Probability Aware Fault-Injection Approach for SER Estimation

Fábio B. Armelin*†‡, Lírida A. B. Naviner† and Roberto d'Amore‡
*Instituto Nacional de Pesquisas Espaciais (INPE), DEA, Av. dos Astronautas, 1758, São José dos Campos, Brazil
†Télécom ParisTech, COMELEC, 46 Rue Barrault, Paris, France
‡Instituto Tecnológico de Aeronáutica (ITA), IEEA, Pça. Mal. Eduardo Gomes, 50, São José dos Campos, Brazil

***Abstract*—The Soft-Error Rate (SER) estimation is used to predict how electronic systems will respond to the transient electrical pulses induced by the ionizing radiation. SER estimation by radiation test is an accurate method, but it is expensive and requires the real device. Traditional simulation methods incorporate logical, temporal and electrical masking effects while injecting faults at the output of the device's functional elements. Nevertheless, they do not consider the probability of the ionizing radiation to produce a transient fault at the output of each class of functional element. On the other hand, studies in the stochastic computing domain deal with a probabilistic fault-injection approach. Since many concomitant faults among the elements may occur, the fault probability of each element is treated independently. This leads to the use of one Pseudo-Random Number Generator (PRNG) and a probability comparator for each functional element. However, the analysis of a single fault is usually enough for SER estimation. In this context, this work presents a different approach for probability-aware fault-injection, in which a weighted distribution of faults is defined considering the relative fault probability of each functional element. This approach enables the use of just one PRNG and a decoder for the entire device, instead of a pair 'PRNG-comparator' per element, leading to a significant reduction in logic blocks consumption. For the example analyzed in this study, the use of relative fault probability decreases the number of logic blocks from 875 (adopting independent fault probability) to 495.**

I. Introduction

Soft-Error Rate (SER) is an essential parameter to properly assess the reliability of electronic systems that operate under the effects of the ionizing radiation particles. This parameter is intrinsically related to the rate of transient electrical pulses, R, generated in the sensitive internal elements of the device,

$$R = \int_0^\infty \phi(E) \cdot \sigma(E) \cdot dE, \tag{1}$$

where ϕ is the particle fluency, σ is the element cross-section, and E is the particle energy.

In digital systems, these sensitive elements are the switching transistors, for which the cross-section is directly related to their drain areas. However, the device's SER is not the sum, or combination, of the R value of each transistor. The logical, temporal and electrical masking effects filter-out many transient pulses. Thus, applying the eq. 1 to estimate the device's SER would require an 'effective σ', including these masking effects. The difficulty in determining this effective σ led to the development of many SER estimation methods.

The SER estimation methods can be analytical, as in [1], simulated fault-injection, as in [2], emulated fault-injection, as in [3], or radiation tests, as in [4]. The radiation tests, in which the real device is irradiated with neutron beams, is considered the most accurate SER estimation method. This method takes into account the physical and operational characteristics of the device and its interaction with a representative radiation particle. The drawbacks are the test cost and the need to have the real device (available late in the design cycle).

Other classes of SER estimation methods had evolved to try to achieve the accuracy obtained with the radiation tests. However, they still lack in considering a parameter that is intrinsic to the radiation test: the probability of the transient pulse, induced by the radiation particle, to produce a soft-error at the hierarchical level above (referenced as the functional element – ASIC's standard cells and FPGA's logic blocks).

For example, simulated fault-injection approaches inject a uniform distribution of faults at the outputs of the functional elements. In other words, they intrinsically consider that each class of functional element has the same probability to produce a soft-error when hit by a radiation particle, no matter if it is an Inverter or a 2-bit full-adder. The analytical and emulated fault-injection methods have the same intrinsic assumption.

The drawback in not consider the probability of the transient pulse to produce a soft-error at the hierarchical level above is more significant for FPGAs. In these devices, the same logic block can be configured for many distinct functions. In fact, a previous study analyzed how the logic block configuration may affect the SER estimation for an FPGA [5].

On the other hand, in the stochastic computing domain, some studies adopt a probabilistic FPGA emulated fault-injection approach [6]. In that domain, it is essential to consider many concurrent faults. For this reason, they consider individual fault probabilities. This leads to a substantial impact on the FPGA resources since it requires a Pseudo-Random Number Generator (PRNG) and a threshold probability comparator for each fault injection location.

This work proposes the SER estimation considering the fault probability of the functional elements. Contrary to the stochastic approach, the probability aware fault-injection approach proposed in this work considers only one fault at a time. For the SER estimation, the analysis of a single fault is usually enough, and this assumption enables the use of relative fault probabilities: the fault probability of each functional element

978-1-5386-1472-3/18/$31.00

is used as a weight for the fault distribution. A single PRNG with a decoder is used to distribute the faults to the functional elements.

This new probability aware approach enables considering the fault probability of each functional element, as in the radiation tests, with a significant reduction in logic block consumption, when compared to [6].

This paper is organized as follows. Section II describes the adopted methodology, including the analyzed example circuit. The results are shown in Section III, and Section IV discusses the proposed approach and compares the results with those from previous works. Finally, the conclusions of this study are given in Section V.

II. Methodology

The probability aware fault-injection approach proposed in this work relies on a weighted fault distribution, with weights defined by the sensitivity of each class of functional elements to the transient pulses induced by the radiation particles. The fault distribution is applied to the circuit under test (CUT) using an autonomous emulated fault-injection environment. This environment considers a specific fault model, that is injected to the outputs of the functional elements using saboteurs. Finally, the analysis of the proposed approach required some fault-injection campaigns. The following subsections describe all these aspects of the applied methodology.

A. Weighted Fault Distribution

The fault-distribution is implemented using a PRNG, to pseudo-randomly distribute the faults, and a decoder. For a uniform distribution, each value generated by the PRNG is used to enable one saboteur, in a one-to-one decoding process. On the other hand, for a weighted distribution, a range of values generated by the PRNG is used to enable one saboteur, in a range decoding process. Different weights are defined with different ranges.

Additionally, the ranges (weights) are proportional to the soft-error susceptibility of each class of functional element. For example, considering two classes of logic blocks, A and B, with the soft-error sensitivity of A being half the value of B, and a circuit composed by $[A; B; A; A; B]$ logic blocks; it could results in fault enabling ranges such as $[1..10; 11..30; 31..40; 41..50; 51..70]$.

Finally, a second PRNG determines the clock cycle of the test vector sequence in which the fault will be injected, to guarantee a uniform fault distribution over time.

B. Fault-Injection Environment

The fault-injection environment has two versions of the CUT: the Golden CUT, used as a reference, and the Faulty CUT, that is instrumentalized with saboteurs, for the fault-injection. Both versions of the CUT execute the same test vector sequence, driven by a Driver, while a Monitor compares the outputs of the CUTs. Any mismatch between the outputs is reported through a serial interface, containing a mismatch code and a time-tag (the clock cycle in which the mismatch occurred). The saboteurs of the Faulty CUT are controlled by a Fault Distribution Controller, that reports the injection of faults through another serial interface. This report contains the fault-injection location (Logic Block ID) and a time-tag (the clock cycle in which the fault was injected).

The fault-injection environment was implemented in the ProASIC3/E Starter Kit [7], with the target device A3PE1500-PQ208, from the flash-based ProASIC3E FPGA family [8].

TABLE I
Fault probability distribution of the CUT Logic Blocks

Macro ID	Logic Block ID	Fault Probab.
AND2	26	0.022
AND3	22, 23, 25, 28, 30, 32	0.020
AOI1B	7, 8, 9, 10, 11, 12	0.030
DFN1C1	13, 14, 15, 16, 17, 18, 19, 20	0.034
INV	35	0.020
NOR2A	6	0.020
NOR2B	1, 2, 5	0.022
NOR3B	3	0.018
NOR3C	4	0.020
XOR2	21, 24, 27, 29, 31, 33, 34	0.037

C. Fault Model

The transient pulse is generated using a programmable clock delay block (macro CLKDLY [9]) and a D flip-flop, with enable and reset (macro DFN1E1C1 [9]), as proposed in [10].

D. Saboteur

The Faulty CUT was instrumentalized with saboteurs at each logic block output. When enabled and in the presence of a fault, the saboteur inverts the logic value of the intercepted signal (eq. 2).

$$\texttt{output}_{\texttt{signal}} = \texttt{input}_{\texttt{signal}} \oplus (\texttt{sab\_enable} \cdot \texttt{fault}) \quad (2)$$

All saboteurs are connected to the same `fault` signal, while each `sab_enable` has a different driver. The delay caused by saboteur does not interfere in the estimation, since all saboteurs have the same delay and the transient pulses shall be distributed at any moment, without bias. On the other hand, the saboteur shall not distort the pulse [10].

E. Circuit Under Test

The circuit under test is an 8-bit counter analyzed in a previous study [5]. This circuit has an 8-bit output, and a clock and a reset inputs. Its synthesis in the target device resulted in 35 logic blocks (1 AND2, 6 AND3, 6 AOI1B, 8 DFN1C1, 1 INV, 1 NOR2A, 3 NOR2B, 1 NOR3B, 1 NOR3C, and 7 XOR2). The probabilities reported in [5] were adopted in this work and used for determining the weighted fault distribution (Table I).

F. Fault-Injection Campaigns

Once started by the user, the fault-injection environment runs autonomously, reporting each injected fault and each observed error. To evaluate the fault-injection environment and the CUT, two types of fault-injection campaigns were performed: one with a uniform fault distribution, as a reference, and another with a weighted distribution.

TABLE II
CUT OBSERVED ERRORS FOR UNIFORM AND WEIGHTED DISTRIBUTIONS.

Delay	Uniform Distribution	Weighted Distribution
10000 (3.9 ns)	38.8 %	39.8 %
01000 (2.3 ns)	31.1 %	32.5 %
00100 (1.5 ns)	25.1 %	27.8 %
00010 (1.1 ns)	22.5 %	24.4 %
00001 (0.9 ns)	20.7 %	22.9 %
00000 (0.7 ns)	12.6 %	15.5 %

For all fault-injection campaigns, the system clock was 40 MHz; the clock used to generate the transients was 40.1 MHz; the test vector sequence was 1,023 cycles (10-bit PRNG); the number of saboteurs to be controlled was 35; and the interval between the executions of the test vector sequences was 80,000 cycles (needed to send the serial messages), resulting in 500 injected faults per second.

For the uniform distribution, a 6-bit PRNG and a one-to-one decoder were used to control the saboteurs. On the other hand, for the weighted distribution, a 14-bit PRNG and a range decoder were used to control them.

III. RESULTS

For the fault-injection campaign with uniform distribution, from 495,243 injected faults, the mean value per Logic Block ID was 14,149.8, with a standard deviation of 0.4; and the mean value per clock cycle was 483.64, with a standard deviation of 27.87.

For the fault-injection campaign with weighted distribution, from 495,565 injected faults, the mean difference between the configured and actually injected faults per Logic Block ID was 41.38, with a standard deviation of 22.65. The mean value of injected faults per clock cycle was 484.93, with a standard deviation of 26.61.

The total logic blocks consumption, for uniform and weighted distributions, were 2366 and 2562, respectively. However, approximately 70% of these values are related to the reporting serial interfaces. For the weighted distribution, considering only the functional blocks used to distribute the faults, the consumption was 495 logic blocks (14 for the fault-time PRNG, 25 for the fault-place PRNG, 259 for the fault-distribution logic, and 197 for the range-decoder).

Finally, the CUT observed errors are presented in Table II, for uniform and weighted fault distributions, considering six different delay configurations for the block CLKDLY (with approximate resulting delay). The presented percentages were obtained with approximately 150,000 injected faults. A consistent greater value was observed when applying the weighted fault distribution.

IV. DISCUSSION

The proposed fault-injection environment leads to a good agreement between the configured and actually applied fault distribution.

Concerning the logic blocks consumption, the functional blocks required for the weighted distribution consumes 495 logic blocks. Adopting the independent fault probability described in [6] to the same CUT would result in at least 875 logic blocks (the CUT has 35 functional elements, each one with a 14-bits PRNGs, that consumes 25 logic blocks), without considering the probability comparators. Besides this significant area reduction, it is important to notice that the study presented in [6] uses a different FPGA technology, with different logic block resources.

The higher soft-error susceptibility of the CUT when adopting a weighted distribution of faults is consistent with the values observed in [5]. However, it is specific to this CUT. Others circuits can present lower susceptibilities when compared to those obtained with a uniform distribution of faults.

Finally, this probability aware fault-injection approach could be adapted to use other fault model, as the one used in [3].

V. CONCLUSION

Considering the soft-error susceptibility of each functional element of an electronic device (ASIC's standard cells and FPGA's logic blocks) can improve its SER estimation. This study presented a probability aware fault-injection approach capable of applying a weighted distribution of faults, based on the relative soft-error sensibility of the functional elements. The results indicate that this approach is more efficient regarding required resources than other probabilistic emulation-based fault-injection approaches found in the literature.

ACKNOWLEDGMENT

This work was partially funded by AEB and CNPq (process number 207364/2015-0/SWE).

REFERENCES

[1] S. Rezaei, S. G. Miremadi, H. Asadi, and M. Fazeli, "Soft error estimation and mitigation of digital circuits by characterizing input patterns of logic gates," *Microelectronics Reliability*, vol. 54, pp. 1412–1420, jun 2014.

[2] A. Lopes Filho and R. D'Amore, "Analysis of the error susceptibility of a field programmable gate array-based image compressor through random event injection simulation," *IET Computers & Digital Techniques*, vol. 6, no. 3, pp. 160–165, 2012.

[3] L. Entrena, M. Garcia-Valderas, R. Fernandez-Cardenal, A. Lindoso, M. Portela Garcia, and C. Lopez-Ongil, "Soft Error Sensitivity Evaluation of Microprocessors by Multilevel Emulation-Based Fault Injection," *IEEE Transactions on Computers*, vol. 61, pp. 313–322, mar 2012.

[4] C. Bottoni, M. Glorieux, J. Daveau, G. Gasiot, F. Abouzeid, S. Clerc, L. Naviner, and P. Roche, "Heavy ions test result on a 65nm Sparc-V8 radiation-hard microprocessor," in *2014 IEEE International Reliability Physics Symposium*, (Waikoloa), pp. 5F.5.1–5F.5.6, IEEE, jun 2014.

[5] F. B. Armelin, L. A. B. Naviner, R. D'Amore, and I. A. Azevedo, "Impact evaluation of logic blocks configuration on FPGA's soft error rate estimation," in *2016 IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, (Monaco), pp. 277–280, IEEE, dec 2016.

[6] D. May and W. Stechele, "A resource-efficient probabilistic fault simulator," in *2013 23rd International Conference on Field programmable Logic and Applications*, (Porto), pp. 1–4, IEEE, sep 2013.

[7] MICROSEMI, "ProASIC3/E Starter Kit User's Guide," tech. rep., Microsemi Corporation, Aliso Viejo, 2012.

[8] MICROSEMI, "ProASIC3E Flash Family FPGAs with Optional Soft ARM Support," tech. rep., Microsemi Corporation, Aliso Viejo, 2015.

[9] MICROSEMI, "IGLOO, ProASIC3, SmartFusion and Fusion Macro Library Guide for Software v10.1," tech. rep., Microsemi Corporation, Aliso Viejo, 2010.

[10] F. B. Armelin, L. A. B. Naviner, and R. D'Amore, "Using FPGA self-produced transients to emulate SETs for SER estimation," in *2018 IEEE Latin American Test Symposium*, 2018.