2014 IEEE 22nd Annual International Symposium on Field-Programmable Custom Computing Machines

A Self-Adaptive SEU Mitigation System for FPGAs with an Internal *Block RAM* Radiation Particle Sensor

Robért Glein*, Bernhard Schmidt†, Florian Rittner‡, Jürgen Teich† and Daniel Ziener†
*Information Technology (Communication Electronics),
Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU), Germany, Erlangen
Email: robert.glein@fau.de
†Hardware/Software Co-Design
Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU), Germany, Erlangen
‡RF and Microwave Design Department
Fraunhofer Institute for Integrated Circuits, Germany, Erlangen

***Abstract*—In this paper, we propose a self-adaptive FPGA-based, partially reconfigurable system for space missions in order to mitigate *Single Event Upsets* in the FPGA configuration and fabric. Dynamic reconfiguration is used here for an on-demand replication of modules in dependence of current and changing radiation levels. More precisely, the idea is to trigger a redundancy scheme such as *Dual Modular Redundancy* or *Triple Modular Redundancy* in response to a continuously monitored *Single Event Upset* rate measured inside the on-chip memories itself, e.g., any subset (even used) internal *Block RAMs*. Depending on the current radiation level, the minimal number of replicas is determined at run-time under the constraint that a required *Safety Integrity Level* for a module is ensured and configured accordingly. For signal processing applications it is shown that this autonomous adaption to the different solar conditions realizes a resource efficient mitigation. In our case study, we show that it is possible to triplicate the data throughput at the *Solar Maximum* condition (no flares) compared to a *Triple Modular Redundancy* implementation of a single module. We also show the decreasing *Probability of Failures Per Hour* by 2×10^4 at flare-enhanced conditions compared with a non-redundant system.**

Our work is a part of the *In-Orbit Verification* of the *Heinrich Hertz* communication satellite.

I. Introduction

Today's FPGAs play a major role in many digital signal processing application areas like telecommunication, radar, video, audio and image processing. Due to their success, FPGAs are also used in space applications, like in the *Fraunhofer On-Board Processor* of the *Heinrich Hertz* communication satellite which is planned to be launched into a *Geostationary Earth Orbit* (GEO) in 2018 followed by an operation time of 15 years [1]. Here, FPGAs offer significant competitive advantages with respect to ASICs. Since FPGAs are in-field reconfigurable, it is possible to change the hardware the FPGA implements after the satellite has launched into orbit. We chose to leverage the FPGAs capability to be reconfigured for various tasks. This saves space, reduces weight and enables the use of future communications protocols.

Unfortunately, the reconfiguration capability comes at a price. SRAM-based FPGAs are highly susceptible to so-called *Single Event Effects* (SEEs) when operating in a harsh environments like space. This effects are the primary short-term reliability concerns in space systems [2]. SEEs can be classified in two main groups, namely temporary and destructive effects. Fig. 1 shows sources of temporary effects which are subdivided into *Single Event Functional Interrupts* (SEFIs), *Single Event Transients* (SETs) and *Single Event Upsets* (SEUs). In this paper, we will focus on SEUs only, because SEFIs are at least three orders of magnitude less then SEUs for the used device. In particular, gamma rays and particles like *Electrons/Positrons*, *Neutrons*, *Protons* (P) and *Heavy Ions* (HI) known as solar and galactic cosmic radiation may alter single or multiple bits in the data of the configuration memory, flip flops or in the content of the embedded memories. The magnetic field of the earth protects terrestrial circuits from the most of the solar and galactic cosmic radiation but the advances in manufacturing process technology to ever smaller scales makes SEUs an increasing concern for ground level applications as well.

For space applications, a high system reliability has to be guaranteed which are commonly specified by the *Safety Integrity Level* (SIL) defined by the IEC 61508 standard [3]. To achieve such high reliabilities, FPGA manufacturer offer special space-grade FPGAs like the *Xilinx Virtex-5QV* FPGA which are far less susceptible to SEUs than conventional FPGAs [4]. Nonetheless, SEUs still occur and, therefore, additional mitigation techniques have to be applied to meet a target SIL. Typically, redundancy techniques, such as *Dual Modular Redundancy* (DMR) or *Triple Modular Redundancy* (TMR) [5], combined with *Scrubbing* [6]–[8] may be applied to such systems. However, DMR and TMR are known to cause an excessive overhead.

To determine the amount of redundancy required to reach a certain SIL, usually the worst-case radiation scenario has to be considered. However, since the intensity of cosmic rays is not constant but may vary over several magnitudes depending on the solar activity, this worst-case radiation protection is far too expensive, if redundant FPGA resources are allocated over the whole mission time. As a remedy for such inefficiency, we propose *a self-adaptive system which monitors the current SEU rate and exploits the partial reconfiguration of FPGAs to implement redundancy such as TMR on demand and in an autonomous way.*

As radiation particle sensor, we propose to utilize the embedded *Block RAM* (BRAM) of the FPGA itself. This determines the current radiation level by detecting the injected upsets in the stored BRAM data via *Error-Correcting Code* (ECC). We use a space-grade FPGA like the *Virtex-5QV* because: It is robust against *Total Ionizing Dose* (TID) greater than 1 Mrad (Si), immune against destructive SEEs and radiation-hardened against temporary SEEs. Moreover, SEFI and configuration SEU rates are at minimum three

978-1-4799-5111-6/14 $31.00 © 2014 IEEE
DOI 10.1109/FCCM.2014.79

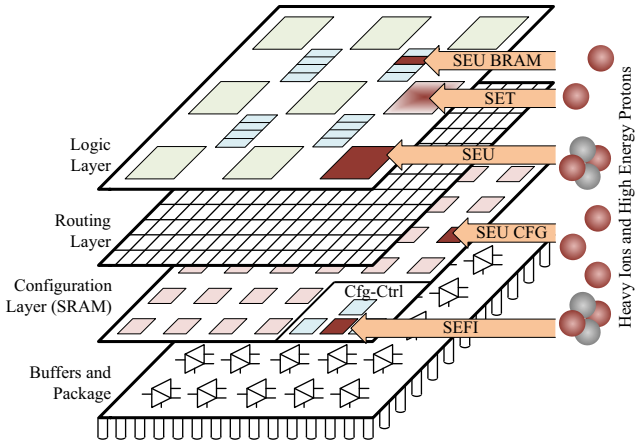

Figure 1: Temporary SEEs in an SRAM-based FPGA

orders of magnitude less than the BRAM SEU rate (see Section IV). This characteristic enables the BRAM as a reliable particle sensor.

The advantages of using BRAMs as radiation sensor are: High sensitivity to radiation, no additional hardware is needed, the radiation is measured at the point of interest and it is scalable by means of BRAM primitive count. Even for such a radiation-sensitive component, it is a challenge to achieve a desired SEU resolution of the BRAM sensor.

The remainder of this paper is structured as follows: Section II gives an overview of our system. Section III reviews the related work. Section IV points out the expected SEU rates for a Xilinx *Virtex-5QV* FPGA in a GEO and explains the mathematical background to determine the level of redundancy for a given SEU rate to ensure a target *Probability of Failures per Hour* (PFH), respectively SIL. Section V explains details of our system implementation, and Section VI finally applies our radiation results and self-adaptive redundancy control to a *Quadrature Phase-Shift Keying* (QPSK) demodulator application.

II. System Overview

As depicted in Fig. 2, our FPGA-based system consists of two subsystems: a) a *BRAM Sensor Subsystem* which utilizes embedded BRAMs to estimate the current SEU rate of the configuration memory, and b) an *Adaptive Subsystem* with a partially reconfigurable area which hosts the modules of the implemented application whereat the introduced redundancy level is controlled according to the current *Estimated Configuration Memory SEU Rate*.

The *BRAM Sensor Subsystem*, as show in Fig. 2, consists of a) at least one *BRAM Fault Detector* (BFD) and b) a *Fault Management Unit* (FMU). Multiple BFDs may be instantiated to improve the estimation of the current SEU rate. The BFD consists of a *BRAM Scrubber* which continuously reads out and checks the content of one embedded BRAM block. Moreover, the *BRAM Scrubber* contains an address counter to cyclically check each data word at the output port of the BRAM. Via the ECC parity bits, the *BRAM Scrubber* is able to immediately correct single bit errors and detect double bit errors whereat each detected single and double bit error are accumulated separately in counters of the *Fault Memory*. The counter values of each *Fault Memory* are accessed by

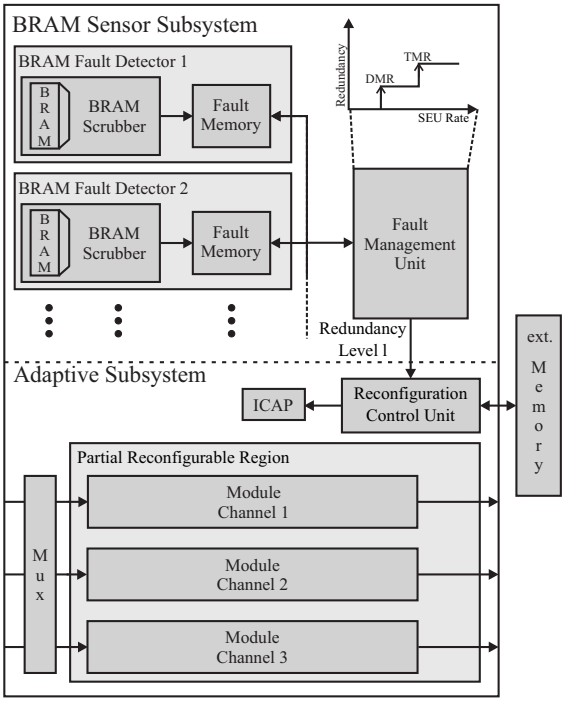

Figure 2: An FPGA-based self-adaptive autonomous SEU mitigation system consisting of a *BRAM Sensor Subsystem* (top) and an *Adaptive Subsystem* (bottom)

the FMU through a proprietary bus system to calculate the current SEU rate μ_{BRAM} of the embedded BRAM. On the basis of μ_{BRAM}, the FMU estimates the SEU rate μ_{CFG} of the configuration memory and determines the level of redundancy as a function of μ_{CFG} and a target PFH value which might be specified by a required SIL (see Section IV). In general, the PFH value also depends on the size of the implemented design which is commonly measured by the number of used configuration bits.

The required level of redundancy is then signaled to the input port of the *Reconfiguration Control Unit* (RCU) which belongs to the *Adaptive Subsystem* and controls the number of replicas via *Internal Configuration Access Port* (ICAP) by loading partial bitstreams of module replicas from an external memory into the configuration memory. In Fig. 2, an example *Adaptive Subsystem* configuration is shown consisting of three data channels with Channel 1 having the highest priority. Channel 2 and 3 can be switched off, if the resources are needed to replicate modules of Channel 1. In the shown scenario, no replicas and, therefore, no voters are assumed in case of a low SEU rate μ_{CFG}. The data of all three data channels are processed in parallel to reach the maximum achievable throughput and area utilization of the system. Furthermore, it is assumed that the configuration memory itself possesses a configuration memory scrubber that continuously corrects any configuration memory upsets via ICAP to prevent the accumulation of SEUs. This is not depicted in Fig. 2.

In our system, the output of redundant modules are used to mask or conceal errors induced by SEUs. On the contrary, the configuration memory scrubber actually corrects the

SEUs in the configuration memory which usually demands some SEU detection and correction time. In Section VI, it will be shown that this combination of redundancy technique and configuration memory scrubbing is necessary to guarantee a target SIL.

III. Related Work

SEU fault mitigation techniques for SRAM-based FPGAs techniques can be categorized into module redundancy techniques such as TMR [5], [9], [10] and techniques that use *Scrubbing* of the FPGA configuration memory [6]–[8]. Also, the combination of both techniques has been shown to be able to increase the reliability of the FPGA modules significantly [11].

FPGAs and partial reconfiguration have been proposed for numerous adaptive system applications. Furthermore work has been reported in which partial reconfiguration is used for adaptation to changing operating conditions to enhance the reliability of a system implementation. In [10], the authors present a self-adaptive FPGA system, which takes aging effects of the FPGA into account and which is updated at run-time. In [12], the authors present an adaptive FPGA-based system which controls the introduced redundancy of the system. Like in our following approach, the authors apply DMR and TMR in combination with scrubbing to their system. This system adaptively switches between DMR and TMR implementations in dependence of the frequency of detected errors. An approach for system and component level mitigation with different granularities is proposed in [13]. In [2], an adaptive reconfigurable fault tolerance framework is presented. This framework overlaps with the *Adaptive Subsystem* in this paper. The proposed system in [2] is restricted in terms of resources due to the triplicated *MicroBlaze* processor and the uniform bus interface of the partially reconfigurable regions.

The primary contribution of the current paper is the BRAM radiation particle sensor which can be combined with any suitable adaptive system similar to [2].

Finally, there exists some work on the usage of SRAMs for radiation detection and measurement. For example, an SRAM can be used to detect neutrons [14] or protons [15] in scientific experiments. In [16], the SRAM-based configuration memory of an XQR2V6000 FPGA is used as an radiation detector. Also in space applications, SRAMs may be used to measure the current radiation [17]. On-orbit SEU results of the configuration SRAM and BRAM in XQVR1000 FPGAs are presented in [18].

IV. Radiation Environment And Reliability Analysis

In the following Section IV-A, the widely accepted tool *Cosmic Ray Effects on Micro-Electronics* (CREME96) is used to determine the radiation environment of the *Heinrich Hertz* satellite. In particular, the SEU rates regarding the configuration memory μ_{CFG} and the embedded BRAM μ_{BRAM} are discussed. In Section IV-B, the required mathematical background is given to determine PFH values in dependency of the configuration memory SEU rate μ_{CFG} and system design parameters, i.e., design size and configuration memory scrub cycle.

A. Device Single Event Upset Rates

The satellite for *In-Orbit Verification* will be launched into a GEO and will be equipped with the *Fraunhofer On-Board Processor* (FOBP), which is a multi-FPGA platform with radiation-hardened *Virtex-5QV* FPGAs. The FPGAs of the FOBP are protected with a 4.5 mm aluminum shielding. With respect to the orbit and the shielding, the impact of solar particles on the *Virtex-5QV* is analyzed in order to determine the expected SEU rates during the mission. The SEU upset rates μ_{HI} and μ_{P} due to heavy ions, respectively protons are commonly calculated by the following equations:

$$\mu_{\mathrm{HI}} = \int_{L_{\mathrm{on}}}^{L_{\mathrm{max}}} \sigma(L)P(L)F(L)\mathrm{d}L \tag{1}$$

$$\mu_{\mathrm{P}} = \int_{E_{\mathrm{on}}}^{E_{\mathrm{max}}} \sigma(E)f(E)\mathrm{d}E. \tag{2}$$

To evaluate Eq. (1) and Eq. (2), the integral flux $F(L)$ for heavy ions and the differential flux $f(E)$ for protons have to be calculated using CREME96, where L and E denote the *Linear Energy Transfer* (LET), respectively the proton energy. CREME96 calculates an average flux of particles for five different solar conditions: *Solar Minimum*, *Solar Maximum*, *Worst Week* (flare-enhanced), *Worst Day* (flare-enhanced), and *Peak 5 Minutes* (flare-enhanced). Furthermore, the so-called *Weibull* fits, the cross section $\sigma(L)$ and $\sigma(E)$ for heavy ions, respectively protons have to be obtained during radiation test campaigns for each considered device. The *Xilinx Radiation Test Consortium* (XRTC) has measured these device parameters for various resource types of the *Virtex-5QV* FPGA [19]. In addition, $P(L)$ in Eq. (1) denotes the percentage of particles at the given LET that will deposit at least a critical charge. [20], [21]

The overall upset rate μ is then given by the summation of μ_{HI} and μ_{P}. Tab. I shows the SEFI rates of the configuration controller, the SEU rates μ_{CFG} and μ_{BRAM} of the configuration memory and of the embedded BRAM. For the sake of completeness, the SEU rates for other resource types of the FPGA are listed as well. Moreover, for each device, respectively resource type, the number of bits or the amount of primitives are given.

Fig. 3 depicts the overall upset rates, integrated from the onset L_{on}, respectively E_{on} to the maximum L_{max} or E_{max} of the flux. The curves are presented separately for heavy ions and protons in dependency of L_{on}, respectively E_{on}. The first left marker of all curves represents the upset rate at the onset of the *Solar Maximum*. From this point, the next points right represents the radiation conditions *Solar Minimum*, *Worst Week*, *Worst Day* and *Peak 5 Minutes*. The *Solar Maximum* creates less upsets than the *Solar Minimum* because in the *Solar Maximum* condition the solar radiation shields more particles from the galactic cosmic ray.

As shown by the SEU rates listed in Tab. I and by the SEU curves depicted in Fig. 3, the embedded BRAM is the most sensitive resource type. This characteristic motivates the utilization of embedded BRAM blocks as particle sensor. The cause of this sensitivity is that the BRAM is not radiation hardened by design [22]. One BRAM primitive consists of 72 bit (64 bit data and 8 bit parity) times a depth of 512. This results in 36,864 bits per primitive.

Table I: Overall upset rates μ of the *Virtex-5QV* in GEO with a 4.5 mm aluminum shielding

Resource Type / Device	Count	*Solar Min.* (Upsets/s/device)	*Solar Max.* (Upsets/s/device)	*Worst Week* (Upsets/s/device)	*Worst Day* (Upsets/s/device)	*Peak 5 Min.* (Upsets/s/device)
Configuration Controller	4	7.14×10^{-12}	2.00×10^{-12}	7.84×10^{-10}	2.94×10^{-9}	1.07×10^{-8}
Configuration Memory	34,087,072 Bit	2.02×10^{-8}	3.48×10^{-9}	1.33×10^{-6}	3.80×10^{-6}	1.37×10^{-5}
BRAM Primitive 512×72 Bit	298	4.59×10^{-5}	1.44×10^{-5}	5.16×10^{-3}	2.13×10^{-2}	7.80×10^{-2}
FF SET Filter off	81,920	4.43×10^{-8}	1.06×10^{-8}	3.54×10^{-6}	1.24×10^{-5}	4.51×10^{-5}
FF SET Filter on	81,920	3.16×10^{-7}	7.44×10^{-8}	1.54×10^{-5}	4.78×10^{-5}	1.74×10^{-4}
DSP M-Register	320	1.47×10^{-6}	5.20×10^{-7}	1.38×10^{-3}	5.44×10^{-3}	1.98×10^{-2}
DSP other Register	1,280	3.24×10^{-6}	1.15×10^{-6}	3.03×10^{-3}	1.19×10^{-2}	4.34×10^{-2}

298 BRAM primitives are available in the *Virtex-5QV*. Tab. II shows the SEU rate μ_{BRAM} for a different number of *BRAM Fault Detector* primitives according to Fig. 2. To achieve a sufficient resolution, we suggest to use at least 64 BRAM primitives for our satellite mission. This number influences and defines the mean time between upsets of the radiation sensor. Using this choice, it is likely to expect an upset approx. every 28 hours for *Solar Minimum*, every 15 minutes for *Worst Week* and every minute for the *Peak 5 Minutes*. The transition from one solar condition to another may be assumed rather smooth. A first coarse estimation averages three measured mean time to upset values (BRAM SEUs) respectively the upset rate. Due to the random *Poisson* distributed nature of the SEUs we assume a low confidence of this scheme. A sophisticated estimation of the BRAM upset rate is part of our future work.

Through measuring μ_{BRAM}, we may determine the corresponding flux, respectively the solar condition. Finally, from the determined flux and according to the relation shown in Fig. 3, we compute a mapping function f for any μ in Fig. 4 where the curves are calculated by linear interpolation at a logarithmic scale between the values of the five solar conditions. In this paper, we focus on the function f_{CFG} which maps μ_{BRAM} to μ_{CFG} (see Eq. 3) in order to evaluate the reliability of application modules in

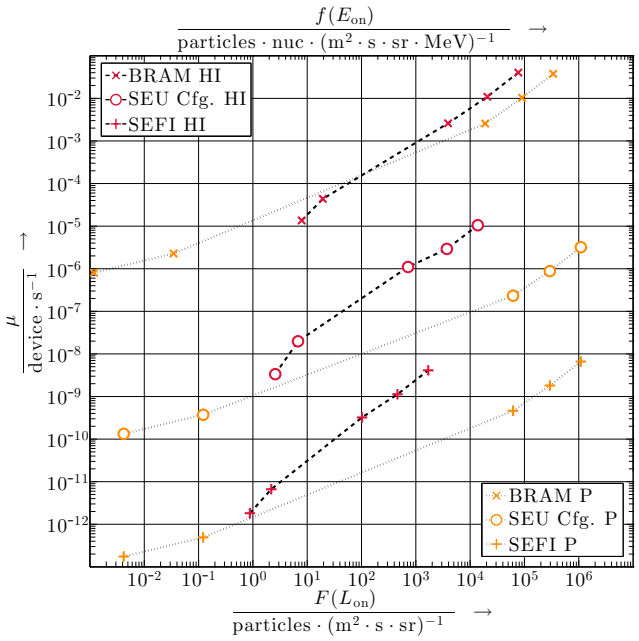

Figure 3: Heavy ion (HI) upset rate as function of the integral flux $F(L_{\mathrm{on}})$ and proton (P) upset rate as function of the differential flux $f(E_{\mathrm{on}})$ for selected resource types

Table II: Upset rate μ_{BRAM} regarding SEUs in BRAMs

# of Fault Detector BRAM Primitives	*Solar Min.* (Upset/s)	*Worst Week* (Upset/s)	*Peak 5 Min.* (Upset/s)
1	1.54×10^{-7}	1.73×10^{-5}	2.62×10^{-4}
32	4.93×10^{-6}	5.54×10^{-4}	8.38×10^{-3}
64	9.86×10^{-6}	1.11×10^{-3}	1.68×10^{-2}
298 (max. amount)	4.59×10^{-5}	5.16×10^{-3}	7.80×10^{-2}

the *Adaptive Subsystem*:

$$\mu_{\mathrm{CFG}} = f_{\mathrm{CFG}}(\mu_{\mathrm{BRAM}}). \tag{3}$$

B. Evaluation of Module Reliability

It is our goal to guarantee a certain module reliability, in particular a target module PFH, for modules instantiated in the partial reconfigurable region of our *Adaptive Subsystem*. Here, the probability that an SEU in the FPGA configuration memory leads to a failure of a module m obviously depends on the number of configuration bits utilized by the module which are commonly referred to as *essential* or *sensitive* bits. This number may be determined by either fault injection test [11], [23] or by tools provided by the FPGA vendor [24]. In the following, we assume that the corruption of each single *essential* bit may lead to a module failure if module m is not triplicated. With this assumption, the module failure rate λ_m can be estimated by

$$\lambda_m = \mu_{\mathrm{CFG}} \cdot \frac{n_{\mathrm{e},m}}{n_{\mathrm{FPGA}}} \tag{4}$$

where $n_{e,m}$ denotes the number of *essential* bits of module m, and n_{FPGA} denotes the overall number of configuration bits of the FPGA. With the module failure rate given by Eq. (4), we assume the reliability of the module to decrease exponentially over time t which is expressed by the module reliability function $R_m(t)$ with

$$R_m(t) = e^{-\lambda_m \cdot t}. \tag{5}$$

The reliability $R_m(t)$ of module m at time t denotes the probability that module m operates without any failure in the interval $[0, t]$. Using Eq. (5), the PFH of module m may then be calculated as

$$\mathrm{PFH}_m = 1 - R_m(T_{\mathrm{h}}) \quad \text{with} \quad T_{\mathrm{h}} = 3600 \text{ s}. \tag{6}$$

In order to achieve a higher module reliability, respectively a smaller PFH_m for a given SEU rate μ_{CFG}, the module can be triplicated. Using TMR, the reliability function for module m is given by

$$R_m^{\mathrm{TMR}}(t) = 3R_m(t)^2 - 2R_m(t)^3. \tag{7}$$

Since we assume the configuration memory to be scrubbed periodically with scrub cycle t_s, the average failure rate λ_m^{TMR} within a scrub cycle for a triplicated module can be estimated according to [11] by

$$\lambda_m^{\mathrm{TMR,s}} = \frac{1 - R_m^{\mathrm{TMR}}(t_s)}{t_s}. \quad (8)$$

Using Eq. (8) and Eq. (5), the reliability function $R_m^{\mathrm{TMR,s}}$ for TMR with scrubbing can be finally calculated as follows:

$$R_m^{\mathrm{TMR,s}}(t) = e^{-\lambda_m^{\mathrm{TMR,s}} \cdot t} \quad (9)$$

with $\mathrm{PFH}_m^{\mathrm{TMR,s}}$ given by

$$\mathrm{PFH}_m^{\mathrm{TMR,s}} = 1 - R_m^{\mathrm{TMR,s}}(T_\mathrm{h}) \quad \text{with} \quad T_\mathrm{h} = 3600 \text{ s}. \quad (10)$$

In these calculations, the reliability of the voter is not considered which we want to account in future work.

V. System Implementation

In the following, the implementation of the system blocks shown in Fig. 2 are explained in detail. Section V-A and Section V-B focuses on the implementation of the *BRAM Sensor Subsystem* with the *BRAM Fault Detector* (BFD) and the *Fault Management Unit* (FMU). A *Xilinx MicroBlaze* soft CPU-core realizes this error handling of the FMU. Section V-C and Section V-D cover the implementation of the *Redundancy Control Unit* and the *Adaptive Subsystem*.

A. BRAM Fault Detector

Fig. 5 shows the structure of the proposed *BRAM Sensor Subsystem* including the BFD. Every BFD has its own *BRAM Scrubber* with an *ECC BRAM* primitive. The BRAM data can be stored in two ways. First, the BRAM contains a deterministic pattern and works as a standalone sensor. A pattern generator provides this deterministic pattern for the initial BRAM content. The second option enables the BRAM as user storage and works as a integrated sensor. The BRAM is accessible by an user BRAM controller (see Fig. 5 left).

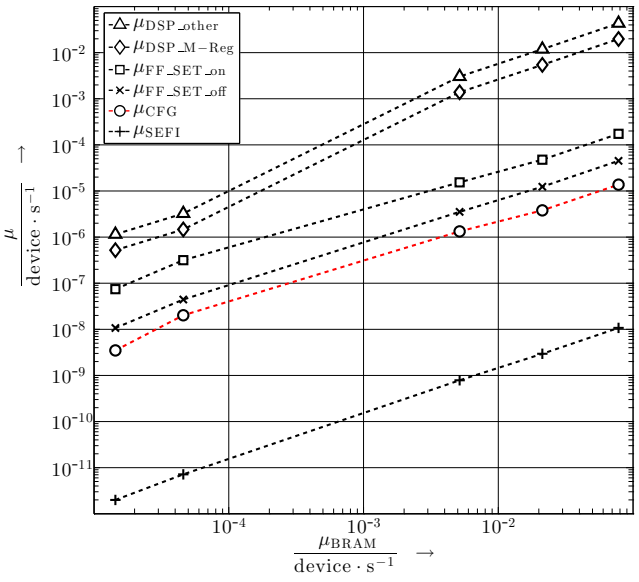

Figure 4: Mapping of the BRAM upset rate μ_{BRAM} to the upset rates of other *Virtex-5QV* resources (entire device)

In order to mitigate SEUs in the BRAM, *Xilinx* provides the *ECC BRAM* primitive which enables error detection and correction (EDAC) to protect the BRAM content. This *ECC BRAM* primitive has a built-in ECC as a simple dual port RAM primitive with a separated read and write port. The use of the *Xilinx ECC BRAM* instantiates a conventional BRAM primitive and activates additional hardware to perform EDAC by using a built-in (72, 64) *Hamming* code to detect single and double errors and to calculate a syndrome vector of a code word. EDAC improves the upset rates of the BRAM approximately by four orders of magnitude. With this mitigation technique, the BRAM may be assumed a reliable component and can be used even as working memory for soft-core microprocessors inside the FPGA. [25] [22]

The *ECC BRAM* primitive corrects an error only at the data output. If an SEU leads to an error in a BRAM cell, this error is not corrected in the particular cell. The *ECC BRAM* primitive only calculates the ECC for the current read and write address. Therefore, the BRAM scrubber is necessary to correct the BRAM content. If the single bit error (*SBITERR*) signal indicates an error, the BRAM will be scrubbed. Only the corrupted word line is scrubbed by a delayed read address and enable. The corrected data from *ECC BRAM* may be used directly without delay. In case of a double bit error (*DBITERR*), the data cannot be corrected. If two or more bits of one BRAM primitive are corrupted during a scrub cycle (in the range of seconds) we interpret this as a *Multiple-Bit Upset* (MBU).

To ensure that the whole BRAM content is error-free, all addresses have to be checked. For the standalone sensor mode, the address generator in the BFD module executes this address access deterministically and steps through the complete address range. By disabling the standalone sensor mode, the integrated sensor mode with a user BRAM controller is enabled with an non-deterministic address access. The user is responsible for the address range sweep to detect or scrub all errors. Furthermore, the BFD module includes counters to accumulate all occurred errors. The fault detector writes these counter values together with additional error data (address, data word and ECC value) to the *Fault Memory* whenever an error signal raises. The FMU reads this *Fault Memory* periodically, analyzes the fault data, performs extended fault statistics, and determines the required redundancy level l for module replication.

According to Tab. II, one BRAM is not enough to reach a sufficient resolution. The number of the BFDs is parameterizable to instantiate multiple BFD instances to achieve a sufficient time resolution. A bus structure combines all BFDs and allows the access to each *Fault Memory*. In our implementation, we chose a star topology for ease of realization. This centralized and polling-based access allows a fast access time to the fault data in the *Fault Memory* of each BFD. The data transfer is uni-directional from the BFD to the FMU and is controlled by a *Bus Access Control* block.

B. Fault Management Unit

The *Fault Management Unit* (FMU) is a major part of the *BRAM Sensor Subsystem* and assigned the following tasks:

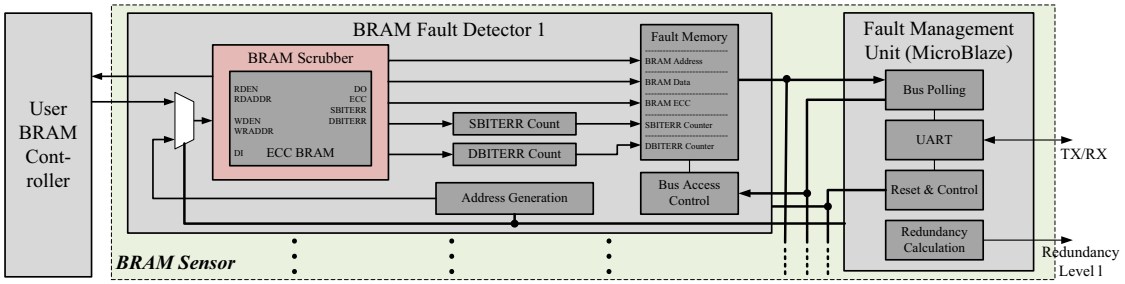

Figure 5: *BRAM Sensor Subsystem* with one *BRAM Fault Detector* (fault counting and storage, data correction and BRAM scrubbing) and the *Fault Management Unit* (bus access control and redundancy calculation)

Bus polling to access the *Fault Memories*, communication of radiation data via *Universal Asynchronous Receiver Transmitter* (UART), reset and control management, calculation of the current upset rate μ_{BRAM}, and signaling of the required redundancy level (see Section VI). We decided to implement these functions in software using a *MicroBlaze* processor. The *MicroBlaze* itself is configured to use the *Fault Tolerance Option*. This provides an ECC for the instruction and data memory, but offers also a higher fault resistance for the memory management unit, the branch target cache and other features. The *MicroBlaze* processor polls each *Fault Memory* cyclically and collects the number of fault events to achieve the required module redundancy level l which will be explained next.

Within each observation interval, the FMU determines the current BRAM upset rate μ_{BRAM}. In our case, we obtain μ_{CFG} according to Fig. 4 by lookup. In Fig. 6, the PFH intervals corresponding to the four SI levels are shown. Also, it can been seen how the SEU thresholds μ^l_{CFG} for the activation of each redundancy level l may be computed according to fixed thresholds PFH^l for a module m occupying $n_{\text{e},m}$ essential configuration bits according to Eq. (4)-(6) as

$$\mu^l_{\text{CFG}} = -\frac{\ln(1-\text{PFH}^l)}{T_{\text{h}}} \cdot \frac{n_{\text{FPGA}}}{n_{\text{e},m}} \quad \text{with} \quad T_{\text{h}} = 3600\ s. \tag{11}$$

With $\text{PFH}^{l=1} = 3 \times 10^{-6}$ and $\text{PFH}^{l=2} = 1 \times 10^{-5}$ to guarantee at least SIL 1 [1], no redundancy ($l = 0$) needs to be introduced for values $\mu_{\text{CFG}} \leq 4.11 \times 10^{-8}$. If $\mu_{\text{CFG}} \in [4.11 \times 10^{-8} ... 1.37 \times 10^{-7}]$, modules need to be duplicated (DMR) which enables fault detection at the module outputs but which does not improve the PFH of the system. Finally, if $\mu_{\text{CFG}} \geq 1.37 \times 10^{-7}$, the modules need to be triplicated (TMR) to keep the system SIL 1 compliant. Notably, this is possible according to Fig. 6 for upset rates greater than the *Peak 5 Minutes* condition. In this example, we used $n_{\text{e},m} = 691,354$ and $n_{\text{FPGA}} = 34,087,072$ which corresponds to the design parameters of the module *demod1* described in Section VI.

A GPIO connection of the processor system provides the required redundancy level l to the *Adaptive Subsystem*.

[1] SIL 1: $\text{PFH} = 10^{-5} ... 10^{-6}$; SIL 2: $\text{PFH} = 10^{-6} ... 10^{-7}$; SIL 3: $\text{PFH} = 10^{-7} ... 10^{-8}$; SIL 4: $\text{PFH} = 10^{-8} ... 10^{-9}$

C. Reconfiguration Control Unit

The *Reconfiguration Control Unit* (RCU) reconfigures, via the ICAP primitive, the partial reconfigurable region of the *Adaptive Subsystem* the required redundancy level l changes. In this case, the RCU generates the partial bitstream header to initiate the partial reconfiguration and to set the location of the partial module. Furthermore, the RCU forwards the header and the data of the partial bitstream to the instantiated ICAP primitive, which is clocked with 100 MHz and has an input data width of 32 bits. The partial bitstreams may be stored in an external radiation hardened memory which will be loaded by the RCU during reconfiguration.

On the FOBP of the *Heinrich Hertz* satellite mission, there is 1 Gbit SDRAM memory available for this purpose [26].

D. Adaptive Subsystem

As depicted in Fig. 2, we assume the *Adaptive Subsystem* to consist of three data channels with Channel 1 having the highest priority. Channel 2 and 3 can be switched off by a multiplexer, if the resources are needed to replicate modules of Channel 1. According to Fig. 7, we distinguish three configurations of the *Adaptive Subsystem* used for

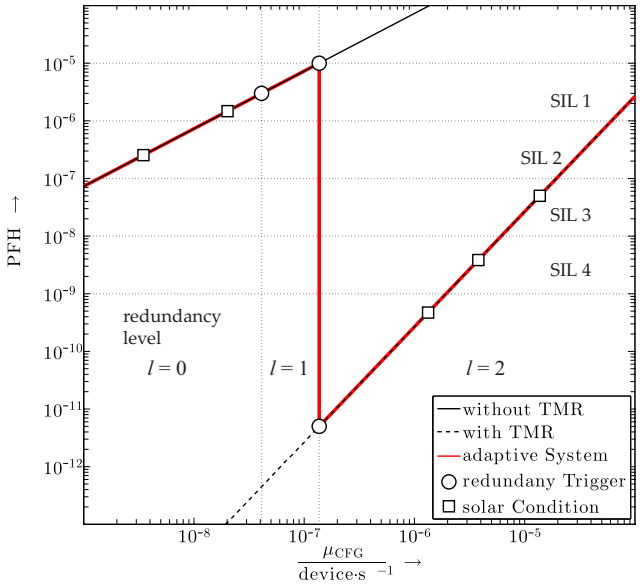

Figure 6: Resulting PFH of an FPGA device in dependence of SEU rate μ_{CFG}. Shown also (red) are the transition points when to switch to $l = 1$ (DMR) and $l = 2$ (TMR) module replication, respectively in order to guarantee a SIL of one.

Table III: Overview regarding to the resource utilization of the demodulator modules of the three data channels with respect to counts on LUTs, flip flops and DSP48 primitives. For each demodulator module m, the number of essential bits n_{e} and the value of PFH_m (regarding configuration memory) for five solar conditions are given.

Module	LUTs	FFs	DSP48	n_{e}	PFH_m *Solar Min.*	PFH_m *Solar Max.*	PFH_m *Worst Week*	PFH_m *Worst Day*	PFH_m *Peak 5 Min.*
demod1	4,176	3,735	69	691,354	1.47×10^{-6}	2.54×10^{-7}	9.71×10^{-5}	2.77×10^{-4}	1.00×10^{-3}
demod2	4,176	3,735	69	673,732	1.44×10^{-6}	2.48×10^{-7}	9.47×10^{-5}	2.70×10^{-4}	9.75×10^{-4}
demod3	4,176	3,735	69	757,174	1.61×10^{-6}	2.78×10^{-7}	1.06×10^{-4}	3.04×10^{-4}	1.10×10^{-3}
demod1 (DMR)	8,352	7,470	138	1,382,708	1.47×10^{-6}	2.54×10^{-7}	9.71×10^{-5}	2.77×10^{-4}	1.00×10^{-3}
demod1 (TMR)	12,528	11,205	207	2,074,062	1.07×10^{-13}	3.23×10^{-15}	4.72×10^{-10}	3.85×10^{-9}	5.01×10^{-8}

the three redundancy levels $l = 0$, 1, and 2. In case of $l = 0$, the configuration in Fig. 7 a) is used in which data of all three data channels are processed concurrently using channel-specific modules for each channel. For $l = 1$, the configuration in Fig. 7 b) is configured in which Channel 2 is switched off by the multiplexer because the resources of the processing modules of Channel 2 are used for the duplicated modules of Channel 1. Additionally, an *Error Detector* (ED) is inserted, which might be used as another reliability indicator or for error concealment techniques. Finally, for $l = 2$, the configuration in Fig. 7 c) is autonomously configured in which Channel 2 and 3 are switched off and the modules of Channel 1 are triplicated (TMR). Furthermore, a voter is inserted in order to mask errors of one potentially corrupted module. Furthermore, we assume that the system has a blind configuration memory scrubber which periodically corrects the configuration memory within the scrub cycle t_s. For synchronization purposes, we reset the duplicated or triplicated modules to an inital state after each reconfiguration.

To implement the partial reconfigurable system, we used the tool flow of the tool *GoAhead* [27]. *GoAhead* enables a two-dimensional grid-style reconfiguration with module re-allocation for all major Xilinx FPGA device types. Furthermore, *GoAhead* provides point-to-point communication structures to enable the data exchange between static and partial designs.

VI. Case Study

As a case study, we use *GoAhead* together with the *Xilinx* design tool *PlanAhead* 13.2 to implemented the proposed

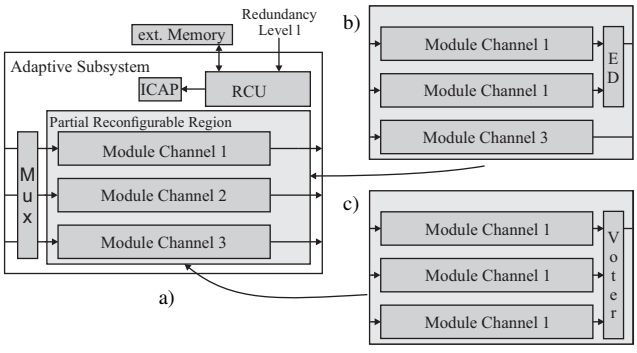

Figure 7: Redundancy levels l of the *Adaptive Subsystem*. In case of $l = 0$, the configuration a) *no redundancy* is used. In case of $l = 1$, the configuration in b) *Channel 1 DMR* is used. In case of $l = 2$, the configuration in c) *Channel 1 TMR* is used.

self-adaptive system on a *Virtex-5QV* FPGA. For the *BRAM Sensor Subsystem*, 64 BFDs and one FMU were instantiated. The *Adaptive Subsystem* consists of three channels in order to demodulate data streams of three different frequency bands with three QPSK demodulator modules. These modules are named *demod1*, *demod2* or *demod3* where each module featured channel-specific filter coefficients. Tab. III presents the resource utilization of the demodulator modules with respect to counts on LUTs, flip flops and DSP48 primitives. Furthermore, for each demodulator module, the number of *essential* bits $n_{\mathrm{e},m}$ and the values of PFH_m for the five solar conditions of each module are given.

In case of $l = 1$, module *demod 1* is duplicated by deactivating Channel 2 and by utilizing the resources of module *demod 2*. In case of $l = 2$, module *demod 1* is triplicated by deactivating Channel 2 and 3. The last two rows in Tab. III show the resource utilization for the duplicated, respectively triplicated module *demod1* and the corresponding PFH values for the five solar conditions. Here, for the calculation of the PFH values, scrubbing with a scrub cycle of $t_{\mathrm{s}} = 60$ s is assumed. Furthermore, the additionally resources for the error detection, respectively for the voter are neglected. Tab. IV presents for each redundancy level the throughput and related SEU rate μ_{CFG} intervals.

It can be seen from Tab. III that TMR with scrubbing may improve the PFH by 4 to 7 orders of magnitude. At the solar condition *Peak 5 Minutes* the PFH of module *demod1* is reduced due to TMR by a factor of 2.00×10^4 which equals a percental decrease of 99.995 %. Nonetheless, the enhanced reliability comes at a price: Due to TMR, the throughput is reduced to one third. DMR does not enhance the reliability but enables an additional error detection which may be used by the application. According to the SEU rate intervals reported in Tab. IV, the redundancy level $l = 0$ is used at SEU rates of the solar condition *Solar Minimum* and *Solar Maximum* whereas redundancy level $l = 2$ is needed to be configured for the upset rates of the solar conditions *Worst Week*, *Worst Day*, and *Peak 5 Minutes*. Finally, redundancy level $l = 1$ (DMR) may be used in the transition region between redundancy level 0 and 2. Here, our proposed system autonomously configures the least amount of redundancy depending on the current radiation scenario so to realize the best tradeoff between reliability and troughput, respectively resource utilization.

VII. Conclusions

In this paper, we propose an FPGA-based self-adaptive reconfigurable system for space missions which relies on on-chip radiation measurements using standard BRAM primitives. The provided adaptivity through partial reconfiguration

Table IV: Results on the achievable throughput in dependence of redundnacy level l and related SEU rate μ_{CFG}

Redundancy level l	Throughput	μ_{CFG} interval
0 (no redundancy)	3×	$< 4.11 \times 10^{-8}$
1 (Channel 1 DMR)	2×	4.11×10^{-8} ... 1.37×10^{-7}
2 (Channel 1 TMR)	1×	$> 1.37 \times 10^{-7}$

is used to increase the reliability in terms of PFH by replicating modules only in times of increased radiation. In times of normal or low radiation, the freed area can be used to instantiate further modules which implement other tasks or increase the throughput of the system. The radiation measurement itself is done by counting SEUs of selective on-chip BRAMs. From this measured BRAM SEU rate, the upset rate of the configuration memory may be derived. From the configuration memory SEU rate, the module specific PFH can be finally calculated. If at run-time, the current PFH value exceeds a module-specific threshold, e.g., according to a specified SIL, the module will be replicated autonomously.

The BRAMs, used as radiation sensors can be further utilized for a user design.

For future work, we want to investigate different TMR granularities to optimize our system, respectively module reliability. Furthermore, we want to include the reliability of the voters and multiplexers in our evaluation of reliability which we want to verify with an fault injection method. In addition the upset rates of flip flops, DSP48s and BRAMs (with and without ECC) will be taken into account for the *Adaptive Subsystem* as well as for the *BRAM Sensor Subsystem* and the RCU. Also a sophisticated estimation model of the measured time between BRAM upsets will be investigated.

REFERENCES

[1] S. Voigt, "The german *Heinrich Hertz* satellite mission," in *2010 Proceedings of the Fourth European Conference on Antennas and Propagation (EuCAP)*, Apr. 2010, pp. 1 –4.

[2] A. Jacobs, G. Cieslewski, A. D. George, A. Gordon-Ross, and H. Lam, "Reconfigurable fault tolerance: A comprehensive framework for reliable and adaptive FPGA-Based space computing," *ACM Trans. Reconfigurable Technol. Syst.*, vol. 5, no. 4, p. 21:121:30, Dec. 2012.

[3] *Functional Safety of electrical / electronic / programmable electronic safety related systems (IEC 61508)*, International Electrotechnical Commission, 2005.

[4] Xilinx Inc., "Space-grade rad-hard virtex-5qv fpga," 2011.

[5] R. E. Lyons and W. Vanderkulk, "The Use of Triple-Modular Redundancy to Improve Computer Reliability," *IBM J. Res. Dev.*, vol. 6, no. 2, pp. 200–209, Apr. 1962.

[6] J. Heiner, B. Sellers, M. Wirthlin, and J. Kalb, "FPGA Partial Reconfiguration via Configuration Scrubbing," in *International Conference on Field Programmable Logic and Applications*. IEEE, 2009, pp. 99–104.

[7] *Product Guide: LogiCore IP Soft Error Mitigation Controller v3.4*, Xilinx Inc., 2012.

[8] A. Sari and M. Psarakis, "Scrubbing-based seu mitigation approach for systems-on-programmable-chips," in *FPT*, 2011, pp. 1–8.

[9] L. Sterpone and M. Violante, "A new Analytical Approach to Estimate the Effects of SEUs in TMR Architectures Implemented through SRAM-based FPGAs," *IEEE Trans. on Nuclear Science*, vol. 52, no. 6, pp. 2217–2223, 2005.

[10] J. Angermeier, D. Ziener, M. Glaß, and J. Teich, "Stress-Aware Module Placement on Reconfigurable Devices," in *Proceedings of International Conference on Field-Programmable Logic and Applications (FPL 2011)*, Chania, Crete, Greece, Sep. 2011.

[11] P. Ostler, M. Caffrey, D. Gibelyou, P. Graham, K. Morgan, B. Pratt, H. Quinn, and M. Wirthlin, "SRAM FPGA reliability analysis for harsh radiation environments," *IEEE Transactions on Nuclear Science*, vol. 56, no. 6, pp. 3519–3526, 2009.

[12] R. Al-Haddad, R. Oreifej, R. A. Ashraf, and R. F. DeMara, "Sustainable modular adaptive redundancy technique emphasizing partial reconfiguration for reduced power consumption," *International Journal of Reconfigurable Computing*, vol. 2011, no. 1, p. 25, Jan. 2011.

[13] C. Bolchini, A. Miele, and C. Sandionigi, "A novel design methodology for implementing reliability-aware systems on SRAM-Based FPGAs," *IEEE Transactions on Computers*, vol. 60, no. 12, pp. 1744–1758, Dec. 2011.

[14] D. Makowski, M. Grecki, B. Mukherjee, S. Simrock, B. Swiercz, and A. Napieralski, "The application of a sram chip as a novel neutron detector," *Journal of Experimental Nanoscience*, vol. 1, no. 2, pp. 261–268, 2006.

[15] G. Soli, B. Blaes, and M. Buehler, "Proton-sensitive custom sram detector," *Nuclear Science, IEEE Transactions on*, vol. 39, no. 5, pp. 1374–1378, 1992.

[16] S. Skutnik and J. Lajoie, "A geant-based model for single event upsets in sram fpgas for use in on-detector electronics," *Nuclear Science, IEEE Transactions on*, vol. 53, no. 4, pp. 2353–2360, 2006.

[17] M. Buehler, G. Soli, B. Blaes, J. Ratliff, and H. Garrett, "Clementine rrelax sram particle spectrometer," *Nuclear Science, IEEE Transactions on*, vol. 41, no. 6, pp. 2404–2411, 1994.

[18] M. Caffrey, K. Morgan, D. Roussel-Dupre, S. Robinson, A. Nelson, A. Salazar, M. Wirthlin, W. Howes, and D. Richins, "On-orbit flight results from the reconfigurable cibola flight experiment satellite (CFESat)," in *17th IEEE Symposium on Field Programmable Custom Computing Machines, 2009. FCCM '09*, 2009, pp. 3–10.

[19] G. Swift and G. Allen, "Virtex-5QV static SEU characterization summary," Jun. 2012.

[20] E. Petersen, *Single event effects in aerospace*. [Piscataway, N.J.]; Hoboken, N.J.: IEEE Press ; Wiley, 2011.

[21] J. Engel, M. Wirthlin, K. Morgan, and P. S. Graham, "Predicting on-orbit static single event upset rates in xilinx virtex FPGAs," 2006.

[22] G. Allen, L. Edmonds, C. W. Tseng, G. Swift, and C. Carmichael, "Single-event upset (SEU) results of embedded error detect and correct enabled block random access memory (block RAM) within the xilinx XQR5VFX130," *IEEE Transactions on Nuclear Science*, vol. 57, no. 6, pp. 3426 –3431, Dec. 2010.

[23] B. Pratt, M. Caffrey, P. Graham, K. Morgan, and M. Wirthlin, "Improving FPGA Design Robustness with Partial TMR," in *IRPS*, 2006, pp. 226–232.

[24] R. Le, "Application Note XAPP538: Soft Error Mitigation using Prioritized Essential Bits," Xilinx Inc., Tech. Rep., 2012.

[25] S. Tam, "Single error correction and double error detection," Aug. 2006.

[26] R. Glein, F. Rittner, and A. Hofmann, "Ensuring FPGA reconfiguration in space," *Xcell Journal*, no. 84, pp. 23–27, Jul. 2013.

[27] C. Beckhoff, D. Koch, and J. Torresen, "Go ahead: A partial reconfiguration framework," in *Field-Programmable Custom Computing Machines (FCCM), 2012 IEEE 20th Annual International Symposium on*, 29 2012-may 1 2012, pp. 37 –44.