BIST Design For Static Neighbourhood Pattern Sensitive Fault Test

Anu Samanta
Dept. of Electronics and comm. Engg.
NSHM, Durgapur
Durgapur

Mousumi Saha
Dept. of Computer Application
NIT, Durgapur
Durgapur

Ajay Kumar Mahato
Dept of Electronics and Comm. Engg. NIT, Sikkim, Sikkim

ABSTRACT
Traditional tests for memories are based on conventional fault models, involving the address decoder, individual memory cells and a limited coupling between them. Built-in self-test (BIST) solutions for testing memories today incorporate hardware for test pattern generation and application for a variety of the algorithms. The NPSF fault model is recognized as a high quality fault model for memory arrays, the excessive test algorithm time cost associated with it, compared to other fault models, restricts its adoption for memory testing. These faults are of different classes and types. More specifically, active, passive and static faults for distance 1 and 2 neighborhoods, of types 1 and 2, are considered. This paper presents a BIST implementation using cellular automata (CA) for detection of static neighborhood pattern sensitive faults (SNPSFs) in random access memories (RAMs).

1. INTRODUCTION
Testing semiconductor RAMs has become of increasing im portance lately. This is due to high density of current memory chips and to the fact that older algorithms required test times of the order n² or n*log$_2$ n (where n is the number of bits in chip). At the same time, due to miniaturization, the types of faults became more complex and therefore more difficult to find, where as the test time had to become of the order n in order to economically acceptable.

Built-in self-test (BIST) methods for testing RAMs, based on conventional March tests and their extensions, are becoming popular. These tests are easy to implement, have a cycle count complexity which is linear to the number of bits or words addressed, and provide good fault coverage for functional faults and some structural faults, covering the address decoder, individual memory cells and a limited coupling between them. Existing BIST implementations for memories are, however, inadequate for some other fault models like stronger and widespread coupling faults [3]. Hence to ensure the desired defect coverage in a memory core, it is necessary to consider these newer fault models and provide a matching BIST implementation for effective test generation. It is also important to make this implementation programmable for the desired combination of fault coverage and test time so that BIST can be efficiently used.

This paper describes a BIST technique for the detection of neighborhoods pattern sensitive faults (NPSFs) in random access memories. Although the NPSF model is not new, it is now becoming important in deep-submicron processes, especially for DRAMs. Traditional March tests are not adequate for detection of such NPSFs. Here a test pattern of Hamiltonian sequence from 3 bit Eulerian graph is generated for static neighborhood pattern sensitive fault (SNPSF) test and also implemented in BIST using 4-neighborhood Cellular Automata (CA) to obtain shortest possible test.

.

2. PRILIMINARIES

2.1 Neighborhoods Pattern Sensitive Faults

A Pattern Sensitive Fault is a conditional coupling fault in which the content of a memory cell, or the ability to change its content, is influenced by a certain bit pattern in other cells in the memory. Here the data retention and transition of the victim cell are affected by a set of aggressor cells. A neighborhood pattern sensitive Fault (NPSF) is a special case of pattern sensitive faults, wherein the influencing (coupling) cells are in the neighborhood of the influenced (coupled) cell [1]. The coupled cell is called the base (or victim) cell and the coupling cells are called the deleted neighborhood cells. The neighborhood includes all the cells in the deleted neighborhood as well as the base cell.

2.1.1 Classification of NPSFs
Different NPSFs can be grouped based on the nature of faults in the base cell and on the neighborhood.

2.1.1.1 Active NPSF: The base cell changes its contents due to changes in the deleted neighborhood pattern. To detect these faults, each cell must be read in state 0 and in state1 for all possible transitions in the deleted neighborhood pattern. There are two different possible values for the base cell (0 and 1), k-1 ways of choosing the deleted neighborhood cell which must undergo one of two possible transitions (↑ or ↓), and 2^{k-2} possibilities for the remaining neighborhood cell contents. The total number of active neighborhood patterns (ANPs) is $2^*(k-1)^*2^*2^{k-2} = (k-1)^*2^k$ [2]. In Figure1. for type-1 active NPSF 2- base cell; 0,1,3 and 4 are deleted neighborhood cells.

Notation: $C_{i,j}$ <d_0, d_1, d_3, d_4; b >

Examples: ANPSF: $C_{i,j}$ < 0, ↓, 1, 1; 0 >

ANPSF: $C_{i,j}$ < 0, ↓, 1, 1; ↕ >

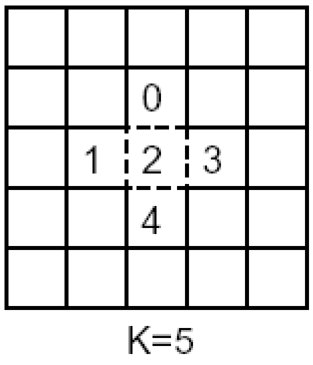


Figure 1. Type-1 Active NPSF

*2.1.1.2 Passive NPSF***:** The contents of the base cell cannot be changed due to a certain neighborhood pattern. Each cell must be written and read in state0 and in state1 for all permutations of the deleted neighborhood pattern. For each of the 2^{k-1} deleted neighborhood patterns, the two possible transitions ↑ and ↓ must be verified. Therefore, the total number of PNPSFs is $2*2^{k-1}=2^k$. The total pattern count for APNPSFs is therefore, $(k-1)*2^k+2^k=k*2^k$.

2.1.1.3 Static NPSF: The content of a base cell is forced to a certain state due to a certain neighborhood pattern. To detect these faults, apply the 2^k combinations of 0s and 1s to the k-cell neighborhood, and verify by reading each cell that each pattern can be stored. It differs from ANPSF that it need not have a transition to sensitize an SNPSF.

Example: $C_{i,j}$ < 0, 1, 0, 1; -/ 0 > means that base cell forced to 0

$C_{i,j}$ < 0, 1, 0, 1; -/ 1 > means that base cell forced to 1

2.1.2 Existing test methods for NPSFs

There are several methods and algorithms to perform tests for Neighborhood Pattern Sensitive Faults (NPSFs). A type 1 neighborhood contains five cells: The base cell and the four cells physically adjacent to the base cell. Usually the type 1 neighborhood is used because the deleted neighborhood of the type 1 neighborhood is most likely to influence the base cell (since all neighborhood share a row or a column with the base cell) and because of its simplicity and smaller test time.

2.1.2.1 Eulerian and Hamiltonian sequences:

For optimal write sequences, it is essential to minimize the number of writes during NPSF testing. An Eulerian graph has a node for each k-bit pattern of 0s and 1s and there is an arc between two nodes, if and only if they differ by exactly one bit [2]. When two nodes are connected, they are connected by only two arcs; depicted in Figure 6. The arcs in the graph correspond to ANPs, PNPs and APNPs of a k-bit neighborhood. An Eulerian sequence traverses each arc in the graph exactly once while a Hamiltonian sequence traverses each node in the graph exactly once. ANPSFs, PNPSFs and APNPSFs are tested with an Eulerian sequence. A Hamiltonian sequence is used for writing during SNPSF tests. Patterns in a k-bit Hamiltonian sequence differ by only 1 bit from their preceding pattern as this minimizes the number of writes needed to generate the patterns.

2.1.2.2 Tiling method:

The tiling method totally covers memory with non-overlapping neighborhoods. Figure-2a and 2b depicts this for a 5-element Type-1 neighborhood. Cell 2 is always the base cell, and the deleted neighborhood cells are numbered shown. When all static neighborhood patterns (SNPs) are applied simultaneously to the neighborhoods of all base cells-2, they are automatically applied to the neighborhoods of all base cells in the memory [1]. This reduces the pattern length from $n*2^k$ patterns to $(n/k)*2^k$ patterns.

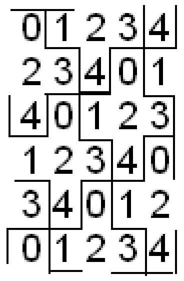


Figure 2a. Type 1 tiling neighborhoods

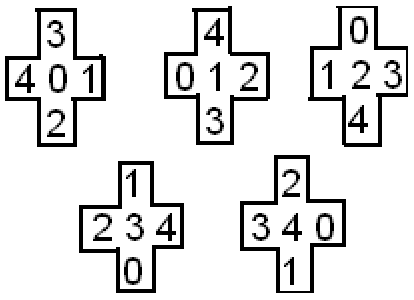


Figure 2b. Neighborhoods of base cells 0,1,2,3, and 4

2.1.2.3 Two-group method:

For the two-group method, a cell is simultaneously a base cell in one group and a deleted neighborhood cell in the other group, and vice versa [2]. With this duality property, cells are divided into two groups, group-1 and group-2, in a checkerboard pattern; depicted in Figure-3. Base cells of group-1 are deleted neighborhood cells of group-2, and vice versa. Each group has n/2 base cells b and n/2 deleted neighborhood cells formed by 4 subgroups A, B, C and D. This only works for Type-1 neighborhoods.

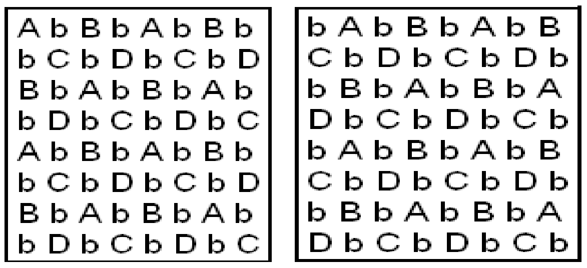


Figure 3. Cell labels in two group method

2.1.2.4 NPSF fault detection and location algorithm:

```
Step 1: write base cells with 0;

Step-2: loop

            Apply a pattern;

                    {it could change the base cell from 0 to 1}

             Read base cell;

         Endloop;

Step 3: write base cells with 1;

Step 4: loop

           Apply a pattern;

                   {it could change the base cell from 1 to 0}

            Read base cell;

        Endloop;
```

2.1.3 BIST for detection of NPSFs:

Built-in self-test (BIST) is a design technique in which parts of a circuit are used to test the circuit itself. MBIST (Memory BIST) is a technique specifically used for testing memories. It typically consists of tests circuits that apply, read and compare tests pattern design to expose defects in the memory device [5]. One of the MBIST algorithms is March algorithm. The advantage of such memory design modification is often offset by the overheads that they introduce. The main goal of these approaches is to reduce the memory testing time.

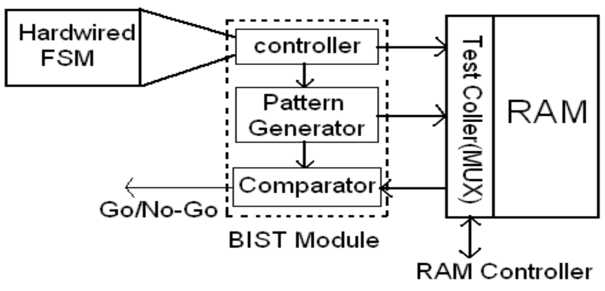


Figure 4. Hardwired-based BIST

A memory BIST unit consists of a controller to control the flow of test sequences and other components to generate the necessary test control and data. A hardwired-based BIST controller is a hardware realization of a selected memory test algorithm, usually in the form of a Finite State Machine (FSM) [4]. This type of memory BIST architecture has optimum logic overhead; however, this results in re-design and re-implementation of the hardwired-based memory BIST for any minor changes for the selected memory test algorithm. Although it is the oldest memory BIST scheme amongst the BIST, hardwired-based BIST is still much in use and techniques have been kept developing. The hardwired- based BIST is shown in Figure 4.

For implementation of BIST, linear feedback shift register (LFSR) is the shift register with feedback linearly related to the nodes using xor gates and cellular automata (CA) is a collection of nodes logically related to their neighbors using xor gates. LFSRs are more compact and simple to design, but CAs provides patterns with higher randomness. In applications LFSR is used to reduce the area overhead but CAs are implemented where high fault coverage is needed.

2.2 Cellular Automata (CA):

Cellular Automata (CA) is introduced to identify the memory with the faulty cells, simultaneously satisfying the requirements of reduced test time as well as the overhead of test logic. Each cell stores a discrete variable at time t that refers to the present state (PS) of the cell. The next state (NS) of the cell at (t+1) is affected by its state and the states of its neighbors at time t [5]. In 3-neighborhood CA (self, left and right neighbors), where a CA cell is having two states 0 or 1 and the next state of ith CA cell is $S_i^{t+1} = f_i (S_{i-1}^t, S_i^t, S_{i+1}^t)$

S_{i-1}^t, S_i^t and S_{i+1}^t are the present states of the left neighbor, self and right neighbor of the ith cell at time t and f_i is the next state function. The next state function of the ith CA cell can be expressed in the form of a truth table (Table 1). The decimal equivalent of the 8 outputs is called Rule R_i. In a 2-state 3-neighbourhood CA, there can be 2^8 (256) rules. Two such rules 192 and 207 are illustrated in Table-1. The first row lists the possible 2^3 (8) combinations of present states of (i-1)th, ith and (i+1)th cells at t. The 3rd and 5th rows indicate the next states of the ith cell at (t+1). The mathematical expression of next state function can be obtained when the next state of a rule is plotted in a Karnaugh Map in Figure 5.

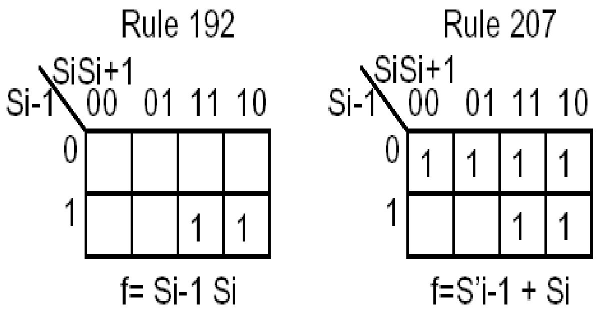
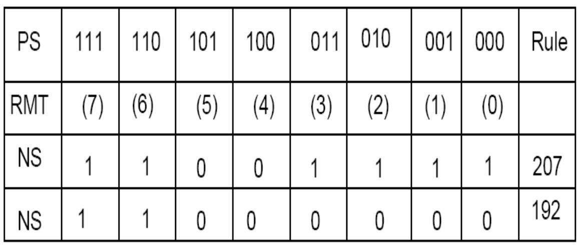


Figure 5. Next state function for rule 192 and 207

Table -1: RMTs of the CA<207,192>



PS	111	110	101	100	011	010	001	000	Rule
RMT	(7)	(6)	(5)	(4)	(3)	(2)	(1)	(0)	
NS	1	1	0	0	1	1	1	1	207
NS	1	1	0	0	0	0	0	0	192

3. PROPOSED WORK

A Hamiltonian sequence is used for writing during SNPSF tests. The Hamiltonian sequence from the Eulerian graph is following: [111 101 001 000 100 110 010 011 111]. We have applied this Hamiltonian sequence or bit pattern using 4-neighborhood CA to detect the SNPSFs.

Table 2. RMTs upto 7of the CA < 39321, 46064, 52464>

PS	0111	0110	0101	0100	0011	0010	0001	0000	Rule
RMT	(7)	(6)	(5)	(4)	(3)	(2)	(1)	(0)	
NS	1	0	0	1	1	0	0	1	39321
NS	1	1	1	1	0	0	0	0	46060
NS	1	1	1	1	0	0	0	0	52464

The next state function of the i^th CA cell can be expressed in the form of a truth table (Table 2 and 3). The decimal equivalent of the 16 outputs is called Rule R_i . In a 2-state 4-neighbourhood CA, there can be 2^{16} (65536) rules. Three such rules 39321, 46064 and 52464 those required to realize the stated Hamiltonian sequence or bit pattern to detect the SNPSFs [2], are illustrated in Table 2 and 3. The first row lists the possible 2^4 (16) combination of present states of 4 cells at t [5]. The 3rd 4th and 5th rows indicate the next states of the i^th cell at (t+1). The mathematical expression of next state function is obtained when the next state of the rule is plotted in a Karnaugh Map (Shown in Figure 7). A circuit diagram is implemented by using those Boolean expressions and shown in Figure 8. The simulation result and RTL schematic of Hamiltonian sequence for SNPSF test is shown in Figure 9 and 10 respectively.

Table 3. RMTs from 7 to 15 of the CA < 39321, 46064, 52464 >

PS	1111	1110	1101	1100	1011	1010	1001	1000	Rule
RMT	(15)	(14)	(13)	(12)	(11)	(10)	(9)	(8)	
NS	1	0	0	1	1	0	0	1	39321
NS	1	0	1	1	0	0	1	1	46060
NS	1	1	0	0	1	1	0	0	52464

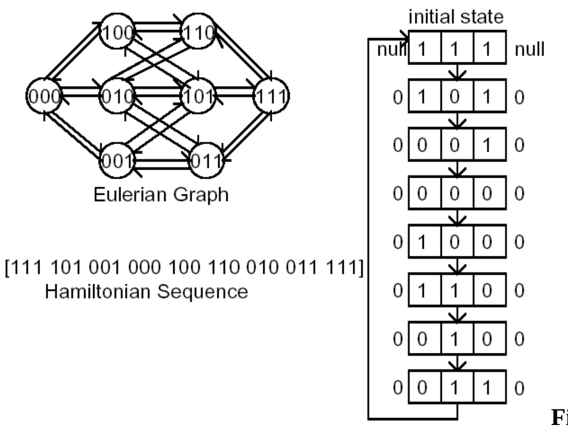


Figure 6. State transition diagram of the Hamiltonian sequence

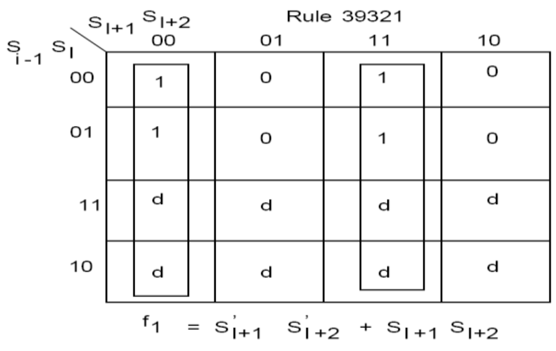


Figure 7. Next state function for rule 39321

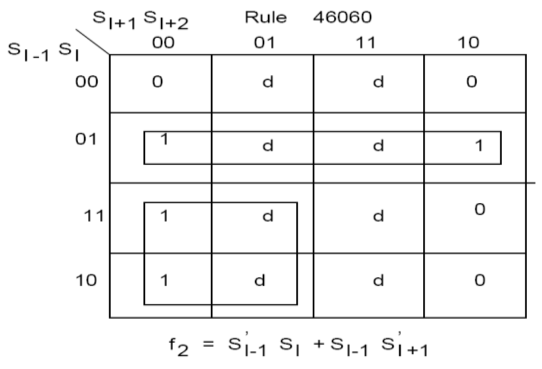
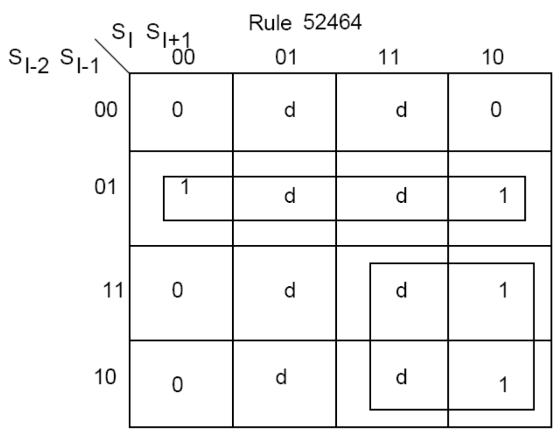


Figure 8. Next state function for rule 46064



Rule 52464

$S_{l-2}S_{l-1}$ \ S_lS_{l+1}	00	01	11	10
00	0	d	d	0
01	1	d	d	1
11	0	d	d	1
10	0	d	d	1

$$f_3 = S'_{l-2}\,S_{l-1} + S_{l-2}\,S_l$$

Figure 9. Next state function for rule 52464

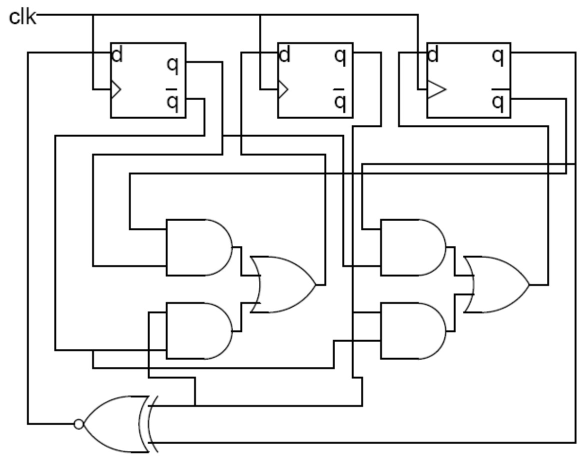


Figure 10. Circuit diagram of Hamiltonian sequence

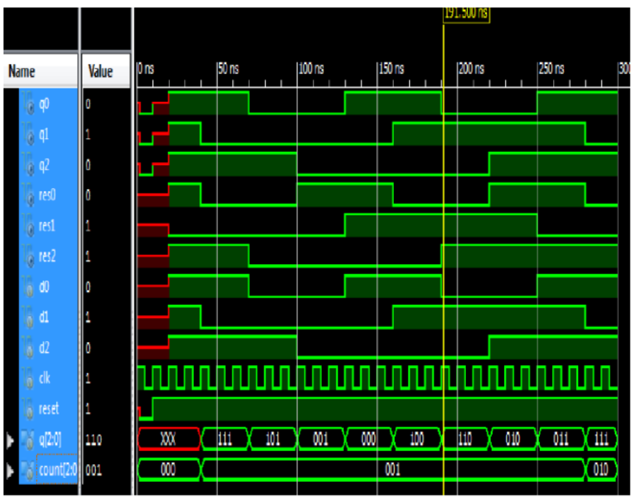


Figure 11. Waveform of Hamiltonian sequence

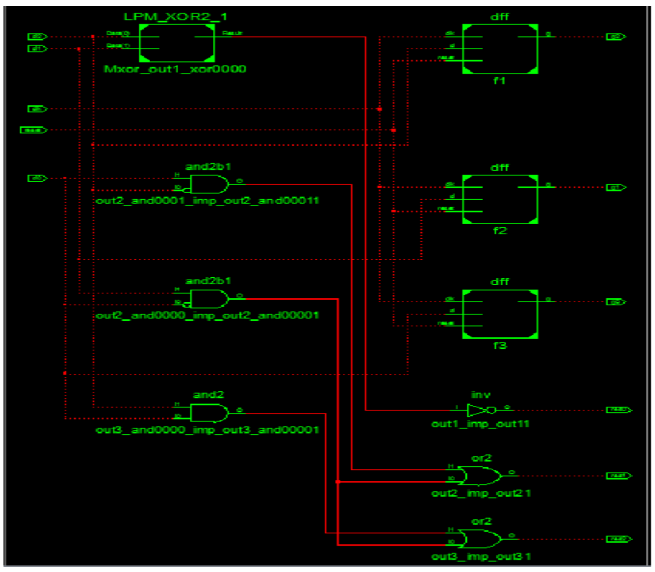


Figure 12. RTL schematic of Hamiltonian sequence

4. CONCLUTION AND FUTURE WORK

We have described the fault model of bit oriented memory (BOM) and test algorithms for that in details. There are so many test pattern generator (TPG) to design efficient Memory Built-in self-test (MBIST). We have shown test patterns using Cellular Automata (CA) to implement in Built-in self-test (BIST). Here a Hamiltonian sequence is implemented in a BIST circuit using 4-neighborhood CA for Static neighborhood pattern sensitive fault (SNPSF) tests in bit oriented memory (BOM). This is essential to minimize the number of writes during NPSF testing, in order to obtain the shortest possible tests. We have also done the verilog coding to generate the test pattern in MBIST for SNPSF tests. The waveform of the verilog coding and RTL schematic is also shown. The future work in this direction includes the efficient algorithm of CA based BIST design for neighborhood pattern sensitive fault (NPSF) detection for word oriented memory (WOM) and also the length of 24 optimal 3 bit Eulerian sequence with the help of Cellular Automata (CA) to implement in Memory Built-in Self-test (MBIST) for Active and passive neighborhood pattern sensitive fault (APNPSF) tests.

5. REFERENCES:

[1] A. J. VAN DE GOOR and C. A.VERRUIJT. "An Overview of Deterministic Functional RAM Chip Testing". ACM Computing Surveys, Vol. 22, No.1, March 1990.

[2] V. D. A. Michael Lee Bushnell, Vishwani D. Agrawal, "Essentials of Electronic Testing for Digital, Memory, and Mixed -Signal VLSI Circuits". New York: Kluwer Academic Publishers, 2nd ed., 2002

[3] Rajeshwar S. Sable, Ravindra P. Saraf, Rubin A. Parekhji and Arun N. Chandorkar. "Built-in Self-test Technique for Selective Detection of Neighbourhood Pattern Sensitive Faults in Memories" Proceedings of the 17 th International Conference on VLSI Design (VLSID 04) 1063-9667/04 $ 20.00@2004 IEEE.

[4] Allen C. Cheng."Comprehensive Study in Designing Memory BIST Algorithms, Implementations and Trade offs" EECS 579, Fall 2002 Digital System Testing.

[5] S.Wolfram, "Cellular Automata and Complexity," collected papers, pp. 1-25, 1994.

[6] M. A. Miron Abramovici, Melvin A. Breuer, Arthur D. Friedman, "Digital Systems Testing and Testable Design"Jaico Publishing House, 2002.