International Journal of Research (IJR)
e-ISSN: 2348-6848, p- ISSN: 2348-795X Volume 2, Issue 07, July 2015
Available at http://internationaljournalofresearch.org

Error Correction & OLS Code using Ced

P.Suresh[1]; Ayesha Tarranum[2]& D. Vijay Kumar [3]
[1]M.Tech, Dept of ECE,Vijaya Engineering College, Telangana, India.
Email: suresh3221@gmail.com
[2]Associate Professor,Dept ofECE,Vijaya Engineering college, Telangana,India,
Email: ayeshabad14@gmail.com
[3] Associate Professor, HOD,Dept of ECE, Vijaya Engineering college, Telangana,India,
Email: vkumar88.d@gmail.com

ABSTRACT
In advanced electronic circuits the reliability has been a major concern. There are number of mitigation techniques proposed to make sure that the errors do not affect the circuit functionality. Among them, to protect the memories and registers in electronic circuits Error Correction Codes (ECC) are commonly used. Whenever any ECC technique is used, the encoder and decoder circuit may also suffer errors. In this brief, concurrent error detection and correction technique for OLS encoders and syndrome computation is proposed and evaluated. The proposed method efficiently implements a parity prediction scheme that detects all errors that affect a single circuit node using the properties of OLS codes. The results constitute simulation of Verilog codes of different modules of the codes in Xilinx 13.2. The results demonstrate that the CED for OLS encoders & Syndrome Computation are very efficient for the detection and correction of burst errors.

Keywords: Error Correction Codes; Ols Codes; Ced; Ols

1. Introduction

Since many years, for detecting and correcting errors ECCs were used [1], [3]. Researchers have proposed wide range of codes for memory applications. For correcting one bit per word, single error correction (SEC) codes are used in general. Advanced codes which can also correct double adjacent errors [2] are also been studied. But the problem with some complex codes that corrects more errors is generally limited by their impact on delay and power, which in turn will limit their applicability to memory designs [4]. To run-over those concerns, a technique is proposed by the use of codes that are one step majority logic decodable (OSMLD). OS-MLD codes are low-latency decodable codes. So, for protecting memories, they are used [7] [8].

In [10], use of different types of codes has also been discussed. The other type of code that is OS-MLD is orthogonal latin squares (OLS) code [9]. For interconnections [11], memories [13], and caches [12] use of OLS codes have gained a renewed interest, because of the modularity such that error correcting capabilities can be easily adapted to the error rate [11] or to the mode of operation [13]. Typically more parity bits are required for OLS codes than other codes for correcting the same

number of errors. However, due to their modularity and the simple, low delay decoding implementations (as OLS codes are OS-MLD); neutralize this disadvantage in many applications.

A major issue is that the encoder and decoder circuits needed to use (ECCs) may also suffer from error sometimes. Whenever, an encoder is affected by an error, to the memory an incorrect word may be written. In decoder, a correct word may be interpreted as erroneous or an incorrect word may be interpreted as a correct word. Protection of the encoders and decoders for Different ECCs has been studied in [14] and [13]. For example, EG codes were studied in [8]. The protection of Reed-Solomon and Hamming Codes were studied in [14] and [10].

]Finally, the protection of encoders for SEC codes against soft errors was discussed in [18]. The ECC encoder first computes the parity bits, and in majority of the cases the decoder detects and corrects the errors by checking the parity bits. In general, this is known as syndrome computation. In some codes, based on the properties of the code, serial encoding and syndrome computation are performed. But, for low delay parallel implementations are preferred. It is the case for OLS codes which are commonly used in high-speed applications. After syndrome computation, the errors are detected and corrected. This means in encoder and decoder generating and checking the parity bits are the important parts. Therefore, its important issue is its protection.

In this brief, for SRAM memories and caches, the protection of the encoders and syndrome computation for OLS codes are considered. Depending on some definite properties, it is presented that parity prediction is an productive technique to detect and correct errors in the encoder and syndrome computation. For most block codes it is not the case for which parity prediction will not provide effective protection. So, it is an advantage of OLS codes in addition to its modularity and low-decoding capability. The rest of this paper is organized as follows. OLS codes are introduced and the some of the properties are summarized in Section II. The proposed parity prediction scheme is discussed in Section III. Evaluation of its time delay is discussed in Section IV. Finally, a brief conclusion is presented in Section V.

2. Related Work& Implementation

2.1 Orthogonal Latin Squares Codes:

A Latin square of size m is an m × m matrix that has permutations of the digits 0, 1 and m − 1 in Both its rows and columns [15].Two Latin squares are said to be orthogonal if when they are superimposed every ordered pair of elements

appears only once. Oscoda's are derived from OLS [11]. The block sizes for OLS codes are k = m2 data bits and 2 tm parity bits, where t is the number of errors that the code can correct and m is an integer. For a given pairof values of t and m, the corresponding OLS code exists only if there're at least 2t OLS of size m. As mentioned in Section I, OLS codes can be decoded usingOS-MLD. OS-MLD is a simple procedure in which each bit isdecoded by simply taking the majority value of the set of theircomputed parity check equations, in which it participates [6].

The reasoning behindOS-MLD is that when an error occurs in bit di, the recomputed parity checks in which it participates will take a value of one. Therefore,a majority ofones in those recomputed checks is an indication thatthe bit is in error and therefore needs to be corrected. However, it may also occur that errors in other bits different from di provoke a majority of ones that would cause miss correction. For a few codes,their properties ensure that this miss correction cannot occur, and therefore OS-MLD can be used. This is the case for some EG codes,DS codes, and for OLS codes as mentioned in Section I.In particular, OLS are by construction such that:

1) Each data bit participates in exactly 2t parity check bits;
2) Each other data bit participates in at most one of those parity check groups will at most share a one with the existing columns inbits.

These properties ofOS-MLD enables it to be used for a number of errors t or smaller, such that when one error affects a given bit, the remaining t − 1errors can also affect t − 1 check bits in the worst case. Hence a majority of t + 1 triggers the correction of an erroneous bit. In the same way, when a given bit is correct, t errors on other bits will not cause omission as a majority of t + 1 is needed. Fig. 1 explains the use of OS-MLD enabling a simple and fast decoding used for high-speed memories. As described, the parity check matrix H for OLS codes is constructed from the OLS. As an example, the matrix for a code withk = m2 = 16 data bits and 2 tm = 16 parity bits that can correct double errors is shown in Fig. 2. As explained before, the decoding starts by recomposing the parity checks.

The result is the syndrome and a value of one in a given parity check (rows of H) means that an error has been detected in that parity check. A given data bit participates in the parity checks that have a one in the column that corresponds to that bit. For example, the first column that corresponds to the first data bit has ones in positions 1, 5, 9, and 13. For OLScodes, as

International Journal of Research (IJR)
e-ISSN: 2348-6848, p- ISSN: 2348-795X Volume 2, Issue 07, July 2015
Available at http://internationaljournalofresearch.org

described before, the decoding is done by taking a majority vote of the syndrome bits in which the bit participates (1, 5, 9, and13 in our example). If the majority is one, then the data bit is inerror and is corrected by inverting the bit. In the example of Fig. 2,all the data bits (first 16 columns) participate in exactly four paritybits (2t)and each pair of columns share at most one position with avalue of one.For an arbitrary value of k = m2, the H matrix for a DEC OLS code is constructed as follows:

$$H=\begin{bmatrix} M1 \\ M2 \\ M3 \\ M4 \end{bmatrix} I_{4m} \quad (1)$$

WhereI_{4m} is the identity matrix of size 4m and M1, M2, M3, and M4are the matrices of size m × m2 derived from OLS of size m × m.Those matrices are illustrated in the example shown in Fig. 2.In a general case, for an OLS code that can correct t errors, the parity check matrix is constructed using 2t Mimatrices. The Mimatrices have only a one in each of its columns. Therefore, the firstk = m2 columns in H have a weight of 2t . Additionally, as the Latin squares used to derive the Mimatrices are orthogonal, any pair of columns has at most a position with a one in common. This, as discussed before, enables the use of OS-MLD for decoding.

2.2 Proposed Concurrent Error Detection & Correction Technique:

Before discussing the proposed error detection and correction techniques, the standard definition of self checking circuits that are used in this section is presented. During fault-free or normal operation, a circuit receives only a subset of the input space, called the input code space, and produces a subset of the output space, called the output code space. The outputs that are not members of the output code space form the output error space. In general, a circuit may be designed to be self-checking only for an assumed fault set. In this brief, we consider the fault set F corresponding to the single stuck-at fault model [19].

Self-checking property is verified for the circuit if and only if it satisfies the following properties:

1) Self-testing

2) fault secure.

A circuit is said to be running under self-testing if, in the fault set F for each fault f, there is at least one input which belongs to the input code space, for which circuit gives an output that belongs to the output error space. A circuit is to have fault-secure if, in the fault set F for each fault f and for each input belonging to the input

International Journal of Research (IJR)
e-ISSN: 2348-6848, p- ISSN: 2348-795X Volume 2, Issue 07, July 2015
Available at http://internationaljournalofresearch.org

code space, the circuit gives the correct output, or an output that belongs to the output error space. The fault-secure property assures that the circuit gives the correct response, or alerts the presence of a fault that provides an output in the error space. Faults are always detected, since an output is provided for the input which can identify the presence of the fault.

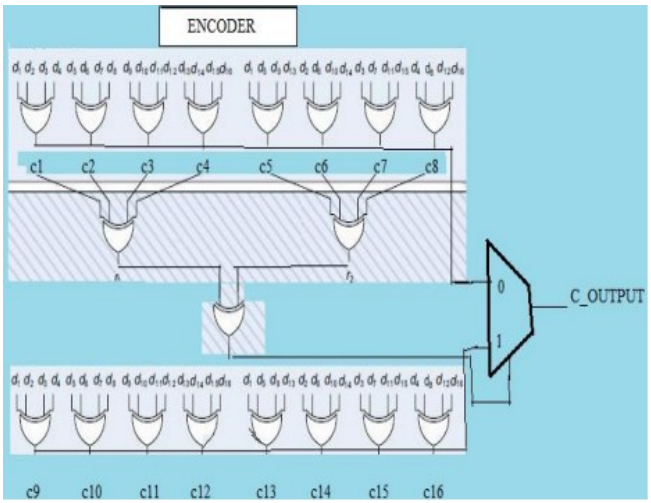

Fig 1: Proposed self-correcting encoder for OLS code with k=16 & t=1.

The proposed technique is based on the use of parity prediction, which is one of the techniques that are used to detect and correct error in typical logic circuits. In our case, the problem is much simpler, given the structure of the OLS codes. For an encoder, the proposed is that the parity of the calculated check bits (ci) is set against the parity of all the check equations. The equation obtained by calculating the parity of the columns in G is simply the parity of all check equations. Since for OLS codes, each column in G will have exactly 2t ones, so null equation is obtained (see, for example, Fig. 1). Therefore, the concurrent error detection (CED) is normally to check

$$C1 \oplus C2 \oplus C3 \oplus \ldots \oplus C2tm=0 \quad (2)$$

$$r1 \oplus r2 = 0 \quad (3)$$

And for Concurrent error detection and correction (CEDC) it is

Where c_i are the check bits of the original circuit and cj represents the check bits of the duplicated circuit. This gives an efficient implementation which is not possible in other ECC codes. For example, in the Hamming code a major part of the columns in G have an odd weight and for some different codes the number is even larger as they are designed to have odd weights and also in that codes we need to correct the codes manually, but in this case the codes are automatically corrected. The input code space of the OLS encoder corresponds to the input space, since the encoder can receive all the possible 2k input configurations. The output code space of the OLS encoder is composed by the outputs satisfying (2), (3) and (4), while the output error space is the complement of the output code space. In order to check whether the output of the OLS encoder belongs to the output code

space or the output error space, a self-checking implementation of a parity checker is used [12].

The checker and correction unit controls the parity of its inputs and is realized with a repetition code. The two outputs (r1, r2) are each equal to the parity of one of two disjoint subsets of the checker inputs (ci), as proposed in [12]. When a set of inputs with the correct parity is provided, the output code {r1, r2} takes the values 00 or 11 and the two bits are compared against each other using a checker to get the output 0. When thechecker receives an erroneous set of inputs, the checker provides the output codes 01 or 10 and so the output checker e will be equal to 1. Also, if a fault occurs in the checker, the outputs are 01 or 10 and the final checker output will be 1. This guarantees the self-checking property of the parity checker [12]. The proposed encoder is illustrated in Fig. 2 for the code with k = 16 and t = 1.

For correction of the errors in the encoder, first the circuit is partially duplicated till the generation of check bits by giving the same input combinations. Then the check bits(cj) of the duplicated circuit are compared against the check bits(ci) of the original (CED) circuit by using the 2X1 multiplexer, where the original circuit check bits (ci) are connected to the first input while the check bits (cj) obtained from duplicated circuit are connected to the second input of multiplexer and the selection bit for that is given from the output(e) of the original circuit, whenever the selection bit is 0, then the original circuit is selected and if it is not equal to 0 then the duplicated circuit gets selected.

The circuit that is proposed can detect and correct any error that affects an odd number of ci bits. For a general code, in most cases there is logic sharing among the calculations of the ci bits [8]. This means that an error may increase to more than one ci bit, and if the number of bits affected is even, then the error is also detected by the proposed scheme.

This means that an error may increase to more than one ci bit, and if the number of bits affected is even, then the error is also detected by the proposed scheme. But this would increase the area of the circuit and also increase the cost compared to an unrestricted implementation. Additionally, even if the error propagates to an odd number of outputs, the delay of each path can be different. Which may cause detecting of only some of the output errors at the clock edge? For OLS codes, as discussed in the previous section a pair of data bits shares at most one parity check. This assures that there isn't any logic sharing among the computation of the ci bits. Therefore, the proposed technique works well to detect and correct all errors that affect a single circuit node.

International Journal of Research (IJR)
e-ISSN: 2348-6848, p- ISSN: 2348-795X Volume 2, Issue 07, July 2015
Available at http://internationaljournalofresearch.org

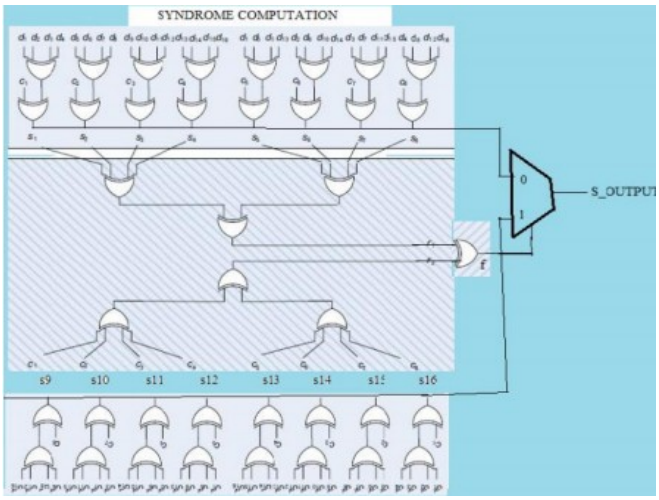

Fig 2: Proposed self-checking and correcting syndrome computation for OLS code with.

k = 16 and t = 1.

The parity prediction for the syndrome computation can be implemented by checking that the following two equations take the same value.

$$r1 = s1 \oplus s2 \oplus s3 \oplus \cdots \oplus s2tm \quad (4)$$

$$r2 = c1 \oplus c2 \oplus c3 \oplus \cdots \oplus c2tm \quad (5)$$

Or else by simply checking the following equation

$$f = r1 \oplus r2 \quad (6)$$

Where si bits are the calculated syndrome bits. The proposed circuit is shown in Fig. 3 for the code with k = 16 and t = 1. If there are any errors in the computed syndrome bits, then for correcting them, the following equation is used:

$$S\_output = f(si) + f(sj) \quad (7)$$

Where si re the syndrome bits of the original circuit while sj are the duplicated circuit syndrome bits. The output code space of the OLS syndrome computation is composed by the outputs given by (3), (4), (5) and (6), while the output error space is the complement of the output code space. The fault-secure property for the syndrome computation is easily demonstrated for the faults in F by observing that the circuits that compute e do not share any gate and both circuits are only composed of XOR gates. Therefore, a single fault could propagate to only one of the outputs, producing an output on the output error space.

Though the circuits are different for encoder and syndrome computation, the syndrome computation circuit operation is similar to that of the operation of encoder, except that in encoder we calculate check bits, here in syndrome computation, syndrome bits are generated. The duplicated circuit of syndrome computation circuit is that partial circuit of the original syndrome circuit i.e., upto the generation of the syndrome bits. The inputs of the duplicated circuit are taken from the original inputs bits. For OLS codes, the cost of the encoder and syndrome computation in terms of the number of two-input XOR gates can be

International Journal of Research (IJR)
e-ISSN: 2348-6848, p- ISSN: 2348-795X Volume 2, Issue 07, July 2015
Available at http://internationaljournalofresearch.org

easily calculated (as each l−input XOR gate is assumed to be equivalent to l − 1 two-input XOR gates).

There are 2tm check bits for a code with k = m 2 and that can correct t errors, and the computation of each of them requires m−1 two input XOR gates. Therefore, the encoder requires 2tm (m − 1) two-input XOR gates. For syndrome computation, an additional XOR gate is to be needed for each parity check bit, giving a total number to 2tm twoinput XOR gates.

The proposed method requires 4tm−2 two-input XOR gates for the encoder and 8tm-4 two-input XOR gates for the syndrome computation.

3. Experimental results

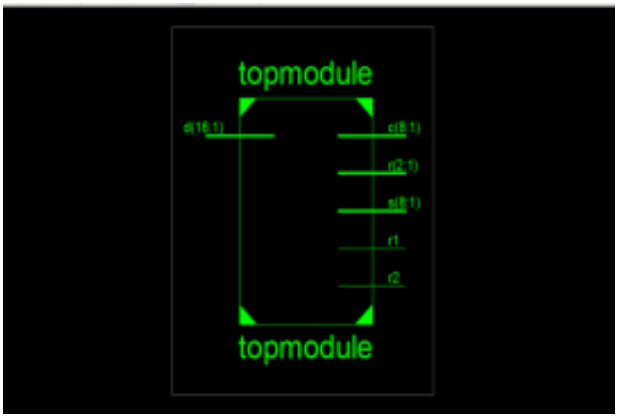

Fig 3:Block Diagram.

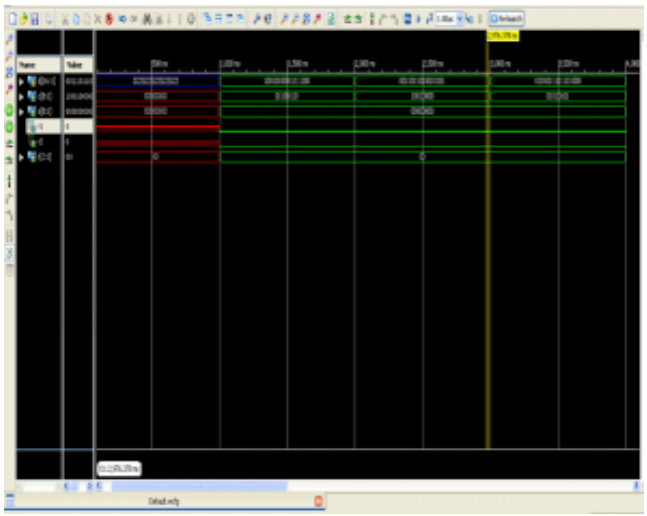

Fig 4: Simulation Output.

4. Conclusion

By the vivid conclusion, an CED method to serve OLS codes encoders and their calculation was projected. This projected method had from the properties of OLS codes for designing a parity prediction scheme which is able to professionally implement and also could detect every errors that will have effect an only circuit nodes. In this brief, a CED method for OLS codes encoders and syndrome calculation was proposed. The proposed method took advantage of the property of OLS codes to design a parity prediction scheme that could be professionally implement and detects all errors that affect an only circuit node. Here proposed scheme to detect the one or extra errors and to correct the single bit errors by using Orthogonal Latin square error correcting technique. The method is applied for different word sizes there by resulted

that the overhead will be small irrespective of how larger words we take. It is attractive even if very are being used say for instance in case of the caches the OLS code has sophisticatedly used in coming times. In the future to check error we need a momentous delay even though their brunt on access point in time can be shortened.

5. References

[1] C. L. Chen and M. Y. Hsiao, "Error-correcting codes for semiconductor memory applications: A state-of-the-art review," IBM J. Res. Develop., vol. 28, no. 2, pp. 124–134, Mar. 1984.

[2] E. Fujiwara, Code Design for Dependable Systems: Theory and Practical Application. New York: Wiley, 2006.

[3] A. Dutta and N. A. Touba, "Multiple bit upset tolerant memory using a selective cycle avoidance based SEC-DED-DAEC code," in Proc. IEEE VLSI Test Symp., May 2007, pp. 349–354.

[4] R. Naseer and J. Draper, "DEC ECC design to improve memory reliability in sub-100nm technologies," in Proc. IEEE Int. Conf. Electron., Circuits, Syst., Sep. 2008, pp. 586–589.

[5] G. C. Cardarilli, M. Ottavi, S. Pontarelli, M. Re, and A. Salsano, "Fault tolerant solid state mass memory for space applications," IEEE Trans. Aerosp. Electron. Syst., vol. 41, no. 4, pp. 1353–1372, Oct. 2005.

[6] S. Lin and D. J. Costello, Error Control Coding, 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, 2004. [7] S. Ghosh and P. D. Lincoln, "Dynamic low-density parity check codes for fault-tolerant nano-scale memory," in Proc. Found. Nanosci., 2007, pp. 1–5.

[8] H. Naeimi and A. DeHon, "Fault secure encoder and decoder for nanoMemory applications," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 17, no. 4, pp. 473–486, Apr. 2009.

[9] S. Liu, P. Reviriego, and J. A. Maestro, "Efficient majority logicfault detection with difference-set codes for memory applications,"IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 20, no. 1, pp. 148–156, Jan. 2012.

[10] P. Reviriego, M. Flanagan, S. Liu, and J. A. Maestro, "Multiple cell upsetcorrection in memories using difference set codes," IEEE Trans. Circuits Syst. I, Regular Papers, vol. 59, no. 11, pp. 2592–2599, Nov. 2012.

[11] M. Y. Hsiao, D. C. Bossen, and R. T. Chien, "Orthogonal Latin squarecodes," IBM J. Res. Develop., vol. 14, no. 4, pp. 390–394, Jul. 1970.

[12] R. Datta and N. A. Touba, "Generating burst-error correcting codes fromorthogonal Latin square codes—A graph theoretic approach," in Proc. IEEE Int. Symp. DFT, 2011, pp. 367–373.

[13] A. R. Alameldeen, Z. Chishti, C. Wilkerson, W. Wu, and S.-L.Lu,"Adaptive cache design to enable reliable low- voltage operation," IEEE Trans. Comput., vol. 60, no. 1, pp. 50–63, Jan. 2011.

[14] S. E. Lee, Y. S. Yanag, G. S. Choi, W. Wu, and R. Iyer, "Low-power,resilient interconnection with orthogonal Latin squares," IEEE DesignTestComput., vol. 28, no. 2, pp. 30– 39, Mar./Apr. 2011.

Authors Profiles

P.Suresh pursuing his M.Tech, from Vijaya Engineering College, Telangana, India. Email: suresh3221@gmail.com

AYESHA TARRANUM completed her M.Techand working as a Associate Professor, from Vijaya Engineering college, Telangana,India, Email: ayeshabad14@gmail.com

9866898754

D. Vijay Kumar completed his M.Techand working as a Associate Professor, HOD , HOD,14-years Experience from Vijaya Engineering college, Telangana,India, Email: vkumar88.d@gmail.com, 9494733835