INTERNATIONAL JOURNAL OF TECHNO-ENGINEERING IJTE

ISSN: 2057-5688

Diagonal Hamming Based Multi-Bit Error Detection and Correction Techniques for Memories

D.SAI PRIYANKA[1], N.NAGA MALLIKARJUNA[2], B.UMAKANTH[3]
[1]PG Student, Dept of ECE, SITS, Kadapa, AP, India.
[2]Associate Professor, Dept of ECE, SITS, Kadapa, AP, India.
[3]Assistant Professor, Dept of ECE, SV College of Engineering, Kadapa, AP, India.

Abstract—

In this paper, an advanced error correction 2-dimensional code based on divide-symbol is proposed to weaken radiation-induced MCUs in memory for space applications. For encoding data bits, diagonal bits, parity bits and check bits were analyzed by XOR operation. To recover the data, again XOR operation was performed between the encoded bits and the recalculated encoded bits. After analyzing, verification, selection and correction process takes place. Temporary errors which are classified under soft errors are created because of fluctuations in the voltage or external radiations. These errors are very common and obvious in memories. In this paper, multi-bit error detection and correction technique is proposed to identify errors 2 bits error for one row. The proposed scheme is simulated and synthesized using Xilinx implemented in Verilog HDL.

I. INTRODUCTION

Binary information is stored in a storage space called memory. This binary data is stored within metal-oxide semiconductor memory cells on a silicon integrated circuit memory chip. Memory cell is a combination of transistors and capacitors where capacitor charging is considered as 1 and discharging considered as 0 and this can store only one bit. Errors which are temporary or permanent are created in the memory cells and need to be eliminated. Single bit error correction is most commonly used technique which is capable of correcting up to one bit. Since technology is increasing rapidly, there are more probabilities of getting multiple errors [1]. Use of Diagonal Hamming method leads to efficient correction of errors in the memories. Memory was divided as SRAM, DRAM, ROM, PROM, EPROM, EEPROM and flash memory [2]. Main advantages of semiconductor memory is easy to use, less in cost, and have high bits per square micrometers. Temporary errors are called transient errors which are caused because of fluctuations in potential level. Permanent errors are caused because of defects during manufacturing process or large amount of radiations [3].

For detection and correction of these soft errors, various methods have been proposed [4]. In this paper, memory bits which are affected by errors are recognized and rectified using Diagonal Hamming based multi-bit error detection and correction technique. The aim of this method is to detect effectively up to 8 bit errors and correcting them. In this project, Section-II gives overview of other coding techniques. The advantages and disadvantages of different coding techniques are discussed. Binary digits is stored in a space called memory. Errors occur in memory due to voltage fluctuations, manufacturing process or due to very high radiations. There are many coding techniques for error correction and detection. These techniques can correct from single to multiple bit errors. Coding methods like HVD, HVDD, HVPDH, DMC, MDMC, 3D parity check with Hamming are used to correct bits in memory. The 3 Dimensional parity check is used for checking and correcting the errors in message bits and hamming is used for correction of parity bits [3]. In HVPDH method, numbers of parity bits are reduced and hence reliability is increased [1]. HVD method [5] is used for correction of soft error bits and the power consumption is less in this method. In HVD method, the parity code uses four different directions in block of data.

Problem Statement Coding theory is interested in providing a reliability and trustiness in each communication system over a noisy channel. In the more communication systems, the error correction codes are used to find and correct the possible bit changing [2], for example, in wireless phones and etc. In any environment, Environmental interference, physical fault noise, electromagnetic radiation and other kinds of noises and disturbances in the communication system affect communication direction to corrupted messages, an errors in the received codeword (message) or bit changing might be happened during a transmission of data [1] [2]. So, the data can be corrupted during a transmission and dispatching from the transmitter (sender) to the receiver, but during a data transmission, it might be affected by a noise, then the input data or generated codeword is not the same as received codeword or output data [1]. The error or bit changing which is happened in data bits can change the real value of the bit(s) from 0 to 1 or vice versa. In the following simple figure, a base structure of communication system is indicated. And this figure indicates that how the binary signal can be affected by a noise or other effects during a transmission on the noisy communication channel:

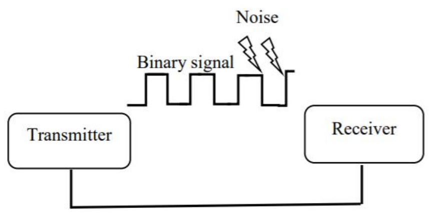


Figure: Data Transmission

To get the best idea of correction by using a redundancy in digital communication, first of all, it is necessary to model the main scheme contains two main parts called encoding and decoding like the following figure and in the next parts all of these parts will be explained in detail and implemented in verilog language.

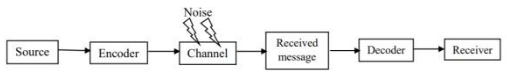


Figure: The main scheme of hamming code

According to this figure [2] [3], first of all, the code will be generated by a source, and then to do encoding part, the parity (redundancy) bits will be added by the encoder to the data bits which sent from a source. After that, the generated codeword which is a combination of data bits and parity bits will be transmitted to the receiver side, and during transmission, the error or bit changing might be happened in the communication channel over produced codeword. At the end, the corrupted error must be detected and corrected by decoder on the receiver side.

II. RELATED WORK

One of the proper subset of information theory is called coding theory, but the concept of these theories are completely different [2]. The main subject is began from a seminar paper which presented by Claude Shannon in the mid of 20th century. And in that paper, he demonstrated that good code exist, but his assertions were probabilistic. Then after Shannon's theorem, Dr Hamming [1] [3] and Marcel Golay [5] presented their first error correction codes which called Hamming and Golay codes [2]. Source Encoder Channel Received message Decoder Receiver
In 1947, Dr Hamming introduced and invented the first method and generation of error correction code called hamming code [1] [6]. Hamming code is capable to correct one error in a block of the received message contains binary symbols [7]. After that, Dr Hamming has published a paper [1] in the Bell technical journal with a subject of error detection and error correction code. In 1960, other methods for error detection and error correction codes were introduced and invented by another inventor, for example, BCH code was invented by Bose, Chaudhuri and Hocquenghem who are a combination of the surname of the initial inventors' of BCH code [7].

Another error detection and error correction code which presented in 1960 was Reed Solomon (RS) code that invented by two inventors called Irving Reed and Gustave Solomon and this method was developed by more powerful computers and more efficient algorithm for decoding part [8]. 2.2. Types of Hamming Code There are three more significant types of hamming codes which called 1. Standard hamming code, 2. Extended hamming code [9], 3. Extended hamming product code, but in this research, the standard hamming code is used and implemented.

III. DESIGN METHODLOGY

A. Design of Diagonal Hamming method for memory: proposed design of Diagonal Hamming based multi-bit error detection and correction technique to the memory is shown in Figure. Using this approach of diagonal Hamming bits, the errors in the message can be recognized and can be rectified which are saved in the memory. In encoding technique message bits are given as input to the Diagonal Hamming encoder and the Hamming bits are calculated. Message and Hamming bits (32+24 bits) are saved in the memory after the encoding technique.

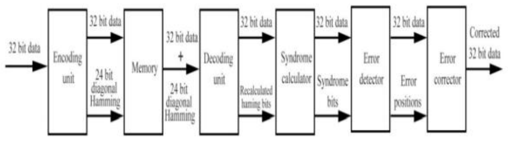
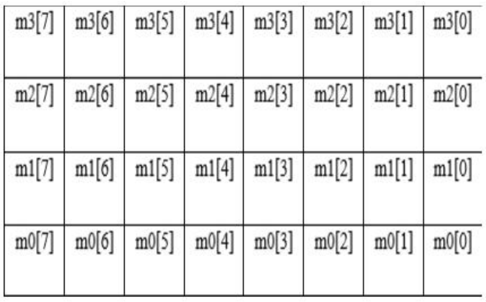


Figure: Proposed Architecture of Diagonal hamming method for memory



m3[7]	m3[6]	m3[5]	m3[4]	m3[3]	m3[2]	m3[1]	m3[0]
m2[7]	m2[6]	m2[5]	m2[4]	m2[3]	m2[2]	m2[1]	m2[0]
m1[7]	m1[6]	m1[5]	m1[4]	m1[3]	m1[2]	m1[1]	m1[0]
m0[7]	m0[6]	m0[5]	m0[4]	m0[3]	m0[2]	m0[1]	m0[0]

Figure: 32-bit message organization (C=8 and R=4)

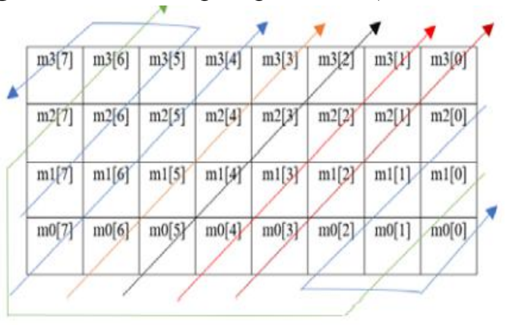
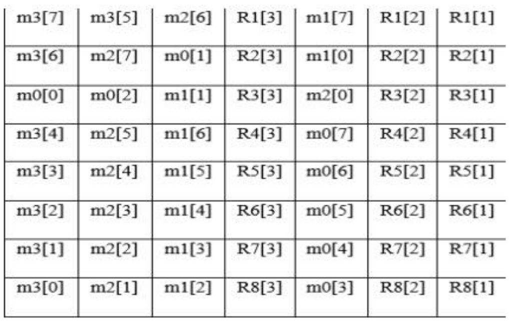

m3[7] m3[6] m3[5] m3[4] m3[3] m3[2] m3[1] m3[0]
m2[7] m2[6] m2[5] m2[4] m2[3] m2[2] m2[1] m2[0]
m1[7] m1[6] m1[5] m1[4] m1[3] m1[2] m1[1] m1[0]
m0[7] m0[6] m0[5] m0[4] m0[3] m0[2] m0[1] m0[0]

Figure: Grouping of the message bits to generate hamming bits

Errors which are occurring in the message bits, are saved in the memory and can be recognized and rectified in the decoding technique.

B. Design of Diagonal Hamming Encoder: For example, message of 32 bits is accounted in the proposed method. The message is represented in the form m x n matrix. The grouping of the message bits is depicted in Figure. The encoder generates the hamming bits and the hamming bits are obtained by grouping the message bits and the hamming bits are calculated with the help of hamming code. The message bits are patterned as shown in Figure. Eight diagonals are considered in this Diagonal Hamming method and each diagonal consists of 4 message bits.



m3[7]	m3[5]	m2[6]	R1[3]	m1[7]	R1[2]	R1[1]
m3[6]	m2[7]	m0[1]	R2[3]	m1[0]	R2[2]	R2[1]
m0[0]	m0[2]	m1[1]	R3[3]	m2[0]	R3[2]	R3[1]
m3[4]	m2[5]	m1[6]	R4[3]	m0[7]	R4[2]	R4[1]
m3[3]	m2[4]	m1[5]	R5[3]	m0[6]	R5[2]	R5[1]
m3[2]	m2[3]	m1[4]	R6[3]	m0[5]	R6[2]	R6[1]
m3[1]	m2[2]	m1[3]	R7[3]	m0[4]	R7[2]	R7[1]
m3[0]	m2[1]	m1[2]	R8[3]	m0[3]	R8[2]	R8[1]

Figure: Positions of the hamming bits in accordance with the grouped message bits

Message bits are grouped as shown in the Fig. 3 in the specified directions. The first diagonal consists of m3[7], m3[5], m2[6], m1[7], the second diagonal has m3[6], m2[7], m0[1], m1[0], the third diagonal has m0[0], m0[2], m1[1], m2[0], the fourth diagonal has m3[4], m2[5], m1[6], m0[7], the fifth diagonal has m3[3], m2[4], m1[5], m0[6], the sixth diagonal has m3[2], m2[3], m1[4], m0[5], the seventh diagonal has m3[1], m2[2], m1[3], m0[4] and the eight diagonal has m3[0], m2[1], m1[2], m0[3]. Each one of the diagonals has four message bits. For the respective groups, the hamming bits are calculated as shown in Figure. The hamming bits are shown as R1, R2, R3, R4, R5, R6, R7, R8 arrays and these arrays consist of 3 bits. The hamming bits are calculated as given in equations (1)- (24) : For the first row: R1[1] = m1[7] ⊕ m2[6] ⊕ m3[7]; (1) R1[2] = m1[7] ⊕ m3[5] ⊕ m3[7]; (2) R1[3] = m2[6] ⊕ m3[5] ⊕ m3[7]; (3) For the second row: R2[1] = m1[0] ⊕ m0[1] ⊕ m3[6]; (4) R2[2] = m1[0] ⊕ m2[7] ⊕ m3[6]; (5) R2[3] = m0[1] ⊕ m2[7] ⊕ m3[6]; (6) For the third row: R3[1] = m2[0] ⊕ m1[1] ⊕ m0[0]; (7) R3[2] = m2[0] ⊕ m0[2] ⊕ m0[0]; (8) R3[3] = m1[1] ⊕ m0[2] ⊕ m0[0]; (9) For the fourth row: R4[1] = m0[7] ⊕ m1[6] ⊕ m3[4]; (10) R4[2] = m0[7] ⊕ m2[5] ⊕ m3[4]; (11) R4[3] = m1[6] ⊕ m2[5] ⊕ m3[4]; (12) For the fifth row: R5[1] = m0[6] ⊕ m1[5] ⊕ m3[3]; (13) R5[2] = m0[6] ⊕ m2[4] ⊕ m3[3]; (14) R5[3] = m1[5] ⊕ m2[4] ⊕ m3[3]; (15) For the sixth row: R6[1] = m0[5] ⊕ m1[4] ⊕ m3[2]; (16) R6[2] = m0[5] ⊕ m2[3] ⊕ m3[2]; (17) R6[3] = m1[4] ⊕ m2[3] ⊕ m3[2]; (18) For the seventh row: R7[1] = m0[4] ⊕ m1[3] ⊕ m3[1]; (19) R7[2] = m0[4] ⊕ m2[2] ⊕ m3[1]; (20) R7[3] = m1[3] ⊕ m2[2] ⊕ m3[1]; (21) For the eight row: R8[1] = m0[3] ⊕

$m1[2] \oplus m3[0]$; (22) $R8[2] = m0[3] \oplus m2[1] \oplus m3[0]$; (23) $R8[3] = m1[2] \oplus m2[1] \oplus m3[0]$; (24) In the encoder, the hamming bits are calculated for message. We get 24 hamming bits in total for 32-bit message.

C. Proposed Diagonal Hamming Decoder: The message bits which are encoded and kept in memory as a matrix as shown in Fig. 4, and are given as input to the decoder. The decoder now segregates message and hamming bits and it recalculates the hamming bits and evaluates syndrome bits. The syndrome bits are evaluated using the equations given in (25)-(48) : For first row $S1[1] = R1[1] \oplus m1[7] \oplus m2[6] \oplus m3[7]$; (25) $S1[2] = R1[2] \oplus m1[7] \oplus m3[5] \oplus m3[7]$; (26) $S1[3] = R1[3] \oplus m2[6] \oplus m3[5] \oplus m3[7]$; (27) For the second row: $S2[1] = R2[1] \oplus m1[0] \oplus m0[1] \oplus m3[6]$; (28) $S2[2] = R2[2] \oplus m1[0] \oplus m2[7] \oplus m3[6]$; (29) $S2[3] = R2[3] \oplus m0[1] \oplus m2[7] \oplus m3[6]$; (30) For the third row: $S3[1] = R3[1] \oplus m2[0] \oplus m1[1] \oplus m0[0]$; (31) $S3[2] = R3[2] \oplus m2[0] \oplus m0[2] \oplus m0[0]$; (32) $S3[3] = R3[3] \oplus m1[1] \oplus m0[2] \oplus m0[0]$; (33) For the fourth row: $S4[1] = R4[1] \oplus m0[7] \oplus m1[6] \oplus m3[4]$; (34) $S4[2] = R4[2] \oplus m0[7] \oplus m2[5] \oplus m3[4]$; (35) $S4[3] = R4[3] \oplus m1[6] \oplus m2[5] \oplus m3[4]$; (36) For the fifth row: $S5[1] = R5[1] \oplus m0[6] \oplus m1[5] \oplus m3[3]$; (37) $S5[2] = R5[2] \oplus m0[6] \oplus m2[4] \oplus m3[3]$; (38) $S5[3] = R5[3] \oplus m1[5] \oplus m2[4] \oplus m3[3]$; (39) For the sixth row: $S6[1] = R6[1] \oplus m0[5] \oplus m1[4] \oplus m3[2]$; (40) $S6[2] = R6[2] \oplus m0[5] \oplus m2[3] \oplus m3[2]$; (41) $S6[3] = R6[3] \oplus m1[4] \oplus m2[3] \oplus m3[2]$; (42) For the seventh row: $S7[1] = R7[1] \oplus m0[4] \oplus m1[3] \oplus m3[1]$; (43) $S7[2] = R7[2] \oplus m0[4] \oplus m2[2] \oplus m3[1]$; (44) $S7[3] = R7[3] \oplus m1[3] \oplus m2[2] \oplus m3[1]$; (45) For the eight row: $S8[1] = R8[1] \oplus m0[3] \oplus m1[2] \oplus m3[0]$; (46) $S8[2] = R8[2] \oplus m0[3] \oplus m2[1] \oplus m3[0]$; (47) $S8[3] = R8[3] \oplus m1[2] \oplus m2[1] \oplus m3[0]$; (48) If all the syndrome bits are equal to zero, then it represents that the message bits are not corrupted and if anyone of the syndrome bits in non-zero, then it represents the message bit(s) are corrupted. These corrupted bits need correction so, the message bits are sent to the error correction part. In the correcting of error part, the location of error is identified by doing the following calculations: Suppose if the error is in the third row of the message organization, then the error position is calculated as: $(S3[3] * (22\ )) + (S3[2] * (21\ )) + (S3[0] * (20\ ))$; (49) After the position is calculated, the error corrector negates the bit corresponding to that position to correct the data bit. This process is done till all the corrupted bits are corrected.

IV. RESULTS

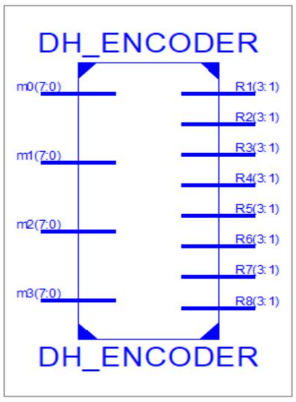


Figure: Encoder

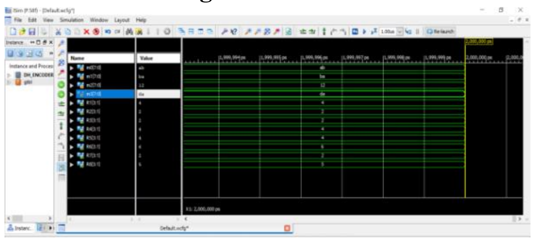


Figure: Encoder output

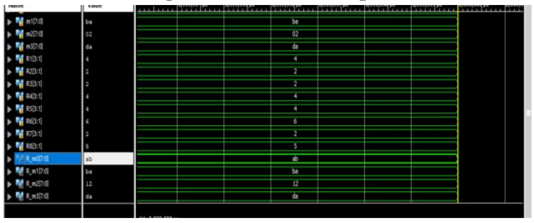
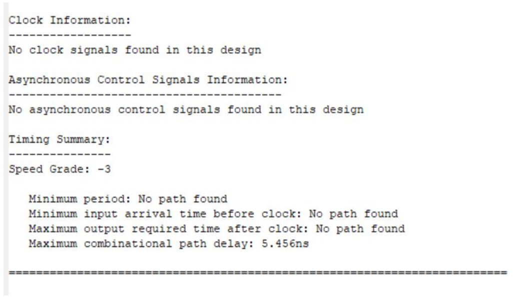


Figure: Decoder output



```
Clock Information:
------------------
No clock signals found in this design

Asynchronous Control Signals Information:
----------------------------------------
No asynchronous control signals found in this design

Timing Summary:
---------------
Speed Grade: -3

   Minimum period: No path found
   Minimum input arrival time before clock: No path found
   Maximum output required time after clock: No path found
   Maximum combinational path delay: 5.456ns

=========================================================================
```

Figure: Delay

CONCLUSION

Diagonal Hamming coding method is proposed in this paper and the main idea behind this work is to reduce maximum errors during transmission of bits in memory. Diagonal Hamming method identifies and corrects up to maximum of 8bit errors in a row for 32-bit input. Diagonal Hamming method ensures less area and delay by 91.76 percentage and 84.18 percentage respectively. Huge amount of error correction is possible using this Diagonal Hamming method.

REFERENCES

[1] Paromita Raha , M. Vinodhini and N.S Murty," Horizontal Vertical Parity and Diagonal Hamming Based Soft errors Detection and Correction of Memories," in Proc. Int Conf on Comp Com and Info, 2017.

[2] S.ASAI, "Semi conductor memory trends," in Proc. of IEEE, Vol.74, 1986.

[3] Shivani Tambatkar, Siddharth Narayana Menon, Sudarshan.V, M.Vinodhini and N.S.Murty, " Error Detection and Correction in Semiconductor Memories using 3D Parity Check Code with Hamming Code," in Proc. International Conf on Comm and Signal Processing, 2017.

[4] Varinder Singh and Narinder Sharma, "A Review on Various Error Detection and Correction Methods Used in Communication," American International Journal of Research in Science, Technology, Engineering and Mathematics, pp. 252 - 257, 2015.

[5] Mostafa Kishani, Hamid R. Zarandi, Hossein Pedram, Alireza Ta- jary, Mohsen Raji and Behnam Ghavami, "HVD: Horizontal-Vertical-Diagonal error detecting and correcting code to protect against with soft errors," Design automation for embedded systems Journal, Vol. 15, no. 3, pp. 289- 310, May 2011.

[6] S. Vijayalakshmi and V. Nagarajan (FEB 2019), "Design and Implementation of Low Power High-Efficient Transceiver for Body Channel Communications", Springer- Journal of Medical Systems

[7] Md. Shamimur Rahman, Muhammad Sheikh Sadi, Sakib Ahammed and Jan Jurjens, " Soft Error Tolerance using Horizontal – Vertical Double – Bit Diagonal Parity Method," in Proc 2nd IEEE International Conference on Electrical Engineering and Information and Commu- nication Technology (ICEEICT) , 21-23 May 2015.

[8] 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, Vol. 22, Issue:1, pp. 127-135, Jan. 2014

[9] Ahilan A. and Deepa P., "Modified Decimal Matrix Codes in FPGA Configuration Memory for Multiple Bit Upsets," in Proc.International Conference on Computer Communication and Informatics (ICCCI), pp. 1- 5, Jan. 2015

[10] M.Vinodhini and N.S. Murty., "Relible Low Power NoC Intercon- nect," Micoprocessor and Microsystems-Embedded Hardware Design, Vol.57, March 2018.