International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-8 Issue-2S2, January 2019

Error Detection and Correction Methods for Memories used in System-on-Chip Designs

Gunduru Swathi Lakshmi, Neelima K, C. Subhas

Abstract: ***Memory is the basic necessity in any SoC design. Memories are classified into single port memory and multiport memory. Multiport memory has ability to source more efficient execution of operation and high speed performance when compared to single port. Testing of semiconductor memories is increasing because of high density of current in the chips. Due to increase in embedded on chip memory and memory density, the number of faults grow exponentially. Error detection works on concept of redundancy where extra bits are added for original data to detect the error bits. Error correction is done in two forms: one is receiver itself corrects the data and other is receiver sends the error bits to sender through feedback. Error detection and correction can be done in two ways. One is Single bit and other is multiple bit. Single bit error detection and correction is categorized into two as Classical Algorithm and March Algorithm. Multiple bits error detection and correction is categorized into Adjacent codes and Random codes. Different methods are applicable for different types of faults that manifest as errors.***

Index Terms— Faults and its types, Error Detection and Correction, Single error detection and correction, Multiple Error detection and correction.

I. INTRODUCTION

Memory is of two different types, Primary memory and Secondary memory. Primary memory is also known as main memory and it is used in semiconductor – chips. It is further divided into Volatile and Non-volatile. Volatile Consists of Random Access Memory (RAM) like SRAM, DRAM, SDRAM, EDRAM, FLASH RAM. Non-volatile consists of PROM, EPROM and EEROM. Secondary memory is the devices which have permanent memory. It is divided into two. One is Magnetic memory and other is Optical memory.

There are two main memory types used in semiconductor technology. They are

Random Access Memory (RAM): RAM consists of millions of transistors and capacitors in it. Here the read and write operations are performed as per the processor required. The stored data is volatile memory. RAM is of two types.

- Dynamic Random Access Memory (DRAM): Here the data can be saved in a capacitor which is operated by transistor. It performs refreshing operation. DRAM is mainly used in main memory. It has advantages like simple structure, low cost, medium speed and high density.
  - Static Random Access Memory (SRAM): It consists of a latch or flipflop to store each bit of memory and it does not required any refresh operation. SRAM is mostly used in cache memory and in hand-held devices. It has advantages like high speed and low power consumption. It has drawback like complex structure and expensive. So, it is not used for high capacity applications.

Read Only Memory (ROM): It is a non-volatile memory. It can only access data but cannot modify data. It is of low cost. Some applications of ROM are scanners, ID cards, Fax and game machines.

- During chip fabrication, the data is written by using photo mask. This is known as Mask ROM.
- In PROM, the chip is fabricated first and then the data is recorded electrically. Once the information is recorded, it cannot be changed.
- In Fuse ROM the data can be neither erased nor modified.
- The UV rays are used to erase the data which is present in glass crystal are known as EPROM. It is two types Ultra violet Erasable Programmable ROM (UEPROM) and Electrically Alterable Programmable ROM (EAPROM).
- EEPROM: It uses high electrical signal to erase data.

In both, EPROM and EEPROM's data can be modified if required and the range of subsequent is permitted up to 10^4 to 10^5.

Compact Disk ROM (CD_ROM): CD-ROM is used to store information. Compact Disk-ReWrite (CD-RW) is used to write information into it and to read information from it. It stores system files and in large program applications.

Cache Memory: It is faster and expensive than RAM and used to reserve the frequently access files and program applications. Cache memory is of two types. One is primary cache (inside microprocessor) and secondary cache (in the mother board or near to processor).

Flash Memory: It is development of EEPROM. The data which is written can be erased in blocks and reads the individual cell data.

Manuscript published on 30 January 2019.
***** Correspondence Author (s)
Gunduru Swathi Lakshmi, M.Tech Student, Department of ECE, Research Centre for VLSI and Embedded Systems, Sree Vidyanikethan Engineering College, Sree Sainath Nagar, Tirupati, Andhra Pradesh, India. (E-mail: swathilakshmi.dola@gmail.com)
Neelima K, Assistant Professor, Department of ECE, Research Centre for VLSI and Embedded Systems, Sree Vidyanikethan Engineering College, Sree Sainath Nagar, Tirupati, Andhra Pradesh, India. (E-mail: neelumtech17@gmail.com)
Dr. C. Subhas, Professor, Department of ECE, JNTUA College of Engineering, Kalikiri, Andhra Pradesh, India. (E-mail: schennapalli@gmail.com)

It is used to erase and can be re-programmed in chip areas. It is non-volatile memory and it used in applications like digital cameras, mobiles and memory sticks.

Ferroelectric RAM (F-RAM): It is non-volatile and similar to DRAM. It performs fast read and write operations. These FRAMs are used in CMOS technology to enable MCUS to have their own technology. The size of memory is limited FRAM because of memory density.

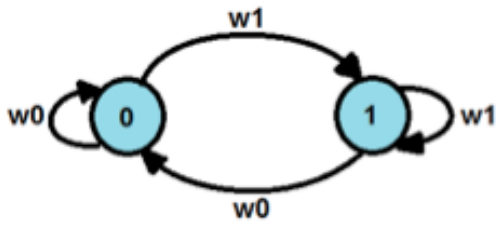
Magneto-resistive RAM (MRAM): It is non-volatile memory and the data is stored in magnetic charges. MRAM memory consumes low power and holds the data even though power is off condition. It is used mainly in electronics industry. It uses devices like disks or tapes.

II. RESULTS AND DISCUSSIONS

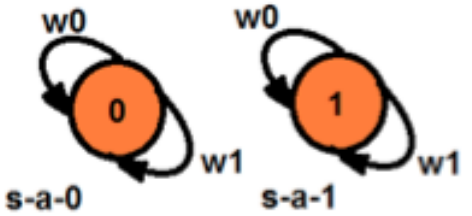
Fault is a physical defect that occurs within hardware or software components [1] [2].

Faults are of five types

Stuck At Fault (SAF): The memory cell value is always zero there exist SA0 fault. If the memory cell value is always one then there exist SA1 fault. SAF are shown in Figure1.



(a)



(b)

Figure 1. State diagram (a) for a good memory cell (b) s-a-0 memory cell and s-a-1 memory cell

Transition fault (TF): If the transition fails from 0 to 1 or vice versa then the cell is faulty. The up TF is indicated as <↑|0> and a down TF is indicated as <↓ |1>. These faults are shown in Figure 2.

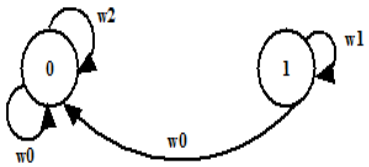


Figure 2. State diagram for up transition fault

Coupling Fault (CF): The cells present in CF are in coupled state. Here the transition of the cell j which causes change in the memory cell i. Due to coupling with other cells the real behavior is changed. CF can be of two types. They are

A. Inversion Coupling Fault

This faults occurs in two possible ways in the memory cells *i*,j. It is denoted as $cfinv_{i,j}$.

Rise: <↑|↕> (If the cell *j* varies from 0 to 1 then the output of cell *i* is complemented)

Fall: <↓|↕> (If the cell *j* varies from 1 to 0 then the output of cell *i* is complemented)

B. Idempotent Coupling Fault

In this fault $cfid_{ij}$ say, involving cells *i* and *j*, transition is done from 0 to1 or vice versa. Based on the value of cell *j* , the value of memory cell *i* can be decided.

There are four possible cases involving cells *i, j* are

Rise 0: <↑|0> (If the cell *j* varies from 0 to 1 then the output of cell *i* is 0)

Rise 1: <↑|1> (If the cell *j* varies from 0 to 1 then the output of cell *i* is 1)

Fall 0: <↓|0> (If the cell *j* varies from 1 to 0 then the output of cell *i* is 0)

Fall 1: <↓|1> (If the cell *j* varies from 1 to 0 then the output of cell *i* is 1)

Bridging fault (BF): These faults occur in two or more memory cells as short circuit.

Neighbourhood Pattern Sensitive Fault (NPSF): In this fault, the content of cell i is manipulated by the content of neighbour cells. Here content can also changed by using read and write operations.

Here the cells are near to each other. So, the cells may get influenced by neighbourhood cells. For example, the value of cell i is zero and all the other cells have value as 1, then automatically exists a changed in the cell i to 1.

In fault modeling there are two types of neighborhoods used for the cell under test.

- Type-1 neighborhood:

The red color cell can be tested and the 4 neighborhood cells are colored in yellow. The neighborhood cells may cause faults in cell under test. Figure 3describes about fault pattern and example of this is describes in Figure 4.

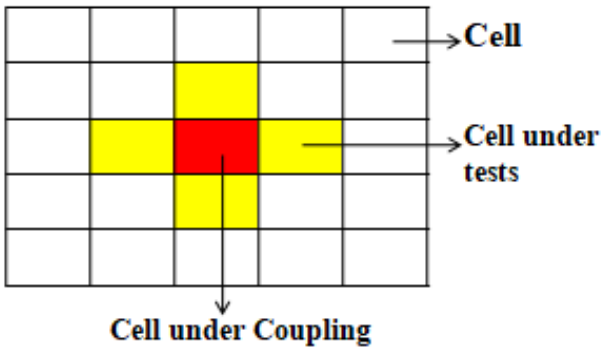


Figure 3. Pattern of Type-1 Neighborhood Faults

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

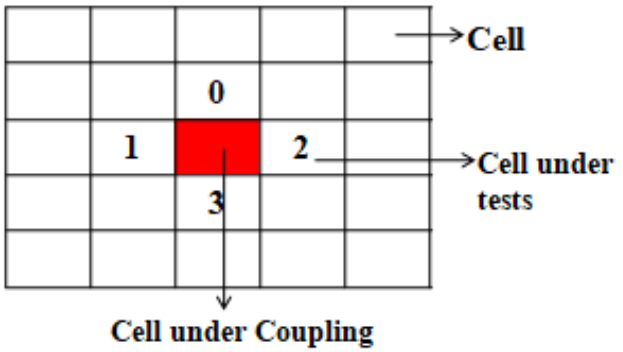


Figure 4. Example for Type-1 Neighborhood Faults

➢ Type-2 neighborhood:

In this cell under test is surrounded by eight neighborhood cells. These faults are more complex fault than type-1 neighborhood. Figure 5 describes about pattern of fault and example of this is describes in Figure 6.

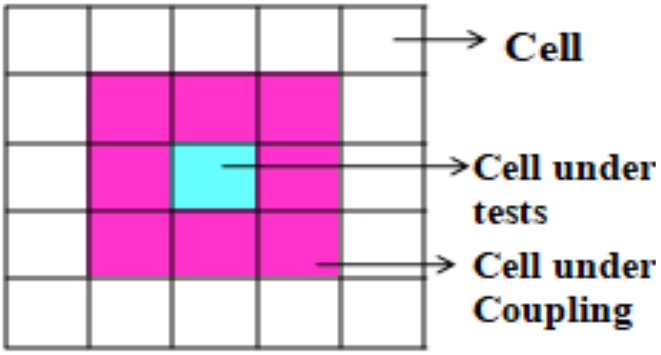


Figure 5. Pattern of Type-2 Neighborhood Faults

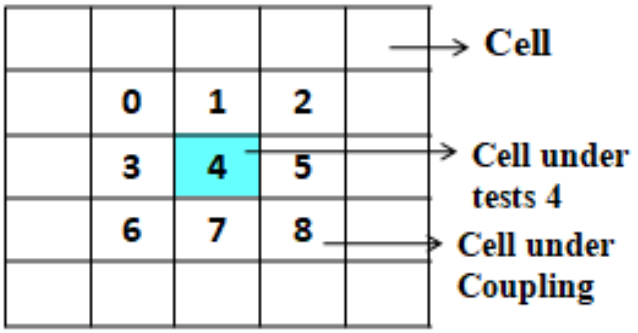


Figure 6. Example for Type-2 Neighborhood Faults

There are two types of NPSFs which are mainly used in fault modeling of memories.

➢ Active NPSF (ANPSF): When the anyone of the neighborhood cell value is changed then the cell under test value changes.

➢ Passive NPSF (PNPSF):

In this changing of values are prevented because of cell under test.

Address decoder faults: In this four types of faults are present in both read and writing.

- No cell is accessed for a certain address.
- No address can access a certain cell.
- For a certain address no cell can be accessed.
- The certain cell has no address can be accessed.
- By using a particular address, multiple cells can be accessed simultaneously.

III. ERRORS DETECTION AND CORRECTION (EDC)

EDC is of two types. One is Single error detection and correction and other is Multiple error detection and correction [3].

In single EDC is used to locate and correct one error bit. This single bit error can also detect and correct by some error correcting codes. The codes of single error is classified as two types of algorithms. One is Classical or Tradtional algorithm and another one is March algorithm . In multiple errror detection and correction is of two different types. One is adjacent and other random. Both have different codes for detectong and correcting multiple bits [5].

IV. SINGLE ERROR DETECTION AND CORRECTION

If the value of bit varies then the error occurs in single bit. For correcting the error, it is required to know which bit is error bit. So, that bits are modified reversely by using receiver [4].

Number of redundancy bits needed

- Let data bits = m
- Redundancy bits = r

$\therefore$ Total message sent = $m+r$

The value of r should satisfy below relation:

$$2^r \geq m+r+1$$

A. Classical or Traditional Algorithm

Classical algorithms are of various types like checkerboard, walking 1/0, GALPAT, Surround disturb, WALPAT, Moving inversion(MOVI), GALROW, GALCOL, Sliding Diag, Butterfly. Among this zero-one and checker board are best algorithm and has a drawback of less fault coverage. Remaining algorithms are complex and slow but good fault coverage.

Checkerboard Algorithm:

This algorithm consists of zeros and ones as checkerboard pattern. It has complexity of 4N. This algorithm creates a physical pattern, but not logical pattern. This algorithm detects only SAF'S, DRF (Data Retention Fault), shorts between cells and half of Transition Faults (TF'S). It is not good for addressing faults and some coupling faults (CF'S) because it cannot be detected.

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

1	0	1
0	1	0
1	0	1

Figure 7. Pattern of Checkerboard

In this checkerboard pattern means that at any point of time it complicate one particular template of this March sequence. Figure 7 describes pattern in the memory.

Butterfly Algorithm:

This algorithm can detect all SAFs and some AFs. It is has complexity of 5N log N. Figure 8 describes Pattern of Butterfly

1. Write background 0;
2. For BC = 0 to N-1;
   {Complement BC; dist=1;
   While dist <= mdist /*mdist <0.5 col/row length*/
   {Read cell @ dist north from BC;
   Read cell @ dist north from BC;
   Read cell @ dist north from BC;
   Read cell @ dist north from BC;
   Read BC; dist* = 2;}
   Complement BC;}
3. Write background 1; repeat step2;

		6			
		1			
9	4	5,10	2	7	
		3			
		8			

Figure 8. Pattern of Butterfly

Surround Disturb Algorithm:

This algorithm examine how the cells in rows are changed when complementary data are written in adjacent cells. This algorithm is shown in Figure 9.

1. For each cell [p,q] /*row p and column q*/
   { write 0 in cell [p,q-1];
   write 0 in cell [p,q];
   write 0 in cell [p,q+1];
   write 1 in cell [p-1,q];
   read 0 in cell [p,q+1];
   write 1 in cell [p+1,q];
   read 0 in cell [p,q-1];
   read 0 in cell [p,q];}
2. Repeat step 1 with complementary data;

		1		
	0	0	0	
		1		

Figure 9. Pattern of Surround Disturb

Sliding Galloping Row/ Column/Diagonal:

It is similar to galloping patterns test. It has complexity of $4n^{3/2}$. It is done by shifting one diagonally. Memory entirely is read after every shift. It has same fault coverage as GALPAT except some CF's. Sliding Galloping is shown in Figure 10.

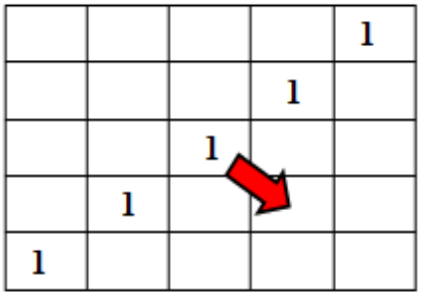


Figure 10. Pattern of Sliding Galloping

Sliding Galloping Row / Column / Diagonal:
pseudocode for implementation

```
{
write background to 0;
for (BC = 0; BC <= (BC − 1); BC + +)
{
   complement diagonal BC's;
   for (OC = 0; OC <= (OC − 1); OC + +)
   {
   read and verify diagonal BC's;
   read and verify other BC's;
   }
   complement diagonal BC's;
}
write 1 to background;
repeat same procedure;
end;
```

Figure 11. Pseudo code of Sliding Galloping

GALPAT:

GALPAT is said to be Galloping patterns test. It has $4n^2$ complexity. Most faults can be easily tested in this pattern. It can detect and correct all SAFs, TFs, AFs and CFs in a complete test. Figure 11 describes about pattern of GALPAT.

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

1. Write background '0'.
2. For BC = 0 to n-1
   {
   complement BC;
   for OC = 0 to N-1,OC!=BC;}
   {Read BC; Read OC;}
   Complement BC;}
3. Write background 1;
4. Repeat step 2.

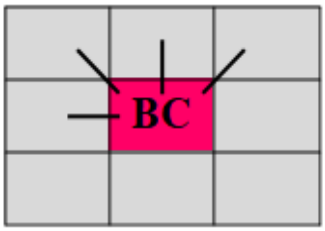


Figure 12. Pattern of GALPAT

$$
\begin{aligned}
&GALPAT:\\
&pseudocode\ for\ implementation\\
&write\ 0\ to\ background\\
&for(BC=0;BC<=(BC-1);BC++)\\
&\{\quad m(i)=\overline{m(i)};\\
&\quad for(OC=0;OC<=(OC-1);OC++)\\
&\quad \{\\
&\quad\quad read\ and\ verify\ BC;\\
&\quad\quad read\ and\ verify\ OC;\\
&\quad \}\\
&\quad\ m(i)=\overline{m(i)};\\
&\}\\
&write\ 1\ to\ background;\\
&repeat\ \ same\ procedure;\\
&end
\end{aligned}
$$

Figure 13. Pseudocode for GALPAT

Moving Inversion (MOVI):

It performs functional and AC parametric test. It has functional complexity of 13N and parametric complexity 12 N log N. It can detect faults like AF, SAF, TF and most CF.

WALPAT:

WALPAT is known as Walking Pattern Algorithm. It has complexity of $2N^2$. It is same as GALPAT, except the Base cell (BC) reads only after all other cells performs read. In this algorithm first read BC then all cells which are excluding BC. Again read BC and repeat the same process. Compliment the BC and again repeat same process. WALPAT complexity is less than GALPAT.

$$
\begin{aligned}
&WALPAT:\\
&pseudocode\ for\ implementation:\\
&write\ 0\ to\ background\\
&for(BC=0;BC<=(BC-1);BC++)\\
&\{\quad m(i)=\overline{m(i)};\\
&\quad for(OC=0;OC<=(OC-1);OC++)\\
&\quad \{\\
&\quad\quad read\ and\ verify\ OC;\\
&\quad \}\\
&\quad\ read\ and\ verify\ BC;\\
&\quad\ m(i)=\overline{m(i)};\\
&\}
\end{aligned}
$$

Figure 14 .Pseudo code for WALPAT

Drawbacks of Classical algorithm:

- Checker board algorithm has low fault coverage.
- Utilization time is more.
- These are of high cost because it requires long time for testing the algorithms.
- The range of complexity is from N^2 to N log N.

B. March Algorithm

It consists of March elements with fixed order. Every bit performs finite number of read or write operation. A bit is known by order of bit address [6] [7].

March Test Notations:

$\Uparrow$: addresss sequence changes in ascending order

$\Downarrow$: address sequence changes in descending order

$\Updownarrow$: address sequence can change either way

$r0$: read operation (reading a 0 from a cell)

$r1$: read operation (reading a 1 from a cell)

$w0$: write operation (writing a 0 to a cell)

$w1$: write operation (writing a 1 to a cell)

N indicates the number of bits in the array memory.

Example: MATS+

- Test consists of 3 March elements: M0, M1 and M2.
- Address order of M0 is irrelevant that is denoted by $\Updownarrow$

M0: $\Updownarrow$ (w0) means 'for i=0 to n-1 do A[i] =0'
M1: $\Uparrow$ (r0,w1) means 'for i=0 to n-1 do {read A[i]; A[i]:=1}'
M2: $\Downarrow$ (r1,w0) means 'for i=n-1 to 0 do {read A[i]; A[i]:=0}'

Table 1.Comparision of different parameters of March algorithms:

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

Algorithm		Description	Types of Faults Detected	Complexity	Redundancy
MSCAN		{⇕ (w0); ⇕ (r0); ⇕ (w1); ⇕ (r1)}	SAF, some AF'S	4N	Irredundant
MATS	Or type	{⇕ (w0); ⇕ (r0,w1); ⇕ (r1)}	SAF, some AF'S	4N	Irredundant
	And type	{⇕ (w0); ⇕ (r1,w0); ⇕ (r1)}			
MATS+		{(w0); ⇑ (r0,w1); ⇓(r1,w0)}	AF, SAF	5N	Irredundant
MARCHING 1/0		{⇑ (w0); ⇑ (r0,w1,r1); ⇓(r1,w0,r0); ⇑(w1); ⇑(r1,w0,r0);⇓(r0,w1,r1)}	AF, SAF, TF	14N	Redundant
MATS++		{⇕ (w0); ⇑ (r0,w1); ⇓(r1,w0,r0)}	AF, SAF, TF	6N	Irredundant
MARCH X		{⇕ (w0); ⇑ (r0,w1); ⇓(r1,w0); (r0)}	AF, SAF, TF, CF'S	6N	Irredundant
MARCH Y		{⇕ (w0); ⇑ (r0,w1,r1); ⇓(r1,w0,r0); (r0)}	AF, SAF, TF, CF	8N	Irredundant
MARCH C		{⇕ (w0); ⇑ (r0,w1); ⇓(r1,w0); ⇕ (r0); ⇓(r0,w1);⇓(r1,w0); ⇕ (r0)}	AF, SAF, TF, CF	15N	Redundant
MARCH C-		{⇕ (w0); ⇑ (r0,w1); ⇓(r1,w0);⇓(r0,w1); ⇓(r1,w0);(r0)}	AF, SAF, TF, CF	10N	Irredundant
MARCH A		{⇕ (w0); ⇑ (r0,w1,w0,w1); ⇓(r1,w0,w1); ⇓(r1w0,w1,w0);⇓(r0,w1,w0)}	AF, SAF, TF, CF	15N	Irredundant
MARCH B		{⇕ (w0); ⇑ (r0,w1,r1,w0,r0,w1); ⇓(r1,w0,w1); ⇓(r1w0,w1,w0);⇓(r0,w1,w0)}	AF, SAF, TF, CF	17N	Irredundant
MARCH CW		{⇕ (wa, wa', ra', wa, ra)}	AF, SAF, TF, CF	(10+5 log w)N	Redundant
MARCH LR		{⇕ (w0); ⇓ (r0,w1); ⇑(r1,w0,r0,w1); ⇑ (r1,w0); ⇑(r0,w1,r1,w0); ⇑(r0)}	AF, SAF, TF, CF	14N	Redundant

V. MULTIPLE ERROR DETECTION AND CORRECTION

Here the data is varied by two or more non-consecutive bits. If parity bit has one or more incorrect, but the overall parity bit is correct, then more than one errors are detected. It is classified into two types. One is random and other is adjacent [8].

Adjacent Codes: Hamming code, matrix codes, and Latin square code are examples of this.

Latin Square: The order n is an n x n array in which n distinct symbols are arranged and there will be no repetition of bits in the row and column [9].

Example: 012 012
120 201
201 120

Orthogonal Latin Squares: It consist of two distinct Latin squares A = (a_{ij}) and B = (b_{ij}) are orthogonal then the n x n ordered pairs (a_{ij},b_{ij}) are all distinct [10]. Example of this is shown below.

Example:

$$A = \begin{pmatrix} 0\,1\,2 \\ 1\,2\,0 \\ 2\,0\,1 \end{pmatrix} \quad B = \begin{pmatrix} 0\,1\,2 \\ 2\,0\,1 \\ 1\,2\,0 \end{pmatrix} \quad \begin{pmatrix} (0,0)\,(1,1)\,(2,2) \\ (1,2)\,(2,0)\,(0,1) \\ (2,1)\,(0,2)\,(1,0) \end{pmatrix}$$

It has advantages like analysis of data is simple and analysis is simple even with missing plots. It is limited by a factor to the number of replicates which seldom exceeds 10.

Hamming Codes: It is a binary linear cyclic codes. Hamming distance is the no. of bits need to be change from one bit to another. Its distance for minimum code is (d_{min}) = 3 and for extended code distance is (d_{min}) = 4. It detects double bit and corrects single bit [11].

- Code length: $n - 2^r - 1$
- Number of information symbols: $k - 2^r - r - 1$
- Number of parity checks symbols: $n - k = r$
- Error correcting capability : $t = 1 (d_{min} = 3)$

Extended Hamming Code: The H-matrix can be obtained by check matrix of a Hamming code by adding a zero at left column and a ones row at the bottom [12]. The extended code matrix is shown below.

$$\left[\begin{array}{c|cccccccc} 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ \hline 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{array}\right]$$

H-Matrix of (8,4) Hamming code

Application for Hamming matrix are DRAMs, satellite communication and other gaming applications.

Random Codes: BCH codes, Reed-Solomon, Golay, TURBO are examples of this codes.

Reed Solomon (RS) Codes: It is a linear cyclic code with non-binary block. It operates the data by dividing message stream into blocks, by redundancy is mainly dependent on present inputs [13].

It is generally represented as

- Block length : n
- Number of Original message symbols : k
- Number of Parity Digits : $n-k=2t$
- The relationship between the symbol size is m and the codeword size is n, is given by $n=2^m - 1$

It has advantages like it is mainly used in multiple errors. Coding gain is very high and less when compare with LDPC and TURBO codes.

BCH codes: This code is invented by Bose-Chaudhuri-Hocquenghem (BCH). These are cyclic error-correcting codes that are constructed using polynomials over a finite field [14]. If $m \geq 3$ is a positive integers and t, a(n, k) is obtained as:

- Block length : $n=2^m -1$
- Number of check bits : $n-k=<mt$
- Minimum distance: $d_{min}>=2t+1$
- $t < (2-1)/2$ random errors detected and corrected. So it is also called 't-error correcting BCH code'.

It has flexible block length and code rate is one of the advantage for BCH codes. It is widely used and low amount of redundancy. It has disadvantages like iterative and complex decoding algorithm [15].

VI. CONCLUSION

The detection and correction of single and multiple bits are used in System-on-Chip Designs, there are many effective approaches. By choosing an appropriate algorithm, the specific fault which has manifested as error can be located and corrected.

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

The choice of algorithm depends on many factors like reliability, redundancy, latency, area overhead, etc. The random errors or burst errors require more memory and computational resources, hence while choosing an algorithm for them, the best choice will be a trade off between the parameters considered.

REFERENCES

1 Balwinderr Singh Lakha, Sukhleen Bindra Narang and Arun Khosla, "Modeling and Simulation of Efficient March Algorithm for Memory Testing", Contemporary Computing - Third International Conference, pp.96-107, August 2010.
2 Joseph, P Elsa and P Rony Antomy, "VLSI Design and Comparative Analysis of Memory BIST controllers", First International Conference on computational Systems and Communications (ICCSC), pp.272-276, December 2014.
3 Manoj S, C Babu, "Improved Error Detection and Correction for Memory Reliability against Multiple Cell Upsets using DMC &PMC", IEEE, 2016.
4 Kavya B S, "Survey on Error Correction and Detection Schemes for Memory Applications", International Journal of Scientific & Engineering Research, Vol. 6, February 2015.
5 J Manikandan and DR. M.Manikandan, "Design of Single Error Correction – Double Adjacent Error Detection – Triple Adjacent Error Detection – Tetra Adjacent Error Detection (SEC – DAED – TAED – Tetra AED) Codes", International Journal of Applied EngineeringReasearch, Vol. 11, pp. 4440-4444, November 2016.
6 Harutunyan G.,Vardanian V. A., Zorian Y., "Minimal March Tests for Unliked Static Faults in Random Access Memories", in proceeding of VLSI Test Symposium, pp. 53-59, 2005.
7 Er. Manoj Arora, Er. Shipra Tripathi, "Comparative Simulaton of MBIST using March Test Algorithms", International Journal of Scientific & Engineering Research, Vol. 2, December2011.
8 R. Naseer and J. Draper, "Dec ECC Design to Improve Memory Reliability in sub-100nm Technologies", IEEE, ICECS, pp. 586-589.
9 Pedro Revirigeo, Salvatore Pontarelli, Adrian Evans, and Juan Antonio Maestro, "A Class of SEC – DEC - DAEC Codes Derived From orthogonal Latin Square Codes", IEEE Transactionon Very Large Scale Integration (VLSI) Systems, 2014.
10 Shanshan Liu, Pedro Reviriego, Liyi Xiao and Juan Antonio Maestro, "Reducing the Cost of Triple Adjacent Error Correction in Double Error Correction Orthogonal Latin Square Codes", IEEE Transactions on Device and Materials Reliablity, Volume :16, June 2016.
11 Hamming R. W. M., " Error Detecting and Error Correcting Codes", Bell System Tech.Jour., 29 (1950), pp. 147-160.
12 Hamming SEC - DAED and Extended Hamming SEC – DEC - TAED codes through Selective Shortening and Bit Placement", IEEE Transactions on Device and Materials Reliability, Volume 14, pp.574-576, March 2014.
13 J. Chen, P. Owsley, "A Brust Error Correcting Algorithm for Reed Solomon codes", IEEE Transaction, vol. 64, pp. 1497-1500, May 2015.
14 Mohammed Hasan Alwan, Mandeep Singh, Hussain Falih Mahdi, "Performace Comparsion of Turbo Codes with LDPC Codes and with BCH Codes for Forward Error Correcting Codes", IEEE Student Conference on Research and Development (SCOReD), December 2015.
15 E. Mytsko, A. Malchukov, I. Novogilov and V. Kim, "Fast decoder of BCH code with cyclic decoding method", International Siberian Conference Control and Communication (SIBCON), IEEE Conferences, 2016.

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)