Space of DRAM Fault Models and Corresponding Testing

Zaid Al-Ars Said Hamdioui Ad J. van de Goor
Delft University of Technology, Faculty of EE, Mathematics and CS
Lab of Computer Engineering, Mekelweg 4, 2628 CD Delft, The Netherlands
E-mail: z.e.al-ars@ewi.tudelft.nl

Abstract: *DRAMs play an important role in the semiconductor industry, due to their highly dense layout and their low price per bit. This paper presents the first framework of fault models specifically designed to describe the faulty behavior of DRAMs. The fault models in this paper are the outcome of a close collaboration with the industry, and are validated using a detailed Spice-based analysis of the faulty behavior of real DRAMs. The resulting fault space is then used to derive a couple of new DRAM-specific tests, needed to detect some of the faults in practice.*

1 Introduction

Despite the importance and the widespread use of DRAMs [Adler95], the analysis of their faulty behavior and testing has yet to be given its due attention. Deemed by the academic memory testing community as a variant of SRAMs, there has been very limited DRAM-specific research on their fault models and testing [Hamdioui04]. Much of the work published on DRAM-specific testing has come directly from the industry in the form of tutorials describing the status quo of their test technology [Vollrath00, McConnell98].

This paper presents a newly developed space of DRAM faults, based on an elaborate study of the faulty behavior of real memories using defect injection and Spice simulation. This new space is the result of a close collaboration with Infineon Technologies to come up with new methodologies to test for the faulty behavior of DRAMs. The new space extends the existing general framework of memory faults with a number of new concepts, such as transient faults, partial faults and soft faults, all of which specifically intended to describe faulty behavior mainly observed in DRAMs [Al-Ars05].

The paper is organized as follows. Section 2 starts by a short introduction to the concept of fault primitives. Then, Section 3 defines the different DRAM-specific fault models, and identifies the whole space of these faults. Section 4 generates tests suitable to detect the DRAM-specific faulty behavior. Finally, Section 5 ends with the conclusions.

2 Fault primitives

Functional faults can be defined as the deviation of the observed behavior from the specified one under a set of performed operations. In order to specify a certain memory fault, one has to represent it in the form of a *fault primitive (FP)*, denoted as $<S/F/R>$. S describes the sensitizing operation sequence that sensitizes the fault, F describes the value or the behavior of the faulty cell (e.g., the cell flips from 0 to 1), $F \in \{0, 1\}$, and R describes the logic output level of a read operation, $R \in \{0, 1, -\}$. R has a value of 0 or 1 when the fault is sensitized by a read operation, while the "$-$" is used when a write operation sensitizes the fault. For example, in the FP $<0w1/0/->$, which is the up-transition fault (TF_1), $S = 0w1$ means that a $w1$ operation is written to a cell initialized to 0. The fault effect $F = 0$ indicates that after performing $w1$, the cell remains in state 0. The output of the read operation ($R = -$) indicates there is no expected output for the memory (since S ends with a write rather than a read).

Functional fault models (FFMs) can be defined as a non-empty set of FPs. There are many classes of FFMs, the most important of which are single-cell static FFMs and two-cell static FFMs.

Single-cell static FFMs consist of FPs sensitized by performing at most one operation on a faulty cell. Table 1 lists all single-cell static FFMs and their corresponding FPs. In total, there are 6 different types of FFMs: state fault (SF), transition fault (TF), write destructive fault (WDF), read destructive fault (RDF), incorrect read fault (IRF), deceptive read destructive fault (DRDF) [Adams96].

Table 1. Single-cell static FFMs and their corresponding FPs.

#	Fault	FP	Name
1	SF	$<0/1/->$, $<1/0/->$	State fault
2	TF	$<0w1/0/->$, $<1w0/1/->$	Transition fault
3	WDF	$<0w0/1/->$, $<1w1/0/->$	Write destructive fault
4	RDF	$<0r0/1/1>$, $<1r1/0/0>$	Read destructive fault
5	IRF	$<0r0/0/1>$, $<1r1/1/0>$	Incorrect read fault
6	DRDF	$<0r0/1/0>$, $<1r1/0/1>$	Deceptive RDF

Two-cell static FFMs consist of FPs sensitized by performing at most one operation while considering the faulty effect of two cells. Such FPs can be represented as $<S_a; S_v/F/R>$, where S_a is the sequence performed on the aggressor (a) and S_v is the sequence performed on the victim (v). Table 2 lists all two-cell static FFMs and their corresponding FPs. In total, there are 7 different types of two-cell static FFMs: state coupling fault (CFst), disturb coupling fault (CFds), transition coupling fault (CFtr), write destructive coupling fault (CFwd), read destructive coupling fault (CFrd), incorrect read coupling fault (CFir), and deceptive read destructive coupling fault (CFdrd).

3-9810801-0-6/DATE06 © 2006 EDAA

Table 2. Two-cell static FFMs and their FPs ($x, y \in \{0, 1\}$).

#	Fault	FP	Name
1	CFst	$<0; 0/1/->$, $<0; 1/0/->$ $<1; 1/0/->$, $<1; 0/1/->$	State coupling fault
2	CFds	$<xwy; 0/1/->$, $<xwy; 1/0/->$ $<xrx; 0/1/->$, $<xrx; 1/0/->$	Disturb coupling fault
3	CFtr	$<0; 0w1/0/->$, $<0; 1w0/1/->$ $<1; 0w1/0/->$, $<1; 1w0/1/->$	Transition coupling fault
4	CFwd	$<0; 0w0/1/->$, $<0; 1w1/0/->$ $<1; 0w0/1/->$, $<1; 1w1/0/->$	Write destructive coupling fault
5	CFrd	$<0; 0r0/1/1>$, $<0; 1r1/0/0>$ $<1; 0r0/1/1>$, $<1; 1r1/0/0>$	Read destructive coupling fault
6	CFir	$<0; 0r0/0/1>$, $<0; 1r1/1/0>$ $<1; 0r0/0/1>$, $<1; 1r1/1/0>$	Incorrect read coupling fault
7	CFdrd	$<0; 0r0/1/0>$, $<0; 1r1/0/1>$ $<1; 0r0/1/0>$, $<1; 1r1/0/1>$	Deceptive read destructive CF

3 DRAM-specific faults

The sensitizing operation sequence (S) in DRAMs can be divided into two parts: the *initialization part* (I) and the *fault activation part* (A). I represents the initial data present in the cells, along with the operations performed to ensure that all relevant cells are set to a known predefined states. A is the sequence of operations needed to activate the fault. Therefore, a DRAM fault primitive is denoted by FP $= <S/F/R> = <IA/F/R>$ [Al-Ars05]. The operations performed in I take place before the operations performed in A. For single-cell and two-cell static faults defined in Section 2, I and A have the following form:

- For SF or CFst (no operation performed), there is no activation part, and $S = I$.
- For other faults (one operation performed), I consists of the state declaration of the cells while A consists of the operation. For example, in the CFds $<0w1; 0/1/->$, I is "$0_a; 0_v$" (i.e., the initialization of a to 0 and v to 0), and A is "$w1$" (the operation $w1$).

The classification of DRAM-specific faults is based on the concepts of the initialization and the activation parts of S, as described in Table 3. As the operations in S are being performed, DRAM faults may take place in four different stages: during I, between I and A, during A, and after A. In addition, DRAM faults take place because of two main effects:

- improperly set voltages resulting in voltage dependent faults, and
- leakage currents resulting in time dependent faults.

Improper voltages can either be within the cell causing *partial faults*, or be in the periphery causing *dirty faults*. Leakage currents, on the other hand, can either strengthen the faulty state in the cell causing *soft faults*, or correct it causing *transient faults*. The name "partial faults" indicated that faults can get *partially* sensitized, unless an operation is performed a multiple number of times. The name "dirty faults" indicates that voltages in the periphery of the memory can be improperly set (or *dirty*), unless extra operations are performed to *clean* them up. The name "soft faults" indicates that faults can get *softly* sensitized, and a period of time should pass before they become *hard* faults and may be detected. The name "transient faults" indicates that faults can get *temporarily* sensitized, after which they get automatically corrected.

Table 3. Classification of DRAM-specific faults.

Cause of problem	Mode of problem	Resulting fault model			
		During I	Bet. I & A	During A	After A
Improper voltages (voltage dependent)	Within cell	Partial faults	—	Partial faults	—
	In periphery	—	Dirty faults	—	Dirty faults
	In cells & periphery	Partial faults	Dirty faults	Partial faults	Dirty faults
Leakage currents (time dependent)	Supports operation	Transient faults			
	Opposes operation	Soft faults			

3.1 Voltage dependent faults

In a DRAM, operations are supposed to properly set the voltage levels on different nodes (cells or bit lines, for example) to a well-defined high or low voltage level. In general, however, a voltage across a capacitor may take any value from a continuous range of voltages. Therefore, operations performed on a defective memory may set improper voltage levels on different memory nodes, thereby causing two types of DRAM faults: partial faults and dirty faults [Al-Ars05].

Partial faults

Definition—These are faults that can only be sensitized when a specific memory operation is successively repeated a number of times, either to properly initialize the faulty cell (causing *partial faults during initialization* I), or to properly sensitize the fault in the cell (causing *partial faults during activation* A). Figure 1(a) shows an example of an open (R_{op}) in the cell, causing a partial fault in I. R_{op} prevents fully initializing the cell to the required voltage with only one operation, which means that full initialization requires repeating the operation a number of times.

Figure 1(b) shows an example of a bridge (R_{br}) between two cells, causing a partial fault in A.

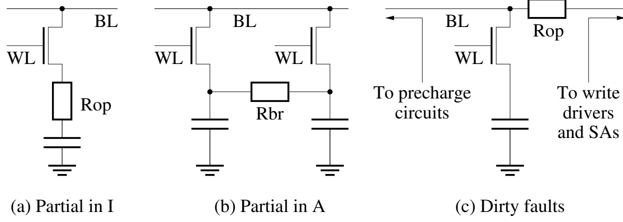

Figure 1. Defects causing (a) partial faults in I, (b) in A, and (c) causing dirty faults.

Fault modeling—This is achieved by performing an operation Ox an h (or *hammer*) number of times, to ensure sensitizing the fault (denoted as Ox^h). For example, if a single-cell fault of the form $<xOy/F/R>$ becomes partial in A, it should be modeled as $<xOy^h/F/R>$, which means that repeating the operation of A on the cell multiple times causes a fault.

Dirty faults

Definition—These 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 ensure that the sensitized fault is detectable, additional operations must be performed to correct the corrupted state of the memory. Figure 1(c) shows an example of an open defect (R_{op}) on the BL that causes dirty faults. This defect disconnects memory cells from the write drivers, thereby limiting the ability of the memory to properly write the cells. At the same time, this defect disconnects the precharge devices from part of the BL, which prevents properly precharging the BL. As a result, a $w0$ operation that fails to write 0 in the cell may end up preconditioning the BL to properly sense a 0 in a subsequent read operation, thereby preventing the detection of the faulty $w0$ and causing a dirty fault.
Fault modeling—This is achieved by the introduction of *completing operations* to the FP, which need to be performed after the initialization (I) or after the activation (A) part of S. There are two different defects known to cause dirty faults in DRAMs, one is an open on the BL, which results in improperly set BL voltages, and the other is an open in the sense amplifier, which results in improperly set data buffers. Both defects cause dirty faults that can be detected using a write completing operation with data opposite to the data in the victim, performed to a cell different from the victim but positioned on the same BL. This gives the following FP: $<xO_v y[w_b \overline{y}]/\overline{y}/->_{b,v\in \mathrm{BL}}$.

3.2 Time dependent faults

Time dependent faulty behavior is caused by leakage currents flowing into faulty cells causing *soft faults* or *transient faults* [Al-Ars05]. Generic faults described in Section 2 are neither soft nor transient, and therefore they may be referred to as *hard faults* to indicate that they are insensitive to time.

Soft faults

Definition—Soft faults only become detectable after some time from their sensitization. These faults have usually been tested for by adding a *delay* within the test, to facilitate detecting the fault, as it is the case for the *data retention fault*, for example [Dekker90]. Soft faults are caused by writing weak voltages into memory cells, that soon get depleted by naturally occurring leakage currents. Soft faults can also take place in other types of memory, but they are much more likely to occur in DRAMs.
Fault modeling—In terms of the FP notation, soft faults are represented as $<S_T/F/R>$, where the sensitizing operation sequence has an added time parameter T to indicate that some time should first elapse before the fault effect is completely sensitized. The open defect in Figure 1(a) shows an example of a cell open that causes soft faults in DRAMs. If the open defect has an intermediate resistance value that is not too high (causing hard faults) and not too low (not causing a fault at all), write operations succeed but are only able to write a *weak voltage* into the cell. As time passes, and due to naturally occurring leakage in DRAM cells, a weakly written voltage is depleted gradually, thereby losing the stored information over time.

Transient faults

Definition—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 for 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. As an example of transient faults, consider the DRAM open shown in Figure 1(a), where R_{op} limits the ability of write operations to charge up and discharge cell voltages. For a specific range of R_{op} values, write operations set a faulty voltage within the cell that is not strong enough to qualify as a hard fault. As time passes, and due to naturally occurring leakage in DRAM cells, a weakly written faulty voltage is depleted gradually, thereby correcting the faulty information over time.
Fault modeling—In terms of the FP notation, transient faults are represented as $<\underline{S}/F_L/R>$, where the under-

score below S means that the operations in S should be performed in back-to-back mode directly after each other, and that the faulty cell value F has an added time parameter L (*life time*) to indicate that these faults are time limited. In terms of detection conditions, an underscore below operations in a transient fault means that the operations have to be performed after each other within one march element. For example, if $S = w1\underline{w0}$ then the detection condition should be $\Updownarrow(..., w1, \underline{w0}, ...)$.

3.3 Space of DRAM faults

Any generic memory fault, described in Section 2, can represent a DRAM-specific fault by adding a DRAM-specific fault attribute to it. For example, it is possible to construct a number of DRAM-specific versions of the transition fault, such as the partial transition fault, the dirty transition fault, the soft transition fault, and so on. As discussed above, there are five DRAM-specific attributes, classified into two different classes. First, voltage dependent faults are: partial faults (p), and dirty faults (d). Second, time dependent faults are: hard (h), soft (s) and transient (t) faults. It is important to note here that there are two different types of partial faults, one is the initialization related partial faults (p_i), while the other is the activation related partial faults (p_a).

In addition to these individual attributes, it is possible to have multiple attributes at the same time associated to a given generic fault model. As a result, it is possible to establish the whole space of DRAM faults, by considering the possibility that multiple attributes apply to a fault at the same time, for a given defect.

First of all, note that any voltage dependent attribute can be combined with any time dependent attribute, without restrictions. The reason behind this is the fact that these two classes of DRAM-specific faults are based on two physically independent root causes, which results in faults that are independent as well. This means that each generic fault can be associated with a voltage dependent attribute in combination with a timing dependent attribute in the following way:

$$\text{Fault} = \left\{ \begin{array}{c} \text{Voltage} \\ \text{attribute} \end{array} \right\} \left\{ \begin{array}{c} \text{Timing} \\ \text{attribute} \end{array} \right\} \text{FP} \qquad (1)$$

Furthermore, note that the different attributes of time dependent faults (h, s and t), are not compatible with each other, since they are based on leakage currents either supporting or opposing the applied sensitizing operation. In other words, the set of timing dependent faults is equal to {h, s, t}. It is worth noting here that the hard fault attribute does not modify a generic fault in any way, which means that it is identical to the absence of an attribute (symbolized by -).

In contrast, the different attributes of voltage dependent faults *are* compatible and can be combined with each other. Therefore, the set of voltage dependent faults is equal to {-, p, d, pd}, where - stands for "no attribute", while pd stands for the combined attribute "partial dirty". It is important to note here that there are three different combinations of partial faults (p): the initialization related partial faults (p_i), the activation related partial faults (p_a), and the initialization and activation related partial fault (p_{ia}).

In conclusion, Expression 1 can be expanded to describe all possible DRAM-specific faults as follows:

$$\text{Fault} = \left\{ \begin{array}{c} \text{-} \\ \text{p} \\ \text{d} \\ \text{pd} \end{array} \right\} \left\{ \begin{array}{c} \text{h or -} \\ \text{s} \\ \text{t} \end{array} \right\} \text{FP} \qquad (2)$$

Expression 2 indicates that any generic fault model can either be regular (-), partial (p), dirty (d) or partial dirty (pd), while being hard (h or -), soft (s) or transient (t) at the same time. In total, this gives a space of $4 \times 3 = 12$ different attributes for DRAM-specific faults.

Based on the results of an elaborate experiment, using defect injection and Spice simulation in a number of real DRAM simulation models [Al-Ars05], performed to analyze the faulty behavior of DRAMs, the following realistic restrictions on the space of DRAM faults have been identified:

1. Restrictions for single-cell faults.
(a) State faults may not be partial.
(b) Single-cell faults can suffer from initialization partial faults (p_i), but not activation partial faults (p_a).
2. Restrictions for two-cell faults.
(a) State coupling faults may not be partial, as these have not been observed in practice.
(b) Coupling faults may not suffer from activation related partial faults (p_a), in case the sensitizing operation is performed on the victim.
(c) Only the aggressor in a coupling fault may suffer from initialization related partial faults (p_i), since the aggressor is the cell that causes the fault.
(d) Coupling faults may not be dirty.
3. General restrictions.
The completing operation $[C]$ of dirty faults (d) can either be a write or a read operation performed on a cell a along the same bit line of the faulty cell v, but with the opposite data to the sensitizing operation on v.

As an example, Table 4 shows the realistic DRAM fault space for the down transition fault (TF_0). Fault #1 in the

table is the hTF_0, which is identical to the generic TF_0. Fault #2 is the partial hard TF_0, which is denoted by the FP $<w1^h\ w0/1/->$. The h (hammer) in the sequence $w1^h$ is caused by the initialization related partial fault (p_i), and it stands for the number of times the initializing $w1$ operation should be performed ($h \geq 0$). Fault #3 is the dirty hard TF_0, which is obtained by adding a completing sequence of operations ($[C]$) to the sensitizing operation sequence (S). Fault #4 in the table is the partial dirty hard TF_0, which is denoted by the FP $<w1^h\ w0\ [C]/1/->$. This fault contains h initializing $w1$ operations, an activating $w0$ operation, in addition to the completing operation sequence $[C]$.

Table 4. Realistic space of DRAM-specific faults for TF_0.

#	Fault	FP	Name
1	hTF_0	$<1w0/1/->$	hard TF0
2	p_ihTF_0	$<w1^hw0/1/->$	partial hard TF0
3	$dhTF_0$	$<1w0[C]/1/->$	dirty hard TF0
4	p_idhTF_0	$<w1^hw0[C]/1/->$	partial dirty hard TF0
5	sTF_0	$<1w0_T/1/->$	soft TF0
6	p_isTF_0	$<w1^hw0_T/1/->$	partial soft TF0
7	$dsTF_0$	$<1w0[C]_T/1/->$	dirty soft TF0
8	p_idsTF_0	$<w1^hw0[C]_T/1/->$	partial dirty soft TF0
9	tTF_0	$<1\underline{w0}/1_L/->$	transient TF0
10	p_itTF_0	$<\underline{w1}^h\underline{w0}/1_L/->$	partial transient TF0
11	$dtTF_0$	$<1\underline{w0}[\underline{C}]/1_L/->$	dirty transient TF0
12	p_idtTF_0	$<\underline{w1}^h\underline{w0}[\underline{C}]/1_L/->$	partial dirty transient TF0

In summary, the difference between realistic single and two-cell DRAM-specific faults can be represented in the following two expressions.

$$\text{Single-cell fault} = \left\{ \begin{array}{c} - \\ p_i \\ d \\ p_i d \end{array} \right\} \left\{ \begin{array}{c} \text{h or -} \\ s \\ t \end{array} \right\} \text{FP} \qquad (3)$$

$$\text{Two-cell fault} = \left\{ \begin{array}{c} - \\ p \end{array} \right\} \left\{ \begin{array}{c} \text{h or -} \\ s \\ t \end{array} \right\} \text{FP} \qquad (4)$$

These expressions indicate that there are fewer realistic restrictions on single-cell faults than there are on two-cell faults. Single-cell faults can be attributed as partial, dirty and partial dirty, while two-cell faults can only be attributed as partial.

4 DRAM-specific tests

In this section, we use the space of DRAM-specific faults derived in the previous section to derive tests that would detect any possible hard DRAM fault. In the following, we discuss the needed detection conditions first, and then we list the tests to detect the faults.

Detection conditions for hard faults

A single-cell hard fault can either be partial with respect to initialization (p_ih), dirty (dh), or both (p_idh). The fault p_ih is modeled by multiple initializing operations, while the fault dh is modeled by performing a write or read operation on a cell along the same BL as the faulty cell, but with opposite data to the sensitization.

Table 5 lists all single-cell hard faults, along with the detection conditions needed to detect them [compare with Table 1]. The table considers the general form of single-cell hard faults, where both partial, as well as dirty faults take place. The detection conditions are designed to detect both faults as well. For example, the (partial, dirty and hard) transition 0 fault ($p_i$dh TF_0), must first be initialized a multiple number of times ($w1_v^h$). Then, the sensitizing write 0 operation should be performed ($w0_v$), before a completing operation with data 1 (a value opposite to that of the sensitizing value) must be applied to a different cell along the same BL ($[O1_b]$). The detection condition starts with multiple $w1$ operations to initialize the cell to 1, followed by the sensitizing $w0$ operation on the victim. Then, the operation $O1_b$ ensures that the opposite data is present in a cell along the same BL just before the fault is detected by the read operation.

Table 5. List of single-cell, hard FPs and their detection conditions. The completing operation Ox_b is performed with a value (x) opposite to that in the sensitizing operation and to a different cell (b) along the same BL.

#	Fault	$<S/F/R>, O \in \{w,r\}$	Detection cond., $O \in \{w,r\}$
1	dh SF_0	$<0_v[O1_b]/1/->$	$\Updownarrow(..w0,..O1_b,..r0..)$
2	dh SF_1	$<1_v[O0_b]/0/->$	$\Updownarrow(..w1,..O0_b,..r1..)$
3	$p_i$dh WDF_0	$<w0_v^h[O1_b]/1/->$	$\Updownarrow(..w0^h,..O1_b,..r0..)$
4	$p_i$dh WDF_1	$<w1_v^h[O0_b]/0/->$	$\Updownarrow(..w1^h,..O0_b,..r1..)$
5	$p_i$dh TF_1	$<w0_v^hw1_v[O0_b]/0/->$	$\Updownarrow(..w0^h,..w1,..O0_b,..r1..)$
6	$p_i$dh TF_0	$<w1_v^hw0_v[O1_b]/1/->$	$\Updownarrow(..w1^h,..w0,..O1_b,..r0..)$
7	$p_i$dh IRF_0	$<w0_v^h[O1_b]r0_v/0/1>$	$\Updownarrow(..w0^h,..O1_b,..r0..)$
8	$p_i$dh IRF_1	$<w1_v^h[O0_b]r1_v/1/0>$	$\Updownarrow(..w1^h,..O0_b,..r1..)$
9	$p_i$dh $DRDF_0$	$<w0_v^hr0_v[O1_b]/1/0>$	$\Updownarrow(..w0^h,..r0,..O1_b,..r0..)$
10	$p_i$dh $DRDF_1$	$<w1_v^hr1_v[O0_b]/0/1>$	$\Updownarrow(..w1^h,..r1,..O0_b,..r1..)$
11	$p_i$dh RDF_0	$<w0_v^h[O1_b]r0_v/1/1>$	$\Updownarrow(..w0^h,..O1_b,..r0..)$
12	$p_i$dh RDF_1	$<w1_v^h[O0_b]r1_v/0/0>$	$\Updownarrow(..w1^h,..O0_b,..r1..)$

In a similar way, one can derive the detection conditions needed to detect all two-cell, hard faults.

Tests for hard faults

Based on the detection conditions of hard faults, it is possible to derive memory tests that detect all single-cell and two-cell hard faults. March H1C below detects all single-cell hard faults.

March H1C = {	
$\Updownarrow(w0^h, r0, w1_b, r0)$;	$\Updownarrow(w1^h, r1, w0_b, r1)$;
ME0	ME1
$\Updownarrow(w0^h, w1, w0_b, r1)$;	$\Updownarrow(w1^h, w0, w1_b, r0)$}
ME2	ME3

This march test has four march elements (ME0 through ME3), each of which begins with a hammer write operation and ends with a detecting read operation. Each two consecutive march elements represent the exact complement of each other, as they are generated to target complementary FPs. The test substitutes the dirty operation (O) in the detection conditions of Table 5 by a write operation, since this choice reduces the length of the test when the completing operation needs to change the data present in b. Note that the test uses a special kind of march operations (Ox_b), where an operations is performed to a different cell within a given march element. The test has a complexity of $(12 \cdot n + 4 \cdot h \cdot n)$. For an h value of about 5, this test has a total complexity of $(32n)$, which is a relatively high complexity when compared to other single-cell march tests. This is caused by the need to detect the partial and the dirty DRAM-specific faults.

Table 6 lists all march elements in March H1C along with the hard single-cell FPs, and indicates the first memory operation in the test that detects the corresponding FP. For example, ME0 (march element 0) shares an entry $\#1/3$ with dh SF. This entry means that FP $\#1$, which refers to the fault dh SF_0, is first detected in March H1C by the 3rd operation of ME0. The table shows that each of ME0 and ME1 detect 4 different FPs, while each of ME2 and ME3 detect a single FP.

Table 6. Detection capabilities of march elements in March H1C.

Fault	dh SF	$p_i$dh WDF	$p_i$dh TF	$p_i$dh IRF	$p_i$dh DRDF	$p_i$dh RDF
ME0	$\#1/4$	$\#3/4$	–	$\#7/4$	$\#9/4$	$\#11/4$
ME1	$\#2/4$	$\#4/4$	–	$\#8/4$	$\#10/4$	$\#12/4$
ME2	–	–	$\#5/4$	–	–	–
ME3	–	–	$\#6/4$	–	–	–

A march test that detects all two-cell hard faults can be represented by March H2C below.

March H2C = {		
$\Updownarrow(w0^h)$;	$\Uparrow(r0^h, w1^h)$;	$\Uparrow(r1^h, w0^h)$;
ME0	ME1	ME2
$\Downarrow(r0^h, w1^h)$;	$\Downarrow(r1^h, w0^h)$;	$\Updownarrow(r0)$}
ME3	ME4	ME5

This march test has 6 march elements (ME0 through ME5), many of which begin with a hammer read operation and end with a hammer write operation. These sequences are characteristic for march tests that aim to detect two-cell faults [Harutunvan05]. The march element ME1 is the exact complementary of ME2, while ME3 is the exact complementary of ME4. This results from the fact that these march elements are constructed to detect complementary FPs. This test has a complexity of $(n + 9 \cdot n \cdot h)$.

5 Conclusions

This paper presented the first realistic space of DRAM-specific faults, using a detailed Spice-based analysis of the faulty behavior of DRAMs. The space identifies five individual fault attributes, that can be combined with generic memory faults to describe the whole space of DRAM faulty behavior. It is shown that DRAM-specific faults are either the result of improperly set voltages (causing partial faults and dirty faults) or the result of leakage currents (causing soft faults and transient faults). The paper also derived new DRAM-specific tests to detect all (partial and dirty) single-cell and two-cell hard faults in practice.

References

[Adams96] R.D. Adams and E.S. Cooley, "Analysis of a Deceptive Destructive Read Memory Fault Model and Recommended Testing," *in Proc. IEEE North Atlantic Test Workshop*, 1996.

[Adler95] E. Adler *et al.*, "The Evolution of IBM CMOS DRAM Technology," *in IBM J. of Research and Development*, vol. 39, no. 1–2, 1995, pp. 167–188.

[Al-Ars05] Z. Al-Ars, *DRAM Fault Analysis and Test Generation*, PhD thesis, Delft Univ. of Technology, Delft, the Netherlands, 2005.

[Dekker90] R. Dekker *et al.*, "A Realistic Fault Model and Test Algorithms for Static Random Access Memories," *in IEEE Trans. on CAD*, vol. C-9, no. 6, 1990, pp. 567–572.

[Hamdioui04] S. Hamdioui, *Testing Static Random Access Memories: Defects, Fault Models and Test Patterns*, Kluwer Academic Publishers, Boston, MA, 2004.

[Harutunvan05] G. Harutunvan, V.A. Vardanian and Y. Zorian, "Minimal March Tests for Unlinked Static Faults in Random Access Memories," *in Proc. IEEE VLSI Test Symp.*, 2005, pp. 53–59.

[McConnell98] R. McConnell, U. Möller and D. Richter, "How we test Siemens' Embedded DRAM Cores," *in Proc. IEEE Int'l Test Conf.*, 1998, pp. 1120–1125.

[Vollrath00] J. Vollrath, "Tutorial: Synchronous Dynamic Memory Test Construction, A Field Approach," *in Proc. IEEE Int'l Workshop Memory Technology, Design and Testing*, 2000, pp. 59–64.