Multiple Bit Upset correction in SRAM based FPGA using self repairable Erasure codes

S.Aishwarya
M.E VLSI Design
Department of ECE
Syed Ammal Engineering College
Ramanathapuram,Tamilnadu,India

Dr.G.Mahendran,M.E,Ph,D
Professor and Head
Department of ECE
Syed Ammal Engineering College
Ramanathapuram,Tamilnadu,India.

Abstract— As technology scales, Multiple Cell Upsets (MCUs) have become more common and affects a larger number of cells in SRAM based FPGA. Occurrence of these errors in the configuration frames of a Field Programmable Gate Arrays (FPGA) device permanently affects the functionality of the design. Triple Modular Redundancy (TMR) along with the Tiling technique has been used to reduce the latency and to reduce the errors that occur in the SRAM based FPGA. 2-dimensional based Hamming code has also been used. But the errors could be corrected only upto two to four adjacent bits. The lost content could not be recovered. A low-cost error-detection code is used to detect MCUs in configuration frames as well as a generic scrubbng scheme to reconstruct the erroneous configuration frame based on the concept of Erasure codes. Horizontal and Vertical parity bits have been used to avoid redundant data. Hence another method used to detect and correct those Multiple Bit Upsets has been proposed by using Erasure codes. The proposed scheme does not require any modification to the FPGA architecture. Self repair scheme is proposed to improve the system reliability further: Implementation of the proposed scheme on a Xilinx Virtex-6 FPGA device shows that the proposed scheme can detect 100% of MBUs in the configuration frames with only 3.3% resource occupation.

Keywords—component; FPGA; Erasure codes; Soft Errors ; DMC

I. INTRODUCTION

Static RAM based Field-Programmable Gate Arrays (FPGAs) are most widely used in variety of applications mainly due to short time-to-market time, flexibility, high density, and cost-efficiency. SRAM-based FPGA stores logic cells configuration data in the static memory organized as an array of latches. FPGA is used for designing complex digital circuits. Power consumption is also reduced by using SRAM. The power consumption of SRAM varies widely depending on how frequently it is accessed; it can be as power-hungry as dynamic RAM, when used at high frequencies, and some ICs can consume many watts at full bandwidth. On the other hand, static RAM used at a somewhat slower pace, such as in applications with moderately clocked microprocessors, draws very little power and can have a nearly negligible power consumption when sitting idle – in the region of a few micro-watts. Several techniques have been proposed to manage power consumption of SRAM-based memory structures. FPGA device customizable by SRAM consists of an array of programmable logic blocks interconnected by a programmable routing network and I/O blocks. SRAM-based FPGA devices are becoming popular because of their high performance, reduced development cost and re-programmability.FPGAs based on a nanometer technology with denser integration schemes. Memories are one of the most widely used elements in electronic systems. Radiation in the environment seriously affect the functionality of a circuit. A single-event upset (SEU) occurs when a charged particle, present in the environment, hits the silicon of a circuit introducing an error in the system. Such errors in FPGA device affects the functionality of the mapped design also called as Soft errors. A soft error will not damage a system’s hardware, the only damage is to the data that is being processed in the memory. To address this issue, Built-in Current Sensors (BICS) have recently been applied in conjunction with Single Error Correction/Double Error Detection (SEC-DED) codes to protect memories from MBUs. But by using those methods only SEU could be corrected. For both the detection and correction of errors, a generic scrubbng scheme to reconstruct the erroneous configuration frame based on the concept of Erasure coding algorithm is introduced in this paper. In this type of Erasure coding algorithm, MBUs are detected by using the interleaving distance which is further classified into horizontal and vertical parity.

II. BACKGROUND

A. FPGA Configuration Frames

A Field-Programmable Gate Array (FPGA) is an integrated circuit designed which is configured by a customer or a designer after manufacturing. FPGAs contain an array of programmable logic blocks, and the array of reconfigurable interconnects that allows the blocks to be wired together, like many logic gates that can be inter-wired in different configurations. The configuration memory of such FPGAs are organized into number of configuration frames that are the smallest addressable units and occupies maximum part of the SRAM cells of the FPGA device. The number and the size of
these configuration frames may vary depending upon the device. Now we are in nanometer technology where 14nm technology is being utilized nowadays. In this type of technology we have the closely packed arrangement of configuration frames. So, most probably there are many chances for MBU occurrence.

Figure 1 depicts the FPGA architecture with Logic blocks which can be configured which could perform complex combinational functions, or merely simple logic gates like AND and XOR. In most FPGAs, logic blocks also include memory elements, which may be simple flip-flops or more complete blocks of memory.

![FPGA architecture with I/O blocks](image)

**Fig 1. FPGA architecture with I/O blocks.**

**B. Static random-access memory (SRAM)**

The Static random-access memory which is shown in figure 2 (SRAM or static RAM) is a type of semiconductor memory that utilizes bistable latching circuitry (Flip-flop) to store each bit.

![6T SRAM](image)

**Fig 2. 6T SRAM**

The term static differentiates it from dynamic RAM (Dynamic random-access memory) which must be periodically refreshed. SRAM exhibits data remanence, but it is still volatile in the conventional sense that data is eventually lost when the memory is not powered. SRAM is more expensive and less dense than DRAM and is therefore not used for high-capacity, low-cost applications such as the main memory in personal computers. The power consumption of SRAM varies widely depending on how frequently it is accessed; it can be as power-hungry as dynamic RAM, when used at high frequencies, and some ICs can consume many watts at full bandwidth. On the other hand, static RAM used at a somewhat slower pace, such as in applications with moderately clocked microprocessors, draws very little power and can have a nearly negligible power consumption when sitting idle—in the region of a few micro-watts. Several techniques have been proposed to manage power consumption of SRAM-based memory structures.

**C. Soft Error**

Soft error is an error occurrence in a computer's memory system that changes an instruction in a program or a data value. Soft errors typically can be remedied by cold booting the computer. A soft error will not damage a system's hardware; the only damage is to the data that is being processed. There are two types of soft errors: chip-level soft error: These errors occur when the radioactive atoms in the chip's material decay and release alpha particles into the chip. Because an alpha particle contains a positive charge and kinetic energy, the particle can hit a memory cell and cause the cell to change state to a different value. The atomic reaction is so tiny that it does not damage the actual structure of the chip. Chip-level errors are rare because modern memory is so stable that it would take a typical computer with a large memory capacity at least 10 years before the radioactive elements of the chip's materials begin to decay. System-level soft error: These errors occur when the data being processed is hit with a noise phenomenon, typically when the data is on a data bus. The computer tries to interpret the noise as a data bit, which can cause errors in addressing or processing program code. The bad data bit can even be saved in memory and cause problems at a later time. Several techniques have been employed in order to reduce those soft errors and Multiple Bit Upsets. Modular Redundancy has been used to reduce the latency and to reduce the errors that occur in the SRAM based FPGA. Also 2-d based Hamming code has been used. But the errors could be corrected only up to two to four adjacent bits. Another method used is Erasure codes. Erasure coding is a method of data protection in which data is broken into fragments that are expanded and encoded with a configurable number of redundant pieces of data and stored across different locations, such as disks, storage nodes or geographical locations.

**D. Erasure Codes**

The goal of erasure coding is to enable data that becomes corrupted to be reconstructed by using information about the data that is stored elsewhere in the array—or even in another location. Erasure coding can be useful with large quantities of data and any applications or systems that need to tolerate failures, such as disk array systems, data grids, distributed storage applications, object stores and archival storage. One common current use case for erasure coding is
object-based cloud storage. It works by creating a mathematical function to describe a set of numbers so that they can be checked for accuracy and recovered if one is lost. Otherwise known as polynomial interpolation or oversampling, this is the key concept behind erasure coding methods that are implemented most often using Reed-Solomon codes.

If a bit flip occurs, the next time the memory word is read, it will be corrected and written back. The main risk is that if a memory is not processed for long time then it may lead to several Multiple Bit Errors. In order to avoid that Memory Scrubbing technique is done, in which every location in the memory is read on a periodic basis. This can be implemented by using the memory controller that, during the idle period checks for any single bit errors that occur in the configuration frames of SRAM based FPGA.

![Fig 3. Erasure Coding Technology](image)

The SPICE netlist and the memory layout as well as the radiation environment information are provided as inputs to the tool to compute the distribution of generated current pulses for each cell according to a nuclear database. Afterward, the SEU and MBU rates are extracted by injecting the obtained current pulses in the SPICE netlist. Using this commercial tool, the occurrence probabilities of neutron-induced MBU patterns in the terrestrial environment on an SRAM memory designed for a 45-nm technology. The neutron energy distribution is described according to the JEDEC89a standard. Furthermore, the secondary particles’ reaction that occurs when neutrons interact with the atoms in the CMOS structure is modeled according to a nuclear database.

In information theory, an Erasure Coding technology which is shown in the figure 3, is a Forward Error Correction (FEC) code for the binary erasure channel, which transforms a message of \( k \) symbols into a longer message (code word) with \( n \) symbols such that the original message can be recovered from a subset of the \( n \) symbols. The fraction \( r = k/n \) is called the code rate, the fraction \( k'/k \), where \( k' \) denotes the number of symbols required for recovery, is called reception efficiency. Optimal erasure codes have the property that any \( k \) out of the \( n \) code word symbols are sufficient to recover the original message (i.e., they have optimal reception efficiency). Optimal erasure codes are maximum distance separable codes (MDS codes). Optimal codes are often costly (in terms of memory usage, CPU time, or both) when \( n \) is large. Except for very simple schemes, practical solutions usually have quadratic encoding and decoding complexity.

Self repairing scheme has been also presented which occupies only 3.3% of logic blocks and 0.9% of block RAMs out of the total available resources in the employed FPGA device. The results also shows that our proposed scheme provides the highest recovery coverage with considerably less area overhead compared with that of the existing solutions at the expense of a negligible increase in the mean time to repair (MTTR). In this paper, I generalize the proposed solution to achieve 100% coverage of MBUP. This technique shows a minimal area overhead of 0.3% with respect to the interleaved 2-D (12D) parity technique, while the recovery time is somewhat same. Proposed technique eliminates the need for such an external memory. On the other hand, it significantly reduces MTTR compared with that scheme. There are many types of block codes, but among the classical ones the most notable is Reed-Solomon coding because of its widespread use on the Compact disc, and in hard disk drives. Other examples of classical block codes include BCH, Multidimensional Parity, and Hamming codes. Hamming ECC is commonly used to correct NAND flash memory errors. This provides single-bit error correction and 2-bit error detection.

### III. LOW COST MBU DETECTION

#### A. MBU Patterns

MBU patterns mostly occur mainly due to the Soft Errors. These errors occur mainly due to the alpha and beta particle present in the environment when affects more than one cell it is stated as Multiple Bit Upsets. There are many types of block codes, but among the classical ones the most notable is Tiling technique because of its widespread use on the Compact disc, the DVD, and in hard disk drives. Hamming ECC is commonly used to correct memory error correction. This provides single-bit error correction and 2-bit error detection. Hamming codes are only suitable for more reliable Single Level Cell (SLC) NAND. Denser Multi Level Cell (MLC) NAND requires stronger multi-bit correcting ECC such as BCH or Tiling Technique. NOR Flash does not use any error correction.

In order to quantify the MBU correction capability of the proposed scheme, detailed information about the possible MBU patterns and their respective occurrence probabilities should be known. In this regard, a 3-D-TCAD-based neutron particle strike simulation is being conducted by employing a commercial soft error assessment tool. The SPICE netlist and the memory layout as well as the radiation environment information are provided as inputs to the tool to compute the distribution of generated current pulses for each cell according to a nuclear database. Afterwards, the SEU and MBU rates are obtained by injecting the obtained current pulses in the SPICE netlist. Using this commercial tool, I have acquired the occurrence probabilities of neutron-induced MBU patterns in the terrestrial environment on an SRAM memory designed for

**Fig 3. Erasure Coding Technology**

![Fig 3. Erasure Coding Technology](image)
In the complete (traditional) 2-D parity, a parity bit is associated for each row (column) which is constructed by XORing all the bits in that particular row (column). In the I2D parity technique, each horizontal (vertical) parity bit is the XOR of the bits in multiple rows (columns). More specifically, all rows (columns) that are separated by a constant distance of \( h \) (\( v \)) form an interleaving group and all the bits within that interleaving group are covered by only one horizontal (vertical) parity bit. For a \( f \times g \) memory array, the complete 2-D parity have \( f \times g \) parity bits while I2D parity requires only \( h + v \) parity bits. In addition, the number of XOR operations in both schemes is of the same order.

**B. InD Parity**

In memory arrays of those typical microprocessors such as cache units, the parity bits employed for error detection in each memory entry (i.e., word) are computed during each memory access. Hence, from the performance point of view, it is crucial that each memory entry has its own error detection coding. In order to increase the cost-efficiency of error detection for the configuration frames, I introduce the idea of InD parity. The main idea is to use the same parity bit for the bits that are separated by a constant distance to minimize the area overhead of those bits (i.e., interleaved parity). In addition, the parity bits are distributed along several dimensions to increase the detection coverage with respect to probability of MBU patterns as there are some MBU patterns which cannot be detected by using one or two dimensions. The number of parity bits in each dimension theoretically has to be at most equal to the largest MBU spread on that dimension. However, in practice, large MBU patterns are typically detected using the parity bits on the other dimensions. Consequently, the number of parity bits required by this technique is always smaller than this theoretical limit.

**C. Operation of Erasure Codes**

Erasure coding makes use of the Decimal Matrix Code for the Error Detection and Correction. Erasure coding creates a mathematical function to describe a set of numbers so they can be checked for accuracy and recovered if one is lost. Referred to as polynomial interpolation or oversampling, this is the key concept behind erasure codes. In mathematical terms, the protection offered by erasure coding can be represented in simple form by the following equation: \( n = k + m \). The variable “\( k \)” is the original amount of data or symbols. The variable “\( m \)” stands for the extra or redundant symbols that are added to provide protection from failures. The variable “\( n \)” is the total number of symbols created after the erasure coding process. For instance, in a 10 of 16 configuration, or EC 10/16, six extra symbols (\( m \)) would be added to the 10 base symbols (\( k \)). The 16 data fragments (\( n \)) would be spread across 16 drives, nodes or geographic locations. The original file could be reconstructed from 10
verified fragments. In the proposed error detection technique, we exploit the fact that the sizes of large MBU patterns are typically much smaller than the size of a configuration frame. Since an MBU incident affects several bits in a localized manner, the bits that are located far enough cannot be simultaneously affected with one MBU incident. Therefore, having separate parities for such bits in configuration frames neither increases the error detection capability nor improves the performance, rather only imposes unnecessary area overhead.

Fig 6 (a) Data blocks + parity (b) Recovery of erased block.

Figure 5 shows how the erased content could be recovered by using the XOR operation and by using the parity bits. This data-recovery technique is widely implemented in reliable storage devices such as hard disks and CDs, error correction in cache arrays multimedia multicasting and signal transfer protocols. A variety of erasure codes with different recovery coverage, and encoding/decoding complexity are presented. The area overhead of an erasure code is expressed as the ratio of redundant blocks to the data blocks (i.e., n/m). The recovery coverage of an erasure code is defined as the maximum number of erasures that could be tolerated. For the optimal erasure codes, the recovery coverage is equal to the number of redundant blocks (i.e., n). Erasure codes are not proposed to detect or correct errors rather to retrieve the original blocks when a subset of blocks is not available (i.e., erased). However, once an error is detected in some blocks, by considering that those blocks are not available, the original blocks could be recovered by means of an erasure code.

D. Proposed Methodology of DMC

The Multiple Bit Upsets are detected by using the horizontal and vertical parity techniques and then Erasure coding methodology is followed.

Horizontal parity could be detected by using the data bits.

\[ H_1H_2H_3H_4H_5 = D_1D_2D_3D_4D_5 + D_1D_2D_3D_4D_5 \]

Vertical Parity could be detected by using the vertical data bits.

\[ V_0 = D_0 \oplus D_{16} \]
\[ V_1 = D_1 \oplus D_{17} \]

The flow chart of the self-healing technique is proposed which consists of the Horizontal and Vertical Parity bits. Redundant bits are found out from which MBUs are found out. The lost content could be recovered by using the syndrome calculation from which the error could be corrected.

Fig 7. Horizontal and vertical parity

Fig 8. Block diagram of self-healing technique

Good error control performance requires the scheme to be selected based on the characteristics of the communication channel. Common channel models include memory-less models where errors occur randomly and with a certain probability, and dynamic models where errors occur primarily in bursts. Consequently, error-detecting and correcting codes can be generally distinguished between random error-detecting/correcting and burst-error-detecting/correcting. Some codes can also be suitable for a mixture of random errors and burst errors. If the channel capacity cannot be determined, or is highly variable, an error-detection scheme may be combined with a system for retransmissions of erroneous data. This is known as automatic repeat request (ARQ), and is most notably used in the Internet.

The proposed has been simulated and the synthesis report can be obtained by using Xilinx ISE 12.1i. The various parameters used for computing existing and proposed systems with Spartan-3 processor are given in the table.

<table>
<thead>
<tr>
<th>s.no</th>
<th>Parameter</th>
<th>Existing</th>
<th>Proposed</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>IOB's</td>
<td>192</td>
<td>160</td>
</tr>
</tbody>
</table>
E. Performance Analysis

The Figure given below is shown that there is a considerable reduction in time and area based on the implementation results which have been done by using Spartan-3 processor. The proposed algorithm significantly reduces area consumption when compared to the existing system.

The implementation results reveal that the proposed scheme occupies only 1% of memory and 3% of logic resource on Xilinx Virtex-6 device. Furthermore, the error correction latency is also very small (0.35 ms for 50 clusters). These results confirm that the proposed scheme is a practical solution for MBU mitigation in FPGA configuration frames.

IV CONCLUSION

Transient multiple cell upsets (MCUs) are becoming major issues in the reliability of memories exposed to radiation environment. To prevent MCUs from causing data corruption, more complex error correction codes (ECCs) are widely used to protect memory, but the main problem is that they would require higher delay overhead. In this project, we presented a cost-efficient scheme based on erasure codes for MBU detection and correction in the configuration frames of SRAM-based FPGAs with minimum redundancy. These results confirm that the proposed scheme is an area efficient solution for MBU mitigation in FPGA configuration frames.

In this paper, a cost-efficient scheme based on erasure codes for MBU detection and correction in the configuration frames of SRAM-based FPGAs is presented. This scheme is implemented as a generic soft core alongside with the user design and does not require any changes to the existing FPGA architecture. Compared with the previous solutions, our scheme provides the highest level of MBU protection at very low costs with a negligible recovery time.

REFERENCES
