Validation of FDIR Strategy for Spaceborne SRAM-based FPGAs Using Proton Radiation Testing

Felix Siegle, Tanya Vladimirova, Christian Poivey, and Omar Emam

Abstract—This paper presents a novel stream processor architecture for SRAM-based FPGAs that is specifically targeted at payload data processing and which employs innovative Fault Detection, Isolation and Recovery (FDIR) mechanisms to cope with failures caused by radiation effects. As part of this FDIR strategy, an availability analysis method is developed that is able to predict the steady state availability of a stream processor in a particular radiation environment. By means of an accelerated proton irradiation test campaign, both the FDIR framework and the availability analysis method are validated. First, it is demonstrated that the FDIR hardware and software components are capable to detect and recover from failures in a real radiation environment. Secondly, it is proven that the availability prediction provides accurate results. The real Mean Time Between Failures (MTBF) value measured during the beam test differs from the prediction by not more than 15.4% while the steady state availability by only 0.9%.

I. INTRODUCTION

Modern approaches to payload data processing on board spacecraft demand increased processing capabilities. Field-Programmable Gate Arrays (FPGAs) are suitable for payload data processing chains because typical data and image algorithms can benefit significantly from their inherent hardware parallelism. In Static Random-Access Memory (SRAM) based FPGAs, processing units can be hosted by separated partitions, which could be dynamically reconfigured during run time. This feature is attractive for space missions in which power and hardware resources are limited. If the functionality of a processing unit can be reconfigured, the same chip area can be time-shared by different units for implementation of processing tasks. However, most spaceborne SRAM-based FPGAs suffer from radiation induced upsets in the configuration memory, embedded RAM blocks, flip-flops and other building blocks.

In cooperation with the European Space Agency and Airbus Defence and Space UK, a novel stream processor architecture was developed that is specifically targeted at payload data processing applications, which employ innovative Fault Detection, Isolation and Recovery (FDIR) mechanisms to cope with failures caused by radiation effects. As part of this FDIR strategy, an availability analysis method was proposed that is able to predict the steady state availability of a stream processor in a particular radiation environment. Earlier this year, a proton irradiation test campaign was conducted that

validated both the hardware framework and the availability analysis method.

The paper is structured as follows. Section II gives a brief overview of related work. Section III describes the proposed hardware architecture and the test setup used during the proton irradiation test campaign. Then, Section IV explains the proposed availability analysis method. Next, the test procedure is described in Section V. Finally, the static and dynamic test results are presented and discussed in Section VI.

II. RELATED WORK

Failure recovery strategies for SRAM-based FPGAs, which are based on spatial redundancy at module level, have been investigated for several years now. For instance, a system is described in [1], which either uses an on-chip test bench or *Dynamic Triple Modular Redundancy (TMR)* to detect a faulty module. A similar concept can be found in [2]. However, in addition to Dynamic TMR, the system also uses a module duplication. Similarly, some work has been done regarding availability analysis of such systems. McMurtrey et al. use Markov models to estimate the reliability of TMR systems [3]. Ostler et al. present a reliability analysis of SRAM-based FPGAs in [4]. Kastil et al. present a dependability analysis of their fault tolerant systems in [5]. Martin et al. also use Markov chains in [6] to model the availability that can be achieved with different scrubber implementations. A detailed survey of related work can be found in [7].

The work proposed in [8], [9] advances the aforementioned concepts in several aspects. First, the developed Distributed Failure Detection technique allows the distribution of redundant modules over several FPGAs because the failure detection is done within a communication network. Secondly, the proposed availability analysis method is more precise than earlier approaches as it also takes into account Block RAM upsets. Finally, to the best of the authors' knowledge, such a framework is validated by proton irradiation testing for the first time in Europe.

III. SYSTEM OVERVIEW

A. Stream Processor Architecture

The architecture of a typical stream processor is shown in Figure 1. An Intellectual Property (IP) core of the desired functionality is embedded into the stream processor. The stream processor further comprises a Network-on-Chip (NoC) interface for the data exchange, some state machine logic and a memory for state variables. The input control words are interpreted by the state machine whereas the input data words are directly fed into the IP core. An additional memory holds

F. Siegle and T. Vladimirova are with the Department of Engineering at University of Leicester, UK.

C. Poivey works at ESTEC, the European Space Research and Technology Centre of the European Space Agency (ESA), Noordwijk, Netherlands.

O. Emam works at Airbus Defence and Space, Stevenage, UK.

Fig. 1. Example of a stream processor.

Fig. 2. Distributed Failure Detection: Example network topology.

all variables necessary to configure the IP core. If the processing pipeline uses a specific protocol, a protocol parser and/or protocol generator may be added to the inputs and outputs of the core. Suitable IP cores are of passive nature, e.g. hardware accelerators that are designed to be connected as slaves to a Central Processing Unit (CPU) bus. With the additional logic in the stream processor, the cores become intelligent enough to process incoming data without the interaction of a CPU, solely by interpreting data and command words in the network input stream. In the following, a Joint Photographic Experts Group (JPEG) image compression stream processor is used as a case study, which is representative of a broad range of satellite payload data processing applications.

B. Distributed Failure Detection

The proposed *Distributed Failure Detection* methodology, first outlined in [8], makes failure detectors part of the network. This novel approach allows the free distribution of redundant processors throughout the network because the output of each processor can be routed to any failure detector, independent of its location in the network. The network is not limited to the interconnection of processors on one single FPGA as it can also span across several FPGAs.

An example network topology is shown in Figure 2. Several partitions (circles) are interconnected via routing switches. Stream processors can be placed on these partitions by means

of partial reconfiguration. In this example, a processor has been triplicated and the resulting instances (grey circles) are placed on some of these partitions.

Suppose, data is sent from a source node *Src* to the processor and the processor sends the resulting data to sink node *Sink*. As the processor is triplicated, the data must first be multicast by the routing switches. For instance, routing switch 1 multicasts the packets to output port 2, 4 and 5. While the data arrives at the first redundant processor instance immediately, switch 2 and 3 forward the data to the other two remaining redundant instances. After data processing, the resulting network packets are routed to the failure detector module *V* (acting as a majority voter), connected to routing switch 3. Finally, the output of the failure detector is routed to the sink node. The failure detector module can handle asynchronous network streams [9]. It can be configured as a comparator or majority voter and it automatically switches from voting mode to comparator mode once an incoming network stream is detected to be faulty. The failure detector module is connected to an external LEON3 SPARC V8 microprocessor, which acts as a FDIR supervisor. Once a failure has been detected, the FDIR supervisor software is informed about which stream processor is faulty. Then, the software repairs the stream processor by partial reconfiguration.

C. Test Setup

The Device Under Test (DUT) is a commercial FPGA device of type Virtex-4 XC4VSX55 by manufacturer Xilinx. The DUT is mounted on a daughterboard that is connected to a mainboard via two high speed interconnects.

A simplified block diagram of the FPGA design, which is used during the irradiation campaign is shown in Figure 3. The first FPGA is mounted on the daughterboard that is irradiated. The FPGA design was minimised as much as possible to simplify the analysis of the resulting data, e.g. the Digital Clock Managers (DCMs) were removed as they are known to be prone to single event effects. Since the proposed FDIR approach is based on dynamic partial reconfiguration, the image compression stream processor is hosted on a reconfigurable partition. Aside from the stream processor, only a NoC to NoC bridge is implemented on the irradiated FPGA, which is necessary to communicate with the second FPGA. The cross-section of the bridge is not known but is assumed to be very small and is thus neglected.

The second FPGA is mounted on a mainboard that is not irradiated during the test campaign. It comprises a second JPEG stream processor, which works in hot redundancy together with the JPEG stream processor hosted on the irradiated FPGA. It further comprises a FDIR routing switch, to which both stream processors are connected. In addition, the NoC is bridged to a SpaceWire interface, which allows the communication with external components. The voter module within the routing switch is used as a comparator. Once this module detects a failure, a health status is flagged to the FDIR supervisor software running on a LEON3 microprocessor, which is implemented on a ProASIC3e FPGA on the same mainboard. The LEON3 microprocessor can access the configuration memory of both Virtex-4 FPGAs via their SelectMAP interface.

Fig. 3. Block diagram of the FPGA design used during the beam test

TABLE I. RANDOM FAULT INJECTION RESULTS

# Tested	# Sensitive	95% confidence interval	
		Min	Max
1,000	134 (13.40%)	11.35%	15.67%
5,000	701 (14.02%)	13.07%	15.01%
10,000	1,382 (13.82%)	13.15%	14.51%
50,000	6,922 (13.84%)	13.54%	14.15%
100,000	13,899 (13.90%)	13.69%	14.11%
150,000	20,870 (13.91%)	13.74%	14.09%

IV. AVAILABILITY ANALYSIS

A new availability analysis method was developed to predict the availability of a stream processor in a particular FDIR configuration and radiation environment.

A block diagram of the methodology can be seen in Figure 4. First, the SEU rates per bit-day for the configuration memory, the Block RAMs and the flip-flops of a particular Virtex-4 device in a specific orbit are determined. The calculations are based on static SEU characterisation data that was gathered from accelerated radiation testing, e.g. as published by Xilinx and NASA [10]. The heavy ion and proton fluxes of the orbit are calculated using radiation models standardised in European standard ECSS-E-ST-10-04C [11]. With the calculated SEU rates, the probability of a bit flip in one single memory element is known. However, to compute the availability of a stream processor, the SEU rate per day and stream processor must be known and not just the rates for the single memory elements. Thus, the bit upset rates must be scaled by the number of sensitive memory elements within the stream processor. In the following, only the configuration memory and the Block RAMs are taken into account since these memory elements are the biggest contributors to the cross-section of the stream processor.

A. Configuration Memory

The number of sensitive bits is determined by randomly injecting faults into the configuration memory space related to the stream processor using a fault injection system (the partial

bitstream of the JPEG image compression stream processor comprises in total 3,735,264 bits). Aside from being able to determine the number of sensitive configuration memory bits, our fault injection system is also able to classify the failures that occur when one of the sensitive bits is hit by a particle. Most failures are transient, i.e. a configuration memory scrub is sufficient to recover the system. Some other failures, however, are persistent and require an additional circuit reset or even a reconfiguration. For our proof-of-concept circuit, a fault injection campaign with 150,000 random injections was conducted. Then, confidence intervals were calculated that estimate the percentage of sensitive bits for the whole stream processor. The results are given in Table I. As can be seen, the estimation of the percentage of sensitive bits becomes better with an increased number of fault injections, i.e. the confidence interval width becomes narrower. Using the worst case assumption, the total number of estimated sensitive bits is:

$$F_C = 0.1409 \cdot 3735264 \text{ bits} = 526299 \text{ bits} \approx 526300 \text{ bits} \quad (1)$$

To prove that this estimation is realistic, a full fault injection campaign was conducted as well, which revealed that 523,543 bits (14.02%) are sensitive. Since this result is within all confidence intervals found in Table I, it is proven that random fault injection can provide accurate results.

B. Block RAM

The sensitivity of the Block RAMs is estimated using a novel custom-built memory profiling tool. In streaming applications, Block RAMs are mainly utilised as FIFOs, i.e. only a fraction of the memory space is used at the same time. In addition, bit upsets in memory cells that are not read out can also not lead to failures. The Block RAM profiling tool analyses a post place & route simulation run to (i) determine the number of used RAM bits and to (ii) calculate a correction factor τ_S , which takes into account that a part of the memory cells is overwritten before an upset can manifest as failure.

The timeline in Figure 5 presents an example, where some read and write accesses of one memory address are shown. The first fault SEU_1 occurs between a read access t_{rd2} and a subsequent write access t_{wr2} . Since the memory row is overwritten with a new value, the fault cannot manifest itself as a failure. The second fault SEU_2 , however, manifests itself as a failure because the memory row is read out at t_{rd3} . All N time spans $T_{m,n}$ in which a memory row m is susceptible (grey boxes in Figure 5) are first accumulated by the memory profiling tool. Then, the results of all M memory rows are averaged. Finally, dividing the averaged value by the total simulation time leads to the correction factor τ_S .

In total, the image compression stream processor comprises 83 Block RAMs. In Virtex-4 devices a Block RAM block comprises 18,432 bits. Therefore, without any knowledge about the usage of the Block RAMs, one could assume a total of 1,529,856 Block RAM bits.

From this large number of total Block RAM bits, the profiling tool revealed that 524,964 bits (34.3%) are used as RAM bits and 27,351 bits (1.8%) as ROM bits.

Fig. 4. Block diagram of the availability analysis methodology.

Fig. 5. Example for the calculation of the correction factor τ_s

Now, taking the correction factor τ_s into account, the Block RAM profiling tool predicts that only 67,858 RAM bits (4.4%) must actually be counted as being susceptible. Thus, together with the ROM bits, a total of 95,209 sensitive Block RAM bits is estimated.

C. Stochastic Petri Nets

Soft errors in SRAM-based FPGAs can be mitigated by a combination of a failure detection and a recovery technique, which makes this type of FPGA a repairable system. The probability that a repairable system functions correctly is called *steady state availability*, often defined as:

$$A = \frac{T_m}{T_m + T_d} \quad (2)$$

where T_m is the mission duration and T_d the observed down time, which is the sum of the time required to detect and recover failures.

For availability modelling, stochastic Petri nets are used that can analytically be solved with the TimeNET 4.1 tool [12]. For the proton irradiation test campaign, the following FDIR configuration is chosen: The image compression stream processor is duplicated and each redundant copy is placed on a dedicated Virtex-4 FPGA (see also Section III-C). One FPGA is irradiated, the other one is protected from the proton beam. Our failure detector module is configured as a comparator and

Fig. 6. Stochastic Petri net

is placed on the second FPGA, i.e. it is assumed that the failure detector itself is fault-free. Once a failure is detected, the external radiation-hardened microprocessor is performing a reconfiguration of the faulty stream processor to recover the system. The system is not available during the reconfiguration.

To predict the availability of this FDIR configuration, the Petri net depicted in Figure 6 is solved. The token in place P_OK represents the health status of both redundant stream processors. The timed transitions mod_seu_ram and mod_seu_cm with exponential random distribution represent the failure rates for all Block RAMs, respectively for the configuration memory of the stream processor. Once the Mean Time Between Failures (MTBF) for one of these memory types elapses, the token moves to place P_FAILED, in which the system is unavailable. The token can move back to place P_OK by means of a failure recovery action: Transition t_detect models the average failure detection time and deterministic transition t_repair models the stream processor repair time, which is known for the implementation. Ultimately, the steady state availability is determined by calculating the probability that the token is on place P_OK.

Fig. 7. A view of the test chamber at PSI.

V. PROTON IRRADIATION TEST PROCEDURE

The proton irradiation test campaign was carried out at the Paul Scherrer Institute (PSI) in Villigen, Switzerland in the night from the 8th to the 9th of May 2015. A view of the test chamber is shown in Figure 7. The following steps were repeated throughout the test campaign:

- 1) Every 100 milliseconds, the so-called instrument simulator software (implemented on a Gaisler SPWRTC board [13]) sends a full raw image via SpaceWire to the second FPGA.
- 2) Within this FPGA, the raw image is multicast to both JPEG stream processors.
- 3) Both JPEG stream processors process the raw image and send the resulting JPEG image to the comparator module.
- 4) The comparator module is doing a bitwise comparison of the two redundant network streams. In case of a mismatch, the comparator module flags a health status to the FDIR supervisor software and stops forwarding any data, i.e. the comparator module is fail-silent. The following steps are initiated after a failure detection:
 - The FDIR supervisor reads back the configuration bitstream of the irradiated FPGA for later data analysis.
 - Since the FDIR supervisor cannot determine which JPEG stream processor failed, both stream processors are reconfigured via their SelectMAP interfaces.
 - After reconfiguration, the FDIR supervisor sends a request to the comparator module to continue the comparison.
- 5) If no mismatch occurs, the comparator module is simply forwarding the JPEG image to the SPWRTC board via SpaceWire.
- 6) The instrument simulator software running on the SPWRTC board is comparing the JPEG image to a “golden

copy”. A counter is keeping track of the number of correctly received JPEG images. At the same time, another counter is keeping track of the number of transmitted raw images. Therefore, the software can continuously determine the availability of the system.

VI. TEST RESULTS

A. Overview

The proton test was conducted at two energy levels: 100 MeV (test runs 17 to 21) and 200 MeV. At 200 MeV, two different fluxes were used, ca. $4.2 \cdot 10^6 \text{ p/cm}^2 \cdot \text{s}$ (test runs 3, 4 and 7 to 12) and ca. $8.3 \cdot 10^6 \text{ p/cm}^2 \cdot \text{s}$ (test runs 5, 6 and 13 to 16). While the test was running, a radiation detector measured the fluence $[\text{p/cm}^2]$. Since the active beam time was measured too, the flux can easily be calculated by dividing the measured fluence by the run time.

It turned out that the shielding of the two FPGAs, which were assumed to be reliable (second Virtex-4 and ProASIC3e device), was not sufficient. During the test, SEUs also occurred in these devices from time to time, most likely due to neutron scattering. Two basic failure modes were observed that indicated upsets in these devices: (i) the ProASIC3e device stopped the transmission of status information to the host PC and (ii) the SPWRTC board did not receive any images back from the second Virtex-4 FPGA. To circumvent this unforeseen issue, the whole system was connected to a power supply that was switchable from the control room. By doing so, the test could be manually stopped every time one of the aforementioned failure modes occurred and the system was power cycled. Then, the system was set up again and the test restarted. Therefore, the test result data presented in the following was gathered from several test runs. The first two test runs no. 1 and no. 2 are not taken into account since the aforementioned issue was detected during these runs and thus the gathered results must be assumed to be (partly) wrong. Also test run no. 8 was skipped because a power cycle was already necessary after the detection of two failures, i.e. the sample size was too small to be taken into account for further data analysis.

B. Static SEU Characterisation

After each failure detection, the bitstream was read back from the Virtex-4 device, transmitted to the host PC via SpaceWire and stored to hard disk. A custom-built tool was developed for post analysis of all bitstreams. First, the readback bitstreams are aligned to a golden bitstream (*.bit file) and a masking file (*.msk file) generated by the Xilinx toolchain. Then, the files are compared byte-wise. To do so, the readback file is XORed with the bitstream file and the inverted mask file is applied by an AND operation. The resulting byte contains logical 1s at the positions where an upset occurred. If the byte is not zero, the algorithm steps through all bits of the byte to identify the exact bit position of the upset. For this bit position, the Frame Address Register (FAR) address of the corresponding frame is determined. Since the tool is aware of the FPGA's internal memory structure, it is able to determine if the upset occurred in the area of the JPEG stream

TABLE II. NUMBER OF SEUs IN DIFFERENT MEMORY BLOCKS

Experiment	SEUs JPEG	SEUs FPGA	all	SEUs Type 0	SEUs Type 1	SEUs Type 2
200 MeV 4.2E+06	2606	16938		9244	1484	6225
200 MeV 8.3E+06	1796	11762		6509	963	4290
100 MeV 8.5E+06	1970	12860		6891	1161	4808
All	6372	41560		22644	3608	15308

TABLE III. NUMBER OF BITS IN GOLDEN BITSTREAM FILE AND MASKING FILE

Block	Configuration Bits	Masking Bits	Comparable Bits
Total FPGA	22702808	2936184	19766624
JPEG	3873024	153072	3719952
Type 0	13844224	539344	13304880
Type 1	2120192	74240	2045952
Type 2	6738392	2322600	4415792

processor, in Type 0, Type 1 or Type 2 blocks. Type 0 blocks comprise CLBs, IOBs and DSPs. Type 1 blocks comprise bits responsible for Block RAM interconnect. Type 2 blocks comprise the Block RAM bits.

Each bitstream that was read back from the device contains several SEUs. This is due to the fact that many bits are not used in the design and that it therefore takes some time until a sensitive bit is hit by a particle, which eventually causes a measurable failure. In the meanwhile, several SEUs accumulate in the configuration memory. We use this fault accumulation to our advantage and calculate the static cross-sections of the device with it. Since the aforementioned tool can distinguish between Type 0, 1 and 2 blocks, separate cross-sections for both the CLBs and the Block RAMs can be calculated. Memory content of Block RAMs actually used by the design cannot be analysed as it is masked out by the masking file. However, around two-thirds of the Block RAMs are not used by the design and were thus available as on-chip radiation detectors. An overview of the number of detected SEUs for the different configuration memory blocks can be found in Table II.

The aforementioned bitstream analysis tool also determined the number of bits in the different blocks of the golden bitstream file and the masking file, the results can be found in Table III. In the following, the number of comparable bits is of interest, which is the number of configuration bits minus the number of masking bits.

Using the figures from Table II and III, the cross-section of a specific memory block type can easily be determined by dividing the number of SEUs by the measured fluence. Then, the cross-section per bit is calculated by dividing the cross-section value of the memory block by the number of comparable bits within this memory block.

To validate the proposed availability analysis method (see Section IV) two cross-sections are of particular interest. The first cross-section covers the configuration memory bits (CLBs, IOBs, DSPs, Block RAM Interconnect etc.) whereas the sec-

TABLE IV. CROSS-SECTION: CONFIGURATION MEMORY (TYPE 0/1)

Experiment	Fluence [p/cm ²]	Runtime [s]	SEUs	X-Section [cm ²]	X-Section per bit [cm ² /bit]
200 MeV 4.2E+06	3.82E+10	9176	10728	2.81E-07	1.83E-14
200 MeV 8.3E+06	2.67E+10	3207	7472	2.80E-07	1.83E-14
100 MeV 8.5E+06	3.42E+10	4028	8052	2.36E-07	1.54E-14

TABLE V. CROSS-SECTION: BLOCK RAM (TYPE 2)

Experiment	Fluence [p/cm ²]	Runtime [s]	SEUs	X-Section [cm ²]	X-Section per bit [cm ² /bit]
200 MeV 4.2E+06	3.82E+10	9176	6225	1.63E-07	3.69E-14
200 MeV 8.3E+06	2.67E+10	3207	4290	1.61E-07	3.64E-14
100 MeV 8.5E+06	3.42E+10	4028	4808	1.41E-07	3.19E-14

ond cross-section is for the Block RAM memory cells. For all experiments, the cross-sections are given in Table IV and V.

C. Dynamic Testing: Experiment No. 1

Test runs 3, 4 and 7 to 12 were conducted at 200 MeV with a flux of ca. $4.2 \cdot 10^6$ p/cm²·s. The overall run time was 9176 s and 439 failures were detected in total. Table VI gives an overview of the individual test runs.

After each failure detection, the FDIR supervisor software transmitted a status message to the host PC, which contained the time that elapsed since the last failure detection. Averaging these time values results in the Mean Time Between Failure (MTBF) figure for a specific test run. An overview of the calculated MTBF values can be found in Table VII.

The averaged MTBF value can now be compared to the predicted MTBF value that is based on our estimation of sensitive configuration memory and Block RAM elements. Therefore, this most crucial part of the proposed availability analysis method can indirectly be validated.

The random fault injection campaign predicted around 526,300 sensitive configuration memory bits within the JPEG stream processor, see Section IV-A. The Block RAM profiling tool determined 67,858 sensitive Block RAM bits used as RAMs and 27,351 sensitive Block RAM bits used as ROMs,

TABLE VI. OVERVIEW: 200 MeV, $4.2 \cdot 10^6$ p/cm²·s

Run	Fluence [p/cm ²]	Run Time [s]	Flux [p/cm ² ·s]	# Failures
3	3.70E+09	889	4.16E+06	49
4	1.00E+10	2410	4.15E+06	118
7	3.84E+09	910	4.22E+06	44
9	4.10E+09	991	4.14E+06	44
10	1.16E+09	275	4.22E+06	13
11	6.80E+09	1639	4.15E+06	63
12	8.60E+09	2062	4.17E+06	108
Total:	3.82E+10	9176	4.16E+06	439

TABLE VII. MTBF VALUES: 200 MeV, $4.2 \cdot 10^6$ p/cm²·s

Run	MTBF [s]
3	17.15010417
4	19.71884615
7	20.2452907
9	21.28947674
10	20.46020833
11	24.83907258
12	19.03834112
Overall mean:	20.22905671

TABLE VIII. OVERVIEW: 200 MeV, $8.3 \cdot 10^6$ p/cm²·s

Run	Fluence [p/cm ²]	Run Time [s]	Flux [p/cm ² ·s]	# Failures
5	1.70E+09	203	8.37E+06	2
6	2.80E+09	333	8.41E+06	37
13	8.22E+09	994	8.27E+06	91
14	2.02E+09	246	8.21E+06	28
15	1.92E+09	231	8.31E+06	27
16	1.00E+10	1200	8.33E+06	140
Total:	2.67E+10	3207	8.31E+06	343

see Section IV-B. Using the measured cross-sections, the MTBF value is predicted as follows:

$$\text{MTBF} = [4.16 \cdot 10^6 \text{p/cm}^2 \cdot \text{s} \cdot (1.83 \cdot 10^{-14} \text{cm}^2/\text{bit} \cdot 526300 \text{bits} + 3.69 \cdot 10^{-14} \text{cm}^2/\text{bit} \cdot (67858 + 27351) \text{bits})]^{-1} = 18.275 \text{s/p} \quad (3)$$

As can be seen, the error of the prediction is low. In relation to the measured value of 20.23 s, the predicted value of 18.28 s is off by only 9.7%.

D. Dynamic Testing: Experiment No. 2

Test runs 5, 6 and 13 to 16 were conducted at 200 MeV with a flux of ca. $8.3 \cdot 10^6$ p/cm²·s. The overall run time was 3207 s and 343 failures were detected in total. Table VIII gives an overview of the individual test runs.

The calculated MTBF values for all test runs can be found in Table IX. Again, the averaged MTBF value can be compared to the predicted MTBF figure:

TABLE IX. MTBF VALUES: 200 MeV, $8.3 \cdot 10^6$ p/cm²·s

Run	MTBF [s]
5	9.822631579
6	8.253333333
13	10.69113889
14	8.05037037
15	8.660288462
16	8.59028777
Overall mean:	9.146973294

TABLE X. OVERVIEW: 100 MeV, $8.5 \cdot 10^6$ p/cm²·s

Run	Fluence [p/cm ²]	Run Time [s]	Flux [p/cm ² ·s]	# Failures
17	8.99E+09	1100	8.17E+06	81
18	2.83E+09	330	8.58E+06	24
19	5.76E+09	669	8.61E+06	44
20	1.00E+10	1164	8.59E+06	92
21	6.57E+09	765	8.59E+06	68
Total:	3.42E+10	4028	8.48E+06	309

TABLE XI. MTBF VALUES: 100 MeV, $8.5 \cdot 10^6$ p/cm²·s

Run	MTBF [s]
17	12.9291875
18	13.61826087
19	14.05593023
20	12.16035714
21	11.14865672
Overall mean:	12.51813322

$$\text{MTBF} = [8.31 \cdot 10^6 \text{p/cm}^2 \cdot \text{s} \cdot (1.83 \cdot 10^{-14} \text{cm}^2/\text{bit} \cdot 526300 \text{bits} + 3.64 \cdot 10^{-14} \text{cm}^2/\text{bit} \cdot (67858 + 27351) \text{bits})]^{-1} = 9.185 \text{s/p} \quad (4)$$

This time, the predicted value of 9.185 s matches the measured value of 9.147 s nearly perfectly, with an error of only 0.4%.

E. Dynamic Testing: Experiment No. 3

Test runs 17 to 21 were conducted at 100 MeV with a flux of ca. $8.5 \cdot 10^6$ p/cm²·s. The overall run time was 4028 s and 309 failures were detected in total. Table X gives an overview of the individual test runs and the calculated MTBF values can be found in Table XI.

The averaged MTBF value is again compared to the estimated MTBF value:

$$\text{MTBF} = [8.48 \cdot 10^6 \text{p/cm}^2 \cdot \text{s} \cdot (1.54 \cdot 10^{-14} \text{cm}^2/\text{bit} \cdot 526300 \text{bits} + 3.19 \cdot 10^{-14} \text{cm}^2/\text{bit} \cdot (67858 + 27351) \text{bits})]^{-1} = 10.586 \text{s/p} \quad (5)$$

For this experiment, the prediction error is 15.4% and is thus slightly greater than the error figures seen in the previous experiments.

F. Availability Analysis

During the beam test, the instrument simulator software was counting the transmitted images and the correctly received images. Thus, the availability could be measured by dividing the number of correctly received images by the number of transmitted images. The results are given in Table XII.

Using the measured MTBF values, given in the previous sections, the steady-state availability can also be predicted with stochastic Petri nets. Since Duplication with Comparison

TABLE XII. MEASURED AVAILABILITY DURING TEST CAMPAIGN

Experiment	TX Images	RX Images	Availability
200 MeV 4.2E+06	80724	77621	0.961560
200 MeV 8.3E+06	28586	26266	0.918841
100 MeV 8.5E+06	35845	33741	0.941303

TABLE XIII. PREDICTED AVAILABILITY

Experiment	Availability using measured MTBF	Error [%]	Availability using predicted MTBF	Error [%]
200 MeV 4.2E+06	0.96388	0.2	0.96017	0.1
200 MeV 8.3E+06	0.92347	0.5	0.92377	0.5
100 MeV 8.5E+06	0.94290	0.2	0.93318	0.9

was employed as the redundancy scheme, the Petri net shown in Figure 6 was used. Instead of multiple exponential transitions `mod_seu_x`, only one transition is necessary, which is triggered with the averaged MTBF value for each test experiment. Two more parameters are needed: the exponential detection time `mod_detect` and the deterministic repair time `mod_rep`. The average detection time was assumed to be 50 ms, which corresponds roughly to the processing time of one image. The average repair time was measured during the beam test as 708 ms. This time span is rather long as it also includes the time required to read back the bitstream.

The availability can also be predicted using the MTBF values that were estimated in the previous sections (see Equations 3 to 5). The results of the availability prediction for both cases are listed in Table XIII. If the measured MTBF value is used, the predicted availability matches the measured availability extremely well with an error of 0.5% or less. This shows that the stochastic Petri net model is well designed, although it must be pointed out that its output is a bit too optimistic.

If the predicted MTBF value is used instead, the predicted availability matches the measured availability quite closely. This result is very positive as it shows that the quantification of sensitive memory elements does not need to be extremely precise. The estimated MTBF value, based on the fault injection experiments and the Block RAM profiling method, differed by up to nearly 16% compared to the measured values. However, since the MTBF is so much longer than the average repair time, the error of the MTBF estimation plays only a negligible role in the availability estimation. This outcome strongly supports the idea of the proposed availability analysis method, which claims that a very good availability estimation can be achieved with a rather simple and coarse-grain approach to the quantification of sensitive memory elements.

VII. CONCLUSIONS

It can be concluded that the accelerated proton irradiation campaign was a full success, validating the theoretical approach. First, it was demonstrated that the FDIR hardware and software components were capable to detect and recover

from failures in a real radiation environment that causes higher SEU rates than any solar particle event observed in history. Secondly, it was indirectly proven that the estimation of the number of sensitive bits (via fault injection experiments and Block RAM profiling) was quite accurate. The predicted value of the MTBF measured during the beam test differed by not more than 15.4%. Even more interestingly, it was shown that despite this difference a much better prediction of the steady state availability was feasible disagreeing with the actual results by only 0.9% or less. Therefore, both the proposed Distributed Failure Detection technique and the availability analysis method were successfully validated.

ACKNOWLEDGMENTS

Sponsorship from ESA under the NPI Programme, Airbus Defence and Space, UK and the University of Leicester is gratefully acknowledged.

REFERENCES

- [1] K. Paulsson, M. Hubner, M. Jung, and J. Becker, "Methods for run-time failure recognition and recovery in dynamic and partial reconfigurable systems based on Xilinx Virtex-II Pro FPGAs," in *IEEE Computer Society Annual Symposium on Emerging VLSI Technologies and Architectures*, 2006, vol. 00, Mar. 2006, p. 6 pp.
- [2] A. Jacobs, A. George, and G. Cieslewski, "Reconfigurable fault tolerance: A framework for environmentally adaptive fault mitigation in space," in *International Conference on Field Programmable Logic and Applications*, 2009. *FPL 2009*, 2009, pp. 199–204.
- [3] D. McMurtrey, K. S. Morgan, B. Pratt, and M. J. Wirthlin, "Estimating TMR reliability on FPGAs using Markov models," 2008. [Online]. Available: <http://contentdm.lib.byu.edu/cdm/ref/collection/IR/id/612>
- [4] 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.
- [5] J. Kastil, M. Straka, L. Miculka, and Z. Kotasek, "Dependability analysis of fault tolerant systems based on partial dynamic reconfiguration implemented into FPGA," in *15th Euromicro Conference on Digital System Design (DSD)*, 2012, pp. 250–257.
- [6] Q. Martin and A. George, "Scrubbing optimization via availability prediction (SOAP) for reconfigurable space computing," in *IEEE Conference on High Performance Extreme Computing (HPEC)*, 2012, pp. 1–6.
- [7] F. Siegle, T. Vladimirova, J. Ilstad, and O. Emam, "Mitigation of radiation effects in SRAM-based FPGAs for space applications," *ACM Comput. Surv.*, vol. 47, no. 2, pp. 37:1–37:34, Jan. 2015.
- [8] F. Siegle, T. Vladimirova, O. Emar, and J. Ilstad, "Adaptive FDIR framework for payload data processing systems using reconfigurable FPGAs," in *8th NASA/ESA Conference on Adaptive Hardware and Systems (AHS)*, 2013, pp. 15–22.
- [9] —, "New voter design enabling hot redundancy for asynchronous network nodes," in *9th NASA/ESA Conference on Adaptive Hardware and Systems (AHS)*, 2014.
- [10] G. Allen, G. Swift, and C. Carmichael, "Virtex-4QV static SEU characterization summary," NASA Jet Propulsion Laboratory (JPL), Xilinx, Tech. Rep., 2008.
- [11] ECSS, "Space engineering: Space environment. ECSS-E-ST-10-04C," ESA/ESTEC, Tech. Rep., 2008.
- [12] A. Zimmermann, "Modeling and evaluation of stochastic Petri nets with TimeNET 4.1," in *6th International Conference on Performance Evaluation Methodologies and Tools (VALUETOOLS)*, 2012, pp. 54–63.
- [13] Aeroflex Gaisler, "SPWRTC development unit. User manual revision 1.1," Aeroflex Gaisler AB, Tech. Rep., 2008.