ISSN: 2456 - 3935
International Journal of Advances in Computer and Electronics Engineering
Volume: 02 Issue: 02, February 2017, pp. 01 – 05

FPGA Implementation of Double Error Correction Orthogonal Latin Squares Codes

E. Jebamalar Leavline
Assistant Professor, Department of ECE, Anna University, BIT Campus, Tiruchirappalli, India
Email: jebilee@gmail.com
G. Manikandan
UG Scholar, Department of ECE, Anna University, BIT Campus, Tiruchirappalli, India
Email: maniudhai007@gmail.com
N. Manivasagam
UG Scholar, Department of ECE, Anna University, BIT Campus, Tiruchirappalli, India
Email: manivasak92@gmail.com

Abstract: *Error correction codes (ECCs) are widely used to protect memories and other electronic circuits against errors. A class of ECC codes namely OLS codes have been recently proposed to protect caches, interconnections and memories. This OLS code is a member of the majority logic (ML) decodable codes family. This paper deals with the FPGA implementation of Double Error Correction Orthogonal Latin Squares (OLS). The OLS decoder scheme has a reduced soft error cross section compared with other implementations. This ECC technique is implemented in SRAM-based FPGAs since they are sensitive to soft errors. The implementation of OLS code is done in Xilinx environment and it is evaluated in terms of device utilization and delay.*

Keyword: *Double error correction; Error correction code; FPGA; OLS code; SRAM;*

1. INTRODUCTION

In recent years, error correcting codes (ECCs) have been used increasingly to enhance the system reliability and the data integrity of computer semiconductor memory subsystems [1]. Error correction codes are widely used to protect memories and other electronic circuits against errors. Forward error correction (FEC) code means that the symbol alphabet consists of just two symbols (0 and 1) [2]. When ECCs are used, in addition to correcting a given number of errors, the code can also detect errors exceeding that number. This ensures that uncorrectable errors are detected and therefore silent data corruption does not occur. There are two types of error correcting codes namely single error correction code and double error correction codes.

A single error correcting code is effective in increasing system reliability. For memory systems in need of increased reliability, a single error correcting and double error detecting (SED-DED) code can be incorporated. These codes need an additional check bit to indicate overall parity. A single error correcting code has the potential to detect double error [3].

A SEC code is generated by appending certain parity check bits to the data bits. These check bits are generated with the help of the parity check matrix of the SEC code. Whenever there is a single error, the check bits will indicate the position of the bit in error and they will indicate a no error condition by an all-0 pattern. The modified SEC code is capable of detecting a significant amount of total double error possibilities without using an extra check bit. More advanced ECCs have been proposed in the literature for memory applications [3]. The recently proposed Orthogonal Latin square codes (OLS) [4] can be used for as a SEC-DEC code.

A Latin square of order (size) m is an m x m square array of the digits 0,1,...,m-1, with each row and column is a permutation of the digits 0,1,…, m-1.Two Latin squares are orthogonal if, when one Latin is superimposed on the other, every ordered pair of elements appears only once. These are framed up on the perception of Latin squares. It is a matrix of order m and has up to m-1 permutations. When superimposed, each diagonal pair elements will show one time, such a kind of squares is called orthogonal ones.

This paper deals with the FPGA implementation of OLS codes proposed in [5] that have data bits of number m^2. They have check bits of number 2tm, where t is the number of errors that the code corrects.

The rest of the paper is organized as follows. Section 2 reviews the related literature. Section 3 presents the methodology and Section 4 discusses the experimental results. Section 5 concludes the paper.

2. LITERATURE REVIEW

Many error detection and correction methods have

ISSN: 2456 - 3935
International Journal of Advances in Computer and Electronics Engineering
Volume: 02 Issue: 02, February 2017, pp. 01 – 05

been proposed in literature. Pedro et al. proposed a concurrent error detection for orthogonal Latin squares encoders and syndrome computation for protecting memories against errors. This error checking scheme has a significant delay. This was achieved by performing the checking in parallel with the writing of the data in case of encoder and in parallel with the majority voting and error correction in case of decoder [6].

Avijit Dutta and Nur A.Touba proposed a multiple bit upset tolerant memory using selective cycle avoidance based SEC-DED-DAEC code. The ECC methodology described in this paper adds the ability to correct adjacent errors at very little cost over conventional SEC-DED codes. The only drawback is the possibility of miss-correction for a small subset of multiple errors [7].

Seung Eun Lee proposed an adaptive error correction in orthogonal Latin square codes for low-power, resilient on-chip interconnection network. In this paper, they proposed an adaptive ECC which provides opportunity and flexibility of adaptively changing the error correction capability according to interconnection's reliability level, reducing the power consumption of the interconnection network [8].

Rudrajit Datta and Nur A. Touba proposed a a graph theoretic approach for generating burst-error correcting codes from orthogonal Latin square codes. This paper proposed a scheme by which an orthogonal Latin square code (OLS) can be modified to correct burst-errors of specific length. Conventional SEC-DED codes can only detect double-bit error, but cannot correct [9].

By carefully selecting and ordering the columns in the H-matrix for an SEC-DED code, it is possible to correct all adjacent double-bit errors in addition to correcting all single bit errors thereby creating an SEC-DAEC code. This is very useful since the most likely double-bit errors will be adjacent. The limitation of SEC-DAEC codes is that they may not detect all non-adjacent double-bit errors [9].

While scaling up conventional parity check code for correcting multi-bit errors requires less check bits, the additional number of syndromes that needs to be stored for correction purposes makes parity check matrices an unattractive solution for multi-bit errors.

Dongsoo Lee, and Kaushik Roy designed soft-error-resilient FPGAs using built-in 2-D Hamming product code. Radiation-induced soft error rate degrades the reliability of static random access memory (SRAM)-based field programmable gate arrays (FPGAs) [10].

To provide efficient 2-D HPC in a built-in logic, they also proposed a new 2-D SRAM buffer. Using the proposed multi bit error correction scheme, system availability of an SRAM-based FPGA can be more than 99.9999999% with SRAM. The simulation results showed that 2-D single bit corrections can provide very high multi bit ECs. This work also included an optimized hardware implementation of the proposed scheme using a new SRAM [10].

C.L Chen and M. Y. Hsiao proposed an error correcting code for semiconductor memory devices. This system increases the system reliability of semiconductor memory. Error correcting codes are used to correct soft error as well as hard errors. In this paper, they reviewed the current status of error correcting code can be handled through Monte Carlo methods [11].

3. METHODOLOGY

3.1 OLS Encoder Implementation

The OLS encoder circuit uses XOR logic gates for which the output is 1 only when either X is equal to 1 or Y is equal to 1, but not when both X and Y are equal to 1. The encoder circuit has 16 bit input and 16 bit output. The basic OLS encoder circuit uses 4 input EX-OR gate as shown in Figure 1 [5].

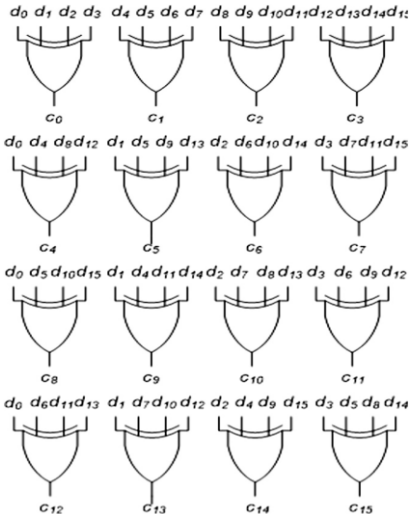

Figure 1. OLS encoder implementation

3.2 Decoder Implementation

Decoding is carried out using majority circuits. In this, if the majority number of inputs is high then the output is high, and if majority number of inputs is low then the output is low. The majority function produces 1 when more than half of the inputs are 1, it produces 0 when more than half of the inputs are 0. Majority circuit is a logical gate used to reduce circuit complexity.

The traditional decoder is implemented using two circuits. The first one is vote to correct another one is correct by voting. The EX-OR logic circuit used for decoder circuit and input to parity check bit added. The EX-OR gate output is majority circuit input. The OLS code can be decoded using OS-MLD. This makes the decoding process simple and fast. In OS-MLD decoded value of each bit is obtained by taking

ISSN: 2456 - 3935
International Journal of Advances in Computer and Electronics Engineering
Volume: 02 Issue: 02, February 2017, pp. 01 – 05

the majority value of the recomputed parity check equations (syndrome bits) in which that bit participates [5].

If just one error occurs in one of the code word bits being the rest of the bits correct), the syndrome equations in which that erroneous bit participates will take a value of 1. A 1 in the associated recomputed parity checks implies that an error has affected that bit, and therefore needs to be corrected. The majority logic will work if up to t errors have affected the system, since in that case t − 1 equations will be wrong with the t + 1 correct ones.

3.1.1Vote to Correct

The majority vote logic is implemented in two different ways. The first option is to re-compute the four parity check bits and take a majority vote among them. When there is a majority of ones, an error has occurred and the bit is corrected. The correction can be done with an XOR gate (Figure 2) [5].

3.1.2 Correct by Voting

The four parity check bits are partially recomputed to obtain the bit being decoded and a majority vote among those and the original bit is used to obtain the decoded bit. This method is named as 'correct by voting' (Figure 3) [5].

3.1.3 Optimized Decoder implementation-Merged Vote to Correct

This decoder is illustrated in Figure 4 and named "merged vote to correct". The two usual decoder implementations and the 'merged to correct' are evaluated in order to compare the speed and resource usage.

4. EXPERIMENTAL RESULTS AND DISCUSSION

This section deals with the experimental setup, experimental results and discussions. The experimental setup deals with the hardware and software used for implementation and the experimental results deals with the results of the different architectures of error correction codes.

Here the outputs of the each circuit are displayed below as their inputs. Then the results discussion deals with their corrected codes and device utilization summary of different circuits.

4.1 Experimental Setup

FPGA Spartan-3E hardware is used for implementation in Xilinx environment with the following specifications.

Device: XCS100E.
Package: cp132.
Processor: Intel(R) Pentium(R)CPU B950@2.10 GHz
RAM: 2 GB
System type: 64-bit operating system
Software: Xilinx ISE 13.3
Family: Spartan-3E.
Top level source type: HDL.
Speed: -4.
Synthesis Tool: XST (VHDL/VERILOG).
Simulator: ISIM (VHDL/VERILOG)
Preferred Language : Verilog
VHDL source analysis: VHDL-93

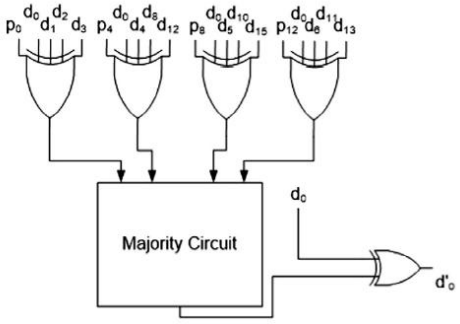

Figure 2. Vote to correct method

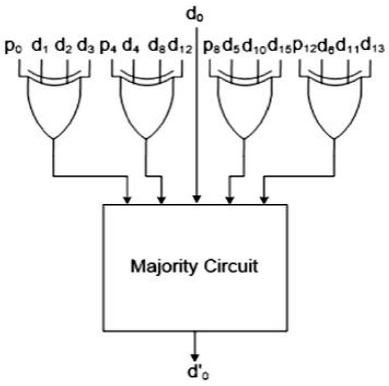

Figure 3.Correct by Vote

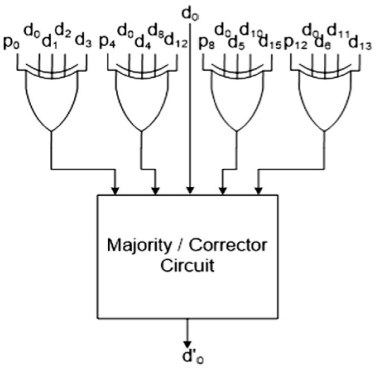

Figure: 4 Merged vote to correct

4.2 Experimental Results

Figure 5 is the output for encoder implementation. For the input D= 0011111111111111, the output is C=0001110011001100. Figure 6 shows output of traditional decoder implementations vote to correct method. For the single error correction inputs

ISSN: 2456 - 3935
International Journal of Advances in Computer and Electronics Engineering
Volume: 02 Issue: 02, February 2017, pp. 01 – 05

D=1011111111111111, and P=1111, it produces the output Y=1. For the double error correction inputs D=1001111111111111, and p=1111, it produces the output of Y=1.

Figure 7 shows output of traditional decoder implementations vote to correct method. For the single error correction inputs D=1011111111111111, and P=1111, it produces the output Y=1. For the Double error correction inputs D=1001111111111111, and p=1111, it produces the output of Y=1. Figure 8 shows output of traditional decoder implementations vote to correct method. For the single error correction inputs D=1011111111111111, and P=1111, it produces the output Y=1. For the Double error correction inputs D=1001111111111111, and p=1111, it produces the output of Y=1.

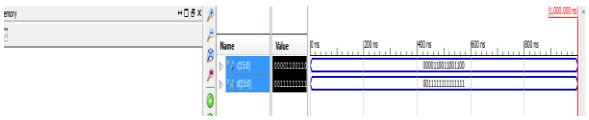

Figure 5. OLS encoder output

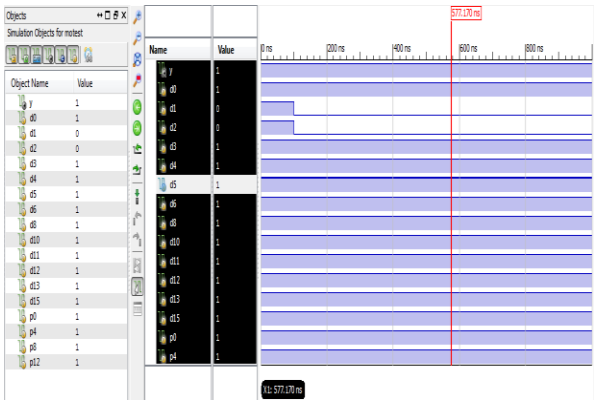

Figure 6. Output of vote to correct

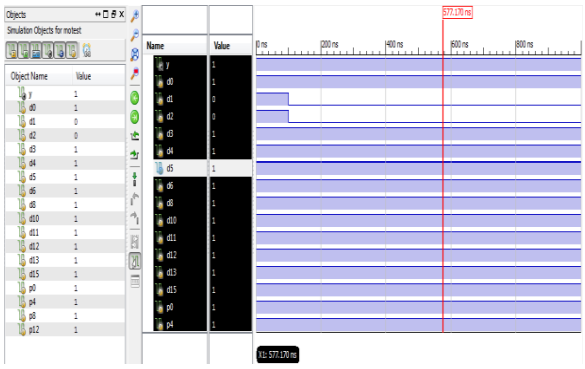

Figure 7. Output of correct to vote

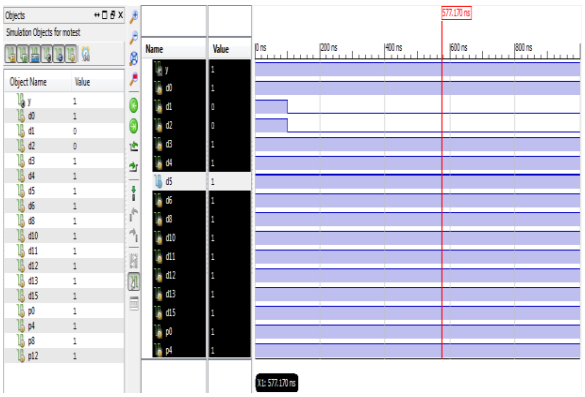

Figure 8 Output of merged vote to correct

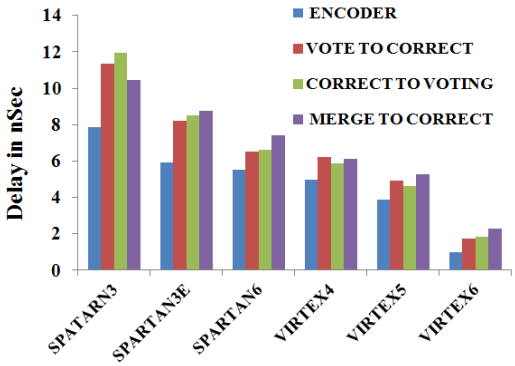

Figure 9. Graphical representation maximum combinational path delay of various devices

4.3 Discussion

Figure 5 to Figure 8 show the output of different encoding and decoding methods and it is obvious that all the circuits are working perfectly. From these figures it is observed that the errors significantly reduced in majority circuits and we can implement double error correction method by using majority voting methods such as vote to correct, correct by voting and merged vote to correct methods. Figure 9 shows the comparison of maximum combinational path delay of various devices. Table 1 shows the number of slices used for different architectures. Table 2 shows the number of LUTs used for different architectures.

ISSN: 2456 - 3935
International Journal of Advances in Computer and Electronics Engineering
Volume: 02 Issue: 02, February 2017, pp. 01 – 05

TABLE 1 COMPARISON OF NUMBER OF SLICES FOR VARIOUS DEVICES

Device	Spartan3	Spartan3e	Spartan6	Virtex4	Virtex5
Method	Usage/ available	Usage/ available	Usage/ available	Usage/ available	Usage/ available
Encoder	9/768	9/960	16/2400	9/5472	16/12480
Vote correct	4/768	4/960	4/2400	4/5472	4/12480
Correct by vote	3/768	3/960	4/2400	3/5472	3/12480
Merge to correct	4/768	4/960	4/2400	4/5472	4/12480

TABLE 2 COMPARISON OF NUMBER OF LUTS FOR VARIOUS DEVICES

Device	Spartan3	Spartan3e	Spartan6	Virtex4	Virtex5
Method	Usage/ available	Usage/ available	Usage/ available	Usage/ available	Usage/ available
Encoder	7/1536	7/1920	4/2400	7/10944	16/12480
Vote correct	7/1536	7/1920	4/2400	4/10944	4/12480
Correct by vote	6/1536	6/1920	4/2400	6/10944	3/12480
Merge to correct	8/1536	8/1920	4/2400	8/10944	4/12480

4.3 Discussion

Figure 5 to Figure 8 show the output of different encoding and decoding methods and it is obvious that all the circuits are working perfectly. From these figures it is observed that the errors significantly reduced in majority circuits and we can implement double error correction method by using majority voting methods such as vote to correct, correct by voting and merged vote to correct methods. Figure 9 shows the comparison of maximum combinational path delay of various devices. Table 1 shows the number of slices used for different architectures. Table 2 shows the number of LUTs used for different architectures.

5. CONCLUSION

In this paper, double error correction–orthogonal Latin square codes are implemented in FPGAs. The results show that by adapting the decoder to the FPGA structure, significant savings can be obtained in the number of LUTs required to implement the decoders. OLS codes have been recently proposed to protect caches, interconnections and memories. These implementations are more appropriate for applications that suffer a higher number of soft errors.

REFERENCES

[1] C.L. Chen, M.Y. Hsiao. (1984), "Error-correcting codes for semiconductor memory applications: A state-of-the-art review" *IBM Journal of Research and Development*, Vol. 28, Issue. 2, pp. 124–134.

[2] S. Lin, D.J. Costello (2004), "Error Control Coding", 2nd ed. Prentice-Hall, Englewood Cliffs, NJ

[3] M.Y. Hsiao. (1970), "A class of optimal minimum odd-weight-column SEC-DED codes" *IBM Journal of Research and Development.*, Vol.14, Issue. 4, pp. 395–401.

[4] M.Y. Hsiao, D.C. Bossen, R.T. Chien. (1970), "Orthogonal Latin square codes", *IBM Journal of Research and Development*. Vol.14, Issue.4, pp. 390–394.

[5] Demirci, Mustafa, Pedro Reviriego, and Juan Antonio Maestro. (2016) "Implementing Double Error Correction Orthogonal Latin Squares Codes in SRAM-based FPGAs." *Microelectronics Reliability*, Vol. 56, pp. 221-227.

[6] Reviriego, Pedro, Salvatore Pontarelli, and Juan Antonio Maestro. (2013) "Concurrent error detection for orthogonal Latin squares encoders and syndrome computation." *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, Vol. 21, Issue.12, pp. 2334-2338.

[7] A. Dutta, N.A. Touba, (2007), "Multiple bit upset tolerant memory using a selective cycle avoidance based SEC-DED-DAEC code" Proc. of 25th IEEE VLSI Test Symposium.

[8] S.E. Lee, Y.S. Yang, G.S. Choi, W. Wu, R. Iyer. (2011), "Low-power, resilient interconnection with Orthogonal Latin Squares", *IEEE Design and Test of Computers*, Vol.28, Issue.2, pp.30–39.

[9] Datta, Rudrajit, Nur A. Touba. (2011) "Generating burst-error correcting codes from Orthogonal Latin Square codes--A graph theoretic approach." IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems, pp. 367-373.

[10] Park, Sang Phill, Dongsoo Lee, Kaushik Roy (2012). "Soft-error-resilient FPGAs using built-in 2-D Hamming product code." *IEEE transactions on very large scale integration (VLSI) systems*, Vol. 20, Issue.2, pp. 248-256.

[11] Chen, Chin-Long, M. Y. Hsiao (1984). "Error-correcting codes for semiconductor memory applications: A state-of-the-art review." *IBM Journal of Research and Development*, Vol. 28, Issue.2, pp. 124-134.

.