INTERNATIONAL JOURNAL OF TECHNO-ENGINEERING
IJTE

ISSN: 2057-5688

Multi Bit Error Detection and Correction using Decimal Matrix Code

[1]T SRILAKSHMI, [2]VENKATARAO T

[1]M. Tech Student. Dept.of ECE, Malineni Lakshmaiah Women's engineering college, AP

[2]Assistant professor, Dept.of ECE, Malineni Lakshmaiah Women's engineering college, AP

Abstract: *The CMOS technology scales down to nano scale and memories are combined with an increasing number of electronic systems; the error rate in memory cells is rapidly increasing. Especially error occurs when memories operate in space satellites due to ionizing effects of atmospheric neutron, alpha-particle, and cosmic rays. SRAM Based FPGAs are used mostly in Space satellites, which are sensitive to radiation. Although single bit upset is a major concern about memory reliability, Multiple Cell Upsets have become a serious reliability concern. In order to make memory cells as fault-tolerant as possible, some error correction codes have been widely used to protect memories. The suggested method has equal error correction capability with smaller count of parity bits as compared to other current techniques. Total parity bits have been concentrated by 30% and area, power and delay time has been concentrated by 1.12%, 33.59%, 55.46% correspondingly. These parameters make the proposed technique an efficient and productive one for protecting memories. This technique can be used in applications which have very strict constraints of parity bits and speed.*

Keywords: *Multiple Cells Upset (MCU), Error Correction Codes (ECCs), Matrix Based Codes.*

I. INTRODUCTION

Error correction codes are widely used to secure and this so-called Soft Error Memory by destroying the circuit that affects the logical significance of memory cells. With technology scales, memory devices develop larger and error correction codes are becoming more efficient. To this end it was recently recommended to use more sophisticated codes. Ses codes may remedy further mistakes, but typically need complicated decoders. The use of one-stage, primarily decodable logic keys was first suggested for memory applications to prevent a large decoding

ISSN: 2057-5688

difficulty. With a rather simple circuitry, however long decoding times are required, one-step major decoding can be implemented. In such a memory, the access time will be expanded. There could only be several coding groups decoded by OS-MLD. These comprise a range of DS-LDPC, EG-LDPC and OLS codes. The usage of OLS codes has become extremely relevant with respect to interconnections, memories and caches. Because of its modularity, their error correcting capabilities will quickly be tailored to the error rate or operating mode. In order to fix the same number of errors OLS codes usually need more parity bits instead of other codes. That modularity as well as the fast and quick decoding method (since OS-MLD is the OLS code) can compensate in many applications for this disadvantage. A big concern is the probability of errors with the required encoder and decoder circuits (ECCs). An incorrect term can be entered throughout the memory whenever a mistake influences the encoder. An error of the decoder will contribute to the misunderstanding of a proper word or even the misunderstanding of a wrong word as the right word. A strategy for speeding the serial execution of the encoding of DS-LDPC codes for maximum logic were recently suggested. The goal of the process is to use the first big logic decoding increments to detect whether the decoded word requires explanation. When no errors occur, decoding may be prevented without the remaining iterations, thereby decreasing decoding period considerably. More logic decoding with simple hardware can be performed serially, and that takes a long decoding duration. This raises the memory access time for device applications. The procedure detects whether a word has mistaken during the first major logical decoding iterations and where no errors arise the decoding stops without the remainder of both the iterations done. As most of the terms in a memory are error-free, the mean time for encoding is decreased dramatically.

II. LITERATURE SURVEY

QiujuDiao, Ying Yu Tai, proposed on “Trapping Set Structure of LDPC Codes on Finite Geometries” LDPC codes are the most promising coding tool for achieving the Shannon capabilities of different networks at present. They do well with iterative, belief propagation-based decoding. And most LDPC codes have such a general extreme vulnerability, referred to as error level with iterative

ISSN: 2057-5688

decoding. Error Floor might prohibit applications that need very low error rates from obtaining LDPC codes. The error level of both the AWGN channel is largely attributed to an unintended layout, known as a trapping package, in the Tanner code graph. This geometric methodology is used to evaluate the trapping set configuration for LDPC codes centred on finite geometer, named LDPC finite geometry (FG). Trapping structures are shown by subgeometries of the geometry over which the code was designed throughout the Tanner Graph of even an FG – LDPC code. This geometrical description can be used to extract and evaluate boundaries and configurations of FG LDPC code trapped sets. Juntan Zhang, S. Yedidia, suggested the EG LDPC codes low latency encoding method We define simulated annealing codes to Euclidean Geometrie dependent low-density parity-check codes, which are ideal for use in applications that need very rapid decoders. The decoders are based on mixed and repeated Bit-flitting iterative (BF) & quantized BF structures. That decoders proposed converge faster than regular BF decoders, offering a better efficiency. We present simulations to demonstrate how these decoders function against uncertainty. In certain instances, we can prove that no major mistake occurs by important sampling.

"On the basis of RCD SPC codes for arrays and their equivalence to codes rendered from Euclidia geometries and partial BIBDs" suggested by Amitkumar Mahadevan and Joel Morris. We present a technique for constructing a Gallager family for low density parity-check codes (LDPC), centered on η ×η sequence in which the η become prime with μm2 diagonal bundles reflecting equations for parity-check. These γ= 3 codes are referred to as the single-parity search (SPC) rowcolumn-diagonal (RCD). The equivalence is provided for the Gallager LDPC codes for diagonal bundles, in which is primary, as well as the Euclidean geometry (EG) codes in which Gallager LDPC codes were constructed, with parameters m=2, s=1, and LDPC codes constructed throughout the (˙) grating of both the gallager LDPCs, in square arrays. They define each building technique, demonstrating that the three building techniques are similar to instances. "Radiation-induced soft error in advanced semiconductor technologies" suggested by Robert C. Baumann

ISSN: 2057-5688

The formerly ephemeral soft error triggered by radiation is becoming a significant challenge for sophisticated electronic commercial substances and systems. Left without problems, soft errors have the potential to cause all the other reliability processes to combine the maximum failure rate. This article describes briefly some types to soft error component failures, the 3 predominant radiation mechanisms which produce soft error in terrestrial applications as well as how the set of radiation-induced charge generates these soft mistakes. Applications also are certainly to be mitigated in soft errors, that soft sensitivity as just a result of both the technology scaling with different memory & logic components would then be addressed. Saad Bin Qaisar suggested "Low Density Parity Verification Codes on Finite and Incomplete Underground Architecture.

Several techniques are used to mitigate upsets in memories.

Hamming Codes

Hamming code is a form of linear error correcting code that can detect up to two-bit error or correct one-bit errors without detection of uncorrected errors. By contrast, the simple parity code cannot correct errors but can only detect odd number of error bits. The decoder can detect and correct a single error and at the same time detect a double error. If the decoder does not attempt to correct errors it can detect up to three errors. Software-based Hamming codes can be used to improve the reliability of the most important sections of memory. Memory is used to store information of various types. Some types of information require strong protection against errors, while other types do not.

Matrix Codes

Matrix Code is based on combining Hamming codes and Parity codes in a matrix format so the detection and correction of multiple faults is achieved. The protection bits are used in a matrix format. The n-bit code word is divided into k1 sub words of width k2. A (k1, k2) matrix is formed where and represent the numbers of rows and columns, respectively. For each of the k1 rows, the check bits are added for single error correction/double error detection. Another k2 bits are added as vertical parity bits.

Drawbacks of Existing System

The main drawback of the existing system is that the error correction capability is less.

ISSN: 2057-5688

The Hamming codes are capable of detecting and correcting only single bit error. It has higher area overhead. The Matrix Codes are capable of detecting and correcting only two errors.

III. PROPOSED SYSTEM

The k-bit data in the proposed technique is arranged in the form of m x n matrix, where rows are represented by m and coloumns are by n in the matrix (i.e.., k = m x n). Unlike the work Adjacent Error Correction using Matrix Based codes, the data is arranged in a matrix row by row. Four Vertical Hamming bits and ten parity sharing bits are calculated from all the data bits. In our work, for the bits M8 and M9 are differently calculated using the bits which are present in the DNA (Deoxyribo Nucleic Acid) shaped curve in the matrix as shown in the Fig.3. And those bits are further divided into two parts which will be explained further. To explain our work, consider a 32-bit data as an example and the data bits (D0 – D31) are arranged in the matrix form with m = 8 and n = 4. The extended Hamming parity bits (V0 – V3) are helpful for identifying the errors in the column bits are calculated as given in equation 1. Equation 1 explains the way in which V0 is calculated, in the same way V1 - V3 is also calculated. The horizontal parity sharing bits (M0 - M9) are calculated as given in Equation 2 to 5. Equation 2 explains the calculation of M2, M4, M6 in the same way as M0 and Equation 3 explains the calculation of M3, M5, M7 in the same way as M1. M8 (coloured as Blue as shown in Fig.3) and M9(coloured as Yellow as shown in Fig.3) are calculated in a unique way which are represented in Equations 4 and 5. The Hamming Equation are given in Equations 1-5. Therefore, all the Equations are formed by using basic operation XOR. Compared to the work Adjacent Error Correction using Matrix Based codes,in this technique total number of redundant bits calculated has been reduced and use of imaginary bits are eliminated. Syndrome bits are calculated for extended.

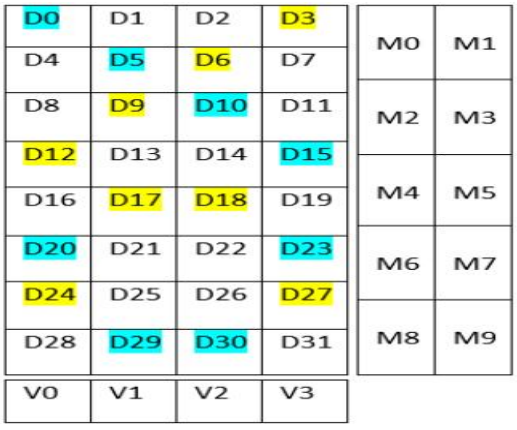

Fig.1 Logic diagram for Proposed Technique

ISSN: 2057-5688

Hamming parity bits (V). If any bit is affected by error, then parity bit of that particular column will help for the identification of that error bit

$$V0 = D0 \oplus D4 \oplus D8 \oplus D12 \oplus D16 \oplus D20 \oplus D24 \oplus D28$$

$$M0 = D0 \oplus D1 \oplus D2 \oplus D16 \oplus D17 \oplus D18$$

$$M1 = D1 \oplus D3 \oplus D17 \oplus D19$$

$$M8 = D0 \oplus D5 \oplus D10 \oplus D15 \oplus D20 \oplus D23 \oplus D29 \oplus D30$$

$$M9 = D3 \oplus D6 \oplus D9 \oplus D12 \oplus D17 \oplus D18 \oplus D24 \oplus D27$$

Then the horizontal parity sharing bits (M0 - M9) are calculated. If any bit is affected by error, then the corre-sponding parity bit of that particular row will help for the detection of that error bit. Finally, the error bit is corrected. Let's understand the process using an example. If D1 bit is affected by the error, then the syndrome (V0 - V3) will be equal to 0100, Which represents an error in 2nd column. Then horizontal parity sharing bits (M0 - M9) will help to detect that particular row of error (1st row) . Then finally the bit with error is detected and corrected. It is important to note that, as there are no separate parity bits for error detection in the parity bits itself, so in case there are errors in both data bits and parity bits which are affected by MCUs, data bits cannot be detected.

IV. EVALUATION

When a 32-bit state is provided to the encoder, it provides a 46-bit output, i.e., the statistics bits and the parity bits. These forty-six bits are then given as input to the decoder, to check for errors in the statistical bits. In this section we describe possible types of MCUs that are examples of memories and interpretation techniques for validating the information.

One Bit Error

D0	D1	D2	D3	M0	M1
D4	D5	D6	D7		
D8	D9	D10	D11	M2	M3
D12	D13	D14	D15		
D16	D17	D18	D19	M4	M5
D20	D21	D22	D23	M6	M7
D24	D25	D26	D27		
D28	D29	D30	D31	M8	M9
V0	V1	V2	V3		

Fig.2 1-bit error

For example, taking D0 as a corrupted bit as shown in Fig.4, then errors will be reflected in M0, V0 and M8. Therefore, decoder will detect D0 bit and data bit is flipped for restoration.

Two Bit Error

ISSN: 2057-5688

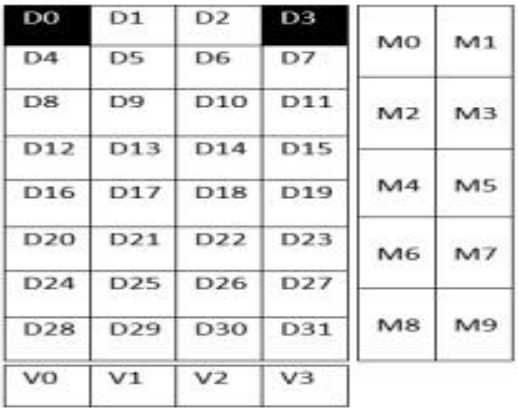

Fig.3 2-bit error

Assuming D0 and D3 as corrupted bits as shown in Fig.5, then errors will be reflected in M0, V0, M1, V3, M9 and M8. Therefore, decoder will detect D0 and D3 bits and data bits are flipped for restoration.

Three Bit Error

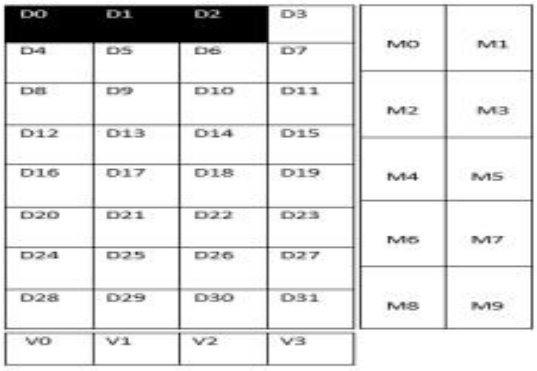

Fig.4 3-bit error

Assuming D0, D1 and D2 as corrupted bits as shown in Fig.6, then errors will be reflected in M0, M1, V0, V1, V2 and M8. Therefore, decoder will detect D0, D1 and D2 bits and data bits are flipped for restoration.

Four Bit Error

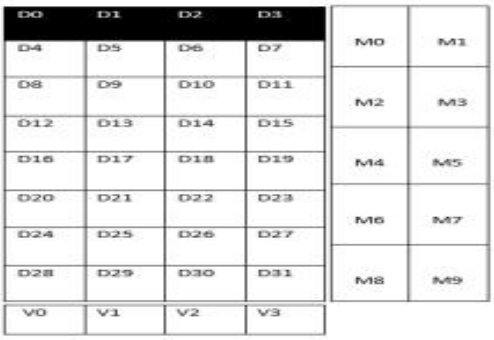

Fig.5 4-bit error

Assuming D0, D1, D2 and D3 as corrupted bits as shown in Fig.7, then errors will be reflected in M0, V0, V1, V2, V3, M9 and M8. Therefore, decoder will detect D0, D1, D2 and D3 bits and data bits are flipped for restoration. This is the main property of our proposed technique, ”Adjacent Horizontal four bit error correction.

V. RESULTS

The proposed code and the hidden parity code are coded using Verilog HDL and simulated in Xilinx Vivado. Both the codes were implemented in Vertex-6 using Xilinx ISE. It can be observed in Table I that very less delay, area and power has been achieved compared to the Adjacent Error Correction using Matrix Based codes. The number of bits stored in the memory

ISSN: 2057-5688

has been reduced from 52-bits to 46-bits and decoding process is done without using imaginary bits, which are been used in the work Adjacent Error Correction using Matrix Based codes. Power, area and delay consumption are reduced by 1.26%, 5.55% and 21.17% respectively as shown in Table II.

Table.1 Comparison of Area, Power and Delay

Codec	Area	Total Power(μW)	Delay(ns)
Matrix based Codec	72	3.649	2.153
Proposed Codec	68	3.603	1.697

Table.2 Percentage Reduction of Area, Power and Delay

Technique	Area %	Power%	Delay %
Encoder	15%	0%	26.96%
Decoder	1.92%	1.95%	57.48%
Proposed Codec	5.55%	1.26%	21.18%

VI. CONCLUSION

Error correction codes are used to improve the memory protection and make the memory fault free. The various ECC are used to detect the occurrence of error and also correct the detected ones. However, the error detection capability and the overheads vary based on the codes used. The proposed technique can easily detect and correct the adjacent errors using only 14 parity bits. This technique has achieved a significant reduction of power, area and delay. These results have been calculated for Vertex-6, showing that the power, area and time delay consumption are reduced at most by 1.26%, 5.55% and 21.17% respectively. Based on results, the proposed code suits better for the memory system which has strict requirements of area and time delay. In this method, a maximum of four-bit adjacent errors can be corrected and every single bit errors and few two-bit errors effectively. Coming to future scope, presented technique can be extended to correct the data bits when there are even number of errors in column bits. And a technique that can correct errors even in parity bits.

REFERENCES

[1] D. Radaelli, S. Wong, 2005, “Investigation of multibit upsets in a 150 nm technology SRAM device,” IEEE Trans, pp. 2433–2437.

[2] R.C. Baumann, “Radiation-induced soft errors in advanced semiconductor technologies”, IEEE Trans Dev Mater Reliab., Vol.5, No.3, pp. 301-316, Sep. 2005.

[3] K. Sai Karan, N. Srikanth, Sonali Agrawal, ”Robust Code for MBU Correction Till 5-Bit Error,” International

ISSN: 2057-5688

Conference on Communication and Electronics Systems,2019.

[4] 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.

[5] Naeimi H, Dehon A. "Fault secure encoder and decoder for nanomemory applications". IEEE Trans Very Large Scale Integr (VLSI) Syst , Vol. 17, No. 4 pp. 473-486, 2009.

[6] H. Naeimi and A. DeHon, "Fault secure encoder and decoder for memory applications," in Proc. IEEE Int. Symp. Defect Fault Toler.VLSI Syst., 2007, pp. 409–417.

[7] B. Vasic and S. K. Chilappagari, "An information theoretical framework for analysis and design of nanoscale fault-tolerant memories based on low-density parity-check codes," IEEE Trans. Circuits Syst.I, Reg. Papers, vol. 54, no. 11, pp. 2438–2446, Nov. 2007.

[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. Lin and D. J. Costello, Error Control Coding, 2nd ed. Englewood Cliffs, NJ: PrenticeHall, 2004.

[10] S. Liu, P. Reviriego, and J. Maestro, "Efficient majority logic fault detection with difference-set codes for memory applications," IEEETrans. VerLarge Scale Integr. (VLSI) Syst., vol. 20, no. 1, pp. 148–156, Jan. 2012.