Accepted in *Reliability Engineering & System Safety*, 2020, DOI: 10.1016/j.ress.2020.107036

Analytical Reliability Estimation of SRAM-based FPGA Designs against Single-bit and Multiple-cell Upsets

Reza Ramezani[1], Juan Antonio Clemente[2], Francisco J. Franco[3]

r.ramezani@eng.ui.ac.ir, juanancl@ucm.es, fjfranco@fis.ucm.es

[1] Faculty of Computer Engineering, University of Isfahan, Isfahan, Iran
[2] Computer Architecture Department, Universidad Complutense de Madrid, Madrid 28040, Spain
[3] Department of Structure of Matter, Thermal Physics, and Electronics and Institute of Particle and Cosmos Physics (IPARCOS), Facultad de Ciencias Físicas, Universidad Complutense de Madrid (UCM), Spain

***Abstract*- This paper addresses the problem of hardware tasks reliability estimation in harsh environments. A novel statistical model is presented to estimate the reliability, the mean time to failure, and the number of errors of hardware tasks running on static random-access memory (SRAM)-based partially run-time reconfigurable field programmable gate arrays (FPGAs) in harsh environments by taking both single-bit upsets and multiple-cell upsets into account. The model requires some features of the hardware tasks, including their computation time, size, the percent of critical bits, and the soft error rates of k-bit events ($k \geq 1$) of the environment for the reliability estimation. Such an early estimation helps the developers to assess the reliability of their designs at earlier stages and leads to reduce the development cost. The proposed model has been evaluated by conducting experiments on actual hardware tasks over several environmental soft error rates. The obtained results, endorsed by the 95% confidence interval, reveal the high accuracy of the proposed model. When comparing this approach with a reliability model (developed by the authors in a previous work) that does not consider the occurrence of multiple-cell upsets, an overestimation of the mean time to failure of 2.88X is observable in the latter. This points to the importance of taking into account multiple events, especially in modern technologies where the miniaturization is high.**

Keywords: Reliability Model, Multiple Cell Upsets, Soft Errors, Hardware Tasks, FPGA-based Designs.

I. Introduction

Static random-access memory (SRAM)-based Field Programmable Gate Arrays (FPGAs) feature low-cost and offer an interesting trade-off between performance and flexibility. Such FPGAs can be reconfigured at runtime to execute different functionalities in a time-multiplexed manner [1]. Therefore, they have found many applications in space computing systems [2], where space qualified software and devices are typically used to guarantee data integrity and a correct functionality [3]. However, Commercial-Off-The-Shelf (COTS) devices are an affordable alternative to rad-hard devices since they implement error detection and correction mechanisms [4]. For example, the Los Alamos National Laboratory satellite CFESat has used 9 COTS FPGAs as part of their satellite high-performance computing payload [5]. The ability of FPGAs to reconfigure logic resources is extremely valuable in a spacecraft since such a reconfiguration supports the ability to upgrade satellite electronics, exploit in-system reconfiguration, or create a design that avoids permanent failures in a device.

However, the negative effects of natural radiations, such as cosmic rays coming from outer space and subsequent cascades of secondary particles (such as neutrons and alpha particles), make nanoscale devices, and in particular SRAMs, vulnerable against soft errors. The negative effect of said radiations is the occurrence of the so-called Single Event Upsets (SEUs), which is a broad category of events by which a single particle strike eventually causes memory cell upsets or bitflips [6]. The single-bit upset (SBU) is the most common consequence when a particle hits the memory. Nevertheless, by shrinking the size of memory cells, each particle hit might affect several adjacent memory cells simultaneously, which is known as multiple-cell upset (MCU). Therefore, despite being flexible and having a good performance, SRAM-based FPGAs suffer from the susceptibility to different kinds of radiation-induced soft errors [7]. Thus, developers have to estimate the operational reliability of their designs before the mission.

Different approaches, such as accelerated radiation tests and fault injection campaigns, have been presented so far to estimate the reliability of hardware tasks running on SRAM-based FPGAs before the operation. The former is very costly and might lead to physical damage of the device, whereas the latter suffer from the time complexity. Analytical models are alternatives to estimate the reliability of hardware tasks while overcoming the drawbacks of said approaches. Nevertheless, the existing models only consider SBUs, and they do not take the effects of MCUs into account.

In this paper, a novel analytical reliability model has been proposed to estimate the reliability of hardware tasks running on SRAM-based FPGAs in harsh environments in early design stages. In contrast to existing models, it targets both SBUs and MCUs of diverse multiplicities that affect the configuration memory of the hardware tasks. The presented model incorporates some features of hardware tasks and the environment, including task computation time, task size, the percent of task critical bits, and the Soft Error Rates (SERs) of k-bit events of different multiplicities. Then, simple equations are presented to estimate the reliability, the mean time to failure (MTTF), and the number of errors of a task during execution. The validity of the proposed model has been examined by conducting experiments on several hardware tasks running on an SRAM-based FPGA in environments with different SERs. The obtained results show that the proposed model is accurate since the estimations fall within the 95% confidence interval of the experiments. It will be demonstrated that this method further refines the previous study of the authors, where only single-bit events were considered.

The rest of the paper is organized as follows. Section 2 introduces some related studies. Section 3 presents the system model, including hardware tasks and the target SRAM-based FPGA that has been used as case study. Section 4 elaborates on the proposed reliability model and its validation. Section 5 gives the experimental evaluations and results, and finally, Section 6 concludes the paper and offers suggestions for future work.

II. Related Work

Accelerated radiation tests [8] and fault-injection campaigns [9] are two well-known strategies to estimate the reliability of FPGA-based designs against single-bit and multiple-cell upsets. The first approach is costly and might damage the device-under-test permanently. In contrast, fault injection approaches provide a higher level of controllability to where and how the fault is emulated. For example, the failure rate of FPGA-based designs has been estimated for both SBUs and MCUs by injecting faults on adjacent physically cells of the configuration memory [10]. Although these methods are useful for reliability estimation purposes, they incur into significant delays since the device must be accessed externally in order not to be intrusive.

Analytical models are optimal alternatives to lower the computation time required for fault injection campaigns and to avoid the high costs of the accelerated radiation tests. These models assess the reliability of the hardware tasks at an earlier stage in the design cycle, which results in the reduction of the design effort while increasing the design confidence.

Markov-chain models are one kind of analytical approaches to estimate the reliability of designs running on FPGAs in which the applications are modeled as data flow graphs, representing the dependency of the application modules. For example, the solution proposed by [11] estimates the reliability of FPGA-based designs using Markov-chain models where the SBUs are the object of concern. The solution proposed in [12] splits a design into multiple partitions each of which hardened with Triple Modular Redundancy (TMR) and scrubbing techniques. Then, it uses a Markov-chain model to estimate the reliability and availability of the design against single-bit and double-cell upsets. Since the MCUs might affect multiple partitions of a design, these models become very complex when the multiplicity of the upsets increases.

Statistical methods are alternative analytical models which have been used by researchers to estimate the reliability of hardware tasks in the presence of soft errors [13]. For example, Heron et al. [14] have introduced a micro-view reliability model which combines soft error sensitivity and physical reliability. This model is based upon estimating the reliability of a design from that of basic elements with the assumption that the reliability of the basic elements is known. In a similar micro view, the reliability of FPGA-based designs is estimated by calculating the error propagation probability of the low-level gates [15].

The model presented by Ostler et al. [16] is another study in this area that estimates the reliability of hardware tasks for different orbits and orbit conditions, during a single configuration scrubbing period, based on a design-specific parameter. This parameter should be obtained by using fault injection or accelerated radiation experiments. In a similar approach [17], the authors have presented a statistical reliability model that uses some features of the hardware tasks and the SER of the environment to estimate the reliability of hardware tasks against SBUs.

Another statistical model has been presented in [18] to estimate the reliability of TMR-hardened FPGA-based designs with different configuration read-back strategies. This model takes critical bits into account and has been evaluated using fault injection experiments. In a similar approach, a statistical model has been presented by [19] that by using the information of the configuration resources and computation time of the tasks, estimates the failure rate of TMR-hardened designs as a function of the SEU rate, the number of mitigation windows, and onboard processor shield thickness. Nevertheless, all these works pay attention to SBUs, and do not take the effects of MCUs into account. A general overview of the proposed reliability models is presented in Table 1.

Table 1 - An overview of reliability models proposed in the literature

Ref	Year	Accelerated radiation	Fault injection	Analytical: formal method	Analytical: statistical method	SBUs	MCUs
[8]	2018	✓	×	×	×	✓	✓
[9]	2018	×	✓	×	×	✓	✓
[10]	2017	×	✓	×	×	✓	✓
[11]	2017	×	×	✓	×	✓	×
[12]	2019	×	×	✓	×	✓	✓
[13]	2018	×	×	×	✓	✓	×
[14]	2005	×	×	×	✓	✓	×
[15]	2007	×	×	×	✓	✓	×
[16]	2009	×	×	×	✓	✓	×
[17]	2016	×	×	×	✓	✓	×
[18]	2018	×	×	×	✓	✓	×
[19]	2019	×	×	×	✓	✓	×
Our Solution	2020	×	×	×	✓	✓	✓

Although analytical models are very useful at estimating the reliability of hardware tasks at early design stages, to the best of our knowledge, none of the presented solutions takes the effects of MCUs into account when estimating the reliability of non-hardened FPGA-based designs. In this paper, a new statistical-based analytical reliability estimation model is presented that targets soft errors that cause SBUs and MCUs within the configuration memory. The proposed solution uses some features of the hardware tasks and the SERs of different k-bit events to estimate the initial reliability, the MTTF, and the number of errors of FPGA-based designs.

III. System Model

In this work, in order to estimate the reliability of hardware tasks, four parameters, namely task computation time, task size, the percent of task critical bits, and the SERs for events with different multiplicities have been used. As the task size depends on the underlying FPGA, in this section, both hardware tasks and the target SRAM-based FPGA have been modeled.

A) Hardware Task Model

A reconfigurable application, known as *hardware task*, is composed of a number of logic circuits that cooperate in implementing specific functionality. In this paper, each hardware task τ is modeled as follows:

$$\tau = \{CT_\tau, CC_\tau, TS_\tau, CB_\tau\} \qquad (1)$$

where CT_τ is task computation time, CC_τ is the number of basic reconfigurable elements, and TS_τ is task size in the configuration memory (number of configuration bits). The computation time of tasks is expressed in terms of milliseconds, which is consistent with typical delays of tasks running on SRAM-based FPGAs [20]. Task size TS_τ depends on CC_τ and some features of the underlying FPGA that will be discussed in the next subsection. Design tools can easily obtain these values for a given hardware task τ. Finally, CB_τ represents the percentage of critical bits of the task.

Any bitflip in a critical bit will eventually affect the functionality of the corresponding task and leads to a failure [21]. The criticality of hardware tasks' bits depends on different factors, such as the way of placing the design in configuration memory and the architecture of the target device. The critical bits of a hardware task can be obtained by fault injection campaigns or radiation-ground experiments [22]. However, a first-order estimation of the essential and critical bits percent can be obtained instead, using task characterization libraries [12, 23].

Since the validity of the presented model has been verified on a Virtex-5 FPGA, in this case, the basic reconfigurable elements are *Configurable Logic Blocks* (CLBs). However, this idea can be easily ported to any other FPGA in the market, since all of them feature any sort of basic reconfigurable elements, such as *Logic Elements* in Altera architectures.

B) Partially Run-time Reconfigurable FPGA Model

The target FPGA features partial run-time reconfigurability, and it includes an array of CLBs so that hardware tasks are synthesized and mapped on a subset of them. Typically, in modern FPGAs, a single basic element is not the minimum addressable segment of the device that can be reconfigured separately. Instead, a set of them must be reconfigured at the same time, which in this paper is referred to as the *CLB group* [24]. Thus, the target SRAM-based FPGA SF is modeled as:

$$\mathrm{SF} = (RO, CO, GS, CB) \qquad (2)$$

where RO and CO indicate row count and column count of the FPGA, respectively. GS stands for CLB group size (in terms of CLB count in each group), and finally, CB is the number of configuration bits that a CLB group contains.

SRAM-based FPGAs allow reconfiguring only a fraction of the resources at runtime, while the remainder of the device continues its normal operation. Related studies on the area of the task configuration typically distinguish between several area models. The area model describes the way in which the hardware tasks are arranged within the FPGA. They mostly assume 1D or flexible 2D organizations [25]. In the 1D area model, hardware tasks have rectangular footprints in a way that they span the entire device height, but they have different widths. On the contrary, in the flexible 2D area model, hardware tasks are modeled as rectangles that can span a rectangular subset of resources within the FPGA. The actual flexibility of the 2D model for each device is determined by its internal architecture and organization.

Task size (TS_τ) –in terms of configuration bits– depends on the FPGA area model. In the 1D area model, let CO_τ be the number of CLBs that the task τ spans horizontally (CO_τ can be easily obtained from CC_τ). Thus, TS_τ is a multiple of CO_τ. In addition, since GS is the minimum addressable set of resources, TS_τ can be obtained as follows:

$$TS_\tau = CO_\tau \times \frac{RO}{GS} \times CB \qquad (3)$$

On the other hand, in FPGAs implementing a 2D-based area model, let RO_τ be the number of CLBs that the task spans vertically (note that, for the 1D area model, $RO_\tau = RO$). Thus, in this case, TS_i can be easily calculated as follows:

$$TS_\tau = CO_\tau \times \left\lceil \frac{RO_\tau}{GS} \right\rceil \times CB \qquad (4)$$

Task size TS_τ affects the reliability of the task. This point will be elaborated in the next section.

IV. Reliability Model and its Validation

A) The Proposed Reliability Model

In the general formulation, reliability is the probability that a system performs correctly during a specific time duration. Reliability follows an exponential failure law, which means that it is gradually reduced over time due to external agents, such as radiation. The reliability of a hardware task depends on its error rate. By assuming that ρ is the error rate, the reliability of a hardware task at time t is obtained as: $R(t) = e^{-\rho t}$ [26]. In this paper, this classical formulation has been extended to include soft error rates induced by events of diverse multiplicity by taking into account the task's size, computation time, and critical bits.

The model presented in this paper assumes that upsets accidentally occur in configuration memory of the running tasks, according to given SERs. In contrast to existing statistical models, in this work, it is assumed that each event might affect one or more configuration bits simultaneously to model both SBUs and MCUs. Thus, there exist M different SERs for events causing k-bit upsets ($1 \leq k \leq M$), M being the highest multiplicity that will be considered. Then, the proposed model estimates the reliability, MTTF, and the number of errors (#Errors) that are expected to observe in said tasks. The SERs can be measured *per event per bit per time unit*. The reliability of a task τ (denoted as R_τ) is the probability that τ executes from its start time to its finish time without any failure, with the condition that it is fault-free when starting its execution. In this regard, it is supposed that a task does not have any error at the time point that it is completely configured.

The proposed model assumes that, at most, one event occurs at a time, but one or more events might occur during task execution. In other words, upsets caused by radiation can be regarded as independent and random statistical events. Thus, it is reasonable to make the following assumptions:

(1) The number of events during a given interval of time depends only on the length of the interval and not on the past history of the system.

(2) For any small interval $(t, t + \delta t)$, the probability of an k-bit event is $\rho_k \cdot \delta t$ wehere ρ_k is a constant (the SER of the k-bit event) and the probability of more than one event occurring simultaneously is negligible.

Based on these assumptions, the upsets caused by radiation follow the Poisson distribution. Let us assume that a task is exposed with given SERs, ρ_1 (i.e., the number of expected SBUs per bit per time unit), ρ_2 (2-bit event SER), …, ρ_k (k-bit event SER). The values of ρ_k can be easily estimated for different devices using existing tools [10]. Therefore, following the general reliability formulation, the probability of an k-bit event occurring m times during the task execution (and, therefore, affecting totally $m \cdot k$ bits) is[1]:

$$P_{EV}(k, m) = \frac{{\mu_k}^m}{m!} \cdot e^{-\mu_k} \qquad (5)$$

where

$$\mu_k = \rho_k \cdot TS_\tau \cdot CT_\tau \qquad (6)$$

in which ρ_k is the k-bit event environmental SER expressed *per bit per time unit*. Thus, the rate of the k-bit event of the task (μ_k) increases with task size and task computation time [28]. For practical reasons, we will accept that the occurrence of k-bit events is negligible for $k > M$.

Let $PBF(n)$ be the probability of observing n bitflips in the configuration memory of the task. Thus, the probability of observing 0 bitflips in the memory is the probability of all the kinds of events being simultaneously null. Let us define the total SER of the task as $\mu_T = \mu_1 + \mu_2 + \cdots + \mu_M$. Therefore:

$$PBF(0) = P_{EV}(1,0) \cdot P_{EV}(2,0) \cdot \cdots \cdot P_{EV}(M,0) = e^{-(\mu_1+\mu_2+\cdots+\mu_M)} = e^{-\mu_T}$$

The probability of occurring 1 bitflip is the probability of occurring 1 SBU and 0 MCUs:

$$PBF(1) = P_{EV}(1,1) \cdot P_{EV}(2,0) \cdot \cdots \cdot P_{EV}(M,0) = \frac{\mu_1}{1!} e^{-\mu_1} \cdot e^{-\mu_2} \cdot \ldots \cdot e^{-\mu_M} = e^{-\mu_T} \cdot \frac{\mu_1}{1!}$$

The probability of observing 2 bitflips is the probability of observing A) 2 SBUs and 0 MCUs plus the probability of observing B) 1 2-bit MCU and 0 events of the rest of classes:

$$PBF(2)_A = P_{EV}(1,2) \cdot P_{EV}(2,0) \cdot \cdots \cdot P_{EV}(M,0) = \frac{{\mu_1}^2}{2!} e^{-\mu_1} \cdot e^{-\mu_2} \cdot \ldots \cdot e^{-\mu_M} = e^{-\mu_T} \cdot \frac{{\mu_1}^2}{2!}$$

$$PBF(2)_B = P_{EV}(1,0) \cdot P_{EV}(2,1) \cdot \cdots \cdot P_{EV}(M,0) = e^{-\mu_1} \cdot \left(\frac{\mu_2}{1!} e^{-\mu_2}\right) \cdot \ldots \cdot e^{-\mu_M} = e^{-\mu_T} \cdot \frac{\mu_2}{1!}$$

$$PBF(2) = PBF(2)_A + PBF(2)_B = e^{-\mu_T} \cdot \left(\frac{{\mu_1}^2}{2!} + \frac{\mu_2}{1!}\right)$$

The probability of observing 3 bitflips can also be modeled in the same way. There are 3 options:

A) 3 SBUs and 0 MCUs:

$$PBF(3)_A = P_{EV}(1,3) \cdot P_{EV}(2,0) \cdot \cdots \cdot P_{EV}(M,0) = \frac{{\mu_1}^3}{3!} e^{-\mu_1} \cdot e^{-\mu_2} \cdot \ldots \cdot e^{-\mu_M} = e^{-\mu_T} \cdot \frac{{\mu_1}^3}{3!}$$

B) 1 SBU, 1 2-bit event, and no more MCUs:

$$PBF(3)_B = P_{EV}(1,1) \cdot P_{EV}(2,1) \cdot P_{EV}(3,0) \cdot \cdots \cdot P_{EV}(M,0) = \left(\frac{\mu_1}{1!} e^{-\mu_1}\right) \cdot \left(\frac{\mu_2}{1!} e^{-\mu_2}\right) \cdot e^{-\mu_3} \cdot \ldots \cdot e^{-\mu_M} = e^{-\mu_T} \cdot \frac{\mu_1}{1!} \cdot \frac{\mu_2}{1!}$$

C) 1 3-bit event and nothing else:

$$PBF(3)_C = P_{EV}(1,0) \cdot P_{EV}(2,0) \cdot P_{EV}(3,1) \cdot P_{EV}(4,0) \cdot \cdots \cdot P_{EV}(M,0) = e^{-\mu_1} \cdot e^{-\mu_2} \cdot \left(\frac{\mu_3}{1!} e^{-\mu_3}\right) \cdot e^{-\mu_4} \cdot \ldots \cdot e^{-\mu_M} = e^{-\mu_T} \cdot \frac{\mu_3}{1!}$$

Therefore, the total probability of observing 3 bitflips is:

$$PBF(3) = PBF(3)_A + PBF(3)_B + PBF(3)_C = e^{-\mu_T} \cdot \left(\frac{{\mu_1}^3}{3!} + \frac{\mu_1}{1!} \cdot \frac{\mu_2}{1!} + \frac{\mu_3}{1!}\right)$$

These expressions can be generalized for N bitflips as follows. Let $\alpha = (k_1, k_2, \cdots, k_M)$ be a solution of the Diophantine equation:

$$\sum_{i=1}^{M} i \cdot k_i = k_1 + 2 \cdot k_2 + \cdots + M \cdot k_M = N,$$

with $k_i \in \mathbb{N} \cup \{0\}$. The set of all the possible solutions α of the previous equation is A. For example, for $N = 1, A = \{(1)\}$, for $N = 2, A = \{(2,0), (0,1)\}$ and for $N = 3, A = \{(3,0,0), (1,1,0), (0,0,1)\}$.

Then, it is possible to demonstrate that:

$$PBF(N) = e^{-\mu_T} \cdot \sum_{\alpha \in A} \left\{ \prod_{i=1}^{N} \frac{{\mu_i}^{k_i}}{k_i!} \right\} \qquad (7)$$

[1] Experimental SEU and k-bit MCU cross-sections σ_k are another (yet very popular) alternative to measure the sensitivity of devices against radiation. A device cross-section is calculated by dividing the number of events (observed in all the device) by the particle fluence ϕ (typically, expressed as $particles \cdot cm^{-2}$). Thus, the unit of a cross-section is $\frac{events \cdot cm^2}{device \cdot particle}$, although it is usually expressed in terms of $\frac{events \cdot cm^2}{bit \cdot particle}$. For the sake of clarity, in the literature, it is usual to find cross-sections express in terms of $\frac{cm^2}{bit}$ (because it is well-known that the sensitivity is the number of events that were caused by a single particle). Anyway, if a memory with S bits, featuring a cross-section σ_k, is irradiated with a total flux of $\phi\ particle \cdot cm^{-2} \cdot S^{-1}$, the number of expected k-bit events for a given exposure time t is $\mu_k = \sigma_k \cdot S \cdot \phi \cdot t$. This is an alternative to compute the value of μ_k when cross-sections are known. In this work, we have used SER values for different k-bit events presented by [27].

Thus, with the assumption that an k-bit event provokes a failure if, at least, one of the k bitflips occurs in critical bits of the task (CB_τ), let $PTF(\tau)$ indicate the probability of task failure of τ. Therefore, we have a conditional probability:

$$PTF(\tau) = \sum_{k=1}^{\infty} PBF(k \mid CB_\tau) \qquad (8)$$

which it can be easily computed as:

$$PTF(\tau) = \sum_{k=1}^{\infty} PBF(k) \times (1 - (1 - CB_\tau)^k) \qquad (9)$$

By having $PTF(\tau)$, the reliability of task τ is obtained as:

$$R_\tau = 1 - PTF(\tau) \qquad (10)$$

Once the task reliability R_τ is achieved, the MTTF is calculated from [16]:

$$MTTF_\tau = \frac{CT_\tau}{1 - R_\tau} \qquad (11)$$

Finally, the number of errors ($\#Errors_\tau$), observed in the task, during the operating time of the system is estimated as:

$$\#Errors_\tau = \frac{LifeTime}{MTTF_\tau} \qquad (12)$$

where $LifeTime$ is the operating time of the system, which is taken to be one year in this paper. The obtained reliability R_τ can be combined with the ideas presented in [29] to obtain the reliability of a hardened application whose tasks have been replicated using an active redundancy-based FT technique.

B) Reliability Model Validation

The validation procedure follows a fault simulation approach and it takes two distinct statistical events into account. The first one is *the occurrence of an k-bit event during the execution of task τ among its configuration bits* (said event is composed of k bitflips occurring in physically adjacent memory cells). The second one is *the probability of, at least, one of these k bitflips occurring in one of the critical bits of task τ*.

The experimental environment runs task τ for $LifeTime$ time units; therefore, it is executed for $LifeTime/CT_\tau$ times. In the experiments, for all the time units in τ and for all the possible multiplicities, it is first determined if an k-bit event occurs. For this purpose, a metric named *Milliseconds to Upsets (k)* is used which stands for the time that on average it takes to occur an k-bit event in the configuration memory of task τ during its execution. The value of $Milliseconds\ to\ Upsets\ (k)$ is obtained from Eq. (13).

$$Milliseconds\ to\ Upsets\ (k) = \frac{CT_\tau}{PBF(k)} \qquad (13)$$

In order to determine if an k -bit event occurs, a random integer number (RND) is generated between 1 and $Milliseconds\ to\ Upsets\ (k)$ at each time unit. If RND = 1, it conventionally means the occurrence of such an event. Note that, by increasing k the value of $PBF(k)$ decreases, which makes $Milliseconds\ to\ Upsets\ (k)$ be higher. Therefore, this makes this event less probable to happen.

When an k-bit event occurs, k upsets are randomly injected in adjacent configuration bits of the task. If, at least, one of the faults flips a critical bit, it means a task failure. Thus, the error count is updated, and the time elapsed between this failure and the previous one is recorded as *Time Between Failure (TBF)*. When the lifetime finishes, $MTTF_\tau$ is calculated as follows:

$$MTTF_\tau = \frac{\sum TBF_\tau}{\#Errors_\tau} \qquad (14)$$

Finally, following Eq. (11), the reliability of task τ is calculated as:

$$R_\tau = 1 - \frac{CT_\tau}{MTTF_\tau} \qquad (15)$$

V. Experimental Results

A) Experimental setup

The proposed model has been validated by means of fault simulation-based experiments, which is a very widespread approach for validation of reliability models [18, 30, 31]. The validation process has been done on a system consisting of a PC and a Xilinx™ Virtex-5 XC5VLX110T FPGA. The occurrence of events has been simulated on the PC by generating random integer numbers. Once an event occurs, the faults are simulated in the FPGA. The FPGA contains 54 columns and 160 rows of reconfigurable units in which each CLB group contains 20 CLBs, and there are 8 CLB groups per column. Each CLB group contains 36 frames, and each frame has 40*32-bit words. Therefore, following Eq. (2), $SF = (160, 54, 20, 46080)$.

The proposed model has been evaluated with SERs obtained from radiation-ground experiments, which were performed by exposing a 90-nm COTS SRAM in a neutron source at various bias voltages [27]. It actually can be observed that the SER increases as the VCC decreases. These SERs have been expressed in terms of *#events per Mbit per day* (Table 2). However, other approaches [32] and tools [10] of calculating ρ_k can be used instead to obtain k-bit event SERs for other environmental conditions. In order to provide the SERs *per bit per millisecond*, the calculation of μ_k in Eq. (6) would be:

$$\mu_k = \left({\rho_k}/{2^{20}}\right) \times TS_\tau \times CT_\tau \times 24 \times 60 \times 60 \times 1000 \qquad (16)$$

Table 2 – SERs (ρ_k) in terms of #events per Mbit per day [27]

VCC	SBU (1-bit)	2-bit	3-bit	4-bit	5-bit	6-bit	Total ($\sum$)
0.5	8.451733276	2.53650725	1.86018488	0.72290743	0.36321596	0.234913458	14.16946226
0.6	4.798645383	1.30543311	1.06412351	0.41622946	0.22094389	0.132091734	7.937467089
0.7	3.52407113	1.00014542	0.80583963	0.34618006	0.18938048	0.10969127	5.975307996
0.8	2.755261795	0.75997813	0.62007518	0.27740829	0.14379133	0.085438702	4.641953418
0.9	2.326609831	0.65767134	0.54002332	0.24944296	0.13420524	0.070805371	3.97875806
1	1.841958985	0.54566241	0.4360424	0.20422587	0.10677492	0.053241159	3.187905746
1.1	1.499192293	0.42896975	0.33179456	0.16270026	0.0847988	0.039696242	2.547151904
1.2	1.37044563	0.39107183	0.2955772	0.14347154	0.07583143	0.039696242	2.316093877
1.3	1.313881407	0.36962248	0.2770928	0.12805318	0.06512609	0.039696242	2.193472196
2	1.233155316	0.35489733	0.27105387	0.12443517	0.06030291	0.035661597	2.079506201
3.3	1.199847355	0.32242924	0.26469329	0.1207631	0.05660542	0.035661597	2.0045536

ITC'99 is one of the best benchmark suites to assess the effects of radiations on hardware tasks [33]. In this paper, tasks b01 to b13 from ITC'99 as well as an FFE filter have been used for the experiments. For a clock speed of 100 MHz, they have been executed for different clock cycles to obtain different computation times in the range of [10 ... 500] *ms*. Details of these tasks are shown in Table 3. The critical bits have been obtained with the NESSY fault-injection tool, developed by the authors [34].

Table 3 - Details of the tasks [34]

Task	# Clock cycles	Computation time (*ms*)	# Total bits	# Critical bits	% Critical bits
b01	1.0E+06	10	46080	622	1.35%
b02	2.5E+06	25	46080	240	0.52%
b03	5.0E+06	50	46080	1719	3.73%
b04	7.5E+06	75	92160	6967	7.56%
b05	1.0E+07	100	138240	69	0.05%
b06	1.3E+07	125	46080	788	1.71%
b07	1.5E+07	150	46080	304	0.66%
b08	2.0E+07	200	46080	2852	6.19%
b09	2.5E+07	250	46080	1862	4.04%
b10	3.0E+07	300	46080	2871	6.23%
b11	3.5E+07	350	92160	7023	7.62%
b12	4.0E+07	400	138240	31215	22.58%
b13	4.5E+07	450	46080	3055	6.63%
FFE	5.0E+07	500	1797120	104233	5.8%

B) Reliability model validation

The experiments have been done for all the SERs of Table 2. Nevertheless, for the sake of clarity, only the results of the lowest VCC (0.5 V) and the highest one (3.3 V) are presented. The lifetime of each experiment is one year and 10 experiments were run for each task to simulate the tasks running for 10 years to obtain more robust results. Thus, as each year is 3.154E+10 *ms*, for each experiment (a given value for k) totally 3.154E+11 random integer numbers have been generated between 1 and $Milliseconds\ to\ Upsets\ (k)$ to simulate a k-bit event. Finally, the average of the 10 results has been presented. The results of the experiments on the FFE filter at VCC = 0.5 V have been depicted in Figure 1 to Figure 4.

In this experiment, 7 different cases have been examined: when only events of a given multiplicity k occur ($1 \leq k \leq 6$), and when events of all multiplicities can occur altogether. Figure 1 shows the #Errors obtained from both the experiments made by fault simulation (Section IV-B) and the estimations made with the proposed model (Eq. (5-12)). The #Errors of the experiments are shown by dotted blue lines, whereas the estimations are shown by red lines. As the results of the experiments are based on different trials, it is required to obtain their 95% confidence intervals to disclose how much the difference between estimations and experiments is reasonable. This is shown in the figure by means of the gray boxes, whose upper and lower legends indicate the upper and lower bounds of the confidence interval, respectively. It can be considered that the #Errors of the estimation is acceptable if it lies within the absolute 95% confidence interval of the corresponding experimental result. The larger the confidence interval is, the higher boxes have been drawn.

When #Errors > 50, said confidence intervals can be approximated with the normal distribution, as shown by Eq. (17) [35]. The relative interval can be easily converted to an absolute one as: *Experiment*±*Relative Interval*. However, when #Errors < 50, the Poisson error bars for absolute 95% confidence intervals, shown in Table 4, have been used instead [36].

$$Relative\ 95\%\ Confidence\ Interval\ for\ \#Error > 50 = \pm\frac{2\sqrt{\#Errors}}{\#Trials} \qquad (17)$$

As the results show, the #Errors obtained by the proposed model are acceptable as they lie within the 95% confidence interval of the experiments.

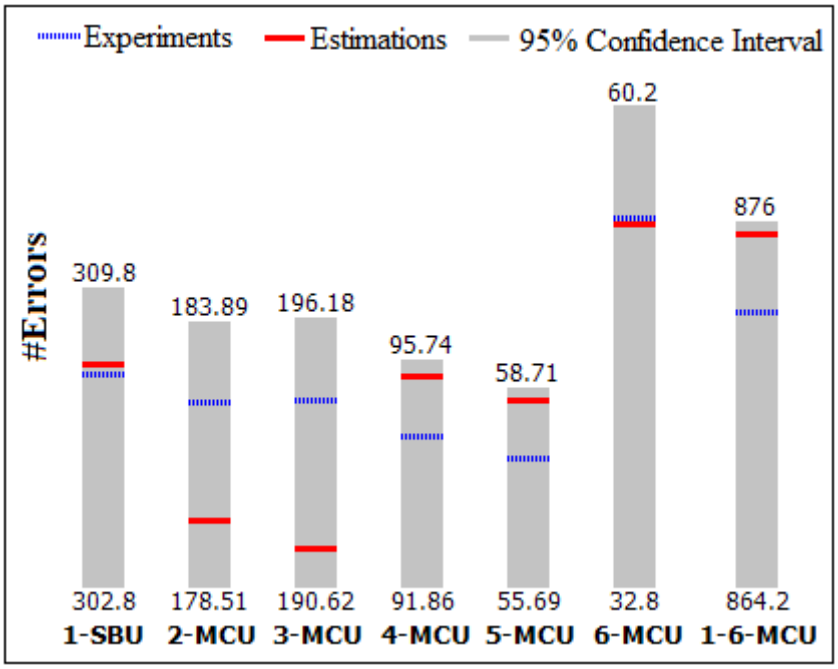

Figure 1 - #Errors of the estimations and experiments for the FFE filter operating under SERs when VCC = 0.5 V (logarithmic scale)

Table 4 – Absolute 95% confidence interval by the Poisson distribution for #Errors ≤ 50 [36]

0 → [0.0 – 3.7]	**17** → [9.90 – 27.2]	**34** → [23.5 – 47.5]
1 → [0.1 – 5.6]	**18** → [10.7 – 28.4]	**35** → [24.3 – 48.7]
2 → [0.2 – 7.2]	**19** → [11.5 – 29.6]	**36** → [25.1 – 49.8]
3 → [0.6 – 8.8]	**20** → [12.2 – 30.8]	**37** → [26.0 – 51.0]
4 → [1.0 – 10.2]	**21** → [13.0 – 32.0]	**38** → [26.8 – 52.2]
5 → [1.6 – 11.7]	**22** → [13.8 – 33.2]	**39** → [27.7 – 53.3]
6 → [2.2 – 13.1]	**23** → [14.6 – 34.4]	**40** → [28.6 – 54.5]
7 → [2.8 – 14.4]	**24** → [15.4 – 35.6]	**41** → [29.4 – 55.6]
8 → [3.4 – 15.8]	**25** → [16.2 – 36.8]	**42** → [30.3 – 56.8]
9 → [4.0 – 17.1]	**26** → [17.0 – 38.0]	**43** → [31.1 – 57.9]
10 → [4.7 – 18.4]	**27** → [17.8 – 39.2]	**44** → [32.0 – 59.0]
11 → [5.4 – 19.7]	**28** → [18.6 – 40.4]	**45** → [32.8 – 60.2]
12 → [6.2 – 21.0]	**29** → [19.4 – 41.6]	**46** → [33.6 – 61.3]
13 → [6.9 – 22.3]	**30** → [20.2 – 42.8]	**47** → [34.5 – 62.5]
14 → [7.7 – 23.5]	**31** → [21.0 – 44.0]	**48** → [35.3 – 63.6]
15 → [8.4 – 24.8]	**32** → [21.8 – 45.1]	**49** → [36.1 – 64.8]
16 → [9.4 – 26.0]	**33** → [22.7 – 46.3]	**50** → [37.0 – 65.9]

Figure 2 shows the MTTF obtained by experiments and the proposed model for different *k*-bit events in case of VCC = 0.5 V (Table 2) for the FFE filter (Table 3). In this figure, the discrepancy between experiments and estimations has been presented as well. The discrepancy of the results, obtained from Eq. (18), is shown by means of individual labels in the figure. Said discrepancy is, on average, 2.15%, which demonstrates the high accuracy of the proposed model.

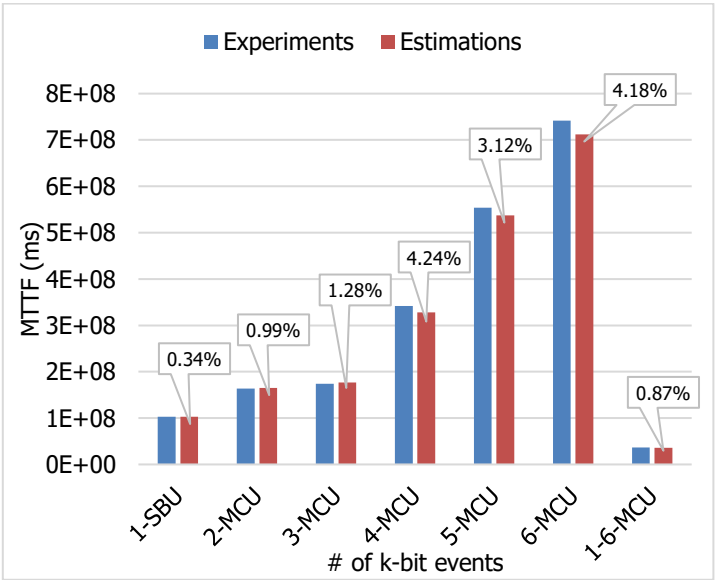

Figure 2 - MTTF of the estimations and experiments for the FFE filter operating under SERs when VCC = 0.5 V

$$Discrepancy\ (\%) = \frac{|Est - Exp|}{Est} \qquad (18)$$

In the next experiment (Figure 3), the FFE filter has been examined under environmental SERs when VCC = 3.3 V. Again, the estimations are acceptable too, as they all lie within the confidence interval of the experiments.

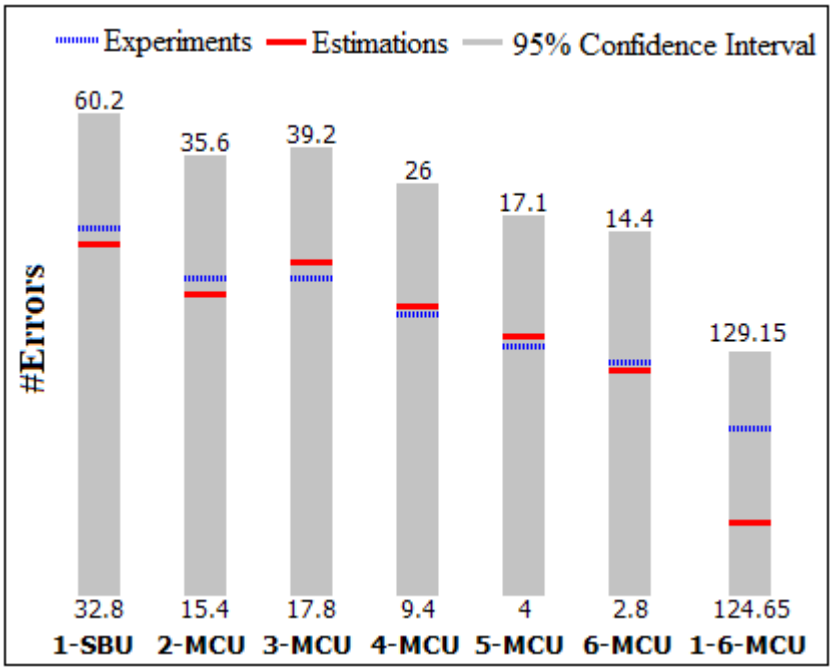

Figure 3 - #Errors of the estimations and experiments for the FFE filter operating under SERs when VCC = 3.3 V (logarithmic scale)

In this case, the MTTF of experiments and estimations has been calculated too (Figure 4). This time, the discrepancy of the results is higher than those of Figure 2. The reason is the large confidence intervals in some cases, which are a consequence of the scarce number of events that are expected to happen. This happens in events with large multiplicities (>3-bit MCUs) where the SER is low.

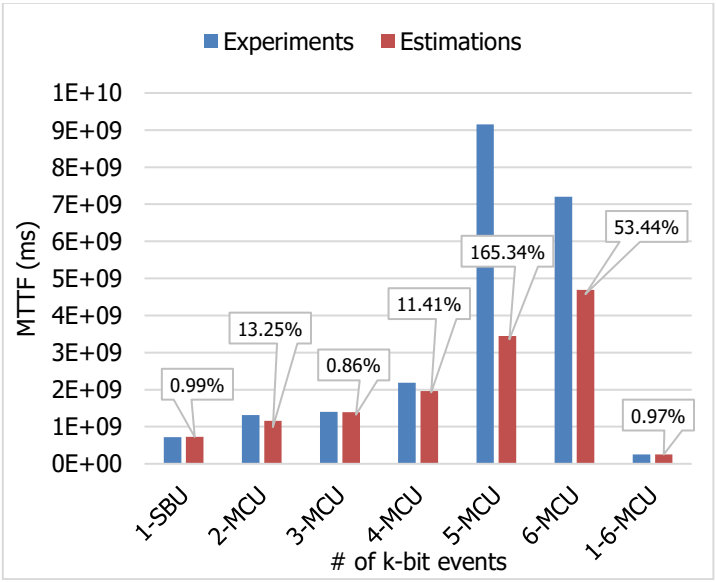

Figure 4 - MTTF of the estimations and experiments for the FFE filter operating under SERs when VCC = 3.3 V

Additional experiments have been done for hardware tasks b01 to b13 from the ITC'99 task set. Again, for practical reasons, only SERs when VCC = 0.5 V and VCC = 3.3 V have been used. The results of #Errors obtained from the experiments and estimations are shown in Figure 5 and Figure 6. In these experiments, all types of events (with multiplicities ranging from 1-bit to 6-bit) can occur. As the results show, in both cases, the values of estimations are acceptable as they lie within the 95% confidence interval of the experiments.

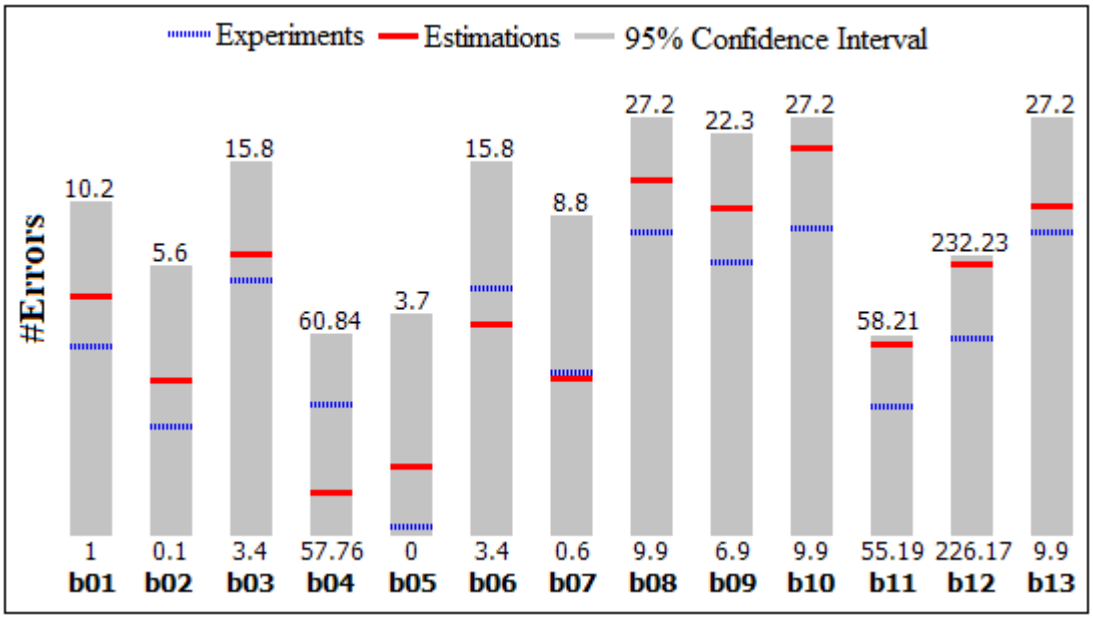

Figure 5 - #Errors of the estimations and experiments for tasks b01-b13 operating under SERs when VCC = 0.5 V (logarithmic scale)

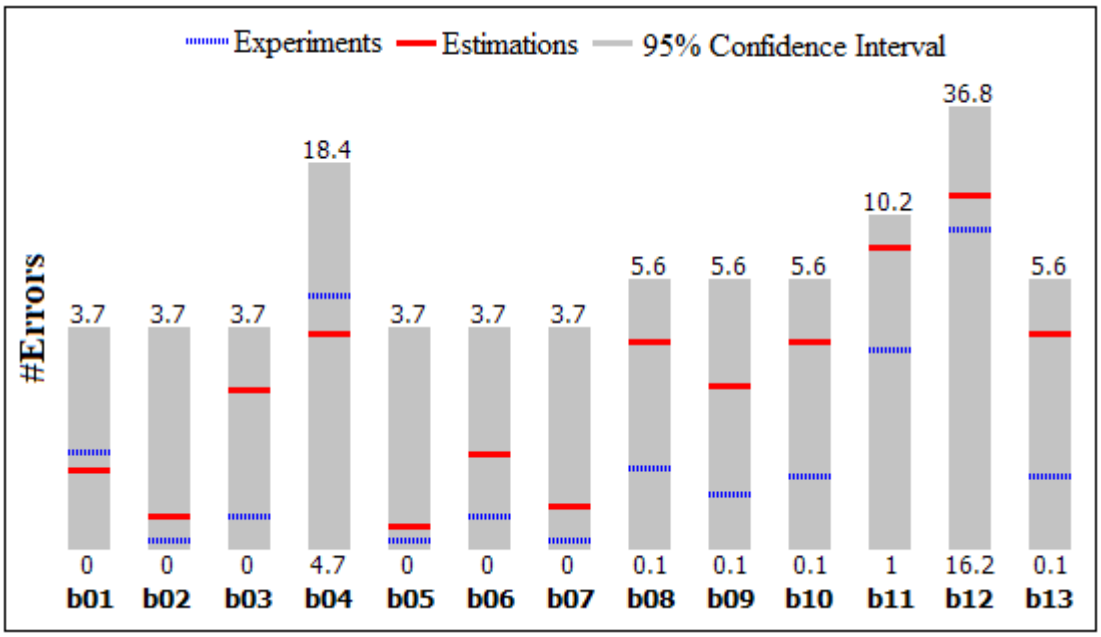

Figure 6 - #Errors of the estimations and experiments for tasks b01-b13 operating under SERs when VCC = 3.3 V (logarithmic scale)

C) Comparison with an only SBU-model

Finally, in order to analyze the impact of considering MCUs on the accuracy of the proposed reliability model, an additional experiment has been conducted. In this experiment, the proposed solution has been compared with the statistical reliability model presented in [17], where it is assumed that only SBUs can occur. For this purpose, the MTTF of the hardware tasks obtained by the proposed model (denoted by "MTTF MCU") and the conducted simulations (denoted by "Experiment MCU") have been compared with the MTTF of hardware tasks estimated by the only-SBU-aware reliability model presented in [17]. The experiment has been conducted for all tasks of Table 3 over the lowest and highest VCCs. These are depicted in Figure 7 and Figure 8, respectively.

In these figures, the labels over "MTTF SBU" represent how much the estimated MTTF is higher than "MTTF MCU", when MCUs are not considered to occur. On the other hand, the labels over "MTTF MCU" represent the discrepancy between MTTF of the proposed solution and the conducted simulations ("Experiment MCU"). The results of Figure 7 show that the MTTF of considering only SBUs is, on average, 2.78 times higher than the case of taking both SBUs and MCUs into account. The reason is that the reliability of the tasks increases when only SBUs are taken into account, which as a result leads to an increment in the MTTF (see Eq. (11)). It can also be observed that the values of the columns "MTTF MCU" are visibly closer to the experimental values of columns "Experiment MCU" than those of "MTTF SBU", which shows that the prediction made is more accurate.

Figure 8 shows the results of the same experiment for VCC = 3.3 V. The obtained results are similar to those of Figure 7 in the sense that the estimated MTTF of the only-SBU-aware reliability model is, on average, 2.89 times higher than that of the proposed MCU-aware model. Again, the "MTTF MCU" approach works better than the "MTTF SBU" one. The results of these experiments demonstrate the importance of considering MCUs in reliability estimation models. In this regard, ignoring the effects of MCUs leads to overestimating the reliability of the running tasks during mission.

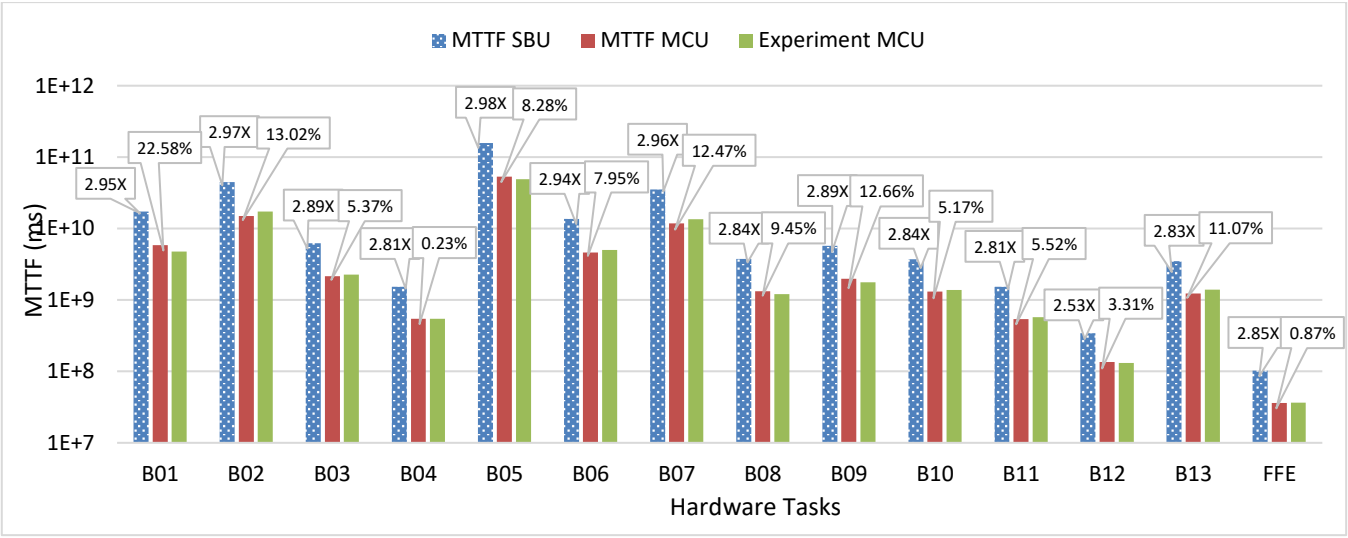

Figure 7 - Comparison of the proposed reliability model (SBU and MCU-aware) with the model presented in [17] (only SBU-aware) for VCC = 0.5 V (logarithmic scale)

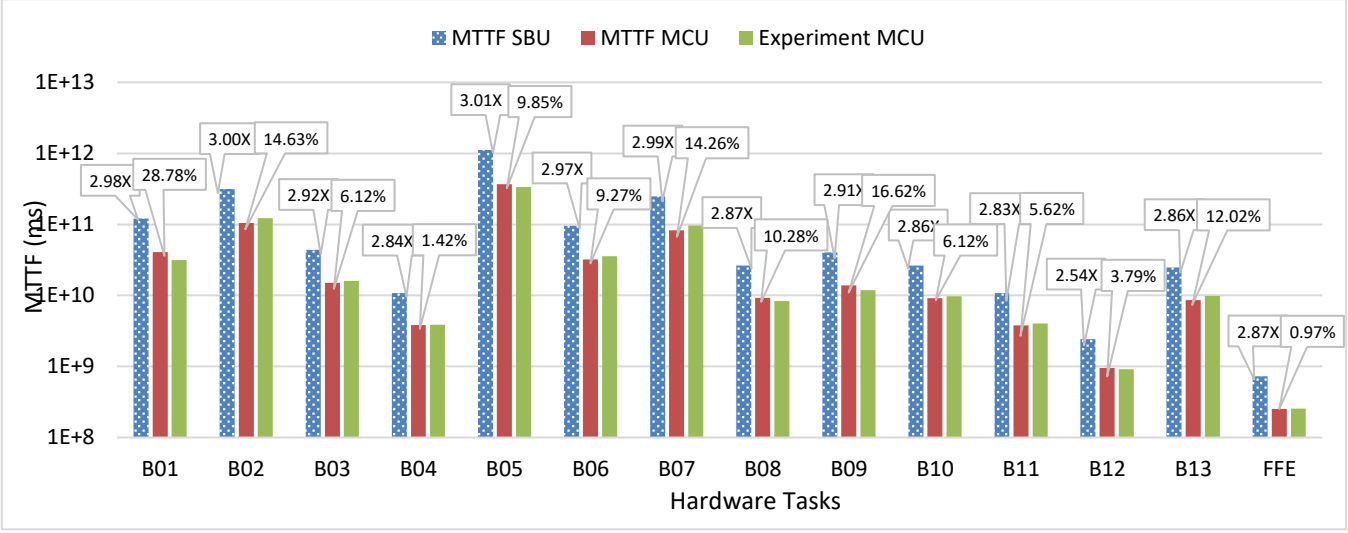

Figure 8 - Comparision of the proposed reliability model (SBU and MCU-aware) with the model presented in [17] (only SBU-aware) for VCC = 3.3 V (logarithmic scale)

VI. Conclusions and Future Work

In this paper, a statistical reliability model has been presented, which takes both SBU and MCU events into account to estimate the reliability of hardware tasks running on SRAM-based FPGAs in harsh environments. The proposed model uses some features of tasks and the environment, including task computation time, task size, the percent of task critical bits, and the SERs of k-bit events of different multiplicities to present simple equations for reliability estimation purposes in early design stages.

The proposed model has been validated by means of experiments on actual hardware tasks. These have shown that the model works well for estimating the number of errors that are expected to occur on tasks running on an FPGA operating in a harsh environment with environmental SERs for events with diverse multiplicities. This approach is a good alternative to radiation-ground tests or fault injection campaigns, in case these are expensive or unpractical to perform. Therefore, before operation, developers can easily estimate the reliability of their designs during the mission, provided that the features of the device and the expected environmental SER are known. Supplementary experiments demonstrate the importance of considering both SBUs and MCUs in the reliability estimation. Thus, it has been shown that ignoring MCUs leads to overestimating MTTF by 2.88X on average.

In this work, only the upsets within CLBs are considered. For future work, we are going to extend the proposed reliability model so that other resources such as DSPs, FFs, registers, and other memory elements are taken into account. In addition, the upsets are supposed to be occurred during task execution. Thus, we wish to introduce a new model to include soft errors that occur during task configuration time. Such a model is more complex since the configuration is carried out frame by frame and it depends on the architecture and the reconfiguration protocol that is used.

Acknowledgement

This work was supported in part by the Spanish MINECO project TIN2017-87237

VII. References

[1] A. Ghavidel, Y. Sedaghat, and M. Naghibzadeh, "Hybrid scheduling to enhance reliability of real-time tasks running on reconfigurable devices," *The Journal of Supercomputing*, pp. 1-30, 2019.

[2] A. D. George and C. M. Wilson, "Onboard processing with hybrid and reconfigurable computing on small satellites," *Proceedings of the IEEE*, vol. 106, no. 3, pp. 458-470, 2018.

[3] R. S. 167, *Software considerations in airborne systems and equipment certification*. RTCA, Incorporated, 1992.

[4] J. A. Clemente *et al.*, "SEU Characterization of Three Successive Generations of COTS SRAMs at Ultralow Bias Voltage to 14.2-MeV Neutrons," *IEEE Transactions on Nuclear Science*, vol. 65, no. 8, pp. 1858-1865, 2018.

[5] M. Caffrey, M. Echave, C. Fite, T. Nelson, A. Salazar, and S. Storms, "A space-based reconfigurable radio," in *Proceedings of the international conference on engineering of reconfigurable systems and algorithms (ERSA)*, 2002: Citeseer, pp. 49-53.

[6] M. Niknahad, O. Sander, and J. Becker, "Fine Grain Fault Tolerance—A Key to High Reliability for FPGAs in Space," in *Aerospace Conference*, 2012: IEEE, pp. 1-10.

[7] F. Siegle, T. Vladimirova, J. Ilstad, and O. Emam, "Mitigation of Radiation Effects in SRAM-Based FPGAs for Space Applications," *ACM Computing Surveys (CSUR)*, vol. 47, no. 2, p. 37, 2015.

[8] M. J. Cannon, A. M. Keller, H. C. Rowberry, C. A. Thurlow, A. Pérez-Celis, and M. J. Wirthlin, "Strategies for Removing Common Mode Failures From TMR Designs Deployed on SRAM FPGAs," *IEEE Transactions on Nuclear Science*, vol. 66, no. 1, pp. 207-215, 2019.

[9] R. Zhang, L. Xiao, J. Li, X. Cao, and C. Qi, "A Fault Injection Platform Supporting Both SEU and Multiple SEUs for SRAM-Based FPGA," *IEEE Transactions on Device and Materials Reliability*, 2018.

[10] I. Villalta, U. Bidarte, J. Gomez-Cornejo, J. Lázaro, and A. Astarloa, "Estimating the SEU failure rate of designs implemented in FPGAs in presence of MCUs," *Microelectronics Reliability*, vol. 78, pp. 85-92, 2017.

[11] J. A. Hogan, R. J. Weber, and B. J. LaMeres, "Reliability Analysis of Field-Programmable Gate-Array-Based Space Computer Architectures," *Journal of Aerospace Information Systems*, vol. 14, no. 4, pp. 247-258, 2017.

[12] K. A. Hoque, O. A. Mohamed, and Y. Savaria, "Dependability modeling and optimization of triple modular redundancy partitioning for SRAM-based FPGAs," *Reliability Engineering & System Safety*, vol. 182, pp. 107-119, 2019.

[13] R. Ramezani, Y. Sedaghat, M. Naghibzadeh, and J. A. Clemente, "A decomposition-based reliability and makespan optimization technique for hardware task graphs," *Reliability Engineering & System Safety*, vol. 180, pp. 13-24, 2018.

[14] O. Héron, T. Arnaout, and H.-J. Wunderlich, "On the Reliability Evaluation of SRAM-based FPGA Designs," in *International Conference on Field Programmable Logic and Applications (FPL)*, 2005: IEEE, pp. 403-408.

[15] H. Asadi, M. B. Tahoori, B. Mullins, D. Kaeli, and K. Granlund, "Soft Error Susceptibility Analysis of SRAM-based FPGAs in High-performance Information Systems," *IEEE Transactions on Nuclear Science*, vol. 54, no. 6, pp. 2714-2726, 2007.

[16] P. S. Ostler *et al.*, "SRAM FPGA Reliability Analysis for Harsh Radiation Environments," *IEEE Transactions on Nuclear Science*, vol. 56, no. 6, pp. 3519-3526, 2009.

[17] R. Ramezani, J. A. Clemente, Y. Sedaghat, and H. Mecha, "Estimation of Hardware Task Reliability on Partially Reconfigurable FPGAs," in *16th European Conference on Radiation and Its Effects on Components and Systems (RADECS)*, 2016: IEEE, pp. 1-4.

[18] N. T. Nguyen *et al.*, "Reconfiguration Control Networks for FPGA-based TMR systems with modular error recovery," *Microprocessors and Microsystems*, vol. 60, pp. 86-95, 2018.

[19] S. Jung and J. P. Choi, "Predicting system failure rates of SRAM-based FPGA on-board processors in space radiation environments," *Reliability Engineering & System Safety*, vol. 183, pp. 374-386, 2019.

[20] R. Ramezani, "A prefetch-aware scheduling for FPGA-based multi-task graph systems," *The Journal of Supercomputing*, pp. 1-21, 2020.

[21] A. M. Keller, "Using on-chip error detection to estimate FPGA design sensitivity to configuration upsets," M.Sc., Ira A. Fulton College of Engineering and Technology; Electrical and Computer Engineering, 2017.

[22] W. Mansour, R. Velazco, and G. Hubert, "Error-rate estimation combining SEE static cross-section predictions and fault-injections performed on HDL-based designs," *IEEE Transactions on Nuclear Science*, vol. 60, no. 6, pp. 4238-4242, 2013.

[23] C. Thibeault *et al.*, "A library-based early soft error sensitivity analysis technique for SRAM-based FPGA design," *Journal of Electronic Testing*, vol. 29, no. 4, pp. 457-471, 2013.

[24] R. Ramezani, "Dynamic scheduling of task graphs in multi-FPGA systems using critical path," *The Journal of Supercomputing*, 2020.

[25] C. Steiger, H. Walder, and M. Platzner, "Operating Systems for Reconfigurable Embedded Platforms: Online Scheduling of Real-time Tasks," *IEEE Transactions on Computers*, vol. 53, no. 11, pp. 1393-1407, 2004.

[26] R. Ramezani, Y. Sedaghat, and J. A. Clemente, "Reliability Improvement of Hardware Task Graphs via Configuration Early Fetch," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, no. 4, pp. 1408-1420, 2017.

[27] J. A. Clemente *et al.*, "Sensitivity characterization of a COTS 90-nm SRAM at ultralow bias voltage," *IEEE Transactions on Nuclear Science*, vol. 64, no. 8, pp. 2188-2195, 2017.

[28] I. Villalta, U. Bidarte, J. Gómez-Cornejo, J. Jiménez, and J. Lázaro, "SEU emulation in industrial SoCs combining microprocessor and FPGA," *Reliability Engineering & System Safety*, vol. 170, pp. 53-63, 2018.

[29] R. Ramezani, Y. Sedaghat, M. Naghibzadeh, and J. A. Clemente, "Reliability and Makespan Optimization of Hardware Task Graphs in Partially Reconfigurable Platforms," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 53, no. 2, pp. 983-994, 2017.

[30] H. Jahanirad, "Reliability Model for Multiple-Error Protected Static Memories," *Journal of Electronic Testing*, vol. 33, no. 2, pp. 189-207, 2017.

[31] J. A. Morales, F. Marc, A. Bensoussan, and A. Durier, "Simulation and modelling of long term reliability of digital circuits implemented in FPGA," *Microelectronics Reliability*, vol. 88, pp. 1130-1134, 2018.

[32] K. A. Hoque, O. A. Mohamed, Y. Savaria, and C. Thibeault, "Probabilistic model checking based DAL analysis to optimize a combined TMR-blind-scrubbing mitigation technique for FPGA-based aerospace applications," in *2014 Twelfth ACM/IEEE Conference on Formal Methods and Models for Codesign (MEMOCODE)*, 2014: IEEE, pp. 175-184.

[33] H. Quinn *et al.*, "Using Benchmarks for Radiation Testing of Microprocessors and FPGAs," *IEEE Transactions on Nuclear Science*, vol. 62, no. 6, pp. 2547-2554, 2015.

[34] V. Alaminos, F. Serrano, J. A. Clemente, and H. Mecha, "NESSY: An Implementation of a Low-cost Fault-injection Platform on a Virtex-5 FPGA," in *Conference on Radiation and its Effects on Components and Systems (RADECS)*, 2012, pp. 1-6.

[35] H. Quinn, "Challenges in Testing Complex Systems," *IEEE Transactions on Nuclear Science*, vol. 61, no. 2, pp. 766-786, 2014.

[36] G. Swift, "SEE Testing Lessons from Dickens, Scouting, and Oz," in *Single-Event Effects Symposium*, 2006, pp. 1-6.