A Memory Specific Notation for Fault Modeling

Zaid Al-Ars   Ad J. van de Goor
Faculty of Information Technology and Systems
Section of Computer Engineering
Delft University of Technology
Mekelweg 4, 2628 CD Delft, The Netherlands

Jens Braun   Detlev Richter
Infineon Technologies AG
Product Engineering Group 2
Balanstr. 73, 81541 Munich, Germany

E-mail: z.e.al-ars@its.tudelft.nl

Abstract: *This paper shows the shortcomings of the current, generic notation for fault models and extends it to allow for describing fault models for DRAMs. The advantage is that the extended fault models can easily be translated into operation sequences and tests that detect the described fault. Examples are given to show that the new notation results in optimized, memory specific, tests that have a shorter run time for a given fault coverage.*

Key words: *functional fault models, fault primitives, memory testing, DRAM, memory specific fault analysis*

1 Introduction

Research on the faulty behavior of memory devices has resulted in a concise, easy to understand description language of the faulty behavior that is commonly used to describe memory faults [vdGoor00]. The building blocks of this description language are referred to as *fault primitives (FPs)*. The importance of FPs lies in their simplicity in the sense that they include all parameters necessary to identify a given observed faulty behavior.

The ever increasing complexity of memory faults has left FPs lagging behind in their fault description capability in terms of the following two areas:

1. The use of stress conditions. Most commercial tests use temperature and voltage stress, for example, to facilitate the fault detection process.

2. The use of *Memory Specific Operations*. Many newer types of memories, and especially DRAMs, allow in addition to the traditional read and write operations, additional operations or modes of operations (such as precharge, page mode, etc.) which have a large impact on the possible faulty behavior of the memory. This makes the FP notation imprecise in describing the faulty behavior, since it is possible to translate the FP operations into memory specific operations in more than one way.

This paper explores the space of stress parameters and DRAM memory specific operations and extends the FP notation such that the faulty behavior and tests can again be described in a precise way.

Section 2 describes the current approach to establish fault models and tests. Section 3 shows the shortcomings of the current, academic approach by describing the industrial test practices. Section 4 proposes an extension to the existing notation for FPs and tests, such that they can be described in a precise manner. Section 5 lists some examples showing the use of the extended notation. Finally, Section 6 ends with the conclusions.

2 Memory test approach

This section describes the process of how faults are detected in memory devices. First, the set of faults of interest has to be established. This can be done by inserting electrical (resistive, capacitive and/or inductive) defects into the electrical design of the memory. Then, SPICE simulation is used to establish the impact of the defect on the functional behavior of the memory. If a faulty behavior is observed, it will be described in terms of FPs [Al-Ars01]. Next, a test will be designed which is capable of detecting the observed faulty behavior, as described by the FPs that resulted from the SPICE simulation.

Section 2.1 describes the space of possible memory faults, in terms of the notation used for FPs. Section 2.2 describes the notation used for describing march tests, and Section 2.3 shows that march test design is a trivial task, given the targeted set of FPs.

2.1 FP notation

Two basic ingredients are needed to describe any fault in a memory: (1) a sequence of performed memory operations, and (2) a list of corresponding deviations in the observed behavior from the expected one.

1. An operation sequence that results in a difference between the observed and the expected memory behavior is called a *sensitizing operation sequence (SOS)*. For example, the SOS for an up-transition fault (TF↑) in a cell is

1081-7735/01 $10.00 © 2001 IEEE

$0w1$, which requires initializing the cell to 0, after which a 1 has to be written into the cell. The observed memory behavior that deviates from the expected one is called a *faulty behavior* or simply a *fault*. For TF↑, the faulty behavior is that after the write 1 operation has been performed the cell still contains a 0. Any SOS can be represented by the following notation:

$$d_{c_1} \ldots d_{c_i} \ldots d_{c_m}\ Od_{c_1} \ldots Od_{c_j} \ldots Od_{c_n}$$

where c_x: cell address used,
O: type of operation on c, $O \in \{w, r\}$,
d: initialization or written data into c, $d \in \{0, 1\}$,
m: number of initializations, and
n: number of operations.

The *initialization part* is applied to m cells (denoted as c_i), while the *operation part* is applied to n cells (denoted as c_j). Note that the value of d in rd_{c_j} of the operation part represents the expected value of the read operation, which may be different from the actual read value detected on the output in case of a faulty memory. As an example of the notation, if an operation sequence is denoted by $0_c w1_c r1_c$ then the sequence starts by accessing cell c (which contains a 0) and writing a 1 into it, then reading the written 1.

2. The second ingredient needed to specify a fault model is a list of deviations in the observed behavior from the expected one. The only functional parameters considered relevant to the faulty behavior are the stored logic value in the cell and the output value of a read operation.

Considering the above, any difference between the observed and expected memory behavior can be denoted by the following notation $<S/F/R>$, referred to as an FP [vdGoor00]. S describes the SOS that sensitizes the fault; F describes the value of the faulty cell, $F \in \{0, 1\}$; and R describes the logic output level of a read operation, $R \in \{0, 1, -\}$. The '−' is used in case a write, and not a read, is the operation that sensitizes the fault. For example, in the FP $<0_c w1_c/0/->$, which a TF↑, the SOS $S = 0_c w1_c$ means that cell c is assumed to have the initial value 0, after which a 1 is written into c. The fault effect $F = 0$ indicates that after performing a $w1$ to c, as indicated by the SOS, c remains in state 0. The output of the read operation $R = -$ indicates that the SOS does not end with a rd_c operation. The notation for the FP $<0_c w1_c/0/->$ can be simplified to $<0w1/0/->_c$.

FPs can be classified into different classes, depending on the SOS. Let $\#C$ be the number of *different* memory cells initialized (c_i) or accessed (c_j) in S, and let $\#O$ be the number of operations (w or r) performed in S. For example, if $S = 0_{c_1} i0_{c_2} w1_{c_2}$ then $\#C = 2$ since two cells (c_1 and c_2) are present in S, while $\#O = 1$ since only one operation is performed ($w1$ to c_2). A taxonomy of FPs is shown in Figure 1.

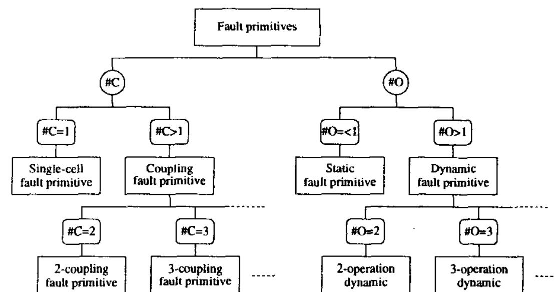

Figure 1. A taxonomy of fault primitives.

A **functional fault model** (FFM) is a non-empty set of fault primitives (FPs) [vdGoor00]. For example, the *transition fault (TF)* FFM consists of 2 FPs: TF $= \{<0w1/0/->, <1w0/1/->\}$.

2.2 March tests

In order to inspect memory devices for possible faulty behavior, memory testing is performed on all produced memory components. A large number of memory tests are being used today, each with its own advantages and disadvantages. *March tests* are among the most popular memory tests, due to their low complexity and high fault coverage.

The idea of march tests is to construct a number of operation sequences and to perform each sequence on all memory cells one after the other. Therefore, a *march test* can be defined as sequence of march elements, where a *march element* is a sequence of memory operations performed on all memory cells. In a march element, the way one proceeds from one cell to the next is specified by the *address order*, which can be increasing (denoted by ⇑) or decreasing (denoted by ⇓). The ⇓ address order has to be the exact opposite of the ⇑ address order. For some march elements, the address order can be chosen arbitrarily which is denoted by the ⇕ symbol. In a march element, it is possible to perform a write 0 operation ($w0$), write 1 ($w1$), read 0 ($r0$) and read 1 ($r1$) operation. The 0 and 1 after read operations represent the expected values. An example of a march element is $\Uparrow(r0, w1)$ where all memory cells are accessed in an increasing address order while performing $r0$ then $w1$ on each cell.

By arranging a number of march elements one after the other, a march test is constructed. An example of a march test is $\{\Updownarrow(w0); \Uparrow(r0, w1); \Downarrow(r1, w0)\}$, which is the well known march test called MATS+. It consists of three march elements denoted as M_0, M_1 and M_2. The test be-

gins with writing 0 into all memory cells in an increasing or decreasing order, then to each cell a read 0 and a write 1 operation is performed in an increasing order, and finally to each cell a read 1 and a write 0 operation is performed in a decreasing order.

2.3 Test generation

The analytical approach to memory testing begins with an analysis of the faulty behavior of the memory, which is then described by a number of FPs. FPs give an exact description of the way the faulty behavior is sensitized, and can easily be used to generate memory tests to detect the observed faulty behavior.

As an example, assume that fault analysis of a given memory indicates that the memory suffers from an up-transition fault TF↑ = $\{<0w1/0/->\}$. The FP gives a precise description of the way the observed fault can be sensitized. Therefore, it is possible to generate a march test to detect this faulty behavior: $\{\Updownarrow(w0); \Uparrow(w1, r1)\}$. The first march element of the test initializes the memory to 0, followed by a second march element that sensitizes the fault by attempting to perform a $w1$ operation. If this $w1$ operation fails, the fault will be detected by performing the $r1$ operation in the second march element.

3 Industrial test practices

Industrial testing of memories is a complex and involved field that uses as many memory aspects as possible to stress the memory and induce a failure. The industrially used aspects can be classified into the category of used stress conditions and the category of applied memory specific operations.

3.1 Stress conditions

A stress condition consists of a number of stresses with assigned values. A *stress* represents some way of facilitating the fault detection process. Several papers have been published, showing the effectiveness of stresses [Goto97, Schanstra99, vdGoor99].

An example of testing using stress is the V_{DD} bump test. This test is used to examine the ability of the memory to charge up the DRAM cell capacitor to the high voltage level required when a w operation is performed to a memory cell [Vollrath00]. This is done by changing the supply voltage (V_{DD}) in an attempt to induce a memory failure. This test can be represented by the following march test $\{\Updownarrow(w0); V_{DD} = V_{nom} - V_{bump}; \Updownarrow(w1); V_{DD} = V_{nom}; \Updownarrow(r1)\}$, where V_{nom} is the nominal voltage and V_{bump} is the bump voltage.

The V_{DD} bump test uses one of the supply voltages as the stress parameter. The stresses commonly used in industry can be divided into the following classes:
1. **Algorithmic stresses**—They specify more precisely the way the algorithm should be applied. This can be in terms of:

a. The specific addressing used (e.g., X-fast, Y-fast, address complement, Gray code, etc.)

b. The specific data to be read/written. This usually is denoted as the data background (DB). The DB specifies the actual value to be written for a 0 and a 1 in the algorithm.
2. **Environmental stresses**—These are stresses which relate to the environment the chip has to be tested in. Temperature, voltage and timing are typical examples of this class of stresses.

The exact way these stress conditions are used depends on the specific design and construction of the memory under test.

3.2 Memory specific operations

When memory tests are applied to a given memory, it is not always possible to use the simple r and w operations, as specified by the traditional FP and march test notation, because memory devices may have additional commands and/or modes to accomplish the desired memory functionality.

For example, DRAMs today have many modes of operation that aim primarily at increasing the performance by reducing the time needed to access stored information. Figure 2 shows a typical functional model of a modern DRAM, which has a data input/output bus, and an input command bus and address bus. Traditionally, memory functionality is described by the simple r and w operations decoded in the command bus. However, DRAMs today can perform these two operations in many different modes, enabling more flexibility and/or more speed in manipulating the stored data. In order to describe the different DRAM modes of operation, five commands should be used that are more primitive than r and w. These commands are described next:

1. ACT: This is the *activate* command. When this command is issued, the address on the address bus is considered as a row address. The address is decoded by the row decoder to activate a *word line (WL)* of cells in the memory and to sense the data of the activated cells.

2. WR: This is the *write* command. When this command is issued, the address on the address bus is considered as a column address. The address is decoded by the column decoder to select a given cell to be written by the data on the data bus.

3. RD: This is the *read* command. When this command is issued, the address on the address bus is considered as a column address. The address is decoded by the column decoder to select a given cell to be read and forward the data to the data bus.
4. PRE: This is the *precharge* command. When this command is issued, any activated WL is deactivated and bit lines are precharged.
5. NOP: This is the *no-operation* command, which represents an idle cycle.

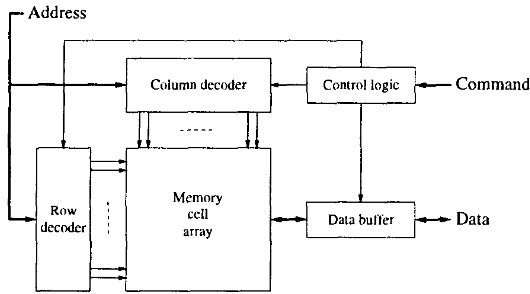

Figure 2. Functional model of a DRAM.

With the above five primitive commands, any DRAM operation can be described. A number of operation modes are described next using these five primitive commands.

Write operation: The write operation is traditionally denoted as wd_c, where d is the data to be written into cell c. Using the five primitive commands, the write operation is performed as ACT$_{c_w}$ WRd_{c_b} PRE, where c_w is the row address (or the WL address) of c and c_b is the column address (or bit line address) of c.

Read operation: The read operation is traditionally denoted as rd_c, where d is the expected data to be read from cell c. Using the five primitive operations, the read operation is performed as ACT$_{c_w}$ RDd_{c_b} PRE.

Refresh operation: The refresh operation is used to restored data into memory cells to prevent losing stored data by leakage. This operation cannot be represented using the traditional r and w operations. Using the five primitives, the refresh operation is performed as ACT$_{c_w}$ PRE for all WL addresses c_w in the memory.

Read modify write operation: This operation performs a read followed by a write on the same cell without the need to precharge the cell in between. This operation cannot be represented using the traditional operations. Using the five primitives, this operation is performed as ACT$_{c_w}$ RDx_{c_b} WRy_{c_b} PRE.

Fast page mode: Operations in this mode are performed on any cell on a given activated WL (page) without precharging. This mode of operation greatly increases the performance of the memory. Using the five primitive operations, the fast page mode is performed as ACT$_{c_w}$ O$d1_{c_{b1}}$ ... O$dn_{c_{bn}}$ PRE, where O is either RD, WR or NOP.

4 Device specific FPs

This section discusses the shortcomings of the current FP notation and suggests ways to improve it. The discussion here is related to the way external operations performed on the memory result in sensitizing faults. The improvements to the FPs concentrate on a given type of DRAMs as the memory of interest, but similar strategies may be used to take other types of memory into consideration.

4.1 Shortcomings of current FPs

The current FP notation, presented in Section 2, is a generic notation that is compatible with almost all RAM devices today. Such a general notation does not address the faulty behavior needs of specific types of memory devices (DRAMs, for example). The inability of FPs to describe DRAM specific faulty behavior originates from the limited types of SOS's S can describe.

The definition of S is based on a reduced memory model, as shown in Figure 3(a), that is the same for all RAM devices [vdGoor98]. This model assumes that a memory has only three input/output terminals: an address input bus (Address), a data input/output bus (Data in/out), and a command input (R/$\overline{\text{W}}$) to perform either a read or a write. For example, this model can represent a TF↑ transition fault as in $<0w1/0/->_c$, where the address bus contains the address of the cell c, the R/$\overline{\text{W}}$ input decodes a write operation and the data in/out bus contains the written data 1. The advantage of this model is that it is simple, thereby keeping the needed analysis of the faulty behavior simple. The model is also generic, which makes analysis results based on this model applicable to many memory devices. The disadvantage of the model, however, is that it neglects a number of parameters that could effect the behavior of the memory (e.g., temperature and voltage). An attempt has been made to improve on this model by including voltages and temperature (T), found to be important in testing [Offerman97]. Still, the model does not include timing or specific memory operations.

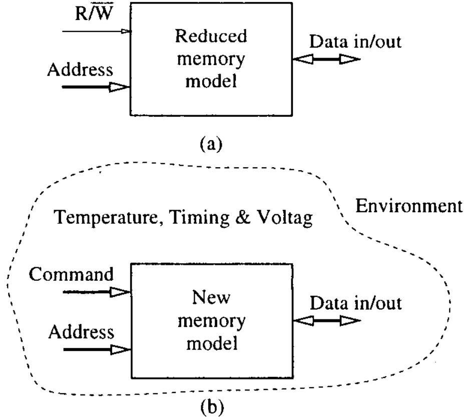

Figure 3. Memory models used for defining S: (a) a reduced memory model, (b) a detailed memory model.

Figure 3(b) shows such a memory model with 3 input and/or output terminals, and which takes supply voltages temperature and timing into consideration. The data in/out bus and the address bus are the same in the reduced as well as the detailed models. The command bus in the new model replaces the R/$\overline{\text{W}}$ input in the old model.

In order to carry out this extension on the new model shown in Figure 3(b), we need to select a specific memory product in order to give the exact definitions of the command bus and specify the type of supply voltages and timing parameters to be modified. Table 1 gives these definitions for the current Infineon DRAM product [Falter00].

Table 1 identifies two voltages for the power supply and two clock related parameters to control timing. In addition, the table lists the five DRAM specific primitive commands.

4.2 Extending the FP notation

Extending the current FP notation can be done by enabling S to describe any possible SOS performed on the new model shown in Figure 3(b) using the specific terminal definitions of the current Infineon DRAM product listed in Table 1. The needed extensions involve: 1) describing the five DRAM specific commands, 2) describing the algorithmic stresses, and 3) describing the environmental stresses. Each one of these extensions is described below.

Describing memory specific operations

The first step to account for the five DRAM operations of Table 1 is to modify the set of possible operations to include all of them. This way the set of possible performed operations should be $\{\text{ACT}_{c_w}, \text{RD}d_{c_b}, \text{WR}d_{c_b}, \text{PRE}, \text{NOP}\}$. Unlike the traditional read (r) and write (w) operations described in Section 2, these five operations are not independent from each other, which means that some SOS's should not be allowed. Therefore, some conditions should be introduced to limit the space of possible SOS's to those practically acceptable:

1. ACT $\text{O}_1 \ldots \text{O}_n$ PRE : $\text{O} \in \{\text{RD}, \text{WR}, \text{NOP}\}$
2. NOP
3. PRE

Describing algorithmic stresses

Algorithmic stresses refer to the specific addressing used to detect a given fault, or to the needed data background. The needed data background is already described by the current FP notation. In order to take addressing into consideration, a number of attributes should be added to the FP to specify the topological relation between the cells accessed within the FP. The most important attributes are BL (cells are along the same bit line), WL (cells are along the same word line), and DG (cells are along the diagonal). For example, the two cells in the fault $<0_a 1_v/0/->_{BL}$ are indicated by BL to be along the same bit line.

Describing environmental stresses

Unlike performed operations on the command bus, environmental stresses are not discrete but continuous quantities. This means that it is not possible to identify *all* allowed stresses and individually integrate them into S. Rather, a parameterization of the stresses can be considered by introducing *stress defining variables* for each stress. Therefore, five stress variables should be introduced into S, one for each stress listed in Table 1. To distinguish stresses from operations, stresses are included in square brackets. For example, S = [V_{arr1} = 3.5] ACT_{c_w} $\text{WR}d_{c_b}$ PRE [V_{arr1} = 3.0] ACT_{c_w} $\text{RD}d_{c_b}$ PRE means that cell c (with row address c_w and column address c_b) is written with data d at an array voltage of 3.5 V and then read at a voltage of 3.0 V.

5 Examples of new notation

In this section, examples are given to justify the need for extending the FP notation. The examples concern the memory specific commands and the environmental stresses included in the new FP notation.

5.1 Memory specific operations

Consider a bridge defect that connects two DRAM memory cells together. This defect causes a write operation to the aggressor to affect the stored voltage in the victim. If

Table 1. Definitions of command bus, supply voltages, timing and temperature parameters for the current Infineon DRAM product.

Terminal	Definition	Description
Command	ACT	Activate: access a row of cells and sense their data content.
	WR	Write: write data into a memory cell.
	RD	Read: forward data from the sense amplifier to output.
	PRE	Precharge: restore data to cells and precharge bit lines.
	NOP	No operation
Supply voltage	V_{arr1}	First array voltage: power supply to the cell array.
	V_{arr2}	Second array voltage: power supply to the cell array.
Timing	Frq	Frequency of the clock signal
	τ	Duty cycle of the clock signal
Temperature	T	Ambient temperature

the bridge resistance is high enough, it would take a number of write operations to the aggressor to change the stored voltage in the victim.

Assume that the victim stores a 0, and that it takes 3 $w1$ operations to the aggressor to flip the state of the victim. The traditional FP notation describes this faulty behavior as $<1_a w1_a w1_a w1_a 0_v/1/->$. This description cannot be uniquely translated into the DRAM primitive commands since it is possible to perform the w and r operations in single cycle mode and in fast page mode. In the new notation, the faulty behavior can be uniquely described using fast page mode as follows $<0_a$ ACT_{a_w} WR1_{a_b} WR1_{a_b} WR1_{a_b} PRE $0_v/1/->$.

5.2 Stress conditions

Again, consider the example where a bridge defect connects two DRAM cells together. This bridge results in the following FP $<0_a$ ACT_{a_w} WR1_{a_b} WR1_{a_b} WR1_{a_b} PRE $0_v/1/->$. To help sensitize this FP, stress conditions can be used to optimize the SOS so that the FP would be detected more easily.

For this faulty behavior in particular, increasing the supply from the nominal voltage (V_n) to a higher voltage (V_h) while writing would shorten the time needed to deplete the victim capacitor. At the same time, modifying the operation temperature to some specific T_s may decrease the resistance of the bridge in favor of the failure mechanism. The value of T_s depends on the nature of the bridge, and may be higher or lower than the nominal operation temperature. Taking these stress conditions into consideration, the new FP description of the failure mechanism would be $<[V_{arr1} = V_h, T = T_c]\, 0_a$ ACT_{a_w} WR1_{a_b} WR1_{a_b} WR1_{a_b} PRE $0_v/1/->$.

6 Conclusions

In this paper, a new fault modeling notation has been developed to study special, memory specific, types of faulty behavior. The notation makes possible the consideration of operations other than traditional reads and writes, and includes temperature, supply voltages, and timing in the fault analysis. Tests generated using the new notation uniquely describe the faulty behavior and can be optimized for the memory under analysis, and thus have a shorter run time for a given fault coverage.

References

[Al-Ars01] Z. Al-Ars and A.J. van de Goor, "Static and Dynamic Behavior of Memory Cell Array Opens and Shorts in Embedded DRAMs," *in Proc. Design, Automation and Test in Europe*, 2001, pp. 496–503.

[Falter00] T. Falter and D. Richter, "Overview of Status and Challenges of System Testing on Chip with Embedded DRAMs," *in Solid-State Electronics*, no. 44, 2000, pp. 761–766.

[Goto97] H. Goto, S. Nakamura and K. Iwasaki, "Experimental Fault Analysis of 1Mb SRAM Chips," *in Proc. IEEE VLSI Test Symp.*, 1997, pp. 31–36.

[Offerman97] A. Offerman and A.J. van de Goor, "An Open Notation for Memory Tests," *in Proc. IEEE Int'l Workshop on Memory Technology, Design and Testing*, 1997, pp. 71–78.

[Schanstra99] I. Schanstra and A.J. van de Goor, "Industrial Evaluation of Stress Combinations for March Tests applied to SRAMs," *in Proc. IEEE Int'l Test Conf.*, 1999, pp. 983–992.

[vdGoor98] A.J. van de Goor, *Testing Semiconductor Memories, Theory and Practice*, ComTex Publishing, Gouda, The Netherlands, 1998, http://ce.et.tudelft.nl/~vdgoor/

[vdGoor99] A.J. van de Goor and J. de Neef, "Industrial Evaluation of DRAM Tests," *in Proc. Design, Automation and Test in Europe*, 1999, pp. 623–630.

[vdGoor00] A.J. van de Goor and Z. Al-Ars, "Functional Memory Faults: A Formal Notation and a Taxonomy," *in Proc. IEEE VLSI Test Symp.*, 2000, pp. 281–289.

[Vollrath00] J. Vollrath, "Tutorial: Synchronous Dynamic Memory Test Construction, A Field Approach," *in Proc. IEEE Int'l Workshop Memory Technology, Design and Testing*, 2000.