RESEARCH ARTICLE OPEN ACCESS

A Survey on Dram Testing and Its Algorithms

K.Manju Priya[1], M. Menaka[2]
Research Scholar[1], Assistant professor[2]
ECE Department (ME *VLSI DESIGN*)
SVS College of Engineering, and Coimbatore
TamilNadu – India

ABSTRACT
Since the minimum feature size of dynamic RAM has been scaled down, several studies have been carried out to sense the faulty cells. In the field of testing, more appropriate test algorithms are required to detect the faults in the cells. In this paper, various test algorithms such as March tests, word line pulsing technique, large Vds Data Retention Test Pattern, interleaving test algorithm are discussed. These algorithms allow the screening of faults in the cells.
Keywords:- Functional Fault Models (FFM), Fault Primitive (FP), Address decoder Fault (AF), Stuck At Fault(SAF), Coupling Fault(CF), Transition Fault(TF),subthreshold leakage.

I. INTRODUCTION

Functional fault models (FFMs) is the deviation of the observed memory behavior from the functionally specified one, under a given sequence of performed memory operations [1]. Therefore, two basic ingredients are needed to define any FFM: (1) a sequence of performed memory operations, and (2) a list of corresponding deviations in the observed behavior from the expected one. The observed memory behavior that deviates from the expected one is called a faulty behavior or simply a fault, which can be denoted by fault primitive (FP). RAM faults are of two types. They are static faults and dynamic faults. Static fault models consist of static FPs, sensitized by at most a single memory operation. In other words, these are faults that describe an incorrect behavior of either the data stored in a cell, or a single memory operation performed on it. Dynamic faults are faults that are sensitized by performing two or more memory operations on the memory.

DRAM faults have two main causes. Improperly set voltages resulting in voltage dependent faults, and leakage currents resulting in time dependent faults. Faults caused by improper voltages stem from the inability of a memory operation in a defective memory to set the full voltage levels expected at different nodes of the memory, resulting in two different fault modes: 1. improper voltages present within the memory cell, and 2. Improper voltages on the nodes of the peripheral circuits . Improper voltages within the cell cause partial faults, while improper voltages in periphery cause dirty faults. Leakage currents, on the other hand, cause time dependent faults to take place, and depending on the direction of the leakage with respect to the performed operation, either soft faults or transient faults take place due to a supporting or an opposing leakage current, respectively.

In a DRAM, operations are supposed to properly set the voltage levels on different nodes (cells or bit lines) in the memory by charging or discharging the capacitors, corresponding to that node, to a predefined high or low voltage level. In general, however, a voltage across a capacitor may take any value from a continuous range of real voltages. Therefore, operations performed on a defective memory may set improper voltage levels on different memory nodes, and therefore it requires a special sequence of operations to make sure that sets them properly. Inappropriate voltages may cause two types of DRAM faults: partial faults and dirty faults. Partial faults are faults that can only be sensitized when a specific memory operation is successively repeated a number of times, to properly initialize the faulty cell and also to properly sensitize the fault in the cell. The definition indicates that there are two different sorts of partial faults: 1. partial faults during Initialization I, and 2. Partial faults during sensitization A.

Partial faults are due to the fact that DRAM cells represent data as analog voltages in a storage capacitor that is supposed to be fully charged or discharged after a single write operation has been performed on a defect-free cell. Dirty faults assume that after proper initialization or sensitization, the state of the memory (voltages on the BLs, the WLs, or in data buffers) is corrupted, such that subsequent detection is prevented. In order to detect the sensitized fault, further operations must be performed to correct the degraded state of the memory. These faults result from the fact that writes and reads are complex operations, requiring a properly set environment in the memory to function correctly. A number of memory defects may result in disrupting this balanced environment. Time dependent faulty behavior in DRAMs cannot be described using the generic fault space. This is done using a special

classification of faults, based on leakage currents, which divide faults into: hard faults, soft faults and transient faults.

Hard faults are memory faults that do not depend on time in any way, neither for sensitization nor for detection. They are directly sensitized after performing a number of memory operations, without the need to wait for a while for the faulty behavior to take effect [1]. Furthermore, once they are sensitized they remain sensitized unless overwritten, giving unlimited amount of time for their subsequent detection. These faults result from defects in the memory that prevents proper functionality under all circumstances for a specific sequence of memory operations. Soft faults are memory faults sensitized by a sequence of memory operations that only become detectable after some time from their sensitization. These faults have usually been tested for by the addition of a delay within the test, to facilitate the detection of the fault, as it is the case for the data retention fault. Soft faults are caused by writing weak voltages into memory cells that gets depleted by the leakage currents that occurs naturally. Transient faults are memory faults that do not remain sensitized indefinitely, but tend to correct themselves after a period of time. Transient faults are tested by performing all the operations in the fault in back-to-back mode directly after each other, and following them with a detecting read operation directly afterwards. Transient faults are caused by writing weak faulty voltages into memory cells, that soon get corrected by naturally occurring leakage currents.

II. MEMORY TESTING

The exponential increase in the integration density of memory components and the increase in the memory complexity, faulty behavior have made fault analysis and memory testing significantly important. Conventional DRAM testing can be grouped into retention testing and functional testing [2]. Retention testing is a test method that screens leakage-current defects by operating read and write functions containing a particular delay time. In functional testing, March elements that are a finite sequence of read or write operations are applied to a cell in memory before proceeding to the next cell. It is conducted on each memory cell in order to detect the cell-to-cell bridge and coupling noise. A test algorithm is a finite sequence of test elements. A test element consists of number of memory operations.

A. Test requirements

1) ***Fault detection:*** A test that fulfills this requirement should result in a fail when applied on a memory that contains the fault. This is a basic requirement of any memory test designed to test for a specific type of faulty behavior.

2) ***Fault localization:*** A test that fulfills this requirement should be able to identify the specific memory cell (or group of cells) where the fault takes place. This requirement is associated with the need to identify and to repair the failing cells by fusing in order to increase the yield.

3) ***Fault diagnosis:*** A test that fulfills this requirement should be able to indicate the physical root cause behind the observed faulty behavior. This requirement is associated with the need to give instant feedback to the fabrication process regarding probable fabrication causes of observed faults.

B. Test components

1) ***Operation sequence*****:** A test should specify a memory operation sequence of writes and reads to be performed in a specific order on memory cells.

2) ***Data pattern (or data background):*** A test should also specify a pattern of 0s and 1s to be written into and read from accessed memory cells.

3) ***Stresses:*** A memory test should include a specification of different operational conditions or stresses (such as timing, temperature and voltage) that the test is supposed to be performed at [1] -[2].This component is important to test for proper functionality within the parameter range defined by the specifications. Stresses are also important to increase the coverage of a test without increasing its length.

III. TEST ALGORITHM

A test algorithm is defined by the test components. The various test algorithms are March tests, word line pulsing technique, large Vds Data Retention Test Pattern.

A. March tests

In order to verify whether a given memory cell is good, it is necessary to carry out a sequence of write and read operations to the cell. The number of read and write operations and the order of the operations depend on the target fault model. Most commonly used memory test algorithms are March tests, in which there are finite sequences of March elements. A March element is a finite sequence of read (**r**) or writes (**w**) operations applied to a cell in memory before processing the next cell. The address of the next cell can be in either ascending or descending address order. When a test algorithm reads a cell, the response will be either 0 or 1, and they are denoted as r0 and r1, respectively. Similarly, writing a 1 (0) into a cell is denoted as w1 (w0).

TABLE 1

SUMMARY OF NOTATIONS

Symbol	Operation

r	A read operation
w	A write operation
⇧	Up addressing order
⇩	Down addressing order
⇳	Any addressing order

1) ***MATS:*** MATS is modified algorithmic Test Sequence. The algorithm requires 4n operations. It detects some AFs and SAFs.

$$\{⇳(w0);⇧(r0,w1);⇩(r1,w0)\} \quad (1)$$

2) ***MATS+:*** MATS+ algorithm is given by (2). This algorithm requires *5.n* operations and detects all AFs because the conditions of above equation is satisfied. In addition, all SAFs are detected because from each cell a 0 value is read (by the 'r0' operation of first march element) and a 1 value is read by the 'rl' operation of next march element.

$$\{⇳(w0);⇧(r0,w1);⇩(r1,w0,r0)\} \quad (2)$$

3) ***MATS++:*** The algorithm for MATS++ (3) which gives an improved version of MATS+. It detects the SAFs and AFs. In addition it also allows the coverage of TFs.

$$\{⇳(w0);⇧(r0,w1);⇩(r1,w0,r0)\} \quad (3)$$

4) ***Marching-1/0:*** Marching-1/0 algorithm is given by (4). This algorithm performs marching 1 and marching 0. Marching-1 begins by writing a background of 0s, then read and write back complement values (and read again to verify) for all cells (from cell 0 to n-1, and then from cell n-1 to 0), it requires 7n operations. Marching-0 follows exactly the same pattern, with the data reversed .This algorithm detects AF, SAF, and TF but only part of the CFs [3]-[4]. It is a complete test, i.e., all faults that should be detected are covered. It however is a redundant test, because only the first three march elements are necessary.

$$\{⇳(w0);⇧(r0,w1,r1);⇩(r1,w0,r0);$$
$$⇧(w1);⇧(r1,w0,r0);⇩(r0,w1,r1)\} \quad (4)$$

5) ***March X:*** March X algorithm is given by (5). This algorithm detects AFs, SAFs, TFs and some CFs. It requires 6n operations.

$$\{⇳(w0);⇧(r0,w1);⇩(r1,w0);⇳(r0)\} \quad (5)$$

6) ***March Y:*** March Y algorithm is given by (6). March Y requires 8n operations. It detects all SAFs, AFs, TFs and some CFs.

$$\{⇳(w0);⇧(r0,w1,r1);⇩(r1,w0,r0);⇳(r0)\} \quad (6)$$

7) ***March C-:*** March C- algorithm is given by (7), which is an improved version of March C. March C- requires 10n operations and detects all AFs. It detects all SAFs, it also detects all TFs.

$$\{⇳(w0);⇧(r0,w1);⇩(r1,w0);$$
$$⇳(r0);⇩(r0,w1);⇩(r1,w0);⇳(r0)\} \quad (7)$$

The following two steps show that March C- detects all unlinked CFins .The proof that March C- detects all unlinked CFids is similar. Let Ci be coupled to any number of cells with addresses lower than *i* and let Cj be the highest of those cells(j < *i).*

(a) If Ci is less coupled to Cj , then M1 will detect the fault, M1 operate on Cj first, causing an ↑ transition thereafter MI will operate on Ci and a 1 will be read instead of the expected 0 value.

(b) If Ci is coupled to Cj then M2 will detect the fault.

8) ***March A:*** March A algorithm is given by (8). March A requires 15n operations. It detects all SAFs, AFs, TFs and some CFs.

$$\{⇳(w0);⇧(r0,w1,w0,w1);⇧(r1,w0,w1)$$
$$⇩(r1,w0,w1,w0);⇩(r0,w1,w0)\} \quad (8)$$

9) ***March B:*** March B algorithm is given by (9). March B requires 17n operations. It detects all SAFs, AFs, TFs and some CFs.

$$\{⇳(w0);⇧(r0,w1,r1,w0,r0,w1);⇧(r1,w0,w1);$$
$$⇩(r1,w0,w1,w0);⇩(r0,w1,w0)\} \quad (9)$$

TABLE II.

COMPARISON OF MARCH TESTS

March algorithm	**Test length**	**Fault coverage**
MATS	4n	Some AFs, SAFs
MATS+	5n	AFs, SAFs
Marching1/0	14n	AFs, SAFs,TFs
MATS++	6n	AFs, SAFs,TFs
March X	6n	AFs, SAFs,TFs,some CFs

March Y	10n	AFs, SAFs,TFs,some CFs
March C-	15n	AFs, SAFs,TFs,some CFs
March A	8n	AFs, SAFs,TFs,some CFs
March B	17n	AFs, SAFs,TFs,some CFs

But with the downscaling of the device, conventional March testing cannot properly detect the leakage-current defects, and these defects have to be assessed using the time-consuming stress method.

B. Word-Line-Pulsing Technique

To test for leakage-current defects, several studies have attempted to implement retention testing using special techniques. The word-line-pulsing technique has been proposed as a means of detecting weak cells by coupling nearby neighbor word lines [5]. This technique results in an adjustable test stress based on setting the word-line enable time. This method can be used to detect subthreshold leakage-current defects, but when it is used in DRAM, it has to consider stress equality according to the cell location during the stress enable time**.** This technique is based on the realization that the precharged bit line BL coupled through the access transistor to the node of the reference cell carrying a "0" will be gradually discharged by the *Iread* of the reference cell. The discharge rate can be expressed by Eq.(11) and is a function of the *total duration* that the word line of the reference cell has been enabled.

$$\Delta V_{BL} = I_{read} * t_{WL\_REF\_Pw} / C_{BL} \qquad (11)$$

where:
ΔV_{BL} – discharge of the bit line after each enabling of the *WL REF*;
I_{read} – cell read current of the reference cell;
$t_{WL\_REF\_Pw}$ – pulse width of the reference cell word line pulse;
C_{BL} – bit line capacitance.

C. March CRF

The causes of different fault models may coexist and sum their effects. The previous algorithms target only one RF model at time and are effective only above a certain fault threshold. The simultaneous presence of multiple causes of RFs, below the detection threshold, may lead to read malfunctions, due to the cumulative effect. This means that in order to test efficiently the RFs it is useful to consider the combination of all the requirements to cover the different RFs. On the base of the requirements to cover Complex RFs, we use a March like test, named March CRF, which allows the best coverage of read faults in memories, because it takes in account multiple electric causes [6].

M0 M1

$$\Uparrow (w1)_i(w0)_{all-\{i,i\pm1\}}\,;\ \Uparrow (w0)_{i\pm1}(r1)_i\,;$$
$$\Downarrow (w0)_i(w1)_{all\{i,i\pm1\}}\,;\ \Downarrow (w1)_{i\pm1}(r0)_i \qquad (10)$$

M2 M3

In order to make easier the understanding of March CRF, we perform its first two elements M0 and M1 (the other two elements are processed similarly with opposite data background) in a hypothetical memory with 16 cells (N=32) in an cell array of 4 rows (m=4) and 4 columns (n=4), *i.e.* 4x4 configuration; one read/write operation is performed for each clock cycle. This process of elements M0 and M1 of March CRF is shown in fig.1.

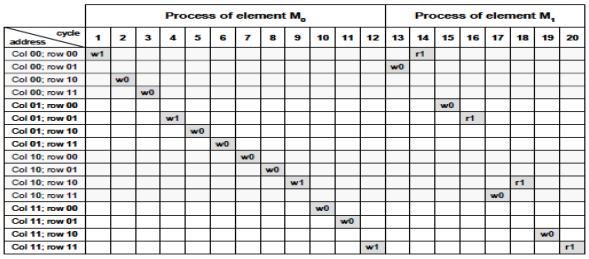

Fig 1. Process of elements of M0 and M1 of March CRF

The element M0 operates a w1 in the ith cell of each column and w0 in all the cells of the column, excluded the ith and the (i±1)th (i+1 for the columns with even address and i-1 for the columns with odd address). The index i={0, 1, 2, …, n}, where n is the number of columns in the memory array, i increases of one unit at the end of each march element process in a column operation. The element M1 operates a w0 in the (i±1)th cell and r1 the ith cell. The following two elements do the same operations inverting the written and read data.

From the process partially shown in the diagram in Figure (elements M0 and M1), we can verify that the March CRF matches all the requirements needed to cover the Complex Read Faults, affecting memories. In fact, in each column the 'X' is written in all the cells but one where '$\bar{X}$' is written and read. For each column, a wX operation is acted immediately followed by an r$\bar{X}$ operation. The requirements are completely accomplished because elements both value of X={0,1} are used.

The complexity of the algorithm March CRF is very low: 2N+2n (~2N), where N is the number of cells and n is the number of columns in the memory.

D. A Large V_{ds} Data Retention Test Pattern for DRAM'S

It describes a test pattern for testing DRAM cell data retention that differs from conventional retention time tests. The test pattern is applicable to non- V_{dd} bit line precharge designs, and is designed to test for worst-case subthreshold leakage through the cell access device. It is achieved by holding bit lines in their latched position for extended periods. This action stresses the cell access devices with the worst case V_{ds} across them[7]. Conventional tests perform cycles in which bit lines latch and then are quickly restored to their equalized state, which is $V_{dd}/2$ for this chip, which has conventional $V_{dd}/2$ sensing. The voltages in the cell after equalization, results in a V_{ds} or $V_{dd}/2$ across the access device. The new large V_{ds} data retention test pattern keeps the bit lines in their latched state for long periods of time, so that a large V_{ds}, equal to V_{dd}, exists across the access device, as shown in Fig.2

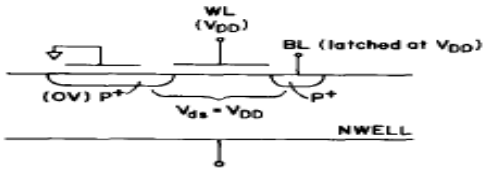

Fig 2. Maximum V_{ds} across access device with V_{dd} latched bit line

In actual chip operation, this access device leakage would be a concern for operating modes that keep bit lines latched for many cycles, such as page-mode operations. This more stressful condition on the access device can detect subthreshold source-drain leakage that could go undetected with conventional data retention patterns. The large source-drain voltage, applied for long intervals at a time, aggravates this leakage in a way that conventional tests do not.

A large V_{ds} data retention test pattern has been found effective in specifically provoking data retention cell fails due to access device leakage. These data retention cell fails can go undetected by conventional data retention test patterns. This test pattern holds bit lines in their latched position for substantially longer than conventional test patterns, thereby subjecting the access devices to a large V_{ds} for extended periods. Only chips with marginal subthreshold characteristics experience fallout to this pattern, underscoring its ability as a screen of device quality. It is applicable to all DRAM designs with sub- V_{dd}, bit-line precharge.

E. Interleaving Test Algorithm

Interleaving test algorithm takes into account the equal bit-line stress regardless of the cell location. This test algorithm allows the screening of weak cells that cannot hold the cell data due to the sub threshold leakage current [8]-[9]. During the stress period, the algorithm can also detect other leakage currents with reduced test time. It utilizes address and data scrambling technique.

1) ***Scrambling:*** The logical structure differs from the physical internal structure of the chip to optimize the memory layout more efficiently [10]. Therefore, logically adjacent addresses may not be physically adjacent. This is called address scrambling. Likewise, data scrambling means that logically adjacent data are not physically adjacent.

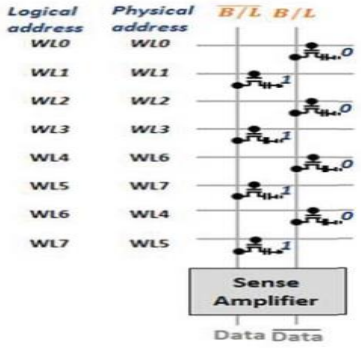

Fig 3. Address and Data scrambling

The above fig shows the twisted address line between the physical and logical addresses and the data status when the value of 0 is transferred to the cells without using scrambling scheme. In this case, the logical address of WL4, 5, 6, and 7 is different from the physical address due to the efficiency of the memory layout. And in case of the memory cell, which is based on the bit line, the inverted data are written into the cells connected to the bit bar line. During test algorithm implementation, different types of data backgrounds are used and the commonly used data backgrounds (DB), are listed below.

1) ***Solid*****:** All cells are filled with "0."
2) ***1-Row Bar*****:** Alternating between "0" and "1," all cells are written in the row direction.
3) ***1-Column Bar*****:** Alternating between "0" and "1," all cells are written in the column direction.
4) ***2-Row Bar*****:** Alternating between a pair of "0" and "1," are written in the row direction. It is essential to write a suitable background data for test algorithms.

2) *2)* ***Concept:*** The concept of the proposed test algorithm is shown in Fig.4. This simplified DRAM array is composed of eight rows and one column, and is implemented using a 2-Row Bar data background with scrambling enabled. Cells of word lines 0, 1, 4, and 5 are stored as "0," and cells of word lines2, 3, 6, and 7 are stored as "1." During the read operation of the first word line, the data stored as "0" is transferred to the bit line through the gate transistor. Then the bit line is pulled down to "0" and the bit bar line is pulled up to "1" by the sense amplifier.

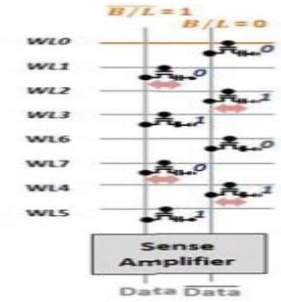

Fig 4. Concept

When the first word line is activated at a specified time, the bit-line cells stored as "1" are stressed during the activated time because of the voltage difference between the cell data and the voltage level of the bit line. The stress of the cells stored at the same voltage level as the bit line can be ignored because that stress is very small compared with that of cells stored at a different voltage level. Cells stored as "0" data at the bit bar line are also stressed by the voltage difference with the bit bar line. If the second word line is activated more than a certain number of times, then the cells stored as opposite data with the bit bar line and bit line are stressed during the activated time. If cells have a defect caused by the threshold leakage-current, then the particular defect-cell data are easily changed to the opposite value during the stress time. In addition, these defects can be detected by setting an appropriate activated time of the word line depending on screenability in order to optimize the test conditions.

IV. CONCLUSION

Thus, most of the conventional algorithms have not considered stress differences among cells caused by cell locations due to the refresh operation of DRAM. Therefore, the quality problems can occur. Thus, a more powerful screening technique is needed to detect subthreshold leakage-current defects. Recently, an interleaving test algorithm has emerged which takes into account the equal bit-line stress regardless of the cell location. Comparatively, this algorithm requires reduced test time.

REFERENCES

[1]. Book on DRAM Fault Analysis and Test Generation(online)

[2]. M.C.-T. Chao, Y. Hao-Yu, H. Rei-Fu, L. Shih-Chin, and C. Ching-Yu, "Fault models for embedded-DRAM macros," in *Proc. Design Autom. Conf.*, Jul. 2009, pp. 714–719.

[3]. AD J. VAN DE GOOR, "Using March Tests to Test SRAMs,"in proc Design and Test of Computers 1993.

[4]. R. Waser, *Nanoelectronics and Information Technology: Advanced Electronic Materials and Novel Devices*. New York, NY, USA: Wiley, 2005,s pp. 527–561.

[5]. A. Pavlov, M. Azimane, J. P. De Gyvez, and M. Sachdev, "Word line pulsing technique for stability fault detection in SRAM cells," in *Proc. Int. Test Conf.*, Nov. 2005, pp. 825–835.

[6]. L. Dilillo and B. M. Al-Hashimi, "March CRF: An efficient test for complex read faults in SRAM memories," in *Proc. Design Diag. Electron. Circuits Syst.*, Apr. 2007, pp. 1–6.

[7]. R. L. Franch, S. H. Dhong, and R. E. Scheuerlein, "A large VDS data retention test pattern for DRAM's," *Proc. IEEE J. Solid, State Circuits*, vol. 27, no. 8, pp. 1214–1217, Aug. 1992.

[8]. C.-M. Chang, M. C.-T. Chao, H. Rei-Fu, and C. Ding-Yuan, "Testing methodology of embedded DRAMs," in *Proc. Int. Test Conf.*, Oct. 2008, pp. 1–9.

[9]. Hyoyoung Shin, Youngkyu Park, Gihwa Lee, Jungsik Park, and Sungho Kang "Interleaving Test Algorithm for Subthreshold Leakage-Current Defects in DRAM Considering the Equal Bit Line Stress*" in proc.IEEE very large scale integration (vlsi) systems, vol. 22, no. 4, April 2014*

[10]. A. J. Van de Goor and I. Schanstra, "Address and data scrambling: Causes and impact on memory tests," in *Proc. 1st IEEE Int. Workshop Electron. Design, Test Appl.*, Jan. 2002, pp. 128–136

[11]. B. Bhat and F. Mueller, "Making DRAM refresh predictable," in *Proc. Real-Time Syst., Euromicro Conf.*, Jul. 2010, pp. 145–154.

[12]. M. Meterelliyoz, H. Mahmoodi, and K. Roy, "A leakage control system for thermal stability during burn-in test," in *Proc. Int. Test Conf.*, Nov. 2005, pp. 991–1000.