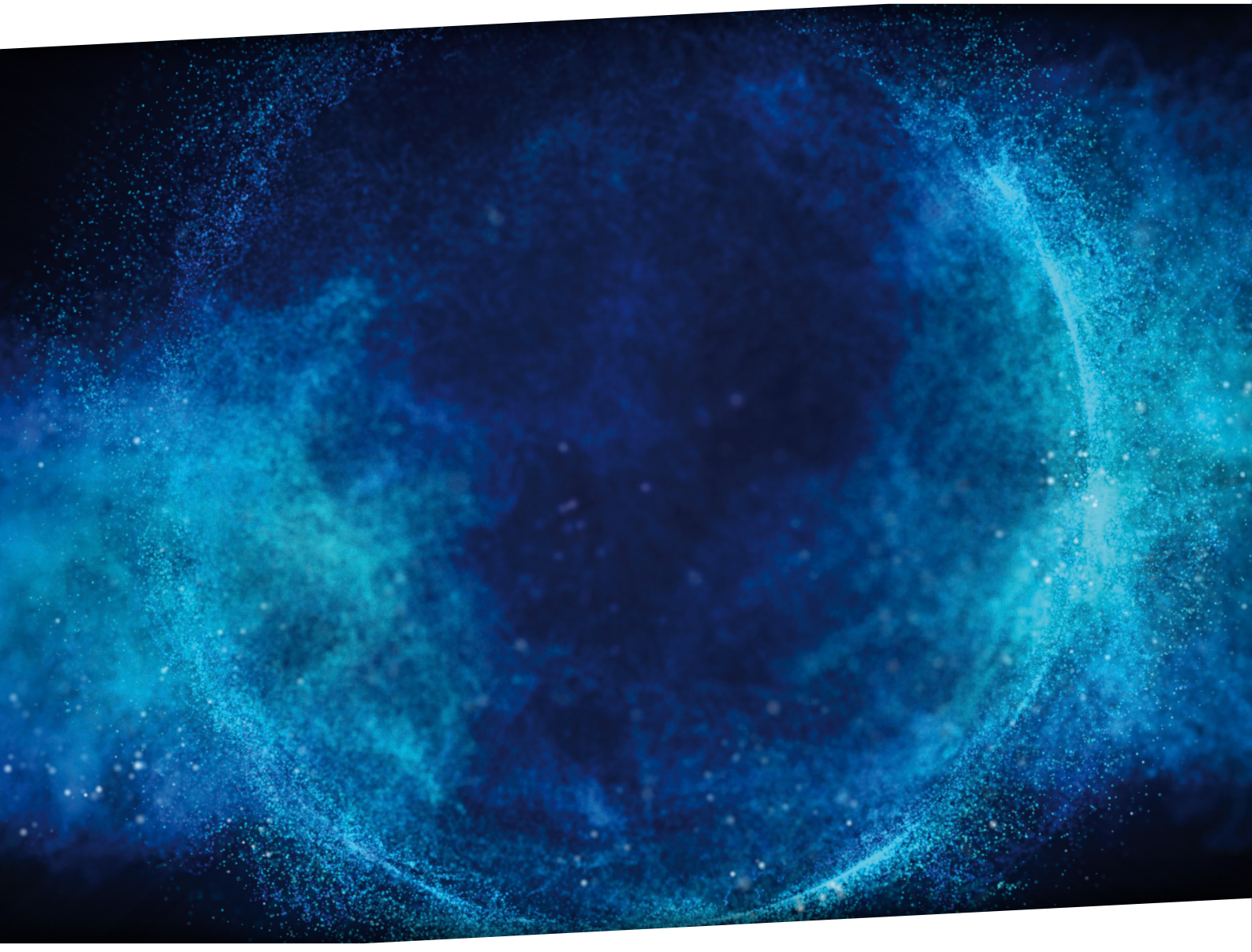
Radiation Qualification of the Cologne Chip GateMate A1 FPGA

Strahlenqualifizierung des Cologne Chip GateMate A1 FPGAs

Master's Thesis

Richard Jung



we focus on students
Fachhochschule Dortmund
University of Applied Sciences and Arts

CERN

Radiation Qualification of the Cologne Chip GateMate A1 FPGA

Strahlenqualifizierung des Cologne Chip GateMate A1 FPGAs

by

Richard Jung

to obtain the degree of Master of Engineering
at the Fachhochschule Dortmund.

Faculty of Information Technology

March 2023

Matriculation number:	7106905	
Thesis committee:	Prof. Dr. Michael Karagounis	Fachhochschule Dortmund
	Dr. Michael Gude	Cologne Chip AG

Cover: A graphic artistic view of the Brout-Englert-Higgs Field by CERN under CERN copyright.

An electronic version of this thesis is available at https://opus.bsz-bw.de/fhdo/
DOI: https://doi.org/10.26205/opus-3364

Fachhochschule Dortmund
University of Applied Sciences and Arts

Acknowledgements

There are many people who have supported me during the preparation and writing of my thesis and I would like to express my gratitude to them.

Firstly, I am grateful to my thesis supervisor, Prof. Dr. Michael Karagounis, for giving me the opportunity to complete my master's thesis in Information Technology at the University of Applied Sciences Dortmund and for arranging my internship at CERN. His expertise, knowledge, and feedback have been invaluable in shaping my work and ensuring its quality.

Furthermore, I would like to thank Cologne Chip for providing the GateMate hardware needed for the realization of this project. In particular, I would like to mention Patrick Urban, whose assistance throughout the FPGA design process helped me immensely when debugging difficult problems.

I would also like to extend my appreciation to the faculty members and staff of CERN, for providing me with a conducive environment for learning and conducting research. Their dedication to their field of study has been inspiring, and I am fortunate to have had the opportunity to learn from them. Thanks to Dr. Rudy Ferraro, Dr. Salvatore Danzeca and Antonio Scialdone for their support and supervision during my internship. I would also like to show my gratitude to my colleagues and friends at CERN, rendering my internship an exciting and unforgettable experience.

I am deeply indebted to my partner, family and friends for their love, encouragement, and unwavering support throughout my academic journey. Their sacrifices and unwavering belief in me have been instrumental in my success.

Richard Jung
Kreuztal, March 2023

Abstract

In this thesis, the radiation sensitivity of the novel Cologne Chip GateMate A1 field-programmable gate array (FPGA) is evaluated. An initial introduction of radiation mechanisms and their effects on electronics is given, followed by a brief overview of radiation test standards. The common elements present in FPGAs are discussed, which is followed by details of the GateMate FPGA device and a description of the software design flow. Afterwards, the development of a purpose-built printed circuit board (PCB) for radiation tests with the GateMate FPGA is detailed.

Four components of the GateMate have been tested during three radiation campaigns, as well as a benchmark circuit to compare the radiation performance of the GateMate with other FPGAs tested at the European Organization for Nuclear Research (CERN). The test architecture consists of the device under test (DUT) FPGA and a TESTER FPGA whose task is to provide inputs to the DUT and record its response. The DUT and TESTER designs developed for all tests are discussed in detail. Finally, the results obtained during the irradiation campaigns are presented, showing that the GateMate FPGA performs similarly to other FPGAs using the same process technology. Only the benchmark test was not finalized, as implementation problems prevented its completion in the given time frame. The thesis concludes with a comprehensive summary and outlook.

In dieser Arbeit wird die Strahlungsempfindlichkeit des neuartigen Cologne Chip GateMate A1 FPGA untersucht. Zunächst wurde eine Einführung in Strahlungseffekte und ihre Auswirkungen auf elektronische Komponenten gegeben, gefolgt von einem kurzen Überblick auf aktuelle Strahlungsteststandards. Die üblichen Elemente in FPGAs werden diskutiert, gefolgt von Details über GateMate spezifische Elementen sowie eines Überblicks über den Software-Design-Flow für GateMate FPGA Anwendungen. Im Anschluss wird die Entwicklung eines PCBs für Bestrahlungstests des GateMates detailliert.

Vier Komponenten des GateMate wurden während drei Strahlungskampagnen getestet, sowie eine Benchmark-Schaltung, um die Strahlungsempfindlichkeit des GateMate mit anderen am CERN getesteten FPGAs zu vergleichen. Die Testarchitektur besteht aus dem DUT FPGA und einem TESTER FPGA, dessen Aufgabe es ist, Eingaben an das DUT zu liefern und dessen Reaktion aufzuzeichnen. Die für alle Tests entwickelten DUT- und TESTER-Designs werden im Detail diskutiert. Schließlich werden die während der Bestrahlungskampagnen erzielten Ergebnisse vorgestellt, die zeigen, dass der GateMate FPGA ähnliche wie andere FPGAs mit vergleichbarer Prozesstechnologie liefert. Lediglich der Benchmark-Test wurde nicht finalisiert, da Probleme bei der Implementierung die Fertigstellung im vorgegebenen Zeitrahmen verhinderten. Die Arbeit schließt mit einer umfassenden Zusammenfassung und einem Ausblick ab.

Keywords — FPGA, radiation qualification, CERN, LHC, GateMate

Contents

Nomenclature

Symbols

Symbol	Definition
B	Boron
Co	Cobalt
e, e^-	Electron
e^+	Positron
h	Hole
H	Hydrogen
He	Helium
$\mathrm{K}^\mp$	Kaon
Li	Lithium
$\mathrm{n}^\mp$	Neutron
Ni	Nickel
O	Oxygen
p, p^+	Proton
Po	Polonium
Si	Silicon
Th	Thorium
U	Uranium
β^-	Beta minus decay
γ	Photon
$\mu^\mp$	Muon
$\pi^\mp, \pi^0$	Pion

Variables

Variable	Definition	Unit
C	Capacitance	F
D	Dose	Gy
E_γ	Photon energy	eV
$E_{\text{bind, e}^-}$	Electron binding energy	eV
D_{DD}	Displacement damage dose	Gy
$E_{\text{kin, e}^-}$	Electron kinetic energy	eV
E_{Neutron}	Neutron energy	eV
D_{TID}	Total ionizing dose	Gy

Variable	Definition	Unit
f	Particle flux	$\mathrm{p/(cm^2\,s)}$
F_{clk}	Clock frequency	Hz
F_{ext}	External frequency	Hz
F_{max}	Maximum frequency	Hz
F_{PLL}	PLL output frequency	Hz
F_{ref}	Reference frequency	Hz
F_{ring}	Ring oscillator frequency	Hz
F_{window}	WSR window frequency	Hz
I_{d}	Drain current	A
LET	Linear energy transfer	MeV/cm
m	Mass	kg
n	Count	—
$N_{\mathrm{components}}$	Number of components	—
N_{CPE}	Number of CPEs	—
N_{events}	Number of events	—
N_{ext}	Number of external clock pulses	—
N_{IT}	Number of interface traps	—
N_{OT}	Number of oxide traps	—
N_{ref}	Number of internal clock pulses	—
q	Charge	C
R	Resistance	Ω
S	Stopping power	$\mathrm{MeV/(g\,cm^2)}$
t_{PD}	Propagation delay	s
$t_{\mathrm{PD,\,CPE}}$	CPE Propagation delay	s
t_{reset}	FPGA reset time	s
V_{CLK}	FPGA clock voltage	V
V_{core}	FPGA core voltage	V
V_{GS}	Gate-to-source voltage	V
V_{WA}	FPGA west I/O bank A voltage	V
x	Distance	cm
Φ	Particle fluence	$\mathrm{p/cm^2}$
ρ	Density	$\mathrm{kg/m^2}$
σ	Cross section	$\mathrm{cm^2}$

Constants

Constant	Definition	Value	Unit
c	Speed of light in vacuum	299792458	m/s
K_{Gy}	Unit conversion factor	1.6×10^{-7}	Gy g MeV
m_{e}	Electron mass	$9.1093837015 \times 10^{-31}$	kg

1

Introduction

Radiation tolerance plays an important role in the use of electronic components in harsh radiation environments such as outer space, nuclear reactors or particle accelerators. The qualification of components is therefore a critical task when designing systems intended for these environments. While radiation hardened devices are available, it is more cost-effective to use commercial off-the-shelf (COTS) components when using devices in high quantities. Therefore, knowing the expected failure rate and lifetime of COTS components allows engineers to estimate the expected lifetime of the systems, and enable preventive replacement in terrestrial applications.

The focus of this master's thesis lies on the particle accelerators at CERN, the world's largest particle physics laboratory. Figure 1.1 shows all accelerators operated at CERN in 2022.

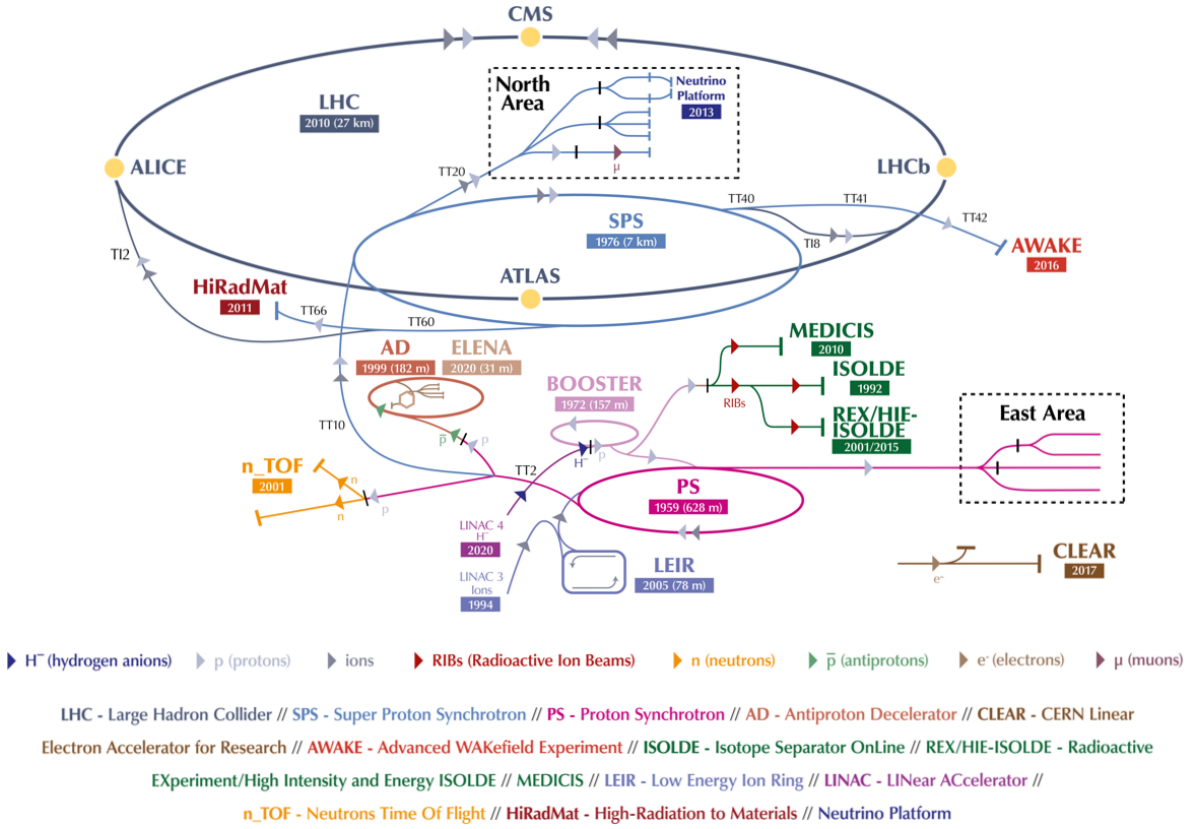


Figure 1.1: Overview of the CERN accelerator complex in 2022 [50]

These accelerators produce a mixed radiation field composed of electrons (e^-), kaons ($K^\mp$), muons ($\mu^\mp$), neutrons (n), pions ($\pi^\mp$), photons (γ) and protons (p^+) [32]. The main focus of CERN is the study of particles which make up the universe, their origins as well as their interactions. For this task the particle accelerators mentioned above are used. As the name suggests, these machines accelerate beams of particles (e.g., protons), to a certain speed before initiating a collision. This collision can take place between two beams, or between a single beam and a target.

The main accelerator in use today is the Large Hadron Collider (LHC) [27], in which two circulating proton beams, one clockwise and one counter-clockwise, are accelerated and brought to collision. The high energies resulting from such a collision replicate the state of the universe shortly after the big bang. To provide the LHC with a proton beam, negative hydrogen ions (H^-, hydrogen atoms with an extra electron) are first accelerated to 160 MeV in the Linear accelerator 4 (Linac4) first and then sent into the Proton Synchrotron Booster (PSB). This accelerator removes the two electrons from the hydrogen atoms, leaving only a beam of protons. The beam is then accelerated to 2 GeV and injected into the Proton Synchrotron (PS). An additional acceleration is performed by the PS, resulting in an energy of 26 GeV before feeding the beam into the Super Proton Synchrotron (SPS). Before them beam is split up for injection to the LHC, it is further accelerated to 450 GeV by the SPS. In the LHC, the protons undergo a final acceleration to 6.5 TeV before colliding at a total energy of 13 TeV in one of the following four experiments:

- A Large Ion Collider Experiment (ALICE): A detector designed to study strongly interacting matter at extreme energy densities (quark-gluon plasma);
- A Toroidal LHC ApparatuS (ATLAS): A general-purpose detector, whose technical implementation differs from Compact Muon Solenoid (CMS);
- CMS: A general-purpose detector, whose technical implementation differs from ATLAS;
- Large Hadron Collider beauty (LHCb): A detector designed to study the beauty quark in order to explain the slight differences between matter and antimatter.

As a noteworthy example, the ATLAS and CMS experiments both discovered the Higgs boson in 2012 [1][13]. This particle is responsible for giving mass to the other particles.

In addition to the process described above, other facilities in the CERN complex can also be fed from the various accelerators. This is exploited by the CERN Highly-Accelerated Mixed Field Facility (CHARM) facility, an irradiation infrastructure dedicated to the qualification of components and systems intended for use in the LHC.

FPGAs are key components of CERN's accelerator electronics. They are low-cost, high-performance integrated circuits (ICs) whose function can be configured to suit the task at hand. An FPGA consists mainly of configurable logic blocks (CLBs), input/output (I/O) pins, various hard macros (e.g. block RAMs (BRAMs)) and clocking resources (phase-locked loops (PLLs)), all interconnected by a routing structure. This allows a custom digital circuit to be designed in the FPGA, without the need to develop an application-specific integrated circuit (ASIC), while retaining the performance benefits of a custom circuit.

One of the biggest threats to electronics in an environment like the LHC or in space is radiation. Depending on the type of radiation and the electrical component affected, different effects can occur. For example, a metal-oxide semiconductor field-effect transistor (MOSFET), when hit by a charged particle in the insulation layer between the gate and the substrate, can temporarily turn on and affect the circuit connected to it. Memories on the other hand can exhibit bit flips, which invalidate the stored information. In addition, the behavior of components can be altered during their lifetime by the

exposure to radiation, which can lead to premature failure. While techniques exist to mitigate these types of errors, it is also important to understand the hardware's tolerance to radiation effects.

At CERN, this radiation hardness assurance (RHA) process custom-built electronic systems is documented in [21] and consists of seven phases (Figure 1.2).

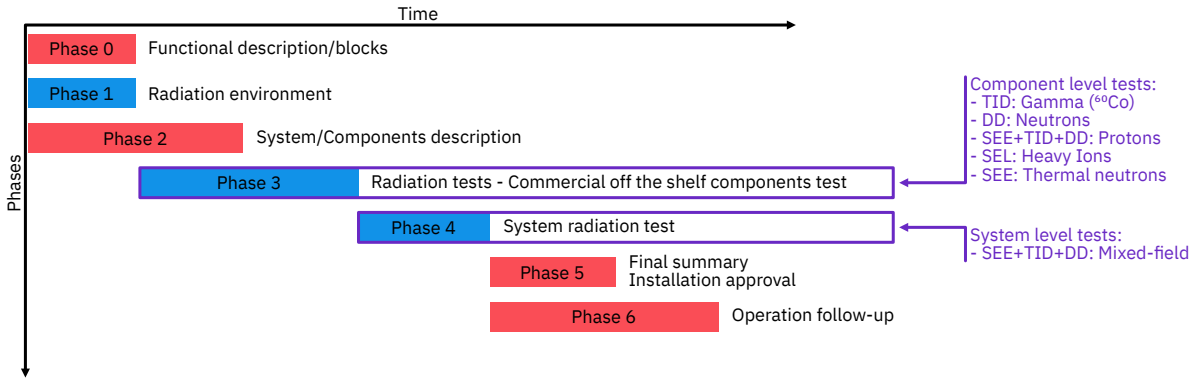


Figure 1.2: RHA process used at CERN for custom-built electronic systems. This figure is adopted from [21] with minor changes.

In Phase 0, the top-level requirements are extracted from the system specifications and serve as the starting point for the RHA process. Phase 1 uses Monte-Carlo simulations to calculate the radiation environment. In parallel, the maximum allowable component and system failures per year and over the required lifetime are defined in phase 2. The individual components of the system are then tested for different metrics (total ionizing dose (TID), displacement damage (DD), single-event effects (SEEs) etc.) in Phase 3. These tests are performed at internal and external radiation facilities, such as the Paul Scherrer Institute (PSI) for the combined TID + DD + SEEs tests using protons. Phase 4 consists of testing the entire system in a mixed field, as can be found in CERN's CHARM facility, since the particle spectrum of this field is similar to that of the LHC. If all these steps described above are completed successfully, the electronic system can be deployed in Phase 5 and the real performance can be monitored in Phase 6.

For testing the radiation sensitivity of FPGAs, the classic method described in literature can be found in [10]. Each individual part of an FPGA, e.g., the flip-flops (FFs) in the CLBs or the BRAM cells, is tested with a specific test design, so that the component-wise failure cross-section can be determined. While these tests give an indication of the sensitivity of the FPGA's components, but fail to provide a realistic overview of the performance when implementing an actual design, the aforementioned tests are also augmented with benchmark-based characterizations. Benchmark circuits, such as those described in [19], can be used for this task, as they represent real-world applications. This allows comparisons between FPGAs from different vendors or varying implementation parameters.

This master's thesis documents the radiation qualification of the GateMate CCGM1A1, a novel FPGA developed by Cologne Chip[1]. The FPGA is manufactured in a 28 nm Super Low Power (SLP) GlobalFoundries[2] complementary metal-oxide semiconductor (CMOS) process in Dresden, Germany, and consists mainly of CLBs called *Cologne Processing Elements (CPEs)*, BRAM cells and PLLs. 162 I/O pins can be used to interface with the FPGA, supporting single-ended and differential signals. Design synthesis is performed via the open-source Yosys Open SYnthesis Suite (Yosys)[3] framework, and the

[1]https://colognechip.com
[2]https://globalfoundries.com/
[3]https://github.com/YosysHQ/yosys

implementation is done using a custom Place and Route (P&R) tool, including static timing analysis (STA).

First, the radiation-matter interaction mechanisms are presented, and the resulting effects on electronic devices. The GateMate FPGA is afterwards presented, both the hardware components and the software design flow. This is followed up by the development of a test system for the GateMate FPGA for radiation tests. The PCB is based on the CERN's CHARM radiation tester board (CRaTeBo) and intended for use in the CHARM radiation testing facility. An additional PCB was also developed to augment the GateMate evaluation board with an FPGA Mezzanine Card (FMC) connector. Afterwards, the test methodology used for the radiation tests is discussed. Finally, the results of three radiation tests campaigns are presented.

2

Radiation Mechanisms and Their Effects on Electronics

This chapter introduces and discusses the interactions of particles with matter and their effects on electronic devices. In particular, the focus lies on metal-oxide semiconductor (MOS) components, as the DUT FPGA is based on CMOS technology.

In the radiation community, the term *flux* f $[p/\text{cm}^2\text{s}]$ is used to describe the amount of particles passing through an area per unit time. Here, p represents the currently observed type of particle, in this case *protons*. The integral of the flux over a given time period is defined as the *fluence* Φ $[p/\text{cm}^2]$. Furthermore, the absorbed *dose* D of ionizing radiation per unit mass is measured in the unit Gray [Gy].

2.1 Particle Interactions with Matter

When matter is struck by a particle, various types of interactions can take place. These interactions depend on the type and energy of the striking particle, as well as the material being hit. The particle interaction can be divided into the groups *photons*, *charged particles* and *neutrons*, which are described in more detail in the following subsections.

2.1.1 Photons

Photons can interact with an atom via three different types of interaction principles, as shown in Figure 2.1.

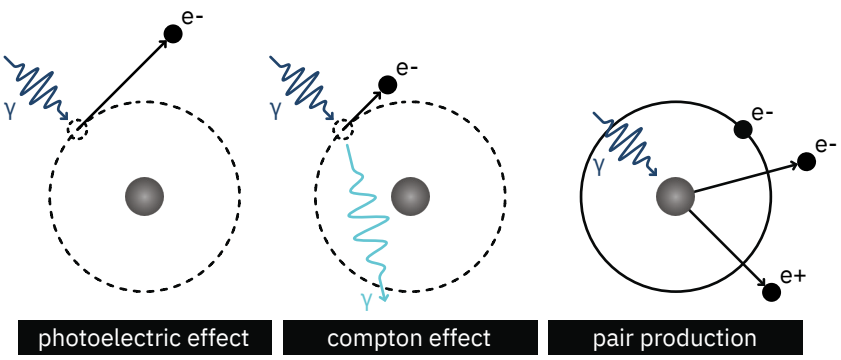


Figure 2.1: Illustration of photon-matter interactions

Photoelectric effect
If the energy of the photon E_γ is similar to the binding energy of an orbital electron $E_{\text{bind,e}^-}$, all the photon's energy is transferred to the electron. The electron is then ejected from the atom with a kinetic energy $E_{\text{kin,e}^-} = E_\gamma - E_{\text{bind,e}^-}$ [32].

Compton effect
If the energy of the photon is much greater than the binding energy of an orbital electron, the photon can collide with the electron. The photon transfers a part of its energy to the electron, resulting in an increase in the photon's wavelength and its scattering. The electron on the other hand is ejected from the atom.

Pair production
This interaction necessitates that the trajectory of the photon is very close to the atom's nucleus, and that the photon's energy is above the threshold $E_\gamma \geq 2\,m_\text{e}c^2$, where m_e is the mass of an electron [66]. In this case, the photon can lose all of its energy and produce an electron/positron pair.

2.1.2 Neutrons

Because neutrons are electrically neutral, their only interaction mechanism is via nuclear interactions (shown in Figure 2.2). These can be split up into the following four principles:

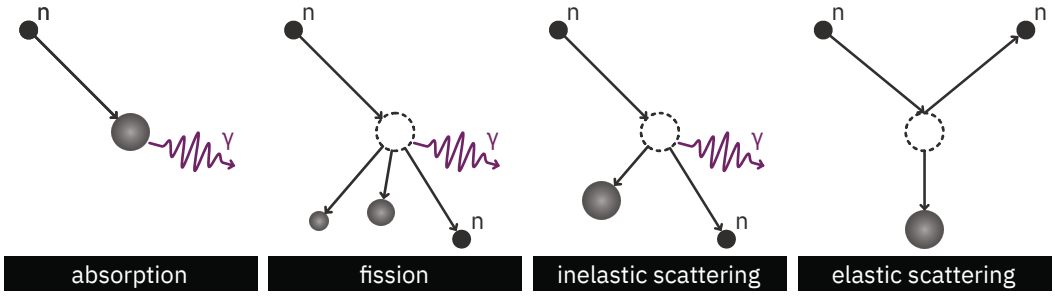


Figure 2.2: Illustration of neutron-matter interactions

Neutron absorption
When a neutron (mainly thermal neutrons with an energy of around 25 meV) is absorbed by a nucleus, the energy of the nucleus increases, which leaves it in a state of excitation. To leave this state, the nucleus must lose energy via photon emission.

Fission
When a neutron is absorbed by a very heavy atomic nucleus, the nucleus can get split into smaller nuclei, also called *fission fragments*. This process also releases energy in the form of photons.

Inelastic scattering
Similar to the absorption process described in the first principle, the incident neutron its absorbed by the nucleus. The nucleus then emits a neutron of lower energy than the incident neutron and becomes excited. This excitation state is then left via photon emission.

Elastic scattering
After colliding with an atomic nucleus, the neutron transfers some of its energy to the nucleus. Both are additionally scattered.

2.1.3 Charged Particles

Charged particles such as pions, protons, kaons, muons etc., as their name implies, have an electric charge $q \neq 0\,\mathrm{C}$. As a result, the following Coulombian interactions (Figure 2.3) can occur when these particles collide with matter:

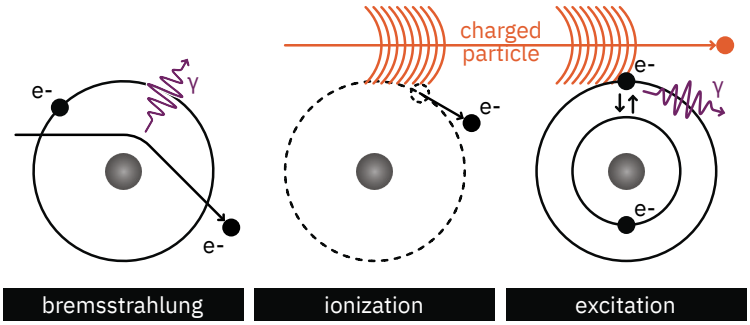


Figure 2.3: Illustration of charged particle-matter interactions

Bremsstrahlung
When a light particle (for example an electron) passes near the nucleus of an atom, the electric charge of the nucleus can deflect the electron's trajectory. During this process, the energy of the electron gets reduced, which results in the emission of a photon.

Ionization
When a particle travels near an orbiting electron and its energy is greater than the binding energy of the orbital electron, the orbital electron is ejected from the atom.

Excitation
When a particle close to an orbiting electron and its energy is lower than the binding energy of the orbital electron, the orbital electron is excited to a higher energy state. After a while, the electron will leave this state of excitation, causing it to release energy in the form of a photon.

In addition to the Coulomb interactions mentioned above, charged particles can also undergo nuclear interactions when interacting with matter.

2.1.4 Energy Loss through Interactions

In the case of ionization, the term *direct ionization* is used to describe when the ionization is caused by the incident *primary particle*. If the particles generated by the initial interaction have sufficient energy, then these so called *secondary particles* can also cause further ionizing interactions. This is called *indirect ionization*. In the target matter, the primary particle's interaction will dominate, as the energy of the particle is the highest at this stage [32]. The energy loss can be categorized into two groups, *ionizing energy loss (IEL)* and *non-ionizing energy loss (NIEL)*, which can be distinguished by the following characteristics:

Ionizing energy loss
Energy lost to ionizing interactions (e.g., photoelectric and Compton effect, ionization).

Non-ionizing energy loss
Energy lost to non-ionizing and nuclear interactions (e.g., pair production, bremsstrahlung, excitation, absorption, fission, inelastic and elastic scattering).

With each interaction, energy of the incident particle is reduced, until it is completely stopped. The amount of energy lost $\mathrm{d}E$ per distance traveled $\mathrm{d}x$ is quantified by the *linear energy transfer (LET)*, which, in its general form, is defined by (2.1).

$$LET = \frac{\mathrm{d}E}{\mathrm{d}x} \quad \left[\frac{\mathrm{MeV}}{\mathrm{cm}}\right] \tag{2.1}$$

The rate of the particle's energy loss due to the various interactions can be normalized to the density ρ of the struck material. This results in the mass stopping power S as defined in (2.2).

$$S = \frac{1}{\rho}\frac{\mathrm{d}E}{\mathrm{d}x} = \frac{1}{\rho}\left(\left.\frac{\mathrm{d}E}{\mathrm{d}x}\right|_{\mathrm{IEL}} + \left.\frac{\mathrm{d}E}{\mathrm{d}x}\right|_{\mathrm{NIEL}}\right) \quad \left[\frac{\mathrm{MeV}}{\mathrm{g/cm^2}}\right] \tag{2.2}$$

Particles that are affected by the strong nuclear force, i.e., hadrons, will have an energy peak at the end of their path through matter, which is called the *Bragg-Peak* [66].

Depending on the type of interaction, two types of damage effects can be observed, *TID* and *DD*. While, for charged particles, the Coulombian interactions prevail, leading to ionizing dose effects, neutrons mainly cause displacement damage [32].

2.2 Radiation Effects on Electronics

This section introduces the effects that particles can have on electronics. These can initially be divided into *cumulative effects* and *SEEs*, with the following characteristics:

Cumulative effects
: As the name implies, cumulative effects are accumulated up over the lifetime of a component and are caused by DD and TID. Although not immediately visible, these effects lead to a degradation of the component's internal parameters, for example a shift in the gate-source threshold voltage of a MOSFET.

Single-event effects
: A single particle inducing an IEL can lead to SEEs, which can be divided into destructive and non-destructive effects. These do not change the component's parameters, but modify its behavior.

Since the DUT is based on CMOS technology, the following explanations will mainly refer to this type of device.

2.2.1 TID Effects

The TID absorbed by a material when hit by particles is defined in (2.3), with the unit conversion factor $K_{\mathrm{Gy}} = 1.6 \times 10^{-7}\,\mathrm{Gy\,g\,MeV}$.

$$D_{\mathrm{TID}} = K_{\mathrm{Gy}} \int \frac{LET(E)}{\rho}\,\Phi(\rho, E)\;\mathrm{d}E \quad [\mathrm{Gy}] \tag{2.3}$$

The TID effects in MOSFETs are caused by ionizing interactions in the SiO_2 oxide layers of the transistor, and the specific layer affected depends on the used process size. For MOSFETs with a large gate oxide thickness ($> 5\,\mathrm{nm}$), such as those used as I/O transistors in FPGAs, the affected oxide layer is between the gate and the SiO_2/Si interface [30]. Modern CMOS processes on the other hand have small gate oxide thickness, resulting in a higher electron-hole mobility in this layer, and reduced radiation effects in the gate oxide [30]. However, they are subject to TID effects in the shallow trench isolation (STI) oxide. This oxide is used at these process nodes to isolate transistors from each other and still has

a large oxide thickness [62]. In the oxide of a MOSFET, upon getting struck by a particle, electron-hole (e-h) pairs are generated. This results in a charge build-up in the insulation and subsequently to the aforementioned degradation of device parameters.

Considering an N-channel MOSFET as an example of the effects on thick gate oxide layers (Figure 2.4), an electric field is applied across the SiO_2 layer during normal operation. The following processes occur after an ionizing interaction (Figure 2.5):

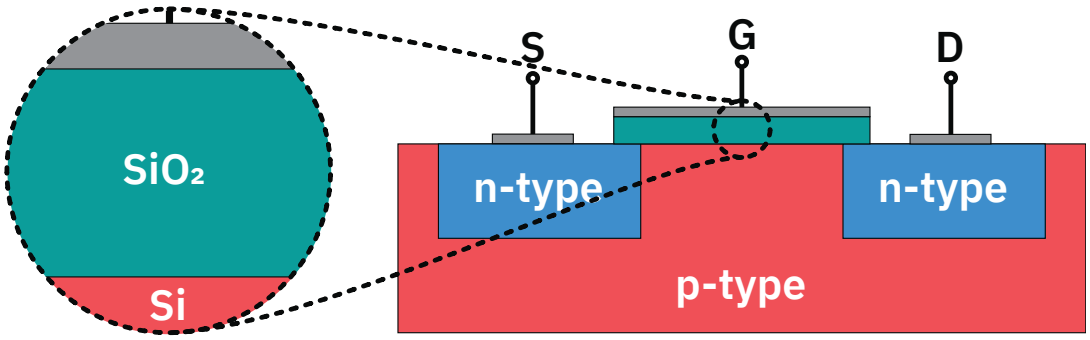
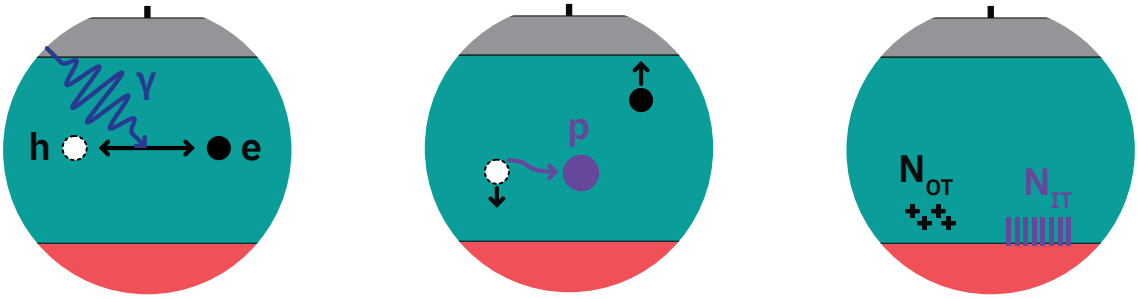


Figure 2.4: Structure of an n-channel MOSFET used to describe the TID effect. The SiO_2/Si interface, which is shown in the magnified view on the left, is the border area between the SiO_2 and Si layers.

1. e-h pairs are generated by the impact;
2. Some e-h pairs recombine after a short amount of time, while other pairs are separated by the applied electric field and the resulting drift processes. The fraction of not-recombined e-h pairs is referred to as *charge yield*;
3. The separated electrons, due to their high mobility, escape through the gate. On the other hand, the slower holes drift through the SiO_2 layer towards the substrate;
4. Some holes can be trapped in the oxide, leading to *oxide traps*;
5. Holes traveling through the oxide also release protons, which can be trapped at the SiO_2/Si interface. These are called *interface traps*;
6. Both traps shift the voltage threshold of the device. In addition, interface traps also lead to a reduction in the mobility of the charge carriers, which in turn reduces the slope of the V_{GS}-I_d curve [11].



(a) A photon hits part of the SiO_2 oxide, creating an electron-hole pair.

(b) The electron travels to the gate, while the hole drifts to the Si/SiO_2 interface, releasing protons.

(c) Trapped holes form oxide traps N_{OT}, while the trapped protons form interface traps N_{IT}.

Figure 2.5: TID effects in an n-channel MOSFET

The TID effects in the gate oxides of both pMOS and nMOS transistors can be summed up as following:

pMOS
Higher voltage required to create an inversion channel – voltage threshold increases – the transistor is harder to turn on – worst case: transistor stays off

nMOS
Lower voltage required to create an inversion channel – voltage threshold decreases – the transistor is easier to turn on – worst case: transistor stays on

One thing to note here is that the voltage threshold of the nMOS transistor can also shift to its pre-irradiation value if the accumulated dose is high enough. It has been shown in [30] that this effect occurs after a dose of 10 kGy to 60 kGy.

The aforementioned effects of the voltage shift and the reduction of the V_{GS}-I_d slope only happens in thick gate oxides. On the other hand, the TID effects in modern CMOS transistors take place at the interface between the STI oxide and the channel. Here, the charges accumulating on the surface of the STI can lead to a current flow between drain and source of the transistor, which results in an increase of leakage current (Figure 2.6).

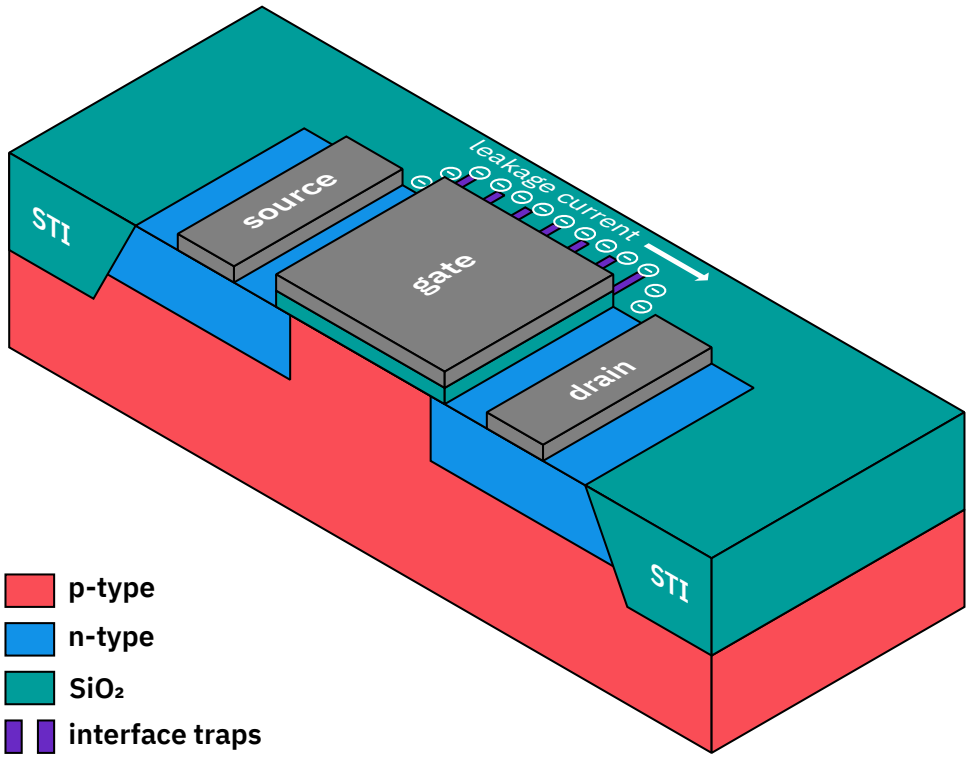


Figure 2.6: Leakage current induced by TID effects in the STI oxide

2.2.2 DD Effects

Unlike TID effects, displacement damage is caused by NIEL, and since the radiation travels through the entire device, displacement damage is also a volumetric effect. An incident particle can strike a silicon nucleus in the semiconductor lattice, displacing it from its current position. This in turn creates defects in the lattice, resulting in asymmetries that change the behavior of the e-h pairs. The displacement damage dose (DDD) is calculated using (2.4).

$$D_{\mathrm{DD}} = \frac{NIEL(E)}{\rho} \Phi(E) \quad [\mathrm{Gy}] \tag{2.4}$$

Figure 2.7 illustrates the four types of defects that can be caused by a particle impact. A *vacancy* defect is where a silicon atom is missing from the lattice. Two vacancies side-by-side are called a *divacancy*. If a displaced atom is located between the lattice, i.e., in a non-lattice position, it is called an *interstitial*. Finally, a combination of a vacancy and an interstitial is called a *Frenkel pair*.

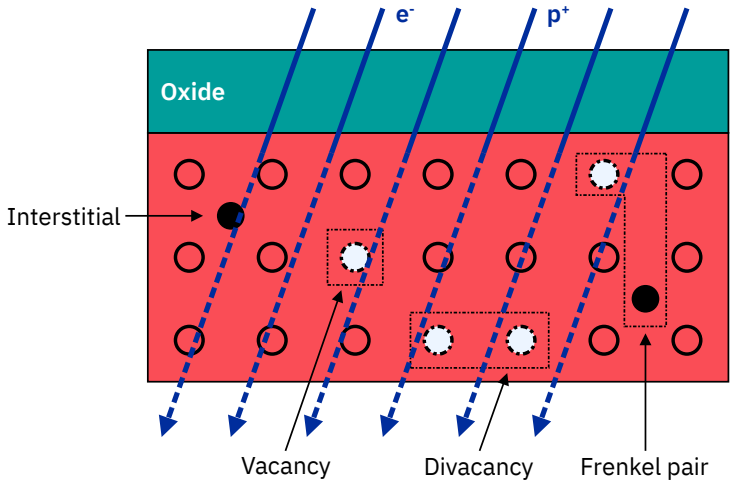


Figure 2.7: Overview of the displacement damage defect types

The displaced nuclei create traps that degrade properties such as carrier generation, recombination and transportation. In a bipolar junction transistor (BJT), for example, an increase in the recombination rate leads to an increase in the base current. As the collector current is unaffected by this, the current gain of the BJT is reduced [9]. MOS devices are generally less susceptible to displacement damage effects, because their active region, i.e., the area between the source and drain, is comparatively thin. This results in a very small active volume in which the displacement damage can be observed [9]. In addition, displacement damage mainly affects minority carrier devices [32]. The function of MOS devices, on the other hand, are based on majority carriers. They therefore have an inherent resistance to this type of damage.

2.2.3 Single-Event Effects

Compared to the cumulative effects described above, which accumulate over the lifetime of the device, SEEs occur instantaneously. The outcome of such an event can either be not observable, a transient resulting in undefined behavior, a flipped bit in a memory, or even permanent damage to a part of the device. As a particle passes through matter, it will lose a part of its energy through the interactions described in Section 2.1. The ionizing interactions lead to a charge build-up inside the device. This leads to internal transient currents due to drift processes through the applied electrical field. If this happens in a critical regions such as a p-n junction, destructive or non-destructive SEEs can occur. The destructive SEEs are:

Single-event latch-up (SEL)
: A low impedance path is created in the device between power and ground. If the current flow through this path exceeds the rated current density, electromigration effects can damage conductive elements or the entire device. Otherwise, the latch-up induced effects can be stopped by a power cycle.

Single-event burnout (SEB)
A charged particle can cause a second breakdown in the MOSFET, resulting in high currents through the transistor. If this fault is not cleared, the transistor goes into thermal runaway and is destroyed.

Single-event gate rupture (SEGR)
A charged particle creates a low-resistance path in the gate oxide, generating charges. These charges propagate to the SiO_2 interface, thus increasing the electric field across the dielectric. If the field strength is high enough, it can lead to the device's failure by gate rupture.

The non-destructive SEEs are:

Single-event upset (SEU)
The logic state of a memory cell can be inverted by an incident particle of sufficient energy. This error can be corrected by rewriting the memory.

Multiple-event upset (MEU)
MEUs also affect memory cells. But instead of hitting one memory cell, they hit several adjacent cells side-by-side and invert their states. This is also called multiple-bit upsets (MBUs).

Single-event transient (SET)
A brief voltage spike is induced into the system. Depending on the specific function of the device, this effect may not always be visible, for example if it is not captured by the next memory element.

Single-event functional interrupt (SEFI)
A particle strike causes the system to go into a non-functional state. While recovery from such an error may occur after some time, most systems require external intervention through a reset or power cycle [9].

2.3 Characterization Standards

The characterization of radiation effects is the subject of several standards, both for space and terrestrial applications For example, the European Space Agency (ESA) details the RHA phases for space projects in [25]:

1. Mission analysis: Definition of the mission environment and top-level radiation requirements;
2. Feasibility: Preliminary radiation characterization to help technology selection;
3. Preliminary definition: Definition of electronic design and spacecraft layout;
4. Detailed definition: Performing of radiation characterization test;
5. Qualification and production: Radiation tests on flight lots.

The test procedures used for the aforementioned radiation tests are detailed in the specifications European Space Components Coordination (ESCC) 22100 [24], 22500 [23] and ESCC 22900 [26], which deal with SEEs, DD effects and TID effects respectively. These documents do not specify dose threshold values that electrical devices must meet for use in an ESCC context, i.e., for use in space missions, as the thresholds differ depending on the current mission. Instead, they describe the test procedures required to qualify components. As an example, ESCC 22900 details the following procedure for testing electrical components regarding TID in a space context:

1. Serialize the test samples;
2. Measure electrical device parameters at room temperature. Replace parts that fail this test;
3. Irradiate the samples at a predetermined dose rate to the specified dose;
4. Measure electrical device parameters at room temperature. Replace lot if the test fails. If multiple exposures are specified, restart irradiation (step 3);
5. Anneal devices at room temperature under bias for 24 hours;
6. Measure electrical device parameters at room temperature. Replace lot if the test fails;
7. Simulate aging by annealing devices at 100 °C under bias for 168 hours;
8. Measure electrical device parameters at room temperature. Replace lot if the test fails, otherwise accept the lot.

ESCC 22100, focusing on SEE qualification, does not have a detailed test procedure. Rather, the standard specifies the requirements necessary for the SEE testing of ICs and discrete semiconductors. For testing, either protons with an energy between 20 MeV and 200 MeV, or heavy ions with a range of at least 40 µm in silicon shall be used. The DUT must also be delidded in case heavy ions are used for the beam. The test hardware for SEUs shall monitor bistable functional blocks, and feature the possibility to vary the data stored in the tested memory cells. The test software is tasked with logging all events, as well as their time and location. To minimize the effects SELs have on the DUT, the test setup shall identify SELs and protect the DUT from device degradation, for example by initiating a power cycle. All device operating conditions are to be tested when considering SEFIs. Here, the test software shall detect, log and correct these faults. Test conditions must either be the worst-case for the device type, or application specific.

For terrestrial applications, Joint Electron Device Engineering Council (JEDEC) standard JESD89B [41] provides guidelines for performing radiation tests for SEEs induced by alpha particles and terrestrial cosmic rays. It also includes test procedure requirements, such as that the test equipment used during testing must tolerate stray radiation, either through shielding or by being mounted remotely in a shielded location. Recommendations are also given on which circuit elements to consider for testing, and describes the various methods how to test specific elements. For example, FPGAs tests setups should be able to detect the following effects:

- SEUs in the configuration RAM (CRAM), the I/O blocks, the flip-flops and in the random access memory (RAM) cells;
- SETs in the flip-flops;
- SEFIs in the logic blocks, the I/O blocks and in the configuration circuitry;
- SELs in the entire device.

Four types of test procedures are described in the standard: a real-time test, an accelerated alpha particle test, an accelerated high-energy neutron test and an accelerated thermal neutron tests. Each of these listed tests is used to evaluate the response of the tested equipment to a specific type of SEE induction vector.

The first method, real-time testing, is the most direct way to qualify a DUT under terrestrial radiation. The device is placed under normal operating conditions and monitored for faults. To detect a statistically significant number of faults in a reasonable amount of time, the standard recommends using multiple devices in parallel, since the number of faults scales linearly with the number of devices used.

During manufacturing of electronic devices, materials containing traces of radioactivity (e.g., uranium (U) and thorium (Th) in packaging materials, or polonium-210 (^{210}Po) in lead and solder tin) can cause SEEs in these devices. To determine the device's susceptibility to these events, the accelerated alpha

particle test is recommended. This test requires that the surface of the DUT can be directly exposed to the alpha particle source.

Cosmic rays originating from outer space can enter the Earth's atmosphere and collide with nuclei in the air. This interaction produces a cascade of resulting particles called *terrestrial cosmic rays*, which consist mainly of neutrons, but also protons and pions. When these particles interact with the electronic devices using the mechanism described in Section 2.1, SEEs can be induced. The high-energy neutron test is specifically designed to acquire the sensitivity of the device to such effects caused by such cosmic rays. The standard considers all neutrons with an energy of $E_{\text{Neutron}} \geq 10\,\text{MeV}$ as *high-energy neutrons*. Dedicated high energy particle beam facilities can provide proton and neutron beams that can be used for such a test.

Neutrons with an energy level of $25\,\text{meV}$ are called *thermal neutrons*. They originate from cosmic rays that have reached thermal equilibrium with their environment. Their energy is not sufficient to cause SEEs directly, but neutron absorption by boron-10 (^{10}B) can lead to its fission into highly ionizing particles as shown in (2.5) [47]. These ionized particles can then in turn cause SEEs [33].

$$\begin{aligned} \text{n} + {}^{10}\text{B} &\longrightarrow {}^{7}\text{Li}\,(0.84\,\text{MeV}) + {}^{4}\text{He}\,(1.47\,\text{MeV}) + \gamma\,(0.48\,\text{MeV}) \qquad 94\,\% \\ \text{n} + {}^{10}\text{B} &\longrightarrow {}^{7}\text{Li}\,(1.02\,\text{MeV}) + {}^{4}\text{He}\,(1.78\,\text{MeV}) \qquad 6\,\% \end{aligned} \tag{2.5}$$

Traces of ^{10}B can be present in the polysilicon and substrate doping, as well as in any other boron sources that may be in the device [31]. Thermal neutron sources used for testing are, for example, particle accelerators or dedicated neutron generators. Here, energetic neutrons are produced and subsequently reduced in energy. Alternatively, nuclear reactors can also provide thermal neutrons for testing purposes.

In addition, JEDEC has published the JESD57A [43] and JESD234 [44] standards. Both standards target the same metric as JESD89B, i.e., the cross-section for SEEs, but use protons and heavy ions instead of neutrons. As with the ESCC standards discussed above, no maximum limits are specified. Instead, these documents detail the steps and procedures to be followed when intending to test electronic devices, including test preparation and analysis of results. The ASTM International standardization institute also published test standards for radiation effects: ASTM F1892-12 [28] for TID effects and ASTM F1192-11 [29] for SEEs.

3

GateMate FPGA

An FPGA is an IC whose internal functionality is defined by an application-specific hardware design. Compared to other embedded devices, such as central processing units (CPUs), graphics processing units (GPUs), microcontrollers or ASICs, FPGAs strike a balance between performance, power efficiency, flexibility and unit cost [5]. The only major drawback is the engineering cost of engineering, as in general it takes more effort to design an FPGA application than a solution based on CPUs, GPUs or microcontrollers [7]. Because of the advantages described above, FPGAs are used in a variety of computationally intensive fields. These include, but are not limited to digital signal processing (DSP) [35, 65, 57], machine learning [69, 63], data center applications [75, 14], computer vision [37], etc.

This chapter describes the Cologne Chip GateMate CCGM1A1 FPGA in more detail. The first part, which is mainly based on Amano [2], consists of an introduction to the inner workings of an FPGA. The second section focuses on the GateMate specific hardware, while the third section introduces the GateMate evaluation board.

3.1 FPGA Primer

An FPGA is an IC that allows the user to define a custom digital circuit, more specifically a *design*, using a *hardware description language (HDL)* like *Verilog* or *VHSIC Hardware Description Language (VHDL)*, with VHSIC standing for Very High Speed Integrated Circuits. To do this, an FPGA uses CLBs connected in a matrix with programmable interconnects to realize the design specified by the design engineer. The typical components of an FPGA can be divided into the following three categories:

Soft blocks
: The CLBs contain *lookup tables (LUTs)* for combinatorial logic, *flip-flops* for sequential logic, and sometimes *full adders (FAs)* for addition. These blocks are called *soft blocks*, as their specific function is not set in hardware, but is defined during the design process.

Hard blocks
: The functionality of these blocks is fixed in silicon, with the advantage that their size and speed are optimized for their specific application. Examples of such hard blocks include *DSP elements*, *embedded RAM*, high-speed *I/O blocks* etc.

Clocking resources
: These components, for example *PLLs*, can synthesize different clock frequencies from a base frequency, and also reduce jitter on the clock lines.

As mentioned above, the design to be loaded into the FPGA is first created using an HDL, or more recently, *high-level synthesis (HLS)* [49]. Then, in the case of an HDL design, the *synthesis* process converts the *register-transfer level (RTL)* description into a gate level netlist, which is device-agnostic (if no vendor primitives are used) and contains the wiring information between each gate in the design. In addition, the information from the previously generated netlist is mapped to the device-specific logic hardware. This is also called *technology mapping*. The next step is *place and route*, which takes the output from the technology mapping process and places all the logic elements from the netlist into the FPGA's fabric and routes the necessary connections. The final step is to convert the placed and routed design into a binary file, the *bitstream*, which can be uploaded to the FPGA.

3.1.1 Configuration Memory

The design uploaded to the FPGA is stored in its configuration memory. Three technologies are available for this, *flash memory*, *antifuse* and *static memory*:

Flash memory
: Flash memory is a type of non-volatile memory technology. A flash memory cell has a similar structure to a MOSFET, but instead of a single gate, it consists of an upper control gate and a lower floating gate. Due to the isolation of the floating gate, an applied electric field does not collapse after the gate voltage is removed, causing the memory cell to store its value until the next application of an electric field.

Antifuse
: An antifuse is a type of switch that is an initially open. When a large current is applied, the antifuse burns out, resulting in a closed state. By selectively burning antifuses, the FPGA can be configured.

Static memory
: A static RAM (SRAM) cell consists of two inverters connected in a feedback loop, with two additional transistors used to read from and write into the memory. As long as power is applied to the memory, the state of the feedback loop is stored.

Table 3.1 provides a comparison of the different types of configuration memories. Most FPGAs in use today implement an SRAM based configuration memory.

Table 3.1: Comparison between the different FPGA configuration memory types [2]. The term Tr. in the row labeled memory area stands for transistor.

Metric	Flash memory	Antifuse	Static memory
Nonvolatile	Yes	Yes	No
Reconfigurability	Yes	No	Yes
Memory area	Mid (1 Tr.)	Small (none)	Large (6 Tr.)
Process	Flash	CMOS + Antifuse	CMOS
Lifetime / Times	10000	1	Infinity

The bitstream is uploaded to the FPGA via a Joint Action Test Group (JTAG) connection and dedicated programmer. When considering an SRAM based FPGA, the bitstream can be stored in a non-volatile fashion by utilizing an external flash memory module. In this case, the FPGA can load its configuration from memory via Serial Peripheral Interface (SPI) and does not need to be programmed manually. The external memory is not required by the flash based FPGA, which, by definition, has non-volatile flash memory on-chip.

3.1.2 Logic Blocks

The most important element of an FPGA is the CLB. Such a block consists of a LUT, flip-flops and an output multiplexer. A LUT is composed of a 1-bit wide memory with connected multiplexers and is responsible for the realization of combinatorial logic. The input to the LUT corresponds to the address of a bit in the previously mentioned memory, and its output is defined by the value stored at that specific location. A k-input LUT is composed of 2^k memory cells and a $2k$-input multiplexer (MUX). Such a LUT can represent a logic function of size 2^{2^k}. An exemplary block diagram of a CLB is shown in Figure 3.1.

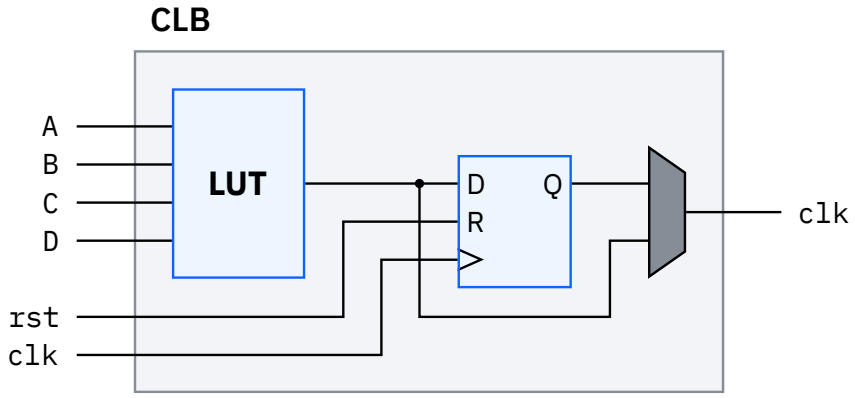


Figure 3.1: Example block level overview of a simple CLB

After the LUT, the flip-flops are responsible for the sequential logic. These flip-flops have dedicated clock and reset lines, that are routed throughout the FPGA fabric, to ensure fast signal propagation. The final output MUX selects weather the logic block outputs the LUT's or the flip-flop's signal. Additionally, dedicated carry lines are also often present to increase the speed of arithmetic operations [2].

3.1.3 Routing Structure

The routing structure in an FPGA is responsible for connecting the logic blocks and other cells used in the final design. There are two different types of routing structures present, the global routing structure and the local routing structure, each with a different purpose.

The global routing structure is responsible for connecting the various hard and soft blocks of an FPGA. Either an island or a hierarchical layout is used to implement the routing. In the hierarchical layout, multiple levels of interconnects are used, with each higher level having a larger number of wire tracks than its previous level. An example of this layout is shown in Figure 3.2, where in the first level several logic blocks are wired together to form a cluster. The second level connects the level 1 clusters in one direction of the fabric, while the third level is used for connections in the other direction. While this type of routing can provide a high-speed connection between the logic blocks in a cluster, the advantage diminishes as soon as clusters need to communicate across the higher levels. On the other hand, in the island layout (see Figure 3.3) the individual logic blocks are interleaved with the routing wires, hence the term island. Adjacent logic blocks can be interconnected to increase the speed of signal propagation between these blocks, while more distant blocks use the routing wires. Modern CMOS processes shift the dominance of signal propagation delay from the gate to the routing delay. Therefore the island layout is now preferred [2], as the parasitic capacitances and resistances are more uniform with the island layout.

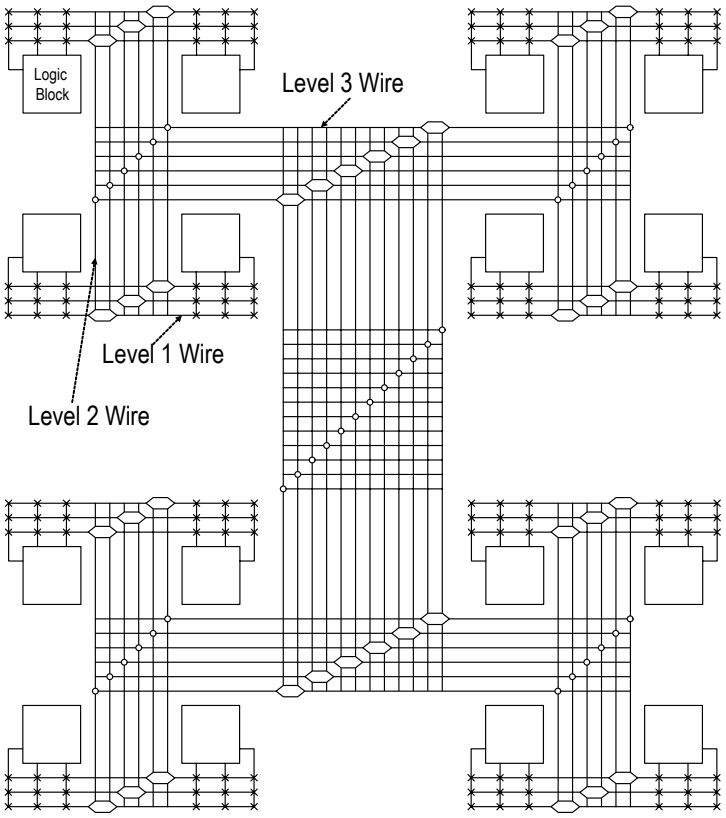


Figure 3.2: Hierarchical routing layout. This figure is adopted from [68] without any changes.

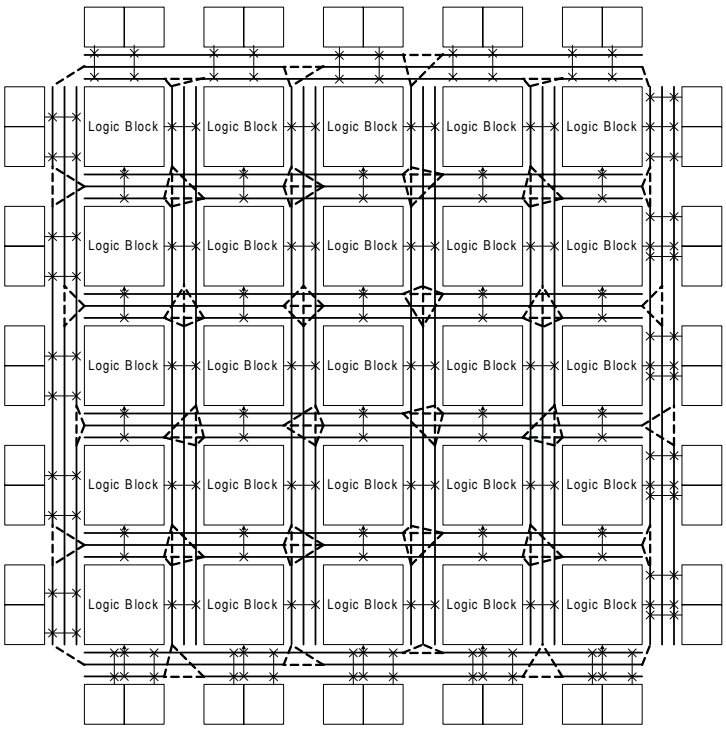


Figure 3.3: Island routing layout. This figure is adopted from [48] without any changes.

The individual logic blocks and routing wires are connected in the local routing structure. Here, connection boxes connect the logic blocks to the routing structure using sparse crossbars. Switch boxes, on the other hand, are placed in the intersection of the horizontal and vertical wiring channels. Three types of switch boxes are shown in Figure 3.4, the disjoint, universal and Wilton type.

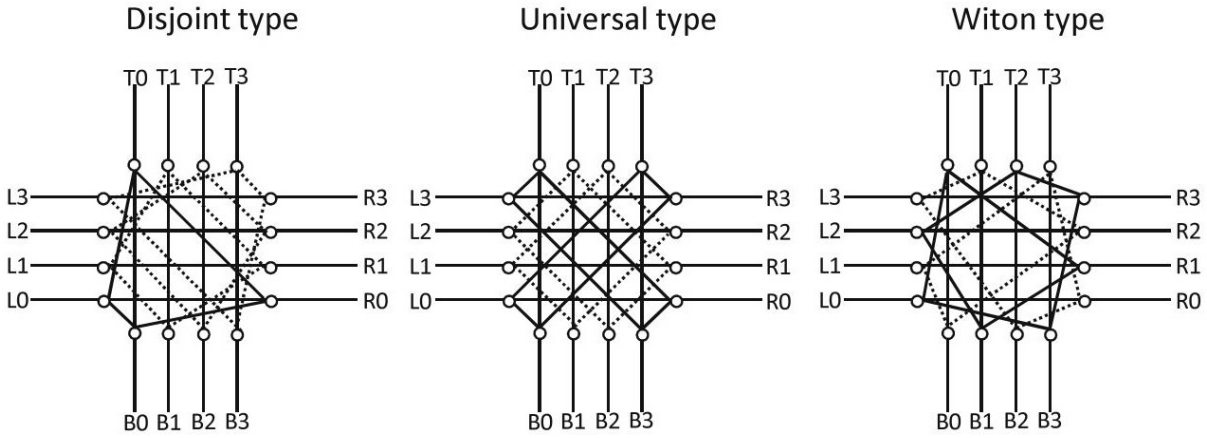


Figure 3.4: Disjoint, universal and Wilton types of switch boxes for a routing structure with four horizontal and vertical lines. This figure is adopted from [2] without any changes.

In a disjoint type of switch box, only channels with the same cardinality can be connected, while the universal switch box type can connect two channel pairs of different cardinality. In case of the Wilton type, the cardinality of each channel is increased by one after two steps, until the initial channel is reached again.

3.1.4 I/O Pads

To connect the fabric of an FPGA with the outside world, I/O blocks are placed around the periphery of the chip that can be connected to the logic blocks through the routing structure. As such, these blocks can operate as user defined I/Os, clock inputs or for supplying power to the various FPGA subsystems, such as the core or the PLLs. The I/O pins can operate at different voltage levels compared to the FPGA's core voltage, and to reduce the number of pins required for this task, I/O pins are grouped into *banks*. The basic features of I/O blocks are listed below:

- Configurable pull-up and pull-down resistors;
- Output buffer with enable operations;
- Latency adjustment with flip-flops;
- Output buffer slew rate control.

Additionally, certain I/O blocks support differential signals and can combine the negative and positive signals without requiring additional logic.

3.1.5 Phase-Locked Loop

Clock signals are a very important building block in digital design circuits. As such, accurate generation of clocks with different frequencies is crucial for the correct functioning of a design implemented in an FPGA. Therefore, PLL blocks are added to the FPGA fabric to generate the required clock signals based on an external reference clock.

Figure 3.5 shows the simplified structure of a PLL. A voltage controlled oscillator (VCO) is the main element in a PLL, which is connected to a phase detector via a feedback circuit. This detector compares the phases of the reference clock F_{ref} to the output of the VCO and, if the phases are not the same, adjusts the input voltage of the VCO. A low-pass filter is used to remove any high frequency components of the input signal, so that the VCO's output is not unstable due to high frequency noise. By including a frequency divider in the feedback circuit, the output frequency of the PLL can be adjusted according to $F_{\text{PLL}} = n \cdot F_{\text{ref}}$.

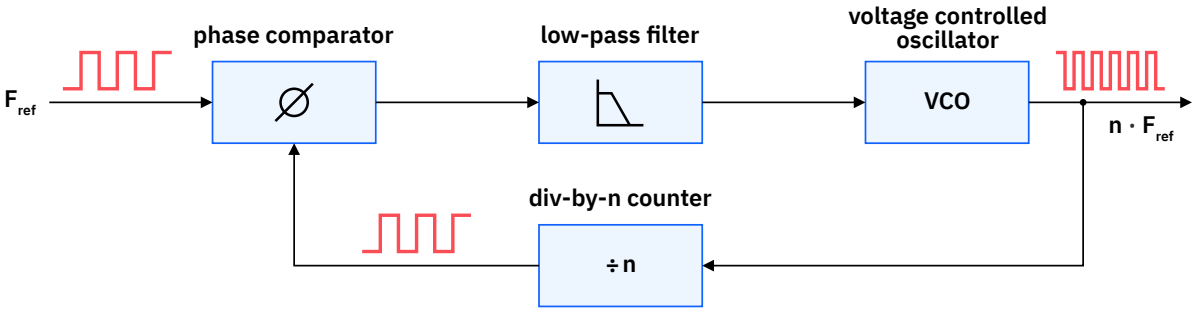


Figure 3.5: Structure of a simple PLL. The output wave has a frequency n times higher than the reference frequency F_{ref}, which is depicted in form of the red waves.

3.1.6 Block RAM

To store large amounts of data, FPGAs use embedded memory cells called BRAMs. These memory cells typically have dedicated input and output signals for data, as well as address and clock lines. If the data bus, address bus and control signals are duplicated, the BRAM can be used as *dual-port* memory, i.e., simultaneous reads and writes are allowed. Otherwise, the BRAM is referred to as a *single-port* memory. Dual-port mode is selected when using a BRAM as a *first-in first-out (FIFO)* memory, which is often the case when dealing with streaming data or clock domain transitions.

3.1.7 DSP Blocks

As one of the main use cases for FPGAs is signal processing, many FPGAs embed specialized hardware for arithmetic operations used in such applications, such as *multiply and accumulate (MAC)* operations. Multiplication using CLBs requires many blocks routed together, as an n-bit $\times$ m-bit multiplication results in an $n{*}m$-bit long number. DSP blocks on the other hand require fewer hardware and routing resources for the same task, resulting in increased performance and reduced wiring delays.

3.2 GateMate Hardware Architecture

The Cologne Chip AG GateMate CCGM1A1 is a D-latch based FPGA manufactured in a 28 nm CMOS process node. The main features of the GateMate are as follows:

- 20480 CLBs, here also referred to as CPEs;
- Eight dedicated general purpose I/O (GPIO) banks, each with 18 I/O pins. A ninth GPIO bank can additionally be made available by setting the configuration bank to an I/O bank after configuration;
- 160 KiB total available BRAM, divided into 32 cells. Each cell has 40 KiB of memory;
- Four PLLs;
- One 2.5 Gbit/s Serializer/Deserializer (SerDes) (not in scope of this thesis).

An overview of the GateMate FPGA's components is shown in Figure 3.6. The following pages describe the components of the GateMate FPGA in more detail.

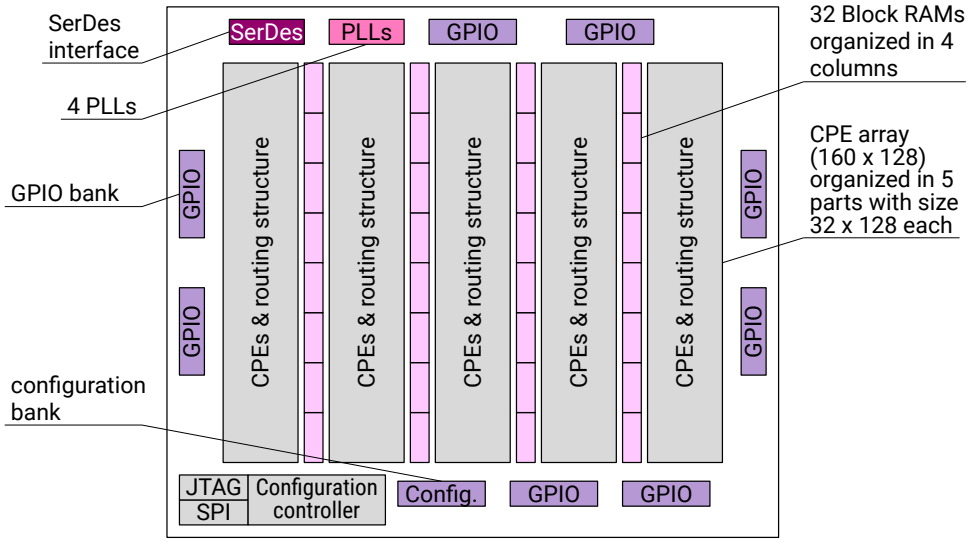
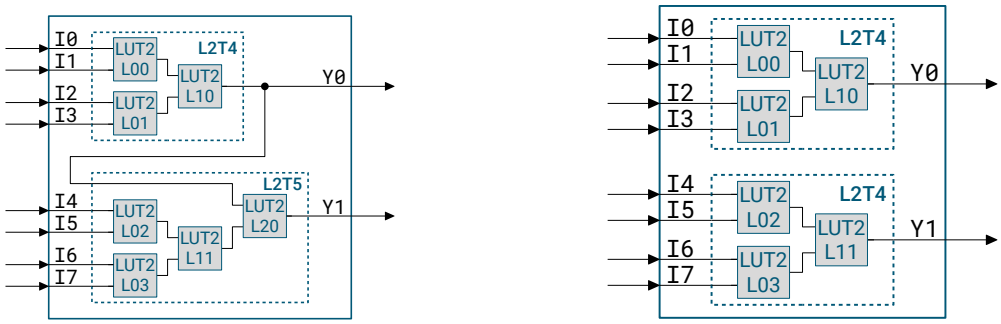


Figure 3.6: GateMate FPGA internal structure overview. This figure is adopted from [16] without any changes.

3.2.1 Cologne Processing Element

A single CPE consists of an 8-input LUT, which is internally organized as multiple 2-input LUTs in a tree-like configuration (Figure 3.7). By chaining multiple smaller LUTs together, a larger LUT can be created. This allows for additional flexibility, as the 8-input LUT can be broken down into smaller LUTs if not all eight inputs are required. The LUT in a CPE supports the following configurations:

1. Two 4-input LUTs;
2. Single 8-input LUT;
3. 4-input MUXs.



(a) 8-input LUT consisting of a tree of 2-input LUTs

(b) Two 4-input LUTs consisting of a tree of 2-input LUTs

Figure 3.7: Possible LUT implementations in a CPE. These figures are adopted from [18] without any changes.

Two flip-flops are used for sequential circuits, and the output of a CPE is switched using a MUX. As shown in Figure 3.8, carry and propagation lines (e.g., *CINX*, *COUTX*, *PINX* ...) are used to distribute fast signals between CPEs. Dedicated *RAM* lines are used to connect a CPE to a BRAM cell. In addition, the arithmetic unit can be used to implement an 1-bit/2-bit full adder or 2-bit multipliers.

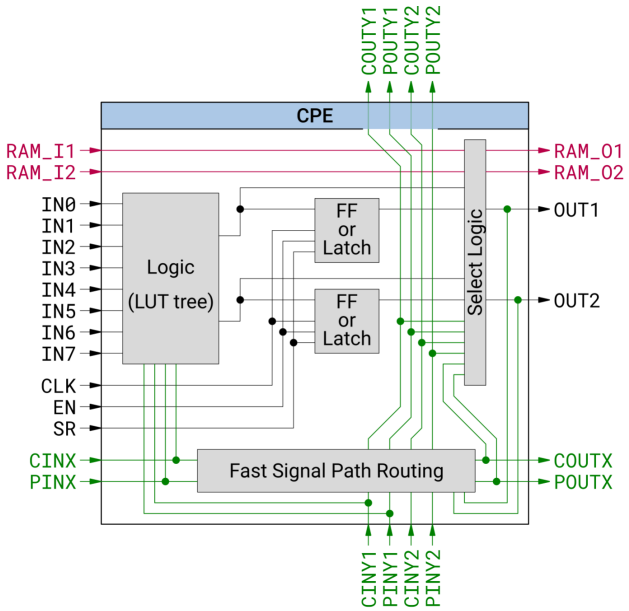


Figure 3.8: Structure of a CPE. This figure is adopted from [16] without any changes.

3.2.2 GPIO Cells

The 162 I/O pads of the GateMate are organized into *GPIO cells*, each containing a pair of signals. Nine of these cells form a single GPIO bank. The GPIO cell can either be used as two individual, single-ended I/O pads, or together as a differential pair. Table 3.2 shows the different voltage standards supported by the I/O pads.

Furthermore, a GPIO cell features integrated flip-flops that support double data rates (DDRs). In single-ended mode, the GPIO pins also support Schmitt trigger inputs, selectable pull-up or pull-down resistors, tristate logic and slew rate control.

Table 3.2: Voltage standards supported by the GateMate I/O pads

Mode	Standard	Voltages
Single-ended	JESD8-5 [40]	2.5 V
Single-ended	JESD8-7 [39]	1.8 V, 1.2 V
Differential	LVDS [64]	2.5 V (1.8 V as min. supply voltage)

The nine GPIO banks are named after their location on the IC's package, and have either the suffix *A*, *B* or *C*. These are: NA, NB, EA, EB, SA, SB, WA (either configuration or I/O bank), WB and WC. Additionally, each pin pair is split up into `_A` and `_B` pins, for the positive and negative signals of differential pairs, respectively.

3.2.3 Routing Structure

The GateMate consists of an 160×128 array of CPEs, interconnected by a 164×132 switch box matrix (Figure 3.9), resulting in an island-style layout. A switch box is placed every other coordinate in the matrix, alternating between Big Switch Boxes (SB_BIGs) and Small Switch Boxes (SB_SMALLs).

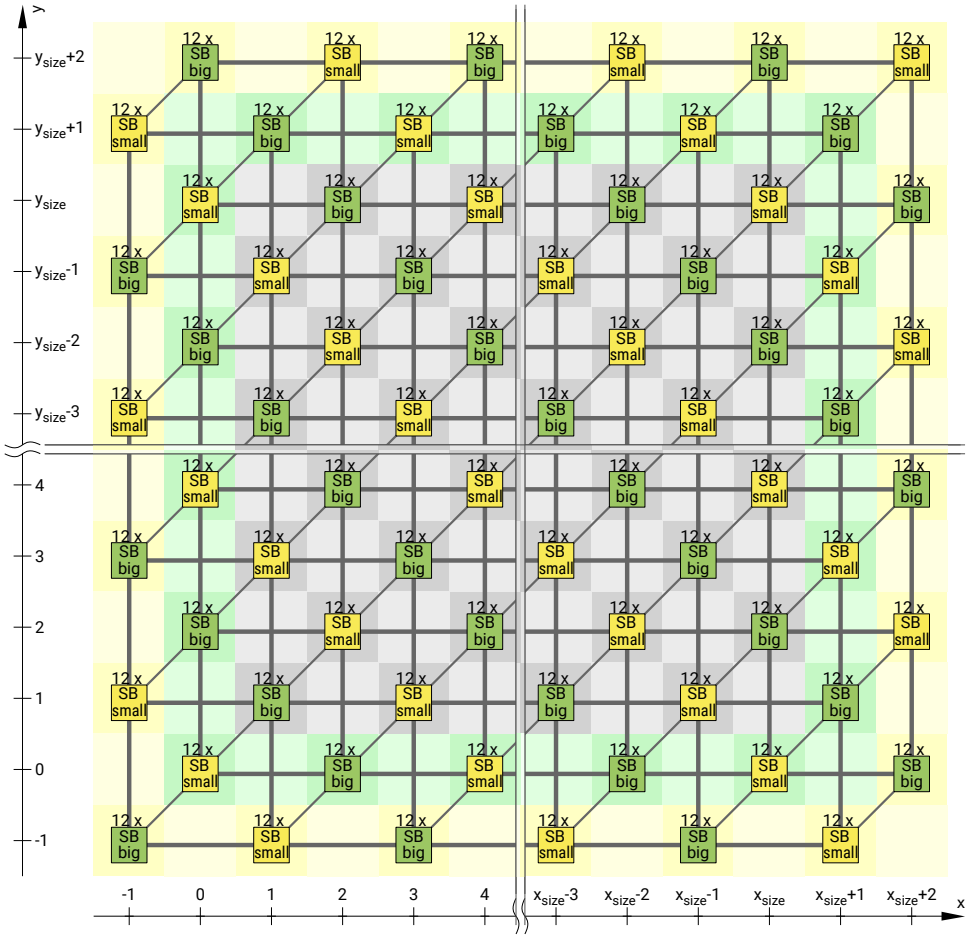


Figure 3.9: Routing structure of the GateMate. This figure is adopted from [16] without any changes.

The only difference between the two switch box types is the number of adjacent switch boxes to which they can be connected. These switch boxes, shown in Figure 3.10, can be connected to neighboring switch boxes horizontally, vertically and diagonally from their lower left and their upper right. The SB_BIG can connect to six other switch boxes, while the SB_SMALLs can only connect two switch boxes. Input and output multiplexers are used to connect a CPE or an I/O pin to the routing structure. In addition, a direct connection from a CPE into the structure is also possible.

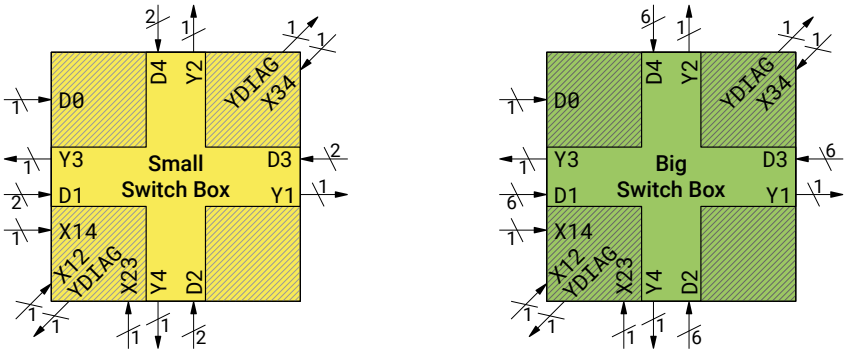


Figure 3.10: Switch boxes used in the GateMate. This figure is adopted from [16] without any changes.

3.2.4 Block RAM

The GateMate CCGM1A1 has 32 BRAM cells, interleaved between the CPEs and the routing structure (see Figure 3.6). Each BRAM cell has a capacity of 40 Kibit, resulting a total available memory of 160 Kibit. A BRAM cell has two independent ports, `A` and `B`, and can be used either in *simple dual port (SDP)* or in *true dual port (TDP)* mode. In TDP mode, both ports `A` and `B` can be used to read and write to memory simultaneously. Each port uses an individual clock, and the data widths can be different, but there is no conflict handling available when accessing the same address at the same time. When using a BRAM cell in SDP mode, port `A` is the write port and port `B` is the read port. This doubles the possible data width while still allowing simultaneous read and write operations.

Each memory cell, shown in Figure 3.11, is composed of several layers around the memory block, and features four signal groups.

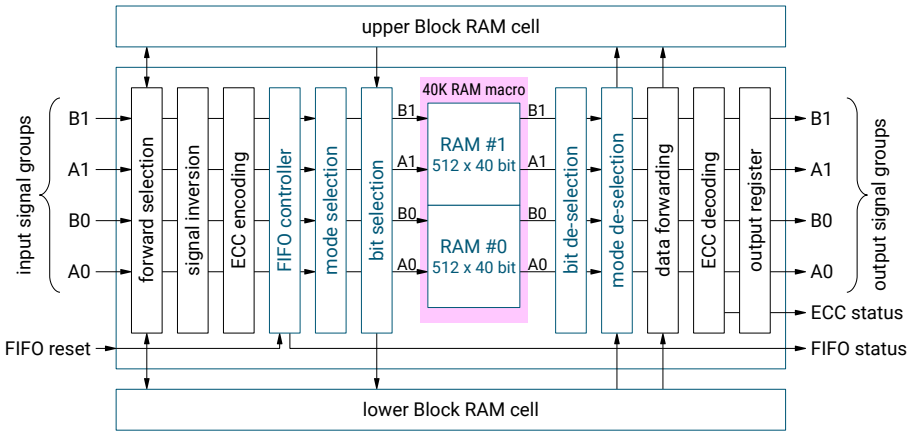


Figure 3.11: Structure of the GateMate BRAM cell. This figure is adopted from [16] without any changes.

The signal groups divide the ports A and B into signals `A0`, `A1`, `B0` and `B1`. This allows the memory block to be configured either as a single unit (40K mode), or split up into two smaller 20 Kibit blocks (20K mode). This also doubles the data width, as in 40K mode the signal groups `A0` and `A1`, as well as `B0` and `B1`, are combined to `A0+A1` and `B0+B1`.

Furthermore, as shown in the layers in Figure 3.11, a BRAM cell supports additional functions. The forward selection can propagate the control and data signals to adjacent memory cells, to increase the usable memory size. In the signal inversion layer, the control signals can be set to active low, while the *error correction code (ECC)* encoding layer, in conjunction with the later ECC decoding layer, handles error detection and correction. This is done using a Hamming code with 39 total bits, 32 data bits and 7 parity bits (Hamming(39,32)), which can correct one bit errors and detect two bit errors. The FIFO layer is deployed when the BRAM is used as a FIFO memory. The mode selection layer forwards the input signals according to either SDP or TDP mode, and the bit selection layer is used for bitwise writing. The bit de-selection and mode de-selection layers have the same function as their input side counterparts, and the data forwarding layer can provide data to adjacent memory cells when configured as a combined memory. Finally, the memory's output can be registered for one clock cycle via the output register layer.

3.2.5 Phase-Locked Loop

Four PLLs are available for clock generation. These are fed by a digitally controlled oscillator (DCO), and for each PLL the reference clock can be injected individually via the following input pins:

- SerDes positive clock input
- Clock input pins `CLK0`, `CLK1`, `CLK2` and `CLK3`

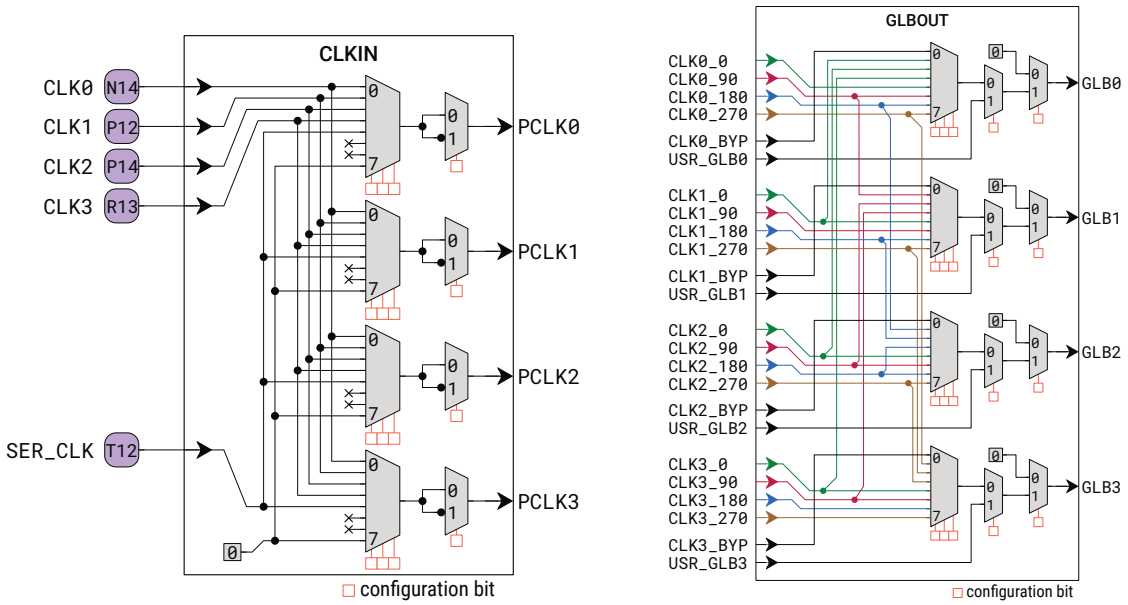
While it is also possible to use any other I/O pin as a clock input, the above pins are fed directly into the PLL without any additional logic. Each PLL generates four output clocks with a $90°$ phase shift. Furthermore, the maximum DCO and PLL frequencies depend on the core voltage. This is shown in Table 3.3.

Table 3.3: Maximum oscillator frequencies when using different core voltages

Mode	V_{core} / V	Max. DCO Frequency / GHz	Max. PLL Frequency / MHz
Low power	0.9	1.000	250.00
Economy	1.0	2.000	312.50
Speed	1.1	2.500	416.75

3.2.6 Global Mesh and Clock Distribution

The Global Mesh is a signal distribution ring around the edge of the GateMate's chip. It consists of four traces that can be used to carry clock signals as well as high fanout signals. Four additional vertical traces are used to connect the mesh to the user circuit Three modules are used to feed signals into the Global Mesh: The *clock input multiplexer* (`CLKIN`, Figure 3.12a), the *PLL wrapper* (Figure 3.12a), and the *mesh output multiplexer* (`GLBOUT`, Figure 3.12b). The clock input multiplexer is used to map the five dedicated clock input pins into the four output signals `PCLK[3:0]`. Optional clock signal conversion is also configured through this block.



(a) Clock input multiplexer to feed the PLL wrappers **(b)** Mesh output multiplexer to feed signals to the Global Mesh

Figure 3.12: Clock input and mesh output multiplexers. These figures are adopted from [16] without any changes.

After the clock input multiplexer, the PLL wrappers (Figure 3.13) are used to generate stable clock signals using the GateMate's PLLs. Next to the signals coming from the clock input multiplexer, user clocks from any other arbitrary input pins can also be used as inputs for the PLL wrapper. The PLL block can also be bypassed if required. Injection of the wrapper's output clocks into the GateMate's fabric is done either through the Global Mesh (using the mesh output multiplexer) or the routing structure (using CPEs).

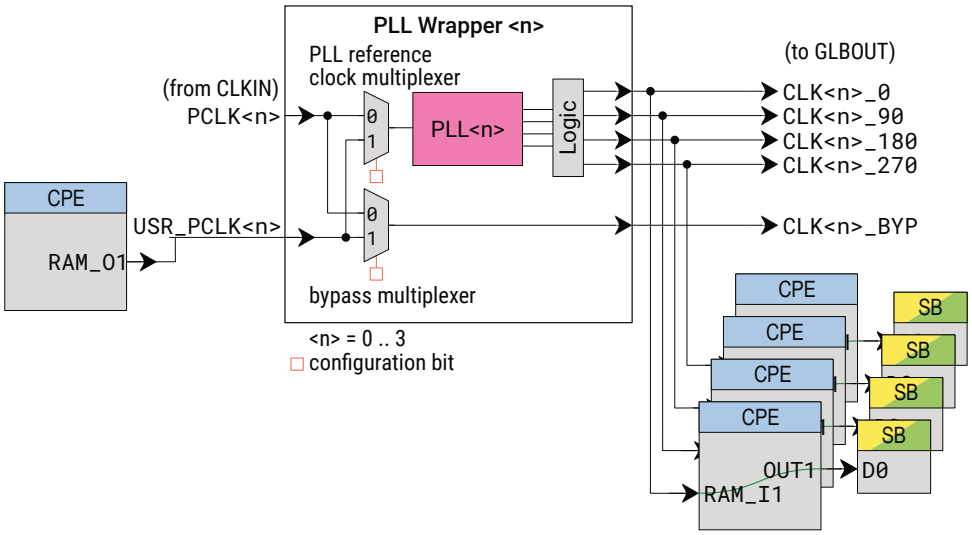


Figure 3.13: Wrapper for the GateMate's PLLs. This figure is adopted from [16] without any changes.

The mesh output multiplexer has four outputs, each for one of the four signal rings of the Global Mesh. The inputs can be either the clock signals the from PLL wrapper (through the PLL or bypassed), or user-defined signals that require fast routing through the Global Mesh. The entire Global Mesh

layout, as well as the clock input path described above, is summarized in Figure A.1.

To extract the signals from the Global Mesh, four pick-up options are available (also shown in Figure A.2):

1. Left Edge Select (LES): These elements are distributed around the left edge for each y position;
2. Bottom Edge Select (BES): These elements are spread around the bottom edge for every x position;
3. SB_BIG: Stacks of Big Switch Boxes that are spread around all four edges of the chip. Each Global Mesh signal is fed into three switch box planes in parallel;
4. CPE array: Each of the four vertical Global Mesh columns can be used to feed the Global Mesh signal into the routing structure.

Which of the four types of Global Mesh signal extraction is used, depends on the `CC_BUFG` usage and weather the `--noclkbuf` synthesis option is used (Table 3.4)

Table 3.4: Overview of the Global Mesh signal extraction types [16]

Clock signals with `CC_BUFG`	`--noclkbuf` Setting	Clocking Scheme(s)
0	Unset	Clock distribution via the routing structure
0	Set	Intake CPE with chained CPE row Intake BES with chained BES row Intake LES with chained CPE row
1..4	Unset	Clock distribution via the Global Mesh
1..4	Set	Intake CPE with chained CPE row Unchained BES row with Global Mesh pick-up Intake BES with chained BES row Intake LES with chained CPE row

Three distinct methods of clock distribution can be selected for a GateMate design. The first one, routing without the Global Mesh, uses the routing structure for clock signals. This does not prevent the use of the Global Mesh for other signals, which can utilize the pick-up options 3 and 4 (SB_BIG and CPE array) for signal injection and extraction. In terms of electromagnetic interference (EMI), this option performs the best. However, by not using the Global Mesh with the dedicated clock resources, skew can be high, which in turn reduces the maximum usable clock frequency [16].

The second method is clock distribution via the Global Mesh. This uses the shortest path from the mesh to the user's design and the pick-up option 3 (SB_BIG) to extract the clock signal. If less than four clock signals are used in the Global Mesh, the remaining lines can be used by custom signals. These can be accessed via pick-up options 3 and 4. This method of clock distribution performs poorer regarding EMI than the first method, but features a reduced clock signal skew [16].

The third, and final, method is to use the carry & propagation signal lines (CP-lines) for clock routing. This method can also be combined with the first and second method. A clock signal is picked up via an input element (e.g. a CPE or a BES), and is then distributed to its destination via the CPE's CP-lines. One limitation is however, that only one clock signal and one enable signal (which is optional) can be routed using this method. To use the third method, the implementation process must be invoked with the `++clk_CP` option.

3.3 GateMate Evaluation Board

The GateMate evaluation board (Figure 3.14) can be used during the development process of embedded systems based on the GateMate FPGA.

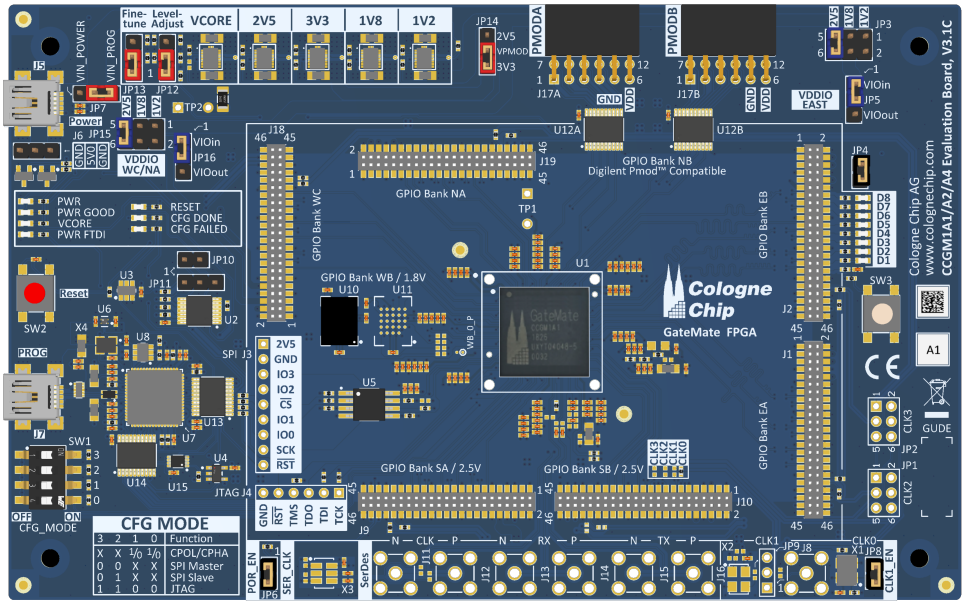


Figure 3.14: Overview of the GateMate evaluation board. This figure is adopted from [17] without any changes.

3.3.1 Power Supply

Three options are available to the user for powering the evaluation board, which are shown in Figure 3.15. `J5` and `J7`, both being *Universal Serial Bus (USB)* connectors, can be connected to an USB socket to provide 5 V. The difference between these two connectors is that `J7` also functions as the programming USB port, which will be covered in a later section. Additionally, the pin header `J6` is also designed as a power input, with an input voltage range of 4 V to 5 V. Jumper `JP7` is used as a power input selection into the power supply unit (Figure 3.16). In here, the voltages listed in Table 3.5 are generated through MPS MPM3833C DC-DC converters, as well as the power good signal required for the power-on reset functionality. Depending on the use case, the core's supply voltage V_{core} can be adjusted from 0.9 V to 1.1 V with jumpers `JP12` and `JP13`, to accommodate the set operation mode of the GateMate.

Table 3.5: Voltages provided by the evaluation board power supply unit

Designation	Voltage / V	Usage
`Vcore`	0.9 … 1.1	Supply voltage for the FPGA core, SerDes and PLLs
`VIO_1V2`	1.2	Optional for I/O banks
`VIO_1V8`	1.8	Optional for I/O banks, HyperRAM
`VIO_2V5`	2.5	Optional for I/O banks, Peripheral module (Pmod) interface
`VIO_3V3`	3.3	Pmod interface

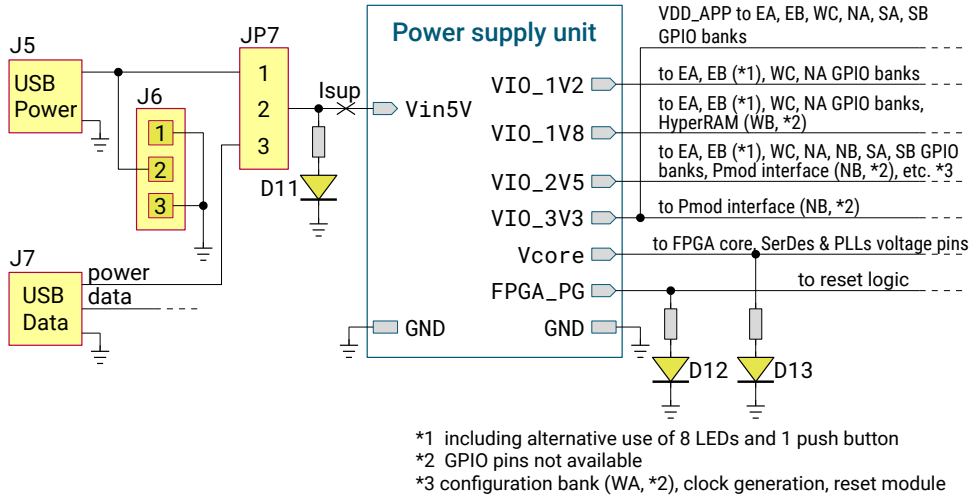


Figure 3.15: Power supply topology of the GateMate evaluation board. This figure is adopted from [17] without any changes.

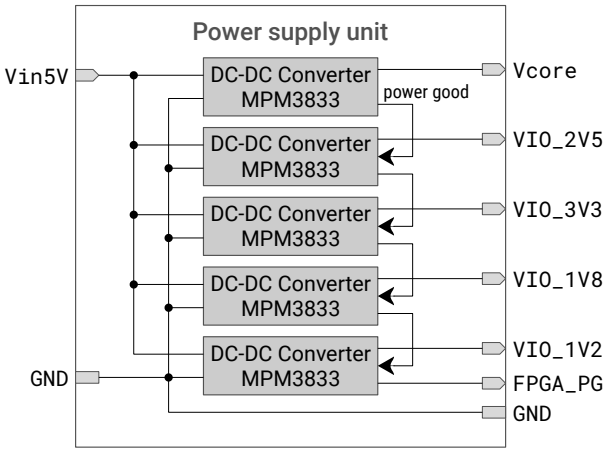


Figure 3.16: Power supply unit internals. Noteworthy is the forwarding of the power good signal from V_{core} to the I/O voltage converters. The FPGA_PG signal is only set when all converters report power good. This figure is adopted from [17] without any changes.

3.3.2 I/O Connections

To interface with user application modules, six of the eight I/O banks (NA, EA, EB, SA, SB and WC) are broken out to pin headers (J1, J2, J9, J10, J18 and J19), whose pin assignments are shown in Figure 3.17.

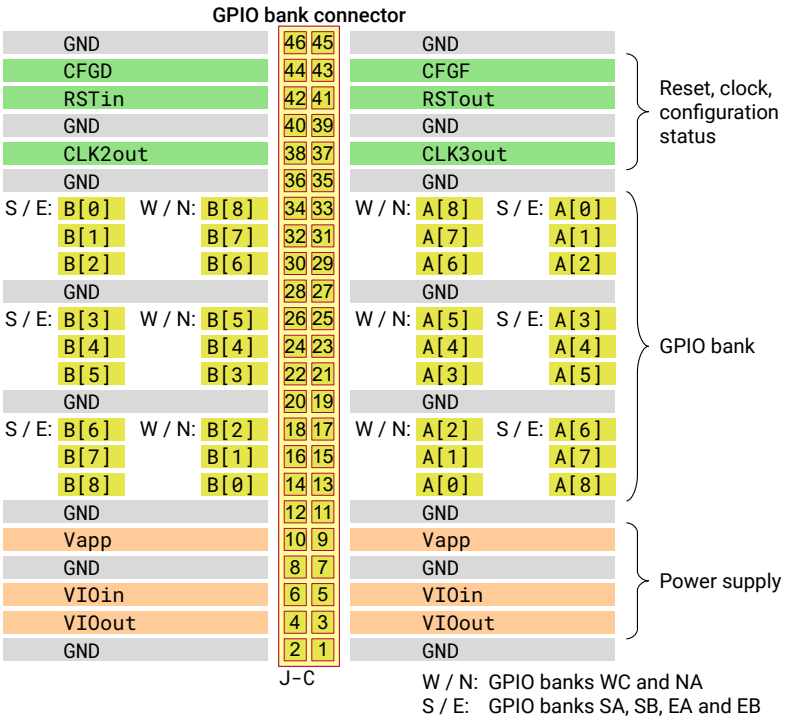


Figure 3.17: Pinout of the GPIO connectors for each GPIO bank. This figure is adopted from [17] without any changes.

While all pin headers are arranged in a manner that differential signals can be used, banks EA and EB are additionally inter-pair and intra-pair length matched. Each bank connector also features the following additional input and output connections, which can be used by external modules connected to the evaluation board:

- `CFGD`: Output of the FPGA pin `CFG_DONE`;
- `CFGF`: Output of the FPGA pin `CFG_FAILED_N`;
- `RSTout`: Reset signal originating from the external module to the FPGA;
- `RSTin`: Reset signal originating from the FPGA to the external module;
- `CLK2out`: External clock signal into the FPGA pin `CLK2`;
- `CLK3out`: External clock signal into the FPGA pin `CLK3`;
- `Vapp`: Fixed 3.3 V, can be used to power components on the external module;
- `V_IO_IN`: When supplying the FPGA's I/O banks with a voltage from the evaluation board, the voltage is also forwarded to the external module via this pin;
- `V_IO_OUT`: When supplying the FPGA's I/O banks with a voltage from the external module, this pin is used.

The `_B` pins of bank EB are connected to light emitting diodes (LEDs) and a push button. Furthermore, a 10 MHz oscillator is attached to the `CLK0` pin input. The NB I/O bank is connected to two Pmod [22] compatible connectors, while the WA bank is reserved for the configuration of the FPGA.

Lastly, an external Infineon S27KS0641 HyperRAM dynamic RAM (DRAM) IC is attached to bank WB. The selectable voltages for each GPIO bank are shown in Table 3.6.

Table 3.6: Voltages available on the GPIO banks [17]

GPIO Bank	V_{IO_1V2}	V_{IO_1V8}	V_{IO_2V5}	V_{IO_3V3}	V_{IO_APP}
NA	X	X	X		X
NB			X	X	
EA	X	X	X		X
EB	X	X	X		X
SA			X		
SB			X		
WC	X	X	X		X

3.3.3 Programming Interface

Once the implementation is finished, the bitstream can be uploaded to the GateMate through the USB bridge, which in this case is implemented by an Future Technology Devices International (FTDI) FT2232HQ USB to serial converter. This IC, together with a Macronix MX25R6435F flash memory, is also used for non-volatile storage of the configuration. To accommodate the different voltage levels between the FT2232HQ USB bridge and the configuration bank, a Texas Instruments (TI) SN74CB3T3245 8-bit bus switch with level shifter is used. This IC translates the FT2232HQ's 3.3 V signals into 2.5 V level signals, and vice-versa. The described configuration architecture is shown in Figure 3.18. Switch `SW1` is used to set the configuration mode to SPI Master Mode, SPI Slave Mode or JTAG.

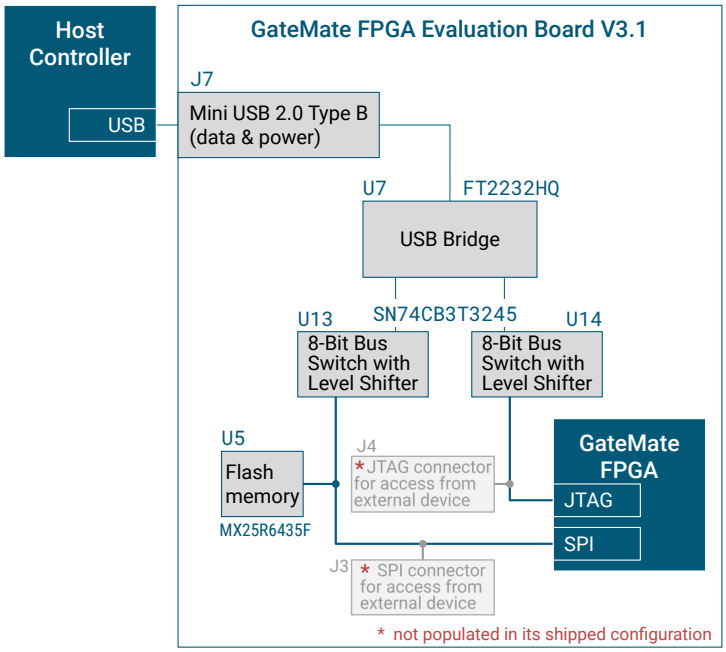


Figure 3.18: Overview of the FPGA configuration interface. The host controller represents the device from which the bitstream is uploaded. This figure is adopted from [17] without any changes.

3.3.4 Clock and Reset

An onboard ECS ECS-5032MV oscillator generates a 10 MHz clock signal, which is fed to the GateMate's clock mesh via `CLK0`. Additional clock signals can also be provided, either originating from the evaluation board or externally. Regarding the `CLK1` input, a second oscillator can be installed onto the evaluation board. Alternatively, an external clock can be injected into this clock pin by using the SubMiniature version A (SMA) jack `J8`. Clock inputs `CLK2` and `CLK3` can be fed through the corresponding inputs on the GPIO bank connectors, as described in Section 3.3.2. The SerDes input clock can also be generated via a dedicated oscillator, or by feeding a single-ended or differential clock signal through jacks `J11` and `J12`.

The reset signal on the GateMate evaluation board can be triggered by various sources, including the button `SW2`, a power good signal set to `LOW`, through the GPIO bank connectors and more. The reset module of the evaluation board is illustrated in Figure 3.19.

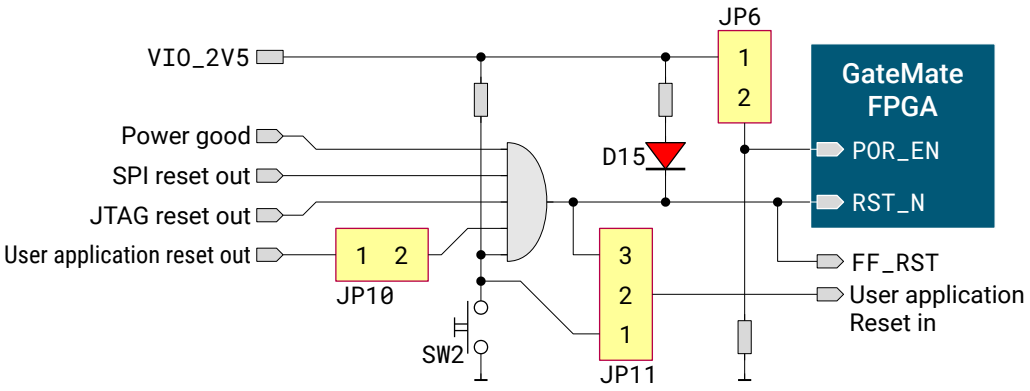
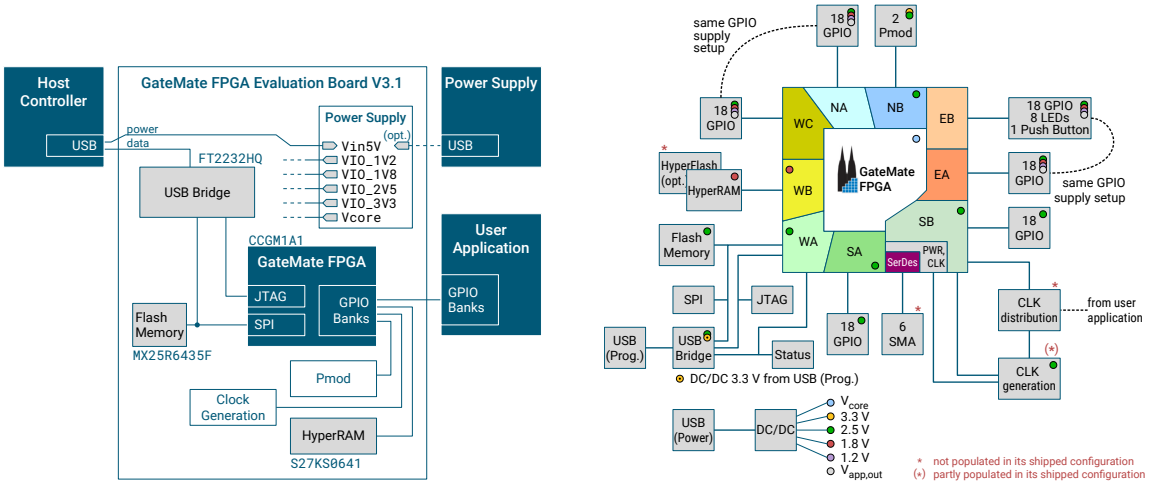


Figure 3.19: Overview of the reset module of the GateMate evaluation board. This figure is adopted from [17] without any changes.

A summarizing block diagram of the evaluations board's components is shown in Figure 3.20a. Furthermore, an overview of the I/O bank assignments on the GateMate FPGA is given in Figure 3.20b.



(a) Block diagram of the GateMate evaluation board **(b)** Evaluation board pin mappings to the GateMate's I/O banks

Figure 3.20: Block diagram of the evaluation board as well as a pin to I/O bank overview. These figures are adopted from [17] without any changes.

3.4 Software Design Flow

The design flow for developing an application for the GateMate FPGA can be divided into four distinct steps: *design entry*, *synthesis*, *implementation*, and *configuration* (Figure 3.21).

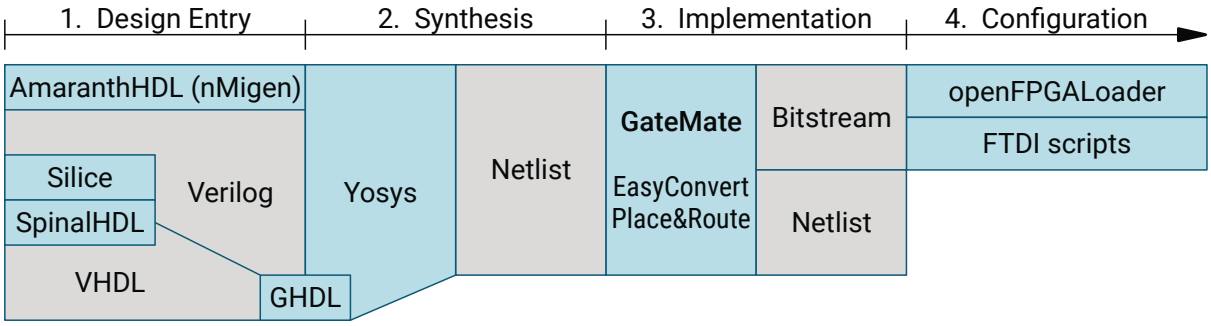


Figure 3.21: Example of a simple design flow for a GateMate FPGA. This figure is adopted from [16] without any changes.

A design is entered with the HDLs Verilog or VHDL, using any text editor. The synthesis tool used is Yosys[1], an open-source Verilog RTL synthesis framework. If VHDL is selected as the HDL language, the G Hardware Design Language (GHDL)[2] VHDL analyzer is used via the ghdl-yosys-plugin[3] to generate a suitable input for Yosys. HLS is also supported, if a Verilog backend is available. Invocation of the synthesis process is performed as shown in Listing 3.1, where `<FILES>` represents all required HDL files, and `<TOP>` specifies the name of the top-level module.

Shell
```
# Verilog
yosys -l yosys.log -p 'read_verilog <FILES>; synth_gatemate -top <TOP> -vlog <TOP>_synth.v
# VHDL
yosys -l yosys.log -p 'ghdl --warn-no-binding -C --ieee=synopsys <FILES> -e <TOP>;
    synth_gatemate -top <TOP> -vlog <TOP>_synth.v'
```

Listing 3.1: Commands to invoke the synthesis process

The GateMate synthesis command provides a number of options. Those used in this thesis are discussed below, a complete list of all available options can be found in [16].

- `-top <name>`: Sets the top-level module of the design to `<name>`;
- `-vlog <filename>`: Writes the synthesis output in Verilog to the specified file;
- `-noclkbuf`: Disables the automatic insertion of `CC_BUFG` clock buffers;
- `-set keep 1`: Keeps architecture as-is, does not remove redundant logic (e.g. when using triple modular redundancy (TMR)).

The synthesis process generates the *<TOP>_synth.v* file, which is a gate-level netlist representation of the developed design, with additional mappings to Cologne Chip primitives (e.g., `CC_LUT2` for a two input LUT or `CC_DFF` for a D-type flip-flop). This netlist is required for the implementation phase, and can also be used in a post-synthesis simulation with an appropriate test bench.

The *P&R* tool, invoked as shown in Listing 3.2, performs a multistep process. First, the Yosys output netlist is converted to a generic netlist for speed or area optimization [16]. The resulting netlist is then used in the mapping, placement, and routing steps. The latter two are depend on the results of the STA after routing, and are repeated if timing constraints are not met.

[1] https://github.com/YosysHQ/yosys
[2] https://github.com/ghdl/ghdl
[3] https://github.com/ghdl/ghdl-yosys-plugin

Shell
```
p_r -i <TOP>_synth.v -o <TOP> -ccf <TOP>.ccf > impl.log
```

Listing 3.2: Command to start the design implementation

The P&R tool also provides options to customize the implementation process. The options relevant to this thesis are listed below; a complete list of all available options can be found in [16]:

- `-ccf`: Name of the constraints file to use;
- `-cCP`: Do not use the carry and propagation lines for CPE clock and enable inputs;
- `-c`: Do not optimize design for area usage;
- `-rifs`: Do not remove redundant CPEs (unofficial feature);
- `+crf`: Generate cross-reference file.

During this process, the aforementioned constraint file is taken into account. This file defines the input and outputs pins, as well as additional I/O parameters. The pin description syntax (shown in Listing 3.3) is composed of three mandatory parts, *direction*, *name*, *location*, and optional *parameters*:

Direction
`Pin_in`, `Pin_out` or `Pin_inout` are used to define input, output or tristate pins respectively;

Name
The identifier used in the top-level module;

Location
The ball name on the GateMate's package to which the signal is connected to;

Parameters
Optional settings such as Schmitt-trigger inputs etc. A full list of options is documented in [16].

CCF Constraint File Format
```
<pin-direction> "<pin-name>" Loc = "<pin-location>" | <opt.-constraints>;
```

Listing 3.3: Syntax to constrain an I/O pin

Running the P&R tool generates several files; the most important ones are described in Table 3.7.

Table 3.7: Files generated by the Cologne Chip P&R tool

Filename	Description
`<TOP>.net`	Netlist generated by P&R from synthesis
`<TOP>.sdf`	Standard Delay Format (SDF) timing file used for timing-based post-implementation simulation
`<TOP>.cfg.bit`	FPGA configuration bitstream
`<TOP>.cdf`	Input file for Cologne Chip CPE Vision
`<TOP>.place`	Placement file for CPEs
`<TOP>.refwire`	Reference file between CPEs and netlist
`<TOP>_00.cfg`	Verbose GateMate configuration file
`<TOP>_00.v`	Verilog netlist generated by P&R from synthesized input

The bitstream file *<TOP>_00.cfg.bit* contains the information needed to configure the GateMate with the developed design. This can be flashed via JTAG into the GateMate's configuration memory using openFPGALoader[4], with the corresponding command shown in Listing 3.4.

Shell

```
# Configure GateMate via JTAG
openFPGALoader -b gatemate_evb_jtag <TOP>_00.cfg.bit

# Configure GateMate via SPI
openFPGALoader -b gatemate_evb_spi -m <TOP>_00.cfg.bit

# Upload configuration to external flash via JTAG-SPI-bypass
openFPGALoader -b gatemate_evb_jtag -f --verify <TOP>_00.cfg.bit

# Upload configuration directly to external flash via SPI
openFPGALoader -b gatemate_evb_spi <TOP>_00.cfg.bit
```

Listing 3.4: Commands to upload the bitstream configuration to the GateMate FPGA

3.4.1 Build Automation

In addition to using the commands above, the build process can also be automated through a Makefile. By default, the file and folder structure of a GateMate project is not specified. However, it is useful to structure projects that consist of multiple file types into subfolders for better organization. This also helps when defining a Makefile, as the locations of files are not arbitrary. Therefore, the following folder structure is suggested:

```
project-dir/
├── log
│   ├── (synth.log)
│   └── (impl.log)
├── net
│   └── (<TOP>_synth.v)
├── src
│   ├── <TOP>.v
│   └── <TOP>.ccf
└── Makefile
```

Figure 3.22: Suggested directory structure for a GateMate project. Note that the files in brackets are generated by the synthesis and implementation tools, and are therefore not present when the project is created.

All Verilog and VHDL source files are located in *src*, as well as the used constraints file. The Verilog file generated by Yosys is stored in the *net* directory. The logfiles generated during synthesis and implementation are placed under the *log* folder. As mentioned above, the Cologne Chip P&R tool generates several output files, which are stored under *project-dir*.

With this project structure, a Makefile like Listing 3.5, adapted from the official Cologne Chip examples, can be used to automate the build process. This file contains the following commands:

- `synth`: Alias for `synth_vlog`;
- `synth_vlog`: Invokes the Verilog synthesis process as documented in Listing 3.1;
- `synth_vhdl`: Invokes the VHDL synthesis process as documented in Listing 3.1;

[4] https://github.com/trabucayre/openFPGALoader

- `impl`: Invokes the implementation process as documented in Listing 3.2;
- `jtag`: Invokes the JTAG upload process as documented in Listing 3.4;
- `jtag-flash`: Invokes the JTAG flash process as documented in Listing 3.4;
- `spi`: Invokes the SPI upload process as documented in Listing 3.4;
- `spi-flash`: Invokes the SPI flash process as documented in Listing 3.4;
- `all`: Runs the commands `synth`, `impl` and `jtag` sequentially;
- `clean`: Removes all files generated during synthesis and implementation, as well as all logfiles.

The only changes a user needs to make are in the first 6 lines. The variable `TOP` must be set to the name of the top-level module, all other source files are gathered using the shell commands in line 14 for Verilog and line 15 for VHDL. Lines 3 and 4 can be used to set custom options for Yosys and P&R. Finally, line 6 sets the location of the Cologne Chip toolchain.

Makefile

```
1  ## custom settings
2  TOP =
3  YOSYS_FLAGS +=
4  PR_FLAGS +=
5
6  CC_TOOLCHAIN = ~/cc-toolchain-linux
7
8  ## toolchain
9  YOSYS = $(CC_TOOLCHAIN)/bin/yosys/yosys
10 PR    = $(CC_TOOLCHAIN)/bin/p_r/p_r
11 OFL   = $(CC_TOOLCHAIN)/bin/openFPGALoader/openFPGALoader
12
13 ## target sources
14 VLOG_SRC = $(shell find ./src/ -type f \( -iname \*.v -o -iname \*.sv \))
15 VHDL_SRC = $(shell find ./src/ -type f \( -iname \*.vhd -o -iname \*.vhdl \))
16
17 ## misc tools
18 RM = rm -rf
19
20 ## toolchain targets
21 synth: synth_vlog
22
23 synth_vlog: $(VLOG_SRC)
24     $(YOSYS) -qql log/synth.log -p 'read -sv $^; synth_gatemate -top $(TOP) -nomx8 $(
           YOSYS_FLAGS) -vlog net/$(TOP)_synth.v'
25
26 synth_vhdl: $(VHDL_SRC)
27     $(YOSYS) -ql log/synth.log -p 'ghdl --warn-no-binding -C --ieee=synopsys $^ -e $(TOP);
           synth_gatemate -top $(TOP) -nomx8 $(YOSYS_FLAGS) -vlog net/$(TOP)_synth.v'
28
29 impl:
30     $(PR) -i net/$(TOP)_synth.v -o $(TOP) -ccf src/$(TOP).ccf $(PR_FLAGS) > log/$@.log
31
32 jtag:
33     $(OFL) $(OFLFLAGS) -b gatemate_evb_jtag $(TOP)_00.cfg
34
35 jtag-flash:
36     $(OFL) $(OFLFLAGS) -b gatemate_evb_jtag -f --verify $(TOP)_00.cfg
37
38 spi:
39     $(OFL) $(OFLFLAGS) -b gatemate_evb_spi -m $(TOP)_00.cfg
40
41 spi-flash:
42     $(OFL) $(OFLFLAGS) -b gatemate_evb_spi -f --verify $(TOP)_00.cfg
43
44 all: synth impl jtag
45
46 clean:
47     $(RM) log/*.log
```

```
48      $(RM) net/*_synth.v
49      $(RM) *.history
50      $(RM) *.txt
51      $(RM) *.refwire
52      $(RM) *.refparam
53      $(RM) *.refcomp
54      $(RM) *.pos
55      $(RM) *.pathes
56      $(RM) *.path_struc
57      $(RM) *.net
58      $(RM) *.id
59      $(RM) *.prn
60      $(RM) *_00.v
61      $(RM) *.used
62      $(RM) *.sdf
63      $(RM) *.place
64      $(RM) *.pin
65      $(RM) *.cfg*
66      $(RM) *.cdf
```

Listing 3.5: Simple makefile to automate the GateMate build process

This Makefile can be used to start the complete design flow through the command `make all`, without having to type the synthesis and implementation commands and their required parameters.

3.4.2 Manual Placement of CPEs

In addition to assigning I/O pins to signals, the constraints file can also be used to manually place CPEs at predefined positions. The command shown in Listing 3.6 constrains the CPE with the number `<number>` to the coordinates (`<x>`, `<y>`) in the GateMate's routing structure.

CCF Constraint File Format

```
1 CPE <number> PosX <x> PosY <y>;
```

Listing 3.6: Syntax to constrain an CPE to a specified coordinate

The *.refwire* file is used to obtain the number of the CPE to place. In this file each CPE is associated with the corresponding signal name of the netlist generated by the P&R tool. The concept of placing CPEs manually is explored in more detail in Section 5.6, where CPEs are placed in series to form a ring oscillator.

4

PCB Design

The CHARM mixed field facility [51] (top down image shown in Figure 4.1) is dedicated to the qualification of components and systems intended for use in the LHC. It is fed by a 24 GeV primary proton beam from the PS accelerator. This beam then either strikes a copper, aluminum or perforated aluminum target to produce a secondary radiation field with a composition similar to the radiation field present in the LHC. An additional 20 cm concrete or iron shielding can be used to modulate the field. By using different test positions, and the aforementioned shields, the intensity and spectrum of the radiation field can be tailored to the specific component or system and it's intended environment.

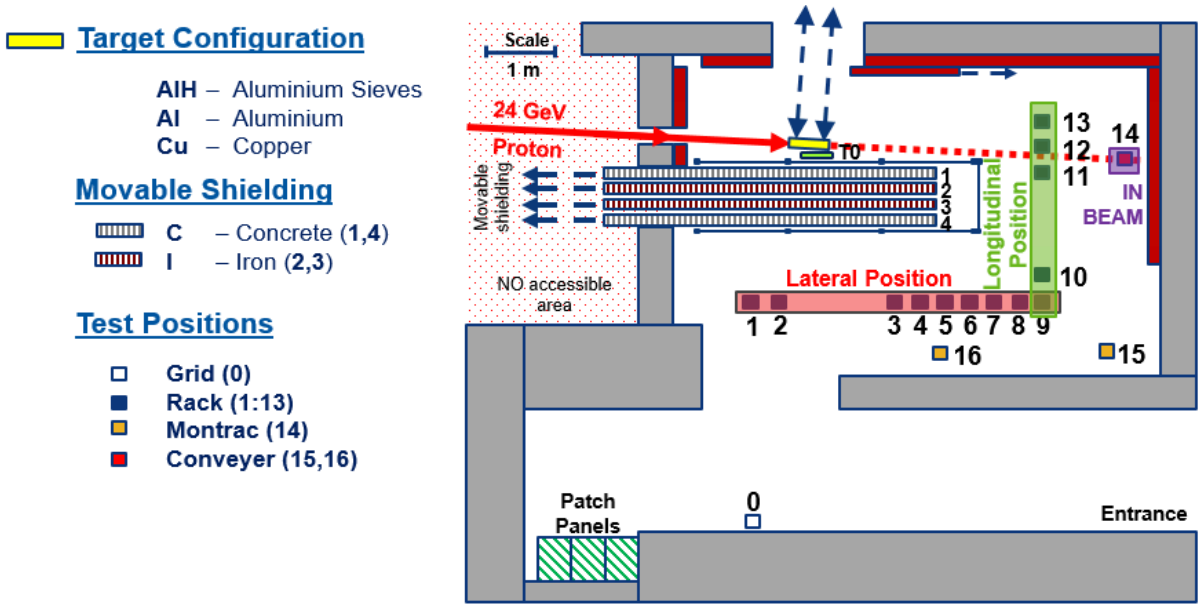


Figure 4.1: Layout of the CHARM facility. Each number represents a position where a DUT can be placed, with 0 representing the position G0 designated for the CRaTeBo. This figure is adopted from [32] without any changes.

4.1 CRaTeBo Carrier Card

One disadvantage of FPGA testing at CHARM is that standard evaluation boards are not suitable for this task, as all components on the boards are exposed to radiation. This can lead to erroneous results

because the true cause cannot be clearly distinguished. For this reason, the CRaTeBo was designed, which can be used to interface a DUT FPGA system-on-module (SoM) with a communication board and optionally FMC compatible devices. This board, consisting of a carrier board and various submodules, is to be permanently mounted at the G0 position at CHARM, and is intended to be used as an FPGA test platform. A block-level schematic of the CRaTeBo is shown in Figure 4.2.

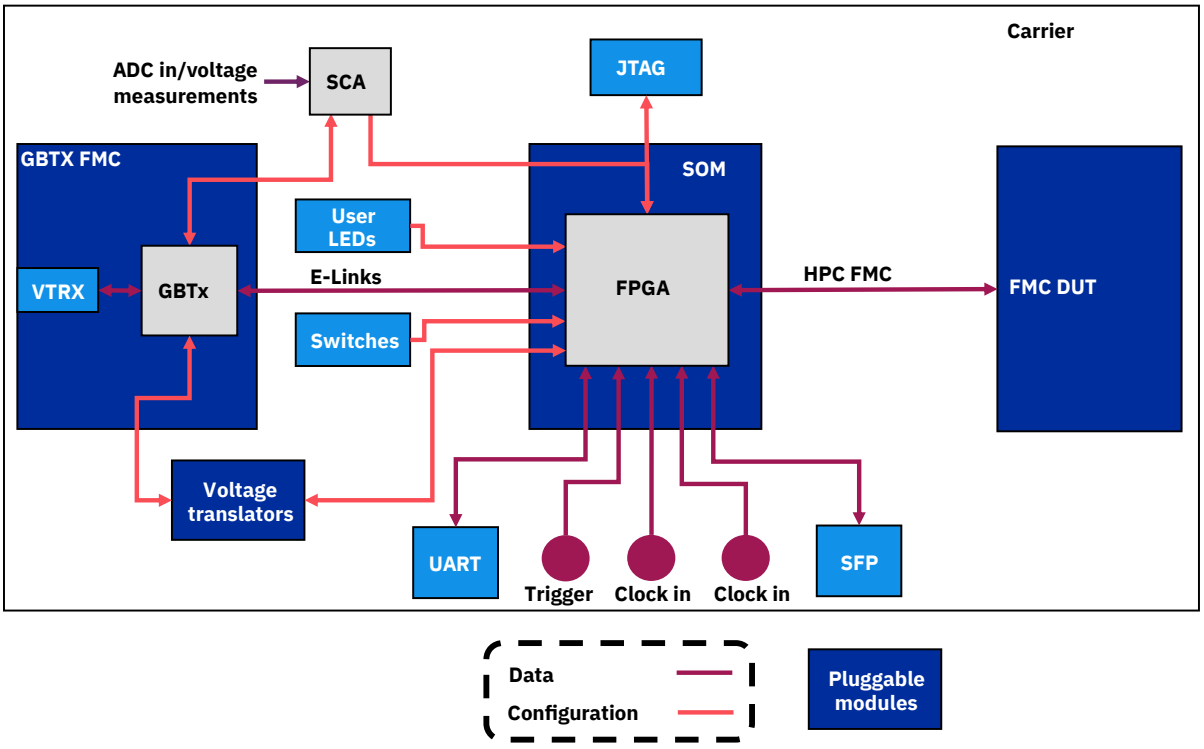


Figure 4.2: Block level overview of the CRaTeBo

4.1.1 Overview

The entire carrier board, with the PCB shown in Figure 4.3, is designed around pluggable modules. This allows for easy replacement of defective components when they have reached their maximum tolerable radiation dose. In addition, the voltage regulators can be freely replaced according to the voltage levels required for the current application.

Additional modules can also be connected via two FMC connectors (the FMC standard is described in more detail in Section 4.1.2). The FMC DUT connector is designed to be used with extension cards when required by the application under test and conforms to the FMC standard. The communication FMC connector is partially compliant with the FMC standard and is specifically designed for an interface card that allows the FPGA to communicate with a backend. Compared to a fully compliant connector, the communication FMC features the low pin count (LPC) connections without JTAG and partially populated high pin count (HPC) signals. The Link-GBT Based Expandable Front-End (L-GEFE)[1] mezzanine card is used for this purpose. This card consists of the GBTx IC, which is a SerDes ASIC based on the GigaBit Transceiver (GBT) protocol [8]. Communication with the FPGA is done through an E-Link [52] interface. This is an electrical interface specifically designed for radiation-hard and low-power chip-to-chip communication and is based on the scalable low-voltage signaling (SLVS)

[1] https://ohwr.org/project/gefe/wikis/home

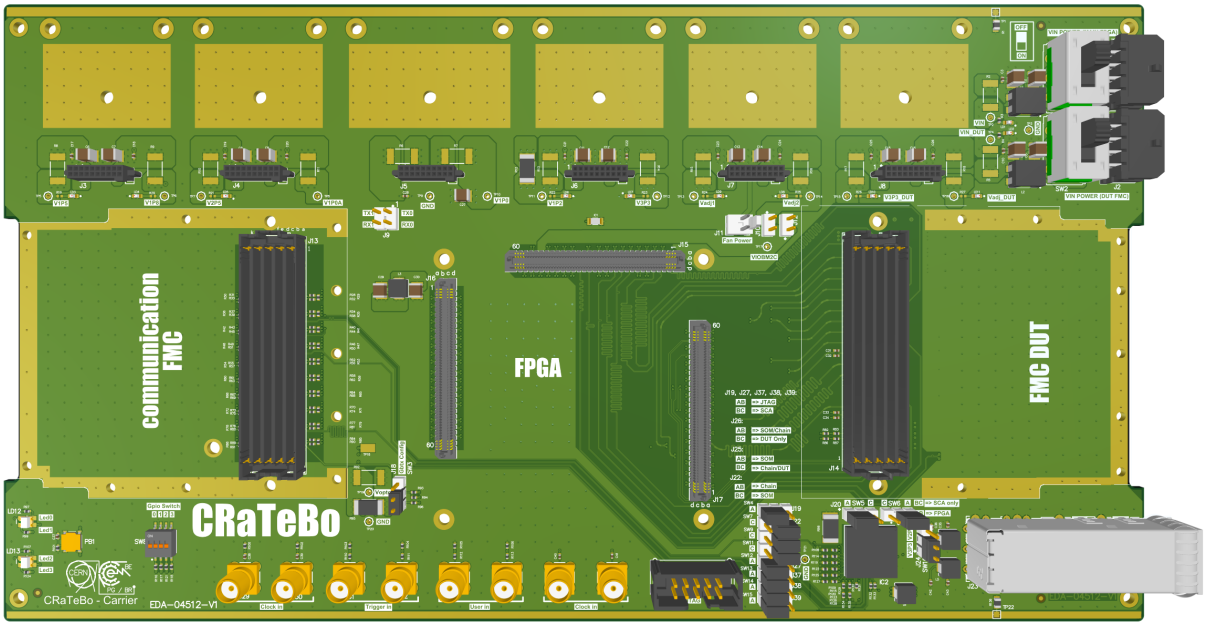


Figure 4.3: 3D rendering of the CRaTeBo carrier board

electrical standard [42]. The SLVS standard defines a differential signal with a voltage swing of 200 mV at a common mode voltage of 200 mV and a load of 100 Ω. The resulting differential voltage is therefore 400 mV. Furthermore, the data transfer from the L-GEFE to a backend (e.g., a computer in a radiation-safe area) is done via CERN's versatile link [3] interface, a radiation tolerant, high-speed optical link. The FPGA is programmed using JTAG or the Slow Control Adapter (SCA) ASIC [12], which is also part of the GBT chip-set and specifically designed as a radiation tolerant chip to distribute control and monitoring signals to front-end electronics on the detectors in the LHC. The carrier board also has four LEDs, four switches and a push button for user input. Additional differential inputs are also available via SMA coaxial connectors, which are connected to two clock in signal pairs, one trigger-in signal pair and one user-in signal pair.

The main component of CRaTeBo, the FPGA, is connected to the carrier board via a custom SoM board and three Samtec ADF6 connectors (ADF6-60-03.5-L-4-2-A-TR on the SoM and ADF6-60-03.5-L-4-2-A-TR on the carrier). The first SoM board designed for CRaTeBo is based on the Microsemi MPF300TS-1FCG1152I FPGA, which was validated in [59] for a TID up to 3 kGy.

4.1.2 Communication

As shown in Figure 4.2, there are four methods available for external data communication with the FPGA SoM:

- An HPC FMC DUT card;
- A Small Form-factor Pluggable (SFP) transceiver interface;
- Two universal asynchronous receiver-transmitter (UART) lines;
- A GBTx card over E-Link.

FMC is a standard developed by the American National Standards Institute (ANSI) [4] that defines an interface between an FPGA carrier board and pluggable mezzanine card. This interface is further divided into an HPC and a LPC connector, where LPC is a subset of HPC (providing 68 and 160 user

defined single-ended signals respectively, or 34 and 80 user defined differential signals respectively). The standard also defines several voltage rails (3.3 V, 12 V and an adjustable rail between 0 V to 3.3 V) to power an attached mezzanine.

The FMC DUT connector mentioned above is used to connect an additional DUT to the FPGA and the GBTx card is used to communicate with a backend for evaluation of the radiation test. The E-Link interface on the GBTx card supports 40, 20 or 10 bidirectional serial links, with individual link speeds of 80 Mbit/s, 160 Mbit/s and 320 Mbit/s respectively. The link assignments used depend on the operating mode. When operating in the 80 Mbit/s mode, all 40 links of the E-Link interface are used. Operating in 160 Mbit/s mode uses every second link (0, 2, 4, ..., 40), and in 320 Mbit/s mode every fourth link (0, 4, 8, ..., 40) is used. This results into a maximum transmission capacity of 3.2 Gbit/s. The SFP port can be used for high-speed data communication over optical or copper media. Finally, the UART interfaces are designed to be used during application debugging.

4.2 CRaTeBo Mezzanine Card

To evaluate the radiation sensitivity of the GateMate in the mixed field environment of CHARM, a custom mezzanine board is developed based on the PolarFire SoM mentioned above. The requirements for this mezzanine board are as follows:

- Use the same PCB dimensions, FPGA connector positions and mounting holes as the PolarFire SoM;
- The oscillator used must be from the same oscillator family as the ones used on the PolarFire SoM (IQD CPFS-39);
- Connect the following signals from the SAMTEC connectors to the GateMate, with decreasing importance:
  1. Clock signal, JTAG, Inter-Integrated Circuit (I^2C), UART, SMA, and SFP;
  2. FMC LPC signals;
  3. GBTx e-link signals;
  4. FMC HPC signals.

4.2.1 Input and Output Signals

In the end, the first and second signal requirements and all GBTx control signals described in the list above were fully satisfied. The GBTx E-Link signals are only partially connected, with a total of seven input lines, three output lines and three input/output lines (the e-link lines used are shown in Table 4.1).

Table 4.1: Used GBTx e-link signals lines. The specific lines were chosen so that the full bandwidth of the GBTx can be utilized.

E-links Signal Direction	Used Lines
Data in	0, 4, 8, 12, 16, 20, 24
Data in/out	0, 4, 8
Data out	16, 20, 24

For user input, LEDs 0 and 1 are connected, as well as GPIO switches 0 and 1 and both UART connections. The complete list of signals connected to the GateMate on the SoM can be found in the appendix in Table B.1.

4.2.2 Onboard Clock Generation

For the onboard clock generation the IQD LFSPXO025560 50 MHz oscillator from the IQD CPFS-39 family was selected. The specific value was chosen after discussions with Cologne Chip, since clock frequencies above 50 MHz allow the GateMate to operate in a low-jitter mode. Figure 4.4 shows the clock generation schematic.

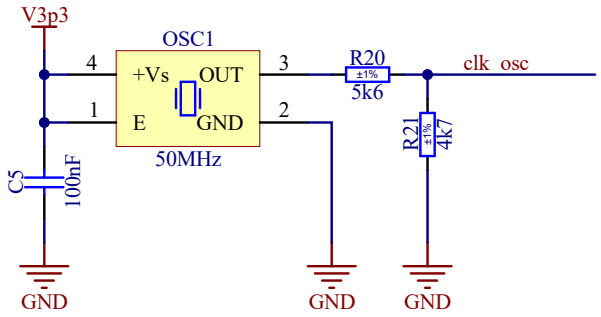


Figure 4.4: Schematic extract of the SoM clock generation. The voltage divider made of `R20` and `R21` drops the oscillator output voltage from 3.3 V to 1.51 V.

Since the rated supply voltage of this oscillator family is (3.30 ± 0.33) V[38], the 3.3 V supply rail of the FMC connector is used for the power supply. On the other hand, the clock input pin of the GateMate belongs to the SA I/O bank, which in turn is supplied with 1.5 V. To compensate for this difference, a voltage divider consisting of the resistors `R20` and `R21`, with the values of 5.6 kΩ and 4.7 kΩ respectively, is used. This reduces the maximum clock signal voltage from 3.3 V to 1.51 V, which is within the acceptable input voltage range of an I/O pin.

4.2.3 FPGA Configuration

To configure the FPGA on the SoM, the carrier's JTAG interface is used. Consequently, the configuration mode value is set to `1100`, which corresponds to configuration via JTAG. For the JTAG lines the resistor setup from the GateMate evaluation board is adapted. The `TCK` line is connected to ground with a 4.7 kΩ resistor, `TMS` and `TDI` are pulled up by pull-up resistors of the same value, and the `TDO` line has a 22 Ω series resistor. This configuration setup is shown in Figure 4.5.

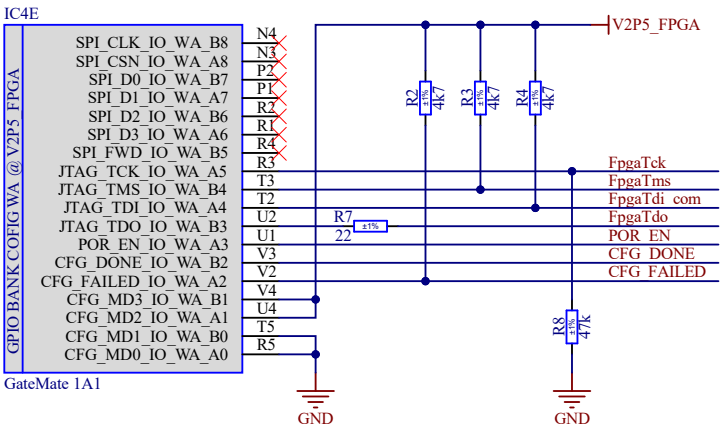


Figure 4.5: Configuration and JTAG lines of the SoM

To identify a failed or successful configuration, the signal lines CFG_DONE and CFG_FAILED are connected to LEDs as shown in Figure 4.6, with their corresponding current limiting resistors. Unlike to the permanent CFG_DONE signal, the CFG_FAILED signal must be latched because it is only momentarily active in the event of an error. For this purpose, a SN74LVC2G00 dual 2-input positive-NAND gate IC is used in a flip-flop configuration. The CFG_FAILED signal is used as the set input, and the SoM reset signal is used as the flip-flop reset.

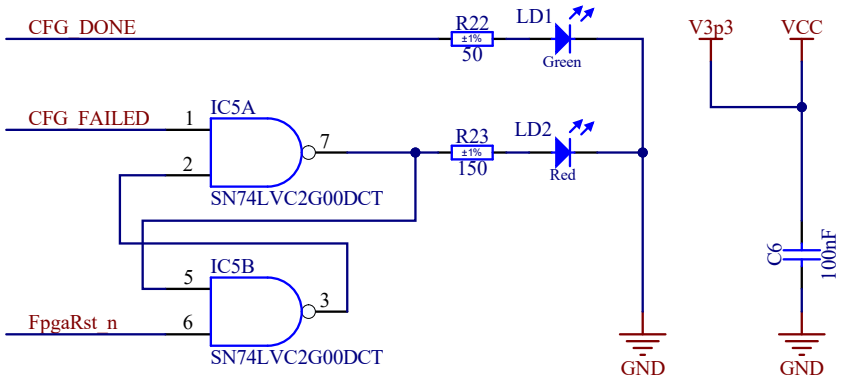


Figure 4.6: Configuration status LEDs with flip-flop latch

4.2.4 Reset and Power-on Reset

The reset and power-on reset (POR) connections are shown in Figure 4.8. Two reset options are available, POR and a manual reset via the PB1 push button.

The POR module is tasked with generating a safe reset signal on power-up when the V_{core} and V_{CLK} voltages are stable and above a defined threshold [16]. In addition, the POR module is also responsible for detecting voltage drops and triggering a reset. To enable the POR functionality, the POR_EN input has to be set to V_{WA} and the RST_N input to V_{CLK}.

When using the POR module, the threshold voltage adjustment is critical for proper operation. For V_{CLK} voltages between 1.8 V and 2.7 V, no additional adjustment is required. Otherwise, the resistors R3a and R3b, as well as capacitor C1, are used (Figure 4.7).

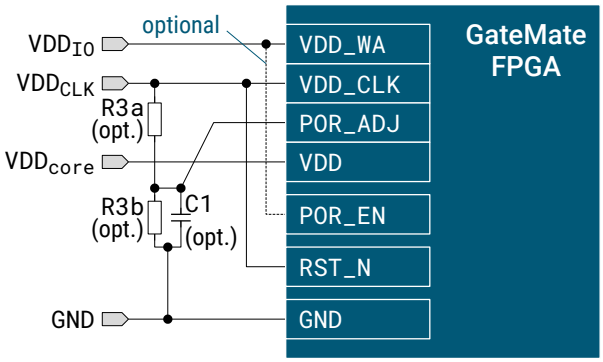
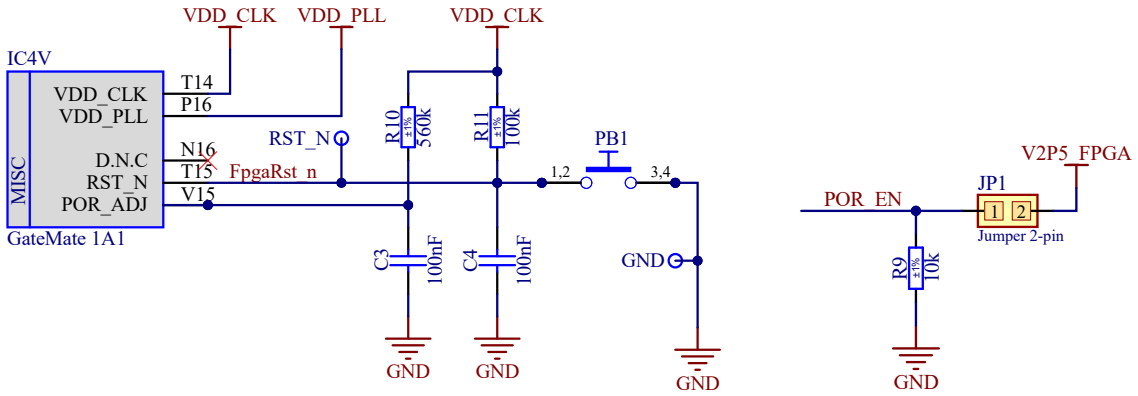


Figure 4.7: POR threshold adjustment with resistors. This figure is adopted from [16] without any changes.

As mentioned in [16], a resistor-capacitor (RC) circuit consisting of R3a and C1 must be used when $V_{\text{CLK}} < 1.8\,\text{V}$. The reset time for this case can be calculated using (4.1)

$$t_{\text{reset}} = \frac{R3a \cdot 75\,\text{k}\Omega}{R + 75\,\text{k}\Omega} C1 \ln\left(\frac{V_{\text{CLK}}}{V_{\text{CLK}} - \frac{0.45\,\text{V}}{75\,\text{k}\Omega}\left(75\,\text{k}\Omega + \frac{R3a \cdot 133\,\text{k}\Omega}{R3a + 133\,\text{k}\Omega}\right)}\right). \tag{4.1}$$

As shown in Figure 4.8a, this circuit is implemented using resistor `R10` and capacitor `C3`, with the values 560 kΩ and 100 nF, respectively. Applying the values in (4.1), the resulting reset time is $t_{\text{reset}} = 8.658\,\text{ms}$.

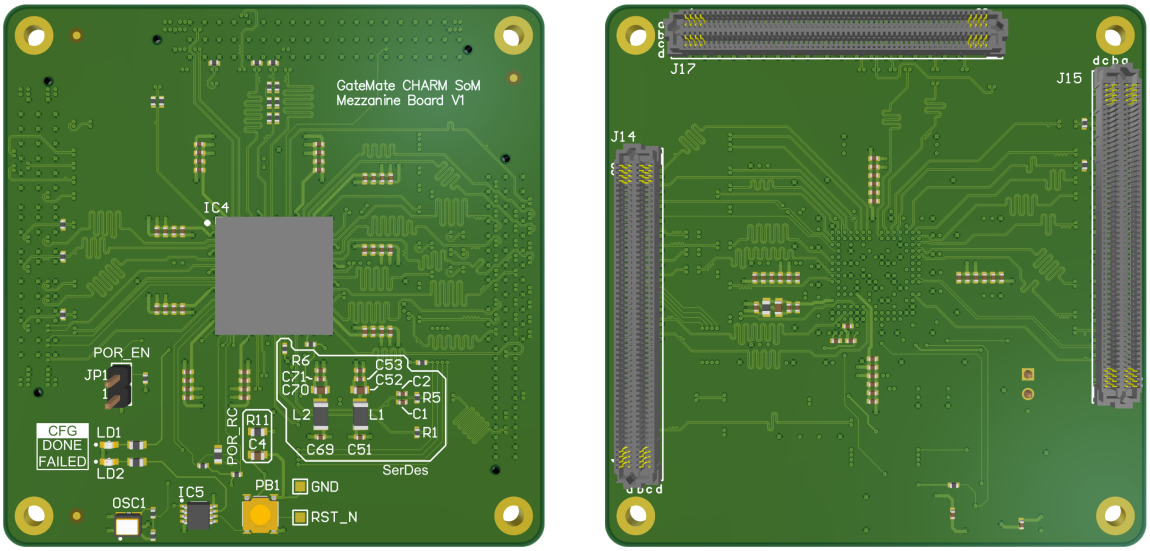


(a) Miscellaneous port connections **(b)** POR connection

Figure 4.8: Reset and POR circuitry

The POR module can be enabled or disabled with jumper `JP1`, shown in Figure 4.8b. If the jumper is not set, i.e., the module is disabled, the RC circuit composed of `R11` and `C4` provides a rising edge after power-up. The values here are initially set to 100 kΩ and 100 nF respectively, but can be adjusted depending on the desired power-up speed.

The final PCB design of the SoM board is shown in Figure 4.9, and a picture of the manufactured and fully assembled SoM mounted on the CRaTeBo is shown in Figure 4.10.



(a) Front side of the GateMate SoM **(b)** Back side of the GateMate SoM

Figure 4.9: 3D rendering the finished GateMate SoM

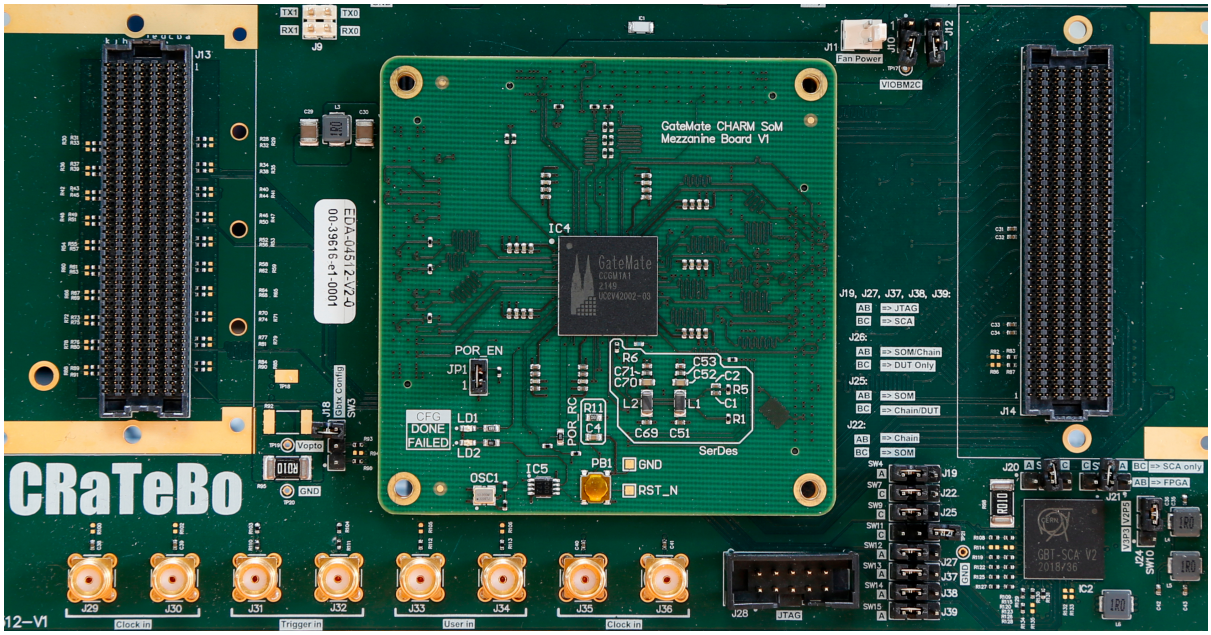


Figure 4.10: GateMate SoM mezzanine mounted on the CRaTeBo

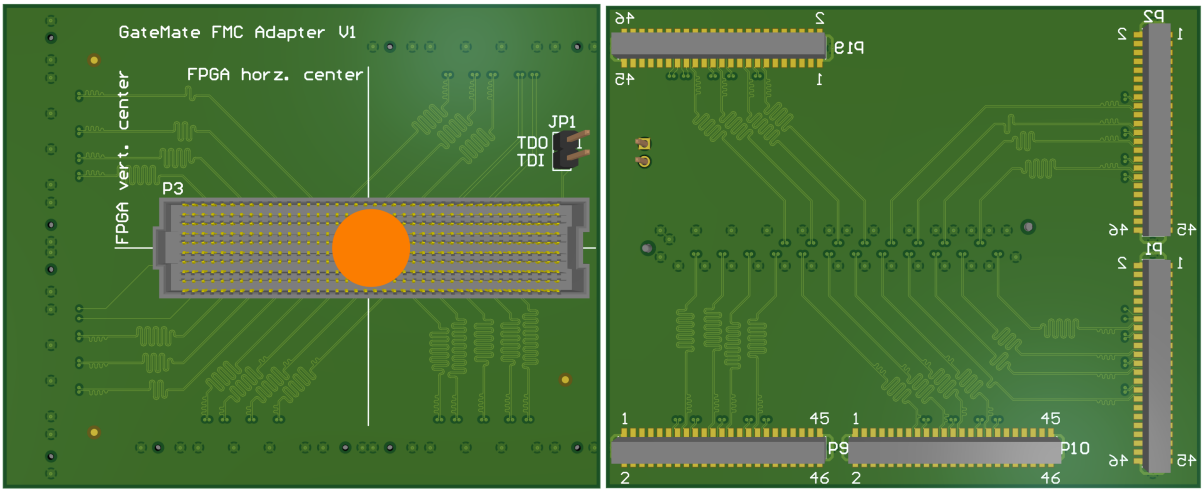
4.3 FMC Adapter PCB

During FPGA irradiation campaigns at PSI, the DUT board is placed in the path of the proton beam (more details can be found under Section 6.1). A laser is used to fine-tune the beam target position, so that only the target DUT is hit. When testing FPGAs, a second TESTER FPGA is connected via a FMC cable to the DUT, over which all communication takes place. As the GateMate evaluation board does not have an FMC connector, a custom FMC adapter PCB has to be created. The following requirements for this adapter are as follows:

- Connect the FMC LPC connectors to the I/O pins of the GateMate;
- Do not connect any power signals, as the evaluation board is self-powered;
- The JTAG loop between the TDI and TDO signals must be closed via a jumper.

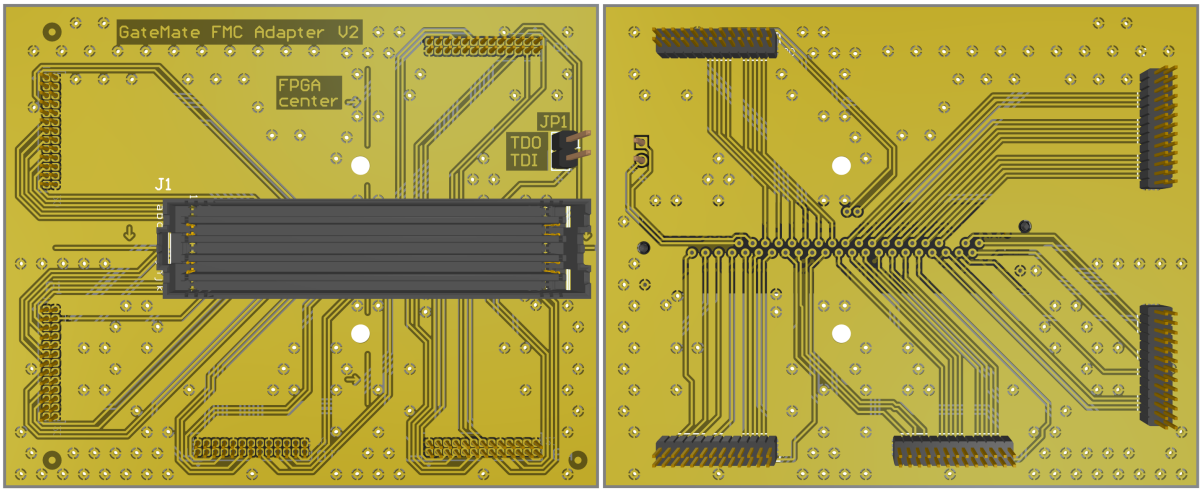
In the first iteration, a four layer PCB was designed, with all signals required by the FMC standard routed as differential pairs, with additional pair and group length matching. To help align the evaluation board at the Proton Irradiation Facility (PIF), a silkscreen alignment cross representing the center of the GateMate package has also been added to the PCB. The resulting board is shown in Figure 4.11.

Due to supply chain issues, where the 1.27 mm pitch pin headers were not available, a second version of the FMC adapter has also been created. In order to speed up the production process, it was decided to manufacture the PCB internally at CERN's Quick PCB service, which required a redesign of the existing board. Since the maximum number of layers that can be produced at CERN is two, the differential pairs are removed and routed as standard signals. This change is negligible, because the signals used in the current radiation tests are signals with frequencies $F_{\text{max}} \leq 10\,\text{MHz}$. In addition, the lack of silkscreen requires the alignment cross to be embedded in the top copper layer. As mentioned above, the headers are also changed from surface mount technology (SMT) to through hole technology (THT), as these types of connectors are available. The second version of the GateMate FMC adapter is shown in Figure 4.12.



(a) Front side of the first GateMate FMC adapter **(b)** Back side of the first GateMate FMC adapter

Figure 4.11: 3D rendering of the first GateMate FMC adapter PCB



(a) Front side of the second GateMate FMC adapter **(b)** Back side of the second GateMate FMC adapter

Figure 4.12: 3D rendering of the second GateMate FMC adapter PCB

5

FPGA Qualification Methodology

This chapter describes the system architecture for FPGA radiation testing at CERN. According to [10] the individual components of an FPGA should be tested with a design that follows the same concepts as an actual real-life design. For the GateMate A1, the following components and features are tested for their radiation tolerance:

- The 32 BRAM memory cells;
- The flip-flops located in the CPEs;
- The four PLLs;
- Real-world applications using proxy benchmark circuits;
- The GateMate's sensitivity to TID effects.

To reduce development time and avoid unnecessary costs, standard evaluation and development boards are used for the FPGA radiation tests whenever possible. Where appropriate, the TMR mitigation technique is used to harden critical parts of a design. The basic principle of TMR is the triplication of elements (functional elements or entire circuits) and to use a majority voter to select the correct output. This reduces the cross-section for SEEs induced errors, with the disadvantage of increasing the complexity of the design and the need for tighter timing.

5.1 Test Architecture

Testing FPGA designs requires stimulating the DUT and observing its response. One implementation of such a setup is to integrate the stimulus into the DUT itself, a so-called built-in self-test. While it is also possible to integrate the output observation into the DUT, it is more advantageous to capture the output data externally, e.g., via a host computer over a UART connection. However, this approach of using a built-in self-test has a major drawback in radiation testing: The added logic and complexity can fail under radiation, just like the components being tested. One could use mitigation techniques such as TMR, but these methods only reduce the possibility of an SEE induced failures. In addition, adding more logic to a design reduces the amount of logic elements available for testing.

The test setup currently in use at CERN was developed in [59] and uses a different method to interface with the components to be tested. Instead of relying on the DUT to perform the tests, a standalone test architecture was designed around the following specifications:

- Provide the DUT with the stimulus required for the components being tested and monitor the system response;
- High-speed DUT output monitoring to capture any errors generated by the DUT;
- Record and store the acquired data for later analysis and retrieval of device sensitivity;
- Ensure visibility of as many DUT signals as possible;
- Compatibility with FPGA development kits from multiple vendors;
- Modification of features and test routines shall require little to no effort;
- Facilitate the qualification process with real-time monitoring during irradiation.

The final architecture is based on an external system-on-chip (SoC), containing both an embedded CPU and an FPGA, called the TESTER. This setup allows the DUT pins to be connected directly to the the TESTER's embedded FPGA without any signal conversion. In addition, this configuration also allows high-speed testing as required by the specifications, since no protocol conversion overhead is required to transfer data between the TESTER and the DUT. This which would otherwise be the case when using a protocol such as UART. To control the test procedure and retrieve the observed data from the DUT, the embedded CPU can be used with high-level application programming interfaces (APIs) to control intellectual property (IP) cores placed in the FPGA region of the SoC.

The TESTER development board selected for radiation testing of the GateMate FPGA is the Xilinx Zynq ZCU104 Evaluation Board [70] (Figure 5.1), which is based on the Zynq UltraScale+ MPSoC XCZU7EV [71] SoC. This IC integrates a 64-bit quad-core Arm Cortex-A53 CPU and a dual-core Arm Cortex-R5F CPU with a Xilinx UltraScale+ FPGA on a single chip. While the ZCU104 evaluation board has an onboard FMC connector, the adapter PCB developed in Chapter 4 is used for the GateMate evaluation board. Both evaluation boards are connected with an FMC-to-FMC cable.

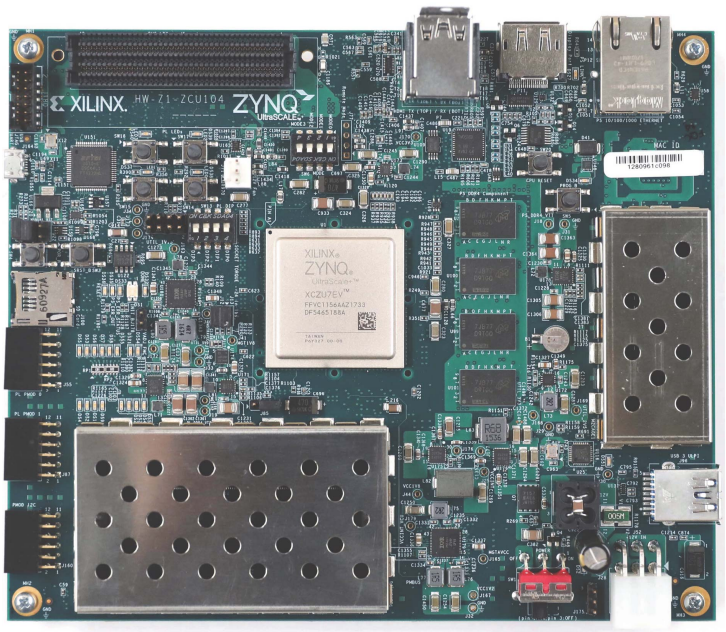


Figure 5.1: Xilinx Zynq ZCU104 Evaluation Board. This figure is adopted from [73] without any changes.

As briefly described above, the various interfaces to the DUT are implemented in the FPGA portion (referred to as programmable logic (PL)) of the SoC. These interfaces are specially designed IP cores, which are aligned to the particular test being performed.. The IP cores delivers stimuli to the DUT and

monitor and record its response under irradiation. The cores used during testing are either derived from [59], or are designed from scratch specifically for the GateMate DUT. Regardless of their origin, all of the IP cores used share the same interface type to communicate with the CPU (referred to as Processing System (PS)). This interface consists of *control*, *data* and *status* registers. By writing operation codes (opcodes) into the control register, the IP core's current operational mode can be changed, e.g., to start the test procedure. Querying the status register returns information about the current state of the IP core, and extracting the data collected during the testing phase is done by reading the data registers. These registers are memory-mapped in the Zynq SoC design, allowing multiple IP cores to be used simultaneously by simply allocating an address space for each core. The SoC's PS is implemented in the Zynq UltraScale+ MPSoC IP [72] (Figure 5.2) provided by Xilinx.

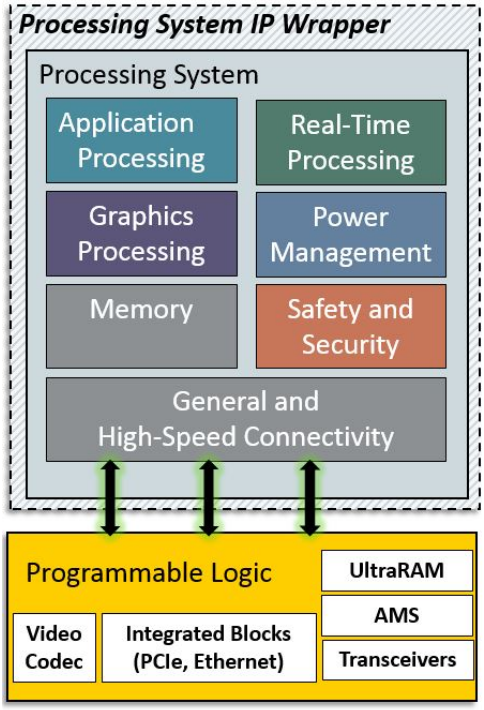


Figure 5.2: Xilinx UltraScale+ Processing System IP wrapper architecture. This figure is adopted from [72] without any changes.

The connection between the PL and the PS uses the Advanced eXtensible Interface (AXI)4 protocol, a common communication standard in ICs. It is specified by the Advanced Microcontroller Bus Architecture (AMBA) standard [6], and provides three different types of interfaces:

1. AXI4: An interface used for high-performance communication with memory-mapped devices;
2. AXI4-Lite: A simplified version of AXI4. Suitable for simple register-style interfaces, for example control and status registers;
3. AXI4-Stream: A one-way interface for high-speed data streaming.

By implementing this industry-standard protocol as the interface to the custom IP cores, turn-key Xilinx IP blocks can be used to build the SoC design. This also enables cross-vendor compatibility in case the TESTER platform changes in subsequent test campaigns. Furthermore, the tool used to develop the IP cores (Vivado by Xilinx[1]) has an IP creation wizard. This wizard creates boilerplate HDL files which handle the AXI4-Lite transactions, so the only task remaining for the developer is to connect the IP core's registers to the boilerplate code. An example of a design implemented in the TESTER FPGA, using both the PS IP and a test specific IP core, is shown in Figure 5.3

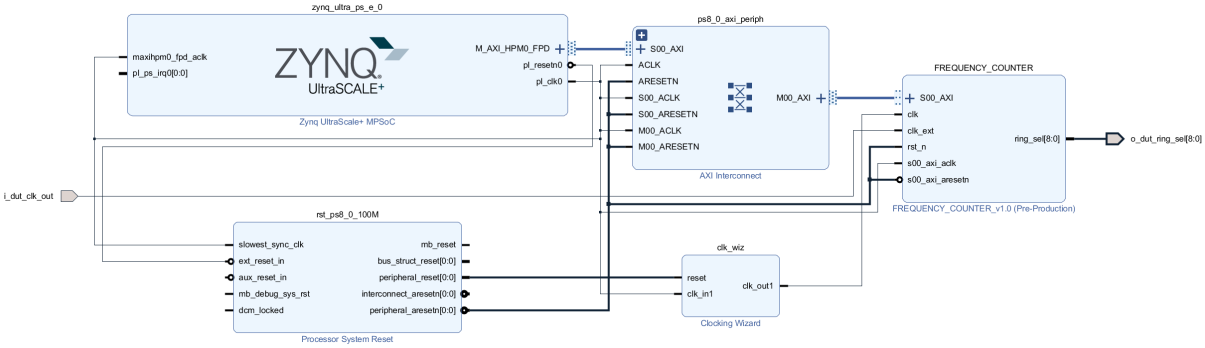


Figure 5.3: Example of a design used in the TESTER FPGA for the GateMate radiation tests. The left block with the Zynq text is the above mentioned Zynq UltraScale+ MPSoC IP block. The rightmost block is the test specific IP core, and the remaining blocks are required for proper communication within the SoC.

Applications that interact with the IP cores over the PS IP are developed using one of two software stacks, the *bare-metal software stack* and the *Linux software stack* [74]. Bare-metal applications are developed using the C and C++ programming languages in an environment called the standalone board support package (BSP), which provides standard inputs and outputs, and access to processor hardware features. These applications are similar to typical microcontroller programs, and peripherals such as IP cores are used in the same way as other microcontroller hardware features, for example timers or UART interfaces.

The Linux software stack on the other hand uses an embedded Linux operating system (such as PetaLinux[2] or the Yocto Project[3]) to run applications written in various high-level programming languages. Using the *Xilinx Runtime Library (XRT)*[4] driver components, the applications can access the peripherals located in the PL of the SoC. Although this approach adds additional overhead by using an operating system, software frameworks like Python Productivity for Zynq (PYNQ)[5] can be used to increase productivity and to simplify the usage of the overall system. PYNQ uses hardware libraries to interface with the PL circuits via Python, increasing the application development productivity. Furthermore, this framework includes the Jupyter[6] web server. This provides web-based notebooks, in which Python applications can be programmed through the browser. The final layout of the GateMate radiation test architecture is shown in Figure 5.4.

[1]https://www.xilinx.com/products/design-tools/vivado.html
[2]https://www.xilinx.com/products/design-tools/embedded-software/petalinux-sdk.html
[3]https://www.yoctoproject.org/
[4]https://github.com/Xilinx/XRT
[5]https://github.com/Xilinx/PYNQ
[6]https://jupyter.org/

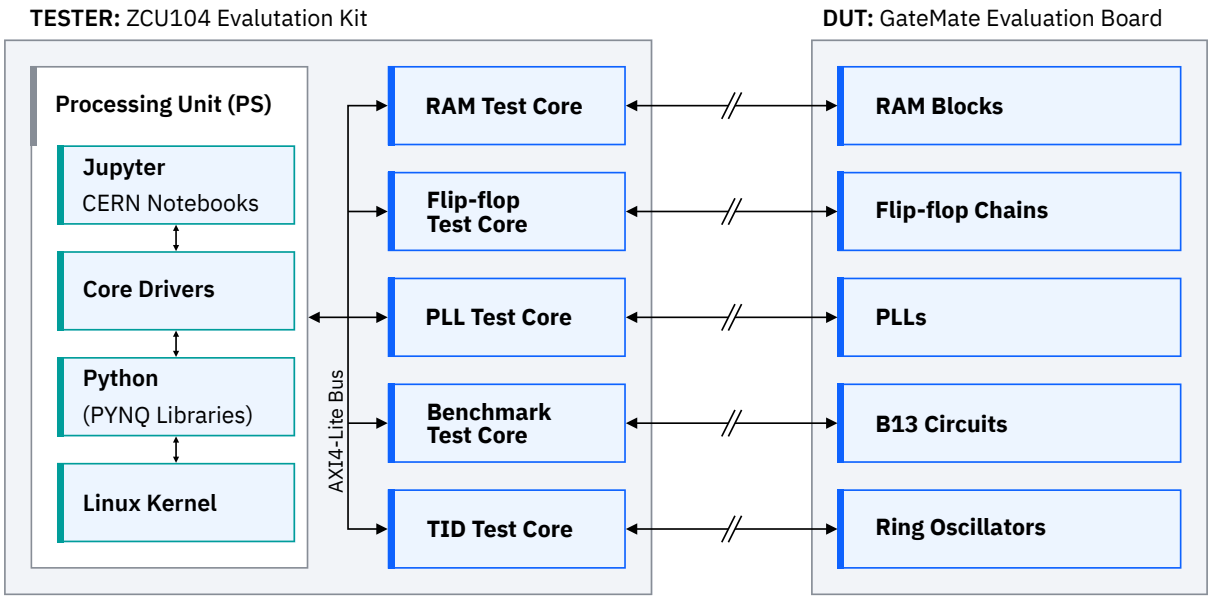


Figure 5.4: Overview of the GateMate radiation testing architecture

5.2 BRAM Cells

To test the BRAM cells of the GateMate, a pre-selected test pattern (all zeros, all ones or a checkerboard pattern, i.e., `101010...`) is written into each available address of each memory cell without radiation. Then the irradiation process is started and all addresses of the cells are read. The read data is compared to the initially selected test pattern to check for bitflips, indicating an SEU or MEU. If at any point a design error is detected at any point, the radiation is stopped and the test restarted. The general test setup is shown in Figure 5.5.

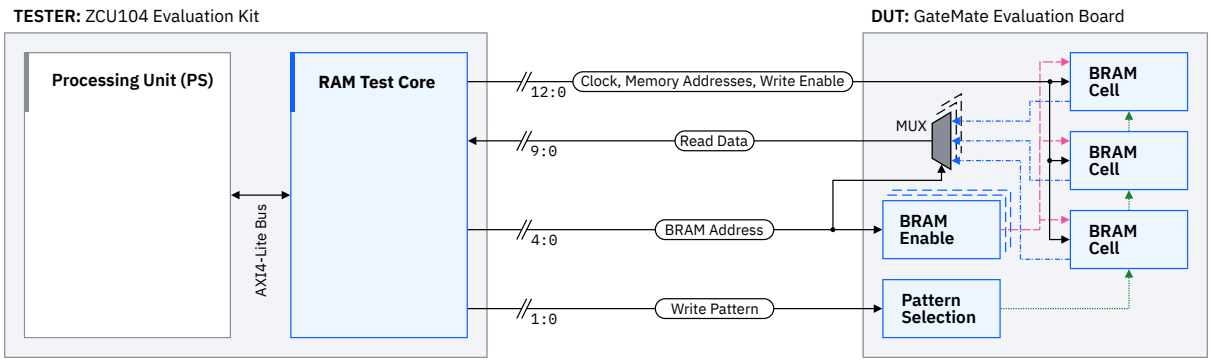


Figure 5.5: System architecture for the BRAM test

The GateMate FPGA does not have enough I/O pins to read and write the data lines of each memory cell. A multiplexer is therefore used to toggle the enable input of the selected BRAM cell. In addition, to eliminate timing issues when writing to the BRAM cells, the write data is generated on the GateMate. The data is only indirectly provided by the tester FPGA through the pattern selection signals. All input signals, except for the clock signal, are captured using TMR'ed clocked flip-flops. This ensures that all signals are processed simultaneously in the FPGA. The BRAM enable signal and the pattern selection signal are used to select the correct BRAM cell and data pattern. Additionally, the BRAM select signal is also used together with a multiplexer to output the read data from the selected cell to the data output lines.

IP Core

The IP core developed in [59] was specifically designed for the Microsemi PolarFire FPGA, hence a custom core has to be created. All of the test logic is contained in the IP, so the requirements are as follows:

- Provide a clock signal to the DUT;
- Perform read and write operations on the BRAM;
- Compare the read data with the selected pattern and count the number of bitflips.

Table 5.1 shows the interface of the developed IP core.

Table 5.1: Port description of the BRAM test IP core

Signal Name	Usage	Description
AXI4-Lite Bus	Internal	Interface to interact with the CPU
`clk`	Internal	Internal clock signal
`dut_clk`	External	Clock signal for the DUT
`dut_wen`	External	Write-enable signal
`dut_sel[n-1:0]`	External	BRAM cell selection
`dut_pattern_sel[1:0]`	External	Pattern selector to write into memory
`dut_addr[k-1:0]`	External	BRAM memory address selection
`dut_rd_data[d-1:0]`	External	BRAM cell output data

The core exposes six registers (Table D.1) to interface with the IP. The control register is used for writing opcodes (Table D.2) that control the operation of the core. The core automatically iterates through all available memory addresses of a cell. The address of the BRAM cell to be accessed is set through the address register, and the test pattern is set using the pattern register and the pattern selection codes (Table D.2). After reading the memory contents, the number of SEUs and MBUs can be accessed by reading the SEU and MBU registers respectively. Finally, the current state of the core can be obtained by reading the status register (Table D.3).

The working principle of the IP core is as follows:

1. The core is in its idle state;
2. The memory cell to be read and written to is selected by writing the address into the address register;
3. The pattern to write into the memory is selected by writing the pattern opcode into the pattern register;

4. The memory write process is started by writing the start opcode into the control register. This procedure starts at memory address `0x0` and writes the pattern to each valid address until the highest valid address is reached;
5. The core is in the wait state, which must be cleared via the clear operation;
6. Reading the memory is performed in the same way as the write command. After each memory address is read, the data is compared with the selected pattern, and the observed SEUs and MBUs are counted;
7. The core is again in the wait state, where the registers containing the SEU and MBU counts can be read;
8. The reset operation resets the counting registers.

5.3 Flip-Flops

The naive way to test the flip-flops contained in the CLBs is to use a long chain of sequentially connected flip-flops, i.e., a shift register (SR). However, as mentioned in [10], it has been shown that the use of window shift registers (WSRs) instead of regular SRs improves signal integrity and allows higher clock frequencies. A WSR consists of a chain of SRs, where the outputs of the last n flip-flops are connected to an output window (Figure 5.6).

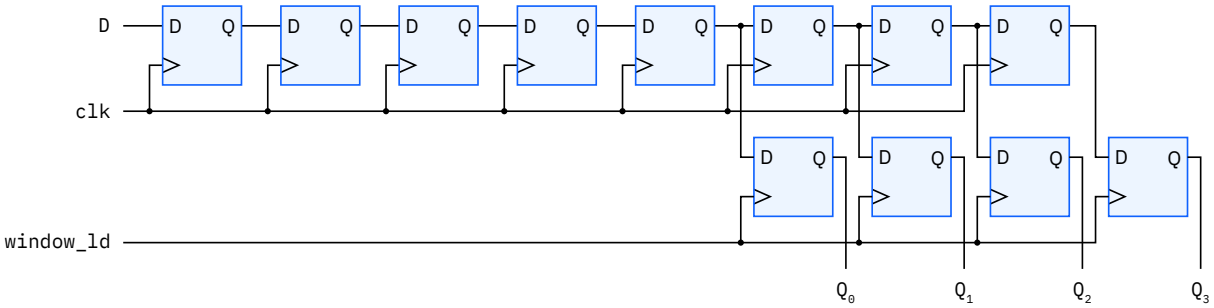


Figure 5.6: Simple WSR circuit. The `window_ld` signal is a secondary clock signal used to load the window flip-flops.

The input pattern to the WSR is either static `0`, static `1`, or alternating between `0` and `1` every clock cycle. While not relevant when using the static input variants, the selection of the window load signal frequency is important for the alternating input. When selecting the frequency of the window load signal using equation (5.1), the output of the window remains the same unless a bitflip has occurred.

$$F_{\text{window}} = \frac{F_{\text{clk}}}{n}, \tag{5.1}$$

This is demonstrated in Figure 5.7, based on the WSR shown in Figure 5.6. Initially the WSR is loaded with the selected pattern, alternating between `0` and `1`. At clock cycle 13, an SEU is captured and its propagation can be observed.

The main components of the DUT RTL design are the 12 WSRs. The input to the SR chain is generated by the pattern generator, based on the selected pattern. The same external signal is also used at the end of the WSR to compare the window outputs with the golden reference. All elements in the design are triplicated, except for the flip-flops in the SR, which are only optionally triplicated. The complete test architecture is shown in Figure 5.8.

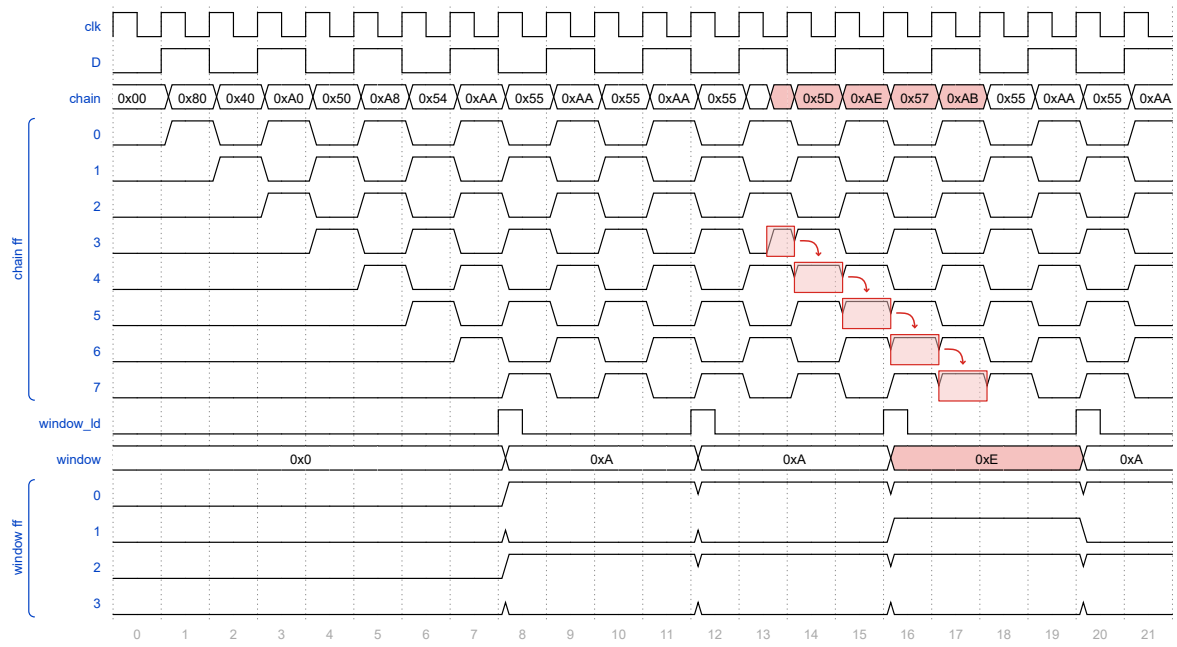


Figure 5.7: Waveform of the WSR in Figure 5.6 in operation. The signals marked in red show the behavior when an SEU is captured. The arrows show the propagation of the SEU error through the SR chain.

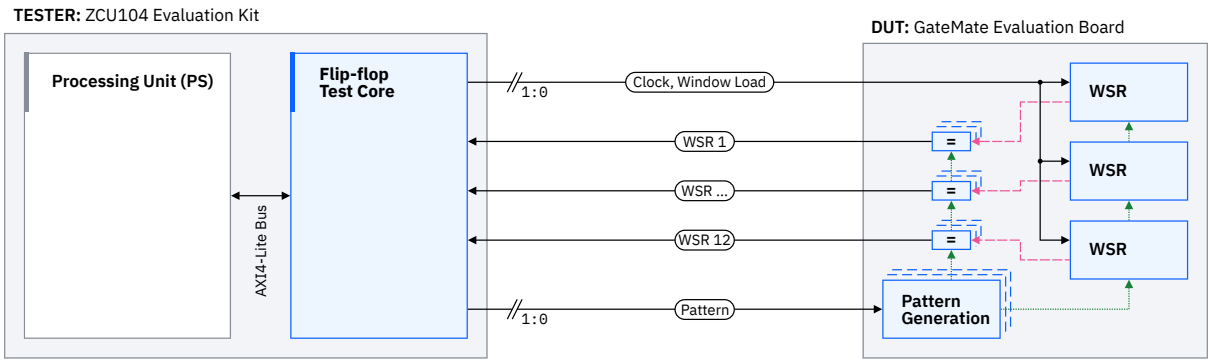


Figure 5.8: System architecture for the flip-flop test

IP Core

The IP core used for the flip-flop design was taken directly from [59]. This core provides the DUT with the required clock and window load signals, as well as the pattern selection signal. The output of the comparator at the end of each WSR is monitored to observe if an SEU has occurred. The interface of the flip-flop IP core is shown in Table 5.2.

Table 5.2: IP core for the flip-flop test

Signal Name	Usage	Description
AXI4-Lite Bus	Internal	Interface to interact with the CPU
`clk`	Internal	Clock signal
`rst_n`	Internal	IP reset signal
`window_ld`	External	Window load signal
`chain_output[n-1:0]`	External	WSR chain selection
`pattern_sel[1:0]`	External	WSR input pattern selector

The core exposes 19 registers (Table D.4) to interface with the IP. The control register is used to control the operation of the core, as well as which pattern to use for testing. Both settings are written into the register according to Table D.5. An additional length register is used to set the number of flip-flops per chain. Finally, the current state of the core can be acquired by obtained the status register (Table D.8).

In the idle state, the test procedure can be started with the start opcode or the SEU registers can be reset with the reset opcode (Table D.7). Starting the test procedure locks in the currently selected pattern (Table D.6), the length of the flip-flop chains, and sets the core's finite state machine (FSM) to the test state. In addition, the comparison output signals of the DUT are captured with each clock cycle. Initially, in the test state, the core waits until all flip-flops in the DUT have been filled based on the value stored in the length register. After the waiting period, the following four steps are executed, one after the other with the rising edge of the clock signal:

1. The window load signal is set to `'1'`;
2. The window load signal is set to `'0'`;
3. Wait for one clock cycle;
4. Check the registered DUT outputs for any comparison failures and record the failure in the corresponding chain SEU register.

These steps are repeated until the stop opcode is received.

5.4 Phase-Locked Loop

PLL blocks are important elements in FPGA designs, as they provide the implemented design with stable clock signals. As such, SEUs destabilizing these elements can have a significant impact on the performance of the FPGA. Such PLL blocks have at least the following two outputs:

- Clock: This is the main output of the PLL, a clock signal with the desired characteristics set by the user;
- Lock: This signal indicates whether the clock signal is stable and ready for use.

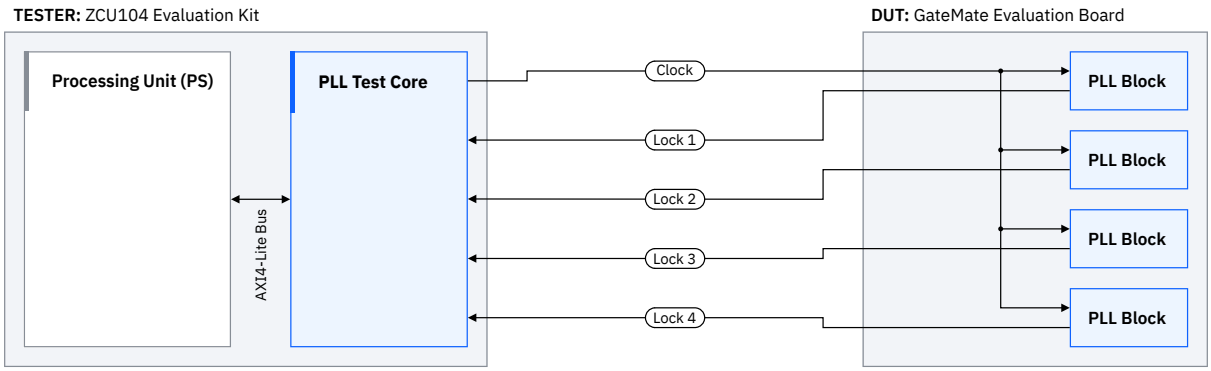


Figure 5.9: System architecture for the PLL test

Consequently, testing the PLL blocks consists of monitoring all PLL lock signals for changes from their default state. The test architecture is illustrated in Figure 5.9.

The RTL design is simple and consists only of four `CC_PLL` primitives with the same settings. The reference clock signal is provided by the TESTER FPGA, and the four lock signals are output via GPIO pins.

IP Core

As with the previous cores, the IP core for the GateMate PLL test was also taken from [59]. This core has only a minimal set of external ports (Table 5.3), as its main functionality is to provide timers for frequency measurement.

Table 5.3: IP core for the PLL test

Signal Name	Usage	Description
AXI4-Lite Bus	Internal	Interface to interact with the CPU
`clk_i`	Internal	Clock signal
`rst_ni`	Internal	IP reset signal
`ref_clk_o`	External	Input for the PLL reference clock
`pll_lock_i[n-1:0]`	External	Lock signals from the PLLs

The core exposes up to 18 registers (Table D.17) to interface with the IP. The control register is used to control the basic operation of the core, while the current status of the core can be obtained by reading the status register (Table D.11). The remaining 16 registers are used to store the number of observed SEUs for each PLL, with a maximum number of 16 PLLs.

From the idle state, the test procedure is started using the start opcode (Table D.10). Here the core continuously monitors the lock signals for each PLL and counts each de-assertion. The test sequence can be stopped with the stop command, and the SEU registers are reset with the reset command.

5.5 Benchmark Evaluation

The radiation sensitivity of an FPGA's individual components can be determined by performing the radiation tests described above. However, these obtained metrics do not provide a true representation on how the FPGA will perform under realistic workloads. For example, if an SEU occurs in a BRAM cell, but the value stored at this address is rewritten before being read, there is no error from the applications

point of view. As a consequence, estimating the device's failure rate is a difficult, if not impossible, task. Therefore in [10] it is proposed to complement the component-level tests with additional application-level tests. Since the use of different projects for each FPGA to be tested and used at CERN is not feasible, [60] describes the use of a publicly available benchmark circuit to perform the application-level test. This decision is supported by the fact that there is no standard implementation strategy for FPGAs at CERN It is therefore beneficial to test all FPGAs with the same benchmark.

The benchmark circuit used is taken from the ITC'99 suite, a collection of RTL circuits of different complexity, but uniform characteristics, developed by the Politecnico di Torino [19]. These circuits originate from public VHDL files and are modified to behave fully synchronously. They are also independent of vendor specific directives and are fully synthesizable. Furthermore, this suite was used in other FPGA reliability experiments [56, 46]. The circuit used for radiation testing is the B13 circuit, which was originally used as an interface to meteorological sensors. Although syntactically correct, the circuit is not functionally meaningful as its original function was stripped during the benchmark suite development process [19]. Hence it is considered as a black box based mainly on a FSM and consists of 339 gates and 53 flip-flops.

As mentioned earlier, one goal in developing the designs for the DUT is to maximize the number of instances tested. On the other hand, since a B13 circuit instance wrapper has two 10 bit wide ports, the number of circuits that can be placed is limited by the available I/O pins of the FPGA. In order to solve this conflict, while maintaining the level of circuit observability, a benchmark structure has to be developed (Figure 5.10).

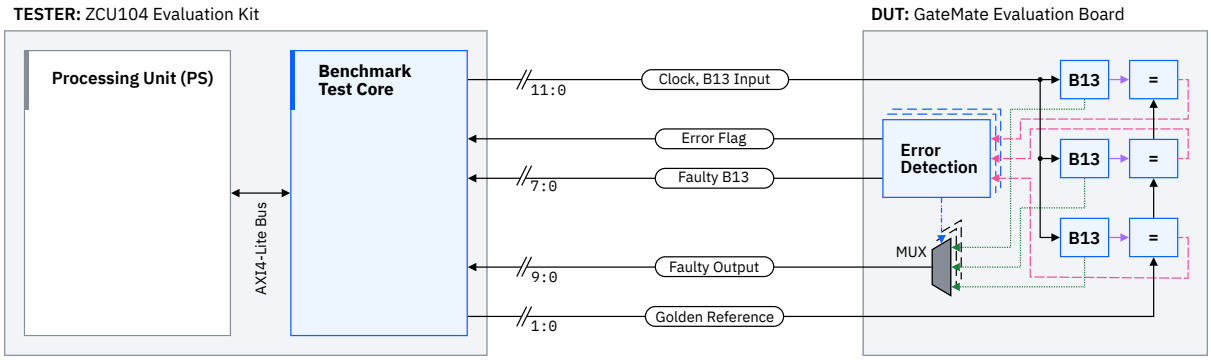


Figure 5.10: System architecture for the benchmark test

Here, all B13 instances are followed by a sequential comparator circuit. These n-bit comparators compare the outputs of their respective B13 circuit to a golden reference, which is provided externally by the TESTER FPGA. Under normal circumstances, the output of this comparator is `1`. If an SEU occurs where the B13 output is not equal to the golden reference, the output of this comparator is set to zero. To further reduce the number of required I/Os, all comparator outputs are combined into a single error bit using a `NAND` operator. This error bit is used as a flag to indicate to the TESTER FPGA that an SEU has been observed. If the error bit is set, the address of the faulty B13 and its current output are passed to the TESTER FPGA. To minimize non-B13 related errors, all additional circuitry is TMR'ed.

IP Core

As with the previous cores, the IP core (Table 5.4) for the GateMate TID test was also taken from [59]. Internally, a linear feedback shift register (LSFR) generates the input data for the DUT and golden B13 reference circuits. It also includes a FIFO to store the observed data, i.e., the erroneous B13 address and

output, if an SEU is detected. All B13 circuits can also be reset via a global reset line from the IP core, which is active-high.

Table 5.4: IP core for the benchmark test

Signal Name	Usage	Description
AXI4-Lite Bus	Internal	Interface to interact with the CPU
`clk`	Internal	Clock signal
`rst_n`	Internal	IP reset signal
`b13_golden_o[9:0]`	External	The golden output of the B13 circuit
`b13_input_o[9:0]`	External	The input to feed to all the B13 circuits
`b13_rst_o`	External	Reset the B13 circuits on the FPGA under radiation
`error_detected_i`	External	Active high, indicates an error on the FPGA under radiation
`b13_address_i[n-1:0]`	External	Address of the failing B13 circuit
`b13_output_i[9:0]`	External	The output produced by the failing B13 circuit

The core exposes seven registers (Table D.13) to interface with the IP. The control register is used to control the basic operation of the core. The SEU registers contain the number of SEUs counted during the test period. The error value and golden value registers contain the data stored in the FIFO for each SEU. The values in these two registers are composed as shown in Table D.14 and Table D.15. The current status of the core can be obtained by reading the IP core status register (Table D.19), and the FIFO status is stored in the FIFO register. Finally, the number of cycles to wait before processing the DUT error signal are set using the wait cycles register.

At idle, the LSFR is disabled, and the reset signal for all B13 circuits is set to `1'b1` (active-high). The test procedure is initiated by writing the start opcode (Table D.12) into the control register. This sets the reset line to `1'b0`, enables the LSFR and starts the error observation process. If the `error_detected_i` signal of the DUT is set to `1'b1`, the LSFR is disabled, the B13 circuits are reset and the SEU counter is incremented. In addition, the output and address of the failed B13 (Table D.14), and the current LSFR and the golden reference output (Table D.15) are stored in the FIFO. A state transition is also performed to `RESET_FPGA_UUT`, which holds the B13 reset signal at `1'b1` until the FIFO write process is complete. Following this, the IP returns to the `TEST` state to repeat the test procedure until stopped. The `TEST` state can be exited by writing the stop opcode to the control register, and resetting the SEU counter is done via the reset opcode. Reading the erroneous values from the FIFO is performed via a second FSM using the read opcode.

5.6 TID Design

As described in Section 2.2.1, cumulative radiation effects cause gradual degradation of device parameters, such as shifting the threshold voltage of a MOSFET. This can increase the leakage current of the FPGA, and change its internal signal propagation delay [53]. Testing for FPGA TID effects is typically done different from the SEE test methodology described in the previous sections. One possible way is to monitor the power consumption of the FPGA externally during irradiation. However, this method relies on the DUT being exposed to a dose near its failure threshold in order to observe a trend. Observing internal defects such as the propagation delay increase, which can violate existing timing constraints, is a second method to qualify the cumulative effects of radiation on FPGAs. This was demonstrated in [45], where the propagation delay degradation of an FPGA reached up to $1100\,\%$ before failure.

A circuit that can be used to dynamically measure the propagation delay is the ring oscillator [36, 61]. This circuit generates an output signal whose frequency is directly related to the propagation delay

of the individual elements that make up the circuit, i.e., the routing connections and the CLBs. Thus the change in propagation delay during irradiation can be obtained by observing the change in the frequency of the ring oscillator. The overall system architecture for this test is shown in Figure 5.11.

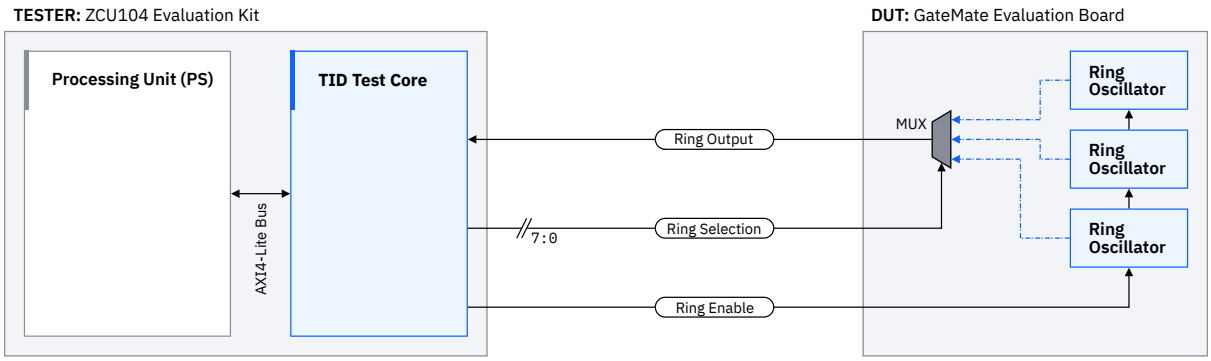


Figure 5.11: System architecture for the TID test

In [59], the ring oscillator was implemented by using a chain of inverters. On the GateMate FPGA this is not possible, because inverters are optimized away by the P&R tool. Instead, `CC_DLT` D-latches are concatenated (Figure 5.12), with the `SR` input of each latch hardwired to `1'b0`, and the `G` input of each latch tied to the `i_ring_en` enable signal. To generate the periodic signal, the output of the last latch is inverted and fed into the input of the chain's first latch. Thus, regardless of the number of latches in the chain, an alternating signal is always generated. The resulting output frequency F_{ring} with n latches in the chain, each having a propagation delay of $t_{\text{PD, CPE}}$, can be calculated using (5.2).

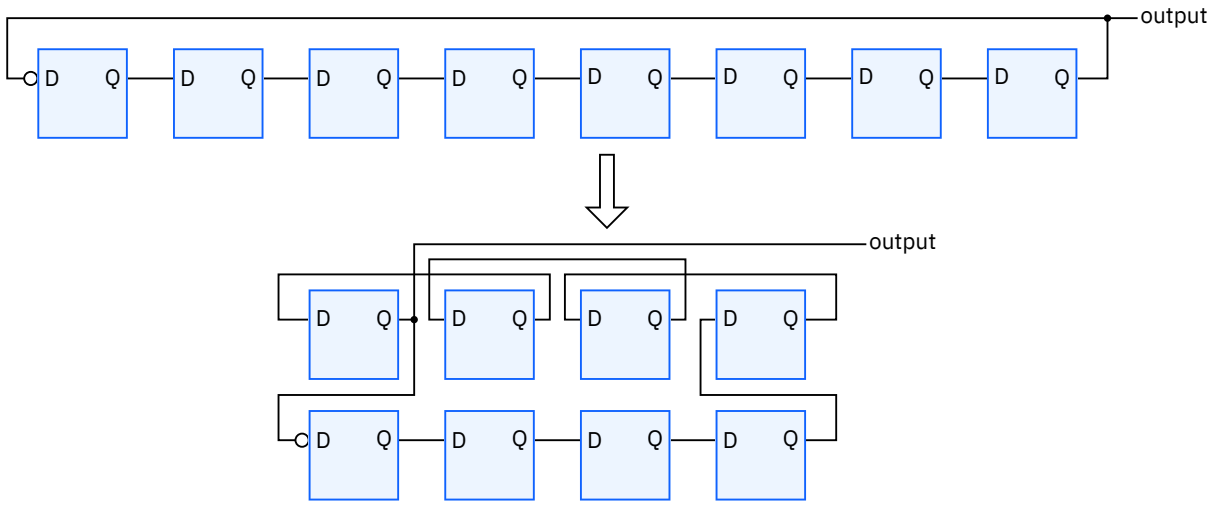


Figure 5.12: Ring oscillator built from a chain of `CC_DLT` latches. The upper ring shows a simple version of the oscillator chain, while the lower chain shows a more compact version of the same oscillator. This layout is used for the FPGA implementation to increase the area usage efficiency.

$$F_{\text{ring}} = \frac{1}{2\, n\, t_{\text{PD, CPE}}} \tag{5.2}$$

To collect meaningful statistics, it is advantageous to fill up as much of the GateMate's fabric as possible with these ring oscillator circuits. This also allows to observe the propagation delay spread across the FPGA. In [16] it is documented that the GateMate consists of a 160×128 CPE array, which is subdivided by the BRAM cells into 5 parts, each 32×128 in size (Figure 3.6). Therefore, the number

of elements per chain can be set to a maximum of $32 \cdot 2 = 64$, so that the chain is not interrupted by the BRAM cells. In order not to be limited by the number of outputs available for measurement, a multiplexer is finally used to output the signal of the selected oscillator.

One issue in implementing the test circuit is the placement method of the P&R tool. In order to achieve comparable frequencies between the oscillator chains, the individual elements in the chains have to be placed manually. Otherwise, the P&R tool might place two consecutive elements in different areas of the FPGA, resulting in longer and incomparable routing delays between the oscillator chains. As described in Section 3.4.2, the coordinates of the CPEs in the design can be manually defined in the constraints file. By using a custom script, which takes the output of an initial, positionally unconstrained implementation process, the required positions for the CPEs can be generated.

The first part of this script (Listing 5.1) reads the *.place* and the *.refwire* files into memory, whilst some initial filtering is performed. From the *.place* input, all non-CPE related lines are removed. From the *.refwire* input, only the lines containing either `Q 1 and inverter_chain` or `Q 1 rings_out` are kept. These two comparisons are necessary, because otherwise the last D-latch in the chain would be ignored.

Python
```python
# Get list of CPEs with coordinates
# Sample line for one CPE:
# CPE 228 PosX 56 PosY 25;
with open(build_folder / f'{top_name}_00.place', 'r') as f:
    cpe_list = [CPE(*(int(x) for x in re.findall(r'\d+', l))) for l in filter(lambda l: 'CPE' in l, f.readlines())]

# Get order of CPEs
# Sample lines for one CPE:
# 228 I        D 1 \ring_osc_gen[0].inverter_chain [14]
# 228 I        G 2 _000_
# 228 I       SR 3 1'h0
# 228 O        Q 1 \ring_osc_gen[0].inverter_chain [15]
with open(build_folder / f'{top_name}.refwire', 'r') as f:
    refs = [l.strip() for l in filter(lambda l: ('Q 1' in l and 'inverter_chain' in l) or ('Q 1 rings_out' in l), f.readlines())]
```

Listing 5.1: Reading the contents of the P&R output and initial filtering

The results of reading the files and initial filtering are two lists. The first list, `cpe_list`, contains all CPEs used for the ring oscillators. Each CPE is stored as a `CPE` Data Class [55] (Listing 5.2), initially containing the number of the CPE and its x and y coordinates. The other members of the `CPE` Data Class are assigned later in the script, and have the following meaning:

- `order`: Represents the order of the CPE in the ring oscillator chain;
- `chain`: Used to assign the CPE to a specific ring oscillator chain;
- `ref`: Reference string used in the *.refwire* file to assign the CPE to a primitive in the netlist.

Python
```python
@dataclass
class CPE:
    number: int
    x: int
    y: int
    order: int = 0
    chain: int = 0
    ref: str = ""
```

Listing 5.2: Python Data Class used to store the information about each CPE in the design

The second list, `refs`, contains the netlist references extracted from the *.refwire* file as strings. Taking the sample lines from Listing 5.1 for this file (lines 10-13), such a string would be `\ring_osc_gen[0].inverter_chain [15]`. With the reference list available. the missing information for the `CPE` objects can be filled in (Listing 5.3). Afterwards, all CPEs without a reference, i.e. the CPEs that are not explicitly used in the oscillator chains, are removed from `cpe_list`.

Python

```
# Assign data to CPEs in cpe_list
current_chain = None
for r in refs:
    numbers = re.findall(r'\d+', r)
    for c in cpe_list:
        if c.number == int(numbers[0]):
            if 'rings_out' in r:
                c.order = latches_per_chain-1
                c.chain = current_chain
                current_chain = None
            else:
                c.order = int(numbers[-1])
                c.chain = int(numbers[-2])
                if current_chain is None:
                    current_chain = c.chain
            c.ref = ''.join(re.sub(r' {2,}', ' ', r).split(' ')[4: ])

cpe_list = [c for c in cpe_list if c.ref != '']
```

Listing 5.3: Adding the netlist references to the `CPE` objects

This list of `CPE` objects can now be used to create a Python representation of the ring oscillator chains (Listing 5.4). This representation is a list of lists, where each inner list corresponds to one chain. The value `number_of_chains` has to be set manually, and is not inferred from the *.refwire* data. Afterwards, the `CPE` objects in each inner list are sorted according to their `order` value and saved in a new list called `sorted_chains`.

Python

```
chains = [[] for i in range(number_of_chains)]
for cpe in cpe_list:
    chains[cpe.chain].append(cpe)

# Sort list of CPEs
sorted_chains = [[] for i in range(number_of_chains)]
for i, ch in enumerate(chains):
    sorted_chains[i] = sorted(ch, key=lambda c: c.order)
```

Listing 5.4: Sorting the `CPE` objects and creating the required CPE order

With these sorted chains, it is now possible to arrange the individual CPE coordinates according to the order of their representative `CPE` objects (Listing 5.5). The constants `START_X_COORD`, `START_Y_COORD`, `MAX_X_COORD` and `MAX_Y_COORD` represent the boundaries of the GateMate routing structure, i.e., 0, 0, 160 and 128 respectively. Furthermore, the `latches_per_chain` variable is initially set to 64, to allow the row between two BRAMs to be fully populated.

Python

```
# Assign new location for CPEs
x = START_X_COORD
y = START_Y_COORD + 1  # Keep one CPE row distance between chip edge
latches_per_row = latches_per_chain // 2
for i in range(len(sorted_chains)):  # Go through each chain in sorted_chains
    for j, c in enumerate(sorted_chains[i]):  # Place each CPE in the chain
        # Check if processing upper or lower row of the oscillator
        if j < latches_per_row:  # Lower oscillator row
            c.x = x
            c.y = y
            if j == latches_per_row - 1: # Check if last row element is reached
                y += 2  # Switch to upper oscillator row
            else:
```

```
14                  x += 1
15          else:  # Upper oscillator row
16              c.x = x
17              c.y = y
18              x -= 1
19
20      x += latches_per_row + 1  # Go to next oscillator in x direction
21      if x < MAX_X_COORD:
22          y -= 2  # Reset y position to lower oscillator row
23      else:
24          x = START_X_COORD
25          y += 3  # Move to next row, but keep one CPE row distance
```

Listing 5.5: Assigning the coordinates for each CPE

After defining the required CPE coordinates, the new position constraints are written to the constraints file by the script. The final step is to invoke the implementation process again. Upon successful execution, the *.place* file can be inspected. Every CPE with a comment `# preplaced by CCF file` is preplaced.

IP Core

As with the previous cores, the IP core for the GateMate TID test was also taken from [59]. This core has only a minimal set of external ports (Table 5.5), as its main functionality is to provide timers for frequency measurement. Two internal frequency counters are used in the IP core, each with a different source signal. The first counter is used as a reference, and counts the number of pulses from an internal clock source. The second counter counts the pulses originating from an external signal, in this case from the selected ring oscillator. After a user-definable measurement period, both values are written to their specific registers and can be read by the CPU. The calculation of the ring oscillator frequency from the two values is described in Section 6.2.4.

Table 5.5: IP core for the TID test

Signal Name	Usage	Description
AXI4-Lite Bus	Internal	Interface to interact with the CPU
`clk`	Internal	Clock signal
`ext_clk`	External	External clock signal
`rst_n`	Internal	IP reset signal
`ring_sel[n-1:0]`	External	Address for the ring oscillator chain to observe

The core exposes seven registers (Table D.17) to interface with the IP. The control register is used to control the basic operation of the core. The reference and external pulse registers contain the number of pulses counted during the measurement period. This period can be configured using the wait period registers. As during testing is was discovered that some ring oscillator chains where not responding, the IP core is also augmented with an timeout function. The timeout period is set with the help of the timeout register. Writing to the ring address register selects a ring oscillator to be measured. Finally, the current state of the core can be obtained by reading the status register (Table D.19).

From the idle state, the test procedure is started with the start opcode. Here, the core remains in a waiting state until either an external pulse is registered, or a timeout is triggered. In case of a timeout, the state has to be cleared with the clear timeout command to return to the idle state. When an external signal is monitored, the counting process is performed until the end of the measurement period. Here, the core waits until the read state is cleared and finally returns to its idle state. Simply writing the waiting period and timeout values into their respective registers does not have an immediate effect. Instead, they have to be loaded using the load command.

6

FPGA Radiation Test Results

This chapter details the process used by CERN to evaluate the radiation tolerance of the GateMate FPGA. The first section describes the two irradiation facilities used during the evaluation, and the second describes the critical metrics observed. The final section presents the results of the evaluation.

6.1 Irradiation Testing Facilities

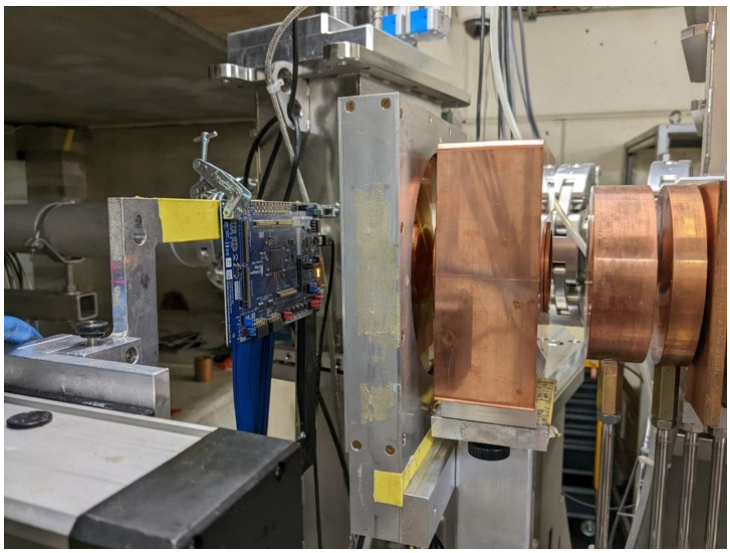
The main facility used for the qualification is the PIF at PSI. The PIF was built as a collaboration between PSI and the ESA and is the standard facility used by CERN to evaluate the radiation sensitivity of electronic components to protons. The cyclotron-type COmpact MEdical Therapy cyclotron (COMET) accelerator, which is mainly used for radiotherapy, supplies the irradiation area of the PIF with a 200 MeV proton beam. This energy of this beam can be reduced further to 10 MeV by using copper plates of varying thickness, ranging from 0.5 mm to 32 mm. To control the beam flux, the current can be adjusted between 0.1 nA and 10 nA, where 5 nA corresponds to a 200 MeV beam with a linearly scaled flux of $2.3 \times 10^8\,\mathrm{p/(cm^2\,s)}$. The beam size and shape is adjusted can be done using copper collimators, and the DUT is placed in front of the collimators output (Figure 6.1a).

Theoretically, TID testing can also be performed using the PIF proton beam. However, due to limited access times and strict testing schedules, TID qualification is done at CERN's own irradiation facility, Cobalt-60 (CC60). There, a ^{60}Co source decays into ^{60}Ni, emitting an electron (β^- decay) and two photons (γ decay). The electron interacts with the container holding the source, while the photons pass unimpeded through a collimator into the irradiation area. The DUT can be mounted inside the area, and the irradiation dose rate can be adjusted over the distance between the collimator and the DUT. In addition, two sources with different dose rates can be selected, a 10 TBq ^{60}Co source and a 110 TBq ^{60}Co source. Cables are routed to the DUT via conduits which are placed through the surrounding walls from the control area to the irradiation area.

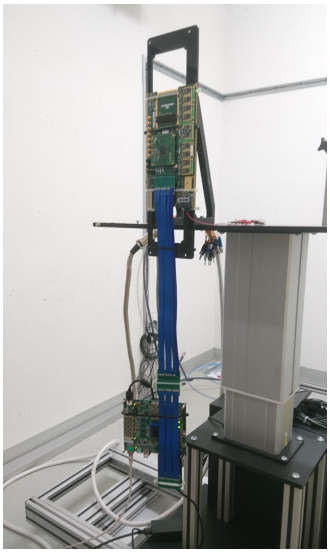
The metrics collected vary depending on the facility where the test is conducted. At PSI, the main targeted metric is the cross-section σ. This is a measure of the probability that a specific radiation-induced event will occur. The calculation of the cross-section requires the number of events N_{events}, the accumulated particle fluence Φ and the number of components tested $N_{\text{components}}$, as shown in (6.1).

$$\sigma = \frac{N_{\text{events}}}{\Phi \cdot N_{\text{components}}} \tag{6.1}$$

The cross-section is a measure of the probability of the occurrence of a particular radiation event. Therefore, it is desirable to minimize this value.



(a) Setup of the GateMate at PSI



(b) Setup of the GateMate at CC60

Figure 6.1: GateMate radiation test setups at PSI (Figure 6.1a) and CC60 (Figure 6.1b)

For an FPGA the cross-section of the individual parts as well as the design failure cross-section are of interest. Since the proton beam is composed of charged particles, the DUT will also receive an ionizing dose. As such, TID effects are also a measurable metric. However, the disadvantage of TID testing at PSI is the availability of beam time per device. A PSI campaign typically includes multiple device tests, which can only be performed sequentially. Therefore, long-term testing is not possible without interfering with other tests. The CC60 facility on the other hand is dedicated to such multi-day TID tests and offers the additional benefit of not inducing SEEs. This is because the photons emitted by the ^{60}Co source do not have an energy high enough to induce SEEs.

6.2 GateMate A1 Characterization

The GateMate A1 FPGA was tested on three PSI campaigns, spanning from May 2023 to September 2023. During these campaigns, the flip-flops, BRAMs and PLLs were tested. The Benchmark design was not implemented successfully during this period, as timing issues caused the error bit to be set even in the absence of radiation. Therefore, no data could be collected for the performance of the GateMate using a real-world example. The upper and lower confidence limits of the SEU cross section were calculated according to [24].

6.2.1 BRAM Results

The BRAM of the GateMate was tested only without the ECC functionality enabled, which led to the results shown in Table 6.1. All 32 cells are used, the data width is set to 10 bits, and the address bus is 12 bits wide. This results in a total of 1310720 addressable memory bits in the design.

Table 6.1: BRAM test measurement results

Pattern	SEUs	Fluence / $\frac{\text{p}}{\text{cm}^2}$	Cross-section / $\frac{\text{cm}^2}{\text{bit}}$	Upper-limit / $\frac{\text{cm}^2}{\text{bit}}$	Lower-limit / $\frac{\text{cm}^2}{\text{bit}}$
0	555	1.89×10^{10}	2.24×10^{-14}	1.95×10^{-14}	2.54×10^{-14}
1	205	6.09×10^{9}	2.57×10^{-14}	2.14×10^{-14}	3.02×10^{-14}
1010	213	5.92×10^{9}	2.75×10^{-14}	2.30×10^{-14}	3.23×10^{-14}
Total	973	3.09×10^{10}	2.40×10^{-14}	2.69×10^{-14}	2.12×10^{-14}

Combining all measurements together gives an average cross-section of $2.40 \times 10^{-14}\,\text{cm}^2/\text{bit}$. No MBUs were observed during the entire irradiation campaigns, resulting in an MBU upper-limit cross-section of $9.10 \times 10^{-17}\,\text{cm}^2/\text{bit}$. It can be observed that the cross-section does not change significantly when the input pattern is changed, i.e., the cross-section is independent of the data stored in the memory cell. In addition to the test above, which was performed at a beam energy of 200 MeV, an energy scan as recommended in [24] was also conducted (compare Figure 6.2).

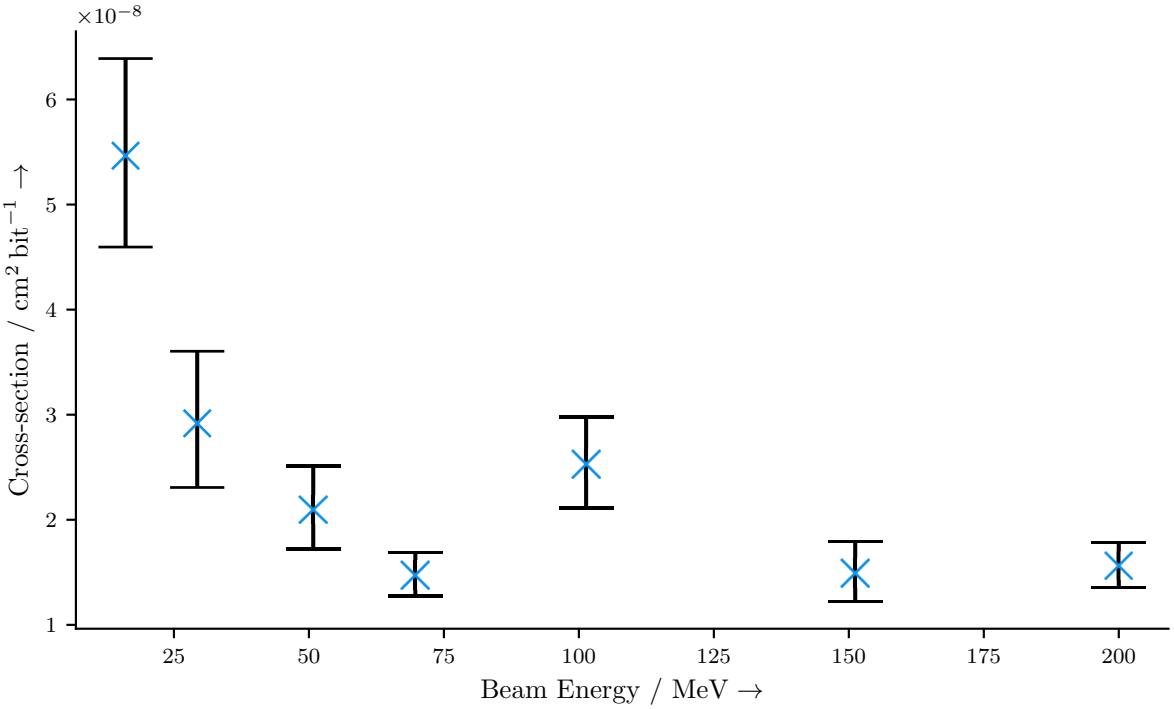
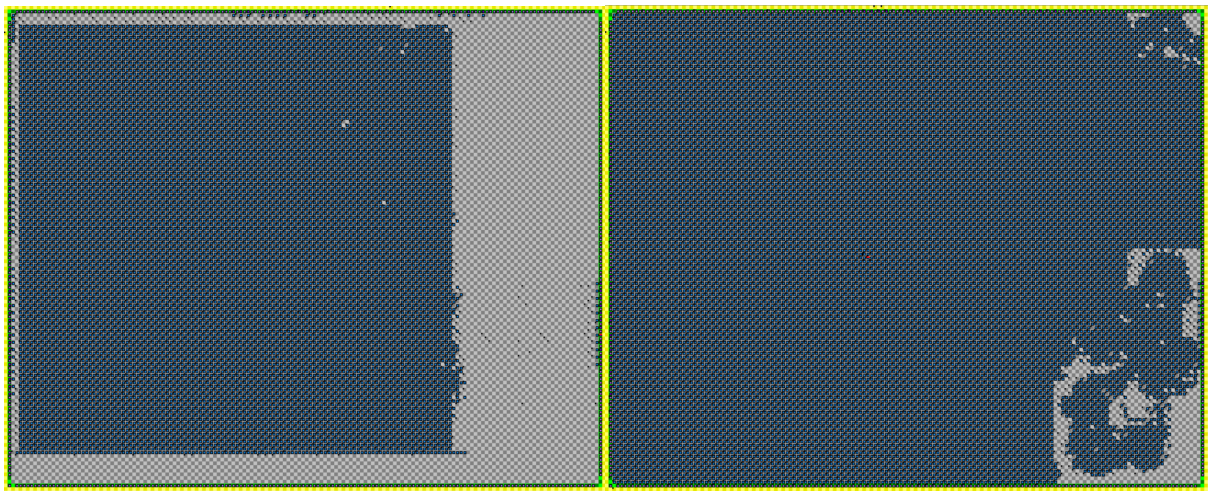


Figure 6.2: BRAM energy scan

The scan shows that the cross-section increases when lower beam energies are used. This can be attributed to proton direct ionization, where a proton-silicon interaction releases one or more electrons [20]. These electrons can deposit all of their energy in a critical part of the memory cell via Coulomb scattering events, which can cause an SEU due to exceeding the critical energy threshold.

6.2.2 Flip-Flop Results

Two variants of the flip-flop design were investigated, with and without TMR flip-flops in the chains. In addition, two implementation strategies were evaluated. By using the `no_space_opt` flag, the P&R tool did not attempt to optimize the design to use as few CPEs as possible. A comparison of the two strategies is shown in Figure 6.3, where the same design is being implemented. It can be seen that without the space optimization, almost all of the FPGA space is occupied.



(a) Implementation of the flip-flop design with space optimization

(b) Implementation of the flip-flop design without space optimization

Figure 6.3: Comparison between the flip-flop design implementation with and without space optimization

The flip-flop test was performed with 12 WSR chains for each implementation variant, but with a different number of flip-flops per chain (Table 6.2).

Table 6.2: CPE usage statistics for each design variant. The values in the configuration column are used in the later tables to reference the corresponding design configurations.

Flip-flop type	Space optimization	FFs per chain	CPEs used	FPGA utilization	Configuration
TMR	Yes	150	6963	34 %	1
TMR	No	150	6963	34 %	2
Simple	Yes	2000	13721	67 %	3
Simple	No	2000	13721	67 %	4

The results for the flip-flop test are shown in Table 6.3. The primary difficulty with the flip-flop test was the inherent sensitivity of the GateMate's CRAM to radiation. During testing, the many of the WSR chains experienced a design corruption before capturing any SEUs. Nevertheless, it was still possible to calculate the flip-flop cross-section, albeit with a high uncertainty. Since the no-TMR implementation without space optimization showed no SEUs at all, only the upper-limit cross-section can be provided for this configuration.

Table 6.3: Results of the flip-flop radiation test. The upper limit is given where no SEUs was observed.

Config.	SEUs	Fluence / $\frac{\text{p}}{\text{cm}^2}$	Cross-section / $\frac{\text{cm}^2}{\text{flip-flop}}$	Upper-limit / $\frac{\text{cm}^2}{\text{chain}}$	Lower-limit / $\frac{\text{cm}^2}{\text{chain}}$
1	1	1.23×10^{12}	4.51×10^{-16}	2.59×10^{-15}	9.38×10^{-18}
2	1	6.61×10^{11}	8.40×10^{-16}	4.68×10^{-15}	1.70×10^{-17}
3	2	7.33×10^{10}	1.14×10^{-15}	4.11×10^{-15}	1.31×10^{-16}
4	0	2.81×10^{10}	N/A	5.47×10^{-15}	N/A

While the WSRs with TMR showed a slightly smaller cross-section than the regular WSRs, all configurations do not differ significantly from each other (Figure 6.4).

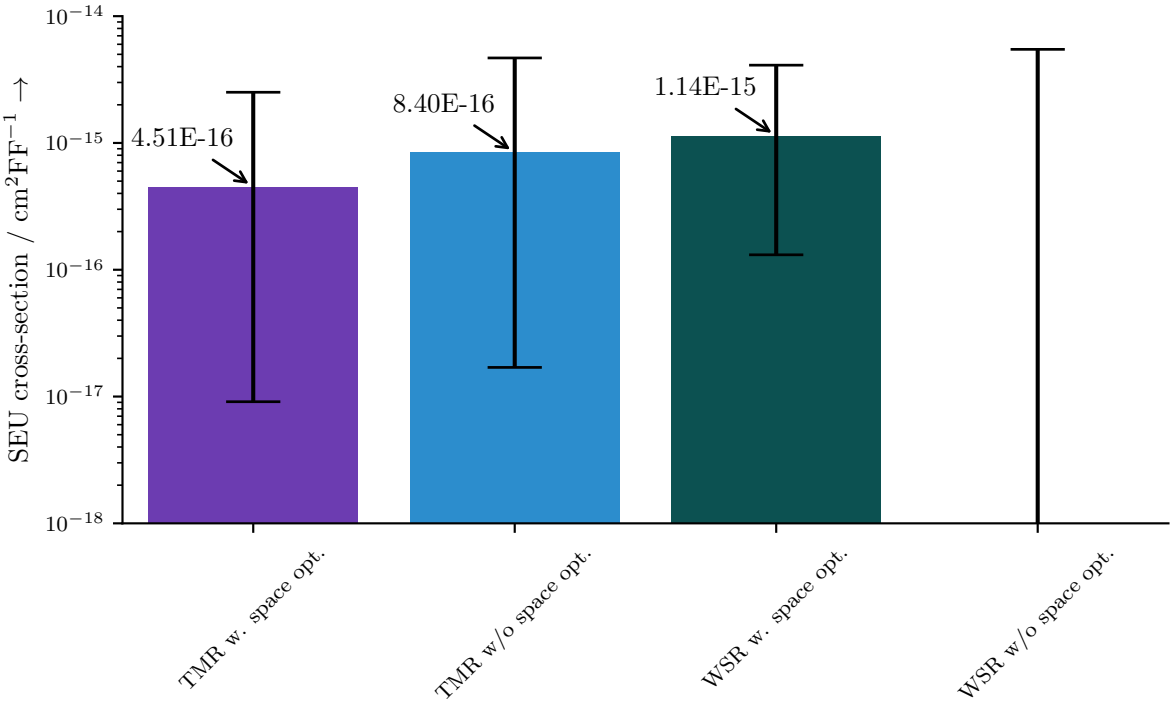


Figure 6.4: Flip-flop SEU cross-section vs. design configuration

Table 6.4 shows the average fluence to design corruption of a WSR chain for each configuration, in addition to the cross-section of a chain regarding design corruptions. This shows that only the design configuration has an effect on the design corruption cross-section, not the inclusion of TMR (Figure 6.5).

Table 6.4: Average fluence to design corruption and design corruption cross-section per configuration

Config	Fluence / $\frac{\text{p}}{\text{cm}^2}$	Cross-section / $\frac{\text{cm}^2}{\text{chain}}$	Upper-limit / $\frac{\text{cm}^2}{\text{chain}}$	Lower-limit / $\frac{\text{cm}^2}{\text{chain}}$
1	1.23×10^{12}	1.80×10^{-14}	2.48×10^{-10}	1.26×10^{-14}
2	6.61×10^{11}	3.19×10^{-14}	4.43×10^{-10}	2.21×10^{-14}
3	7.33×10^{10}	1.65×10^{-14}	2.39×10^{-10}	1.08×10^{-14}
4	2.81×10^{10}	3.26×10^{-14}	4.97×10^{-10}	2.00×10^{-14}

By optimizing the area usage of the design, and thus increasing the density of CPEs used, the likelihood of a design failure is reduced. As the optimization reduces the area where CPEs are placed, the chance of a critical hit in an element (CPE, switch box etc.) decreases.

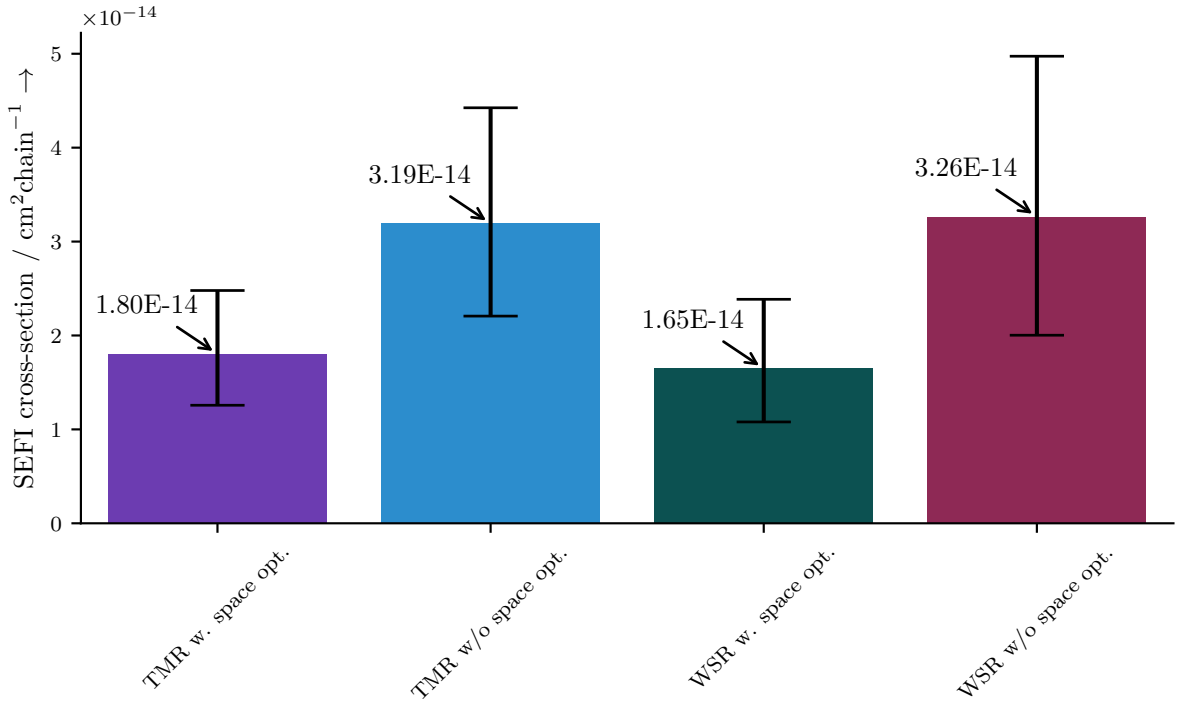


Figure 6.5: Design corruption cross-section vs. design configuration

6.2.3 PLL Results

To optimize the beam time usage, the PLL test design was bootstrapped to the flip-flop test, as both designs do not interfere with each other. The FPGA was exposed to a total fluence of 1.44×10^{11} p/cm^2 during the campaign, but no loss of a PLL lock was observed. This results to an upper limit cross-section of 6.40×10^{-12} cm^2 for an SEE error in a PLL block.

6.2.4 TID Results

The TID test was performed using 130 ring oscillator chains, each consisting of 64 D-type latches (Figure 6.6). The number of latches per chain was chosen as the GateMate's routing structure is interleaved with BRAM cells every 32 CPEs [16]. By using 64 CPEs per chain, which are divided into two rows with 32 CPEs per row as described in Section 2.2.1, no BRAM cells are crossed during routing. This reduces potential propagation delay differences between the individual chains. The 130 chains allow full use of the GateMate's fabric, while leaving space between each CPE chain for easy routing.

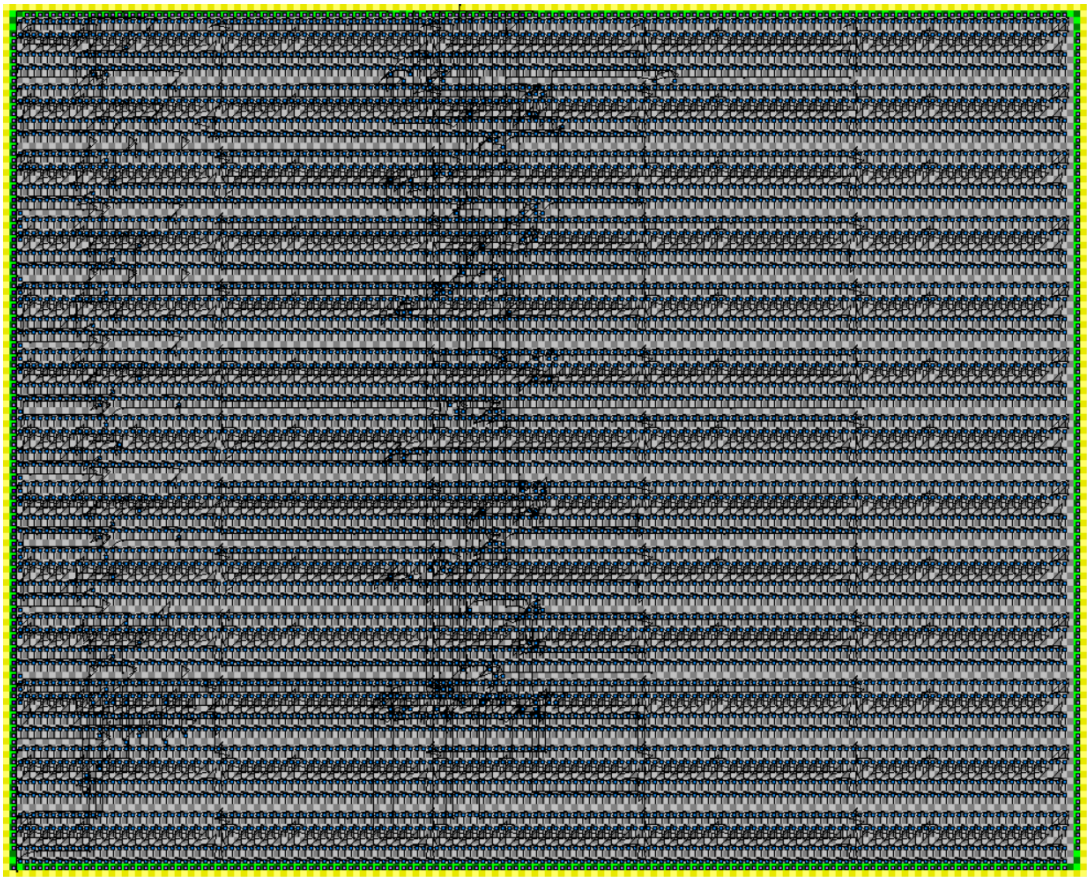


Figure 6.6: Implementation of the 130 ring oscillators

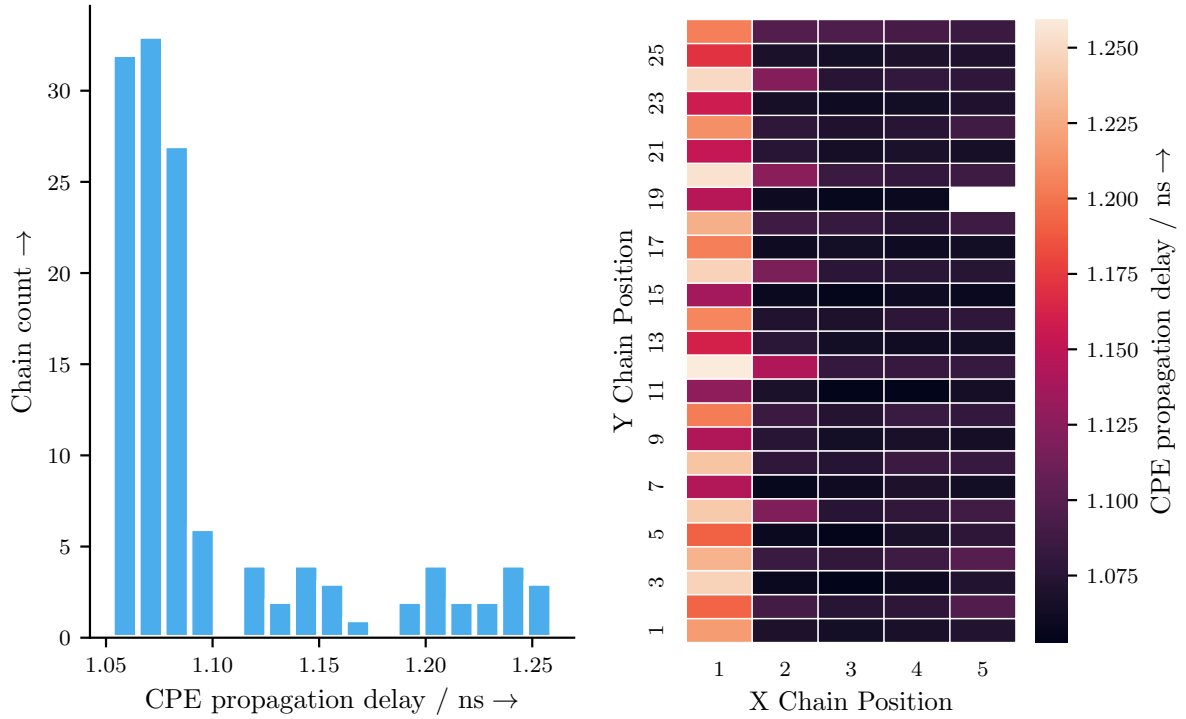
Prior to irradiation, a baseline is established by measuring the oscillation frequency of the output of each chain. This is done on the Zynq platform by calculating the external frequency, i.e., the frequency of the measured chain, using (6.2). N_{ext} represents the number of counted pulses from the chain, while N_{ref} is the number of pulses originating from the internal clock source wiht frequency F_{ref}.

$$F_{\text{ext}} = \frac{N_{\text{ext}}}{N_{\text{ref}}} F_{\text{ref}} \tag{6.2}$$

The average propagation delay of each CPE can then be approximated by (6.3).

$$t_{\text{CPE}} = \frac{1}{2 \cdot F_{\text{ext}} \cdot N_{\text{CPE}}} \tag{6.3}$$

It is important to note however, that the delay calculated above ignores the routing delays that exist between each CPE, and is only used to compare pre- and post-irradiation values. Using the average propagation delay per CPE for each chain a distribution (Figure 6.7a) is created. This shows the average CPE propagation delays, with $66\,\%$ of all chains having a CPE propagation delay between $1.05\,\text{ns}$ and $1.10\,\text{ns}$. The same data can also be used to determine if there is a correlation between chain location and CPE propagation delay. As can be seen in Figure 6.7b, the propagation delay for the chains from x positions 2 to 5 do not vary much. Only the chains in x position 1, and a few in x position 2, differ. The reason for this behavior is not clear. It can be argued that the path from x position 1 to the outputs of the GateMate is longer, and therefore fewer events could be measured in the same time compared to the other chains.



(a) CPE propagation delay distribution

(b) Spatial distribution of CPE propagation delays

Figure 6.7: Distribution of the CPE delays for the ring oscillator test

Once the baseline was established, the irradiation of the GateMate was performed at the CC60 facility. The GateMate Mezzanine Board for the CRaTeBo was used, which was placed in line with the aperture of the ^{60}Co source. Over the course of three days, the FPGA was irradiated to a total dose of 9 kGy. During this time, the propagation delays of all chains were monitored, as well as the current consumption of the entire CRaTeBo (Figure 6.8).

After reaching a maximum delay change of 8 %, the change slowly decreases. While initially the current consumption reduces briefly up to 500 Gy, it then increases afterwards. At its highest point, which is at the end of the test, the current consumption increased by 166 %, from 360 mA to 660 mA. This change in current consumption can be attributed to the TID effects that occur in the STI of the GateMate as described in Section 2.2.1. When the delay change curve is superimposed on the curve of the current consumption curve, it can be seen that both curves exhibit a similar increase up to 4 kGy. From this point however, the curves begin to diverge, with the delay change decreasing while the current consumption continues to increase. Figure 6.9 shows the relative delay change distribution at the point of maximum delay change and at the end of irradiation.

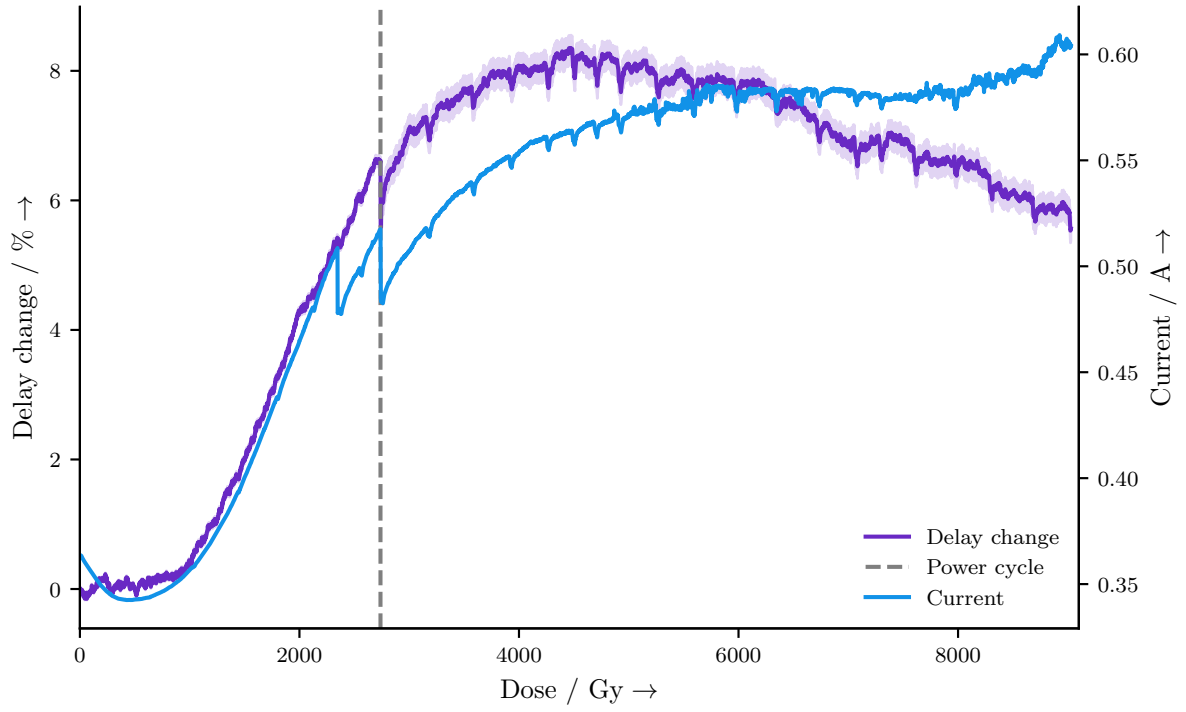
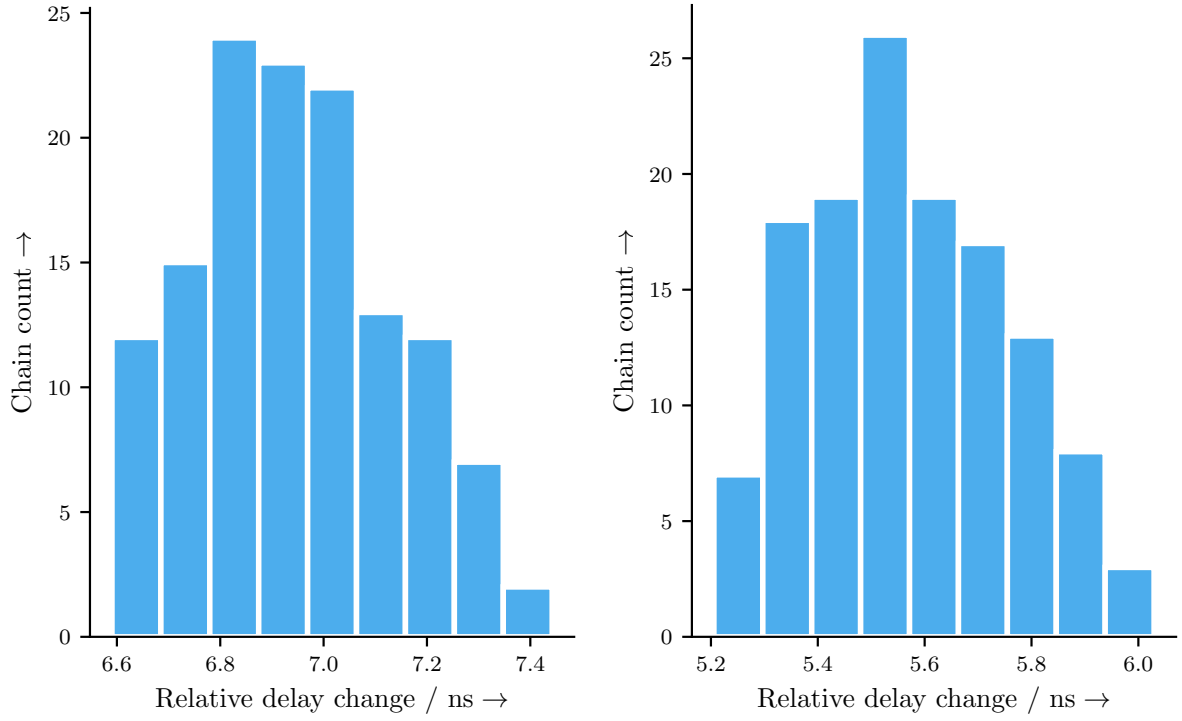


Figure 6.8: Relative delay change vs. dose and current consumption vs. dose

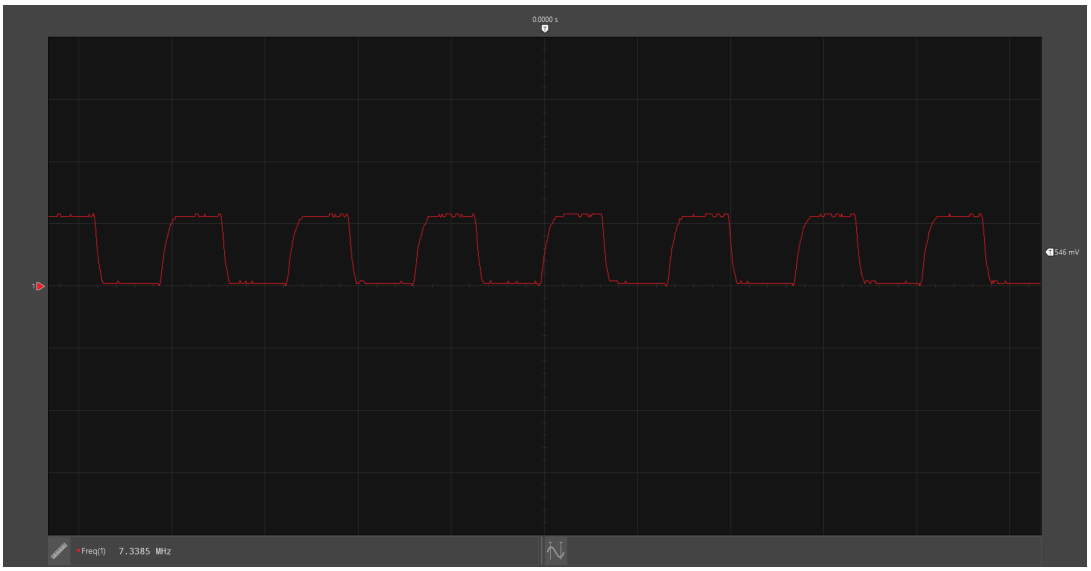


(a) Relative delay distribution at maximum delay change

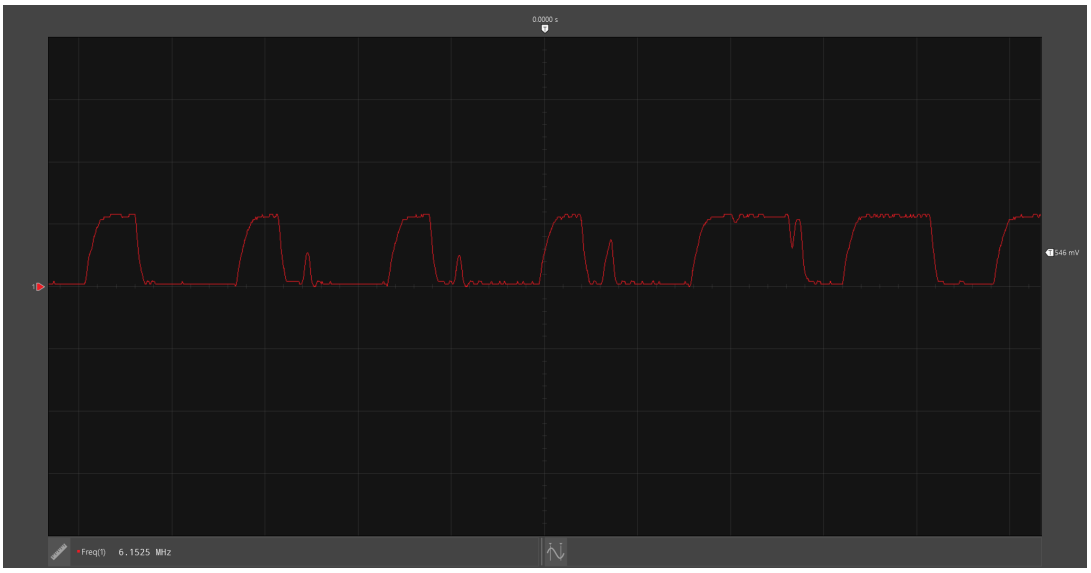
(b) Relative delay distribution at the end of exposure

Figure 6.9: Distribution of the CPE delays for the ring oscillator test

To check for additional effects of the irradiation on the GateMate, the output of a chain was recorded with an oscilloscope, both with the irradiated sample, and with a new device. This is shown in Figure 6.10, where the difference between the two outputs is clearly visible.



(a) Ring oscillator output signal of an non-irradiated GateMate FPGA



(b) Ring oscillator output signal of an irradiated GateMate FPGA

Figure 6.10: Output signal comparison between an irradiated and non-irradiated GateMate FPGA

While the new GateMate produces an almost perfect square wave, the irradiated GateMate produces highly irregular signals with inconsistent pulse widths.

6.2.5 Design Corruption and SEFI Cross-Section

To determine the GateMate's sensitivity to a design corruption, all observed design failures and the fluence at which these failures occurred are processed. A total of 134 failures were recorded, with a total fluence of $2.05 \times 10^{12}\,\mathrm{p/cm^2}$. This results in a design corruption cross-section of $6.54 \times 10^{-11}\,\mathrm{cm^2}$, with an upper limit of $7.91 \times 10^{-11}\,\mathrm{cm^2}$ and a lower limit of $5.29 \times 10^{-11}\,\mathrm{cm^2}$.

During all campaigns, no SEFIs could be observed. As a result, only the upper limit of the SEFI cross section can be calculated, which results in $1.80 \times 10^{-12}\,\mathrm{cm}^2/\mathrm{device}$.

6.3 Comparison of Results

Since it is common to use FPGAs in radiation environments, other FPGAs were also tested for their sensitivity to radiation effects. This section summarizes the following reports regarding the components tested on the GateMate:

1. Microsemi SmartFusion2 M2S010: A flash-based FPGA built using a 28 nm process. Evaluated in [67] with a 200 MeV proton beam;
2. Microsemi PolarFire MPF300TS: A flash-based FPGA built using a 28 nm process. Evaluated in [59] with a 200 MeV proton beam;
3. NanoXplore NG-MEDIUM NX1H35S: An SRAM-based FPGA built using a 65 nm radiation-hardened process. Evaluated in [34] with a 200 MeV proton beam;
4. Xilinx Kintex-7 XC7K325T: An SRAM-based FPGA built using a 28 nm process. Evaluated in [15] with a 180 MeV proton beam;
5. Xilinx Kintex-7 XC7K70T: An SRAM-based FPGA built using a 28 nm process. Evaluated in [54] with a 35 MeV proton beam.

For this comparison, the results for BRAM cells (Table 6.1), flip-flops (Table 6.6), PLLs (Table 6.7), SEFIs (Table 6.8) and TID (Table 6.9) tests are presented where applicable.

Table 6.5: Comparison of BRAM test results

Vendor	FPGA	Cross-section $\left[\frac{\mathrm{cm}^2}{\mathrm{bit}}\right]$
Microsemi	M2S010	2.36×10^{-14}
	MPF300TS	1.37×10^{-14}
NanoXplore	NX1H35S	$6.13 \times 10^{-14} \leq \sigma \leq 8.01 \times 10^{-14}$
Xilinx	XC7K325T	8.19×10^{-15}
	XC7K70T	6.9×10^{-15}
Cologne Chip	CCAGA1	2.40×10^{-14}

Table 6.6: Comparison of flip-flop test results

Vendor	FPGA	Cross-section $\left[\frac{\mathrm{cm}^2}{\mathrm{flip\text{-}flop}}\right]$	TMR enabled
Microsemi	M2S010	4.30×10^{-15}	No
	M2S010	7.90×10^{-15}	Yes
	MPF300TS	2.31×10^{-14}	No
	MPF300TS	3.13×10^{-15}	Yes
NanoXplore	NX1H35S	1.3×10^{-15}	No
Xilinx	XC7K70T	5.00×10^{-15}	No
Cologne Chip	CCAGA1	1.14×10^{-15}	No
	CCAGA1	4.51×10^{-16}	Yes

Table 6.7: Comparison of PLL test results

Vendor	FPGA	Cross-section $\left[\frac{\text{cm}^2}{\text{PLL}}\right]$
Microsemi	M2S010	4.76×10^{-12}
	MPF300TS	$\leq 3.21 \times 10^{-14}$
NanoXplore	NX1H35S	$\leq 1.02 \times 10^{-11}$
Cologne Chip	CCAGA1	$\leq 6.40 \times 10^{-12}$

Some additional clarification is needed for the values in Table 6.8. The Microsemi MPF300TS SEFIs can be attributed to two sources, the clock tree and the SpaceWire IP. In the case of the former, an SEE induced in the clock tree can propagate to different areas of the FPGA depending on which level of the clock tree the fault is induced. Because a SEE in the higher levels of the clock tree is more visible than at lower levels, the cross-section of such a fault can vary [58]. The latter source, the hardcoded SpaceWire IP core, implements a feature that resets the FPGA in the event of a core failure. This reset can be triggered even if the core is not used in the design. Since the core is active by default, special care must be taken to disable it to prevent this type of SEFI. This is also the case for the NanoXplore NX1H35S, where several hardcoded IP cores trigger resets in the event of core failures. Again, manual deactivation is required to prevent such radiation induced resets. As mentioned above, the GateMate did not show any SEFIs during the campaigns. Therefore, only the upper limit cross-section can be used as a value to contextualize the SEFIs cross-sections between the FPGAs.

Table 6.8: Comparison of SEFI test results

Vendor	FPGA	Cross-section
Microsemi	MPF300TS	$2.29 \times 10^{-11}\ \frac{\text{cm}^2}{\text{device}}$
NanoXplore	NX1H35S	Between $1.25 \times 10^{-11}\ \frac{\text{cm}^2}{\text{device}}$ and $2.9 \times 10^{-11}\ \frac{\text{cm}^2}{\text{device}}$
Cologne Chip	CCAGA1	$\leq 1.80 \times 10^{-12}\ \frac{\text{cm}^2}{\text{device}}$

Table 6.9: Comparison of TID test results. As the doses differ between the reports, all values shown are after an irradiation to 3 kGy. It is also worth noting that most radiation tolerant systems are designed around a maximum dose of 500 Gy to 1000 Gy. In addition, the Microsemi MPF300TS is manufactured using the same process technology used in radiation tolerant devices and the NanoXplore NX1H35S is radiation hardened by design.

Vendor	FPGA	Comment
Microsemi	MPF300TS	$0.4\,\%$ delay difference from before irradiation
NanoXplore	NX1H35S	No degradation was observed
Cologne Chip	CCAGA1	$8.10\,\%$ delay difference from before irradiation and a $157\,\%$ increase in current consumption

Comparing the values presented in the tables above with the results obtained during the GateMate radiation campaigns, it can be seen that the GateMate values are similar to those for other FPGAs. The only deviation here is for the TID test, as the FPGAs tested in other publications did not show any degradation after irradiation. A possible explanation for this behavior may be the different process technologies used, but to further validate this claim, TID tests with non-radiation-hardened SRAM-based FPGAs have to be performed.

7

Summary and Outlook

The subject of this thesis was the radiation qualification of the Cologne Chip GateMate A1 FPGA. This is an SRAM based FPGA manufactured in a 28 nm process with 20480 CLBs, 160 Kibits total RAM, 162 I/O pins, four PLLs and one SerDes. Each CLB contains an 8-input LUT and two flip-flops. The RTL code is synthesized using the *Yosys* open source synthesis framework. The implementation part of the software design flow is handled by a proprietary *Place and Route* tool. The upload of the final bit stream to the FPGA is carried out using an additional open source tool, *openFPGALoader*.

After an introduction to the interaction mechanisms by which radiation interacts with matter, the effects on electronic devices are discussed. These can be broadly divided into two types of effects, *cumulative* and *single-event*. Cumulative effects are further subdivided into *total ionizing dose* and *displacement damage dose*. The former is caused by interactions with charged particles and results in a degradation of the electrical properties of the affected components due to the accumulation of trapped charges in oxide layers. The latter is caused by non-ionizing interactions that cause physical damage to the device by striking atoms in the semiconductor lattice. Single-event effects, on the other hand, typically induce non-permanent defects. The behavior of an electrical device can be temporarily affected if an incident particle strikes a critical part of the device. For example, an impact on a flip-flop cell can cause the contents of the cell to flip from a `1` to a `0` or vice versa (a so-called SEU). A false clock or reset signal can be caused by a sufficiently energetic particle hitting a clock or reset driver. While mitigation techniques such as triple modular redundancy, where parts of a design are triplicated and their output fed to a majority voter, exist, this approach cannot deal with all the types of effects mentioned above. Therefore, a radiation qualification process is required to estimate component failure rates.

Five types of radiation tests have been developed for the qualification of the GateMate A1 FPGA: Three component-specific tests (BRAM cells, flip-flops and PLLs), an application-level benchmark test and a test to evaluate the TID response. A custom setup, consisting of two FPGAs, was developed for each of the tests: The GateMate A1 as the DUT and an Xilinx Zynq UltaScale+ as the TESTER FPGA, which provides an input stimulus to the DUT, captures its response and provides this information to the user. The designs for the GateMate were developed specifically for testing, with a focus on maximizing the number of tested component instances and reducing any unnecessary logic. The TESTER contains custom IP cores that are accessed and controlled by the Zynq's CPU via AXI4-Lite. In addition, a PYNQ server running on the Zynq allows the user to interface with the IP core via a browser and the Python programming language.

The GateMate A1 was evaluated for single-event effects in three radiations campaigns at the Paul Scherrer Institute using a proton beam. The BRAM cells show a combined cross-section of $2.40 \times 10^{-14}\,\mathrm{cm}^2$/bit, which is comparable to other FPGAs with the same process technology. Additionally, an energy scan from 16 MeV to 200 MeV shows an increase in the SEU cross-section at lower energies.

Extracting the sensitivity of the flip-flops on the other hand was more complex because design failures were observed more frequently than SEEs. Thus, the flip-flops have an SEU cross-section of 1.14×10^{-15} cm^2/flip–flop, albeit with a high uncertainty. Adding TMR to the flip-flop design or changing the implementation strategy of the Place and Route tool did not significantly change the SEU cross-section. However, the design corruption cross-section was affected by the implementation strategy. When the default option to optimize for area usage was disabled, no SEUs were observed. This results in an SEU upper-limit cross-section of 5.47×10^{-15} cm^2/flip–flop. No single-event effects were observed for PLLs, resulting in a 6.40×10^{-12} cm^2/PLL as the upper limit of their cross-section. Since the final implementation of the benchmark design was not finalized before the last irradiation campaign, no test was performed on the application benchmark. The evaluation of the TID response was carried out at CERN's own Cobalt-60 facility, which is dedicated to TID testing of electronics. The GateMate FPGA was irradiated for three days at a total dose of 9 kGy. During this time period, no failures were detected, current consumption increased by $166\,\%$ and the CLB propagation time showed a maximum delay increase of $8\,\%$. Finally, by processing all design failures that occurred during the campaigns, the cross-section of the GateMate to CRAM corruption is estimated to be 6.54×10^{-11} cm^2, with an upper limit of 7.91×10^{-11} cm^2 and a lower limit of 5.29×10^{-11} cm^2. No single-event functional interrupts have been observed, therefore the upper limit cross-section for SEFIs is estimated to 1.80×10^{-12} cm^2/device.

Several components of the Cologne Chip GateMate A1 FPGA were evaluated under radiation. However, due to time constraints, not all available parts of the FPGA were included in the tests. For example, the BRAM cells have an ECC mode to correct 1-bit errors and detect 2-bit errors. An interesting test would be to see if the SEU and MEU cross-sections of the BRAM cells decrease when this mode is used. As mentioned above, the benchmark circuit was not completed in the time available. A test under radiation is also missing here.

Since many designs exhibited failures due to CRAM corruption, the extraction of its cross-section is an important FPGA radiation qualification metric. The standard way of obtaining this information is to read out the FPGA's configuration after irradiation. Unfortunately, the GateMate has no such feature. Another approach could be a test design that targets the cross-section of the LUTs. This is possible because the GateMate's LUTs consist of the same elements as the CRAM, which can be used to determine the desired metric.

List of Acronyms

ALICE A Large Ion Collider Experiment
AMBA Advanced Microcontroller Bus Architecture
ANSI American National Standards Institute
API application programming interface
ASIC application-specific integrated circuit
ATLAS A Toroidal LHC ApparatuS
AXI Advanced eXtensible Interface

BES Bottom Edge Select
BJT bipolar junction transistor
BRAM block RAM
BSP board support package

CC60 Cobalt-60
CERN European Organization for Nuclear Research
CHARM CERN Highly-Accelerated Mixed Field Facility
CLB configurable logic block
CMOS complementary metal-oxide semiconductor
CMS Compact Muon Solenoid
COMET COmpact MEdical Therapy cyclotron
COTS commercial off-the-shelf
CP-line carry & propagation signal line
CPE Cologne Processing Element
CPU central processing unit
CRAM configuration RAM
CRaTeBo CHARM radiation tester board

DC direct current
DCO digitally controlled oscillator
DD displacement damage
DDD displacement damage dose
DDR double data rate
DRAM dynamic RAM
DSP digital signal processing
DUT device under test

e-h electron-hole
ECC error correction code
EMI electromagnetic interference
ESA European Space Agency
ESCC European Space Components Coordination

FA full adder
FF flip-flop
FIFO first-in first-out
FMC FPGA Mezzanine Card
FPGA field-programmable gate array
FSM finite state machine
FTDI Future Technology Devices International

GBT GigaBit Transceiver
GHDL G Hardware Design Language
GPIO general purpose I/O
GPU graphics processing unit

HDL hardware description language
HLS high-level synthesis
HPC high pin count

I/O input/output
I^{2}C Inter-Integrated Circuit
IC integrated circuit
IEL ionizing energy loss
IP intellectual property

JEDEC Joint Electron Device Engineering Council
JTAG Joint Action Test Group

L-GEFE Link-GBT Based Expandable Front-End
LED light emitting diode
LES Left Edge Select
LET linear energy transfer
LHC Large Hadron Collider
LHCb Large Hadron Collider beauty
Linac4 Linear accelerator 4
LPC low pin count
LSFR linear feedback shift register
LUT lookup table

MAC multiply and accumulate
MBU multiple-bit upset
MEU multiple-event upset
MOS metal-oxide semiconductor

MOSFET metal-oxide semiconductor field-effect transistor
MPSoC multiprocessor system-on-chip
MUX multiplexer

NIEL non-ionizing energy loss

opcode operation code

P&R Place and Route
PCB printed circuit board
PIF Proton Irradiation Facility
PL programmable logic
PLL phase-locked loop
Pmod Peripheral module
POR power-on reset
PS
- **PS** Processing System
- **PS** Proton Synchrotron

PSB Proton Synchrotron Booster
PSI Paul Scherrer Institute
PYNQ Python Productivity for Zynq

RAM random access memory
RC resistor-capacitor
RHA radiation hardness assurance
RTL register-transfer level

SB_BIG Big Switch Box
SB_SMALL Small Switch Box
SCA Slow Control Adapter
SDF Standard Delay Format
SDP simple dual port
SEB single-event burnout
SEE single-event effect
SEFI single-event functional interrupt
SEGR single-event gate rupture
SEL single-event latch-up
SerDes Serializer/Deserializer
SET single-event transient
SEU single-event upset
SFP Small Form-factor Pluggable
SLP Super Low Power
SLVS scalable low-voltage signaling
SMA SubMiniature version A
SMT surface mount technology
SoC system-on-chip
SoM system-on-module
SPI Serial Peripheral Interface
SPS Super Proton Synchrotron
SR shift register
SRAM static RAM
STA static timing analysis
STI shallow trench isolation

TDP true dual port
THT through hole technology
TI Texas Instruments
TID total ionizing dose
TMR triple modular redundancy

UART universal asynchronous receiver-transmitter
USB Universal Serial Bus

VCO voltage controlled oscillator
VHDL Very High Speed Integrated Circuits (VHSIC) Hardware Description Language
VHSIC Very High Speed Integrated Circuits

WSR window shift register

XRT Xilinx Runtime Library

Yosys Yosys Open SYnthesis Suite

List of Tables

List of Figures

List of Listings

Bibliography

[1] G. Aad et al. "Observation of a new particle in the search for the Standard Model Higgs boson with the ATLAS detector at the LHC." In: *Physics Letters B* 716.1 (2012), pp. 1–29. ISSN: 03702693. DOI: 10.1016/j.physletb.2012.08.020.

[2] Hideharu Amano. *Principles and Structures of FPGAs*. Singapore: Springer Singapore, 2018. ISBN: 978-981-13-0824-6. DOI: 10.1007/978-981-13-0824-6.

[3] L. Amaral et al. "The versatile link, a common project for super-LHC." In: *Journal of Instrumentation* 4.12 (2009), P12003–P12003. DOI: 10.1088/1748-0221/4/12/P12003.

[4] American National Standards Institute. *ANSI-VITA 57.1-2008: American National Standard for FPGA mezzanine card (FMC) standard*. Fountain Hills (Arizona), 2010.

[5] Juan José Rodríguez Andina, Eduardo de La Torre Arnanz, and María Dolores Valdés Peña. *FPGAs*. CRC Press, 2017. ISBN: 9781315162133. DOI: 10.1201/9781315162133.

[6] ARM Limited. *AMBA AXI and ACE: Protocol Specification*. Ed. by ARM Limited. Jan. 26, 2021. URL: https://developer.arm.com/documentation/ihi0022/latest/ (visited on 02/09/2023).

[7] Arrow Electronics. *FPGA vs CPU vs GPU vs Microcontroller: How Do They Fit into the Processing Jigsaw Puzzle?* Ed. by Arrow Electronics. 2018. URL: https://www.arrow.com/en/research-and-events/articles/fpga-vs-cpu-vs-gpu-vs-microcontroller (visited on 09/23/2022).

[8] S. Baron et al. *The GBT Project*. 2009. DOI: 10.5170/CERN-2009-006.342.

[9] Robert Baumann and Kirby Kruckmeyer. *Radiation Handbook for Electronics*. 2019. URL: https://www.ti.com/radbook (visited on 09/13/2022).

[10] Melanie Berg. *Field Programmable Gate Array (FPGA) Single Event Effect (SEE) Radiation Testing*. Ed. by NASA/Goddard Space Flight Center. Feb. 2, 2012. URL: https://nepp.nasa.gov/files/23779/fpga_radiation_test_guidelines_2012.pdf (visited on 08/27/2022).

[11] Matteo Brucoli. "Total Ionizing Dose Monitoring for Mixed Field Environments." PhD thesis. Université Montpellier, 2018. URL: https://tel.archives-ouvertes.fr/tel-02155482 (visited on 06/13/2022).

[12] A. Caratelli et al. "The GBT-SCA, a radiation tolerant ASIC for detector control and monitoring applications in HEP experiments." In: *Journal of Instrumentation* 10.03 (2015), pp. C03034–C03034. DOI: 10.1088/1748-0221/10/03/C03034.

[13] S. Chatrchyan et al. "Combined results of searches for the standard model Higgs boson in pp collisions at s=7 TeV." In: *Physics Letters B* 710.1 (2012), pp. 26–48. ISSN: 03702693. DOI: 10.1016/j.physletb.2012.02.064.

[14] Eric Chung et al. "Serving DNNs in Real Time at Datacenter Scale with Project Brainwave." In: *IEEE Micro* 38 (2018), pp. 8–20. URL: https://www.microsoft.com/en-us/research/publication/serving-dnns-real-time-datacenter-scale-project-brainwave/ (visited on 09/13/2022).

[15] M. Citterio et al. "Radiation testing campaign results for understanding the suitability of FPGAs in detector electronics." In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 824 (2016), pp. 270–271. ISSN: 01689002. DOI: 10.1016/j.nima.2015.11.033.

[16] Cologne Chip AG. *GateMate FPGA Datasheet: DS1001*. Ed. by Cologne Chip AG. Dec. 2022. URL: https://colognechip.com/docs/ds1001-gatemate1-datasheet-latest.pdf (visited on 12/12/2022).

[17] Cologne Chip AG. *GateMate FPGA Evaluation Board Datasheet: DS1003*. Ed. by Cologne Chip AG. Aug. 2022. URL: https://colognechip.com/docs/ds1003-gatemate1-evalboard-3v1-latest.pdf (visited on 12/12/2022).

[18] Cologne Chip AG. *GateMate FPGA User Guide Primitives Library: UG1001*. Ed. by Cologne Chip AG. Dec. 2022. URL: https://www.colognechip.com/docs/ug1001-gatemate1-primitives-library-latest.pdf (visited on 12/12/2022).

[19] F. Corno, M. S. Reorda, and G. Squillero. "RT-level ITC'99 benchmarks and first ATPG results." In: *IEEE Design & Test of Computers* 17.3 (2000), pp. 44–53. ISSN: 07407475. DOI: 10.1109/54.867894.

[20] Andrea Coronetti et al. "Proton direct ionization upsets at tens of MeV." In: *IEEE Transactions on Nuclear Science* (2022), pp. 1–1. DOI: 10.1109/TNS.2022.3207877.

[21] Salvatore Danzeca, Rudy Ferraro, and Georgios Tsiligiannis. *Guidelines for the Radiation Hardness Assurance (CERN-RHA) for CERN accelerators equipment*. Ed. by CERN. CERN EDMS 2066950 V1.0, 2019. URL: https://edms.cern.ch/document/2066950/1 (visited on 11/27/2022).

[22] Digilent, Inc. *Digilent Pmod Interface Specification: Revised October 28, 2020*. Ed. by Digilent, Inc. Oct. 28, 2020. URL: https://digilent.com/reference/_media/reference/pmod/pmod-interface-specification-1_3_1.pdf (visited on 07/16/2022).

[23] European Space Agency. *Guidelines for Displacement Damage Irradiation Testing: ESCC Basic Specification No. 22500*. Ed. by European Space Agency. Oct. 2021.

[24] European Space Agency. *Single Event Effects Test Method and Guidelines: ESCC Basic Specification No. 25100*. Ed. by European Space Agency. 2014.

[25] European Space Agency. *Space product assurance: ECSS-Q-ST-60-15C*. Ed. by European Space Agency. 2012.

[26] European Space Agency. *Total Dose Steady-State Irradiation Test Method: ESCC Basic Specification No. 22900*. Ed. by European Space Agency. 2016.

[27] Lyndon Evans and Philip Bryant. "LHC Machine." In: *Journal of Instrumentation* 3.08 (2008), S08001–S08001. DOI: 10.1088/1748-0221/3/08/S08001.

[28] F01 Committee. *Guide for Ionizing Radiation (Total Dose) Effects Testing of Semiconductor Devices*. West Conshohocken, PA. DOI: 10.1520/F1892-12R18.

[29] F01 Committee. *Guide for the Measurement of Single Event Phenomena (SEP) Induced by Heavy Ion Irradiation of Semiconductor Devices*. West Conshohocken, PA. DOI: 10.1520/F1192-11R18.

[30] F. Faccio and G. Cervelli. "Radiation-induced edge effects in deep submicron CMOS transistors." In: *IEEE Transactions on Nuclear Science* 52.6 (2005), pp. 2413–2420. ISSN: 0018-9499. DOI: 10.1109/TNS.2005.860698.

[31] Yi-Pin Fang and A. S. Oates. "Thermal Neutron-Induced Soft Errors in Advanced Memory and Logic Devices." In: *IEEE Transactions on Device and Materials Reliability* 14.1 (2014), pp. 583–586. ISSN: 1530-4388. DOI: 10.1109/TDMR.2013.2287699.

[32] Rudy Ferraro. "Development of Test Methods for the Qualification of Electronic Components and Systems Adapted to High-Energy Accelerator Radiation Environments." PhD thesis. Université Montpellier, 2019. URL: https://hal.archives-ouvertes.fr/tel-02879667 (visited on 08/09/2022).

[33] Robert L. Fleischer. "Cosmic Ray Interactions with Boron: A Possible Source of Soft Errors." In: *IEEE Transactions on Nuclear Science* 30.5 (1983), pp. 4013–4015. ISSN: 0018-9499. DOI: 10.1109/TNS.1983.4333061.

[34] Gilles Foucard. *NG-MEDIUM FPGA: Radiation Test Report at PSI*. CERN EDMS, July 30, 2018. URL: https://edms.cern.ch/document/2227145/1 (visited on 11/27/2022).

[35] Zoltan German-Sallo. "Signal Processing using FPGA Structures." In: *Procedia Technology* 12 (2014), pp. 112–118. ISSN: 22120173. DOI: 10.1016/j.protcy.2013.12.463.

[36] H. Hatano and M. Shibuya. "Total dose radiation effects on CMOS ring oscillators operating during irradiation." In: *IEEE Electron Device Letters* 4.12 (1983), pp. 435–437. ISSN: 0741-3106. DOI: 10.1109/EDL.1983.25793.

[37] Xiao Hu et al. "The 2020 Low-Power Computer Vision Challenge." In: *2021 IEEE 3rd International Conference on Artificial Intelligence Circuits and Systems (AICAS)*. IEEE, 2021, pp. 1–4. ISBN: 978-1-6654-1913-0. DOI: 10.1109/AICAS51828.2021.9458522.

[38] IQD Frequency Products Ltd. *CFPS-39 Crystal Clock Oscillator Specification*. Ed. by IQD Frequency Products Ltd. Nov. 2021. URL: https://www.iqdfrequencyproducts.com/products/details/cfps-39x.pdf (visited on 07/21/2022).

[39] JEDEC Solid State Technology Association. *1.8 V ± 0.15 V (Normal Range) and 1.2 V - 1.95 V (Wide Range) Power Supply Voltage and Interface Standard for Nonterminated Digital Integrated Circuits: JESD8-7*. 2006. URL: https://www.jedec.org/sites/default/files/docs/JESD8-7A.pdf (visited on 08/14/2022).

[40] JEDEC Solid State Technology Association. *2.5 V ± 0.2 V (Normal Range) and 1.8 V – 2.7 V (Wide Range) Power Supply Voltage and Interface Standard for Nonterminated Digital Integrated Circuits: JESD8-5*. 2007. URL: https://www.jedec.org/sites/default/files/docs/JESD8-5A-01.pdf (visited on 08/14/2022).

[41] JEDEC Solid State Technology Association. *Measurement and Reporting of Alpha Particle and Terrestrial Cosmic Ray-Induced Soft Errors in Semiconductor Devices*. 2021.

[42] JEDEC Solid State Technology Association. *Scalable Low-Voltage Signaling for 400 mV (SLVS-400)*. 2001. URL: https://www.jedec.org/sites/default/files/docs/jesd8-13.pdf (visited on 08/17/2022).

[43] JEDEC Solid State Technology Association. *Test Procedures for the Measurement of Single-event Effects in Semiconductor Devices from Heavy Ion Irradiation*. 2017.

[44] JEDEC Solid State Technology Association. *Test Standard for the Measurement of Proton Radiation Single Event Effects in Electronic Devices*. 2013.

[45] Fernanda Lima Kastensmidt et al. "TID in Flash-Based FPGA: Power Supply-Current Rise and Logic Function Mapping Effects in Propagation-Delay Degradation." In: *IEEE Transactions on Nuclear Science* 58.4 (2011), pp. 1927–1934. ISSN: 0018-9499. DOI: 10.1109/TNS.2011.2128881.

[46] Andrew M. Keller et al. "Dynamic SEU Sensitivity of Designs on Two 28-nm SRAM-Based FPGA Architectures." In: *IEEE Transactions on Nuclear Science* 65.1 (2018), pp. 280–287. ISSN: 0018-9499. DOI: 10.1109/TNS.2017.2772288.

[47] Glenn F. Knoll. *Radiation detection and measurement*. 3rd ed. New York, N.Y. and Chichester: Wiley, 2000. ISBN: 0471073385.

[48] Ian Kuon, Russell Tessier, and Jonathan Rose. "FPGA Architecture: Survey and Challenges." In: *Foundations and Trends® in Electronic Design Automation* 2.2 (2007), pp. 135–253. ISSN: 1551-3939. DOI: 10.1561/1000000005.

[49] Sakari Lahti et al. "Are We There Yet? A Study on the State of High-Level Synthesis." In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 38.5 (2019), pp. 898–911. ISSN: 0278-0070. DOI: 10.1109/TCAD.2018.2834439.

[50] Ewa Lopienska. "The CERN accelerator complex, layout in 2022. Complexe des accélérateurs du CERN en janvier 2022." In: (2022). URL: https://cds.cern.ch/record/2800984 (visited on 07/17/2022).

[51] J. Mekki et al. "CHARM: A Mixed Field Facility at CERN for Radiation Tests in Ground, Atmospheric, Space and Accelerator Representative Environments." In: *IEEE Transactions on Nuclear Science* 63.4 (2016), pp. 2106–2114. ISSN: 0018-9499. DOI: 10.1109/TNS.2016.2528289.

[52] P. Moreira, K. Kloukinas, and S. Bonacini. *E-link: A Radiation-Hard Low-Power Electrical Link for Chip-to-Chip Communication*. 2009. DOI: 10.5170/CERN-2009-006.422.

[53] T. R. Oldham and F. B. McLean. "Total ionizing dose effects in MOS oxides and devices." In: *IEEE Transactions on Nuclear Science* 50.3 (2003), pp. 483–499. ISSN: 0018-9499. DOI: 10.1109/TNS.2003.812927.

[54] Vlad-Mihai Placinta and Lucian N. Cojocariu. "Radiation Hardness Studies and Evaluation of SRAM-Based FPGAs for High Energy Physics Experiments." In: *Proceedings of Topical Workshop on Electronics for Particle Physics — PoS(TWEPP-17)*. Ed. by S. Veneziano et al. Trieste, Italy: Sissa Medialab, 2018, p. 085. DOI: 10.22323/1.313.0085.

[55] Python Software Foundation. *dataclasses — Data Classes: Python 3.10 documentation*. Ed. by Python Software Foundation. 2023. URL: https://docs.python.org/3.10/library/dataclasses.html (visited on 01/18/2023).

[56] Heather Quinn et al. "Using Benchmarks for Radiation Testing of Microprocessors and FPGAs." In: *IEEE Transactions on Nuclear Science* 62.6 (2015), pp. 2547–2554. ISSN: 0018-9499. DOI: 10.1109/TNS.2015.2498313.

[57] Rosula S. Reyes et al. "FPGA-Based Digital Signal Processing Trainer." In: *2009 WRI World Congress on Computer Science and Information Engineering*. IEEE, 2009, pp. 343–347. ISBN: 978-0-7695-3507-4. DOI: 10.1109/CSIE.2009.799.

[58] Nadia Rezzak et al. "In Orbit Programming and SEE characterization of the Microchip RT PolarFire® FPGA Fabric." In: *2021 21th European Conference on Radiation and Its Effects on Components and Systems (RADECS)*. 2021, pp. 1–6. DOI: 10.1109/RADECS53308.2021.9954504.

[59] Antonio Scialdone. "FPGA qualification for the LHC radiation environment." Masterthesis. Politecnico di Torino, 2021. URL: http://webthesis.biblio.polito.it/id/eprint/19117 (visited on 10/09/2022).

[60] Antonio Scialdone et al. "FPGA Qualification and Failure Rate Estimation Methodology for LHC Environments Using Benchmarks Test Circuits." In: *IEEE Transactions on Nuclear Science* (2022), p. 1. ISSN: 0018-9499. DOI: 10.1109/TNS.2022.3162037.

[61] Dongkyu Seo et al. "Total Ionizing Dose Effect on Ring Oscillator Frequency in 28-nm FD-SOI Technology." In: *IEEE Electron Device Letters* 39.11 (2018), pp. 1728–1731. ISSN: 0741-3106. DOI: 10.1109/LED.2018.2872345.

[62] M. R. Shaneyfelt et al. "Challenges in hardening technologies using shallow-trench isolation." In: *IEEE Transactions on Nuclear Science* 45.6 (1998), pp. 2584–2592. ISSN: 0018-9499. DOI: 10.1109/23.736501.

[63] Sudarshan Srinivasan et al. *High Performance Scalable FPGA Accelerator for Deep Neural Networks*. 2019. DOI: 10.48550/arXiv.1908.11809.

[64] Telecommunications Industry Association. *Electrical Characteristics of Low Voltage Differential Signaling (LVDS) Interface Circuits: TIA-644*. 2001.

[65] The MathWorks, Inc. *Digital Signal Processing Design for FPGAs and ASICs*. Ed. by The MathWorks, Inc. 2022. URL: https://ch.mathworks.com/help/dsphdl/gs/dsphdl-design-for-fpgas-and-asics.html (visited on 10/08/2022).

[66] Paul A. Tipler and Gene Mosca. *Physics for Scientists and Engineers*. Seventh edition. Berlin, Heidelberg: Springer Spektrum, 2015. ISBN: 978-3-642-54166-7.

[67] Georgios Tsiligiannis, Rudy Ferraro, and Salvatore Danzeca. *Radiation Test Report – Smartfusion2-M2S010 FLASH-based FPGA: PSI Radiation Test Report*. Ed. by CERN. CERN EDMS, Apr. 25, 2015. URL: https://edms.cern.ch/document/1607767/1 (visited on 11/27/2022).

[68] William Tsu et al. "HSRA." In: *Proceedings of the 1999 ACM/SIGDA seventh international symposium on Field programmable gate arrays - FPGA '99*. Ed. by Sinan Kaptanoglu and Steve Trimberger. New York, New York, USA: ACM Press, 1999, pp. 125–134. ISBN: 1581130880. DOI: 10.1145/296399.296442.

[69] Hendrik Woehrle and Frank Kirchner. "CAEMO - A Flexible and scalable high performance matrix algebra coprocessor for embedded reconfigurable computing systems." In: *Microprocessors and Microsystems* 56 (2018), pp. 47–63. ISSN: 01419331. DOI: 10.1016/j.micpro.2017.10.005.

[70] Xilinx, Inc. *ZCU104 Evaluation Board User Guide: UG1267*. Ed. by Xilinx, Inc. Oct. 9, 2018. URL: https://docs.xilinx.com/v/u/en-US/ug1267-zcu104-eval-bd (visited on 02/02/2023).

[71] Xilinx, Inc. *Zynq UltraScale+ MPSoC Data Sheet: Overview: DS891*. Ed. by Xilinx, Inc. Nov. 7, 2022. URL: https://docs.xilinx.com/v/u/en-US/ds891-zynq-ultrascale-plus-overview (visited on 12/23/2022).

[72] Xilinx, Inc. *Zynq UltraScale+ MPSoC Processing System v3.4 LogiCORE IP Product Guide: PS201*. Ed. by Xilinx, Inc. May 10, 2022. URL: https://docs.xilinx.com/r/en-US/pg201-zynq-ultrascale-plus-processing-system (visited on 02/02/2023).

[73] Xilinx, Inc. *Zynq UltraScale+ MPSoC ZCU104 Evaluation Kit*. Ed. by Xilinx, Inc. 2018. URL: https://www.xilinx.com/products/boards-and-kits/zcu104.html (visited on 02/02/2023).

[74] Xilinx, Inc. *Zynq UltraScale+ MPSoC: Software Developers Guide: UG1137*. Ed. by Xilinx, Inc. Jan. 5, 2021. URL: https://docs.xilinx.com/r/2020.2-English/ug1137-zynq-ultrascale-mpsoc-swdev (visited on 02/02/2023).

[75] Xiaoyu Yu et al. "A Data-Center FPGA Acceleration Platform for Convolutional Neural Networks." In: *2019 29th International Conference on Field Programmable Logic and Applications (FPL)*. IEEE, 2019, pp. 151–158. ISBN: 978-1-7281-4884-7. DOI: 10.1109/FPL.2019.00032.

A

GateMate Global Mesh Images

The images regarding the Global Mesh referenced in Chapter 3 are appended in the following order:

1. Clock and user-defined signal injection methods available for the GateMate;
2. Four ways to extract signals from the Global Mesh into the fabric.

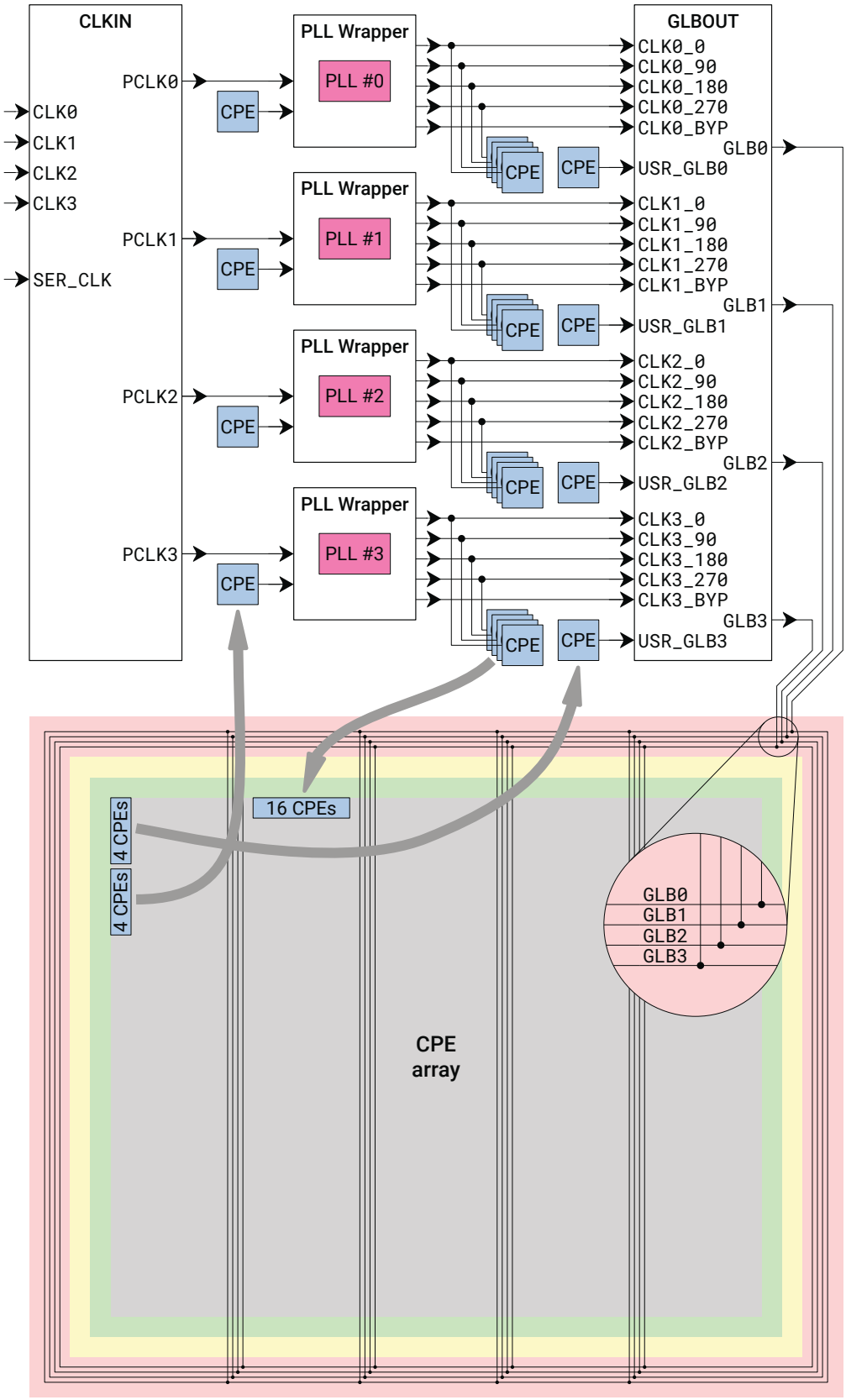


Figure A.1: Clock and user-defined signal injection methods available for the GateMate. This figure is adopted from [16] without any changes.

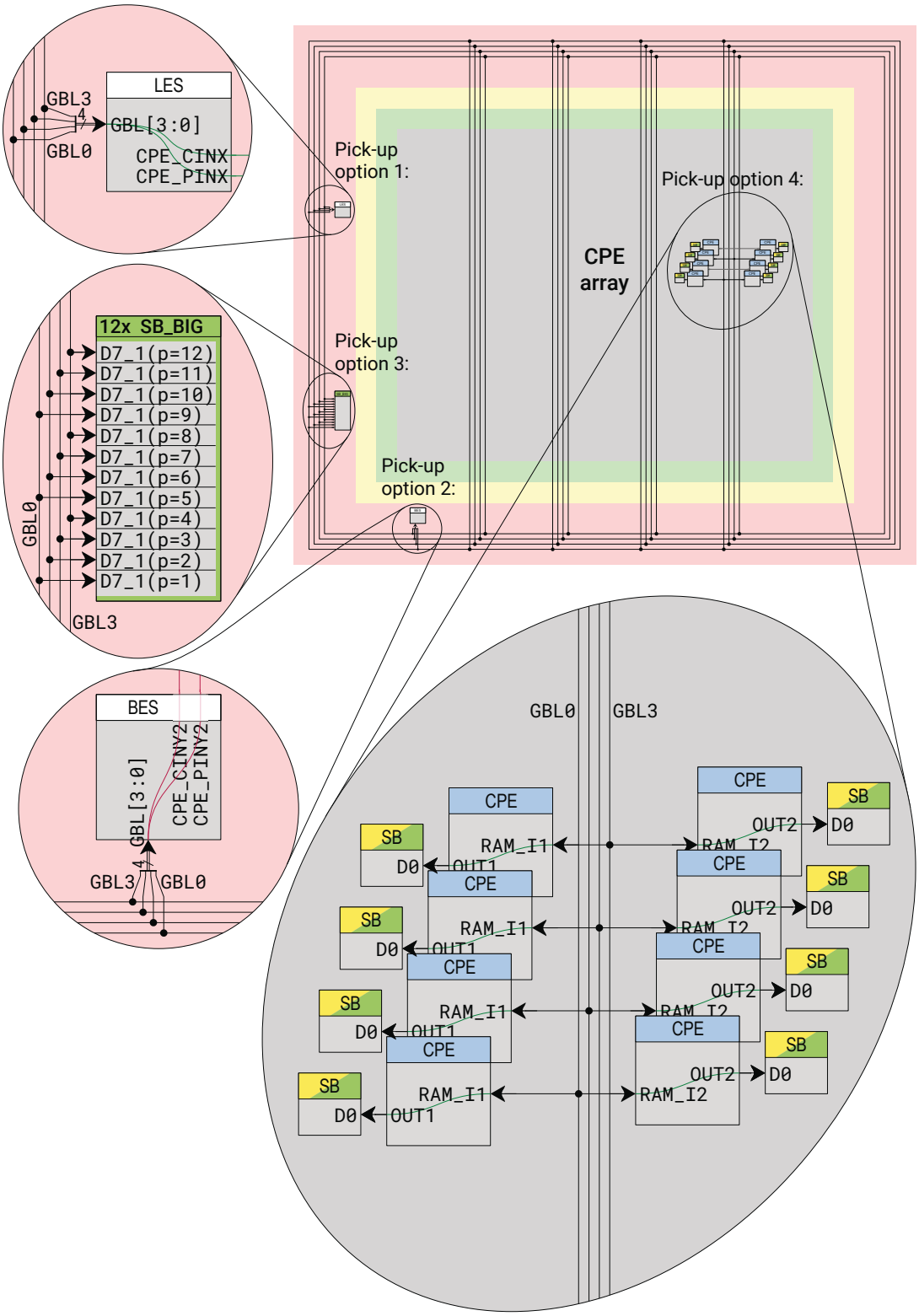


Figure A.2: Four ways to extract signals from the Global Mesh into the fabric. This figure is adopted from [16] without any changes.

B

Pin Assignments

Table B.1: I/O pin assignments on the GateMate SoM

Signal Name	Ball	Pin	Voltage	Comments
clk_in_n0	C11	IO_NB_B3	Vadj1	
clk_in_n1	M4	IO_WB_B1	V1p5	
clk_in_p0	D11	IO_NB_A3	Vadj1	
clk_in_p1	M3	IO_WB_A1	V1p5	
FmcClkM2c_c_n0	C12	IO_NB_B5	Vadj1	
FmcClkM2c_c_n1	A9	IO_NB_B0	Vadj1	
FmcClkM2c_c_p0	D12	IO_NB_A5	Vadj1	
FmcClkM2c_c_p1	B9	IO_NB_A0	Vadj1	
FmcDpC2m_n	V7	IO_SA_B3	Vadj1	
FmcDpC2m_p	U7	IO_SA_A3	Vadj1	
FmcDpM2c_n	U10	IO_SA_B8	Vadj1	
FmcDpM2c_p	V10	IO_SA_A8	Vadj1	
FmcGbtClkM2c_c_n0	U18	IO_SB_B0	Vadj1	
FmcGbtClkM2c_c_p0	U17	IO_SB_A0	Vadj1	
FmcLa_n00	V8	IO_SA_B5	Vadj1	
FmcLa_n01	T6	IO_SA_B1	Vadj1	
FmcLa_n02	P13	IO_SB_B7	Vadj1	
FmcLa_n03	R11	IO_SB_B4	Vadj1	
FmcLa_n04	T7	IO_SA_B2	Vadj1	
FmcLa_n05	T8	IO_SA_B4	Vadj1	
FmcLa_n06	P15	IO_SB_B6	Vadj1	
FmcLa_n07	R14	IO_SB_B5	Vadj1	
FmcLa_n08	T9	IO_SA_B6	Vadj1	
FmcLa_n09	T10	IO_SA_B7	Vadj1	
FmcLa_n10	R18	IO_SB_B1	Vadj1	
FmcLa_n11	P18	IO_SB_B2	Vadj1	
FmcLa_n12	M14	IO_EA_B0	Vadj1	
FmcLa_n13	N15	IO_SB_B8	Vadj1	
FmcLa_n14	M18	IO_SB_B3	Vadj1	
FmcLa_n15	L15	IO_EA_B1	Vadj1	

Table B.1 continued from previous page

Signal Name	Ball	Pin	Voltage	Comments
FmcLa_n16	K15	IO_EA_B3	Vadj1	
FmcLa_n17	G17	IO_EA_B7	Vadj1	
FmcLa_n18	K17	IO_EA_B2	Vadj1	
FmcLa_n19	J17	IO_EA_B4	Vadj1	
FmcLa_n20	J15	IO_EA_B5	Vadj1	
FmcLa_n21	A17	IO_EB_B5	Vadj1	
FmcLa_n22	C16	IO_EB_B6	Vadj1	
FmcLa_n23	F17	IO_EB_B0	Vadj1	
FmcLa_n24	H15	IO_EA_B6	Vadj1	
FmcLa_n25	A15	IO_EB_B8	Vadj1	
FmcLa_n26	C15	IO_EB_B7	Vadj1	
FmcLa_n27	D17	IO_EB_B2	Vadj1	
FmcLa_n28	G15	IO_EA_B8	Vadj1	
FmcLa_n29	A14	IO_NB_B7	Vadj1	
FmcLa_n30	C13	IO_NB_B6	Vadj1	
FmcLa_n31	C17	IO_EB_B4	Vadj1	
FmcLa_n32	A12	IO_NB_B4	Vadj1	
FmcLa_n33	C10	IO_NB_B1	Vadj1	
FmcLa_p00	U8	IO_SA_A5	Vadj1	
FmcLa_p01	R6	IO_SA_A1	Vadj1	
FmcLa_p02	P12	IO_SB_A7	Vadj1	CLK1
FmcLa_p03	P11	IO_SB_A4	Vadj1	
FmcLa_p04	R7	IO_SA_A2	Vadj1	
FmcLa_p05	R8	IO_SA_A4	Vadj1	
FmcLa_p06	P14	IO_SB_A6	Vadj1	CLK2
FmcLa_p07	R13	IO_SB_A5	Vadj1	CLK3
FmcLa_p08	R9	IO_SA_A6	Vadj1	
FmcLa_p09	R10	IO_SA_A7	Vadj1	
FmcLa_p10	R17	IO_SB_A1	Vadj1	
FmcLa_p11	P17	IO_SB_A2	Vadj1	
FmcLa_p12	M15	IO_EA_A0	Vadj1	
FmcLa_p13	N14	IO_SB_A8	Vadj1	CLK0
FmcLa_p14	M17	IO_SB_A3	Vadj1	
FmcLa_p15	L16	IO_EA_A1	Vadj1	
FmcLa_p16	K16	IO_EA_A3	Vadj1	
FmcLa_p17	G18	IO_EA_A7	Vadj1	
FmcLa_p18	K18	IO_EA_A2	Vadj1	
FmcLa_p19	J18	IO_EA_A4	Vadj1	
FmcLa_p20	J16	IO_EA_A5	Vadj1	
FmcLa_p21	B17	IO_EB_A5	Vadj1	
FmcLa_p22	D16	IO_EB_A6	Vadj1	
FmcLa_p23	F18	IO_EB_A0	Vadj1	
FmcLa_p24	H16	IO_EA_A6	Vadj1	
FmcLa_p25	B15	IO_EB_A8	Vadj1	
FmcLa_p26	D15	IO_EB_A7	Vadj1	
FmcLa_p27	D18	IO_EB_A2	Vadj1	

Table B.1 continued from previous page

Signal Name	Ball	Pin	Voltage	Comments
FmcLa_p28	G16	IO_EA_A8	Vadj1	
FmcLa_p29	B14	IO_NB_A7	Vadj1	
FmcLa_p30	D13	IO_NB_A6	Vadj1	
FmcLa_p31	C18	IO_EB_A4	Vadj1	
FmcLa_p32	B12	IO_NB_A4	Vadj1	
FmcLa_p33	D10	IO_NB_A1	Vadj1	
FmcPgC2m	D14	IO_NB_A8	Vadj1	
FmcPgM2c	C14	IO_NB_B8	Vadj1	
FmcPrsntM2cL	F15	IO_EB_B1	Vadj1	
FmcScl	E16	IO_EB_A3	Vadj1	
FmcSda	E15	IO_EB_B3	Vadj1	
GbtxClockDes0_bank_n	B2	IO_WC_B8	V1p8	
GbtxClockDes0_bank_p	B1	IO_WC_A8	V1p8	
GbtxClockDes0_gbt_c_n	B8	IO_NA_B7	V1p8	
GbtxClockDes0_gbt_c_p	A8	IO_NA_A7	V1p8	
GbtxConfigSelect	N2	IO_WB_B0	V1p5	
GbtxElinksDclk0_r_n	B6	IO_NA_B4	V1p8	
GbtxElinksDclk0_r_p	A6	IO_NA_A4	V1p8	
GbtxElinksDin_r_n0	G2	IO_WC_B0	V1p8	
GbtxElinksDin_r_n12	D9	IO_NA_B8	V1p8	
GbtxElinksDin_r_n16	D4	IO_WC_B6	V1p8	
GbtxElinksDin_r_n20	C4	IO_WC_B7	V1p8	
GbtxElinksDin_r_n24	D2	IO_WC_B5	V1p8	
GbtxElinksDin_r_n4	F4	IO_WC_B2	V1p8	
GbtxElinksDin_r_n8	D6	IO_NA_B3	V1p8	
GbtxElinksDin_r_p0	G1	IO_WC_A0	V1p8	
GbtxElinksDin_r_p12	C9	IO_NA_A8	V1p8	
GbtxElinksDin_r_p16	D3	IO_WC_A6	V1p8	
GbtxElinksDin_r_p20	C3	IO_WC_A7	V1p8	
GbtxElinksDin_r_p24	D1	IO_WC_A5	V1p8	
GbtxElinksDin_r_p4	F3	IO_WC_A2	V1p8	
GbtxElinksDin_r_p8	C6	IO_NA_A3	V1p8	
GbtxElinksDio_r_n0	G4	IO_WC_B1	V1p8	
GbtxElinksDio_r_n4	E2	IO_WC_B3	V1p8	
GbtxElinksDio_r_n8	D7	IO_NA_B5	V1p8	
GbtxElinksDio_r_p0	G3	IO_WC_A1	V1p8	
GbtxElinksDio_r_p4	E1	IO_WC_A3	V1p8	
GbtxElinksDio_r_p8	C7	IO_NA_A5	V1p8	
GbtxElinksDout_r_n16	D8	IO_NA_B6	V1p8	
GbtxElinksDout_r_n20	C5	IO_NA_B2	V1p8	
GbtxElinksDout_r_n24	E4	IO_WC_B4	V1p8	
GbtxElinksDout_r_p16	C8	IO_NA_A6	V1p8	
GbtxElinksDout_r_p20	B5	IO_NA_A2	V1p8	
GbtxElinksDout_r_p24	E3	IO_WC_A4	V1p8	
GbtxI2cScl_Fmc	L3	IO_WB_A2	V1p5	
GbtxI2cSda_Fmc	L4	IO_WB_B2	V1p5	

Table B.1 continued from previous page

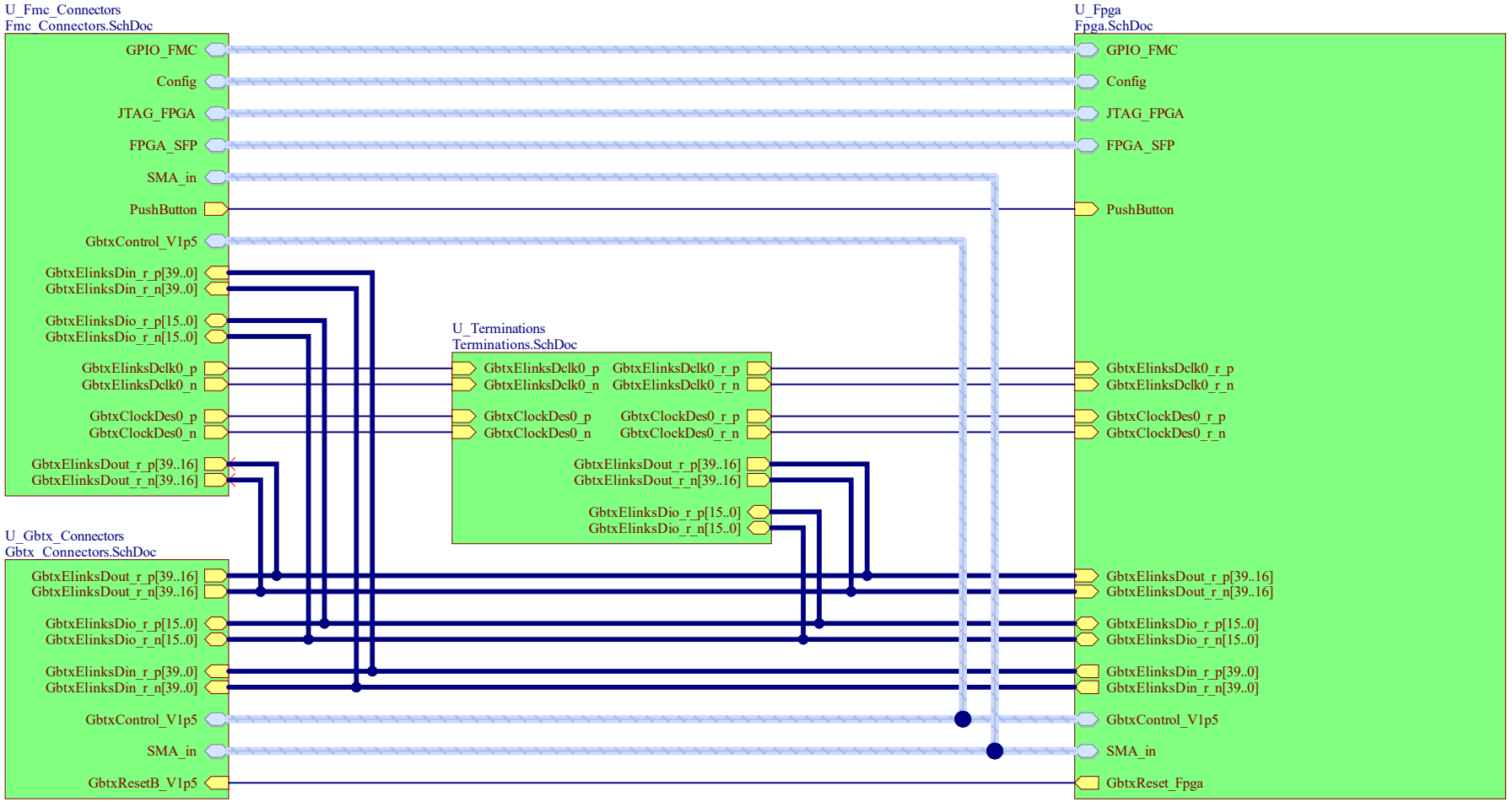
Signal Name	Ball	Pin	Voltage	Comments
GbtxReset_Fpga	N1	IO_WB_A0	V1p5	
GbtxRxDataValid	H2	IO_WB_B7	V1p5	
GbtxRxRdy	H1	IO_WB_A7	V1p5	
GbtxTxDataValid	L2	IO_WB_B3	V1p5	
GbtxTxRdy	L1	IO_WB_A3	V1p5	
GpioSwitch0	A4	IO_NA_A1	V1p8	
GpioSwitch1	B4	IO_NA_B1	V1p8	
LED0	A3	IO_NA_A0	V1p8	
LED1	B3	IO_NA_B0	V1p8	
PushButton	A11	IO_NB_B2	Vadj1	
trigger_in_n	K2	IO_WB_B5	V1p5	
trigger_in_p	K1	IO_WB_A5	V1p5	
UART_RX0	K3	IO_WB_A4	V1p5	
UART_RX1	J3	IO_WB_A6	V1p5	
UART_TX0	K4	IO_WB_B4	V1p5	
UART_TX1	J4	IO_WB_B6	V1p5	
user_in_n	H4	IO_WB_B8	V1p5	
user_in_p	H3	IO_WB_A8	V1p5	
	F16	IO_EB_A1	Vadj1	Not connected
	B11	IO_NB_A2	Vadj1	Not connected
	U5	IO_SA_A0	Vadj1	Not connected
	V5	IO_SA_B0	Vadj1	Not connected

C

Schematics

The schematics for the PCBs described in Chapter 4 are appended in the following order:

1. GateMate CRaTeBo SoM Mezzanine Board
    a) Top Level View
    b) FPGA and Miscellaneous Connections
    c) FPGA Power Supply
    d) Power Decoupling
    e) GBTx Connections
    f) FMC LPC Connections
    g) Signal Termination
2. GateMate FMC Adapter
3. GateMate FMC Adapter V2



FTG1 FTG2 FTG3

Project/Equipment	CHARM Radiation Tester Board (Mezzanine)		
Document	**GateMate CRaTeBo SoM Mezzanine Board**	Designer	R. Jung
		Drawn by	R. Jung 04/04/2022
		Check.by	-
		Last Mod.	- 20/09/2022
		File	GateMate_CHARM_Mezzanine.SchDoc
		Print Date	20/09/2022 17:04:30 Sheet 1 of 7

BE/CEM

European Organization for Nuclear Research
CH-1211 Genève 23 - Switzerland

Size A4 | Rev -

FPGA I/O ports

IC4A — GPIO BANK WB @ V1p5 — GateMate 1A1

IO_WB_B8	H4	user_in_n
IO_WB_A8	H3	user_in_p
IO_WB_B7	H2	GbtxRxDataValid
IO_WB_A7	H1	GbtxRxRdy
IO_WB_B6	J4	UART_TX1
IO_WB_A6	J3	UART_RX1
IO_WB_B5	K2	trigger_in_n
IO_WB_A5	K1	trigger_in_p
IO_WB_B4	K4	UART_TX0
IO_WB_A4	K3	UART_RX0
IO_WB_B3	L2	GbtxTxDataValid
IO_WB_A3	L1	GbtxTxRdy
IO_WB_B2	L4	GbtxI2cSda_Fmc
IO_WB_A2	L3	GbtxI2cScl_Fmc
IO_WB_B1	M4	clk_in_n1
IO_WB_A1	M3	clk_in_p1
IO_WB_B0	N2	GbtxConfigSelect
IO_WB_A0	N1	GbtxReset_Fpga

IC4C — GPIO BANK WC @ V1p8 — GateMate 1A1

IO_WC_B8	B2	GbtxClockDes0_bank_n
IO_WC_A8	B1	GbtxClockDes0_bank_p
IO_WC_B7	C4	GbtxElinksDin_r_n20
IO_WC_A7	C3	GbtxElinksDin_r_p20
IO_WC_B6	D4	GbtxElinksDin_r_n16
IO_WC_A6	D3	GbtxElinksDin_r_p16
IO_WC_B5	D2	GbtxElinksDin_r_n24
IO_WC_A5	D1	GbtxElinksDin_r_p24
IO_WC_B4	E4	GbtxElinksDout_r_n24
IO_WC_A4	E3	GbtxElinksDout_r_p24
IO_WC_B3	E2	GbtxElinksDio_r_n4
IO_WC_A3	E1	GbtxElinksDio_r_p4
IO_WC_B2	F4	GbtxElinksDin_r_n4
IO_WC_A2	F3	GbtxElinksDin_r_p4
IO_WC_B1	G4	GbtxElinksDio_r_n0
IO_WC_A1	G3	GbtxElinksDio_r_p0
IO_WC_B0	G2	GbtxElinksDin_r_n0
IO_WC_A0	G1	GbtxElinksDin_r_p0

IC4G — GPIO BANK NA @ V1p8 — GateMate 1A1

IO_NA_B8	D9	GbtxElinksDin_r_n12
IO_NA_A8	C9	GbtxElinksDin_r_p12
IO_NA_B7	B8	GbtxClockDes0_gbt_c_n
IO_NA_A7	A8	GbtxClockDes0_gbt_c_p
IO_NA_B6	D8	GbtxElinksDout_r_n16
IO_NA_A6	C8	GbtxElinksDout_r_p16
IO_NA_B5	D7	GbtxElinksDio_r_n8
IO_NA_A5	C7	GbtxElinksDio_r_p8
IO_NA_B4	B6	GbtxElinksDclk0_r_n
IO_NA_A4	A6	GbtxElinksDclk0_r_p
IO_NA_B3	D6	GbtxElinksDin_r_n8
IO_NA_A3	C6	GbtxElinksDin_r_p8
IO_NA_B2	C5	GbtxElinksDout_r_n20
IO_NA_A2	B5	GbtxElinksDout_r_p20
IO_NA_B1	B4	GpioSwitch1
IO_NA_A1	A4	GpioSwitch0
IO_NA_B0	B3	LED1
IO_NA_A0	A3	LED0

IC4I — GPIO BANK NB @ Vadj1 — GateMate 1A1

IO_NB_B8	C14	FmcPgM2c
IO_NB_A8	D14	FmcPgC2m
IO_NB_B7	A14	FmcLa_n29
IO_NB_A7	B14	FmcLa_p29
IO_NB_B6	C13	FmcLa_n30
IO_NB_A6	D13	FmcLa_p30
IO_NB_B5	C12	FmcClkM2c_c_n0
IO_NB_A5	D12	FmcClkM2c_c_p0
IO_NB_B4	A12	FmcLa_n32
IO_NB_A4	B12	FmcLa_p32
IO_NB_B3	C11	clk_in_n0
IO_NB_A3	D11	clk_in_p0
IO_NB_B2	A11	PushButton
IO_NB_A2	B11	
IO_NB_B1	C10	FmcLa_n33
IO_NB_A1	D10	FmcLa_p33
IO_NB_B0	A9	FmcClkM2c_c_n1
IO_NB_A0	B9	FmcClkM2c_c_p1

IC4M — GPIO BANK EA @ Vadj1 — GateMate 1A1

IO_EA_B8	G15	FmcLa_n28
IO_EA_A8	G16	FmcLa_p28
IO_EA_B7	G17	FmcLa_n17
IO_EA_A7	G18	FmcLa_p17
IO_EA_B6	H15	FmcLa_n24
IO_EA_A6	H16	FmcLa_p24
IO_EA_B5	J15	FmcLa_n20
IO_EA_A5	J16	FmcLa_p20
IO_EA_B4	J17	FmcLa_n19
IO_EA_A4	J18	FmcLa_p19
IO_EA_B3	K15	FmcLa_n16
IO_EA_A3	K16	FmcLa_p16
IO_EA_B2	K17	FmcLa_n18
IO_EA_A2	K18	FmcLa_p18
IO_EA_B1	L15	FmcLa_n15
IO_EA_A1	L16	FmcLa_p15
IO_EA_B0	M14	FmcLa_n12
IO_EA_A0	M15	FmcLa_p12

IC4K — GPIO BANK EB @ Vadj1 — GateMate 1A1

IO_EB_B8	A15	FmcLa_n25
IO_EB_A8	B15	FmcLa_p25
IO_EB_B7	C15	FmcLa_n26
IO_EB_A7	D15	FmcLa_p26
IO_EB_B6	C16	FmcLa_n22
IO_EB_A6	D16	FmcLa_p22
IO_EB_B5	A17	FmcLa_n21
IO_EB_A5	B17	FmcLa_p21
IO_EB_B4	C17	FmcLa_n31
IO_EB_A4	C18	FmcLa_p31
IO_EB_B3	E15	FmcSda
IO_EB_A3	E16	FmcScl
IO_EB_B2	D17	FmcLa_n27
IO_EB_A2	D18	FmcLa_p27
IO_EB_B1	F15	FmcPrsntM2cL
IO_EB_A1	F16	
IO_EB_B0	F17	FmcLa_n23
IO_EB_A0	F18	FmcLa_p23

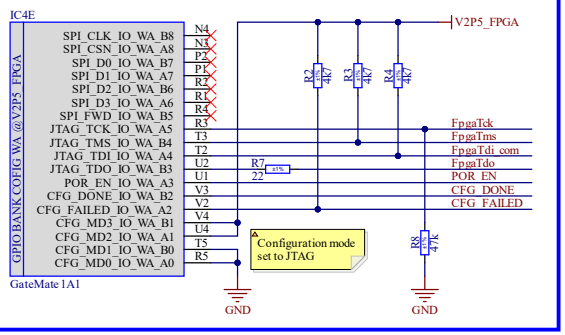
IC4Q — GPIO BANK SA @ Vadj1 — GateMate 1A1

IO_SA_B8	V10	FmcDpM2c_p
IO_SA_A8	U10	FmcDpM2c_n
IO_SA_B7	T10	FmcLa_n9
IO_SA_A7	R10	FmcLa_p9
IO_SA_B6	T9	FmcLa_n8
IO_SA_A6	R9	FmcLa_p8
IO_SA_B5	V8	FmcLa_n0
IO_SA_A5	U8	FmcLa_p0
IO_SA_B4	T8	FmcLa_n5
IO_SA_A4	R8	FmcLa_p5
IO_SA_B3	V7	FmcDpC2m_n
IO_SA_A3	U7	FmcDpC2m_p
IO_SA_B2	T7	FmcLa_n4
IO_SA_A2	R7	FmcLa_p4
IO_SA_B1	T6	FmcLa_n1
IO_SA_A1	R6	FmcLa_p1
IO_SA_B0	V5	
IO_SA_A0	U5	

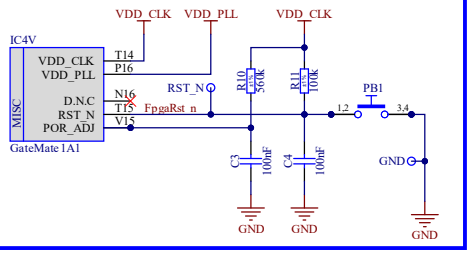
IC4O — GPIO BANK SB @ Vadj1 — GateMate 1A1

IO_SB_B8	N15	FmcLa_n13
IO_SB_A8	N14	FmcLa_p13
IO_SB_B7	P13	FmcLa_n2
IO_SB_A7	P12	FmcLa_p2
IO_SB_B6	P15	FmcLa_n6
IO_SB_A6	P14	FmcLa_p6
IO_SB_B5	R14	FmcLa_n7
IO_SB_A5	R13	FmcLa_p7
IO_SB_B4	R11	FmcLa_n3
IO_SB_A4	P11	FmcLa_p3
IO_SB_B3	M18	FmcLa_n14
IO_SB_A3	M17	FmcLa_p14
IO_SB_B2	P18	FmcLa_n11
IO_SB_A2	P17	FmcLa_p11
IO_SB_B1	R18	FmcLa_n10
IO_SB_A1	R17	FmcLa_p10
IO_SB_B0	U18	FmcGbtClkM2c_c_n0
IO_SB_A0	U17	FmcGbtClkM2c_c_p0

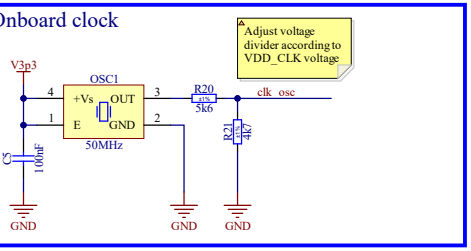
FPGA config ports



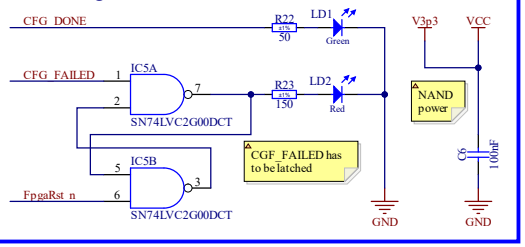
FPGA misc ports



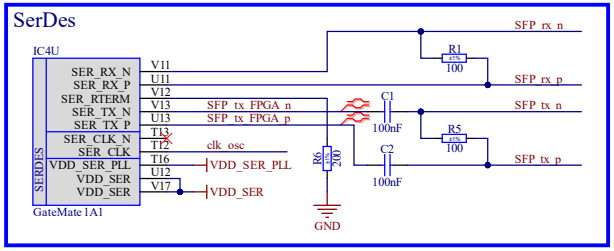
Onboard clock



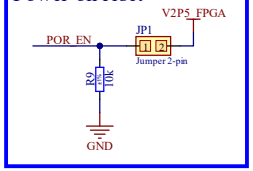
FPGA configuration status



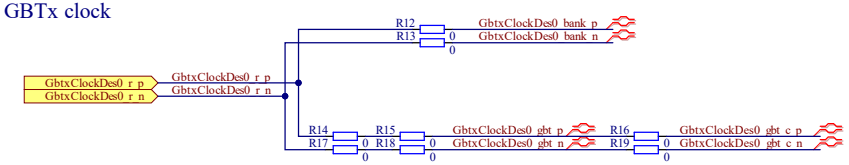
SerDes

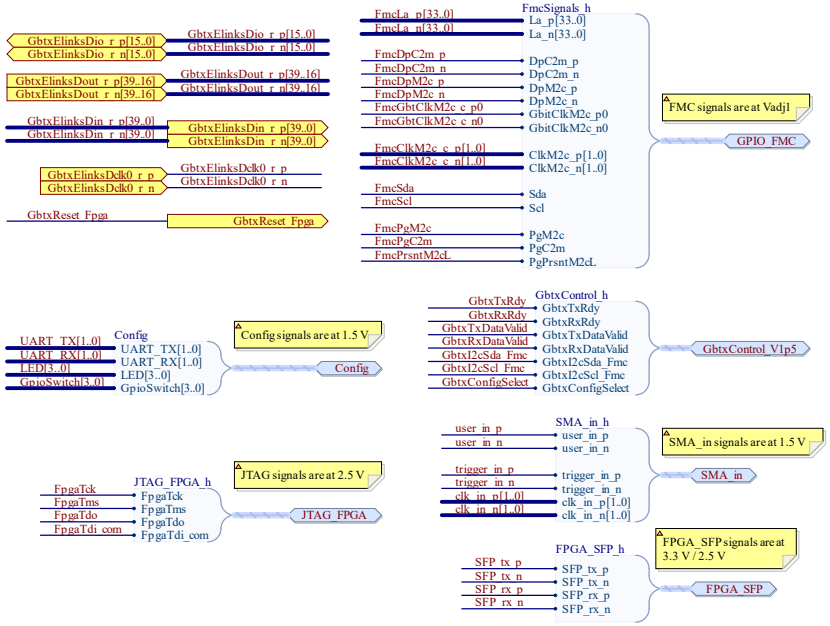


Power-on reset



GBTx clock



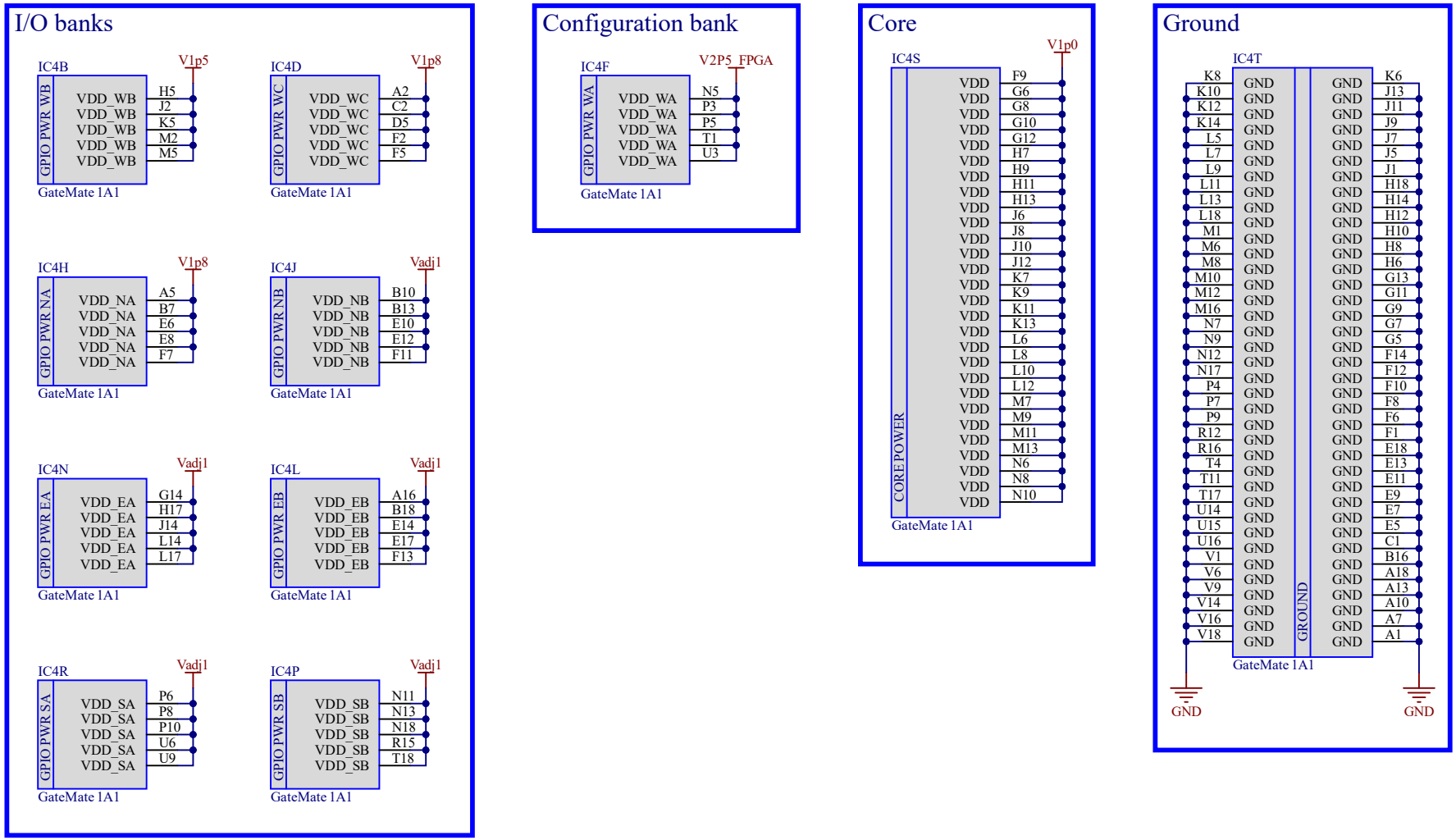


U_Fpga_Decoupling
Fpga_Decoupling.SchDoc

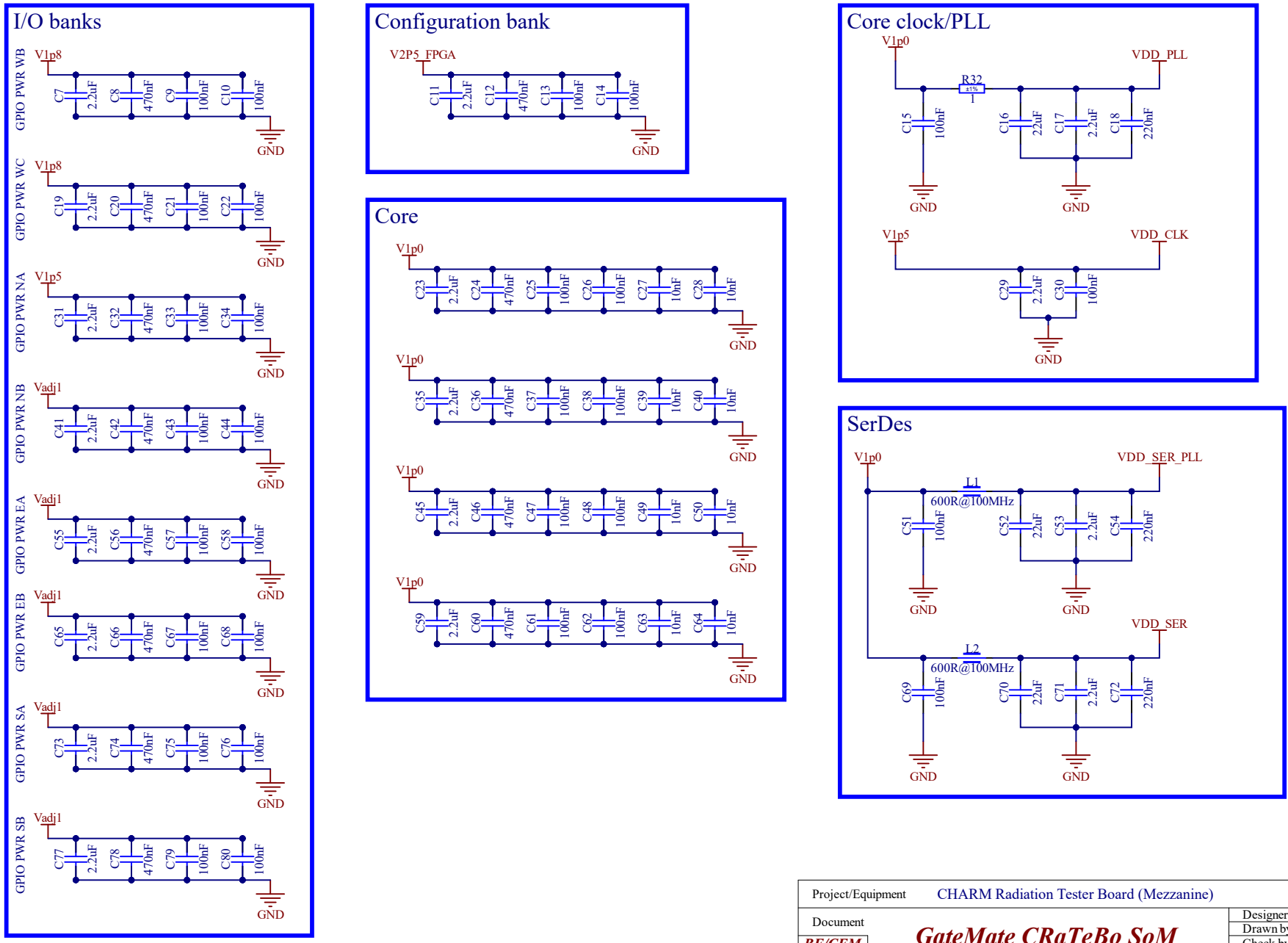
U_Fpga_Power
Fpga_Power.SchDoc

Project/Equipment	CHARM Radiation Tester Board (Mezzanine)
Document	GateMate CRaTeBo SoM Mezzanine Board
BE/CEM	
Designer	R. Jung
Drawn by	R. Jung 04/04/2022
Check.by	- -
Last Mod.	- 20/09/2022
File	Fpga.SchDoc
Print Date	20/09/2022 17:04:30
Sheet	2 of 7
Size	A3
Rev	-

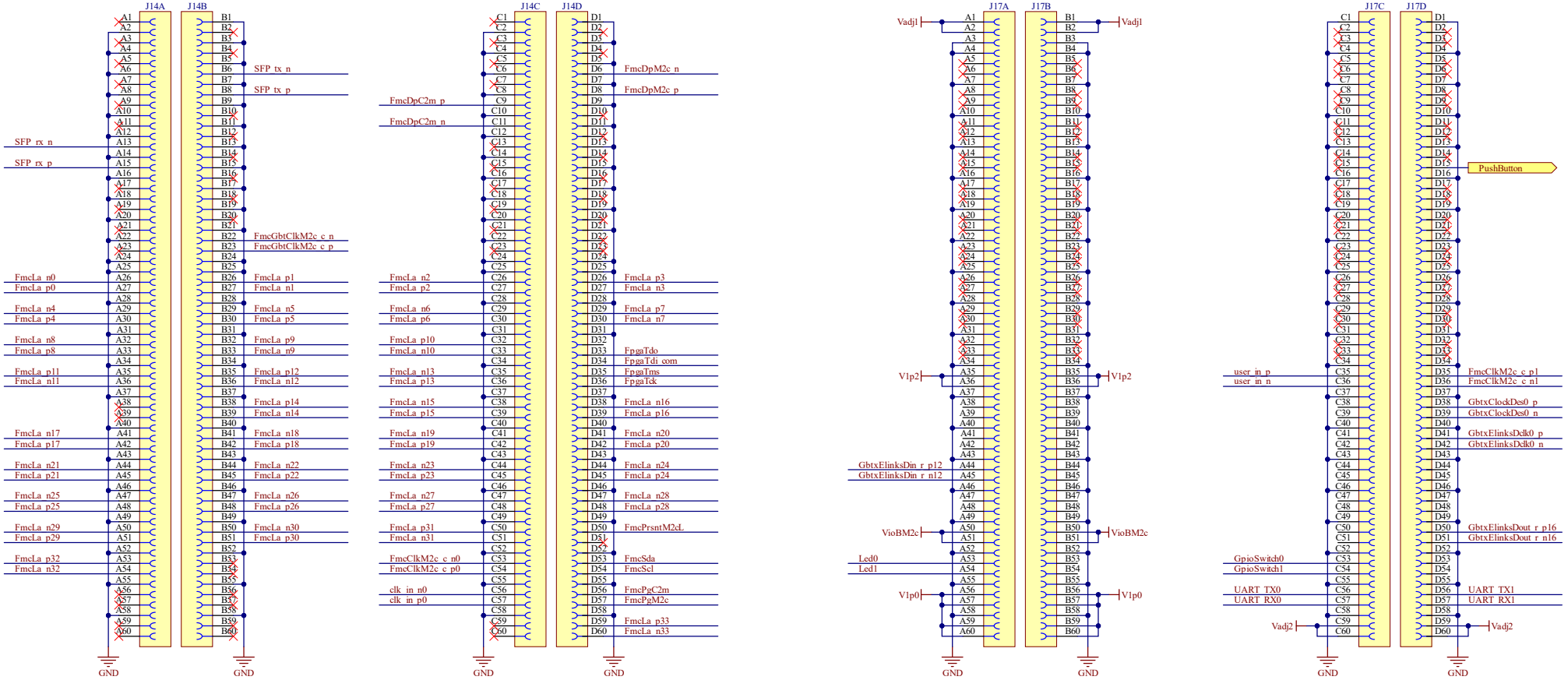
European Organization for Nuclear Research
CH-1211 Genève 23 - Switzerland

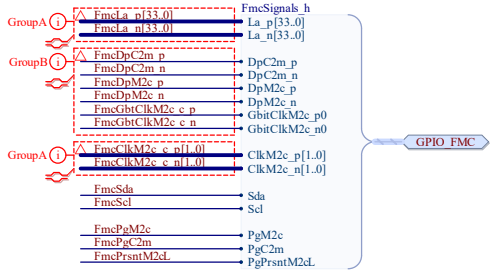


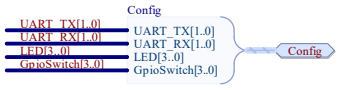
Project/Equipment	CHARM Radiation Tester Board (Mezzanine)		
Document	**GateMate CRaTeBo SoM Mezzanine Board**	Designer	R. Jung
		Drawn by	R. Jung 04/04/2022
		Check.by	-
		Last Mod.	- 20/09/2022
		File	Fpga_Power.SchDoc
		Print Date	20/09/2022 17:04:31 Sheet 3 of 7
BE/CEM	*European Organization for Nuclear Research* *CH-1211 Genève 23 - Switzerland*	Size	A4
		Rev	-

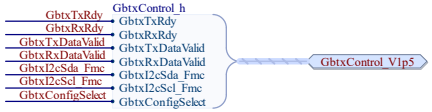


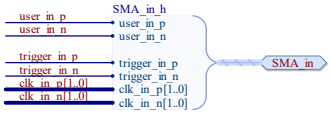
Project/Equipment	CHARM Radiation Tester Board (Mezzanine)		
Document	GateMate CRaTeBo SoM Mezzanine Board	Designer R. Jung	
		Drawn by R. Jung	04/04/2022
		Check.by -	-
		Last Mod. -	20/09/2022
		File Fpga_Decoupling.SchDoc	
		Print Date 20/09/2022 17:04:31	Sheet 4 of 7
BE/CEM	European Organization for Nuclear Research CH-1211 Genève 23 - Switzerland		Size A4 / Rev -











P1v0	max 10A - 8 pins
P1v2	max 3A - 4 pins
Vadj(GPIO)	max 3A - 4 pins
Vadj(GPIO)	max 3A - 4 pins
P1v8	max 3A - 4 pins
P2v5	max 3A - 4 pins
P3v3	max 3A - 4 pins

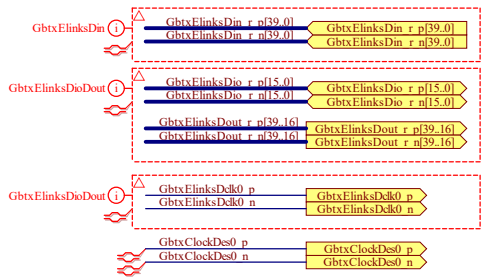
DIFFERENTIAL LANES OF FMC GROUP B HPC CONNECTOR
HIGH PRIORITY ROUTING - HIGH SPEED PARALLEL BUS (up to 12.7 Gbps)

* Diff. pair must have 100 Ohm of diff. impedance and each line taken alone 60 Ohm with respect to Gnd

* 2 Groups:
  - Group A: La[33:0], Ha[23:0], Hb[21:0], ClkM2c[0:1], ClkBidir[2:3]
  - Group B DpC2m[0:9], DpM2c[0:9], GbtClkM2c[0:1]

* Length matching constraints: (+/- 2.5 mm):
  - Same length of P/N
  - Same length of Group A and B lines

* Design guidelines for Vias:
  - Add symmetrical return Vias adjacent to every diff. Via pair
  - Pitch between two Vias as tigth as practical
  - Remove all Non-Functional Pads (NFPs)
  - Clearance hole as large as practical



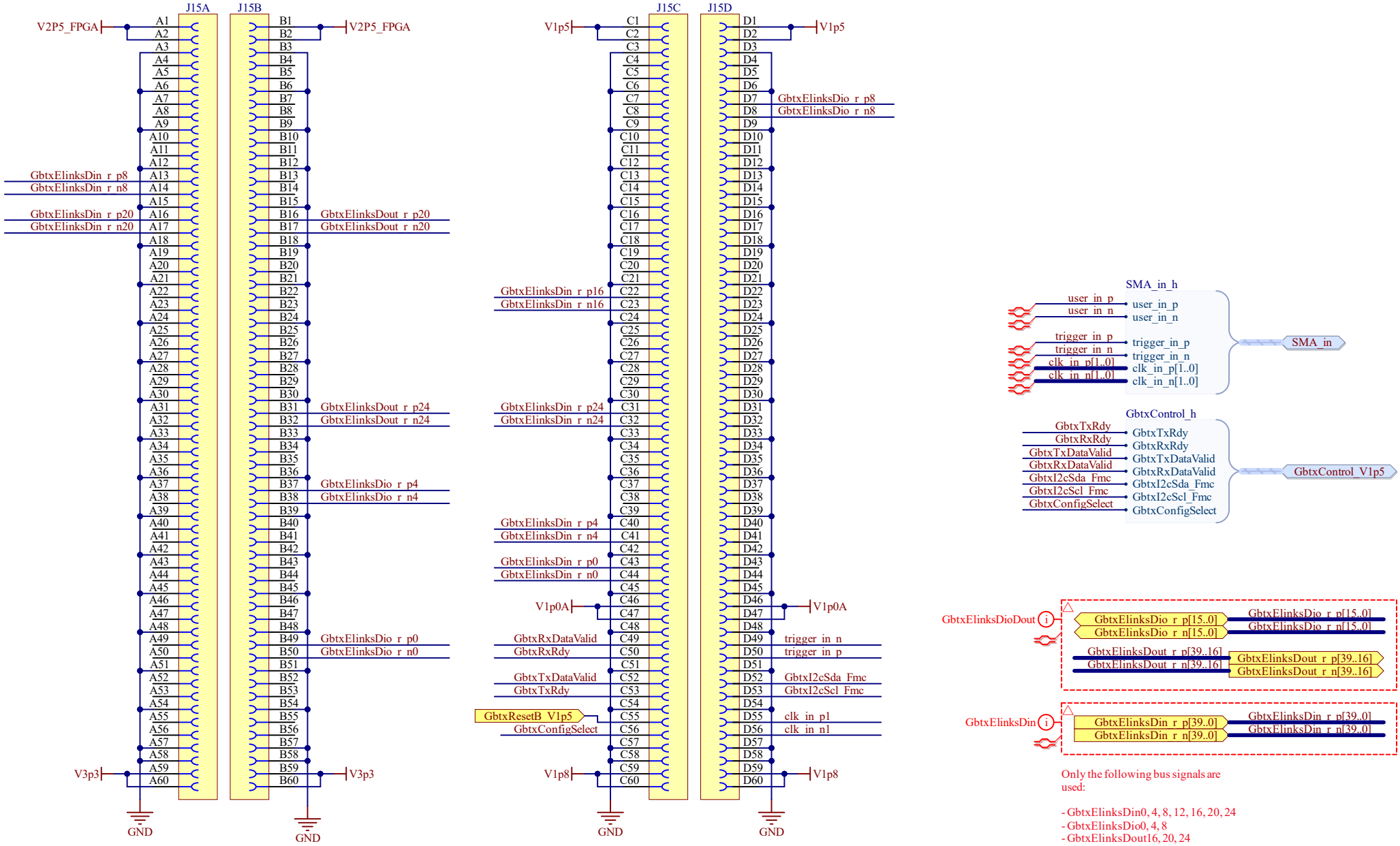
Only the following bus signals are used:

- GbtxElinksDin0, 4, 8, 12, 16, 20, 24
- GbtxElinksDio0, 4, 8
- GbtxElinksDout16, 20, 24

Project/Equipment	CHARM Radiation Tester Board (Mezzanine)
Document	**GateMate CRaTeBo SoM** *Mezzanine Board*
Designer	R. Jung
Drawn by	R. Jung 04/04/2022
Check.by	- -
Last Mod.	- 20/09/2022
File	Fmc_Connectors.SchDoc
Print Date	20/09/2022 17:04:31
Sheet	5 of 7
Size	A3
Rev	-

BE/CEM

European Organization for Nuclear Research
CH-1211 Genève 23 - Switzerland



Only the following bus signals are used:

- GbtxElinksDin0, 4, 8, 12, 16, 20, 24
- GbtxElinksDio0, 4, 8
- GbtxElinksDout16, 20, 24

----------1.4A/ pin----------

P1v0	max 10A -	8 pins
P1v2	max 3A -	4 pins
Vadj(GPIO)	max 3A -	4 pins
Vadj(GPIO)	max 3A -	4 pins
P1v8	max 3A -	4 pins
P2v5	max 3A -	4 pins
P3v3	max 3A -	4 pins

Project/Equipment: CHARM Radiation Tester Board (Mezzanine)

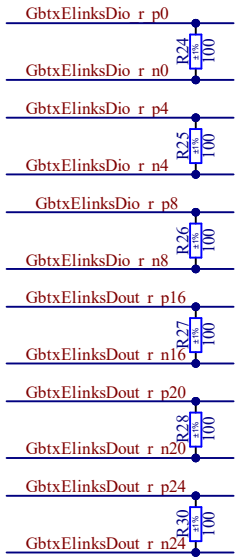
Document

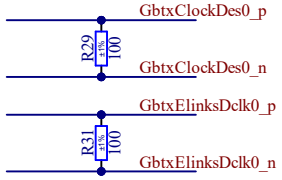
GateMate CRaTeBo SoM Mezzanine Board

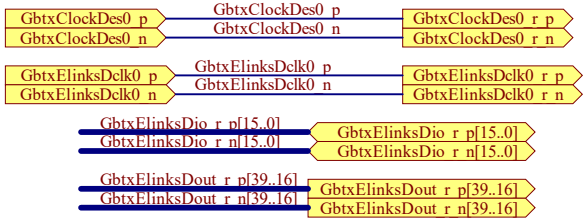
BE/CEM

European Organization for Nuclear Research
CH-1211 Genève 23 - Switzerland

Designer	R. Jung	
Drawn by	R. Jung	04/04/2022
Check.by	-	-
Last Mod.	-	20/09/2022
File	Gbtx_Connectors.SchDoc	
Print Date	20/09/2022 17:04:31	Sheet 6 of 7
Size	A4	Rev -

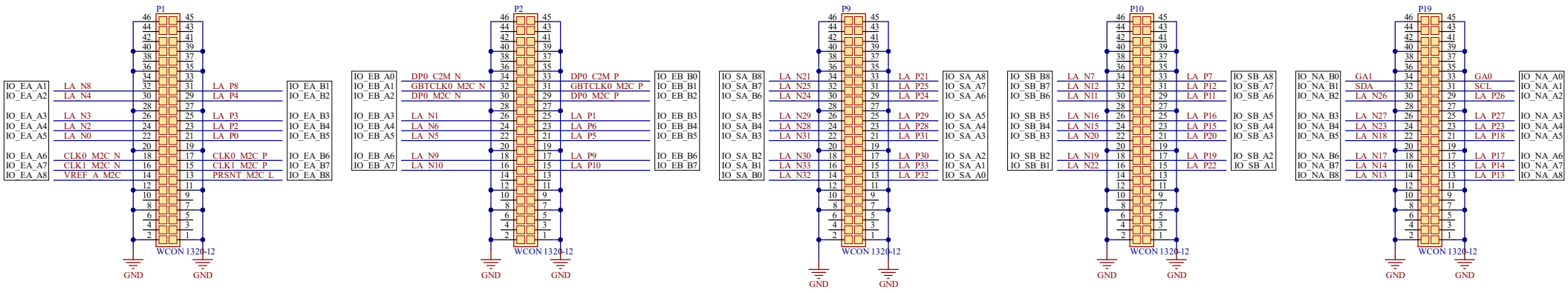




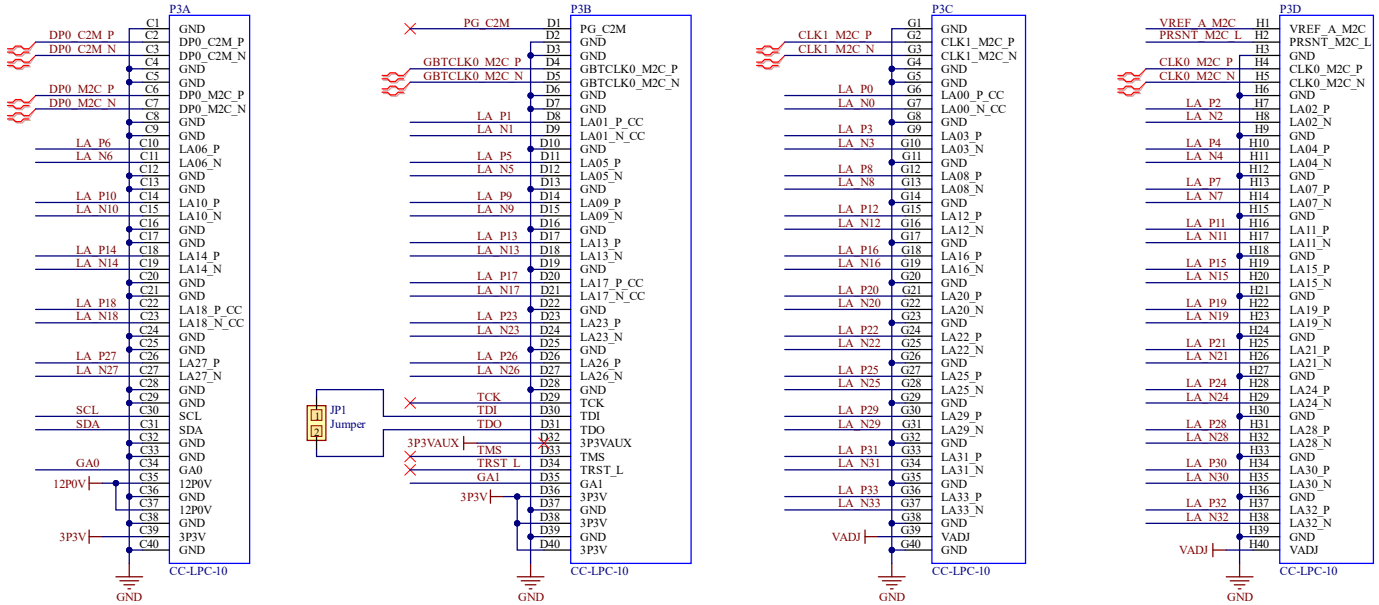


Project/Equipment	CHARM Radiation Tester Board (Mezzanine)		
Document	**GateMate CRaTeBo SoM Mezzanine Board**	Designer	R. Jung
		Drawn by	R. Jung 04/04/2022
		Check.by	-
		Last Mod.	- 20/09/2022
		File	Terminations.SchDoc
		Print Date	20/09/2022 17:04:31 Sheet 7 of 7
BE/CEM	*European Organization for Nuclear Research* *CH-1211 Genève 23 - Switzerland*	Size	A4
		Rev	-

GateMate Connectors



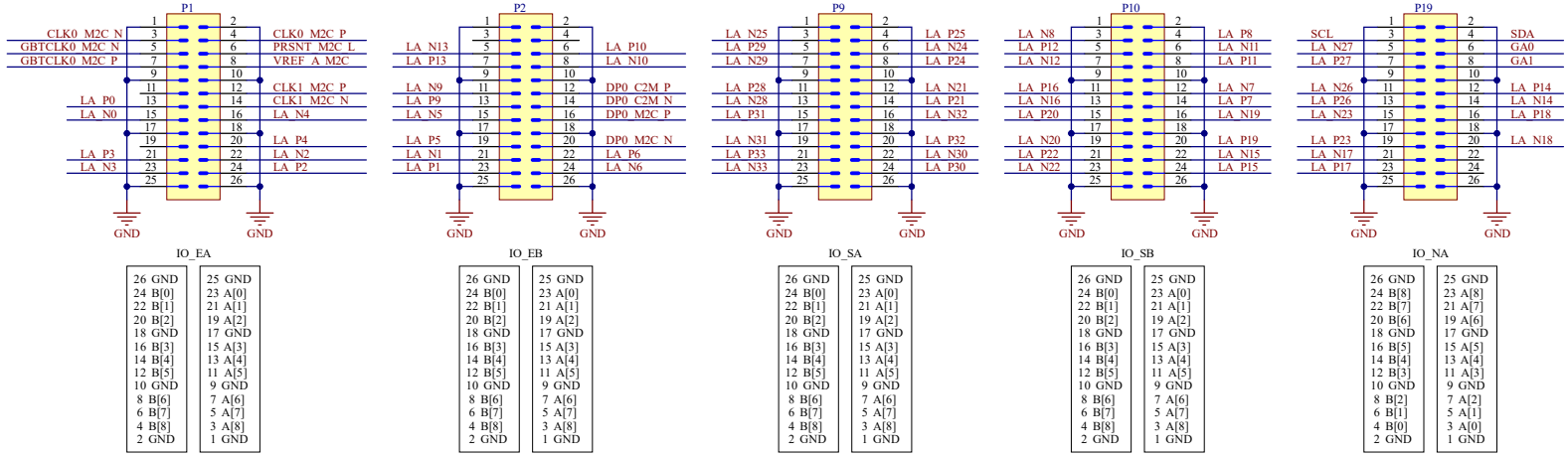
FMC Connector



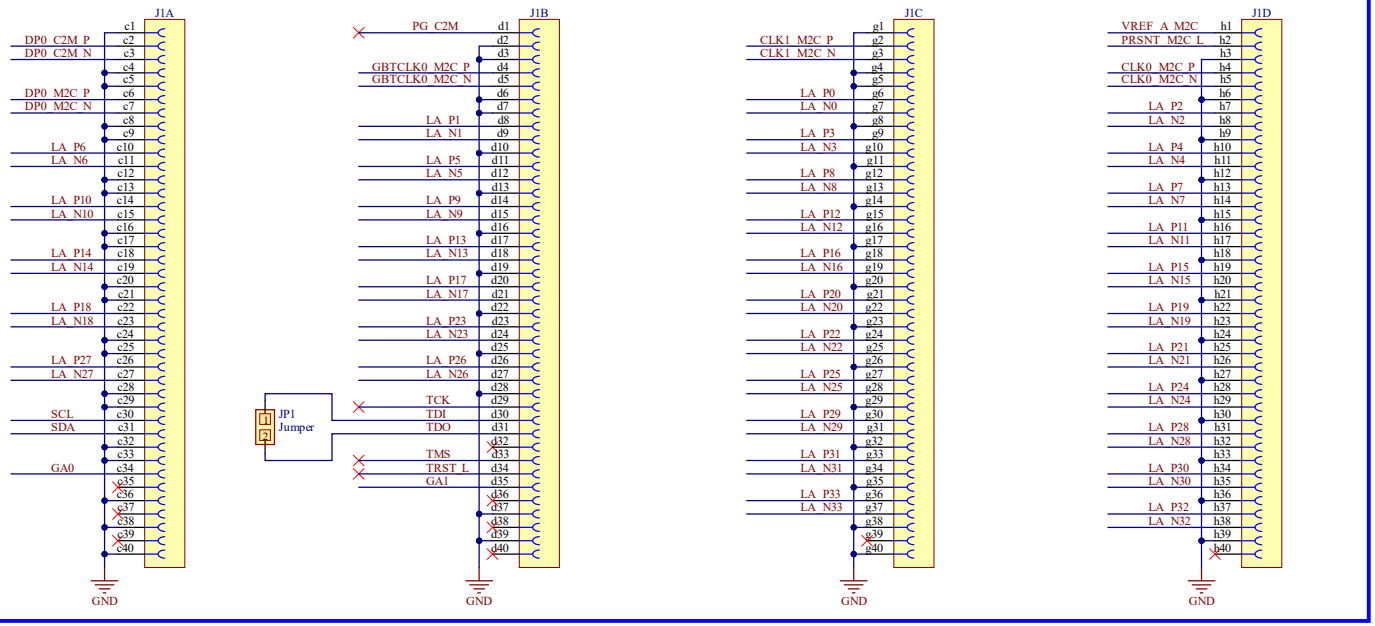
LA_P[33..0]
LA_N[33..0]

Project/Equipment	-		
Document	**GateMate FMC Adapter**	Designer R. Jung	
BE/CEM		Drawn by R. Jung	21/04/2022
		Check by -	-
		Last Mod. -	10/08/2022
		File GateMate_FMC_Adapter.SchDoc	
		Print Date 20/09/2022 17:09:00	Sheet 1 of 1
European Organization for Nuclear Research CH-1211 Genève 23 - Switzerland		Size A3	Rev -

GateMate Connectors



FMC Connector



LA_P[33..0]
LA_N[33..0]

FTG1 FTG2 FTG3

Project/Equipment	-	
Document	**GateMate FMC Adapter V2**	
Designer	R. Jung	
Drawn by	R. Jung	14/05/2022
Check.by	-	-
Last Mod.	-	10/08/2022
File	GateMate_FMC_Adapter_V2.SchDoc	
Print Date	20/09/2022 17:08:41	Sheet 1 of 1

BE/CEM

European Organization for Nuclear Research
CH-1211 Genève 23 - Switzerland

Size A3 | Rev -

D

Design Tables

BRAM Registers and Opcodes

Table D.1: BRAM IP core register description

Name	Address	Description
Control	0x00	Controls the core's functions
Address	0x04	Selects the BRAM cell to read/write to
Pattern	0x08	Selects the pattern to write
SEU	0x0C	Contains the total number of SEUs
MBU	0x10	Contains the total number of MBUs
Status	0x14	Shows the current state of the core

Table D.2: BRAM IP core control and pattern selection codes

Opcode	Control Register	Pattern Select Register
0x00	Clears the waiting flag	Zero pattern
0x01	Starts the writing process	One pattern
0x02	Starts the reading process	Checkerboard pattern
0x04	Resets the SEU and MBU counters	Inv. checkerboard pattern

Table D.3: BRAM IP core status codes

Status Code	Description
0x00	IP core not ready
0x01	IP core is idle
0x02	IP core is reading
0x04	IP core is writing

Flip-Flop Registers and Opcodes

Table D.4: Flip-flop IP core register description

Name	Address	Description
Control	0x00	Controls the core's functions
Address	0x04 $*(n+1)$	Reads the SEU counter from WSR n
Status	0x14	Shows the current state of the core
Length	0x48	Register to set the number of flip-flops per SR chain

Table D.5: Register map for the flip-flop core's control register

Unused	Pattern code	Opcode
31:4	3:2	1:0

Table D.6: Pattern selection codes for the flip-flop test IP core

Code	Selected Pattern
0x0	Zero pattern
0x1	One pattern
0x2	Checkerboard pattern

Table D.7: Opcodes for the flip-flop test IP core

Opcode	Operation
0x1	Starts the testing procedure
0x2	Stops the testing procedure
0x3	Resets the SEU and MBU counters

Table D.8: Flip-flop IP core status codes

Status Code	Description
0x00	IP core not ready
0x01	IP core is idle
0x02	IP core is resetting

Phase-Locked Loop Registers and Opcodes

Table D.9: PLL IP core register description

Name	Address	Description
Control	0x00	Controls the core's functions
Reference pulses	0x04 $*(n+1)$	Reads the SEU counter from PLL n
Status	0x44	Shows the current state of the core

Table D.10: Opcodes for the PLL test IP core

Opcode	Operation
0x1	Starts the testing procedure
0x2	Resets the SEU counter register
0x3	Stops the testing procedure

Table D.11: PLL IP core status codes

Status Code	Description
0x1	IP core is idle
0x2	IP core is working
0x4	IP core is stopped

Benchmark Registers and Opcodes

Table D.12: Opcodes for the benchmark test IP core

Opcode	Operation
0x1	Starts the testing procedure
0x3	Resets the core
0x4	Stops the testing procedure
0x5	Reads the FIFO counter
0x6	Clear the states

Table D.13: Benchmark IP core register description

Name	Address	Description
Control	0x00	Controls the core's functions
SEUs	0x04	Number of SEUs counted
Faulty value	0x08	Contains the output from the faulty B13 circuit
Golden value	0x0C	Contains the output from the golden reference B13 circuit
IP Core status	0x10	Shows the current state of the core
FIFO status	0x14	Shows the current state of the FIFO
Wait cycles	0x18	Set the cycles to wait before processing the error signal

Table D.14: Benchmark IP faulty value register

Faulty B13 Address	Faulty B13 Output
31:10	9:0

Table D.15: Benchmark IP golden reference register

Unused	Golden Reference Output	LSFR Output
31:20	19:10	9:0

Table D.16: Benchmark IP core status codes

Status Code	Description
0x1	IP core is idle
0x2	IP core is working
0x3	IP core is reading
0x4	IP core is waiting
0x5	IP core is resetting

TID Design Registers and Opcodes

Table D.17: TID IP core register description

Name	Address	Description
Control	0x00	Controls the core's functions
Reference pulses	0x04	Contains the number of pulses from the reference clock
External pulses	0x08	Contains the number of pulses from the external clock
Wait period	0x0C	Set the number of cycles to measure the clocks
Status	0x10	Shows the current state of the core
Ring address	0x14	Sets the address of the ring oscillator to read
Timeout	0x18	Set the read timeout to interrupt the measuring cycle

Table D.18: Opcodes for the TID test IP core

Opcode	Operation
0x1	Starts the testing procedure
0x2	Load the wait and timeout values into the core
0x3	Clear the read state
0x4	Clear the timeout state
0x5	Clear the load state

Table D.19: TID IP core status codes

Status Code	Description
0x1	IP core is idle
0x2	IP core is waiting
0x3	IP core is loading from the registers
0x4	IP core is counting
0x5	IP core is reading
0x6	IP core is timed out

Eidesstattliche Erklärung

Hiermit versichere ich an Eides statt, dass die von mir vorgelegte Prüfungsleistung selbstständig und ohne unzulässige fremde Hilfe erstellt worden ist. Alle verwendeten Quellen sind in der Arbeit so aufgeführt, dass Art und Umfang der Verwendung nachvollziehbar sind.

Kreuztal, 16.3.2023

Unterschrift