International Journal of Advanced Information Science and Technology (IJAIST) ISSN: 2319:268
Vol.2, No.3, March 2013 DOI:10.15693/ijaist/2013.v2i3.52-58

Efficacious Redundancy Technique For Enriched Lockstep Architecture

N.Vaishnavadevi[1] S.Ramkumar[2] Dr. R.Ganesan [3]

Abstract - **Fault tolerance system plays a prominent role in many digital systems. A new intensifying lockstep scheme is proposed and implemented in EDA VLSI tool, which can mitigate Single-Event Upsets (SEU). Faults are detected and eliminated without interrupting the normal functioning of the circuit. Single point of failure, is eliminated and implemented as a fault tolerant using a Triple Modular Redundancy (TMR). This technique incorporates both transient and permanent faults. The new intensifying lockstep scheme requires significantly shorter recovery time than conventional lock step. It uses significantly less number of slices. Conventional lockstep scheme uses duplication with comparison (DWC), the presence of fault is detected, but it fails to indicate the location of fault which is overcome in enriched lockstep by triple modular redundancy (TMR). A grid lock technique is proposed for elimination of permanent faults. The efficiency of proposed method is validated using fault injection method and helps in verifying the performance of testable systems. It is based on the correlation between the effects of the SEU fault model with the Stuck-At (SA) faults.**

Keyword - **Fault tolerance, Single-Event Upset (SEU), Triple Modular Redundancy (TMR), Grid lock, Fault Injection.**

I INTRODUCTION

The basic idea of redundancy is to implemented multiple copies of the same circuit, and compare the outputs of each circuits. Disparity in these outputs indicates the occurrence of an error. Redundancy technique can be implemented at various levels such as circuits, systems etc. This process of switching is a simple process when both the designs meet the system restrictions identically [2]. Logic paths in between the flip-flops are composed of hard-wired, non-reconfigurable gates. Hence they are immune to SEUs. A fully fault tolerant system has the ability to detect and then corrects the hardware occurrence and return the system to its normal functionality. An optimal design will minimize the amount of extra logic required to detect and then correct the occurrence of the fault An extreme temperature change is one of the reasons in which fault tolerance is necessary for devices operating in harsh operating environments, as found, for example, in space and military applications[13][6]. Faults are separated into two categories: Permanent and Transient [8]. Permanent faults that exist in logic circuits are normally identified during offline testing by the manufacturer of the IC, so the transient fault is of major concern after a chip is in the hands of the consumer. The ability to simulate the occurrence of a transient fault in the VHDL description of a system is extremely important to verify the performance of an on-line testable system.

The ability to insert permanent faults on single bits or a data word must also be taken into consideration. Faults in an online testable system are assumed as a single bit fault, where a single bit is flipped from logic 1 to a 0 or vice-versa. They can be both transient and permanent in nature. Grid lock technique has a better simulation time and reduced number of LUTs, slices.

In this paper section II focus on single-event upsets (SEU) and types of upsets. Then SEU mitigation techniques were discussed. In section III proposed method; grid lock scheme was given along with fault injection. In Section IV, the experimental results of various architectures were discussed and the performance analysis has been done in section V. Finally, the paper was concluded in section VI.

II LOW OVER HEAD FAULT TOLORENCE TECHNIQUE

Single-event upset (SEU) is a state change for a memory buffer, whether it is in a processor, a memory component of an FPGA. SEUs are also called as soft errors, since it can be corrected by resetting or rewriting of the device [4]. It is mainly caused by an ion striking the transistor and causing it to change its

state Therefore a transistor might change its state from bit 1 to bit 0, hence making the data stored in that part of the memory invalid. Upsets in configuration memory can be detected by comparing its contents with a known, good state and can then be corrected by refreshing the state of memory [17]. Static upsets in configuration memory do not affect functionality.

Upsets need to be corrected only to ensure that errors do not accumulate. And transient fault changes the mapped circuit permanently, when it hits the memory. In addition to affecting memory, charged particles also change the logic function of the mapped circuit when they hit the on-chip configuration. In this process, partial re-configuration is used to correct the upsets once the errors are detected without interfering with the operation of the loaded design. Transient faults occur because of radiations, electromagnetic interference, and power glitches [20]. This has the advantage of improving the availability of the system without introducing any hardware overhead [13]. SEUs can affect both combinational and sequential circuits. It cause transient pulses in combinational logic paths.

A Configuration Upset

Configuration memory bits are those affected by SEU, it can be classified as sensitive (upset, which induces errors) and non-sensitive. SEUs affecting the configuration bits that are not utilized by a specific design will not affect the behavior of that design. SEUs may have permanent effects until the device is reconfigured e.g. by readback or scrubbing [20], [17]. In addition to configuration Sensitivity, the sensitive bits can be further categorized into two following categories [20].

Non-persistent bits [4], [6] are those configuration bits which, when upset, may induce non-persistent functional errors which disappear once the device is reconfigured, so that the circuit can return to normal operation. The non-persistent bits occurs in purely combinational circuitry

Persistent bits are those configuration bits which, when upset, induce persistent functional errors, which do not disappear even after the device is reconfigured. The persistent bits generally occur in the sequential circuitry. An internal reset of the registers and flip-flops is a feasible solution to avoid such types of functional errors

B *Single Event Upset Mitigation Techniques*

It is a process of applying design techniques to strengthen the functional integrity of the circuit, and protect it from the effect of any Single Event Upset. Fault-tolerant methods [6], [8] used to mitigate logic errors in FPGA based on redundancy technique are as follows. **Duplication with Comparison (DWC)** for detecting faults and **Triple Modular Redundancy (TMR)** with majority voter for masking faults

C Duplication with Comparison

Duplication with Comparison (DWC) is a detecting technique, in which the circuit to be protected is replicated twice and the results produced by the original circuit and the outputs of replicated circuits are compared to detect faults is given in figure 1. The implementation of DWC at processor level, supported by Xilinx ISE [21].

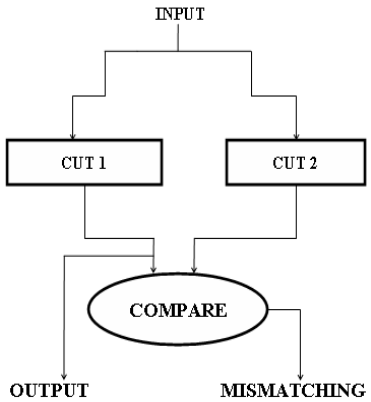

Figure 1 Duplication with Comparison (DWC)

Two identical circuits CUT1 and CUT2 receive the same inputs and simultaneously execute the same instructions, their results are compared step by- step at each clock cycle. Circuit CUT2 generates the reference results to be compared against those of CUT1 that provides the system output. Basically, DWC is able to detect but not to correct errors and also fails to indicate the fault location, since it cannot point out the faulty circuit. However, it could be capable to tolerate temporary faults, provided that it is supported by some re-execution procedure. In case of FPGA implementation, the system needs also to be reconfigured to recover correct functionalities.

D *Triple Modular Redundancy*

Triple Modular Redundancy (TMR) is the most reliable safeguard for total device failure as it rapidly detects and corrects SEUs[10], [11], [12].Three copies of the same circuit are connected to a "majority voter" which is used to obtain the fault free output is shown in figure 2. This method works as long as all the faults are confined to one of the redundant blocks. The latency

will be increased because of the voter in the circuit's critical path.

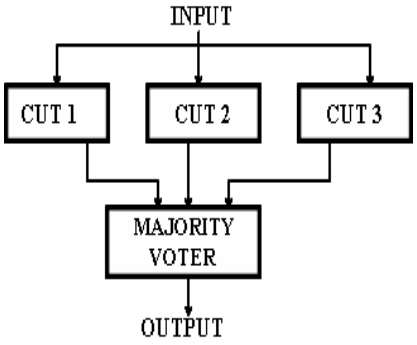

Fig 2 Triple Modular Redundancy (TMR)

The triple modular redundant ripple carry adder (TMR-RCA) is used as the reference circuit. This adder is the simplest approach for both detecting and correcting faults. The block diagram of the TMR adder circuit using the ripple carry adders is shown in Figure 3. The technique involved in information redundancy includes the use of error-correcting codes. Fault occur in any one of the adder is compared with the fault-free adders, in order to indicate the faulty one.

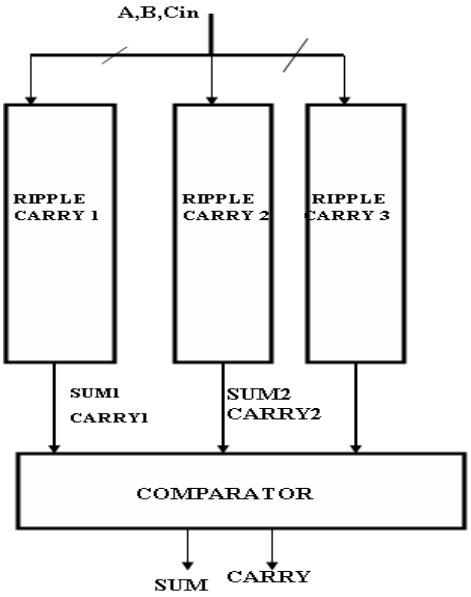

Figure 3 TMR adder circuit using ripple carry

TMR has a capability to protect both sequential and combinational circuits. A more efficient implementation of the TMR is focused in the sensitive logic, for example the memory cells to be protecting again SEU is shown in figure 4. It operates with the main aim of removing all single points of failure from the circuit. Each set of the triplicated circuit has its own set of inputs, to avoid errors occurring due to propagation of wrong inputs

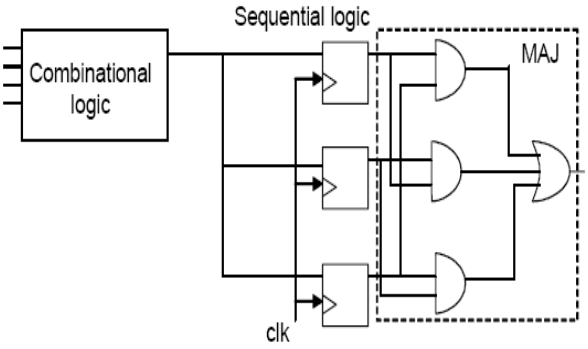

Figure 4 TMR memory cell with single voter

In the reconfigurable logic devices, user logic and logic paths are susceptible to SEUs.This makes the triple modular redundancy an effective technique. It has the advantages of complete data retention and autonomous recovery [20]. Xilinx builds its majority voters from the Output Buffer Three-state cell (OBUFT) provided by Xilinx library primitives[12]. The circuit is made immune to functional errors [13]. However, this method adds the system level overhead and increases the power dissipation. TMR can be implemented in various forms such as Simple TMR, TMR circuit with three voters, TMR circuit with three voters and clock, and feedback TMR[3], [9], [11] ,[13]

E *Enhanced Lockstep Architecture*

The basic lockstep scheme [7] uses the realization of DWC at the processor level. Unfortunately, it can only detect errors without indicating the faulty module. In order to alleviate this limitation, the new Enhanced Lockstep scheme is shown in Figure 7, which provided with the mean to identify the faulty circuit. It allows continuing the execution with the remaining fault-free circuit.

This technique involves with the operation of two identical circuits with synchronized clocking. A mismatching between the output values of the circuit indicates the occurrence of SEU. Recovery actions such as reinitializing and switching to safe mode are implemented. Figure 7 shows the architecture of the fault-tolerant system, whose two main blocks are Enhanced Lockstep scheme and the fault-tolerant (FT) Configuration Engine. Error Correcting Code (ECC) is used for detection and identification of single and double-bit errors in the given data. Each data are read from the configuration memory. Maximum clock

frequency is limited to 100MHz, but it can operate at frequency below 100 MHz. Based on this comparison, the ECC module produces indications for no error, single-bit error, and double-bit error conditions in addition to a syndrome indicating the location of single-bit errors. The synchronous reset input forces the SEU controller into an inactive state, releasing the configuration interface for use by other applications. In order to reset the data address to the first data of configuration memory and clears the error output. If the reset is released, the SEU controller will resume normal operation from the first frame of configuration memory on the next rising edge of clock.

The reset input may be tied to logic 0 for free-running SEU detection and correction in the circuit that do not require access to the configuration memory during normal operation.[12] Two identical circuits CUT1and CUT2 and are the most essential part of the Enhanced Lockstep scheme. Their outputs are identical during fault-free functioning, any mismatching indicating error(s). If there is an error indication, the output signals of the PLB bus and the peripheral outputs are compared by the Comparator/Multiplexer (COMP MUX). If there is a mismatching occurs between any of the two circuits, a signal will be generated. When an error is detected between the circuits, since all the resources of two circuits are blended together it, there is no possibility of differentiating the faulty circuit.

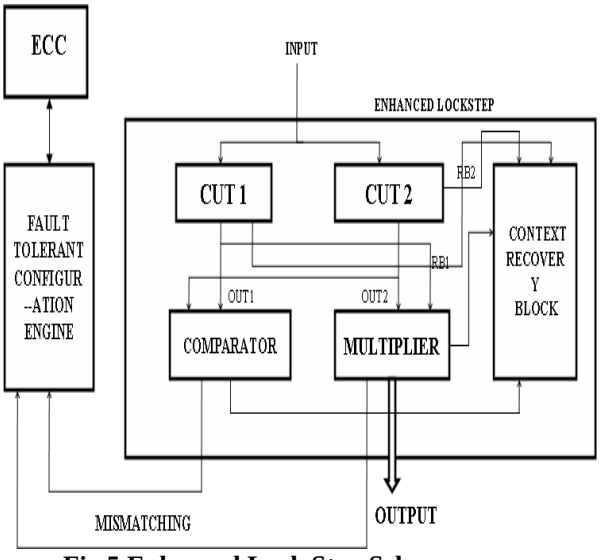

Fig 5 Enhanced Lock Step Scheme

In order to distinguish the faulty circuit, two blocks are used are the Comparator (COMP) and Multiplier (MUX). COMP that indicates any mismatch between the outputs Out1 and Out2 of CUT1 and CUT2 (containing the PLB and final output signals).[18] And the MUX, which connects one of the circuit to the system output, so that if one of them is reported to be faulty, the other is switched on. The switching is an atomic operation executed in one clock cycle. Once the error is localized by the FT Configuration Engine the affected processor is reconfigured to eliminate its configuration upset. Synchronization process is used for the newly reconfigured one to the same state as the correct one, thus enabling them to continue executing the same task in lockstep again. The recovery process of the Enhanced Lockstep scheme is handled by the Context Recovery Block (CRB).

III PROPOSED METHOD

GRID LOCK TECHNIQUE

TRIPLE MODULAR REDUDNANCY (TMR) is costly and time inefficiency and it also have longer delays. Transient faults are detected and corrected by using TMR, it fails for permanent faults. The disadvantage of TMR is overcome by using grid lock technique. STUCK-AT (SA-0 and SA-1) faults are considered to be permanent faults. If any line connected with ground produces 0 logic irrespective of other inputs called as stuck at 0 faults. If any line connected with supply produces one logic irrespective of other inputs called as stuck at 1 fault is given in figure 6

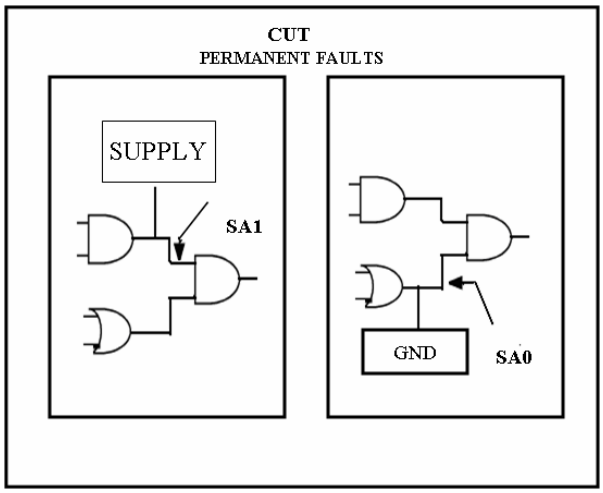

Figure 6 Isolation of faulty circuit

The principle of grid lock technique avoids the usage of the faulty zone of the circuit by pre-compiling the same design with various implementations. The correct location of the faulty circuit is detected by enhanced lockstep, hence in order to avoid stuck-at faults, it gets isolated from testing of circuits, those faulty circuits is considered as a

prohibited zone, hence a permanent fault can be masked by charging the appropriate configuration of the implementation in which the prohibited zone overlaps the faulty area. Thus grid lock technique reduces the simulation time of our testing.

B *VALIDATION OF PROPOSED METHOD*

Fault injection is a widely used technique for fault tolerance studies of electronic circuits. The main idea is to recreate the fault conditions of the circuit to evaluate, in a representation of the circuit itself made at a specific abstraction level [19]. This allows checking the sensitivity of a digital design. This technique proposes to harden the Single-Event-Upset (SEU) fault injection campaigns**.** SEU fault injection implies the insertion of a fault (i.e., a bit flip) in a flip-flop (FF) or memory cell at a specific time. [23] The total number of different SEU faults for a given circuit and workload can be expressed as N_{SEU}

$$N_{SEU} = F\ T$$

Where F is the number of FFs in the circuit and T is the application duration in clock cycles. The main idea is the correlation between the effects of SEU and Stuck-At (SA) fault models., in the very large majority of cases, in a digital circuit, a FF that does not produce output error neither when stuck at 0 nor when stuck at 1 under a specific set of stimuli, will not produce errors if affected by a SEU under the same stimuli.

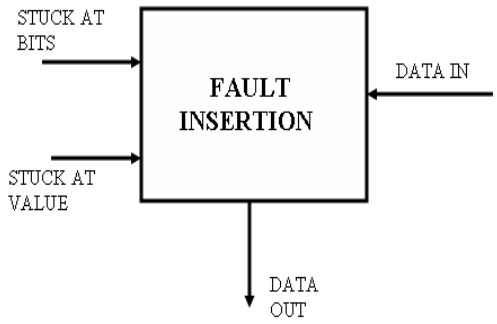

Figure 7 Data Flow for Permanent Fault Injection

The fault insertion system can introduce a permanent fault on a predetermined bit in a data word. Certain easy to use control settings are employed in order to insert a stuck-at- 0 or stuck-at-1 fault at a location selected by the user. Bits can be targeted easily with permanent injection by means of the **StuckAtBit** and **StuckAtValue** in the injection block. **StuckAtBit** is the location of the permanent fault and **StuckAtValue** is the logic value of the stuck-at fault. The flow for data that will have permanent injection of a fault is shown in Figure 7

The block monitors the control inputs to the circuit to evaluate whether it needs to perform transient or permanent fault injection in the data that is sent to it. The control code which initializes permanent fault injection is "1111". If "1111" is given as an input to circuit, the fault injection logic inserts a stuck-at fault Otherwise, injection is considered to be the transient nature.

The circuit operates differently when this "1111" is passed to it in the form of a control code. That code is the only one that uses **StuckAtBit** and **StuckAtValue**. The remaining control codes are for transient injection and range from "0001" → 50% injection to "1110" → < .01% injection. A control code of "0000" is 0% fault injection (fault-free), while a control code of "1111" is 100% fault injection (permanent fault). The control code is used to control the point, at which the fault is injected. By incrementing this control code by '1' for each code, fault injection is dropped by ½ from the previous rate.

IV RESULTS & IMPLEMENTATION

A fault tolerant system was implemented using triple modular redundancy (TMR). And proposed grid lock technique was validated using fault injection method. It was simulated by using Xilinx ISE 12.1i. The experimental results are given in table 1.

REQUIREMENT	ENCHANCED LOCKSTEP SCHEME	GRID LOCK TECHNIQUE
Number of slices	17	16
Number of LUTs	31	28
Number of bonded IOBs	50	46
Number of GCLKs	1	1
Time (ns)	**15.925ns** (9.942ns 5.983ns route) (62.4% logic,37.6%route)	**14.040ns** (3.683ns logic, 0.357ns route) (91.2% logic, 8.8% route)

Table 1 Experimental results

The proposed grid lock technique has a better performance and reduced the usage of LUTs, IOBs than enhanced lockstep scheme.

V PERFORMANCE ANALYSIS

The performance analysis of enhanced lockstep scheme with grid lock technique based on time, area, numbers of slices used are given in the figure 8

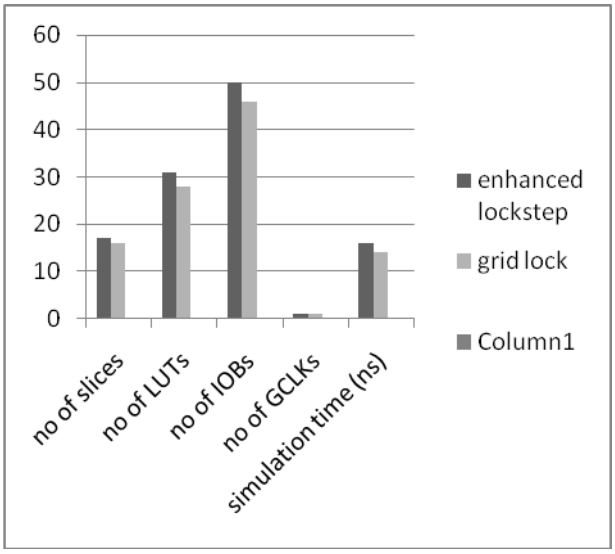

Figure 8 Area and Time comparison

Area and time was compared in between enhanced lockstep and gridlock technique, in which the proposed technique gives better reduction in LUTs and bonded IOBs, and simulation time.

VI CONCLUSION

Enhanced lockstep scheme uses triple modular redundancy (TMR) [22] as a fault tolerant to detect and eliminate transient faults, but it fails to detect the permanent faults. Hence grid lock technique is proposed, used to overcome the drawback of enhanced lockstep scheme. It reduces both area and time consumption considerably. The proposed technique is validated by using fault injection technique. Future work can be done in multiple error detection and correction. The performance and efficiency of the circuit can be improved by reducing the cost and simulation time.

REFERENCES

[1] Brian Pratt, Michael Caffrey, James F. Carroll, Paul Graham, Keith Morgan, and Michael Wirthlin, "Fine-Grain SEU Mitigation for FPGAs Using Partial TMR*", IEEE TRANSACTIONS ON NUCLEAR SCIENCE, VOL. 55, NO. 4, AUGUST 2008*

[2] M. Lanuzza., "An efficient and low-cost design methodology to improve SRAM-based FPGA robustness in space and avionics applications," *Proc. Int. Workshop on Reconfigurable Computing: Architectures, Tools and Applications, Lect. Notes on Comput. Sci.*, vol. 5453, pp. 74–84, 2009.

[3] H. Quinn, K. Morgan, P. Graham, J. Krone, M. Caffrey, and K. Lundgreen,"Domain crossing errors: Limitations on single device triple modular redundancy circuits in Xilinx FPGAs," *IEEE Trans. Nucl. Sci.*, vol. 54, no. 6, pp. 2037–43, Dec. 2007.

[4] L. Sterpone, M. Violante, R. H. Sorensen, D. Merodio, F. Sturesson,R. Weigand, and S. Mattsson, "Experimental validation of a tool for predicting the effects of soft errors in SRAM-based FPGAs," *Trans.Nucl. Sci.*, vol. 54, no. 6, pp. 2576–2583, Dec. 2007.

[5] P. Bernardi, L. Bolzani, M. Rebaudengo, M. S. Reorda, F. Vargas, and M. Violante, "Hybrid fault detection techniques in systems-on-a-chip," *IEEE Trans. Computers*, vol. 55, no. 2, pp. 185–198, Feb. 2006.

[6] M. Pignol, "DMT and DT2: Two fault-tolerant architectures developed by CNES for COTS-based spacecraft supercomputers," in *Proc. IEEE Int. On-Line Testing Symposium 2006*, 2006, pp. 10–12.

[7] PPC405 Lockstep system on ML310 Xilinx App. Note, XAPP564

[8] S. Rezgui, G. Swift, K. Somervill, J. George, C. Carmichael, and G. Allen, "Complex upset mitigation applied to a re-configurable embedded processor," *IEEE Trans. Nucl. Sci.*, vol. 52, no. 6, pp. 2468–2474, Dec. 2005.

[9] B. Pratt, M. Caffrey, P. Graham, E. Johnson, K. Morgan, and M.Wirthlin, "Improving FPGA design robustness with partial TMR,"presented at the IRPS Conf., Mar. 2006.

[10] P. K. Samudrala, J. Ramos, and S. Katkoori, "Selective triple modular redundancy (STMR) based single-event upset SEU tolerant synthesis for FPGAs," *IEEE Trans. Nucl. Sci.*, vol. 51, no. 6, pp. 2957–2969, Oct. 2004.

[11] N. Rollins, M. Wirthlin, M. Caffrey, and P. Graham, "Evaluating TMR techniques in the presence of single event upsets," in *Proc. 6th Annu. Int. Conf. Military and Aerospace Programmable Logic Devices (MAPLD)*, NASA Office of Logic Design, AIAA, Washington, D.C., Sep. 2003, p. P63.

[12] C. Carmichael, "Triple module redundancy design techniques for Virtex FPGAs," Xilinx Corp., Tech. Rep. XAPP197 (vl.O), Nov. 1, 2001.

[13] J. Arlat *et al.*, "Fault injection for dependability validation: A methodologyand some applications," *IEEE Trans. Softw. Eng.*, vol. 16, no. 2, pp. 166–182, Feb. 1990.

[14] P. K. Samudrala, J. Ramos, and S. Katkoori, "Selective triple modular redundancy for SEU mitigation in FPGAs," in *Proc. 6th Annu. Int. Conf. Military and Aerospace Programmable Logic Devices (MAPLD)*, NASA Office.

[15] D. Bhaduri, S. K. Shukla, P. S. Graham, and M. B. *equation reference goes here*Gokhale, "Reliability analysis of large circuits using scalable techniques and tools," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 54, no. 11, pp. 2447–60, Nov. 2007.

[16] D. Bhaduri, S. K. Shukla, P. Graham, and M. Gokhale, "Comparing reliability-redundancy trade-offs for two Von Neumann multiplexing architectures," *IEEE Trans. Nanotechnol.*, vol. 6, no. 3, pp. 265–279, May 2007.

[17] D. Bhaduri and S. Shukla, "NANOLAB—A tool for evaluating reliability of defect-tolerant nanoarchitectures," *IEEE Trans. Nanotechnol.*, vol. 4, pp. 381–394, 2005.

[18] M. Grosso and H. Guzman-Miranda, "Advanced speeding-up techniques for SEU sensitivity assessment," in *IEEE Int. Symp. Ind. Electron*, 2010, pp. 1995–2000.

[19] M. H. Kim, S. Lee, and K. C. Lee, "Kalman predictive redundancy system for fault tolerance of safety-critical systems," *IEEE Trans. Ind. Informat.*, vol. 6, no. 1, pp. 46–53, Feb. 2010.

[20] X. Liu, Q. Wang, S. Gopalakrishnan, W. He, L. Sha, H. Ding, and K. Lee, "ORTEGA: An efficient and flexible online fault tolerance architecture for real-time control systems," *IEEE Trans. Ind. Informat.*, vol. 55, no. 4, pp. 213–224, Oct. 2008.

[21] Haissam Ziade, Rafic Ayoubi and Raoul Velazco, "A Survey on Fault Injection Techniques" *The International Arab Journal of Information Technology, Vol. 1, No. 2, July 2004*

[22] D. González, "The SEUs simulation tool (SST), functional description,"Eur. Space Agency (ESA), Paris, France, Document Reference TECEDM/DGG-SST, 2004.Eur. Space Agency (ESA), Paris, France, Document Reference TECEDM/DGG-SST, 2004.

[23] *The International Technology Roadmap for Semiconductors (ITRS)*, 2007 ed. [Online]. Available:http://www.itrs.net/Links/2007ITRS/Home2007.

[24] Hung-Manh Pham, S´ebastien Pillement, Stanislaw J. Piestrak, "Low Overhead Fault-Tolerance Technique for Dynamically Reconfigurable Softcore Processor*"*, 27 *IEEE TRANSACTIONS ON COMPUTERS*, ISSN: 0018-9340, February 2012

[25] Michelangelo Grosso, Hipólito Guzman-Miranda, Miguel A. Aguirre, "Exploiting Fault Model Correlations to Accelerate SEU Sensitivity Assessment", IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, VOL. 9, NO. 1, FEBRUARY 2013

N.VAISHNAVADEVI received her B.E degree in Engineering from Sethu Institute of Technology, Kariyapatti, Virudhunagar Dist, Tamil Nadu in 2010, currently she perusing her M.E (VLSI Design) from Sethu Institute of Technology, Kariyapatti, Virudhunagar Dist, and Tamil Nadu. Her research interests include design for fault tolerance, design verification, area and power reduction.