A Hierarchical Priority Based Multi-Bit Error Tolerant Design for FPGA Configuration Memory

1st Pavana J, 2nd Lisa C
1 M.Tech Student,VLSI
2 Assistant Professor
Electronics and Communication Engineering
Nehru College of Engineering and Research Centre
Pampady,Thrissur,Kerala,India

Abstract—The reconfigurability of a field-programmable gate array (FPGA), which is utilized in many applications, permits changes to be made to an existing design. In hostile environments such as space, where radiation and ionizing particles increase the risk for failure and human maintenance is either impossible or time-consuming, this reconfigurable capability becomes even more important. Single event upsets (SEUs) and multi-bit upsets (MBUs) on configuration memory can cause soft errors and circuit malfunctions in SRAM-based field programmable gate arrays (FPGAs). The ever-increasing quantity of configuration bits in contemporary FPGAs makes it more difficult for traditional scrubbing to identify faults in a timely manner, leading to a mismatch between scrubbing performance and MBU sensitivity.A hierarchical multi-bit error correcting method fully utilizes the MBU sensitivity at various configuration frame locations. It employs distinct techniques for varying priorities and differentiates across configuration frames with multiple priorities. They divided code words into several sections and used distinct error-correction techniques in each section. In high radiation conditions, the suggested integrated error correction technique enhances FPGA configuration memory protection by substituting memory scrubbing. The suggested approach lowers total TTD and enables priority-based asynchronous MBU detection and correction.

Index Terms—FPGA,MBU,Sensitivity,Configuration memory

I. INTRODUCTION

Electronic components are subjected to radiation and charged particles in extreme settings such as space, which can result in Single-Event Effects (SEEs) and Single-Event Transients (SETs). Electronic equipment, especially FieldProgrammable Gate Arrays (FPGAs), which depend on sequential elements to retain configuration data, may become inoperable due to these SEEs. While memory scrubbing and triple modular redundancy are two traditional techniques used to reduce SEUs, they are slow and lead to non-negligible Time To Detect (TTD) faults.

Reconfigurable Field-Programmable Gate Arrays (FPGAs) are susceptible to SEUs because of their sequential element dependency. In-Memory Error Correction Code Checking (IMECCC) is implemented to decrease TTD following a SEU. Using a sensor network, this technique asynchronously finds SEUs and incorporates Error Correction Code (ECC) checkers on each CRAM word, causing repair when SEU is identified. In order to prevent errors during programming, IMECCC combines ECC parity bits at the system level and lessens the possibility of compounded errors from successive SEUs. When compared to a reference architecture that resembles a Xilinx Virtex 5 QV, the technique shows an average TTD reduction of at least 116,000×.

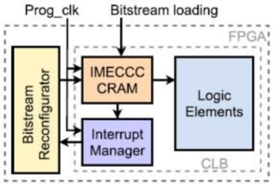


Fig. 1. IMECCC CLB block diagram

II. SYSTEM ARCHITECTURE

By introducing a novel architecture and design methodology that substitutes asynchronous Single Event Upset (SEU) detection for readback bitstream scrubbing, the proposed work seeks to address the shortcomings of conventional Rad-Hard FPGAs. This will reduce system liability and improve FPGA utilization for critical applications. Improvements in Error Correction Code (ECC) integration and detection time obtained by integrating ECC checkers as SEU sensors with a customized design to boost ECC error coverage are among the major achievements. The design process for incorporating this architecture into a fabric that compares to the Xilinx Virtex 5 QV is described in this section. As a proof of concept, the Google Skywater 130nm Process Design Kit (PDK) is used.It begins with a discussion of ECC implementation and its effects, and then moves on to a memory design variation meant to reduce the likelihood of undetected errors and simplify physical design. The use of CRAM cells for in-memory ECC checking is next explored, along with the floor planning procedure for these new CRAMs and how ECC checking can

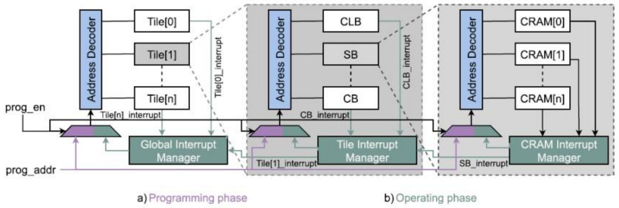


Fig. 2. Illustration of memory decoders controlled by system-level signals

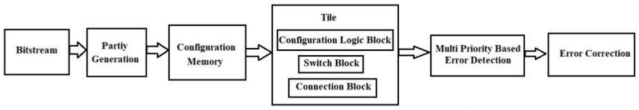


Fig. 3. Block Diagram

be used as a sensor for SEU detection. The section goes into greater detail on SEU sensing as an improved protection option that is more effective than bitstream scrubbing. Lastly, a selfprotection analysis of the suggested hardware is given; Figure 1's block diagram provides a high-level summary.

A. Word based CRAM Access

Configuration is usually stored in latches called CRAM in SRAM-based FPGAs. These latches can be loaded via a variety of protocols, such as serial or parallel techniques. We use a word-accessible approach in this work, which allows us to directly access each CRAM word using its own address. Similar to BRAM, this organization makes it easier to scrub and reconfigure by providing simple access to memory words using pre-existing decoders. Memory decoders are controlled by interrupt managers when in operation and at the system level during programming, as shown in Figure. Crucially, memories are unaffected in cases when a SET happens on a clock signal until the "write enable" signal is triggered, protecting the integrity of data that has been stored.

B. SEU Detection

Commercial FPGAs have historically relied on readback bitstream scrubbing for SEU detection, which inevitably causes delays in the process because errors might not line up with scrubbing addresses. Modern techniques combine scrubbing with Triple Modular Redundancy (TMR) to lessen this problem. By adding more ECC checkers—one for each memory word—we improve SEU detection in our proposed method. Similar to the asynchronous sensors seen in current FPGAs, these ECC checkers function as SEU detection sensors. For ECC implementation, both methods make use of standard equations.

III. Proposed Hierarchial Priority Based Multi-Bit Error Detection and Correction

The suggested block design is shown in Fig. 3, where the 32-bit data is separated into two groups: the LSB 16 bits, which require greater security since they are updated more frequently, and the MSB 16 bits, which have less transition data and require less protection. The current approach, which uses a Hamming code, can identify and fix up to two faults and one bit error. Shift logic combined with two-dimensional row and column parity generation allows for the detection and correction of multi-bit faults. The most important portion, as indicated in Table II, only uses eight bits—four horizontal and four vertical parity bits—for error detection and correction. On the other hand, as shown in Table 1, the least significant portion uses 12 bits—eight horizontal and four vertical parity bits—for error detection and correction.

15	14	13	12	H0
11	10	9	8	H1
7	6	5	4	H2
3	2	1	0	H3
V0	V1	V2	V3	

TABLE I
16-bit Configuration frame with horizontal and vertical parity

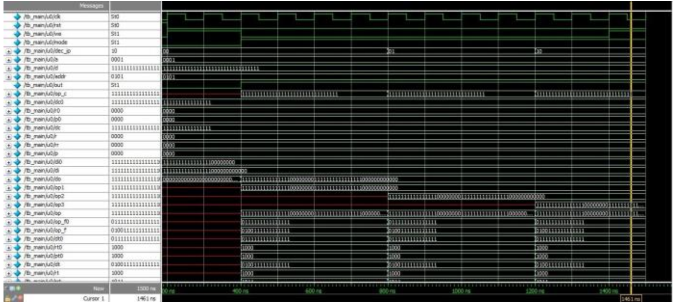
15	2	5	8	HH0
11	14	1	4	HH1
7	10	13	0	HH2
3	6	9	12	HH3
V0	V1	V2	V3	

TABLE II
16-bit Configuration frame with horizontal and vertical parity

Table II demonstrates the calculation of horizontal parity (HH0-HH3) and vertical parity (V0- V3) for a 4x4 array based on the presence of even and odd ones, as depicted. In the first column, the data is identical to that of Table 3.2. Subsequent columns in Table 3.3 represent data after performing 1-bit, 2-bit, and 3-bit column shifts, respectively. In cases where a single-bit error or double or triple bit errors occur in different rows and columns, the horizontal and vertical parity from Table I suffice to correct them. However, when two or three bit errors occur in the same row or column, a single set of horizontal and vertical parity is insufficient to locate the error. Therefore, either row or column shifts are performed for detection. The parity for both the shifted data and the non-shifted data is used to detect multi-bit errors. With the proposed method, all 3-bit errors and 60 percentage of 4-bit errors can be corrected effectively.

EXPERIMENTAL RESULTS

we first summarize the outcomes obtained from ModelSim simulations, highlighting any observed trends, insights, or noteworthy findings. We then transition to the implementation phase on Xilinx hardware, detailing the process and addressing any encountered challenges. Subsequently, we present the results derived from the hardware implementation and compare them with the simulation outcomes, emphasizing similarities and differences. Following this comparative analysis, we delve into a comprehensive discussion, evaluating the overall performance of the proposed method. We assess its effectiveness in meeting the stated objectives, discuss any limitations identified during the evaluation, and offer insights into potential avenues for further improvement or optimization. Through this integrated approach, we provide a thorough examination of the proposed method's performance, drawing meaningful conclusions grounded in both simulation and hardware-based assessments.



```
Messages
/tb_main/u0/clk      St0
/tb_main/u0/rst      St0
/tb_main/u0/we       St1
/tb_main/u0/mode     St1
/tb_main/u0/dec_ip   10
/tb_main/u0/a        0001
/tb_main/u0/d        1111111111111111
/tb_main/u0/addr     0101
/tb_main/u0/out      St1
/tb_main/u0/op_c     1111111111111111
/tb_main/u0/dc0      1111111111111111
/tb_main/u0/r0       0000
/tb_main/u0/p0       0000
/tb_main/u0/dc       1111111111111111
/tb_main/u0/r        0000
/tb_main/u0/rr       0000
/tb_main/u0/p        0000
/tb_main/u0/di0      1111111111111111
/tb_main/u0/di       1111111111111111
/tb_main/u0/do       1111111111111111
/tb_main/u0/op1      1111111111111111
/tb_main/u0/op2      1111111111111111
/tb_main/u0/op3      1111111111111111
/tb_main/u0/op       1111111111111111
/tb_main/u0/op_f0    0111111111111111
/tb_main/u0/op_f     0100111111111111
/tb_main/u0/dt0      0111111111111111
/tb_main/u0/rt0      1000
/tb_main/u0/pt0      1000
/tb_main/u0/dt       0100111111111111
/tb_main/u0/r1       1000
Now                  1500 ns
Cursor 1             1461 ns
```

Fig. 4. Simulation Result of the Proposed System

ANALYSIS	EXISTING SYSTEM	PROPOSED SYSTEM
TOTAL NUMBER OF 4 INPUT LUT	175	166
NUMBER OF BOUNDED IOB'S	139	77
TOTAL EQUIVALEN T GATE COUNTS	6185	5925
POWER	149mW	160mW
DELAY	2.872ns	2.872ns

TABLE III RESULT ANALYSIS

The Table III compares key metrics between the existing system and the proposed system. In the existing system, there are 175 4-input LUTs, while in the proposed system, this number reduces to 166, indicating optimization in resource utilization. Similarly, the number of bounded IOBs decreases significantly from 139 in the existing system to 77 in the proposed system, reflecting a more efficient allocation of input/output resources. Despite these reductions, the total equivalent gate counts also decrease from 6185 in the existing system to 5925 in the proposed system, indicating overall improvement in gate-level efficiency. However, there is a slight increase in power consumption from 149mW in the existing system to 160mW in the proposed system, which could be attributed to additional logic or activity changes. Notably, the delay remains the same at 2.872ns in both systems, indicating that the proposed optimizations do not compromise performance. Overall, the proposed system demonstrates improvements in resource utilization and gate-level efficiency while maintaining comparable performance metrics.

CONCLUSION

In summary, with an emphasis on improving performance, resource efficiency, and reliability, the project has made great progress toward optimizing FPGA architecture and design methodology. By introducing new features such as partial reconfiguration based on interruption and by using sophisticated error correction coding, the research has successfully solved the drawbacks of the current setup. Through extensive study and simulation using technologies like Xilinx and ModelSim, the suggested system has shown appreciable advancements. Enhancements in total equivalent gate counts combined with fewer 4-input LUTs and bounded IOBs highlight the system's increased resource efficiency. A dependable and predictable performance was ensured by the delay's consistency even with a slight increase in power consumption. Furthermore, the project's benefits go beyond optimization metrics; it provides strong answers to problems brought on by single-event upsets and other errors. The combination of sophisticated error correction coding techniques with interruption-based partial reconfiguration improves the system's flexibility and resilience in crucial applications dramatically. This project opens the path for the creation of more advanced FPGA systems that can satisfy the changing requirements of mission-critical jobs while guaranteeing high reliability, efficiency, and performance by setting the foundation for future developments.

REFERENCES

[1] S. Whitaker, J. Canaris, and K. Liu, "SEU hardened memory cells for a CCSDS Reed–Solomon encoder," IEEE Trans. Nucl. Sci., vol. 38, no. 6, pp. 1471–1477, Dec. 1991. [Online]. Available: https://ieeexplore.ieee.org/document/124134/.

[2] J. Kim and P. Mazumder, "A robust 12T SRAM cell with improved write margin for ultra-low power applications in 40 nm CMOS," Integr., VLSI J., vol. 57, pp. 1–10, Mar. 2017. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0167926016300700.

[3] M. Ebrahimi, P. M. B. Rao, R. Seyyedi, and M. B. Tahoori, "Lowcost multiple bit upset correction in SRAM-based FPGA configuration frames," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 24, no. 3, pp. 932–943, Mar. 2016.

[4] B. Narasimham et al., "Characterization of digital single event transient pulse-widths in 130-nm and 90-nm CMOS technologies," IEEE Trans. Nucl. Sci., vol. 54, no. 6, pp. 2506–2511, Dec. 2007.

[5] W. Calienes, R. Reis, C. Anghel, and A. Vladimirescu, "Bulk and FDSOI SRAM resiliency to radiation effects," in Proc. IEEE 57th Int. Midwest Symp. Circuits Syst. (MWSCAS), Aug. 2014, pp. 655–658..

[6] B. Pratt, M. Caffrey, J. F. Carroll, P. Graham, K. Morgan, and M. Wirthlin, "Fine-grain SEU mitigation for FPGAs using partial TMR," IEEE Trans. Nucl. Sci., vol. 55, no. 4, pp. 2274–2280, Aug. 2008.
[Online]. Available: http://ieeexplore.ieee.org/document/4636895/

[7] G. He, S. Zheng, and N. Jing, "A hierarchical scrubbing technique for SEU mitigation on SRAM-based FPGAs," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 28, no. 10, pp. 2134–2145, Oct. 2020. [Online]. Available: https://ieeexplore.ieee.org/document/9162497/

[8] C. Bolchini, A. Miele, and C. Sandionigi, “A novel design methodology for implementing reliability-aware systems on SRAM-based FPGAs,” IEEE Trans. Comput., vol. 60, no. 12, pp. 1744–1758, Dec. 2011

[9] L. Pereira-Santos, G. L. Nazar, and L. Carro, “Exploring redundancy granularities to repair real-time FPGA-based systems,” Microprocess. Microsyst., vol. 51, pp. 264–274, Jun. 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0141933117302272

[10] Y. Li et al., “Feedback-based low-power soft-error-tolerant design for dual-modular redundancy,” IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 26, no. 8, pp. 1585–1589, Aug. 2018.