International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2013): 4.438

A New Methodology for Error Detection and Correction to Realize Fault Tolerant Memory

Neethu V[1], Anju S L[2]

[1]P G Scholar, VLSI and Embedded Systems, Department of ECE, T K M Institute of Technology, Kollam, India

[2]Assistant Professor, Department of ECE, T K M Institute of Technology, Kollam, India

Abstract*: Scaling of CMOS technology to nanoscale increases soft error rate in memory cells. Both single bit upset and Multiple Cell Upsets (MCUs) causes reliability issues in memory applications. Hence to provide fault-tolerant memory cells, Error Correction Codes (ECCs) are used. But these codes require more area, power and higher delay overhead. Thus Matrix Codes (MCs) based on Hamming codes and parity codes are used for detection and correction of multiple errors with less decoding delay. The use of matrix codes is capable of correcting only two errors. Hence to maximize the capability of error detection and correction, Decimal Matrix Code (DMC) based on decimal algorithm is used. This algorithm utilizes decimal integer addition and decimal integer subtraction to detect and correct errors. DMC uses encoder reuse technique for fault tolerant memory protection. The use of this algorithm enables more errors to be detected and corrected by the utilization of large number of redundant bits. Thus the framework can be modified so as to reduce the number of redundant bits by means of using parity matrix codes in which a 32 bit data is divided into 2bits of 16 blocks. Coding can be done by means of VHDL language and ModelSim can be used for simulation.*

Keywords: Soft errors, decimal algorithm, syndrome, redundant bit, DMC encoder, DMC decoder.

1. Introduction

Scaling down of semiconductor devices to nanoscale causes threshold voltage variation, negative bias temperature instability, short channel, single bit upsets and multiple cell upsets [5]. Thus to make memory cells fault tolerant as possible Error Correction Codes (ECCs) are used [3]. Most commonly used error correction codes are the hamming codes, Bose Chaudhuri Hocquenghem codes and Reed Solomon codes. Hamming codes are a set of error-correction code that are mainly used to detect and correct bit errors. It makes use of the concept of parity bits. These parity bits are added to data so as to ensure the validity of the data when it is read or after it has been received in a data transmission. This error-correction code can not only identify a single bit error in the data unit, but also its location in the data unit by using more than one parity bit. When the number of ones is incorrect, the parity count indicates that single bit errors are detected which in turn reveals that a data bit has been flipped. By using more than one parity bit, hamming codes are capable of detecting two bit errors. The number of parity bits required depends on the number of bits that is sent during the data transmission. The major disadvantage of hamming code is that it is capable of protecting only memories that are effected by single bit upsets.

Bose Chaudhuri Hocquenghem codes are capable of correcting a given number of bits in any position. These codes uses complex or sophisticated decoding algorithms like complex algebraic decoders that decode in fixed time. Moreover these are computationally very costly. Reed Solomon groups the bits into blocks to correct soft errors. The major disadvantage of these codes is it require complex decoding algorithms and tables of constants. Moreover all these error correction codes require more area, power and delay overheads. Thus interleaving technique is used. In interleaving the bits that belong to the same logical word are placed apart such that MCU effect only a bit per word. But interleaving causes complex memory design, area and power consumption. MCUs that can be corrected by ECC can still be problematic in high performance devices such as content addressable memories that are used in network processors and routers. Thus matrix codes in which hamming codes and parity codes are combined in matrix format is used to assure the reliability of memory chips. Matrix Codes (MCs) are used to efficiently correct MCUs per word [4]. It has low decoding delay and moreover in this a single word is divided into multiple rows and columns. Hamming codes are used to protect bits per row, while parity code is added in each column [4]. The vertical syndrome bits get activated, when two errors are detected by Hamming, hence it enables the correction of two errors. So the major disadvantage of matrix code is, it is capable of correcting only two errors. Thus Decimal Matrix Code (DMC) is used to provide memory reliability. DMC utilizes decimal algorithm i.e. decimal integer addition and decimal integer subtraction to detect errors and correct errors.

In this work, the main objective of using decimal matrix code is to maximize the error detection and correction capability much more than that of the matrix code. Moreover by using parity matrix codes, the large utilization of redundant bits in decimal matric codes for error detection and correction can be reduced.

2. Theory

The impact of technology scaling has come to a level that reliability of memory gets affected. Especially, SRAM memory failure rates are increasing significantly, therefore posing a major reliability concern in many applications. The unexpected or unwanted changes in the value of a bit(or bits) inside a memory is called as soft errors. Due to the sudden or random change in the data, it gets stored inside the memory

as if it was a valid data [10]. By means of replacement or restoring the erroneous data, the value of the system starts operating in the manner it should be. There are usually two types of soft errors. They are single bit upset and multiple cell upset. Single-bit upsets are errors that occur when only one bit of given data unit is changed from 1 to 0 or from 0 to 1. In multiple cell upset, two or more bits in the data unit changes from 0 to 1 or vice-versa. Such type of errors does not mean that error occurs in consecutive bits.

For detection and correction of errors, extra bits are added to every data byte of the memory. The extra bit that is added is called as parity check bit. Parity bits are the simplest form of error detecting codes. This parity bit is used to determine whether the byte that is stored in the memory has the correct number of 0's and 1's in it. If the count changes, it indicates that there is an an error. There are two types of parity bits- even parity bit and odd parity bit. In even parity bit, if the count of ones in the given data is odd then the parity bit is set as one. If the count of ones is even then the parity bit is set as zero. In odd parity bit, if the count of ones in the given data is even then the parity bit is set as one. If the count of ones is odd then the parity bit is set as zero. But in some cases these parity bit itself may have an error and thus the error cannot be detected. For this purpose error correcting codes were used. An algorithm which expresses a sequence of numbers, such that the errors that are present in it can be easily detected and corrected is called as an error-correction code. The study of error-correcting codes is called as coding theory. Error-correcting codes are more complex compared to error detecting codes moreover it require more redundant bits. Due to this the number of bits required to correct multiple bit errors are very high.

A schematic of a microscopic soft-error model for an SRAM cell is as shown in Figure 1.

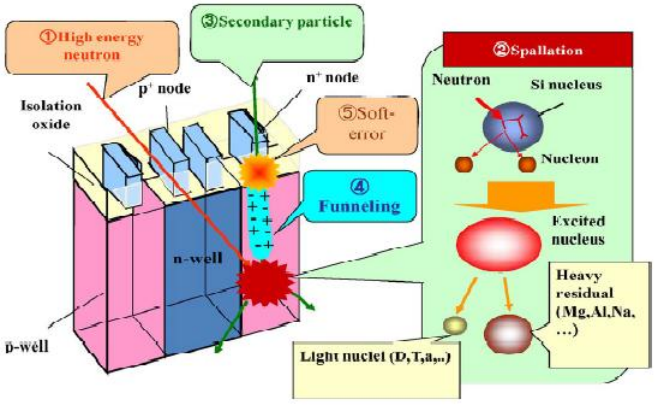

Figure 1: Neutron induced soft error in SRAM

An SRAM cell consist of two n+ nodes in the p-well and two p+ storage nodes in the n-well. Two sets of adjacent n+ and p+ nodes correspond to two potential states: high or low. Nuclear spallation reaction takes place between the neutron and the nucleus (mostly Si) of the device, when a ballistic neutron penetrates into the SRAM [5]. Spallation reaction takes place when two nuclei collide with high energy in which the involved nuclei are either disintegrated into protons, neutrons, light nuclei and the nucleons. As a result of such a reaction, nucleons that is the protons and neutrons collide with each other in the nucleus of Si. Some of these nucleons escape from the nucleus when they have large value of kinetic energies. This process is called as intra nuclear cascade. Due to this process, light nuclei gets evaporated from the nucleus of Si. Hence it produces electron-hole pairs along with the ion track when nucleons, light nuclei and the residual nucleus run inside the SRAM cell. These are called the secondary ions [5]. Energy necessary to produce one such pair of electron and hole is 3.6 eV in Si. Some of the charges are collected to the storage node mainly through the diffusion process, whenever secondary ions hit the storage nodes. When the amount of the charges exceeds a critical charge, then flipping occurs in the logical state of SRAM. Hence soft-error takes place in the SRAM.

3. Methodology

Decimal Matrix Codes (DMC) are mainly used to avoid soft errors due to multiple cell upsets in memory unit. DMC make use of decimal algorithm that is decimal integer addition and decimal integer subtraction to detect errors. In this the information bits are provided to the DMC encoder during the encoding process. As a result, the horizontal redundant bits, vertical redundant bits and the information bits are obtained. Once encoding process gets completed, this codeword is stored to the memory. If multiple cell upset occurs in memory, this can be corrected during the decoding process. Thus the error detection and correction capability is maximized due to the usage of decimal matrix code.

3.1 Fault Tolerant Memory

The basic block diagram for a fault tolerant memory protected with DMC is as shown in Figure 2. A fault tolerant memory protected with DMC consists of an encoder, memory unit for both information and redundant bit storage and a decoder. The decoder consist of syndrome calculator, error locator and an error corrector.

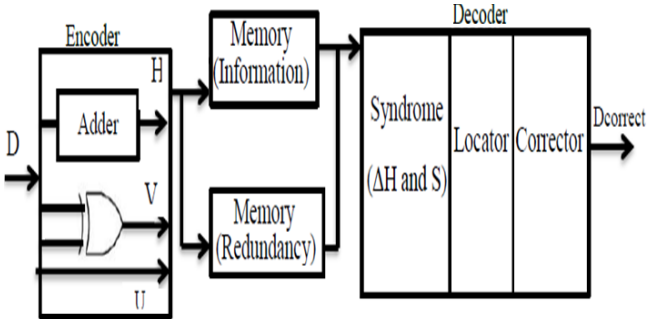

Figure 2: Block Diagram of a fault tolerant memory protected with DMC.

3.2 DMC Encoder

At first an N-bit word is divided into 'k' symbols of 'm' bits such that N=k*m where the symbols are arranged as k=k1*k2. Here k1 represents the number of rows and k2 represents the number of columns. Secondly the horizontal redundant bits (H) are calculated by using decimal integer addition of selected symbols per row. Thirdly the vertical redundant bits (V) are calculated by binary operation of the bits per column. Here a 32 bit word is chosen and it is

divided into 8 symbols of 4 bit, such that k1=2 and k2=4. The logical organization of a 32 bit word is as shown in Figure 3.

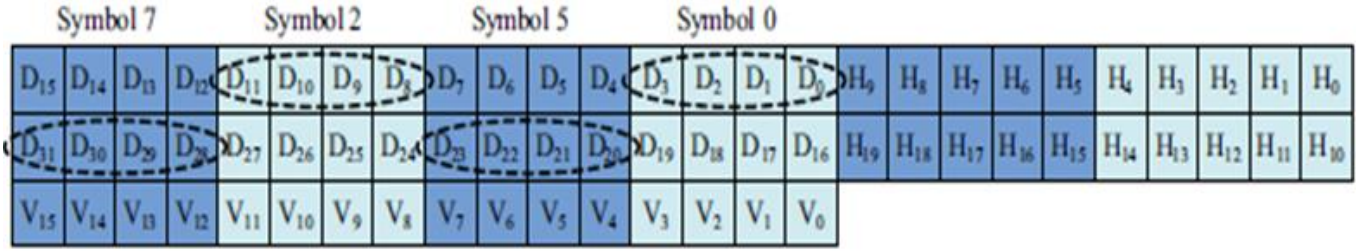

Figure 3: 32 bit DMC logical organization

Here D0-D31 are the information bits, H0-H19 are the horizontal redundant bits and V0-V15 are the vertical redundant bits. The maximum correction capability and the number of redundant bits are different for different values of 'k' and 'm'. Hence 'k' and 'm' should be properly chosen to maximize the error correction capability and to reduce the number of redundant bits. The encoder that calculates the redundant bits using adders and XOR gates is as shown in Figure 4.

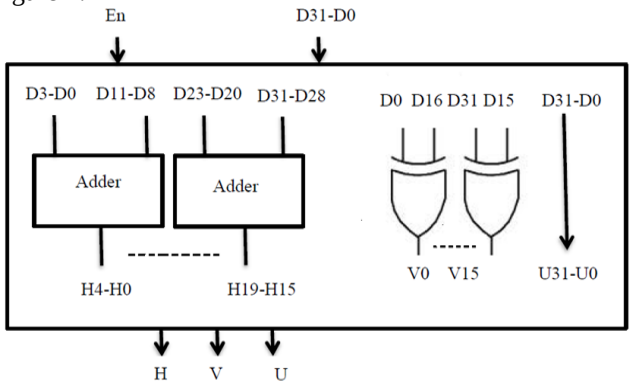

Figure 4: DMC encoder.

The horizontal bits are calculated by decimal integer addition as follows:

$$H4H3H2H1H0 = D3D2D1D0 + D11D10D9D8 \quad (1)$$

$$H9H8H7H6H5 = D7D6D5D4 + D15D14D13D12 \quad (2)$$

$$H14H13H12H11H10 = D19D18D17D16 + D27D26D25D24 \quad (3)$$

$$H19H18H17H16H15 = D23D22D21D20 + D31D30D29D28 \quad (4)$$

Where '+' indicates decimal integer addition. The vertical redundant bits are calculated as follows:

$$V\,0 = D0 \text{ XOR } D16 \quad (5)$$

$$V\,1 = D1 \text{ XOR } D17 \quad (6)$$

$$V\,2 = D2 \text{ XOR } D18 \quad (7)$$

$$V\,3 = D3 \text{ XOR } D19 \quad (8)$$

Accordingly all the vertical redundant bits are calculated. Thus at the output of the encoder horizontal redundant bits, vertical redundant bits and the information bits (U0-U31) are obtained. These outputs are then given to the memory unit.

3.3 Memory

The output from the encoder is provided to the memory information block and to the memory redundant block. Suppose if the information bits stored in the memory gets flipped it can be corrected during the decoding process. This information data, that is flipped is denoted as D0'-D31'.

3.4 DMC Decoder

The decoding process takes place in two steps: Firstly the received redundant bits [H4H3H2H1H0]' and [V0- V3]' are calculated from the information bits obtained from the memory information block. Secondly the horizontal syndrome bits ΔH4H3H2H1H0 and vertical syndrome bits S3-S0 are calculated. The DMC decoder is as shown in Figure 5.

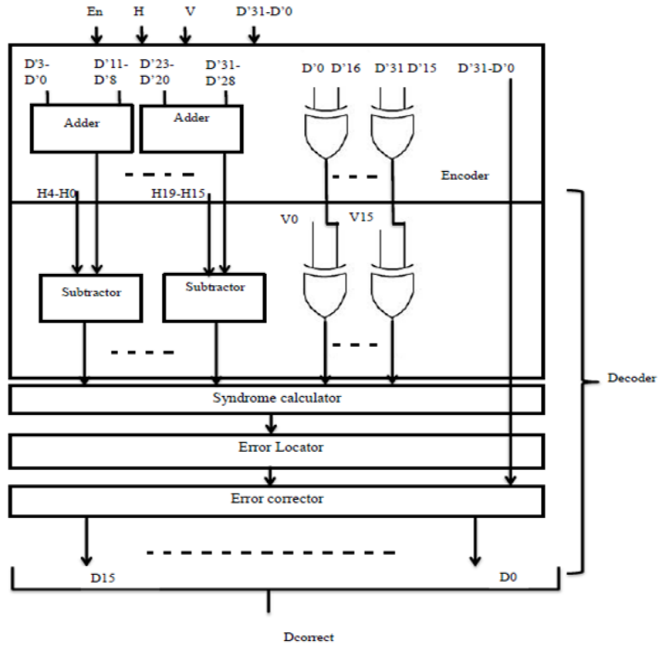

Figure 5: DMC decoder

To calculate the horizontal syndrome bits, decimal integer subtraction is performed between the received horizontal redundant bits and the horizontal redundant bits obtained from memory redundant block and is as shown below:

$$\Delta H4H3H2H1H0 = [H4H3H2H1H0]' - H4H3H2H1H0 \quad (9)$$

$$\Delta H9H8H7H6H5 = [H9H8H7H6H5]' - H9H8H7H6H5 \quad (10)$$

$$\Delta H14H13H12H11H10 = [H14H13H12H11H10]' - H14H13H12H11H10 \quad (11)$$

$$\Delta H19H18H17H16H15 = [H19H18H17H16H15]' - H19H18H17H16H15 \quad (12)$$

Where '-' indicates decimal integer subtraction. To calculate vertical syndrome bits, XOR operation is performed between the received vertical redundant bits and the vertical

redundant bits obtained from memory redundant block and is as shown below:

S0 = V 0' XOR V 0 (13)
S1 = V 1' XOR V 1 (14)
S2 = V 2' XOR V 2 (15)
S3 = V 3' XOR V 3 (16)

Accordingly all the vertical syndrome bits, are calculated. Suppose if all the bits of ΔH4H3H2H1H0 and S3-S0 are zero then there is no error. This indicates that there is no error in symbol zero. If ΔH4H3H2H1H0 and S3-S0 is non-zero, it indicates that there is error in symbol 0 and this error can be corrected as follows:

D0correct = D0' XOR S0 (17)
D1correct = D1' XOR S1 (18)
D2correct = D2' XOR S2 (19)
D3correct = D3' XOR S3 (20)

The algorithm for a fault tolerant memory protected with DMC is as shown in Figure 6.

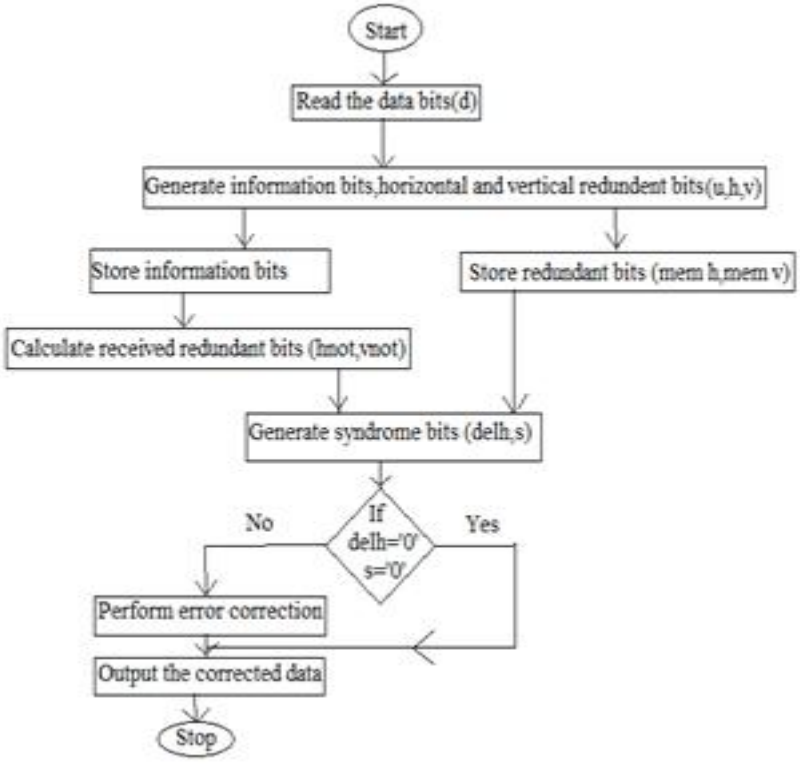

Figure 6: Algorithm of DMC.

Thus decimal matrix code maximizes the error detection and correction capability. But it results in decoding error when the following factors occur:

- When upsets occurs in both the rows.
- When the decimal integer sum value of both the original data and that of the flipped data is the same.
- When the bits in symbol 0 and symbol 2 are upset such that their decimal integer sum of information bit is equal to 2^m-1.

Moreover DMC requires large number of redundant bits to detect and correct errors. The below Table1 depicts this.

Table 1: Redundant bits needed for a given information bit.

Number of information bits	Number of redundant bits
32	36

3.5 Parity Matrix Code

Parity Matrix Codes (PMC) are used to reduce the large number of redundant bits for error detection and correction. The block diagram for a fault tolerant memory protected with PMC is as shown in Figure 7.

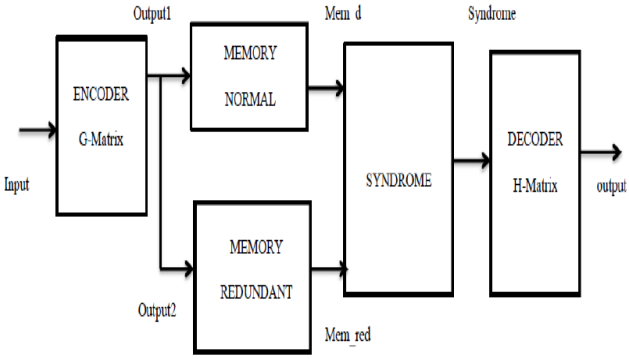

Figure 7: Block diagram for a fault tolerant memory protected with PMC

In this at first the information bits of 32-bit is applied to the encoder. This information bit is divided into 2 bits of 16 blocks. These 2 bit of 16 blocks are encoded with the generator matrix (G-matrix). A generator matrix is formed by the combination of both identity matrix and parity bits. Suppose if any error occurs in the memory it can be corrected at the decoder. For decoding a codeword parity check matrix [H-matrix] is required. H-matrix is a combination of transpose of parity bits and that of identity matrix. Thus when compared to DMC, the redundant bit is reduced from 36 to 32 in PMC.

4. Results and Discussion

The modules are modelled using VHDL in Xilinx ISE Design Suite 8.1i and the simulation of the design is performed using Modelsim SE 6.3f to verify the functionality of the design. Here a structural model of fault tolerant memory protected with DMC is developed. The fault tolerant memory protected with DMC contains modules such as an encoder, memory information block, memory redundant block, reuse encoder, syndrome calculator and an error corrector. Simulation results of these modules are as shown below.

4.1 Simulation of Encoder

Any information bit given to the encoder is divided into 'k' symbols of 'm' bits. Where k=k1*k2, k1 represents the number of rows and k2 represents the number of columns. If the information bit is

d=11000101001010000001111001001010
This 32 bit data gets divided into eight symbols of four bits each. Then the DMC logical organization is as shown in Figure 8.

0001 1110 0100 1010 1100 0101 0010 1000	00101 11000 01110 01101
1101 1011 0110 0010	

Figure 8: Logical organization

At the encoder when the information bit (d) is applied, horizontal redundant bits(h), vertical redundant bits(v) and information bit(u) is obtained. Horizontal redundant bits are obtained by the decimal integer addition and the vertical redundant bits are obtained by the binary operation. To obtain h4-h0 decimal integer addition is performed between symbol 0 and symbol 2. To obtain v0 XOR operation between d0 and d16 is performed. The simulated waveform is as shown in Figure 9. Hence
h= 01110 01101 00101 11000
v= 1101 1011 0110 0010
u=11000101001010000001111001001010

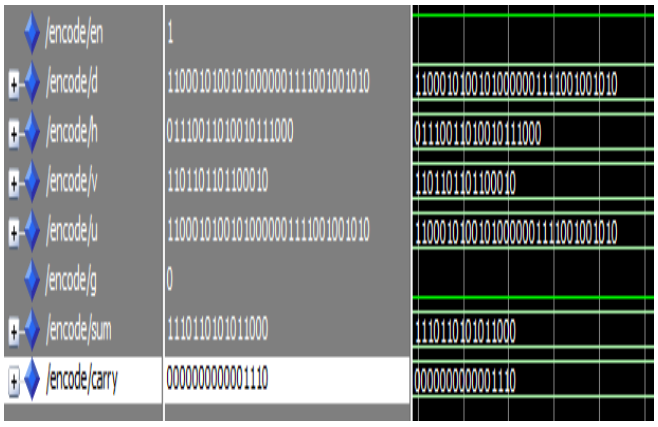

Figure 9: Simulation of encoder

4.2 Simulation of Memory Information Block

To the memory information block, the information bit (d) is applied. Suppose if this information bit gets flipped it gets stored as flipped information bit(dd). The simulated waveform is as shown in Figure 10. Hence
dd=11000101001010001110000110110101

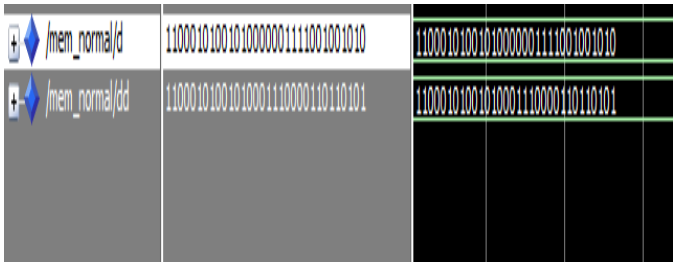

Figure 10: Simulation of memory information block

4.3 Simulation Of Memory Redundant Block.

At the memory redundant block, both horizontal redundant bits(h) and vertical redundant bits (v) are applied and memory horizontal redundant bits(mem_h) and memory vertical redundant bits (mem_v) are obtained. The simulated waveform is as shown in Figure 11. Hence
mem_h= 01110 01101 00101 11000
mem_v=1101 1011 0110 0010

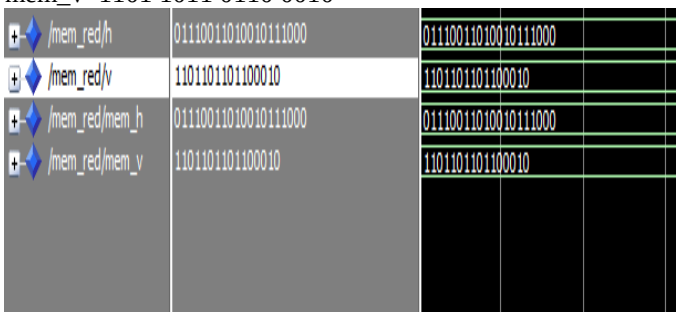

Figure 11: Simulation of memory redundant block.

4.4 Simulation of Reuse Encoder

At the reuse encoder part of the decoder, the received horizontal redundant bits (hnot) and received vertical redundant bits (vnot) are calculated from the flipped information bit (dd). This is mainly done to determine the syndrome bits during syndrome calculation. The received horizontal redundant bits are obtained by the decimal integer addition and the received vertical redundant bits are obtained by the binary operation. The simulated waveform is as shown in Figure 12. Hence
hnot= 01110 01101 11001 00110
vnot= 0010 0100 1001 1101

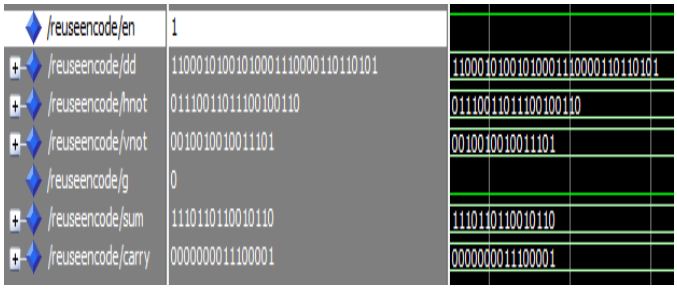

Figure 12: Simulation of reuse encoder

4.5 Simulation Of Syndrome Calculator

To the syndrome calculator section, the horizontal and the vertical redundant bits obtained from the memory redundant block and that from the reuse encoder block is applied. Thus the horizontal syndrome bits (delh) and the vertical syndrome bits (s) are calculated. The horizontal syndrome bits (delh) are obtained by the decimal integer subtraction of the horizontal redundant bits obtained from the reuse encoder and that of the horizontal redundant bits obtained from the memory redundant block. The vertical syndrome bits (s) are obtained by the XOR operation of the vertical redundant bits obtained from the reuse encoder and that of the vertical redundant bits obtained from the memory redundant block. If the entire bits of horizontal syndrome bit(delh) and the vertical syndrome bit (s) is zero then there is no error. If any one of the bit of horizontal syndrome bit (delh) and the vertical syndrome (s) bit is one then it indicates that there is an error. The simulated waveform is as shown in Figure 13. Hence

delh = 00000 00000 10100 01110

s=1111111111111111

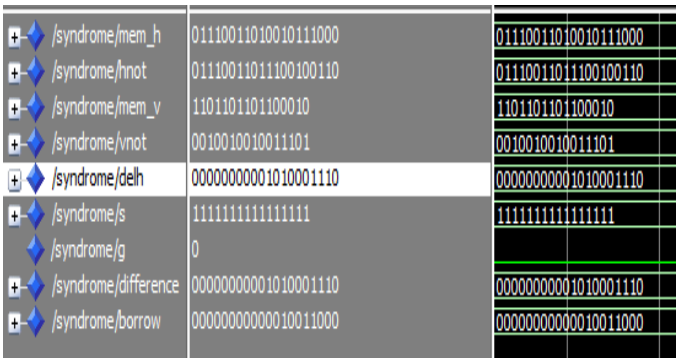

Figure 13: Simulation of syndrome calculator

4.6 Simulation Of Error Corrector

At the error corrector section the output information bit (output) is obtained as equivalent to the information bit (d) by the XOR operation of flipped information bit (dd) and that of the vertical syndrome bit(s).The simulated waveform is as shown in Figure 14. Hence

Output= 11000101001010000001111001001010

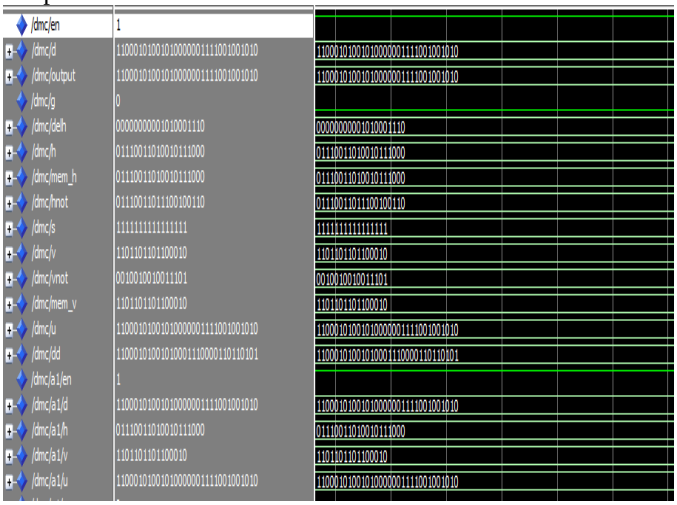

Figure 14: Simulation of error corrector.

5. Conclusion

Soft errors caused due to multiple cell upsets can lead to serious problems in memory applications. Thus to avoid such errors decimal matrix codes were used. Decimal Matrix Codes (DMC) are based on decimal algorithm which is decimal integer addition and decimal integer subtraction. The use of this algorithm maximizes the error detection and correction capability, hence a fault tolerant memory can be obtained. When an information bit is provided to the encoder, the horizontal and vertical redundant bits are calculated. Suppose if the data that is stored in the memory gets flipped its redundant bits are also calculated by the reuse encoder. Then the syndrome calculator calculates the syndrome bits and finally at the error corrector, binary operations are performed to obtain the original information bit. The only disadvantage of DMC is it requires large number of redundant bits for error detection and correction. This large utilization of redundant bits can be avoided by using parity matrix code. PMC divides the 32 bit data into 2 bit of 16 blocks and performs encoding by using generator matrix and decoding by using parity check matrix. As a future work, application based design can be done.

References

[1] Jing Guo, Liyi Xiao, Zhigang Mao, and Qiang Zhaoz, "Enhanced Memory Reliability Against Multiple Cell Upsets Using Decimal Matrix Code", IEEE Transactions On Very Large Scale Integration (VLSI) Systems, Vol. 22, No. 1, January 2014.

[2] Alfonso Snchez-Macin, Pedro Reviriego and Juan Antonio Maestro, "Hamming SECDAED and Extended Hamming SEC-DED-TAED Codes Through Selective Shortening and Bit Placement", IEEE Transactions On Device And Materials Reliability, Vol. 14, No. 1, March 2014.

[3] S. Liu, P. Reviriego, and J. A. Maestro, "Efficient majority logic fault detection with difference-set codes for memory applications", IEEE Very Large Scale Integr. (VLSI) Syst., vol. 20, no. 1,Jan. 2012.

[4] C. Argyrides, D. K. Pradhan, and T. Kocak, "Matrix codes for reliable and cost efficient memory chips,", IEEE Trans. Very Large Scale Integr.(VLSI) Syst., vol. 19, no. 3 Mar. 2011.

[5] Eishi Ibe, Hitoshi Taniguchi, Yasuo Yahagi, Ken-ichi Shimbo, and Tadanobu TobaJ. Rearick, "Impact of Scaling on Neutron-Induced Soft Error in SRAMs From a 250 nm to a 22 nm Design Rule",in IEEE Transactions On Electron Devices, Vol. 57, No. 7, July 2010.

[6] S. Baeg, S. Wen, and R. Wong, "Interleaving distance selection with a soft error failure model ", IEEE Trans. Nucl. Sci., vol. 56, no.4,pp,Aug. 2009.

[7] Daniele Radaelli, Helmut Puchner, SkipWong, and Sabbas Daniel, "Investigation of Multi- Bit Upsets in a 150 nm Technology SRAM Device", IEEE Transactions On Nuclear Science, Vol. 52, No. 6, December 2005.

[8] G. Neuberger, D. L. Kastensmidt, and R. Reis, "An automatic technique for optimizing Reed-Solomon codes to improve fault tolerance in memories ", IEEE Design Test Comput., vol. 22, no. 1,Jan.Feb. 2005.

[9] Massoud Malek "Coding Theory-Binary Linear Hamming Codes,", California State University, East Bay.

[10] " Error Detection and Correction", Data Link control Version 2 CSE IIT, Kharagpur Lesson 39.