Error Detection and Correction using Decimal Matrix Algorithm

P.Pushpalatha[1], Kundeti Harshavardhan[2]

[1]Assistant Professor, JNTUK, Andhra Pradesh
[2]Student, M.Tech, JNTUK, Andhra Pradesh

Abstract - *Decimal matrix code (DMC) is grounded on dividing the information bits into symbols and is suggested to improve memory reliability with lesser delay overhead. A concern when using memories is that they can be affected by soft errors that corrupt the stored bits. The DMC technique utilizes a decimal algorithm to gain the maximum error detection capability and furthermore, the encoder-reuse technique (ERT) is suggested to reduce the area overhead of extra circuits without intervening in the entire encoding and decoding processes.*

Key Words: Decimal matrix code, Encoder reuse technique, Soft errors.

1. INTRODUCTION

Transient multiple cell upsets (MCUs) have become major issues in the reliability of memories open to the radiation environments. To prevent these MCUs from triggering data corruption, more difficult error correction codes (ECCs) are broadly used to protect memory, but the key problem is that they would need higher delay overhead. In modern times matrix codes (MCs) grounded on hamming codes have been projected for memory protection. The key concern is that it has double error correction codes and the error correction abilities are not increased in all cases.

1.1 Block diagram of DMC

First of all, during the encoding (write) procedure, information bits D are given as input to the DMC encoder, and then the horizontal redundant bits H and vertical redundant bits V are acquired from the DMC encoder. When the encoding procedure is done, the received DMC codeword is stored in the memory. If MCUs take place in the memory, the errors can be altered in the decoding (read) procedure. The advantage of the decimal algorithm has helped the DMC to attain higher fault-tolerant ability with lower performance overheads. In the fault-tolerant memory, the ERT method is proposed to decrease the area overhead of extra circuits and will be discussed in the following sections.

The proposed schematic of DMC is depicted in the below figure

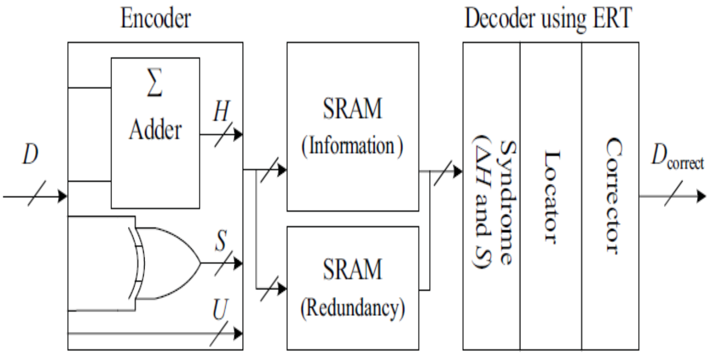


Fig -1: Schematic of DMC

2. 32-Bit DMC Encoder

In the suggested DMC, initially, the information bits are divided into symbols and arranged into a matrix form, i.e., the N-bit data is divided into k symbols of m bits (N = k × m), these symbols are organized in a k1 × k2 2-D matrix (k = k1 × k2, where the values of k1 represent the number of rows and k2 represents the number of columns in the logical matrix respectively). Second, the horizontal redundant bits H are generated by performing decimal integer addition of selected symbols per row. Here, each symbol block is considered as a decimal integer. Third, the vertical redundant bits V are captured by binary operation among the bits per column. It should be distinguished that both dividing into symbols and arranging them in matrix form are implemented in logical rather than physical. Therefore, the proposed DMC doesn't need any change in the physical structure of the memory.

To describe the proposed DMC technique, we took a 32-bit word as an example. The cells from D0 to D31 are data bits. This 32-bit data word has been divided into eight symbols of 4-bit each. k1 = 2 and k2 = 4 have been chosen concurrently. H0–H19 are horizontal bits; V0-V15 are vertical bits. Nonetheless, it should be noted that the maximum error correction ability and the number of redundant bits are different when the values for k and m are altered. So, k and m should be wisely rearranged to increase the error correction ability and reduce the number of redundant bits.

For example, in this case, when k = 2×2 and m = 8, only a 1-bit error can be corrected and the number of redundant bits is 80. When k = 4 × 4 and m = 2, 3-bit errors can be corrected and the number of redundant bits is decreased to 32. But, when k = 2 × 4 and m = 4, the maximum error correction ability has become 5 bits and the number of redundant bits has come down to 72. In this paper the effort is to enhance the reliability of memory, the error correction capability is first considered, so k = 2 × 8 and m = 4 are utilized to construct DMC.

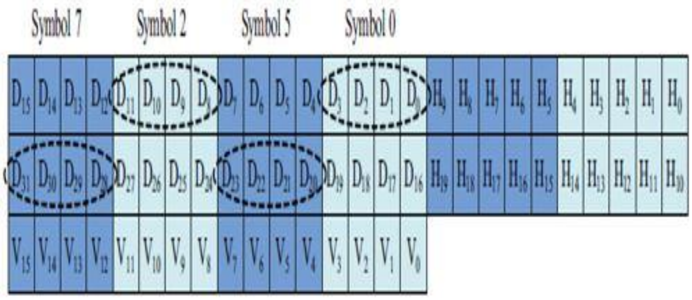

Symbol 7
Symbol 2
Symbol 5
Symbol 0
D15 D14 D13 D12 D11 D10 D9 D8 D7 D6 D5 D4 D3 D2 D1 D0 H9 H8 H7 H6 H5 H4 H3 H2 H1 H0
D31 D30 D29 D28 D27 D26 D25 D24 D23 D22 D21 D20 D19 D18 D17 D16 H19 H18 H17 H16 H15 H14 H13 H12 H11 H10
V15 V14 V13 V12 V11 V10 V9 V8 V7 V6 V5 V4 V3 V2 V1 V0

Fig -2: 32 bit word divided as 8 symbols.

Horizontal redundant bits are calculated as follows

$H_9 H_8 H_7 H_6 H_5 = D_7 D_6 D_5 D_4 + D_{15} D_{14} D_{13} D_{12}$

$H_4 H_3 H_2 H_1 H_0 = D_3 D_2 D_1 D_0 + D_{11} D_{10} D_9 D_8$

In a similar way for horizontal redundant bits $H_{14} H_{13} H_{12} H_{11} H_{10}$ and $H_{19} H_{18} H_{17} H_{16} H_{15}$. Here the "+" symbol denotes decimal integer addition. Vertical redundant bits are calculated as

$V_0 = D_0 \oplus D_{16}$

$V_1 = D_1 \oplus D_{17}$

Likewise for the rest vertical redundant bits. The encoding can be performed by decimal and binary addition operations. The encoder that computes the redundant bits using multi-bit adders and XOR gates is shown in Fig-3. In this figure, $H_{19}–H_0$ are horizontal redundant bits, $V_{15}–V_0$ are vertical redundant bits, and the remaining bits $U_{31}–U_0$ are the information bits that are directly copied from D_{31} to D_0. The enable signal En will be discussed in the next section.

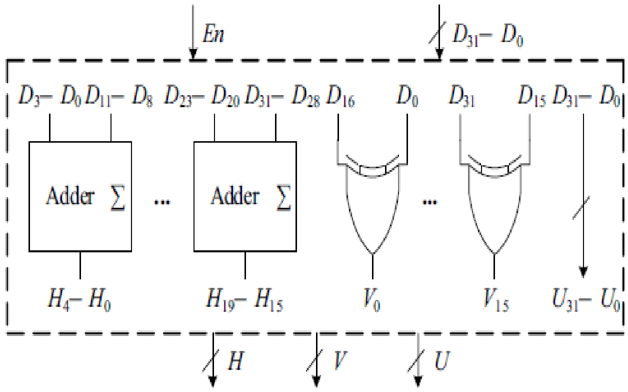


Fig-3: 32-bit DMC encoder structure using multi bit adders and XOR gates.

3. 32-Bit DMC Decoder

To acquire the whole data that is correct, performing the decoding procedure is a must. In this procedure, at first, the obtained redundant bits $H_4H_3H_2H_1H_0$' and V_0'-V_3' formed from the received information bits D'. Then, the horizontal syndrome bits $\Delta H_4H_3H_2H_1H_0$ and the vertical syndrome bits $S_3–S_0$ are calculated in the following way

$\Delta H_4H_3H_2H_1H_0 = H_4H_3H_2H_1H_0' – H_4H_3H_2H_1H_0$

$S_0 = V_0' \wedge V_0$

In the same way, the rest of the vertical syndrome bits are calculated, and here the "–" symbol denotes the decimal integer subtraction. After the calculation of $\Delta H_4H_3H_2H_1H_0$ and $S_3–S_0$ if the results are zero, the acknowledged code word has original information bits in symbol 0 and it can confirm that there are no errors occurred. If in case $\Delta H_4H_3H_2H_1H_0$ and $S_3 – S_0$ results are not equal to zero, then the induced errors which are four in this example are detected and located in symbol 0, and then to rectify these errors the follow the statement given below

$D_{0correct} = D_0 \wedge S_0$

The following figure shows the 32-bit DMC decoder. The DMC decoder is depicted in Fig-4. The whole diagram comprises of the following sub-blocks and each one of them executes a specific task in the decoding procedure. It consists of three main components syndrome calculator, error locator, and error corrector. It can be seen from this figure that the redundant bits must be recalculated from the received information bits D' and matched to the original set of redundant bits in order to get the syndrome bits ΔH and S. Then error locator uses ΔH and S to detect and locate in which bits errors have occurred. At last, in the error corrector, these errors can be corrected by inverting the values of error bits.

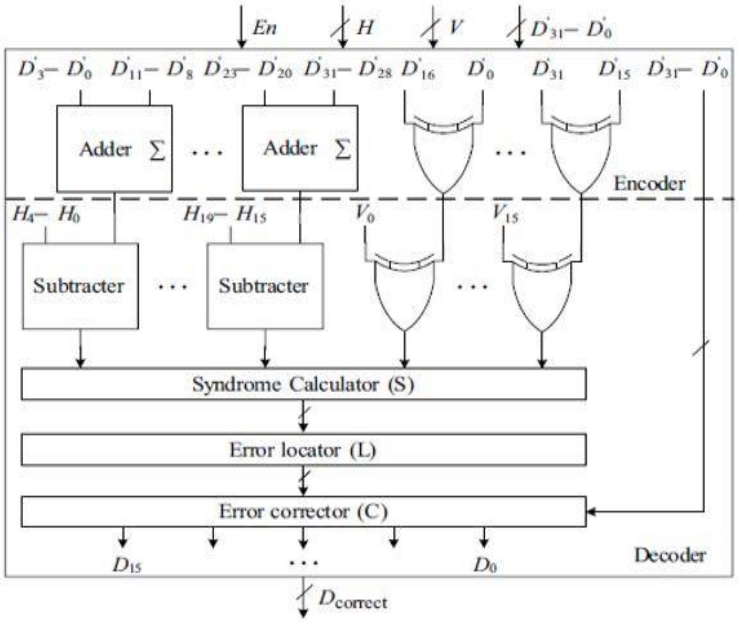
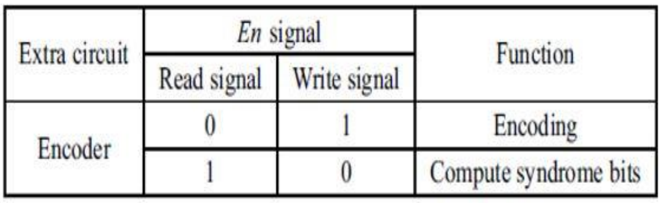


Fig-4: 32-bit DMC decoder



Extra circuit	En signal		Function
	Read signal	Write signal	
Encoder	0	1	Encoding
	1	0	Compute syndrome bits

Fig-5: Encoder reuse technique

In this outline, the circuit area of DMC is diminished by reusing its encoder. This is called the ERT. The ERT can reduce the area overhead of DMC without troubling the entire encoding and decoding processes.

4. Algorithm

1. Initially the clock is given which is used for transmitting and receiving the data within a single clock of positive edge.
2. The reset condition is defined for on/off conditions.
3. This approach mainly consists of 2 main blocks
i) Encoder ii) Decoder
4. Input the 32-bit data to the DMC Encoder.
5. Encoder calculates horizontal and vertical redundant bits.
6. Encoder adds the horizontal and vertical redundant bits to the actual data thus encoding of data can be performed.
7. Decoder checks whether the error induced in the received information by calculating redundant bits for the received information.
8. Decoder having three main blocks
i) Syndrome calculator ii) Error locator iii) Error corrector
9. Syndrome calculator computes the syndrome bits and checks these syndrome bits are zero or not. Non-zero syndrome bits represent errors in the received information.
10. Errors detected by Error locator and corrected by Error corrector using modulo-2 arithmetic operations between syndrome bits and actual data.
11. Decoded data will be given as output by the decoder.

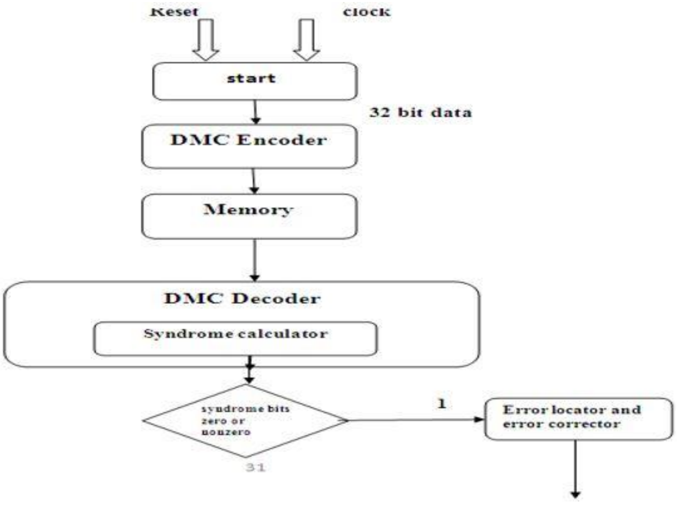
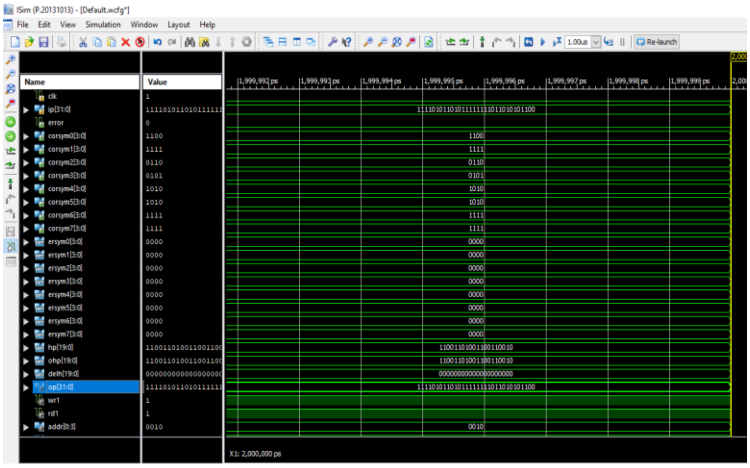


Fig-6: Flowchart

5. Results and Discussions

In this section, the DMC is executed in VHDL and simulated with Isim. The device utilization, power and critical path delay are attained using Xilinx software.



Name	Value
clk	1
ip[31:0]	1111010110101111111
error	0
corsym0[3:0]	1100
corsym1[3:0]	1111
corsym2[3:0]	0110
corsym3[3:0]	0101
corsym4[3:0]	1010
corsym5[3:0]	1010
corsym6[3:0]	1111
corsym7[3:0]	1111
ersym0[3:0]	0000
ersym1[3:0]	0000
ersym2[3:0]	0000
ersym3[3:0]	0000
ersym4[3:0]	0000
ersym5[3:0]	0000
ersym6[3:0]	0000
ersym7[3:0]	0000
hp[19:0]	110011010011001100
ohp[19:0]	110011010011001100
delh[19:0]	000000000000000000
op[31:0]	1111010110101111111
wr1	1
rd1	1
addr[0:3]	0010

Fig-7: Timing waveform of DMC

I. Overhead Analysis:

S.no	ECC codes	Device utilization (Gate count)	Delay(ns)	Power (mW)
1	Decimal Matrix code(proposed)	980	0.780	67.8
2	Parity protected (Existing)	1228	1.117	90

Table 1. Area, Delay and Power analysis

The results of analysis are shown in Table 1. These results express how our proposed technique delivers single and double-error correction, but can also offer effective tolerance capabilities compared to large MCUs that exceed the performance of other codes.

6. CONCLUSIONS

To prevent MCUs from causing data corruption, more complex error correction codes (ECCs) are widely used to protect memory, but the main problem is that they would require higher delay overhead. In recent times matrix codes (MCs) based on hamming codes have been suggested for memory protection. The proposed technique significantly reduces the resources needed to protect the memories and can be an interesting option for designs on which reliability is a concern but resources are limited. The study that has been carried out in this paper can be extended to many other potential fields. The major possibility is to develop an error correcting system that can provide a better performance, with less delay overhead, lower power requirements, and less area consumption. The study can be carried out by pipelining the existing codes into an efficient form so that the delay overhead is reduced. And by changing the adders and other elements used in realization, the area can be decreased, by the proper implementation of the abovementioned two ideas for the power consideration can also be substantially reduced.

REFERENCES

[1] PedroReviriego, Salvatore pontarelli and Anees ullah, "Error Detection and Correction in SRAM Emulated TCAMs", IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol.27, Feb 2019

[2]. Jing Guo, Liyi Xiao, Zhigang Mao, and Qiang Zhao ,Enhanced Memory Reliability against Multiple Cell Upsets Using Decimal Matrix Code, IEEE Transactions On Very Large Scale Integration (VLSI) Systems, March 2014.

[3] Ahilan.A and Deepa.P ,Modified Decimal Matrix Codes in FPGA Configuration Memory for Multiple Bit Upsets, IEEE January 2015.

[4] Pedro Reviriego, Salvatore Pontarelli, Juan Antonio Maestro, and Marco Ottavi, A Method to Construct Low Delay Single Error Correction Codes for Protecting Data Bits Only, IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems., vol. 32, no. 3, pp. 479 - 483, March 2013.

[5] Dejan Georgiev, "Low Power Concept for Content Addressable Memory (CAM) Chip Design," International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering, Vol.2, Issue 7, July 2013.

[6] S. J. Ruan, C.Y. Wu, J. Y. Hsieh, "Low power design of pre-computation based content-addressable memory," IEEE Transactions Very Large Scale Integration (VLSI) Systems, Vol. 16, No. 3, p.p. 331-335, March 2008.

[7] K.Pagiamtzis and A.Sheikholeslami, "ContentAddressable Memory (CAM) Circuits and Architectures: A Tutorial and Survey," IEEE Journal of Solid-State Circuits, Vol. 41, p.p. 712-727, March 2006.

[8] K.Pagiamtzis and A.Sheikholeslami, "A low-power content-addressable memory (CAM) using pipelined hierarchical search scheme," IEEE J. Solid-State Circuits, Vol. 39, No. 9, p.p. 1512–1519, Sep. 2004.

[9] C.A.Zukowski and S.-Y. Wang, "Use of selective precharge for low power content-addressable memories," in Proc. IEEE Int. Symp. Circuits Syst. (ISCAS), Vol. 3, p.p. 1788–1791, 1997.

[10] V. Gherman, S. Evain, N. Seymour, and Y. Bonhomme, Generalized parity-check matrices for SEC-DED codes with fixed parity, in Proc. IEEE On-Line Testing Symp., Jul. 2011, pp. 198–20.