TESTING OF MEMORY USING FRANKLIN METHOD

Mythri Myneni[1], Madhavi Mallam[2]

[1,2]*Assistant professor, DVR & Dr. Hima Sekhar, MIC college of Technology, kanchikacherla*
mythri.ece07@gmail.com and madhavi.418@gmail.com

Abstract

It is natural to be skeptical when someone says it is possible to determine the state-of-health of a complex integrated circuit with a few simple tests. Justifiable doubt in the unfamiliar can be aided by experience and investment in a traditional approach [5]. So only the process known as testing is involved after each and every design. Fault may occur even in the memory .So we are going to test the memory in this paper by the most suitable method [4]. Therefore we are going to test the memory using one of the pattern sensitivity test method i.e., Franklin method by Verilog approach [6]. The results obtained are in excellent agreement with theoretical results.

Keywords: *Memory, RAM, Pattern sensitivity test, Franklin Method, Verilog approach.*

----------******----------

1. INTRODUCTION

The use of redundancy to increase the yield of random access memories (RAM's) has become very popular. State-of-the-art memory chips are now invariably designed with spare rows and spare columns. Once a memory chip is reconfigured, physically adjacent cells in the spare rows and columns that were brought in are very unlikely to have consecutive logical addresses. Test algorithms used at the stages for the detection of physical neighbourhood faults in these RAM has to consider that the logical and physical neighbourhoods of some of the cells are not identical' [2]. A simple solution to this problem might be to store the actual logical-to- physical address mapping of the relevant memory cells for later recall. In an external tester environment, the tester has to derive the address mapping either by using post repair diagnostic techniques such as *signature and roll call* [SI or by using *Stroboscopic Scanning Electron Microscopes (SSEM)* modify the test algorithm according to the mapping, and then apply the test patterns[1]. In a Built-In Self-Test (BIST) environment, the BIST logic either has to be modified to incorporate the reconfiguration done to the RAM array, or must have the means to determine the mapping. Either of these solutions increases the complexity of the BIST logic[2]. To decrease the complexity of the design we are using the Franklin and Saluja Method.

2. RAM CELL

Before embarking on the testing problem, we shall briefly review the organization of RAM's (in particular, reconfigurable RAM's), as it plays a major role in determining the relationship between physical and logical neighbourhoods. A RAM chip consists of memory cell arrays, address decoders, address registers, data registers, and read write logic [2]. The memory cells are physically organized as a 2 x k 2-D grid of identical m x n 2-D arrays of cells, as shown in Figure.2.1

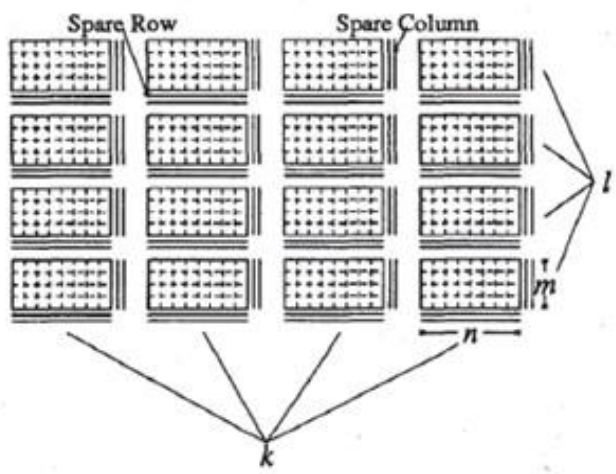


Fig: 2.1: RAM cell

This type of partitioning is done to keep the active power, peak currents, access times, and cycle times within reasonable limits. By far, the most popular approach to incorporate redundancy in RAM's is to add spare rows (word lines), spare columns (bit lines), or both for each memory array. In some RAM's, a spare row or column can be shared by two or more arrays. In some others, rows or columns cannot be replaced singly; instead, they must be replaced in pairs, quads, or larger groups. Interested readers are referred to [2], for further details of different redundancy schemes and reconfiguration algorithms. When a RAM chip is physically organized as a collection of

smaller arrays, cells and their contents in each of these physical arrays are independent of the cells in the remaining arrays. By assuming that no interaction can take place between cells of different physical arrays, we need consider only a 2- D physical array for modeling faults, allowing each array tobe tested independently from the rest of the chip. If spares are shared among a group of physical arrays, then each such group of arrays must be tested as a single unit to detect faults due to interactions between physically adjacent spare rows (and columns)[2].

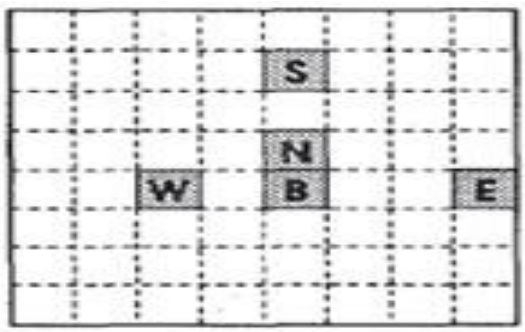


Fig: 2.2 : RAM

2.1. RAM Memory Cell:

As RAM density increases, PSF's become the predominant faults [9]. Moreover, other memory fault classes such as shorts, stuck-at faults, and coupling faults can be regarded as special types of PSF's. Because testing for unrestricted PSF's is impractical, researchers have considered restricted neighbourhoods, which assume interactions to be limited to cells within certain physical proximity. These are called physical neighbourhood PSF's (PNPSFs). The most frequently considered neighbourhood is the five-cell physical neighborhood, in which the four physical neighbors of a cell are often called N (North), E (East), W (West), and S (South).

Static PSF (SPSF): In this fault model, a cell is said to be faulty if its contents change when a certain pattern of 0's and -1's exists in the neighborhood cells.

Dynamic PSF (DPSF): In this fault model, a cell is said to be faulty if its state changes because of a change in its neighbourhood pattern [2].

2.2. Partitioning and Testing of 8×8 RAM array

To apply the patterns, namely the patterns in which the N and S physical neighbors of the cell have distinct values, we consider all possible pairs of rows in the array as the N-S neighbors, and initialize the rows in each pair to distinct values. For identifying all pairs of rows, we use the following partitioning technique. In a series of steps, successively partition the set of rows into two; the partitions at any step can be determined by a specific bit position in the row address [6]. Thus, at the ith iteration, if the ith bit position of a row's address is 0, then that row belongs to the first partition of the RAM array for that step; if the bit is 1, the row belongs to the second partition. The different figures in Figure 2.1 show the partitioning steps for an 8 x 8 RAM array. This partitioning technique for identifying all possible pairs of rows is similar to the one presented for partitioning memory cells. After each partitioning step, initialize the rows in the first partition to 0 and the rows in the second partition To apply the remaining patterns, namely the patterns in which the N and S physical neighbors of the cell have distinct values, we consider all possible pairs of rows in the array as the N-S neighbors, and initialize the rows in each pair to distinct values [2]. For identifying all pairs of rows, we use the following partitioning technique [4].

3. FRANKLIN METHOD

3.1 Pattern Sensitivity Faults

Testing is a process which includes test pattern generation, test pattern application, and output evaluation. The capacity of random-access memories chips enhances, thus increasing the test time and cost; on the other hand, the density of memory circuits grows, therefore more failure modes and faults need to be taken into account in order to obtain a good quality product. he quality of a test set depends on its fault coverage (FC) as well as its size. With the increase in memory density, neighbourhood pattern sensitive faults (NPSFs) are not only an important fault model for DRAMs but will also become so for SRAMs. RAM errors will become more of a problem as memory sizes grow and device geometry shrinks With the growing in memory density and the shrinking into design rule, neighbourhood pattern sensitive faults (NPSFs) are becoming more and more important. Though previously proposed patterns for NPSFs have sufficient fault coverage, the complexity of those requires long test time. These are common in high density RAMs. Before designing a test it's wise to consider exactly what kinds of failures may occur. Contents of memory cell are affected by contents of neighbouring cells. Common in memory cells of high density RAMs. Detected with specific memory test algorithms.

3.2 Franklin and Saluja algorithm:

Franklin and Saluja in presented algorithms that test reconfigured RAM's and scrambled address RAM's for five-cell and nine-cell physical neighbourhood pattern sensitive faults (PSF's)[2]. Those algorithms require O (N [10g3N14) reads and writes to test an N-bit RAM array. Furthermore, those algorithms are not simple enough to

allow easy BIST implementations. We can easily understand the Franklin Method by the following procedure.

Assume the RAM memory cell as portioned rows and columns as follows:

Columns 0 1 2 3
Rows
0 1 1 1 1
1 1 1 1 1
2 1 1 1 1
3 1 1 1 1

First change the 3_{rd} column

Columns 0 1 2 3
Rows
0 1 1 1 0
1 1 1 1 0
2 1 1 1 0
3 1 1 1 0

Then change the 0_{th} row

Columns 0 1 2 3
Rows
0 0 0 0 1
1 1 1 1 0
2 1 1 1 0
3 1 1 1 0

Then change the 2_{nd} column

Columns 0 1 2 3
Rows
0 0 0 1 1
1 1 1 0 0
2 1 1 0 0
3 1 1 0 0

Then change the 1_{st} row

Columns 0 1 2 3
Rows
0 0 0 1 1
1 0 0 1 1
2 1 1 0 0
3 1 1 0 0

Then change the 1_{st} column

Columns 0 1 2 3
Rows
0 0 0 1 1
1 1 1 0 0
2 1 1 0 0
3 1 1 0 0

Continue this till 0_{th} column so after 3^{rd} row

We will get the answer as follows:

Columns 0 1 2 3
Rows
0 0 0 0 1
1 0 0 0 1
2 0 0 0 1
3 0 0 0 1

Like this we need to continue till 1st column and changing all the rows [4].

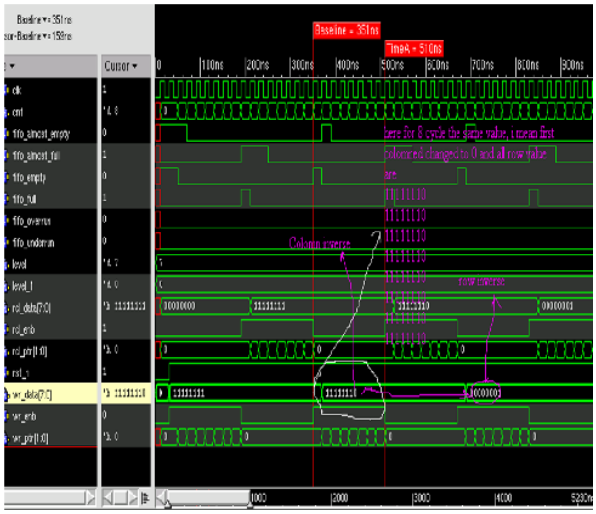
And finally if we change the 0th column we will get the same answer.

i.e.
Columns 0 1 2 3
Rows
0 1 1 1 1
1 1 1 1 1
2 1 1 1 1
3 1 1 1 1

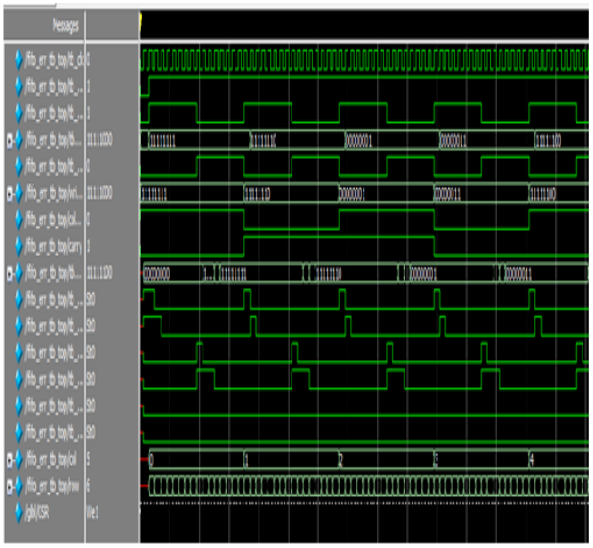
Suppose if there is any fault in any of the row or column it may be 0.So by observing the rows and coloumns we can say where the fault occurs.This is one of the way to identify the fault by using Franklin and Saluja method.

4. RESULTS

4.1.RAM(FIFO) Without Fault:



4.2. RAM (FIFO) With Fault:



CONCLUSIONS

In this work, we have succeeded in generating Structural test data from HDL functional descriptions of a memory cell (RAM). The proposed technique is based on a software testing technique: Franklin and Saluja Method [4].High low-level fault coverage can be achieved with short high-level test sequences but further investigations should improve the current results. Even if our study was initially limited to a given type of memory cell, the first results obtained, whatever the number of fault sequences or the fault coverage, encourage us to apply our approach to other types of memory cells. We can eliminate the maximum number of faults being occurred in the device as the memory is tested.

REFERENCES

[1] Richard A. DeMillo, Aditya P. Mathur, and W.Eric Wong. “Some Critical Remarks on a Hierarchy of Fault-Detecting Abilities of Test Methods” IEEE TRANSACTIONS ON SOFTWARE ENGINEERING,VOL. 21, NO. 10, OCTOBER 1995.

[2] Manoj Franklin and Kewal K.Saluja.“Reconfigured RAM’s and Scrambled RAM’s for Pattern Sensitive Faults.”

[3] Michael J. Liebelt “Detecting Exitory Stuck-At Faults in Semi modular asynchronous Circuits”.
IEEE TRANSACTIONS ON COMPUTERS, VOL.48, NO.4,APRIL 1999.

[4] Parag K. Lala. “An Introduction to Logic Circuit Testing”.Editor: Mitchell Thornton,Southern Methodist University.Morgan & Claypool Publishers,2009.

[5] M. Abramovici, M.A. Breuer, and A.D. Friedman,Digital Systems Testing and Testable Design, IEEE,1990.

[6] R.Leveugle. “Fault Injection in VHDL Descriptions and Emulation”. TIMA Laboratory.