International Journal For Advanced Research In Science & Technology
A peer reviewed international journal
www.ijarst.in
ISSN: 2457-0362

Efficient Majority Logic Fault Detection using Double Error Correction Orthogonal Latin Square Codes

KETHIREDDY KARUNASRI[1], S.BASKAR RAO[2]
[1]PG SCHOLAR.DEPT OF ECE, SIR C.V. RAMAN INSTITUTE OF TECHNOLOGY & SCIENCE, AP, INDIA
[2] ASST. PROFESSOR, DEPT OF ECE, SIR C.V. RAMAN INSTITUTE OF TECHNOLOGY & SCIENCE, AP, INDIA

ABSTRACT:
Error Correction Codes (ECCs) are commonly used to protect memories against soft errors with an impact on memory area and delay. For large memories, the area overhead is mostly due to the additional cells needed to store the parity check bits. In terms of delay, the overhead is mostly needed to detect and correct errors when the data is read from the memory. Most ECCs that can correct more than one error have a complex decoding process and so are limited in high speed memory applications. One exception is One Step Majority Logic Decodable (OS-MLD) codes for which decoding can be done in parallel at high speed. Unfortunately, there are only a few OS-MLD codes that provide a limited choice in terms of block sizes, error correction capabilities and code rate. Therefore, there is considerable interest in a novel construction of OS-MLD codes to provide additional choices for protecting memories. In this paper, a new method to construct Double Error Correction (DEC) OS-MLD codes is presented. This method is based on the use of parity check matrices in which two bits have at most two parity check equations in common; the proposed method provides codes that require a smaller number of parity check bits than existing codes like Orthogonal Latin Square (OLS) codes. The drawback of the proposed Two Bit Overlap (TBO) codes is that they require slightly more complex decoding than OLS codes. Therefore, they provide an intermediate solution between OLS and non OS-MLD codes in terms of decoding delay and number of parity check bits. The proposed TBO codes have been implemented for some block sizes and compared to both OLS and BCH codes to illustrate the trade off in delay and memory overhead. Finally, this project discusses the generalization of the proposed scheme to codes with larger error correction capabilities

1. INTRODUCTION

Error correction codes are commonly used to protect memories from so called Soft Errors, which change the logical value of memory cells without damaging the circuit. As technology scales, memory devices become larger and more powerful error correction codes are needed. To this end the use of more advanced codes has been recently proposed. These codes can correct a larger number of errors, but generally require complex decoders. To avoid a high decoding complexity, the use of one-step majority logic decodable codes was first proposed in for memory applications. One step majority logic decoding can be implemented serially with very simple circuitry but requires long decoding times. In a memory this would increase the access time. Only few classes of codes can be decoded using OS-MLD. Among those are some DS-LDPC codes, EG-LDPC codes and OLS codes.

The use of OLS codes has gained renewed interest for interconnections, memories, and caches. This is due to their modularity such that the error correction capabilities can be easily adapted to the error rate or to the mode of operation. OLS codes typically require more parity bits than other codes to correct the same number of errors. However, their modularity and the simple and low delay decoding implementation (as OLS codes are OS-MLD), offset this disadvantage in many applications. An important issue is that the encoder and decoder circuits needed to use (ECCs) can also suffer errors. When an error affects the encoder, an incorrect word may be written into the memory. An error in the

International Journal For Advanced Research In Science & Technology
A peer reviewed international journal www.ijarst.in
ISSN: 2457-0362

decoder can cause a correct word to be interpreted as erroneous or the other way around, an incorrect word to be interpreted as a correct word.

A method was recently proposed in to accelerate a serial implementation of majority logic decoding of DS-LDPC codes. The idea behind the method is to use the first iterations of majority logic decoding to detect if the word being decoded contains errors. If there are no errors, then decoding can be stopped without completing the remaining iterations, therefore greatly reducing the decoding time. And majority logic decoding can be implemented serially with simple hardware but requires a large decoding time. For memory applications this increases the memory access time. The method detects whether a word has errors in the first iterations of majority logic decoding, and when there are no errors the decoding ends without completing the rest of the iterations. Since most words in a memory will be error-free, the average decoding time is greatly reduced.

II. IMPLEMENTATION OF PROPOSED ARCHITECTURE

The use of Error Correction Codes (ECCs) for memory protection has some key differences with their use in communications. The most significant difference is that a memory word is read in parallel and the correct data is expected at the end of the clock cycle. In communication, data is typically received serially and can be decoded on a bit by bit basis. The need to complete the decoding pro-cess in one cycle for an entire word makes decoding very challenging. Traditionally, Single Error Correction (SEC) codes have been used to protect memories [3]. In such case, decoding can be done for each bit by just checking if the syndrome matches the corresponding column in the parity check matrix. However, when more than one bit needs to be corrected, such approach, known as syn-drome decoding, needs to compare the syndromes of multiple bit errors that include that bit. This leads to a large increase in decoding complexity, specially for large word sizes [10].

To overcome the limitations of syndrome based decod-ing, One Step Majority Logic Decodable (OS-MLD) codes have been proposed [11] to protect memories. Some OS-MLD codes, such as the Orthogonal Latin Square (OLS) codes were proposed decades ago for memory protection. However, in most cases, they require a large number of parity check bits [13] . Recently, Euclidean Geometry (EG) or Difference Set (DS) codes have been proposed [11],[12],[16]. These codes reduce the number of parity check bits. The significant concern with EG and DS codes is that they support only a few block sizes and error cor-rection capabilities [9]. For example, for Double Error Correction (DEC), EG codes only support (15, 7) and DS only (21, 11), where (n, k) refers to the codeword size n and the data block size k (thus the size of parity check block is n-k). Orthogonal Latin Square (OLS) codes pro-vide a wider range of choices; the parameters of the DEC OLS codes that have k equal to a power of two are shown in Table 1 for k up to 1024.

Double error correction Orthogonal Latin Square codes are built such that their parity check matrices H have the following properties [13]:

1. Each column has a size of. $4 \cdot k$
2. Each column that corresponds to a data bit has ex-actly four ones.
3. Each pair of columns only has at most a single po-sition with a one in common (one bit overlap).

Based on these properties, a simple decoding scheme can be designed. In this scheme, for each data bit a major-ity vote of the four parity check equations that are in-volved is performed. If the result is one, then the bit is in error and therefore, is thus corrected. This procedure is usually referred to as one step majority logic decoding. It will correct all errors that affect data bits in the presence of single and double errors. If a data bit is in

International Journal For Advanced Research In Science & Technology
A peer reviewed international journal
www.ijarst.in
ISSN: 2457-0362

error, an-other error can only affect one of its parity checks and thus a majority will always occur and the bit will be cor-rected. Conversely, if the bit is not in error, errors on the other two bits can only affect at most two of its parity checks, so that there will be no majority to start a correc-tion.

As an example, the parity check matrix *H* for the (32, 16) DEC OLS code is shown in Figure 1, in which the left 16 columns refer to the data bits and the other 16 columns to the parity bits. The OS-MLD decoding of the first data bit is shown in Figure 2, where each parity equation corre-sponds to one row of the parity check matrix in Figure 1. The required logic circuit is simpler than checking all possible two bit error patterns for a bit. This feature is more advantageous when the codeword size is large, because that number of double bit error patterns is pro-portional to *n*.

As discussed in the introduction, the main drawback of OLS codes is that they require a large number of parity check bits. For example, for $k = 256$, a DEC OLS code requires 64 parity check bits compared to the 18 bits re-quired by a DEC BCH code. Therefore, code constructions that support OS-MLD while requiring a smaller number of parity check bits are of significant interest. In the next section, a new construction for such codes is presented.

III. TWO BIT OVERLAP CODES

This section presents the proposed Two Bit Overlap (TBO) codes; the parameters of these codes are also ana-lyzed and established. The first subsection describes the features of the matrices that are used to construct the codes and shows how DEC OS-MLD codes can be ob-tained from them. The second subsection presents the method to construct the matrices, while the third subsec-tion summarizes the parameters of the proposed codes.

$$\left[\begin{array}{cccccccccccccccc|cccccccccccccccc}
1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0 & 1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&1&1&1&1&0&0&0&0&0&0&0&0 & 0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&1&1&1&1&0&0&0&0 & 0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&1&1&1&1 & 0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0 & 0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&0&0&1&0&0&0&1&0&0&0&1&0&0 & 0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0\\
0&0&1&0&0&0&1&0&0&0&1&0&0&0&1&0 & 0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0\\
0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&1 & 0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0\\
1&0&0&0&0&1&0&0&0&0&1&0&0&0&0&1 & 0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
0&1&0&0&1&0&0&0&0&1&0&0&1&0&0&0 & 0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0\\
0&0&1&0&0&0&0&1&1&0&0&0&0&1&0&0 & 0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0\\
0&0&0&1&1&0&0&1&0&0&1&0&0&0&0&0 & 0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0\\
1&0&0&0&0&0&1&0&1&0&0&0&0&1&0&0 & 0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0\\
0&1&0&0&0&0&1&0&0&1&0&1&0&0&0&0 & 0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0\\
0&0&1&0&1&0&0&0&0&0&0&1&0&0&0&1 & 0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0\\
0&0&0&1&0&1&0&0&1&0&0&0&0&0&0&0 & 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1
\end{array}\right]$$

Figure 1.Parity check matrix *H* of the (32, 16) DEC OLS code.

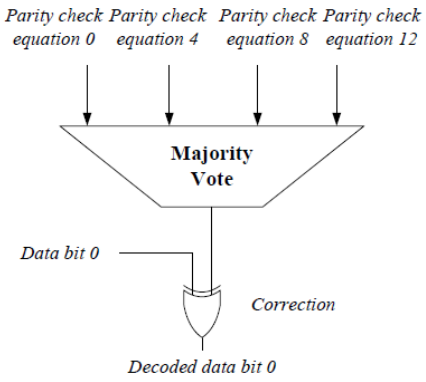

Figure 2.Majority logic decoding of data bit 0 in the (32, 16) DEC OLS code.

IV Matrix Features

As discussed in the previous section, a method for con-structing Double Error Correction (DEC) One Step Major-ity Logic Decodable (OS-MLD) codes requires to design matrices with columns such that:

1. Each column has exactly four ones.
2. Each pair of columns only has at most a position with a one in common.

Orthogonal Latin Squares with double error correction are a particular case of the above construction. Then this matrix can be used to form the parity checks such that each data bit corresponds to a column and each row to a parity check bit and each data bit participates in the pari-ty checks for which it has a one in the column.

As explained previously, the OS-MLD property is simple, because each data bit participates in four parity checks and the

International Journal For Advanced Research In Science & Technology
A peer reviewed international journal
www.ijarst.in
ISSN: 2457-0362

other bits share at most one parity check with it. Therefore, a majority of the four is obtained when the bit is in error and another bit is also in error. Con-versely, when the bit is not in error, two errors on other bits can affect at most two of the parity checks of the ini-tial bit and thus no miscorrection will occur.

Consider a construction in which the matrices are giv-en such that:

1. Each column has exactly seven ones.
2. Each pair of columns only has at most two posi-tions with a one in common.

Then, the code will be OS-MLD for DEC by taking a majority of at least five on the seven equations for the participating bit. The possible cases are as follows:

1. An error free bit and two other bits in error give a worst case of four parity check errors on the error free bit so there is no majority and no miscorrec-tion.
2. A bit in error and another bit in error cause at least five parity check errors on the erroneous bit, so it will be corrected.

A disadvantage of this construction is that the encod-ing and decoding are more complex than for a DEC OLS code. This is due to two features:

1. The matrix has a larger number of ones (seven per column instead of four) and thus, more logic is needed to compute the parity checks.
2. The majority voting is done over seven parity checks and not over four which is again more complex.

However, the decoding is still significantly simpler than for a non OS-MLD code as detailed in the evaluation section of this paper. To have advantage over existing DEC OLS codes, the proposed codes need to have a lower number of parity check bits.

V. RESULTS

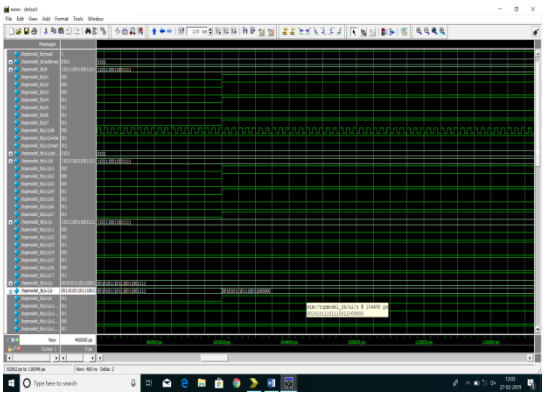

Fig 6.2 Simulation Result for Proposed Method

VI. CONCLUSION AND FUTURE SCOPE

This project has presented a new construction for Double Error Correction (DEC) One Step Majority Logic Decodable (OS-MLD) codes based on Two Bit Overlap (TBO). The codes obtained provide a trade-off between the number of parity check bits and the decoding complexity. They are significantly simpler and faster to de-code than non OS-MLD codes such as BCH codes but slightly more complex than existing DEC OS-MLD codes (such as Orthogonal Latin Square (OLS) codes). Also, they require a smaller number of parity check bits than OLS codes but more than BCH codes. Therefore, they provide additional choices to memory designers in terms of de-coding speed and memory overheads.

The work presented in this paper is being investigated and extended for future research for designing TBO codes that can correct more than two bit errors. Another area for future work is to find alternative matrix constructions that provide additional choices in term of block sizes and number of parity check bits for DEC codes.

References

[1] R. C. Baumann, “Radiation-induced soft errors in advanced semiconductor

technologies," *IEEE Trans. Device Mater. Reliab.*, vol. 5, no. 3, pp. 301–316, Sep. 2005.
[2] M. A. Bajura, Y. Boulghassoul, R. Naseer, S. DasGupta, A. F.Witulski, J. Sondeen, S. D. Stansberry, J. Draper, L. W. Massengill, and J. N. Damoulakis, "Models and algorithmic limits for an ECC-based approach to hardening sub-100-nm SRAMs," *IEEE Trans. Nucl. Sci.*, vol. 54, no. 4, pp. 935–945, Aug. 2007.
[3] R. Naseer and J. Draper, "DEC ECC design to improve memory reliability in sub-100 nm technologies," *Proc. IEEE ICECS*, pp. 586–589, 2008.
[4] S. Ghosh and P. D. Lincoln, "Dynamic low-density parity check codes for fault tolerant nano-scale memory," presented at the Foundations Nanosci. (FNANO), Snowbird, Utah, 2007.
[5] S. Ghosh and P. D. Lincoln, "Low-density parity check codes for error correction in nanoscale memory," SRI Computer Science Lab., Menlo Park, CA, Tech. Rep. CSL-0703, 2007.
[6] N. Kanekawa, E. H. Ibe, T. Suga and Y. Uematsu, "Dependability in electronic systems: mitigation of hardware failures, soft errors, and elec-tro-magnetic disturbances," (Springer Verlag, New York, USA, 2010)
[7] S. Dolev, Y.A. Haviv, "Self-stabilizing microprocessor: analyzing and overcoming soft errors," IEEE Transactions on Computers, vol.55, no.2, pp.385-399, Apr. 2006.
[8] 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.
[9] S.C. Krishnan, R. Panigrahy, S. Parthasarathy, "Error-correcting codes for ternary content addressable memories," IEEE Transactions on Computers, vol.58, no.2, pp.275-279, Feb. 2009.
[10] E. Fujiwara, "Code design for dependable systems: theory and practical application," John Wiley & Sons, Inc., Hoboken, New Jersey, 2006.
[11] J. Li, P. Reviriego, L. Xiao and C. Argyrides, "Extending 3-bit Burst Error Correction Codes with Quadruple Adjacent Error Correction", IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 26, no. 2, pp. 221-229, Feb. 2018.