Overview of Fault Tolerance Techniques and the Proposed TMR Generator Tool for FPGA Designs

Abdul Rafay Khatri
Department of Computer Architecture and System Programming,
University of Kassel, Kassel, Germany.

***Abstract*—The FPGA has been involved in many safety and mission-critical applications in the last few decades. FPGA designs are also critical to errors and failures due to radiations. Fault-tolerant systems should be designed to overcome the effect of faults or failure during the operation of the systems. The primary objective of any fault tolerance technique is to produce a dependable system. Fault tolerance techniques add the capability to perform proper functioning in the presence of a fault. Fault-tolerant techniques can detect the faults and correct them, or mask the faults. The overview of the most standard techniques used for FPGA designs is described in the paper. Among them, it is found that the Triple Modular Redundancy (TMR) technique is the most straight forward in terms of implementation and easy to use. The proposed TMR code generator for implementing the FPGA design is also described. These FPGA designs are written in Verilog Hardware Description Language (HDL) at the different abstraction levels.**

Keywords*—FPGA designs; fault tolerance; TMR technique; Verilog HDL

I. Introduction

As modern digital systems have become increasingly large and complicated, their dependability parameters are of great concern in playing a critical role in supporting next-generation science, engineering and commercial applications [1]. In mission-critical and safety applications, reliability is a primary goal. In order to improve the reliability of systems, many different approaches are devised such as fault-masking, fault avoidance and fault-tolerant approaches [2]. Fault tolerance is intended to develop systems which deliver an accurate service, even in the presence of active faults. Fault tolerance is an essential ability of the system because it is not possible to develop a perfect system. There are various approaches used to achieve fault-tolerant design, e.g. redundancy. However, fault tolerance considered as an important means for target designs used in various applications, which are mentioned above [3]. Faults must be either masked or detected by the system to achieve these goals:

1) Fault masking: Fault masking is a technique that allows the system to perform correctly in the presence of an error, without doing an explicit detection of the error.
2) Fault detection: Fault detection is a process that allows the system to realise that a fault has occurred. Some examples of this technique are self-integrity checks.

FPGA has made a significant improvement in the system's design because of various features if offers such as reconfigurability. Due to this, it decreases the time to market and increases design flexibility. The FPGA has been involved in various applications in the last couple of decades, such as communication, medical imaging, safety-critical applications [4], [5], [6], [7]. These applications are implemented on Static Random Access Memory (SRAM)-based FPGA. SRAM FPGA devices are very critical to radiations and this may cause Single Event Effects (SEE) [5], [8], [9]. SEE is the combination of Single Event Transient (SET) and Single Event Upset (SEU). When an FPGA design is exposed to the radiations, it produces the transient pulse (SET) in combinational design and SEU in a memory element (which is also known as a bit-flip effect) [10], [6]. To develop the SEU mitigation scheme for FPGA-based designs, the designer must be careful about the following parameters:

- Reducing time to market
- Lowering development cost
- Increasing performance
- Reduction in power dissipation
- Lowering area overhead
- Improving reliability

Fault-tolerant circuits on SRAM-based FPGA can be implemented by two methods. The first method comprises developing a new FPGA matrix for fault-tolerant components. Another method is based on redundancy applying to the FPGA architecture [2].

The remainder of this paper is organised as follows: Section II describes the literature related to the methodology developed. Section III introduces the most widely used fault tolerance schemes for FPGAs. In Section IV, the proposed TMR code generator tool is described, which works on the code level of FPGA-based designs. Finally, Section V concludes the paper and presents some future directions.

II. Related Work

Now-a-days, soft errors are the biggest challenge in the design, development and evaluation of the reliability of digital circuits due to technology scaling. Owing to these errors, digital designs can be operated incorrectly and failed. Soft errors are categorised in two depending on the type of digital circuits. In combinational designs, soft errors are occurred and are called single event transient. However, in sequential design with memory, soft errors are called single event upset [11]. To measure the effect of soft errors in digital design is known as Soft Error Rate (SER). There are two methods

used for the evaluation of SER, i.e. dynamic and static. The dynamic SER method is used mostly with the fault injection techniques and logic simulation methodologies [12], [13]. To overwhelm the consequences of these errors in digital designs, many fault tolerance techniques have been presented in the last few decades. Fault tolerance techniques are categorised into three main classes: hardware redundancy-based, physical characteristics-based and synthesis-based techniques [11].

These error mitigation techniques are used to improve the reliability of FPGA systems. The most straight forward and widely used technique is Triple Modular Redundancy (TMR). The main problem with this technique is high area overhead which is about to 200% or higher. In [14], authors presented an approach and named it "Selective Triple Modular Redundancy (STMR)". In the approach, firstly the sensitivity of a gate towards the input signal probabilities is calculated to an SEU. After that, these gates are being triplicated by applying the conventional TMR techniques to all the gates of the target design. Authors in [15] presented a soft error mitigation technique which based on logic implication. The selective functionally redundant wires are added to the combinational logic of a target design. The method is described in this work.

In this paper, various SEU mitigation techniques are described for the FPGA-based systems to improve fault tolerance capabilities. All these techniques can be used at each stage of the development flow.

III. FPGA Upset Mitigation Techniques

The reliability of digital circuits, which are implemented on the FPGA, can be improved by several error mitigation techniques. These techniques are combined for the circuits to estimate reliability [2], [16]. The primary goal of these techniques is to enhance the dependability of digital designs. Some of the methods are designed explicitly for FPGAs, as shown in Fig. 1. Fault-tolerant techniques are divided into three categories, i.e. detection, mitigation and recovery methods.

These error mitigation techniques are also helped the researcher to improve FPGA reliability [16]. The most common techniques are briefly described in the sequel. These techniques are used individually or in a grouped to achieve high reliability with low cost and less time to market.

A. Radiation Hardening

Radiation hardening is the technique in which the semiconductor devices or electronic components are made robust against radiation to avoid damage and malfunctioning. Semiconductor devices operating in an environment with radiation are susceptible to many different failure mechanisms when radioactive particles strike their circuit elements. Various sources can cause many particles and they are divided into two classes [2], [10]:

1) The particles such as electrons, protons and heavy ions radiation are called charged particles.
2) Another class contains electromagnetic radiations such as x-ray, gamma-ray, or ultraviolet light.

Radiation Hardening (RH) is commonly achieved by using one of two methods [16].

1) RH By Design (RHBD): It is a redundant method in which many transistors are combined to form a one SRAM cell. This technique is architecture-dependent. In this technique, transistors are structured in such a way that the same charge particle cannot strike with different transistors of the same SRAM cell, hence making it redundant to cause an upset [17].
2) RH By Process (RHBP): In this technique, the transistors are shielded during the fabrication process in such a way that it is protected from ionising radiation at the silicon level. Gated resistor hardening is an example of RHBP. Gated resistor is a variable resistor which increases the threshold voltage to change the state of a memory cell.

Using the radiation hardening techniques, make the electronic devices more secure, but at the same time, the cost of the devices is increased too much. There are a variety of applications, such as military and space applications which requires high performance, high density and radiation-hardened FPGAs to decrease the design cost and the cycle time [18], [19].

B. Scrubbing

Scrubbing is the error mitigation technique used to correct the error occurred in the FPGA-based designs. The circuit which performs this task is called a scrubber. Scrubber implementation consists of simple or complex circuitry depending on the application, e.g. radiation environments. Scrubbing is the technique which works at the bit-stream file and overwrites the configuration file with its original contents when an error has been detected. Two different methods are devised i.e. detection methods and correction methods. In the scrubbing strategy, if the detection method or methods are involved it is called read-back strategy. However, blind scrubbing strategy is one which does not involve any detection method [20], [21], [22], [23].

There are two types of techniques used for scrubbing with the correction methods, i.e. error syndrome correction and golden copy correction. In the first type, scrubbing occurs when an error is detected. Error syndrome technique is mostly used in read-back scrubbing. Whereas, in the other technique, a fault-free copy of original configuration memory is kept in non-volatile memory or radiation-hardened memory. Blind scrubbing strategies consist of an only golden copy of the configuration, and it is a widely accepted technique for FPGA-based space platforms.

Blind scrubbing is performed at a specified rate, which is also known as the scrub rate. It can be defined as "the rate at which a scrub cycle should occur". There are few parameters such as scrub rate, design size, design reliability and design safety directly related to each other; hence, this rate can be measured by the upset rate in a system under test [21]. Some techniques such as bit-stream scrubbing or bit-stream read-back exist for the Xilinx FPGAs which detect errors/faults in the bit-stream or utilise this technique for the system under investigation [1]. The scrubbing technique can be used with hardware redundant techniques such as Error Correction Code (ECC) or Triple Modular Redundancy (TMR) techniques in order to improve the reliability of FPGA designs [24].

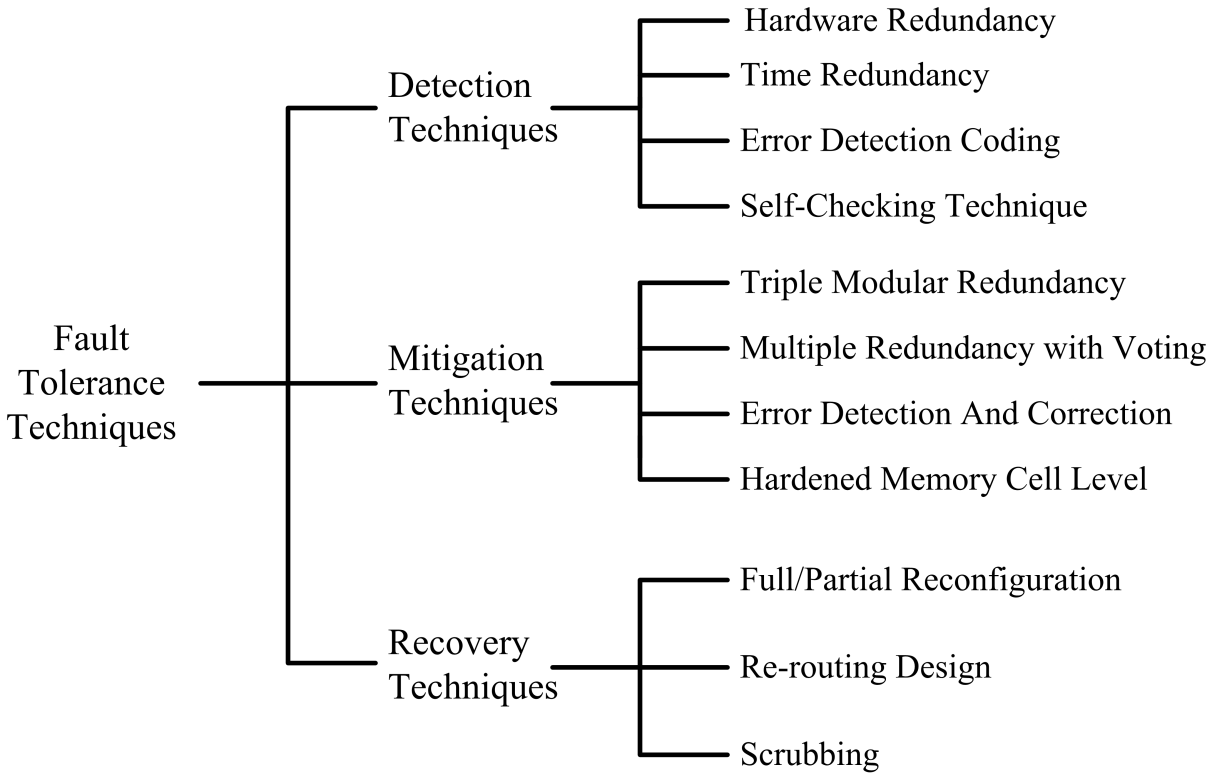


Fig. 1. Fault tolerance techniques used for FPGAs.

C. Error Detection and Correction

Soft errors are a significant concern for advanced digital designs, transmission channels and mostly for memories. When a soft error is occurred in a system, the contents of memory bits is altered which can be result in a system failure [25], [26], [3]. These techniques are divided into two classes:

1) Error Detection Techniques: In these techniques, errors are detected which occurs between the transmission and receiving channels. Error detection is realised by using a suitable hash function which appends a tag of fixed length to a message before transmission. When the receiver receives the message, it recomputes the tag and compares to the original tag [27]. These techniques include:
   a) Repetition Codes: This technique is a coding technique in which each transmitted bit is sent multiple times over a noisy channel to obtain an error-free communication. The efficiency of these codes is very low and are used very rarely.
   b) Parity Codes (Even and Odd Parity): It is again a straightforward coding technique in which parity bit is calculated and added with the message. It only detects single or odd number of errors appears in the received message. The parity codes are divided into even parity and odd parity. An even number of flipped bits in the transmitted message results in the calculation of same parity hence make the parity bit appear correct.
   c) Checksums Codes: In the checksum code method, a checksum of a transmitted message is calculated using a modular arithmetic sum and added to the message. On the receiving end, when the transmitted message is received then again checksum is calculated and compared.
   d) Cyclic Redundancy Checks (CRC): CRC is a cycle code technique used to detect the single burst error. It is a non-secure hash function which is used in digital computer networks to find accidental changes. It is calculated by the polynomial division and the remainder become the result [28].
   e) Cryptographic Hash Functions Codes: These codes provide any change appeared accidentally in the data (i.e. due to transmission errors over the channel). In this process, a hash value is computed and on the receiving end, this value is again calculated. Any change in the transmitted and received data is computed through comparison and the mismatched values of the hash detected the errors.
2) Error Correction Techniques:- In these techniques, the detected errors are corrected. These techniques include:
   a) Forward Error Correction Codes (FECC): Data reliability can be enhanced by using the forward error correction code techniques. It is used in digital signal processing. In

this technique, a redundant data is added to the message prior transmission. FEC method enables the receiver to correct the errors.

b) Automatic Repeat Request Codes (ARQ): This is an error-detection code to achieve reliable data transmission using acknowledgement messages (positive or negative), codes acknowledgement for error detection and time-outs.

Some techniques are used only to detect faults, and some are used to correct those detected faults. An ECC is a redundant technique that is more effective to correct the single-bit failure. A simple ECC circuitry consists of the XOR logic gate chain. All these gates are combined in some predefined way to compute a checksum [16]. If we look into the structure of configuration frame in Xilinx FPGAs, it contains an ECC word (a.k.a. checksum) to serve necessary single bit upset correction. There are some techniques used to locate the bit, which is changed because every bit in the configuration memory represents some point in the circuit.

D. Hardware Redundancy

Fault tolerant designs are achieved by various approaches as described above. The simplest of all techniques is to add redundancy. Redundancy can be described into four different forms such as hardware, time, information and software redundancy techniques. In hardware redundancy technique, an extra hardware is attached to the design. This additional hardware is used to either detect or mask the errors of a failed component [3], [29]. Hardware redundancy brings some penalties as well. Few are mentioned below,

1) Increase in overall design weight
2) Increase in size and power consumption
3) Also, increase cost, time to design, test and fabrication issues

These penalties can be overcome by various ways such as weight increase can be reduced by applying redundancy to higher-level components. Cost increase can be depreciated if the expected improvement in dependability diminishes the cost for the system [3]. Hardware redundancy is divided into three types: active, passive, and hybrid. Passive redundancy performs masking for the faults without requiring any action from the system.

1) TMR & Other Techniques for FPGA Designs: Triple modular redundancy is considered to be the most popular form of passive redundancy. In active redundancy technique, a fault is needed to be detected first before it is tolerated [3]. The most common forms of active redundancy are Duplication With Comparison (DWC), standby redundancy (which further divided into hot and cold standby), and Pair-And-A-Spare (which combines standby redundancy and DWC techniques). Hybrid redundancy can be achieved by combining the passive and active methods. These techniques are usually used in safety-critical applications such as medical equipment, aircraft, automotive, and so on. The most common hybrid redundancy techniques are Self-Purging redundancy and N-Modular redundancy with spares.

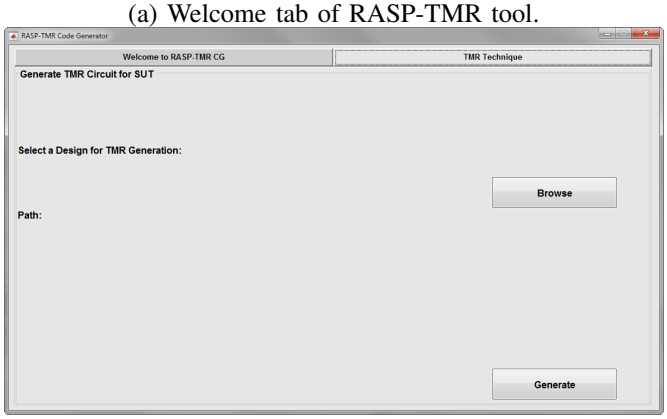
IV. Proposed TMR Code Generator

The RASP-TMR code generator tool is developed in Matlab. The graphical user interface is a tabbed-based tool as shown in Fig. 2. It contains 9 functions and 254 lines of code in Matlab. To provide easy usage, a stand-alone app is made using deploytool command of Matlab. This app can be installed on the system having Windows operating system by the user very easily [30]. This tool is developed for the following purposes.

1) Triplicates the design
2) Generates Top file and adds instantiation of all three modules
3) Adds the proposed majority voter circuits

More about the tool is described in [10].

(a) Welcome tab of RASP-TMR tool.



(b) Code generator tab.

Fig. 2. The proposed RASP-TMR code generator.

V. Conclusion

In this paper, fault tolerance techniques are described which are used for the FPGA-based designs. These techniques can be used on each stage of the FPGA development cycle. For example, hardware redundancy technique can be used at the code level, net-list or bitstream file. In order to implement hardware redundancy (specifically TMR), the RASP-TMR tool is developed for the FPGA-based design at the code level. In future work, more features are included to the proposed tool such as TMR with multiple voting etc. Different majority voter circuits will be added to the tool in future work.

REFERENCES

[1] M. Straka, J. Kastil, Z. Kotasek, and L. Miculka, "Fault tolerant system design and SEU injection based testing," *Microprocessors and Microsystems*, vol. 37, pp. 155–173, Mar 2013.

[2] F. Kastensmidt, L. Carro, and R. Reis, *Fault-Tolerance Techniques for SRAM-based FPGAs*, vol. 32. Boston, MA: Springer US, 2006.

[3] E. Dubrova, *Fault-Tolerant Design*. Springer, New York, NY, first ed., 2013.

[4] A. R. Khatri, A. Hayek, and J. Börcsök, *Applied Reconfigurable Computing*, vol. 9625 of *Lecture Notes in Computer Science*. Cham: Springer International Publishing, 2016.

[5] W. Xin, "Partitioning Triple Modular Redundancy for Single Event Upset Mitigation in FPGA," in *2010 International Conference on E-Product E-Service and E-Entertainment*, (Henan), pp. 1–4, IEEE, Nov 2010.

[6] A. R. Khatri, A. Hayek, and J. Börcsök, "Validation of the Proposed Hardness Analysis Technique for FPGA Designs to Improve Reliability and Fault-Tolerance," *International Journal of Advanced Computer Science and Applications*, vol. 9, no. 12, pp. 1–8, 2018.

[7] A. R. Khatri, A. Hayek, and J. Börcsök, "Fault Injection and Test Approach for Behavioural Verilog Designs using the Proposed RASP-FIT Tool," *International Journal of Advanced Computer Science and Applications*, vol. 10, no. 4, pp. 57–63, 2019.

[8] M. Desogus, L. Sterpone, and D. M. Codinachs, "Validation of a tool for estimating the effects of soft-errors on modern SRAM-based FPGAs," in *2014 IEEE 20th International On-Line Testing Symposium (IOLTS)*, (Platja d'Aro, Girona, Spain), pp. 111–115, IEEE, Jul 2014.

[9] L. A. C. Benites and F. L. Kastensmidt, "Automated design flow for applying Triple Modular Redundancy (TMR) in complex digital circuits," in *2018 IEEE 19th Latin-American Test Symposium (LATS)*, pp. 1–4, IEEE, Mar 2018.

[10] A. R. Khatri, A. Hayek, and J. Börcsök, "RASP-TMR: An Automatic and Fast Synthesizable Verilog Code Generator Tool for the Implementation and Evaluation of TMR Approach," *International Journal of Advanced Computer Science and Applications*, vol. 9, no. 8, pp. 590–597, 2018.

[11] A. H. El-Maleh and K. A. K. Daud, "Simulation-Based Method for Synthesizing Soft Error Tolerant Combinational Circuits," *IEEE Transactions on Reliability*, vol. 64, pp. 935–948, Sep 2015.

[12] M. Raji, H. Pedram, and B. Ghavami, "Soft error rate estimation of combinational circuits based on vulnerability analysis," *IET Computers & Digital Techniques*, vol. 9, pp. 311–320, Nov 2015.

[13] P. Shivakumar, M. Kistler, S. Keckler, D. Burger, and L. Alvisi, "Modeling the effect of technology trends on the soft error rate of combinational logic," in *Proceedings International Conference on Dependable Systems and Networks*, pp. 389–398, IEEE Comput. Soc, 2002.

[14] P. Samudrala, J. Ramos, and S. Katkoori, "Selective triple Modular redundancy (STMR) based single-event upset (SEU) tolerant synthesis for FPGAs," *IEEE Transactions on Nuclear Science*, vol. 51, pp. 2957–2969, Oct 2004.

[15] S. Almukhaizim and Y. Makris, "Soft Error Mitigation Through Selective Addition of Functionally Redundant Wires," *IEEE Transactions on Reliability*, vol. 57, pp. 23–31, Mar 2008.

[16] A. Gerald, *Configuration Scrubbing Architectures for High-Reliability FPGA Systems*. PhD thesis, Brigham Young University, 2015.

[17] M. R. Gardiner, *An Evaluation of Soft Processors as a Reliable Computing Platform*. PhD thesis, Brigham Young University, 2015.

[18] L. Rockett, D. Patel, S. Danziger, B. Cronquist, and J. Wang, "Radiation Hardened FPGA Technology for Space Applications," in *2007 IEEE Aerospace Conference*, pp. 1–7, IEEE, 2007.

[19] A. M. Keller, T. A. Whiting, K. B. Sawyer, and M. J. Wirthlin, "Dynamic SEU Sensitivity of Designs on Two 28-nm SRAM-Based FPGA Architectures," *IEEE Transactions on Nuclear Science*, vol. 65, pp. 280–287, Jan 2018.

[20] I. Herrera-Alzu and M. Lopez-Vallejo, "Design Techniques for Xilinx Virtex FPGA Configuration Memory Scrubbers," *IEEE Transactions on Nuclear Science*, vol. 60, pp. 376–385, Feb 2013.

[21] K. A. Hoque, *Early Dependability Analysis of FPGA-Based Space Applications Using Formal Verification*. PhD thesis, Concordia University Montreal, Quebec, Canada., 2016.

[22] G. L. Nazar, L. P. Santos, and L. Carro, "Scrubbing unit repositioning for fast error repair in FPGAs," in *2013 International Conference on Compilers, Architecture and Synthesis for Embedded Systems (CASES)*, (Montreal, QC), pp. 1–10, IEEE, Sep 2013.

[23] G. L. Nazar, L. P. Santos, and L. Carro, "Fine-Grained Fast Field-Programmable Gate Array Scrubbing," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 23, no. 5, pp. 893–904, 2015.

[24] R.-s. Zhang, L.-y. Xiao, X.-b. Cao, J. Li, J.-Q. Li, and L.-z. Li, "A Fast Scrubbing Method Based on Triple Modular Redundancy for SRAM-Based FPGAs," in *2018 14th IEEE International Conference on Solid-State and Integrated Circuit Technology (ICSICT)*, pp. 1–3, IEEE, Oct 2018.

[25] P. Reviriego, S. Pontarelli, and A. Ullah, "Error Detection and Correction in SRAM Emulated TCAMs," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 27, no. 2, pp. 1–5, 2018.

[26] F. G. D. Lima, *Designing single event upset mitigation techniques for large SRAM-based FPGA devices*. PhD thesis, UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL, 2001.

[27] J. Singh and J. Singh, "A Comparative Study of Error Detection and Correction Coding Techniques," in *2012 Second International Conference on Advanced Computing & Communication Technologies*, pp. 187–189, IEEE, Jan 2012.

[28] J. Proakis and M. Salehi, *Digital Communications*. McGraw-Hill, 2008.

[29] T. Ban, *Methods and architectures based on modular redundancy for fault-tolerant combinational circuits*. Theses, Télécom ParisTech, Sept. 2012.

[30] A. R. Khatri, "A Technical Guide for the RASP-FIT Tool," *International Journal of Advanced Computer Science and Applications*, vol. 10, no. 12, 2019.