," in *2016 IEEE Radiation Effects Data Workshop (REDW)*, pp. 1–5, July 2016.

List of acronyms

ACORDE	ALICE Cosmic Ray Detector
ALICE	A Large Ion Collider Experiment
ALICE O^2	ALICE Online and Offline computing system
ALM	Adaptive Logic Block
ALPIDE	ALICE Pixel Detector
ASIC	Application-Specific Integrated Circuit
ATLAS	A Toroidal LHC Apparatus
BRAM	Block Random-Access Memory
BTMR	Block Triple Modular Redundancy
CBM	Compressed Baryonic Matter
CERN	European Organization for Nuclear Research
CLB	Configurable Logic Block
CMOS	Complementary Metal-Oxide Semiconductor
CMS	Compact Muon Solenoid
CRAM	Configuration Random-Access Memory
CRC	Cyclic Redundancy Check
CTP	Central Trigger Processor
DAQ	Data Acquisition System
DCS	Detector Control System
DD	Displacement Damage
DFF	D-type flip-flop
DLL	Delay-locked loop
DMR	Dual Modular Redundancy
DSP	Digital Signal Processing
DTMR	Distributed Triple Modular Redundancy
DUT	Device Under Test

ECC	Error Correction Code
EDAC	Error Detection and Correction
EEPROM	Electrically Erasable Programmable Read-only Memory
EMCal	Electromagnetic Calorimeter
FF	Flip-flop
FIFO	First In First Out
FMC	FPGA Mezzanine Card
FMD	Forward Multiplicity Detector
FPC	Functionally Predefined Component
FPC	Flex Printed Circuit
FPGA	Field-Programmable Gate Array
FSM	Finite State Machine
GBT	Gigabit Transceiver
GBTx	Gigabit Transceiver Chipset
GEO	Geostationary Orbit
GPIO	General-purpose input/output
GTMR	Global Triple Modular Redundancy
HDL	Hardware Description Language
HEH	High-Energy Hadron
HEP	High-Energy Physics
HIC	Hybrid Integrated Circuit
HLT	High-Level Trigger
HMPID	High-Momentum Particle Identification Detector
HPC	High-Pin Count
I/O	Input-Output
IB	Inner Barrel
ICAP	Internal Configuration Access Port
IP	Interaction Point
ITS	Inner Tracking System
JCM	JTAG Configuration Manager
JTAG	Joint Test Action Group

LB Logic Block
LEO Low Earth Orbit
LET Linear Energy Transfer
LHC Large Hadron Collider
LHCb Large Hadron Collider beauty
LS2 Second Long LHC Shutdown
LTMR Local Triple Modular Redundancy
LTS Logic Test Structure
LUT Look-up Table

MAPS Monolithic Active Pixel Sensor
MBU Multiple Bit Upset
MCU Multiple Cell Upset
MLVDS Multipoint Low-Voltage Differential Signaling
MOSFET Metal-Oxide Semiconductor Field-Effect Transistor
MRI Magnetic resonance imaging
MTBF Mean Time Between Failures

OB Outer Barrel

PB Power Board
PC Personal Computer
PCB Printed Circuit Board
PET Positron emission tomography
PHOS Photon Spectrometer
PLL Phase-locked Loop
PMD Photon Multiplicity Detector
PR Partial Reconfiguration
PSB Proton Synchrotron Booster

QGP Quark-Gluon Plasma

RAM Random-Access Memory
RB Readout Board
RCU Readout Control Unit
RF Radio frequency

RS Readout System
RU Readout Unit

SBU Single Bit Upset
SCA Slow Control Adapter
SDR Software Defined Radio
SEC Single Error Correct
SEC-DED Single Error Correct Double Error Detect
SEE Single Event Effect
SEFI Single Event Functional Interrupt
SEL Single Event Latch-Up
SEM Soft Error Mitigation
SET Single Event Transient
SEU Single Event Upset
SFP Small Form-Factor Pluggable
SoC System-on-a-chip
SPS Super Proton Synchrotron
SRAM Static Random-Access Memory

TID Total Ionising Dose
TMR Triple Modular Redundancy
TNID Total Non-Ionizing Dose
TOF Time-Of-Flight Detector
TPC Time-Projection Chamber
TRD Transition Radiation Detector

UART Universal Asynchronous Receiver-Transmitter
USB Universal Serial Bus

VME Versa Module Eurocard bus

ZDC Zero Degree Calorimeter

Appendix A

SEU rate calculation

The following Appendix shows the calculation of the mean time between SEUs t_{MTBSEU} for two SRAM-based FPGA devices. Tab. A.1 presents the number of BRAM N_{BB} and CRAM N_{CB} bits in Xilinx Kintex-7 XC7K325T and Kintex Ultrascale XCKU060 which was calculated according to [182]. The mean time between SEUs can be calculated using Eq. A.1.

Configuration bits	**XC7K325T**	**XCKU060**
BRAM (N_{BB})	16,404,480	39,813,120
CRAM (N_{CB})	75,144,416	153,186,144
Total	91,548,896	192,999,264

Table A.1: Number of BRAM and CRAM bits in Xilinx Kintex-7 XC7K325T [137, 166] and Kintex Ultrascale XCKU060 [136, 152] FPGAs.

$$t_{\mathrm{MTBSEU}} = \frac{1}{\sigma_{\mathrm{CRAM}} N_{\mathrm{CB}} F} \tag{A.1}$$

The CRAM cross-section σ_{CRAM} for the Kintex-7 XC7K325T equals $4.52 \cdot 10^{-15}\,\mathrm{cm}^{-2}\,\mathrm{bit}^{-1}$ [71, 75, 174, 175]. Whereas, the CRAM cross-section for the Ultrascale XCKU060 equals $2.50 \cdot 10^{-15}\,\mathrm{cm}^{-2}\,\mathrm{bit}^{-1}$ [69, 70, 75, 183, 184].

The HEH flux F at the position where the ALICE Inner Tracking System (ITS) Readout will be installed equals $10^3\,\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$. Thus, the mean time between SEUs t_{MTBSEU} for the Xilinx Kintex-7 XC7K325T and Kintex Ultrascale XCKU060 operating in the target radiation environment equals 2944.20 s and 2611.20 s, respectively. However, as the ALICE ITS Readout will utilize 192 SRAM-based FPGAs, the mean time between SEUs for the Readout System based on the Xilinx Kintex-7 XC7K325T or Kintex Ultrascale XCKU060 decreases to 15.33 s or 13.60 s, respectively.

Appendix B

Radiation testing

B.1 CRAM and BRAM cross-section measurements

The Xilinx Kintex-7 325T [166] was irradiated at the isochronous cyclotron [102] in several radiation campaigns using a 30 MeV proton beam at Linear Energy Transfer (LET) of $1.469 \cdot 10^{-2}\,\mathrm{MeV\,cm^2\,mg^{-1}}$. The measured CRAM cross-section of the FPGA at the given LET equals $(4.52 \pm 0.03) \cdot 10^{-15}\,\mathrm{cm^2\,bit^{-1}}$. Also, the BRAM cross-section was measured to be $(5.07 \pm 0.08) \cdot 10^{-15}\,\mathrm{cm^2\,bit^{-1}}$. The systematic error for the calculated cross-sections is 10 %, due to the uncertainty in the flux measurement. Both measurements are consistent with other tests carried out using this same device [71, 75, 174, 175].

Appendix C

Hybrid scrubber

The custom active scrubber (Fig. C.1), based on [160], consists of the Xilinx Soft Error IP [141] and a dedicated Python-based software running on the PC. The SEM IP operates in the repair mode which means that it corrects configuration memory frames with single-bit errors. It also covers the correction of multi-bit upset events when errors are distributed one per frame as a result of configuration memory interleaving. The software continuously processes the output data sent by the SEM IP over UART interface. All the single bit errors that are reported by the SEM IP (Listing C.1) are logged. Before operating the custom active scrubber, partial bitfiles must be generated. These are bitfiles (Fig. C.2 (b)) which contain only configuration data for one configuration frame. When a multiple-bit bit error is detected, then, based on the logical address in the SEM IP output message (Listing C.2), a correct partial bitfile is fetched. Next, the physical address (PA) corresponding to the logical address (LA) is written to the frame address register (FAR) in the partial bitfile. Finally, the edited file is written to the FPGA over JTAG interface using Xilinx Impact configuration tool and the programming script presented on Listing C.3.

Correction of the single-bit upset takes 610 µs [141] and repairing the multiple-bit upset (updating the entire configuration frame) takes 2 s. Thus, 2 s is the maximum time that the flipped CRAM bit remains uncorrected.

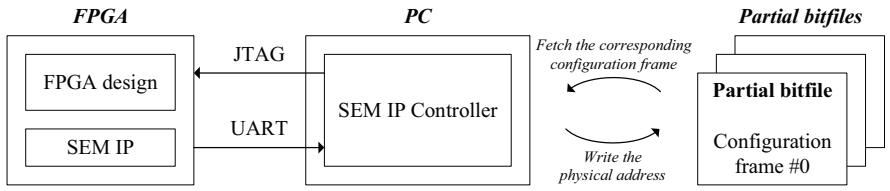
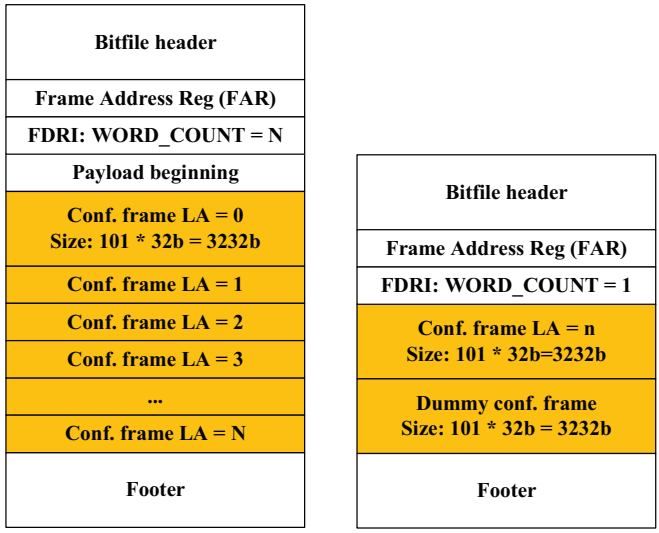


Figure C.1: Custom active scrubber architecture.



(a) Main bitfile (b) Partial bitfile

Figure C.2: Main bitfile and partial bitfile architectures.

Listing C.1: Report on correcting single bit error.

```
SC 04
SED OK
PA 01C00080
LA 00005AC9
WD 4B BT 07
COR
WD 4B BT 07
END
FC 00
SC 08
FC 40
SC 02
O>
```

Listing C.2: Report on detecting 2-bit ECC error.

```
SC 04
DED
PA 00041923
LA 00001F63
COR
END
FC 60
SC 08
FC 60
SC 00
I>
```

Listing C.3: Bash sript to load partial bitlife to an FPGA

```
#!/bin/bash
# Loads the bitfiles via Impact/JTAG cable.
# Usage: loadbit.sh <bitfile>

BATCHFILE=temp_load.impact

if [ ${#} != 1 ]
then
  echo 'Specify the bitfile!'
  exit 1
fi

BITFILE=${1}

rm -f ${BATCHFILE}

echo setmode -bscan >> ${BATCHFILE}
echo setcable -p auto >> ${BATCHFILE}

echo addDevice -p 1 -file ${BITFILE} >> ${BATCHFILE}

echo program -e -p 1 >> ${BATCHFILE}
echo quit >> ${BATCHFILE}

/opt/Xilinx/14.7/LabTools/LabTools/bin/lin64/impact -batch ${BATCHFILE}
wait
rm -f ${BATCHFILE}
```