DRAM–Specific Space of Memory Tests

Zaid Al–Ars[1] Said Hamdioui[1] Ad van de Goor[1] Georgi Gaydadjiev[1] Joerg Vollrath[2]

[1]Delft University of Technology
Faculty of EE, Mathematics and CS
Laboratory of Computer Engineering
Mekelweg 4, 2628 CD Delft, The Netherlands

[2]Infineon Technologies AG
Memory Products Business Group
Product Engineering Department
Balanstr. 73, 81541 Munich, Germany

E–mail: z.e.al–ars@ewi.tudelft.nl

Abstract: *DRAM testing has always been theoretically considered as a subset of general memory testing, despite the disagreement of this assumption with the DRAM test practice. This paper presents a recently developed space of DRAM faults that describes the unique aspects of DRAM behavior, it validates this fault space using extensive Spice simulation, and it identifies the memory tests necessary to detect these faults. Six different tests are derived and shown to correspond to highly effective DRAM tests in practice.*

1 Introduction

DRAMs have traditionally played an important role as the main memory of the microprocessor, but they gradually find themselves serving in a continuously growing list of applications, in fields ranging from high performance to low power and from networking to graphics [Prince03]. DRAM testing is considered a complex and overly costly activity, requiring a time consuming test development cycle that should be repeated for every new DRAM technology [Falter00, Antonin91]. In order to reduce the cost of DRAM testing, memory tests should be developed specifically for DRAMs to target the faults commonly observed in these memories.

In this paper, we describe a new space of memory tests specifically developed for DRAMs, based on a long term analysis of their faulty behavior in the industry [Al–Ars05]. The tests target all DRAM-specific faults identified to be realistic for their behavior. Three different sets of tests are given: one for hard faults, one for transient faults and one for soft faults. In each set, two tests are given (one for single-cell and one for two-cell faults), making a total of 6 DRAM–specific tests. Some of the tests are also shown to correspond closely to some of the most effective DRAM tests derived in practice [vdGoor00].

The paper starts with a description of the concept of fault primitives in Section 2, followed by a definition of the different DRAM–specific fault models in Section 3. Section 4 shows the simulation model used to validate the fault space. The results of the simulation study are discussed in Section 5. Section 6 derives the space of memory tests suitable to detect the DRAM–specific faulty behavior, while Section 7 identifies the similarities these tests have with DRAM tests known to be effective in practice. Section 8 ends with the conclusions.

2 Fault primitives

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 operation sequence that sensitizes the fault, F describes the logic level in the faulty cell ($F \in \{0, 1\}$), 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.

Functional fault models (FFMs) can be defined as a non–empty set of FPs. The most important FFM classes 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).

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

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

3 DRAM-specific faults

DRAM faults can either be attributed to leakage currents (resulting in time dependent faults), or to improperly set voltages (resulting in voltage dependent faults) [Al-Ars06]. Figure 1 shows a summary of DRAM-specific faults.

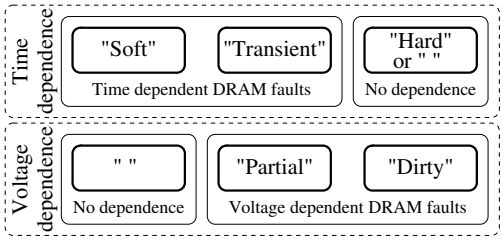
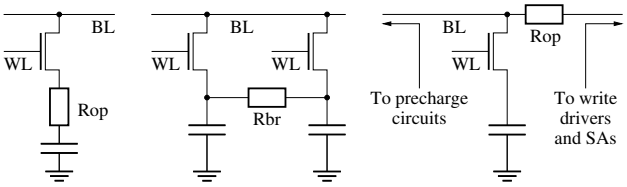


Figure 1. Summary of the space of DRAM-specific faults.

3.1 Time dependent faults

Time dependent faults are caused by leakage currents in faulty cells [Keshavarzi97]. Time dependence divides all faults into three classes: soft, transient and hard.

Soft faults—Soft faults (s) only become detectable after some time from their sensitization. These faults can be detected by adding a *delay* within the test, 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 are represented as sFP $=<S_T/F/R>$, where S has an added time parameter T to indicate that some time should elapse before full sensitization. The open defect in Figure 2(a) shows an open that may cause soft faults in a DRAM cell. 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 store a *weak voltage* into the cell. If leakage opposes the weak voltage, the stored information gets lost over time.



(a) s, t & piFP (b) paFP (c) dFP

Figure 2. Defects causing (a) s, t, p_i, (b) p_a, and (c) causing dirty faults.

Transient faults—Transient faults (t) 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 directly following them by a read. The DRAM open in Figure 2(a) may cause transient faults. 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. If leakage tends to correct the weak faulty voltage, the stored voltage gets corrected over time. Transient faults are represented as tFP $=<\underline{S}/F_L/R>$, where the underscore below S means that the operations in S should be performed in back-to-back mode. Furthermore, F has an added time parameter L (*life time*) to indicate that these faults are time limited. An underscore below operations implies 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 $\Updownarrow(..., w1, \underline{w0}, ...)$.

Hard faults—Identifying a fault as being hard ("h" or "–") indicates that it is neither soft nor transient (i.e., it is insensitive to time). All the generic faults described in Section 2 are hard faults.

3.2 Voltage dependent faults

Operations performed on a defective DRAM may set improper voltage levels on memory nodes, thereby causing two types of DRAM faults: partial faults and dirty faults.

Partial faults—Partial faults (p) 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 (*partial faults during initialization* p_i), or to properly sensitize the fault in the cell (*partial faults during fault sensitization or activation* p_a). Figure 2(a) shows an example of an open (R_{op}) in the cell, causing p_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 2(b) shows an example of a bridge (R_{br}) between two cells, causing p_a. These faults are modeled by performing an operation Ox an h (or *hammer*) number of times. For example, if $<xOy/F/R>$ becomes partial during initialization p_i, it should be modeled as p_iFP $=<x^hOy/F/R>$.

Dirty faults—Dirty faults (d) 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 detectability, additional operations (so called *completing operations*) must be performed to correct the corrupted state of the memory. Figure 2(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, which prevents the memory from writing the cells. This defect also prevents properly precharging the BL. As a result, a $w0$ operation that fails to write 0 in the cell ends up preconditioning the BL to properly sense a 0, thereby causing a dirty fault. These fault are modeled by adding completing operations in square brackets to the FP ($[Ox]$). Detectability of all known dirty faults can be ensured using a completing write operation with data opposite to the data in the victim ($[w\overline{y}]$), performed on a cell (b) different from the victim (v) but positioned on the same BL pair. This can be denoted as dFP $=<xO_vy\ [w_b\overline{y}]/\overline{y}/->_{b,v\in \mathrm{BL}}$.

3.3 Realistic space of DRAM faults

Any generic memory fault, described in Section 2, can represent a DRAM-specific fault by adding DRAM-specific fault attributes to it. First, there are voltage dependent attributes: partial (p), dirty (d), or neither (–). Second, there are time dependent attributes: hard (h or –), soft (s) and transient (t). Furthermore, the partial attribute can either be initialization related (p_i), or activation (or sensitization) related (p_a).

Based on a detailed analysis of the characteristics of these faults, the full realistic space of DRAM faults can be constructed for singe-cell faults, as well as two-cell faults [Al-Ars05].

$$\text{Single-cell fault} = \left\{ \begin{array}{c} - \\ p_i \\ d \\ p_i d \end{array} \right\} \left\{ \begin{array}{c} \text{h or } - \\ s \\ t \end{array} \right\} \text{FP} \quad (1)$$

$$\text{Two-cell fault} = \left\{ \begin{array}{c} - \\ p \end{array} \right\} \left\{ \begin{array}{c} \text{h or } - \\ s \\ t \end{array} \right\} \text{FP} \quad (2)$$

These expressions indicate that any generic single-cell fault can either be regular (–), initialization partial (p_i), dirty (d) or partial dirty (p_id), while being hard (h or –), soft (s) or transient (t) at the same time. Two-cell faults can regular (–) or partial (p), while being hard, soft or transient. Note that some faults classes are considered unrealistic, such as activation partial (p_a) single-cell faults, and therefore they are not included in the space. More info about these unrealistic faults can be found in the literature [Al-Ars05, Al-Ars06].

For example, a transition 0 fault can be hard (hTF_0), which is the same as the generic TF_0. It can also be partial hard (p_ihTF_0), dirty hard ($dhTF_0$) and partial dirty hard fault (p_idhTF_0). The same combinations apply for soft TF_0 and transient TF_0.

4 Fault model validation using Spice

In order to validate the theoretical framework of DRAM faults discussed in Section 3, an elaborate Spice-based study have been carried out in an industrial setting at Infineon Technologies on real memory simulation models, manufactured in a number of different technologies, ranging from $0.35\ \mu$m to $0.11\ \mu$m feature sizes. This section shows the simulation results performed on a memory model manufactured in a $0.2\ \mu$m technology, by injecting resistive defects into the model and subsequently simulating them. This approach has been successfully used in the past to analyze the faulty behavior of both digital and analog devices [Naik93, Nagi96].

4.1 Memory simulation model

The model used here is a based on a design validation Spice model. The model has been reduced in size to limit simulation time. Figure 3 shows the different parts of the memory model. The model has 3 BL pairs[1], one at the top (BTt and BCt), one in the middle (BTm and BCm), and one at the bottom (BTb and BCb). Each BL pair contains 2 functional memory cells, while the rest of the cells are replaced by capacitive and resistive loads. This gives a total of 6 memory

[1]Three BL pairs are used to simulate BL coupling effects, but these effects are not discussed further in this paper [Al-Ars05].

cells, three of which are connected to a true BL and controlled by the word line WLt, while the other three are connected to a complement BL and controlled by WLc. The model also contains 3 sense amplifiers (SAs), 3 precharge circuits, access devices, one read buffer (to inspect the output of a read operation), and one write buffer (to enable writing the cells).

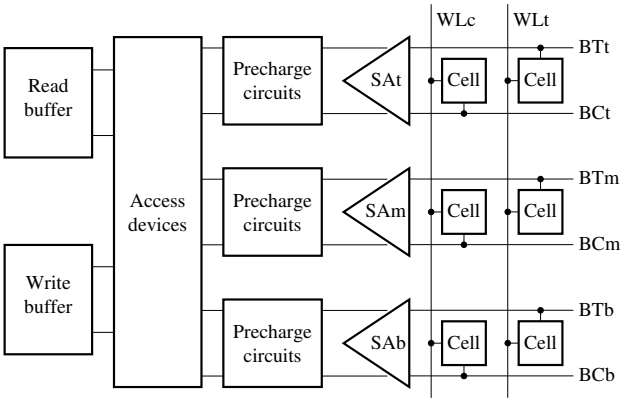


Figure 3. Reduced memory simulation model used for fault analysis.

4.2 Classification of defects

Depending on the signal lines the injected defects are connected to, the defects may be classified into the following three categories:

- **Open**—Opens represent unwanted resistances on a signal line that is supposed to conduct perfectly.
- **Short**—Shorts are undesired resistive paths between a signal line and power supply (V_{dd} or GND).
- **Bridge**—Bridges are unwanted resistive paths between two signal lines.

The faulty behavior resulting from many defects can be deduced using symmetrical relationships with other defects. Therefore only symmetrically unrelated defects are simulated. Figure 4 shows an overview of the locations of the defects to be simulated. The defects are classified into opens, shorts and bridges. The opens and shorts are divided into defects in memory cells, on BLs or on WLs, while bridges are divided into defects involving no cells, one cell and two different cells. Table 3 gives a summary of the simulated defects and the simulation results.

Location of opens

At the layout level, opens are usually caused by broken lines, or particle contamination that results in increasing line resistivity at the open position. Figure 5 shows a layout level example of a BL open caused by particle contamination, resulting in an increase in the BL resistance and inducing some kind of faulty behavior in the memory.

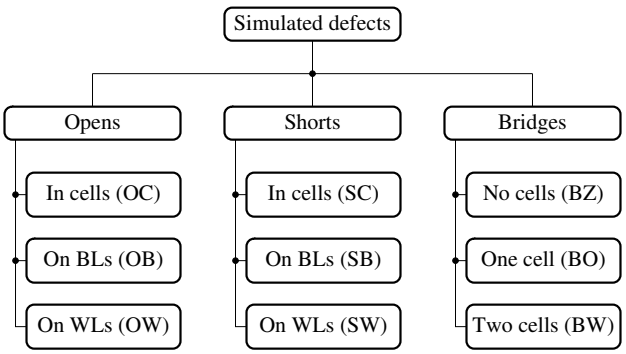


Figure 4. Overview of simulated defect locations.

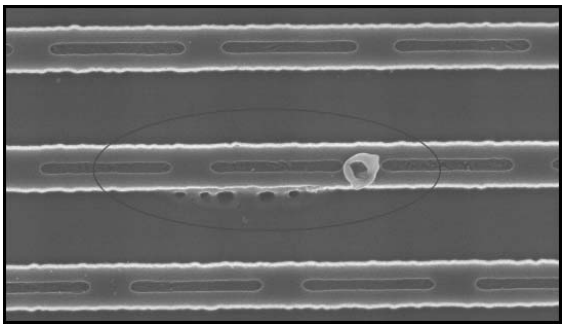


Figure 5. Layout level example of a BL open.

Opens in the memory cell array can be either opens within cells (OC), opens on BLs (OB) or opens on WLs (OW). Figure 6 shows these three different locations of opens. There are three types of OC, on top (t), in the middle (m) and at the bottom (b), all of which (partially) disconnect the cell from the BL and limit the ability of the memory to write and read the voltage within the cell. There are two types of OB, disconnecting the cell from circuitry on the top of the BL (t), and disconnecting the cell from circuitry at the bottom of a the BL (b). There is only one word line open called OW, which limits the ability of the memory to properly access the cell.

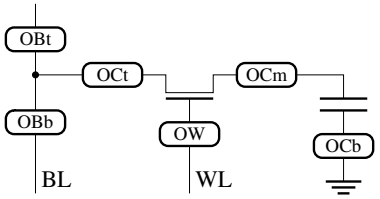


Figure 6. Location of opens in the cell array.

Location of shorts

At the layout level, shorts can be caused by a number of physical failures, such as extra metal deposition or isolation breakdown. This results in faulty connections between

power supply lines and other signal lines in the memory. Figure 7 shows a layout example of a power supply short caused by extra metal deposition, resulting in a faulty new connection being formed between two otherwise disconnected lines.

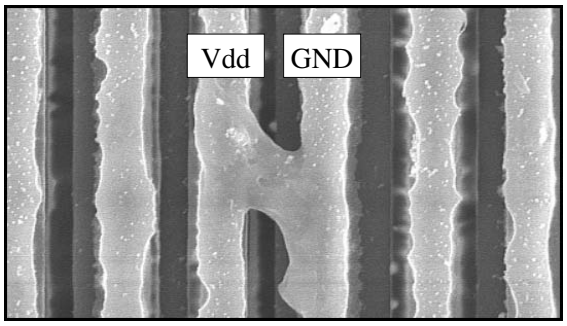


Figure 7. Physical example of power supply shorts.

Shorts, similar to opens, can be either within cells (SC), on bit lines (SB), or on word lines (SW). Figure 8 shows the short positions of these three different types of shorts. At each position indicated in the figure, a short may connect the node either to V_{dd} or GND. Shorts to V_{dd} are indicated by the letter (v) as in SCv and SBv, while shorts to GND are indicated by the letter (g) as in SCg and SBg. BL in the figure can either be the true bit line (BT), or the complementary bit line (BC).

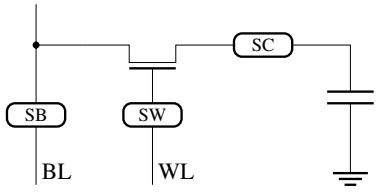


Figure 8. Positions of shorts in the cell array.

Location of bridges

At the layout level, bridges can be the result of isolation layer misalignment or mask contamination, resulting in faulty connections between different lines in the memory. Figure 9 shows a layout example of a bridge between the WL and the BL contacts caused by a misalignment in the isolation layer. The small black structure in the middle of the figure represents the WL contact, which is sandwiched between two relatively large BL contacts. The WL and BL contacts are separated from each other by a thin isolation layer, that is difficult to manufacture. The figure shows that the isolation layer is not created properly, leaving a small space on top, through which the BL contact stretches and connects to the WL.

At the electrical level, bridges are resistive connections between two nodes in the memory. In order to take all cell array bridges into consideration, we need to consider

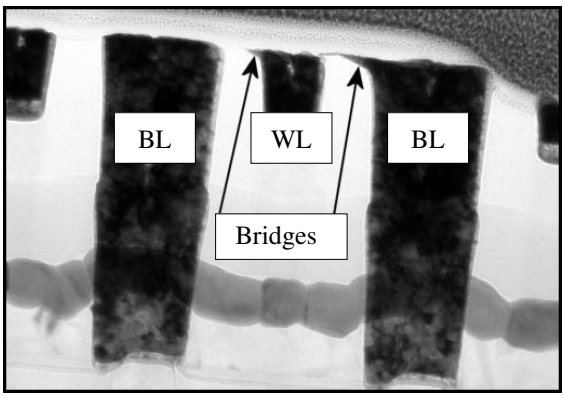


Figure 9. Physical example of a BL-WL bridge.

bridges between any two nodes in the cell array. Since the cell array has a repetitive structure, it is possible to consider only the region surrounding a single cell in the array as a representative of the whole. Such a representative part is shown in Figure 10, where all nodes are given names for later reference.

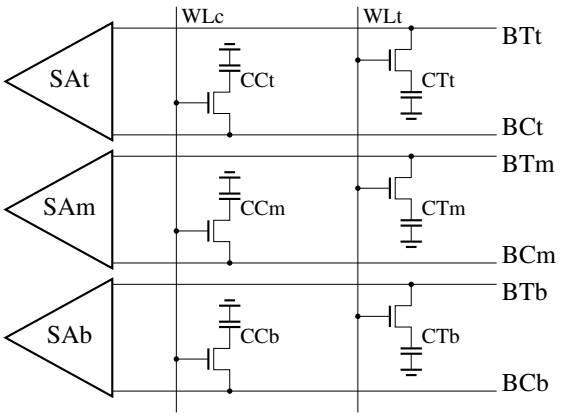


Figure 10. Nodes where bridges can take place.

The figure shows three BL pairs, at the top (BTt and BCt), in the middle (BTm and BCm), and at the bottom (BTb and BCb). Each BL pair has two memory cells connected to it, giving a total of six cells, three on BT (with nodes CTt, CTm and CTb) and three on BC (with nodes CCt, CCm and CCb). The cells connected to BT are controlled by the true WL (WLt), while the cells connected to BC are controlled by the complement WL (WLc).

We can classify bridges into those involving zero cells (BZ), one cell (BO) and two cells (BW). BZ are bridges that connect nodes outside memory cells, rather than connecting nodes within memory cells. BO are bridges that connect nodes within a single memory cell. BW are bridges that connect nodes belonging to two different memory cells. A summary of the simulated defects is given in Table 3.

Table 3. Definitions of injected defects and the simulation results of the faulty behavior.

Defect	FP	h	ph	dh	pdh	s	ps	ds	pds	t	pt	dt	pdt	R_c	Defect definition
OCt	$<w1^3w0/1/->$	−	+	−	−	−	+	−	−	−	+	−	−	331 kΩ	Pass transistor disconnected from BL
OCm	$<w1^3w0/1/->$	−	+	−	−	−	+	−	−	−	+	−	−	323 kΩ	Pass transistor disconnected from capacitor
OCb	$<w1^3w0/1/->$	−	+	−	−	−	+	−	−	−	+	−	−	323 kΩ	Cell disconnected from GND
OBt	$<w0^2w1/0/->$	−	+	−	−	−	+	−	−	−	+	−	−	50 kΩ	Cell disconnected from read/write circuits
OBb*	$<0w_v1^2[w_b0]/0/->$	−	+	+	+	−	+	+	+	−	+	+	+	8.5 kΩ	Cell disconnected from precharge devices*
OW	$<w0/1/->$	+	−	−	−	+	−	−	−	+	−	−	−	24.6 MΩ	WL disconnected from cell
SCv	$<0/1/->$	+	−	−	−	+	−	−	−	−	−	−	−	896 kΩ	Cell capacitor shorted to V_{dd}
SCg	$<1/0/->$	+	−	−	−	+	−	−	−	−	−	−	−	693 kΩ	Cell capacitor shorted to GND
SBv	$<w1^2w0/1/->$	−	+	−	−	−	+	−	−	−	+	−	−	250 kΩ	BL shorted to V_{dd}
SBg	$<0w1/0/->$	+	−	−	−	+	−	−	−	+	−	−	−	190 kΩ	BL shorted to GND
SWv	$<0/1/->$	+	−	−	−	+	−	−	−	−	−	−	−	0.34 kΩ	WL shorted to V_{dd}
SWg	$<w1^3w0/1/->$	−	+	−	−	−	+	−	−	−	+	−	−	8.29 kΩ	WL shorted to GND
BO1	$<w1^2w0/1/->$	−	+	−	−	−	−	−	−	−	+	−	−	357 kΩ	Cell bridged to own WL
BO2	$<1/0/->$	+	−	−	−	+	−	−	−	−	−	−	−	693 kΩ	Cell bridged to idle WL
BO3	$<0/1/->$	+	−	−	−	+	−	−	−	−	−	−	−	150 kΩ	Cell bridged to own BL
BO4	$<0/1/->$	+	−	−	−	+	−	−	−	−	−	−	−	400 kΩ	Cell bridged to own comp. BL
BO5	$<0/1/->$	+	−	−	−	+	−	−	−	−	−	−	−	300 kΩ	Cell bridged to another comp. BL
BZ1	$<w0/1/->$	+	−	−	−	+	−	−	−	+	−	−	−	1.71 kΩ	Bridge between BTm and BCm
BZ2	$<0w1/0/->$	+	−	−	−	+	−	−	−	+	−	−	−	4 kΩ	Bridge between BTm and BCt
BZ3	$<0w1/0/->$	+	−	−	−	+	−	−	−	+	−	−	−	0.6 kΩ	Bridge between BCm and BTb
BZ4	$<1w0/1/->$	+	−	−	−	+	−	−	−	+	−	−	−	270 kΩ	Bridge between BTm and WLt
BZ5	$<0w1/0/->$	+	−	−	−	+	−	−	−	+	−	−	−	250 kΩ	Bridge between BCm and WLt
BZ6	$<0w1/0/->$	+	−	−	−	+	−	−	−	+	−	−	−	0.15 kΩ	Bridge between WLt and WLc
BW1	$<w1^2;0/1/->$	−	+	−	−	−	+	−	−	−	+	−	−	3 MΩ	Bridge between CTm and CCm
BW2	$<w0^2w1/0/->$	−	+	−	−	−	+	−	−	−	+	−	−	460 kΩ	Bridge between CTm and CTb

*This defect is injected and simulated in a memory model different from that shown in Figure 3. The model used here is shown in Figure 2(c), where the precharge circuits are located on one end of the BL, while read/write circuits are located on the other end. This BL design was common in older DRAM technologies [Prince91].

5 Simulation results

The simulation and analysis of the faulty behavior of the DRAM has been done using the concept of result planes, as described in the literature [Al–Ars05, Al–Ars02]. The simulation results for each defect are listed in Table 3. The first column in the table lists the name of the simulated defect, followed by the FP notation to describe the resulting faulty behavior in the third column. The following 12 columns indicate the type of the DRAM–specific fault being modeled by the FP. There are five basic types of faults: hard (h), soft (s), transient (t), partial (p) and dirty (d). Together, they combine to make up 12 possible fault combinations [see Section 3]. The R_c column describes the value of the *critical resistance* of the fault, which is the defect resistance at which the fault begins to take effect. The final column gives a description of the defect being simulated.

The table shows that *all* the DRAM–specific fault models defined in Section 3 do actually take place for one defect or another. This indicates that DRAM–specific fault models are realistic and do take place in practice. The table also shows that, in general, if a defect causes a hard fault to take place, then it might also cause a corresponding soft and transient fault depending on the direction and strength of the leakage current flowing into the cell. This is however not always true, especially when the defect itself forces current to lake into the cell in a specific direction. This is the case, for example, with cell shorts and cell bridges to BLs and WLs.

Note that the least represented DRAM–specific faults in the table are the dirty (d) and the partial dirty (pd) faults. This is true because in order for these faults to take place, the BLs should be designed in the way shown in Figure 2(c), where the precharge circuits are located on one end of the BL, while the read/write circuits are located on the other end. This BL design was common in older DRAM technologies [Prince91]. The simulation results for the defect OBb listed in Table 3 correspond to a simulation performed on such an old model that belongs to a 0.35 μm technology. The simulation results of the OBb defect injected into the 0.2 μm memory model used for the rest of table has not resulted in any faulty behavior.

The results in the table make it possible to optimize the test strategy of the memory depending on the DRAM design and the defects that take place there. If, for example, the structure of the BL is known to have both precharge circuits and read/write circuits located on the same side of the BL, then it is not needed to test for dirty faults in the memory.

6 Space of DRAM tests

In this section, we derive the space of DRAM tests for (single–cell as well as two–cell) hard, then transient, and finally soft faults. The test are presented in their general form, without any optimization or reduction in test time.

6.1 Detecting hard faults

Hard faults are time–independent faults that get sensitized once their sensitizing operation sequence is performed, and they remain sensitized afterwards until they get overwritten or otherwise detected. Next, we discuss the detection conditions needed, followed by the corresponding tests.

Detection conditions for hard faults

A single–cell hard fault can either be partial with respect to initialization (p_ih), dirty (dh), or both ($p_i$dh). 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 4 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. 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, it should be sensitized by $w0$, 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 only requirement the completing write operation has to fulfill is to change the state of the BLs connected to the victim cell. The exact written cell address is therefore unimportant, only the fact that it lies along the same BL. The detection condition starts with multiple $w1$ operations to initialize the cell, 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.

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 single–cell and two–cell hard faults, it is possible to derive memory tests that detect all single–cell and two–cell hard faults. March H1C (for hard, 1–cell) below detects all single–cell hard faults.

Table 4. Single–cell, hard FPs and their detection conditions. Ox_b is performed with a value (x) opposite to that in the sensitizing operation and to a cell (b) different from v, but along the same BL as v.

#	Fault	FP (b on same BL as v)	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	$<w0_v^h w1_v[O0_b]/0/->$	$\Updownarrow(..w0^h, ..w1, ..O0_b, ..r1..)$
6	$p_i$dh TF_0	$<w1_v^h w0_v[O1_b]/1/->$	$\Updownarrow(..w1^h, ..w0, ..O1_b, ..r0..)$
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 $DRDF_0$	$<w0_v^h r0_v[O1_b]/1/0>$	$\Updownarrow(..w0^h, ..r0, ..O1_b, ..r0..)$
10	$p_i$dh $DRDF_1$	$<w1_v^h r1_v[O0_b]/0/1>$	$\Updownarrow(..w1^h, ..r1, ..O0_b, ..r1..)$
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	$<w1_v^h[O0_b]r1_v/0/0>$	$\Updownarrow(..w1^h, ..O0_b, ..r1..)$

March H1C = {
$\Updownarrow(w0^h, r0, w1_b, r0)$; ME0
$\Updownarrow(w1^h, r1, w0_b, r1)$; ME1
$\Updownarrow(w0^h, w1, w0_b, r1)$; ME2
$\Updownarrow(w1^h, w0, w1_b, r0)$} 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 4 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 operation is performed to a different cell within a given march element. The test has a relatively high complexity of $(12 \cdot n + 4 \cdot h \cdot n)$ compared to other single–cell march tests, as a result of the partial and the dirty DRAM–specific faults.

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

March H2C = {
$\Updownarrow(w0^h)$; ME0
$\Uparrow(r0^h, w1^h)$; ME1
$\Uparrow(r1^h, w0^h)$; ME2
$\Downarrow(r0^h, w1^h)$; ME3
$\Downarrow(r1^h, w0^h)$; ME4
$\Updownarrow(r0)$} 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 $(n + 9 \cdot h \cdot n)$.

6.2 Detecting transient faults

Transient FPs mean that, after a fault is sensitized, leakage results in correcting the fault before it gets detected.

Detection conditions for transient faults

An FP has two components to describe a fault: F (the value of the faulty cell) and R (the output on a read operation). Only F can be transient (get corrected by leakage), whereas R cannot, since it gets sensitized *and* detected on the output at the same time.

Table 5. List of single-cell, transient FPs and their detection conditions. The underlined operations must be performed back-to-back.

#	Fault	$<S/F_L/R>, O \in \{w, r\}$	Detection cond., $O \in \{w, r\}$
1	dt SF_0	$<0_v[\underline{O1}_b]/1_L/->$	$\Updownarrow(..., w0, \underline{O1}_b, \underline{r0}, ...)$
2	dt SF_1	$<1_v[\underline{O0}_b]/0_L/->$	$\Updownarrow(..., w1, \underline{O0}_b, \underline{r1}, ...)$
3	$p_i$dt WDF_0	$<\underline{w0}^h_v[\underline{O1}_b]/1_L/->$	$\Updownarrow(..., \underline{w0}^h, \underline{O1}_b, \underline{r0}, ...)$
4	$p_i$dt WDF_1	$<\underline{w1}^h_v[\underline{O0}_b]/0_L/->$	$\Updownarrow(..., \underline{w1}^h, \underline{O0}_b, \underline{r1}, ...)$
5	$p_i$dt TF_1	$<\underline{w0}^h_v\underline{w1}_v[\underline{O0}_b]/0_L/->$	$\Updownarrow(..., \underline{w0}^h, \underline{w1}, \underline{O0}_b, \underline{r1}, ...)$
6	$p_i$dt TF_0	$<\underline{w1}^h_v\underline{w0}_v[\underline{O1}_b]/1_L/->$	$\Updownarrow(..., \underline{w1}^h, \underline{w0}, \underline{O1}_b, \underline{r0}, ...)$
7	$p_i$dt IRF_0	$<\underline{w0}^h_v[\underline{O1}_b]\underline{r0}_v/0_L/1>$	$\Updownarrow(..., \underline{w0}^h, \underline{O1}_b, \underline{r0}, ...)$
8	$p_i$dt IRF_1	$<\underline{w1}^h_v[\underline{O0}_b]\underline{r1}_v/1_L/0>$	$\Updownarrow(..., \underline{w1}^h, \underline{O0}_b, \underline{r1}, ...)$
9	$p_i$dt $DRDF_0$	$<\underline{w0}^h_v\underline{r0}_v[\underline{O1}_b]/1_L/0>$	$\Updownarrow(..., \underline{w0}^h, \underline{r0}, \underline{O1}_b, \underline{r0}, ...)$
10	$p_i$dt $DRDF_1$	$<\underline{w1}^h_v\underline{r1}_v[\underline{O0}_b]/0_L/1>$	$\Updownarrow(..., \underline{w1}^h, \underline{r1}, \underline{O0}_b, \underline{r1}, ...)$
11	$p_i$dt RDF_0	$<\underline{w0}^h_v[\underline{O1}_b]\underline{r0}_v/1_L/1>$	$\Updownarrow(..., \underline{w0}^h, \underline{O1}_b, \underline{r0}, ...)$
12	$p_i$dt RDF_1	$<\underline{w1}^h_v[\underline{O0}_b]\underline{r1}_v/0_L/0>$	$\Updownarrow(..., \underline{w1}^h, \underline{O0}_b, \underline{r1}, ...)$

Table 5 lists all single-cell transient faults, along with their detection conditions [compare with Table 4]. For example, the (partial, dirty and transient) transition 0 fault ($p_i$dt TF_0), must first be initialized a multiple number of times ($w1^h$). Then, the sensitizing write 0 operation can be performed ($w0$), before a completing operation with data 1 must be applied to a different cell along the same BL as v ($[O1_b]$). The detection condition starts with multiple $w1$ operations to initialize the cell to 1, directly followed by the sensitizing $w0$, the completing $O1_b$, and a detecting $r0$. Note that this detection condition is not a regular one, since it requires operations to be performed on two different cells (b and v) within only one march element. The fact that the operations in these detection conditions need to be performed back-to-back is indicated by the underline below the corresponding operations.

In the same way, one may derive the detection conditions corresponding to all two-cell, hard faults.

Tests for transient faults

Based on the detection conditions of single-cell and two-cell transient faults, it is possible to derive memory tests that detect all these faults. A march test that detects all single-cell transient faults can be represented by March T1C (for transient, 1-cell) below.

March T1C = {

$\Updownarrow(\underline{w0}^h, \underline{w1}_b, \underline{r0})$; ME0

$\Updownarrow(\underline{w1}^h, \underline{w0}_b, \underline{r1})$; ME1

$\Updownarrow(\underline{w0}^h, \underline{w1}, \underline{w0}_b, \underline{r1})$; ME2

$\Updownarrow(\underline{w1}^h, \underline{w0}, \underline{w1}_b, \underline{r0})$; ME3

$\Updownarrow(\underline{w0}^h, \underline{r0}, \underline{w1}_b, \underline{r0})$; ME4

$\Updownarrow(\underline{w1}^h, \underline{r1}, \underline{w0}_b, \underline{r1})\}$ ME5

This march test has six march elements (ME0 through ME5), each of which begins with a hammer write operation and ends with a detecting read operation. This test has a complexity of $(16 \cdot n + 6 \cdot h \cdot n)$. The march elements have special operations (such as $w1_b$) to be performed on a different cell along the same BL as the current cell of the march element. The operations in each march element must be performed back-to-back directly after each other (hence the underline below the operations in the test).

A march test that detects all two-cell, transient faults can be represented by March T2C below.

March T2C = {

$\Updownarrow_i(\Updownarrow_j(w0_i, \underline{w0}^h_j, \underline{r0}_i, \underline{r0}_i))$; ME0

$\Updownarrow_i(\Updownarrow_j(w0_i, \underline{w1}^h_j, \underline{r0}_i, \underline{r0}_i))$; ME1

$\Updownarrow_i(\Updownarrow_j(w1_i, \underline{w0}^h_j, \underline{r1}_i, \underline{r1}_i))$; ME2

$\Updownarrow_i(\Updownarrow_j(w1_i, \underline{w1}^h_j, \underline{r1}_i, \underline{r1}_i))$; ME3

$\Updownarrow_i(\Updownarrow_j(w0_i, \underline{w0}_j, \underline{w1}^h_j, \underline{r0}_i))$; ME4

$\Updownarrow_i(\Updownarrow_j(w1_i, \underline{w0}_j, \underline{w1}^h_j, \underline{r1}_i))$; ME5

$\Updownarrow_i(\Updownarrow_j(w0_i, \underline{w1}_j, \underline{w0}^h_j, \underline{r0}_i))$; ME6

$\Updownarrow_i(\Updownarrow_j(w1_i, \underline{w1}_j, \underline{w0}^h_j, \underline{r1}_i))$; ME7

$\Updownarrow_i(\Updownarrow_j(w0_i, \underline{w0}_j, \underline{r0}^h_j, \underline{r0}_i))$; ME8

$\Updownarrow_i(\Updownarrow_j(w1_i, \underline{w0}_j, \underline{r0}^h_j, \underline{r1}_i))$; ME9

$\Updownarrow_i(\Updownarrow_j(w0_i, \underline{w1}_j, \underline{r1}^h_j, \underline{r0}_i))$; ME10

$\Updownarrow_i(\Updownarrow_j(w1_i, \underline{w1}_j, \underline{r1}^h_j, \underline{r1}_i))$; ME11

$\Updownarrow_i(\Updownarrow_j(w0_i, \underline{w0}^h_j, \underline{w0}_i, \underline{r0}_i))$; ME12

$\Updownarrow_i(\Updownarrow_j(w0_i, \underline{w1}^h_j, \underline{w0}_i, \underline{r0}_i))$; ME13

$\Updownarrow_i(\Updownarrow_j(w1_i, \underline{w0}^h_j, \underline{w1}_i, \underline{r1}_i))$; ME14

$\Updownarrow_i(\Updownarrow_j(w1_i, \underline{w1}^h_j, \underline{w1}_i, \underline{r1}_i))$; ME15

$\Updownarrow_i(\Updownarrow_j(\underline{w1}^h_i, \underline{w0}^h_j, \underline{w0}_i, \underline{r0}_i))$; ME16

$\Updownarrow_i(\Updownarrow_j(\underline{w1}^h_i, \underline{w1}^h_j, \underline{w0}_i, \underline{r0}_i))$; ME17

$\Updownarrow_i(\Updownarrow_j(\underline{w0}^h_i, \underline{w1}^h_j, \underline{w1}_i, \underline{r1}_i))$; ME18

$\Updownarrow_i(\Updownarrow_j(\underline{w0}^h_i, \underline{w0}^h_j, \underline{w1}_i, \underline{r1}_i))\}$ ME19

This test has 20 march elements (ME0 through ME19), each of which contains a *nested march element*. This test has a complexity of $(56 \cdot n^2 + 24 \cdot h \cdot n^2)$. The reason behind the high computational complexity is the assumption that an aggressor can cause a fault in any victim anywhere in the memory. This assumption is, however, unrealistic. The impact of an aggressor is almost always limited to the adjacent neighboring cells. This observation can significantly simplify March T2C, by limiting the value of j to a limited number of adjacent cells. This reduces the complexity of the test from quadratic to linear with the number of cells.

6.3 Detecting soft faults

Soft FPs mean that a correct (but weak) voltage in the cell can gradually be depleted and cause a detectable fault in the cell after a period of time.

Detection conditions for soft faults

An FP has two components to describe a fault: F and R. Only F can cause soft faults (fail after some time), whereas R cannot be soft, since its value is read at moment a read operation is performed.

Table 6. List of single-cell, soft FPs and their detection conditions. Ox_b is performed with a value (x) opposite to that in the sensitizing operation and to a cell (b) different from v, but along the same BL as v.

# Fault	FP (b belongs to BL of v)	Detection cond., $O \in \{w, r\}$
1 ds SF_0	$<0_v[O1_b]_T/1/->$	$\Updownarrow(..w0, ..O1_b, ..T, ..r0..)$
2 ds SF_1	$<1_v[O0_b]_T/0/->$	$\Updownarrow(..w1, ..O0_b, ..T, ..r1..)$
3 $p_i$ds WDF_0	$<w0_v^h[O1_b]_T/1/->$	$\Updownarrow(..w0^h, ..O1_b, ..T, ..r0..)$
4 $p_i$ds WDF_1	$<w1_v^h[O0_b]_T/0/->$	$\Updownarrow(..w1^h, ..O0_b, ..T, ..r1..)$
5 $p_i$ds TF_1	$<w0_v^h w1_v[O0_b]_T/0/->$	$\Updownarrow(..w0^h, ..w1, ..O0_b, ..T, ..r1..)$
6 $p_i$ds TF_0	$<w1_v^h w0_v[O1_b]_T/1/->$	$\Updownarrow(..w1^h, ..w0, ..O1_b, ..T, ..r0..)$
7 $p_i$ds IRF_0	$<w0_v^h[O1_b]r0_v/0/1>$	$\Updownarrow(..w0^h, ..O1_b, ..r0..)$
8 $p_i$ds IRF_1	$<w1_v^h[O0_b]r1_v/1/0>$	$\Updownarrow(..w1^h, ..O0_b, ..r1..)$
9 $p_i$ds $DRDF_0$	$<w0_v^h r0_v[O1_b]_T/1/0>$	$\Updownarrow(..w0^h, ..r0, ..O1_b, ..T, ..r0..)$
10 $p_i$ds $DRDF_1$	$<w1_v^h r1_v[O0_b]_T/0/1>$	$\Updownarrow(..w1^h, ..r1, ..O0_b, ..T, ..r1..)$
11 $p_i$ds RDF_0	$<w0_v^h[O1_b]r0_v/1/1>$	$\Updownarrow(..w0^h, ..O1_b, ..r0..)$
12 $p_i$ds RDF_1	$<w1_v^h[O0_b]r1_v/0/0>$	$\Updownarrow(..w1^h, ..O0_b, ..r1..)$

Table 6 lists all single-cell soft faults, along with the detection conditions needed to detect them. This table is easy to construct based on the detection conditions in Table 4, by introducing a delay time T after every sensitizing operation. Note that the detection conditions for soft IRF and RDF do not include the T, since these faults are detected as soon as they get sensitized. As an example, the (partial, dirty and soft) transition 0 fault ($p_i$ds TF_0), must first be initialized a multiple number of times ($w1^h$). Then, the sensitizing write 0 operation can be performed ($w0$), before a completing operation with data 1 is applied to a different cell along the same BL ($[O1_b]$). To ensure the detection of this soft fault, a delay time T must be introduced after the completing operation to allow for sensitization to take place.

In the same way, one can derive the detection conditions corresponding to all two-cell, soft faults.

Tests for soft faults

Based on the detection conditions of single and two-cell soft faults, it is possible to derive memory tests that detect all these faults. A march test that detects all single-cell soft faults can have the form of March S1C below (for soft, single-cell).

March S1C = {
$\Updownarrow(w0^h, r0, w1_b, T, r0)$; $\Updownarrow(w1^h, r1, w0_b, T, r1)$;
ME0 ME1
$\Updownarrow(w0^h, w1, w0_b, T, r1)$; $\Updownarrow(w1^h, w0, w1_b, T, r0)\}$
ME2 ME3

This march test has 4 march elements (ME0 through ME3). This test is similar to the test for hard single-cell DRAM faults (March H1C), which is expected since the space of soft faults is derived from the space of hard faults. The test uses special nonstandard march elements, where and operation should be performed on a cell b different from the current cell the march element is accessing. This ensures that the BL has the opposite state as compared to the sensitized cell. The test has a complexity of $(12 \cdot n + 4 \cdot h \cdot n + 4 \cdot T \cdot n)$, which is higher than H1C by $(4 \cdot T \cdot n)$. For a typical retention time of $T > 64$ ms, the total idle test time is impractical. It is possible to reduce the idle time to as low as $4 \cdot T$ using specific read/write sequences and data backgrounds (such as the checkerboard). It is also important to implement *design-for-testability* (*DFT*) techniques, or use test stresses that force soft faults to become directly detectable hard faults, which in turn do not require any delay time to detect [Al-Ars05, Wang01].

A march test that detects all two-cell soft faults can be represented by March S2C below.

March S2C = {
$\Updownarrow(w0^h)$; $\Uparrow(r0^h, T, r0, w1^h)$;
ME0 ME1
$\Uparrow(r1^h, T, r1, w0^h)$; $\Downarrow(r0^h, T, r0, w1^h)$;
ME2 ME3
$\Downarrow(r1^h, T, r1, w0^h)$; T; $\Updownarrow(r0)\}$
ME4 ME5 ME6

This march test has 7 march elements (ME0 through ME6). It is based on March H2C for hard faults, with a number of added delays T to a number of march elements in the test (sometimes separating one read operation from a hammer read sequence). Note that ME5 is simply a single delay T added to sensitize the faults before the final detecting read operations are performed in ME6. This test has a time complexity of $[5 \cdot n + 9 \cdot h \cdot n + (4 \cdot n + 1)T]$, which is prohibitively expensive due to the $(4 \cdot n + 1)T$ term. Restricting the impact of every aggressor to a limited number of adjacent victim cells reduces the complexity of this term to $(4 \cdot j + 1)T$, where j is the number adjacent victims (typically $3 \leq j \leq 8$). Even with this optimization, this test remains very costly, and a number of test time reduction methods need to be implemented, such as test stresses, which force soft faults to become directly detectable hard faults [Al-Ars05, Wang01].

7 Industrial support

The tests presented in this paper are being industrially evaluated at the moment. In the mean time, it is possible to use previously published work to illustrate the practical strength of our tests. One of the rather effective DRAM tests investigated in the literature is the Gal9R $= \{\Uparrow(w0); \Uparrow_i(w1_i, \Uparrow_j(r0_j, r1_i), w0_i); \Uparrow(w1); \Uparrow_i(w0_i, \Uparrow_j(r1_j, r0_i), w1_i)\}$. In an experiment of 5952 different tests performed on 800 DRAMs, this tests proved to be far better than any other memory test known in the literature [vdGoor00].

According to the DRAM test theory presented in this paper, Gal9R can simply be considered as a reduced form of march elements ME9 and ME10 of March T2C. ME9 $=\Updownarrow_i(\Updownarrow_j(w1_i, \underline{w0}_j, \underline{r0}_j^h, \underline{r1}_i))$ detects p_at CFds$_{0r0;1}$ $=<0\underline{r0}^h; 1/0_L/->$, where the initialization of the victim to 1 ($w1_i$) and the aggressor to 0 ($\underline{w0}_j$) are considered to be *transient* and have to be performed back-to-back directly before other operations. If we assume this not to be necessary, and only the final detecting read operation ($\underline{r1}_i$) has to be performed directly after the sensitizing read ($\underline{r0}_j^h$), then ME9 can be rewritten as follows $\Updownarrow(w0); \Updownarrow_i(w1_i, \Updownarrow_j(r0_j^h, \underline{r1}_i), w0_i)$. This form of ME9 is equal to the first half of Gal9R (taking $h = 1$). The same transformation is also true for ME10, which yields the second half of Gal9R.

In other words, the new DRAM test space can theoretically explain the underlying strength of Gal9R, by indicating that it is nothing more than a test that targets a simplified form of the following two partial transient coupling faults: p_at CFds$_{0r0;1}$ $=<0\underline{r0}^h; 1/0_L/->$ and p_at CFds$_{1r1;0}$ $=<1\underline{r1}^h; 0/1_L/->$.

8 Conclusions

This paper presented the first DRAM-specific test space needed to test for all DRAM-specific fault models. Six different tests have been derived for (both single as well as two-cell) hard, transient and soft DRAM faults. It was also shown how the new DRAM test space can be used to explain the strength of some well-known effective DRAM tests in practice. The paper also discussed the results of an elaborate simulation-based fault analysis experiment performed at Infineon Technologies to validate the new memory faults. The results show that *all* newly proposed DRAM faults are realistic. The results also indicate that dirty faults are only possible in older DRAM designs.

Our future work is geared toward the optimization of the costly DRAM tests proposed here, and the introduction of more practical design-specific tests. Early indications show that test complexity can be reduced significantly in most cases. We are also working to implement the tests 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.

[Al-Ars02] Z. Al-Ars and A.J. van de Goor, "Approximating Infinite Dynamic Behavior for DRAM Cell Defects," *in Proc. IEEE VLSI Test Symp.*, 2002, pp. 401–406.

[Al-Ars05] Z. Al-Ars, *DRAM Fault Analysis and Test Generation*, PhD thesis, Delft Univ. of Technology, Delft, the Netherlands, 2005, http://ce.et.tudelft.nl/~zaid/

[Al-Ars06] Z. Al-Ars, A.J. van de Goor and S. Hamdioui, "Space of DRAM Fault Models and Corresponding Testing," *in Proc. Design, Automation and Test in Europe*, 2006, pp. 1252–1257.

[Antonin91] G. Antonin, H.-D. Oberle and J. Kolzer, "Electrical Characterization of Megabit DRAMs, 1. External Testing," *in IEEE Design & Test of Computers*, vol. 8 , no. 3, 1991, pp. 36–43.

[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.

[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.

[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.

[Keshavarzi97] A. Keshavarzi, K. Roy and C.F. Hawkins, "Intrinsic Leakage in Low Power Deep Submicron CMOS ICs," *in Proc. IEEE Int'l Test Conf.*, 1997, pp. 146–155.

[Nagi96] N. Nagi and J. Abraham, "Hierarchical Fault Modeling for Linear Analog Circuits," *in Analog Integrated Circuits and Signal Processing*, vol. 10, no. 1–2, June–July 1996, pp. 89–99.

[Naik93] S. Naik, F. Agricola and W. Maly, "Failure Analysis of High Density CMOS SRAMs," *in IEEE Design and Test of Computers*, vol. 10, no. 2, 1993, pp. 13–23.

[Prince91] B. Prince, *Semiconductor Memories: A Handbook of Design Manufacturing and Application*, 2nd ed., John Wiley & Sons, West Sussex, UK, 1991.

[Prince03] B. Prince, "Application Specific DRAMs Today," *in Proc. IEEE Int'l Workshop Memory Technology, Design and Testing*, 2003, pp. 7–13.

[vdGoor00] A.J. van de Goor and A. Paalvast, "Industrial Evaluation of DRAM SIMM Tests," *in Proc. IEEE Int'l Test Conf.*, 2000, pp. 426–435.

[Wang01] M.-J. Wang *et. al*, "Guardband Determination for the Detection of Off-State and Junction Leakages in DRAM Testing," *in Proc. Asian Test Symposium*, 2001, pp. 151–156.