Partial Dynamic Reconfiguration in an FPGA-based Fault-Tolerant System: Simulation-based Evaluation

Richard Panek, Jakub Lojda, Jakub Podivinsky, Zdenek Kotasek
Faculty of Information Technology, Brno University of Technology,
Centre of Excellence IT4Innovations
Bozetechova 2, 612 66 Brno, Czech Republic
{ipanek, ilojda, ipodivinsky, kotasek}@fit.vutbr.cz

Abstract

Field Programmable Gate Arrays (FPGAs) are popular not only for their wide range of usage in embedded systems, however, they are susceptible to radiation effects. Charged particles cause the so-called Single Event Upsets (SEUs) in their configuration memory. SEUs can induce failure of the whole system. This problem is fundamental for space applications where sun radiation is more considerable than in the Earth. Two main approaches to SEU mitigation technique exist: fault masking and repair. The most popular masking method is Triple Modular Redundancy (TMR). For the faults repair, FPGA's capability of reconfiguration is used. It is possible to combine these approaches to obtain improved fault tolerant system. It is important to assess reliability rate of this system and, therefore, its estimation by a simulation is the main part of this paper. We propose evaluation environment which assesses the reliability of a TMR system with malfunction module reconfiguration depending on faults occurrence frequency and reconfiguration time necessary for fault repair.

1. Introduction

Field Programmable Gate Arrays (FPGAs) are very eligible for embedded systems implementation. The reasons are the possibility of production in small series compared to Application-Specific Integrated Circuits (ASICs) and the execution speed of implemented application, which is faster than processor-based implementation [6]. Moreover, FPGAs provide more advantages as a flexibility – the possibility to re-program their function, easy application prototyping, etc. The possibility to reprogram their configuration during application execution is a main characteristic, which is useful either for an application adaptation or for faults mitigation. The particular FPGA configuration is given by the bitstream, which is placed in the configuration memory. The bitstream utilizes FPGA's particular components (e.g. look-up tables, flip-flops, BRAMs, etc.), which are organized to configurable logic blocks and connected by a programmable interconnect network. The FPGAs are used in a wide range of industries, such as automotive or aerospace. The most commonly used FPGAs are based on an SRAM configuration memory (i.e. SRAM-based FPGAs). The utilization of SRAM memories enables faults to be induced by charged particles. These radiation effects produce bits flipping in the configuration memory that implies to a damage of the implemented circuit. This fault is known as a Single Event Upset (SEU) and it is necessary to evaluate its impact on space applications [9].

Lots of fault tolerance methodologies exist, the aim of which is the protection of implemented circuit in FPGA against impact of SEU. Considerable portion of these methodologies are based on spatial redundancy [9]. Triple Modular Redundancy (TMR) is one of the main approaches which is used as a base for the development of improved methods of fault tolerance. However, the utilized configuration memory cells that are responsible for correct circuit function are hit by only 1-20% of SEUs [6]. Therefore, it is realizable to reduce the amount of computation requisite for the verification of these methods. The TMR-based methods are presented in the following papers. For example, authors of [1] combine spatial with temporal redundancy, nevertheless as a base approach, they use the TMR. They reduced the area overhead by reducing the requirements on faults masking delay. Paper [14] classifies the used LUTs into SEU-sensitive and SEU-insensitive category. Thus TMR is applied only at SEU-sensitive LUTs class. This approach reduces the chip-area re-

978-1-5386-5710-2/18/$31.00

quirement in comparison with the classical TMR, and, also, circuit reliability is increased.

The TMR approach is only capable of masking faults. However, these faults remain in the configuration memory. When faults hit at least two of the units of the circuit, then TMR is unable to mask them and a system failure occurs. In this case, faults reparation abilities are important. Therefore, FPGA capability of reconfiguration is essential for correcting SEUs [4]. Moreover, it is necessary to have the whole reconfigurable fault-tolerant control system (FTCS) to utilize the FPGA reconfiguration [13]. FTCS is assembled from three main sub-systems: (1) reconfigurable controller which contains implemented circuit, FPGA is used in our case, (2) Faults Detection and Diagnostic (FDD) and (3) reconfiguration controller (RC) which is responsible for faults reparation on the base information about fault from FDD. It is possible to use TMR approach as FDD and Partial Dynamic Reconfiguration (PDR) append. This method is described in paper [3]. TMR majority voter is obtained by supplementary functionality which contrives to identify stuck TMR module. Then it is practical to repair this malfunctioning module by FPGA reconfiguration. Paper [5] describes FTCS based on processor implementation of RC and temporal redundancy to reach low power consumption. Paper [2] presents FTCS which is spread to multiple FPGAs. RC is located to either external processor or soft core processor in the FPGA. Another possibility for RC is the hardware implementation which is presented in paper [10]. In this case, RC can be located either in the same FPGA as the implemented circuit or in a different FPGA.

In this paper, we focus on a potential reliability of a reconfigurable FTCS depending on faults occurrence intensity. Authors of [2] presuppose only one fault occurrence in system concurrently and next fault appear after previous fault was repaired by reconfiguration. We investigate when this requirement will be fulfilled for TMR reconfigurable FTCS. Faults occurrence intensity depends on the environment dangerousness. The time which is necessary to repair circuit by reconfiguration depends on circuit taken area and require time for fault detection. For example, Soft Error Mitigation Controller [12], IP core which is developed by Xilinx, needs approximately 20 ms for detection and reparation of an SEU. Presently, we use simulation for rapid estimation of general FTCS reliability. In future work we want to affirm this estimation due to experiments with TMR FTCS implementation on FPGA. Thereafter, we will have a tool for rapid FTCS reliability estimation.

This paper is organized as follows. Section 2 introduces simulation tool that is able to evaluate the benefits of the reconfiguration. Experiments and results with the proposed tool are summarized in Section 3. Section 4 contains the conclusion of the paper and presents our plans for future research.

2. Evaluation Environment and Monitored Parameters

The main content of this work is the introduction of a tool for rapid evaluation of the use of reconfiguration in dependence on the environment properties in which the fault tolerant system is operated. We are particularly interested in the impact of Mean Time To Failure (MTTF) and the speed of reconfiguration on the resulting reliability of the entire system. MTTF is a parameter based on the environment in which the system is used or for which the system is designed. On the other hand, the speed of reconfiguration is based on chosen technology (cheap and slow FPGA or expensive and fast FPGA).

In order to make a quick evaluation, we created a simulation tool built on the *SimPy* library [11], which is a process-based discrete-event simulation framework based on standard Python. Processes are defined by Python generator functions and can model active components. *SimPy* library also proposes various types of shared resources to model limited capacity. Simulations can be performed in three modes: 1) as fast as possible, 2) in real time or 3) by stepping through the events.

We chose Triple Modular Redundancy as the experimental system, which we modeled for our experimental purposes. Figure 1 shows the overview of N-Module Redundancy (NMR) system, which is a generalized version of the TMR. Such system is composed of N-copies of functional units whose inputs are equivalent and outputs are processed by a majority voter. The whole system produces correct outputs if the absolute majority of functional units remains in failure-free state, which means that failure of two or more modules has a potential to cause a failure of the TMR system. For our experimentation purposes, we omit the failure of the voter component, as it usually causes failure of the whole system, and, thus, covering the actual data we want to collect in this simulation.

For the simulation purposes, the actual design of the implemented system is not important because only statistical data are obtained and evaluated. In this simulation, functional units store only their operational status (*failure or failure-free*), which can be changed only through Reconfiguration Controller and Fault Injector. Simulated Reconfiguration Controller is able to change

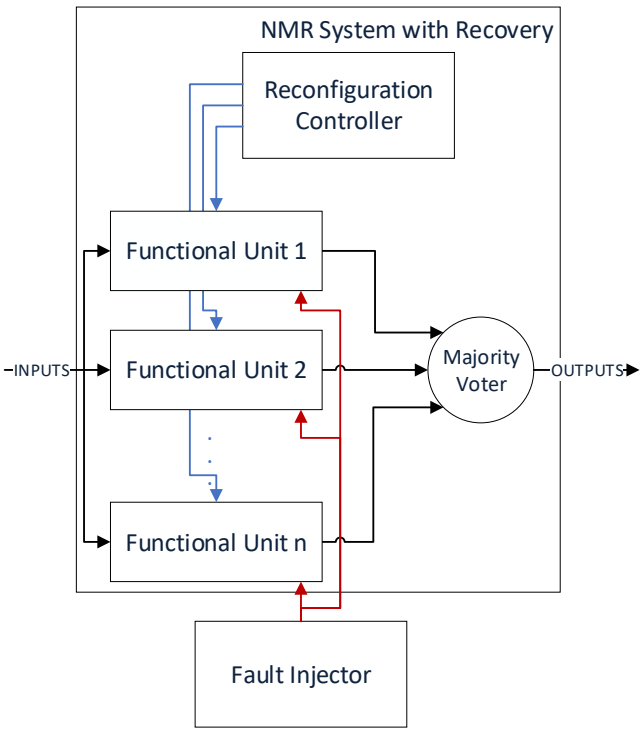

Figure 1. Overview of the model of NMR system with Fault Injector.

the status of functional unit from failure to failure-free with respect to time to reconfigure one unit. If there are more than functional units waiting in queue for its recovery by Reconfiguration controller, the reconfiguration is executed in a sequential manner.

On the other hand, Fault Injector, as is shown in Figure 1 is responsible for artificial fault injecting according to specified MTTF. However, to simplify implementation, fault injector is not a separate unit in our simulation. In the implementation, each functional unit moves from a failure-free state to the failure state autonomously with respect to the Mean Time To Failure of one unit. So each unit is implementing the timing of the fault injection. The timing is done with respect to the probability distribution, which is characterized by two parameters [7]: 1) *location* and 2) *scale*. The location is equivalent to the MTTF. For the scale, we empirically proposed Equation 1, which reflects the experiences we obtained through our previous experimentation with fault injection. In other words, that scale varies with the location to simulate higher dispersion of values for higher values.

$$scale = \frac{location}{10} + 1\,[-] \qquad (1)$$

The status of each functional unit is observed during the simulation run and the time the whole system is in failure state is measured. The output of one simulation run is the percentage number of time that the entire system was in failure state. The calculation is shown in Equation 2 and an example run of the simulation is shown in Figure 2.

$$failure\_state = \frac{time\_in\_failure}{total\_time} * 100\,[\%] \qquad (2)$$

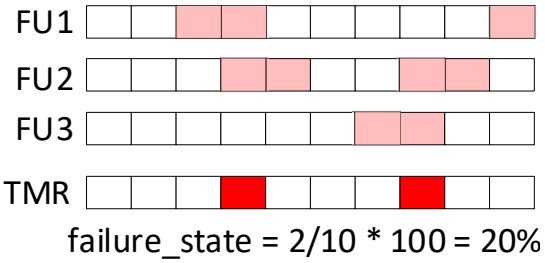

Figure 2. Example of failure state measurement.

3. Experiments and Results

Experiments were done by using previously described evaluation environment which is based on the process-based discreet-event simulation. A simple TMR system composed of three identical functional units and reconfiguration controller were used as the experimental circuit. Each of the functional units can be reconfigured by using the reconfiguration controller. The failure of the whole system occurs when more than two units fail. As was mentioned above, it is possible to define the time needed for the reconfiguration of one functional unit and also the MTTF of one unit. Table 1 shows, that the values of time to execute the reconfiguration was selected from the interval of $1-10$. The MTTF of one unit was selected as $10-100$. These values were selected on the basis of empirical monitoring of our real experimental electro-mechanical system (robot in maze) which were done as a part of our previous experiments [8]. These values are dimensionless numbers, specific units can be deployed depending on the particular system. We set the running time of one simulation to 1000 time units and the number of runs for one scenario was 10000. One scenario means one combination of MTTF and time to reconfigure one unit (one field in the table).

Experimental results are shown in Table 1 which shows the percentage number of time for which the

Table 1. Table of Failure State Percentage during the Operation of the System Obtained through the Simulation.

Failure State Representation [%]	Time To Reconfigure One Unit [-]									
MTTF of One Unit [-]	10.00	9.00	8.00	7.00	6.00	5.00	4.00	3.00	2.00	1.00
10.0	91.93	84.43	72.91	56.81	38.70	25.91	18.67	13.23	7.84	2.55
15.0	50.27	36.78	26.31	20.04	16.14	13.11	10.26	7.26	4.04	1.20
20.0	20.91	17.18	14.60	12.55	10.70	8.85	6.86	4.69	2.47	0.70
25.0	13.61	12.07	10.70	9.38	8.03	6.57	4.98	3.29	1.67	0.46
30.0	10.60	9.57	8.55	7.49	6.36	5.13	3.80	2.43	1.20	0.32
35.0	8.83	8.00	7.14	6.21	5.20	4.12	2.98	1.87	0.90	0.24
40.0	7.61	6.88	6.10	5.26	4.35	3.39	2.41	1.48	0.70	0.18
45.0	6.68	6.01	5.29	4.52	3.70	2.84	1.99	1.20	0.56	0.15
50.0	5.94	5.31	4.64	3.92	3.17	2.40	1.66	0.99	0.46	0.12
55.0	5.33	4.74	4.10	3.44	2.75	2.06	1.41	0.84	0.39	0.10
60.0	4.80	4.24	3.65	3.03	2.40	1.79	1.21	0.71	0.33	0.08
65.0	4.35	3.82	3.26	2.69	2.12	1.56	1.05	0.61	0.28	0.07
70.0	3.97	3.46	2.93	2.40	1.88	1.38	0.92	0.54	0.25	0.06
75.0	3.63	3.15	2.65	2.16	1.68	1.22	0.81	0.47	0.22	0.05
80.0	3.33	2.87	2.41	1.95	1.50	1.09	0.72	0.42	0.19	0.05
85.0	3.06	2.63	2.20	1.77	1.36	0.98	0.65	0.37	0.17	0.04
90.0	2.82	2.41	2.01	1.61	1.23	0.88	0.58	0.34	0.15	0.04
95.0	2.61	2.22	1.84	1.47	1.12	0.80	0.53	0.30	0.14	0.03
100.0	2.42	2.05	1.69	1.34	1.02	0.73	0.48	0.27	0.12	0.03

whole system fails for all combinations of MTTF and time to reconfigure one unit. The highest probability of failure is in case of the shortest mean time to failure and longest time to reconfigure one unit. On the other hand, the short time to reconfigure one unit and long MTTF lead to low probability of system whole failure. These results are predictable, but our simulation tool allows a system designer to model his system and monitor its behavior in various environments. System designer can choose critical probability of system failure and evaluate if his system meets the chosen boundary. The system designer can use these results as a basis for decision, whether the increased demands on reconfiguration speed will lead to a sufficient reduction of probability of system failure.

The achieved results are also presented in 3D chart shown in Figure 3. The chart shows the same values as Table 1, but from another point of view. One can see, that the increase of probability of failure is not linear. In some cases it can be possible to accept a little bit higher probability of failure which relieves demands on reconfiguration speed. On the other hand, the system can be optimized for environment with expected MTTF and our simulation tool can show the probability of the system failure in the case of an unexpected fall of MTTF (e.g. if the system is designed for an operation in a laboratory with a specified radiation and some consideration will lead to an increased radiation, which decreases MTTF of each unit). Thanks to the nonlinear increase of the probability of the system failure, a great reduction in speed requirements will lead to a small increase of probability that system will fail.

4. Conclusions and Future Research

The system useful for fault tolerant system designers was presented in this paper. Our tool offers the possibility to check expected benefit of the reconfiguration even before the reconfiguration is applied to the real FPGA-based system. A designer can find a compromise between the system reliability and costs of the appropriate reconfiguration implementation. They are able to evaluate if a great effort expended to accelerate the reconfiguration (e.g. a superior fast FPGA, another way of reconfiguration control, etc.) results in the expected advantage. An optimized system for the operation in specific environment with the expected reliability will be capable to guarantee enough reliability

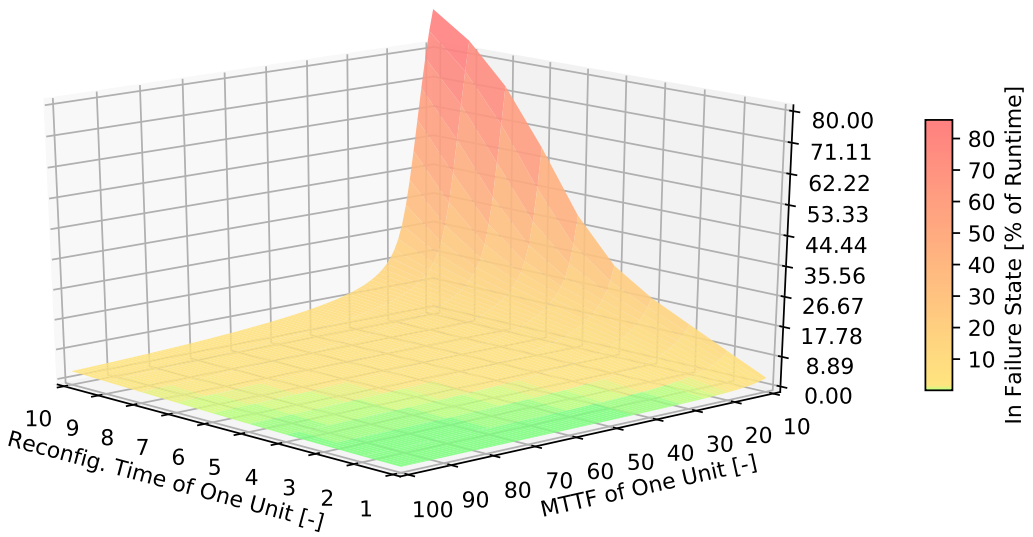

Figure 3. The Chart Showing Percentage of Time the System Sustained in Failure State for each Combination of MTTF and Reconfiguration Time of Each Unit.

even for short-term worsening of environment quality will be easily checked. Design system can be moved to the increased faults occurrence environment e.g. satellite flight through a source of radiation. A decrease of the system reliability depending on the reconfiguration speed and MTTF is not linear which is obvious from our results. Therefore, minor degradations of MTTF will not cause a significant degradation of the overall reliability.

As for future research, our goal is to use the reconfiguration as a tool for faulty module recovery in a real application and compare the result with results obtained by the presented estimation tool.

Acknowledgements

This work was supported by The Ministry of Education, Youth and Sports from the National Programme of Sustainability (NPU II), the project IT4Innovations excellence in science – LQ1602, the BUT project FIT-S-17-3994 and the JU ECSEL Project SECREDAS (Product Security for Cross Domain Reliable Dependable Automated Systems), Grant agreement No. 783119.

References

[1] J. Belohoubek, P. Fiser, and J. Schmidt. Error Masking Method based on the Short-duration Offline Test. *Microprocessors and Microsystems*, 52:236 – 250, 2017.

[2] C. Bolchini, L. Fossati, D. M. Codinachs, A. Miele, and C. Sandionigi. A Reliable Reconfiguration Controller for Fault-Tolerant Embedded Systems on Multi-FPGA Platforms. In *2010 IEEE 25th International Symposium on Defect and Fault Tolerance in VLSI Systems*, pages 191–199, Oct 2010.

[3] C. Bolchini, A. Miele, and M. D. Santambrogio. TMR and Partial Dynamic Reconfiguration to Mitigate SEU Faults in FPGAs. In *22nd IEEE International Symposium on Defect and Fault-Tolerance in VLSI Systems (DFT 2007)*, pages 87–95, Sept 2007.

[4] C. Carmichael, M. Caffrey, and A. Salazar. Correcting Single-Event Upsets Through Virtex Partial Configuration. Technical Report XAPP216, Xilinx, June 2000.

[5] C. Frenkel, J. D. Legat, and D. Bol. A Partial Reconfiguration-based Scheme to Mitigate Multiple-bit Upsets for FPGAs in Low-cost Space Applications. In *2015 10th International Symposium on Reconfigurable Communication-centric Systems-on-Chip (ReCoSoC)*, pages 1–7, June 2015.

[6] P. Gaillardon. *Reconfigurable Logic: Architecture, Tools, and Applications*. Devices, Circuits, and Systems. CRC Press, 2015.

[7] M. Natrella. *NIST/SEMATECH e-handbook of statistical methods*. NIST/SEMATECH, 2010.

[8] J. Podivinsky, J. Lojda, O. Cekan, R. Panek, and Z. Kotasek. Reliability Analysis and Improvement of FPGA-Based Robot controller. In *Digital System Design (DSD), 2017 Euromicro Conference on*, pages 337–344. IEEE, 2017.

[9] F. Siegle, T. Vladimirova, J. Ilstad, and O. Emam. Mitigation of Radiation Effects in SRAM-Based FP-

GAs for Space Applications. *ACM Comput. Surv.*, 47(2):37:1–37:34, Jan. 2015.

[10] M. Straka, J. Kaštil, and Z. Kotásek. Generic Partial Dynamic Reconfiguration Controller for Fault Tolerant Designs Based on FPGA. In *NORCHIP 2010*, pages 1–4. IEEE Computer Society, 2010.

[11] Team SimPy. SimPy: Discrete Event Simulation for Python. `<https://simpy.readthedocs.io/>`, 2017. Accessed: 2018-06-10.

[12] Xilinx. *Soft Error Mitigation Controller v4.1*, April 2018. PG036.

[13] Y. Zhang and J. Jiang. Bibliographical Review on Reconfigurable Fault-tolerant Control Systems. *Annual Reviews in Control*, 32(2):229 – 252, 2008.

[14] M. S. Zheng, Z. L. Wang, J. Tu, J. Y. Wang, and L. J. Li. Reliability Oriented Selective Triple Modular Redundancy for SRAM-Based FPGAs. In *Applied Mechanics and Materials*, volume 713, pages 1127–1131. Trans Tech Publ, 2015.