Improving FPGA Design Reliability Using Embedded Hamming Scheme

Shyamsundar Venkataraman, Rui Santos, Anup Das and Akash Kumar
National University of Singapore
Email: {shyam, elergvds, akdas, akash}@nus.edu.sg

I. Introduction

Single Event Upsets (SEUs) inadvertently change the configuration bits of Static-RAM (SRAM)-based Field Programmable Gate Arrays (FPGAs), leading to erroneous output until the error has been corrected. Scrubbing using an Error Correction Code (ECC) such as hamming is a popular method to correct such faults. However, current works either require a large external memory to store the ECCs or can at most correct only one error in a frame. This paper proposes a novel bit-interleaved embedded hamming scheme along with scrubbing, to correct single (SBUs) and multi-bit upsets (MBUs) in SRAM-based FPGAs. This scheme does not require an external memory to store the ECCs, as they are embedded within the configuration memory itself. Experiments conducted show that the proposed scheme can handle multiple errors per frame, with an embedding efficiency of over 99.3%.

II. Motivation

The utilisation of the MCNC Benchmarks' frames were computed from the Xilinx Essential Bits tool. From this, it was observed that more than 90% essential frames (frames that are relevant to the user design) are only half or less than half utilised. These frames contain more than 50% of non-essential bits that can be used to store ECCs. Since less than 0.25% frames were found to have a 100% utilisation, it would not hinder the effectiveness of embedding ECCs in the non-essential frames. These observations motivate the use of non-essential bits to store ECC bits instead of an external memory.

III. Proposed Scheme

The scheme proposed, uniformly interleaves the bits of each frame into a number of *sub frames* and uses the non-essential bits of the frames to embed hamming parity bits, making the entire sub frame hamming code compliant. At runtime, each frame of the user design is readback and the sub frames are decoded to check for errors. If an error is detected, the embedded hamming code is used to find the location of error and the corrected frame is written back to the design. The following sections discuss the scheme in detail.

An *embedded hamming code* manipulates the non-essential bits of a frame such that the entire frame becomes hamming code compliant. A typical hamming code consists of a number of data bits (d_i) with parity bits (p_i) embedded in indices (ix) that are powers of two. These parity bits are computed by choosing a different set of data bits for each parity bit as specified by a *hamming check matrix (H)*. Each row of H corresponds to a parity bit while the columns specify whether a data bit is included in the parity bit of that row. Computing the values of the parity bits using H, results in a set of XOR equations with the different non-essential bits. Solving these equations gives the values of the non-essential bits that make the frame hamming code compliant.

If the embedded hamming code was applied to the entire frame, only SBUs can be corrected, as hamming code is capable of correcting only one error. Bit interleaving has been considered in combination with embedded hamming to increase the number of errors than can be detected and corrected. A bigger frame is decomposed into smaller sub frames, each of them containing some essential and non-essential bits. Moreover, the bit interleaving also spreads out the non-essential bits of the frame to all sub frames. Hence, this technique enables to correct one error in every sub frame.

IV. Results

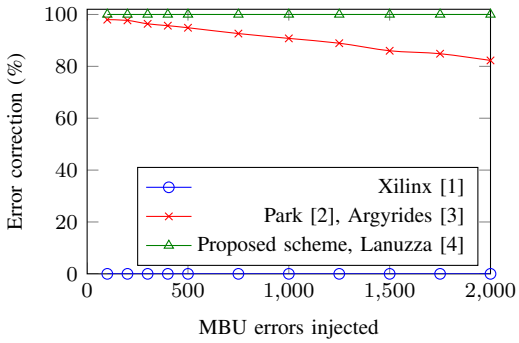

Fig. 1. MBU error correction performance

All experiments were conducted on a Virtex-6 FPGA board. Since the reserved bits of the Xilinx ECC (13 bits) are not used by the proposed scheme, these bits are used to track the sub frames that were not embedded with hamming code. The proposed scheme has an embedding efficiency of over 99% for all MCNC benchmark designs considered. Moreover, the proposed scheme is able to correct more than 90% of the errors when up to $5,000$ SBUs or $2,000$ MBUs are injected (shown in Fig. 1). Moreover, its performance is better or on par with the other state-of-the-art techniques. The proposed scheme also excels over other works in terms of memory overhead, since it requires no extra memory to store the ECCs, while other works require at least 1 MB of extra memory for the same. On average, the time taken to embed hamming code into the designs was less than a minute.

Hence, our idea proposes an innovative technique to detect and correct SEUs in SRAM-based FPGAs as has been verified by the experiments performed.

References

[1] K. Chapman, "SEU Strategies for Virtex-5 Devices," Xilinx Application Note, 2010.

[2] S. P. Park, D. Lee, and K. Roy, "Soft-Error-Resilient FPGAs Using Built-In 2-D Hamming Product Code," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 2012.

[3] C. Argyrides, D. Pradhan, and T. Kocak, "Matrix codes for reliable and cost efficient memory chips," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 2011.

[4] M. Lanuzza, P. Zicari, F. Frustaci, S. Perri, and P. Corsonello, "A self-hosting configuration management system to mitigate the impact of Radiation-Induced Multi-Bit Upsets in SRAM-based FPGAs," in *IEEE International Symposium on Industrial Electronics*, 2010.