Algorithm for Testing NAND Flash Random Access Memory

G. Lokheshwara Guptha, M. Anil Kumar

Abstract: This paper presents an overview of the problem of testing semiconductor random access memories (RAM's). An important aspect of this test procedure is the detection of permanent faults that cause the memory to function incorrectly. Functional-level fault models are very useful for describing a wide variety of RAM faults. Several Fault models are discussed throughout the paper, including the stuck-at-0/1 faults, coupled-cell faults presented and their fault coverage and execution times are discussed.

Keywords: Algorithms, Reliability, Checking experiments, Fault detection

I. INTRODUCTION

Recent developments in semiconductor memory technology have greatly increased the use of semiconductor random access memories (RAMs). These memories are usually large-scale-integration (LSI) devices and employ either bipolar or metal oxide semiconductor (MOS) techniques. To detect the variety of faults that occur within a memory chip, extensive testing of semiconductor memories is carried out by both users and manufacturers. As a result of the large number of cells involved, most of the standard fault detection techniques are not suitable for memory testing.

A RAM chip consists mainly of a memory cells array, an address decoder, a memory address register (MAR), a memory data register (MDR), and read/write logic (sense amplifiers, write drivers, etc.). The general organization of a semiconductor RAM is given in Figure 1.

Physical faults can occur in any part of the memory chip. The causes of failure in RAMs, though not yet fully understood, are known to depend on such factors as component density, circuit layout, and method of manufacture. In this paper, we limit ourselves to the fault-detection, rather than the fault-location, problem. In this way, testing for faults that have similar effects on the functional behavior of the memory system can be simplified by testing for one of these faults.

We intend to survey the different method for detecting functional faults in semiconductor RAMs and to compare these methods in terms of their time efficiency and fault coverage. We begin in Section 1 by describing the three most commonly used fault models for semiconductor RAMs, concentrating on the stuck-at faults model, the coupling faults model, and the pattern-sensitive faults model.

Manuscript received April, 2013.

G. Lokheshwara Guptha, Department of Electronics and Communication Engineering, K L University, India.

M. Anil Kumar, Asst Professor, Department of Electronics and Communication Engineering, K L University, India.

II. MEMORY FAULT MODELS

A wide variety of internal faults can occur in semiconductor memories, causing various types of failures in the memory function. Test procedures necessary to detect these faults can be classified-as in any digital circuit test-into three classes: DC parametric testing, AC parametric testing and functional testing

DC parametric testing, as the name suggests, measures the DC parameters in the semiconductor RAMs. It checks for unacceptable output level, high power consumption, fan-out capability, noise margins, rise and fall times in electrical signals, etc. Clearly this kind of testing depends on the implementation details of the RAM under test, and usually the DC parametric test is supplied by the manufacturer. Because DC parametric testing is so implementation dependent, we do not discuss it further in this paper.

AC parametric testing, or dynamic testing, measures AC parameters of the RAM to detect any malfunction. The parameters measured include memory access time, setup time and hold time. Examples of malfunctions that can be detected by AC testing include [Breuer and Friedman 1976]

(1) Slow write-recovery: the memory may not produce the correct information at the specified access time when each read cycle is preceded by a write cycle.

(2) Sleeping sickness: memory loses information in less than specified hold time.[1]

Although AC parametric test procedures are not considered in this paper, functional test procedures detect many AC malfunctions. For example, the stuck-at faults tests, when applied to the memory at the maximum possible rate, may detect slow write-recovery faults.

Fictional testing is designed to detect permanent faults that cause the memory to function incorrectly.[2] Generally speaking, a read/write RAM can be defined as functional if it is possible to store a 0 or 1 into every cell of the memory, to change every cell from 0 to 1 as well as from 1 to 0, and to read every cell correctly when it stores a 0 as well as when it stores a 1, regardless of the contents of the remaining cells or the previous memory access sequences.[1]

[1] Metal oxide semiconductor (MOS) dynamic memory store their data via the charge across a capacitor, because of the leakage this charge must be refreshed at fixed intervals or the value in the cell will be lost. The maximum period of the time that data can be stored without refreshing is called the hold time.

[2] The contrast between software program testing and functional testing of digital systems is worth nothing. In program testing, the main concern is achieving an acceptable assurance that the programmer has put together his primitives (i.e. language constructs) correctly. Although functional testing of digital systems is similarly concerned with whether hardware primitives have been put together

correctly, it is further concerned with the integrity of the primitives themselves. Moreover, while human factors and software complexity make program testing largely an art at present, the relative simplicity of digital systems (memories, in particular) makes functional testing more methodological. The program testing work that is most analogues to functional testing is that of complier validation [Hoyt 1977]. An area of special interest is test data selection techniques.

A functional test that will cover all the possible faults is impossible from the practical point of view, since the complexity of such a test will be on the order of 2^n, where n is the number of cells in the memory (we have to check every cell for all possible states of the remaining cells). For example, assuming an access time of ~500 nanoseconds, testing a 1 Kbit RAM would take approximately 10^{293} seconds. Therefore, I order to develop any practically feasible test procedure, we should restrict ourselves to a subset of faults that are most likely to occur. This practice is known as "selecting a fault model."

The following three fault models for RAMs are the most widely used ones (listed according to their degree of complexity): [2]

1. *Stuck-at Faults Model*: One or more logic values in the memory system cannot be changed. For example, one or more cells are "stuck at" 1 or 0. Stuck-at faults are also useful for modeling faults in other parts of the memory system (see Figure-1).
2. *Coupling Faults Model*: There exist two or more cells that are coupled. A pair of memory cells, i and j, are said to be coupled if a transition from x to y in one cell of the pair, say cell I, changes the state of the other cell, that is, cell j, from 0 to 1 or from 1 to 0. This, of course, does not necessarily imply that a similar transition in cell j will influence cell I in a similar manner.
3. *Pattern-Sensitivity Faults (PSF) Model*: the fault that alters the state of a memory cell as a result of certain patterns of zeros, ones, zero-to-one transitions, and/or one-to-zero transitions in the other cells of the memory is called a PSF. Also, the fault that causes a read or a write operation of one cell of the memory to fail owing to certain patterns of zeros and ones in the other cells of the memory is called a PSF. General pattern-sensitivity tests are infeasible in practice. However, some restricted, and realistic, models of PSFs allow for the generation of efficient test sequences. It should be noted that coupling faults are a special case of PSF.

There are many test procedures for RAMs. These procedures can be partitioned into three classes, according to the fault model that each procedure adopts. Procedures for testing stuck-at faults are simple and fast. Procedures for testing coupling faults are more complicated and usually require more time, but they detect all stuck-at faults, as well as all coupling faults. Procedures for testing pattern-sensitivity faults are very complicated and they only consider some restricted case of PSFs. In the rest of the paper we describe different representatives from each class of procedure, their time complexity, and their degree of fault coverage.

III. TESTING FOR STUCK-AT FAULTS

This section describes testing algorithms intended to detect stuck-at faults in read/write semiconductor RAMs. This kind of fault is independent of the memory state and can occur in any part of the memory system. Thus we have to consider stuck-at faults in the memory data register, memory address register, address decoder, the memory cells array, and the read/write logic. Note that faults detection is our main concern, not fault location, since the whole RAM is assumed to be on one chip. [3]

The MSCAN Test Procedure:

This test writes each word, first with a 0 and then with a 1, and verifies the writing with two read statements.

The above test procedure will not detect any stuck-at fault in the memory address register or in the decoder. But on the assumption that those two are fault free, the test can detect any stuck-at fault in the array, in the memory data register, or in the read/write logic. Of course, such a test is of little value because at the end of this procedure we can only be sure that there is at least one word I the memory that is functionally correct. The complexity of the MSCAN test is 4n.

IV. TESTING FOR COUPLING FAULTS

An important type of fault that can cause a semiconductor RAM to function incorrectly is the cell's coupling fault. Such faults can occur as a result of a short circuit or a circuit's parasitic effects such as current leakage or stray capacitances.

In this section we discuss procedure that adopts the coupling fault model. A common feature is that they cover most of the stuck-at faults in the RAM while testing for coupling faults.

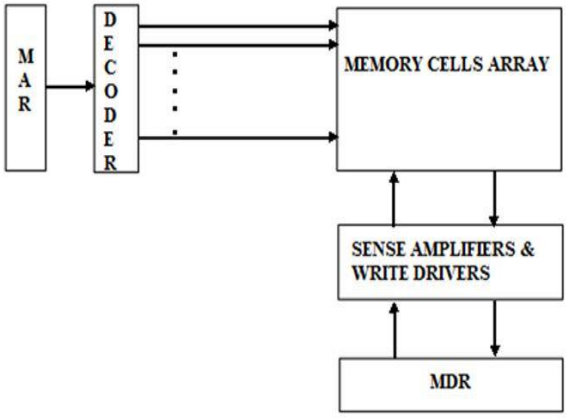

Figure-1: A semiconductor RAM organization at the functional level of description

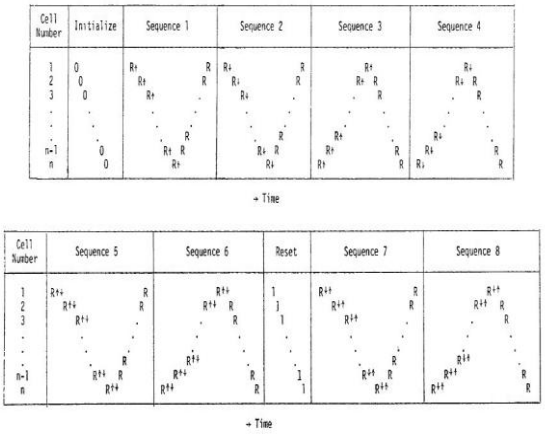

Figure 2: Nair, Thatte, and Abraham's testing procedure [Nair et al. 1978].

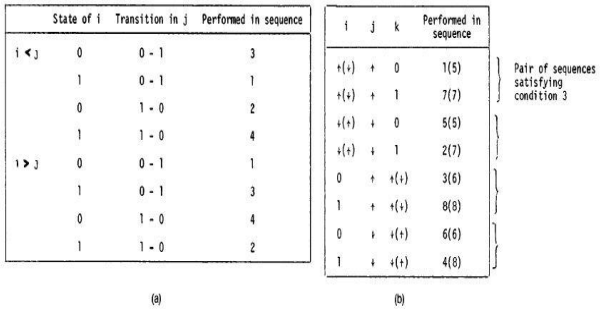

Figure 3: (a) Satisfying Condition 2, (b) Satisfying Condition 3.

It is very important to note that when considering the coupling fault model, memory bits cannot be treated as being functionally independent, as they could in the stuck-at fault model. Therefore in the rest of our discussion we shall assume that the memory under test is an array of n cells (bits) rather than an array of multiple-bit words.

Various test procedures such as Column Bars, Marching 1's and 0's, and Galloping 1's and 0's have been proposed and used in the industry to test for different classes of coupling faults and stuck-at faults. The complexity of these test procedures ranges from O(n) to $O(n^2)$.

They vary in complexity mainly because they vary in the amount of fault coverage they provide. However, more powerful test procedures have been proposed in the literature with complexities $O(n \log_2 n)$, and even O(n). These procedures provide more fault coverage than the ones used commercially with comparable or faster execution times.

Column Bars Test Procedure:

Assume that the RAM contains n accessible cells and is organized as an array of two dimensions, such that the even addresses are in the even columns and the odd addresses are in the odd columns. The test procedure is given as

Step 1: Write 1's in all even addresses and 0's in all odd addresses.

Step 2: Read all the memory cells

Step 3: Repeat steps 1 and 2, interchanging 1's and 0's.

The procedure is designed to test for possible coupling between any two cells in the adjacent columns. It is not very effective, though, because it only guarantees that there are two cells in the whole memory that are not coupled and that they do exist. This limitation results from the possibility that a stuck-at fault could have occurred in the memory decoder, causing all the even addresses to be mapped to one cell and all the odd addresses to be mapped to the other cell. Such a fault will not be detected by the Column Bars test. Even if we were to assume that the decoder is fault free, the Column bars test would not detect some of the coupling faults between cells in the same column. The complexity of this procedure is 4n, but it is of limited value because it does not cover the decoder or the MAR stuck-at faults and it also fails to cover many coupling faults in the RAM.

V. MORE EFFECTIVE PROCEDURES FOR THE DETECTION OF COUPLING AND STUCK-AT FAULTS

Thatte and Abraham [1977] considered a functional fault model for semiconductor RAMs covering multiple stuck-at faults affecting the various memory subsystems, as well as coupling faults between pairs of memory cells. They also developed a procedure of complexity O(nlog2n) to detect all the faults covered by the fault model. Nair et al. [1978] developed a more efficient testing procedure of complexity O(n) to detect the same type of faults.[4]

Nair, Thatte, and Abraham's Testing Procedure

Nair et al. [1978] considered testing RAMs for both stuck-at faults and coupling faults. Recall from section 1 that stuck-at faults can occur in any place in the RAM, whereas coupling faults occur only between pairs of cells in the memory cell array.

Assuming that the decoder will not change into sequential circuit, any failure in the decoder will make it behave in one of the following ways: [5]

(a) The decoder will jot access the addressed cell. In addition, it may access non addressed cells.

(b) The decoder will access multiple cells including the addressed cell.

The multiple access faults can be viewed as a coupling fault in the memory cell array. Also the no access case can be viewed as a stuck-at fault in the addressed cell. Thus we do not have to consider the decoder faults since it is enough to consider the memory cell array faults. Similarly, some output lines or write driver lines may be stuck-at 0 or 1. These faults can be considered as stuck-at faults in the corresponding memory cells. Also, the data input or output lines may have shorts or coupling between them. This fault can be visualized as coupling between the memories cells that correspond to the coupled data lines. Therefore, in order to test a semiconductor RAM for stuck-at and coupling faults, it is enough to concentrate on the faults in the memory cell array. [6]

On the assumption that the memory consists of n cells, the following is a set of necessary and sufficient conditions for all faults in the memory cell array to be detected by a test sequence.

Condition 1: Each cell must undergo a 0-to-1(↑) transition and a 1-to-0 (↓) transition, and must be read after each transition and before undergoing any subsequent transition. If this condition is not satisfied, a cell "stuck at" 0 or 1 will not be detected.

Condition 2: For every pair of cells (i, j), cell I must be read after cell j makes a transition and before cells i and j further transitions, for all the possible states of cell I (two states), and all possible transitions in cell j (two transitions). If this is not satisfied, a coupling fault between a pair of cells may not be detected.

Condition 3: For every cell triple (i, j, k), if the test makes a transition in cell j from y to y^1 after cell i makes a transition from x to x^1 and before cell k in state z is read , then the test must possess another sequence, where either

- Cell k in state z is read after an x to x^1 transition in cell i and before a y to y^1 transition in cell j, or
- Cell k in state z^1 is read after a y to y^1 transition in cell j ad before an x to x^1 transition in cell i.

If this condition is not satisfied, a coupling fault between cell j and cell k may mask the effect of another coupling fault between cells i and cell k, and prevent the fault from being detected. Note that the pair of faults, which mask the effect of each other, cannot be detected by using the marching 1's and 0's test procedure.

The three conditions stated above have been shown to be necessary and sufficient for a test to detect all the faults in the memory fault model.

A test procedure that can detect all the faults in the fault model and satisfies all the conditions stated above is shown in Figure 2 in the tabular form.

To prove that the above procedure detects all the faults in the fault model, it is enough to show that the procedure satisfies the above three conditions:

(1) Condition 1 is satisfied in sequences 1 and 2 of the procedure.
(2) Condition 2 is satisfied for the two cases, when i<j and when i>j, as shown in Figure 3(a)
(3) Eight cases are sufficient to demonstrate that condition 3 is satisfied; these are shown in Figure 3(b) fori < j< k.

Hence the procedure is a complete test for the memory.

The number of operation required to perform the entire memory test is about 30n, where n is the number of cells in the memory. The above procedure is very efficient when compared with the ones previously considered because of its high fault coverage and its low execution time.

VI. CONCLUDING REMARKS

In this paper we have addressed the problem of testing semiconductor random access memories, which form a large percentage of today's LSI products. An important aspect of testing semiconductor RAMs is the detection of permanent faults the cause the memory to function incorrectly. Functional-level fault models are very useful for describing a wide variety of faults. Several models have been discussed, including the single and multiple stuck-at fault models, the coupling fault model, and the pattern-sensitive fault (PSF) model.

Various procedures exist for memory testing. However, because the faults detected by these procedures from overlapping subsets of the set of all possible faults, a comparison of these procedures efficiency is extremely difficult.

Many functional testing schemes have been proposed and used in the industry, most of them tend to be ad hoc procedures and lack well-defined fault models.

A detailed description of a sample of those test procedures has been given.

We presented a survey of various RAM testing procedures proposed in the literature. Generally, these procedures are more effective than the ones used by the industry, from the viewpoints of the both the time complexity and the degree of fault coverage. As a final note we want to stress the fact that the problem of selecting a good testing procedure for a ram is a trade-off between the time required for applying the test and the degree of fault coverage obtained. Total test time is important because it is not economical to tie up equipment for a long period of time when testing a high-volume, inexpensive product. Test time is warranted, though, because the cost to the customer of using a defective chip could be quite high.

VII. ACKNOWLEDGMENT

It is with immense gratitude that I acknowledge the support and help of my Professor Mr. M. Anil Kumar and also I am indebted to my parents and my classmates for helping me to complete this thesis.

REFERENCES

[1] ABAHIR, M. S., AND REGHBATI, H. K. 1983. LSI testing techniques. IEEE Micro. 3, 1(Feb.), 34-51.

[2] Akers, S. B. 1980. Test generation techniques. IEEE Computer 13, 3 (Mar.), 9-16.

[3] BARRACLAUGH, W., CHAIANG, A.C. L., AND SOHI, W. 1976. Techniques for testing the microcomputer family. Proc. IEEE 64, 6 (JUNE), 943-950.

[4] THATTE, S. M., AND ABRAHAM, J. A. 1977. Testing of semiconductor random access memories. In Proceedings of the 7th Annual International Conference on Fault- Tolerant Computing, pp. 81-87.

[5] NAIR, R., THATTE, S. M., AND ABRAHSM, J.A. 1978. Efficient algorithms for testing semiconductor random-access memories. IEEE Trans Comput. C-27, 6(June), 572-576.

[6] NAIR, R. 1979. Comments on an optimal algorithm for testing stuck-at faults in random access memories. IEEE Trans. Comput. C-28, 3 (Mar), 258-261.