Test Generation and Optimization for DRAM Cell Defects Using Electrical Simulation

Zaid Al-Ars, *Student Member, IEEE,* and Ad J. van de Goor, *Fellow, IEEE*

***Abstract*—Although electrical simulation has become a vital tool in the design process of memory devices, memory testing has not yet been able to employ electrical simulation as an integral part of the test generation and optimization process. This is due to the exponential complexity of the simulation-based fault analysis, a complexity that made such an analysis impractical. This paper describes new methods to reduce the complexity of the fault analysis from exponential to constant with respect to the number of analyzed operations, thereby making it possible: 1) to use electrical simulation to generate test patterns; and 2) to perform simulation-based stress optimization of tests. The paper also discusses ways to analyze the impact of idle time on the faulty behavior. In addition, results of a fault analysis study performed to verify the new analysis method are shown, where the new analysis reduces the analysis time by a factor of 30.**

***Index Terms*—Defect simulation, dynamic RAM (DRAM), memory testing, pattern generation, stress optimization.**

I. Introduction

THE increasing complexity of the faulty behavior of memory devices, associated with the ever increasing costs of memory testing, makes it important to look for new innovative ways to tackle fault analysis and test issues for memories [1]. More often than not, test development is done in a brute force fashion: applying a large number of patterns to identify the best read-write sequences, while scanning a large range of *stresses (STs)* to identify the best *stress combinations (SCs)* to detect the desired faulty behavior.

The complexity of the fault analysis is particularly demanding for dynamic RAMs (DRAMs), as a result of their vulnerability to faults involving multiple memory operations (called *dynamic faults*), since each additional analyzed operation requires an exponential increase in fault analysis time [2]. Previous work on dynamic faults has either been limited to the impact of specific types of memory operations (sequences of reads, for example) [3], or only concerned with analyzing a limited number of dynamic sequences to limit simulation time [4].

Added to the complexity of analyzing dynamic faults, DRAM testing heavily employs modifications to various STs, either to ensure a higher fault coverage of a given test or to target specific failure mechanisms not detected at nominal operational conditions [5]. A test designer, faced with the task of pattern generation in combination with ST optimization for a given defect, can greatly benefit from an internal understanding of the faulty behavior to reduce the time needed to come up with an effective test.

In this paper, a new fault analysis approach is introduced to approximate the total (infinite) faulty behavior for a given defect using electrical simulation. The analysis provides a deeper insight into the faulty behavior of the memory and greatly accelerates the fault analysis process in a way that is independent from the number of investigated operations. This makes possible both simulation-based pattern generation and ST optimization within a reasonable amount of time.

Section II starts with a description of the basics of fault modeling. Section III outlines the conventional way to perform fault analysis using electrical simulation. Then, Section IV presents the new simulation-based analysis approach. Section V discusses STs and the way they are optimized in practice, followed by a description of the simulation-based ST optimization approach in Section VI. Section VII presents the results of the fault analysis performed to validate the approach, and Section VIII ends with the conclusions.

II. Modeling Memory Faults

This section presents the modeling language used to describe the memory faulty behavior observed in a defective memory.

A. Basics of Fault Modeling

Functional fault models (FFMs) are informally understood as the deviation of the observed memory behavior from the functionally specified one, under a sequence of performed memory operations. Therefore, two basic ingredients are needed to define any FFM: 1) a sequence of performed memory operations; and 2) a list of corresponding deviations in the observed behavior from the expected one.

1) Any sequence of performed operations on the memory is called an *operation sequence*. An operation sequence that results in a difference between the observed and the expected memory behavior is called a *sensitizing operation sequence* (S). For example, the S for an up-transition fault (TF↑) in a cell requires that the cell is initialized to zero, after which a one 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-one operation has been performed, the cell still contains a zero.

   In order to describe any faulty behavior in the memory, it is important to be able to describe any operation

Manuscript received October 21, 2002; revised January 8, 2003 and March 11, 2003. This work was supported by Infineon Technologies under the auspices of the Faculty of Information Technology and Systems at the Delft University of Technology. This paper was recommended by Associate Editor S. Reddy.

The authors are with the Computer Engineering Section, Faculty of Information Technology and Systems, Delft University of Technology, 2628 CD Delft, The Netherlands (e-mail: z.e.al-ars@its.tudelft.nl).

Digital Object Identifier 10.1109/TCAD.2003.818125

0278-0070/03$17.00 © 2003 IEEE

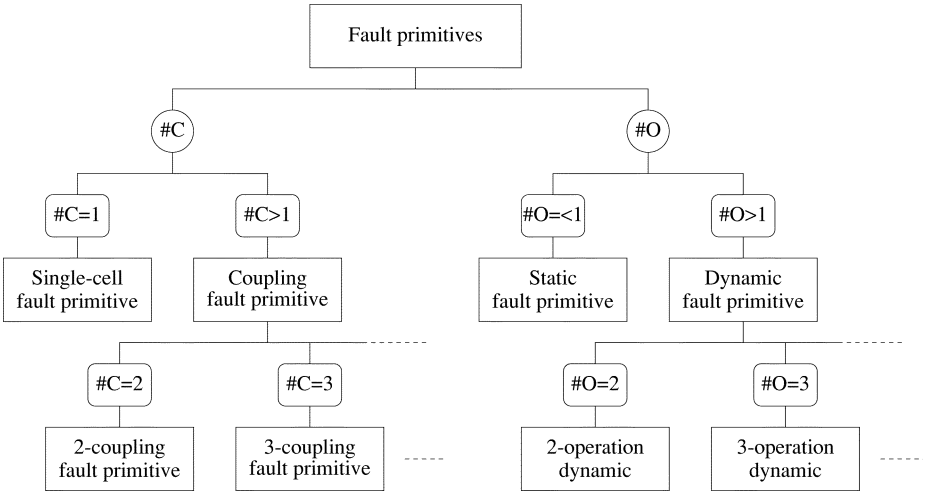

Fig. 1. Taxonomy of FPs.

sequence performed on the memory. An operation sequence has an *initialization part* and an *operation part* (O). The initialization part represents the data present in the memory cells prior to the start of a test; this may be random (due to power-on, for example) or deterministic (due to a previously applied test). The operation part represents operations performed to sensitize the faulty behavior; these can either be writes w or reads r. Therefore, any memory operation sequence expected to result in a faulty behavior can be represented by the following notation:

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

where

c_x cell address used,
d initialization or written data into c_x, $d \in \{0,1\}$,
O type of operation on c_x, $O \in \{w,r\}$,
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 zero) and writing a one into it, then reading the written one.

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 a *fault primitive (FP)* [6]. S describes the operation sequence 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, $R \in \{0,1,-\}$. R has a value of zero or one 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 $\langle 0_c w1_c/0/-\rangle$, which is a TF$\uparrow$, $S = 0_c w1_c$ means that cell c is assumed to have the initial value zero, after which a one is written into c. The fault effect $F = 0$ indicates that after performing a $w1$ to c, as indicated by S, c remains in state zero. The output of the read operation $R = -$ indicates that S does not end with a read operation. The notation for the FP $\langle 0_c w1_c/0/-\rangle$ can be simplified to $\langle 0w1/0/- >_c$.

FPs can be classified into different classes, depending on S as shown in Fig. 1. 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 $S = 0_{c_1} 0_{c_2} w1_{c_2}$, 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).

Depending on $\#C$, FPs can be divided into the following classes.

1) If $\#C = 1$, then the FP sensitized by the corresponding S is called a *single-cell FP*.
2) If $\#C > 1$, then the FP sensitized by the corresponding S is called a *coupling FP*. If $\#C = 2$ then it is described as a *two-coupling FP* or a *two-cell FP*. If $\#C = 3$, then it is described as a *three-coupling FP*, etc.

In case an FP is a coupling FP $(\#C > 1)$, then one of the cells in the S should be considered as a *victim (v)* while the other cells are considered as *aggressors (a)*. In any FP, the described faulty behavior is related to a victim while the aggressors are considered to contribute to the fault.

Depending on $\#O$, FPs can be divided into the following classes.

1) If $\#O \leq 1$, then the FP sensitized by the corresponding S is called a *static FP*.
2) If $\#O > 1$, then the FP sensitized by the corresponding S is called a *dynamic FP*. If $\#O = 2$, then it is described as a *two-operation dynamic FP*. If $\#O = 3$, then it is described as a *three-operation dynamic FP*, etc.

TABLE I
ALL POSSIBLE SINGLE-CELL STATIC FPS

#	S	F	R	FP	Fault model	Fault name
1	0	1	–	$<0/1/->$	SF_0	State fault 0
2	1	0	–	$<1/0/->$	SF_1	State fault 1
3	$0w0$	1	–	$<0w0/1/->$	WDF_0	Write destructive fault 0
4	$0w1$	0	–	$<0w1/0/->$	TF↑	Up-transition fault
5	$1w0$	1	–	$<1w0/1/->$	TF↓	Down-transition fault
6	$1w1$	0	–	$<1w1/0/->$	WDF_1	Write destructive fault 1
7	$0r0$	0	1	$<0r0/0/1>$	IRF_0	Incorrect read fault 0
8	$0r0$	1	0	$<0r0/1/0>$	$DRDF_0$	Deceptive read destructive fault 0
9	$0r0$	1	1	$<0r0/1/1>$	RDF_0	Read destructive fault 0
10	$1r1$	0	0	$<1r1/0/0>$	RDF_1	Read destructive fault 1
11	$1r1$	0	1	$<1r1/0/1>$	$DRDF_1$	Deceptive read destructive fault 1
12	$1r1$	1	0	$<1r1/1/0>$	IRF_1	Incorrect read fault 1

The notion of FPs makes it possible to give a precise definition of an FFM as understood for memory devices. This definition is presented next. An FFM is a nonempty set of FPs.

B. Characteristics of FPs

The most well-known class of FPs is the class of single-cell static FPs, where at most one operation and only one cell is associated with sensitizing the fault. The restriction to single-cell static FPs restricts S in the FP notation $<S/F/R>$ to $S \in \{0, 1, 0w0, 0w1, 1w0, 1w1, 0r0, 1r1\}$. Table I shows all single-cell static FPs possible in the FP description. The faults listed in the table are state faults (SFs), write destructive faults (WDFs), transition faults (TFs), read destructive faults (RDFs), incorrect read faults (IRFs), and deceptive read destructive faults (DRDFs) [7]. From the table, two observations can be made about the capability of this notation to describe different sorts of single-cell static faulty behavior.

1) A write operation is capable of sensitizing four FPs.
2) A read operation is capable of sensitizing six FPs.

In total, if precisely one operation is performed, then ten FPs can be sensitized.

As operations are added to S, in order to investigate the dynamic faulty behavior of the memory, the possible number of different Ss, and the associated number of dynamic FPs, increases rapidly. For a single-cell FP, S typically starts with an initialization of either zero or one, followed by one of three possible memory operations $w0$, $w1$, or r for each increment in $\#O$. As a result, the possible number of different Ss can be calculated by [2]

$$\#S = 2 \cdot 3^{\#O}.$$

When no operation is performed ($\#O = 0$), the number of possible FPs is two, and when one or more operations are performed, S is able to sensitize ten different FPs for each increment in $\#O$, as summarized in the following relation:

$$\#\text{single-cell FPs} = \begin{cases} 2 & : \#O = 0 \\ 10 \cdot 3^{(\#O-1)} & : \#O \geq 1 \end{cases}$$

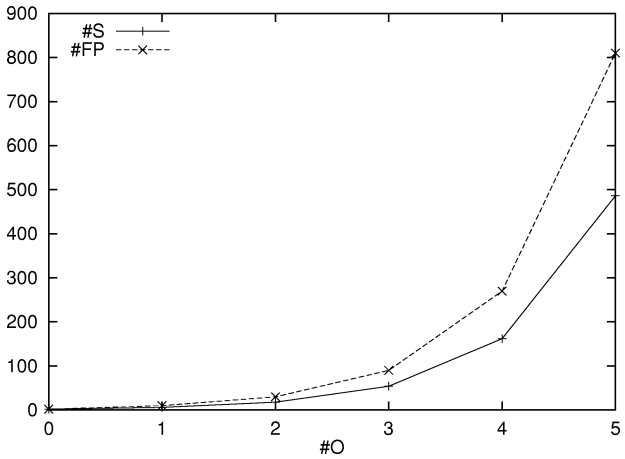

Fig. 2. Plot of $\#S$ and $\#\text{FP}$ as a function of $\#O$.

Fig. 2 plots the number of possible Ss and the number of possible FPs against $\#O$, where the exponential nature of these relations can be clearly seen. A straightforward attempt to investigate the dynamic faulty behavior of the memory, by directly applying all possible Ss, is limited by the practical analysis time and computing power available. Using the analysis approach presented in this paper, the total *infinite* space of dynamic faulty behavior can be approximated within a short amount of analysis time.

C. Extending the FP Description

In Section II-B, faults are defined in such a way that they can describe the three generic memory operations ($w0, w1$ and r) in any possible sequence. However, FPs need not only handle performed operations, but also the absence of performed operations. A typical commodity DRAM today has a capacity of 256 Mb, and has an I/O interface with a maximum width of 32 b, which means that the average probability of a given cell being accessed when an operation is performed amounts to a mere $32/256\ 000\ 000 = 125 \cdot 10^{-9}$, in case the memory is actually accessed.

This shows that it is more probable for a cell to stay idle than actually being accessed, a fact that may have profound implications on the faulty behavior of a memory cell. Extending the FP description has to account for two different types of idle time: 1) idle time in the sensitizing operation sequence (S) before the fault is sensitized; and 2) idle time in the faulty behavior (F) after the fault is sensitized. Note that idle time can have no impact on a faulty output (R), since the output is directly latched and observed externally.

1) Idle Time in S: idle time included in the sensitizing operation sequence may take one of two different forms:

- *Optional idle time*—This idle time does not influence sensitizing the fault by S; in other words, S sensitizes the fault whether there is idle time in S or not. This type of delay time can be represented by three dots $(\ldots)$ in S, since these three dots are usually used in tests to mean an arbitrary sequence of operations. For example, $S = 0_{c_1} \ \ldots \ w1_{c_1}$ means that starting with a zero stored in c_1, we can wait an arbitrary amount of time and then write oneinto c_1.

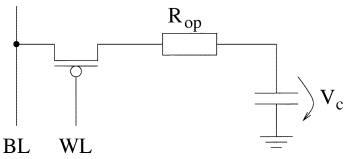

Fig. 3. DRAM memory cell with an open defect.

- *Necessary idle time*—This idle time is needed to sensitize the fault by S; in other words, S only sensitizes the fault when the idle time is included in S. This type of delay can be represented by Del, which stands for "Delay," since Del is usually used in test description to indicate a necessary idle time. For example, $S = 0_{c_1}\ Del\ w1_{c_1}$ means that starting with a zero in c_1, we have to wait for Del amount of time then a write-one is performed.

These two types of idle time are important, since they set different conditions on the tests needed to detect the faulty behavior described by the FP. Optional idle time can be viewed as a *don't care* condition within S that could be used to simplify the needed memory test, while necessary idle time is required in the memory test. As a result, optional idle time results in reducing test complexity while necessary idle time increases test complexity.

2) Idle Time in F: In addition to the need to describe the absence of performed operations, there is also a need to add a timing parameter for the faulty behavior description, since it is possible that the faulty behavior itself is time dependent. This can be done by introducing a timing parameter L as a subscript to $F(\langle S/F_L/R\rangle)$ to indicate that the FP is only detectable within a period L after performing the sensitizing operation sequence S. This type of faulty behavior is referred to as *transient faults* and is discussed in more detail in the literature [8].

III. Conventional Analysis

In this section, the conventional fault analysis approach, called the *precise simulation*, is discussed. The section starts with an example, then the properties of the precise simulation are presented.

A. Example of Analysis

Consider the open defect $(R_{\rm op})$ within a DRAM cell, as shown in Fig. 3, where Spice simulations are to be used to analyze the faulty behavior resulting from this open. The simulation model used here is the same one used in Section VII to generate the results of the analysis study (refer to that section for more information on the model). The analysis takes a range of possible open resistances into consideration (1 $\Omega \le R_{\rm op} \le 10\,{\rm M}\Omega$, for example). The injected open in the cell model creates a floating node of the cell capacitor, the voltage of which (V_c) may vary between $V_{\rm dd} = 2.4V$ and GND. Determining which of the two sides of an injected open is floating depends on the type of the open. The floating node for opens within memory cells is taken to be the node connected to the cell capacitor, since the other node is controlled by the pass transistor.

Next, the fault analysis is performed for some points in the $(V_c, R_{\rm op})$ plane, which is called the *analysis space*. Therefore, a number of values for V_c and $R_{\rm op}$ are selected to perform the fault analysis. This usually corresponds to applying a grid on the analysis space giving rise to a number of intersection points where the analysis is performed.

For each point in the analysis space, the faulty behavior of the memory is analyzed by simulating a number of memory operations. The more operations are simulated, the more accurate our understanding of the faulty behavior becomes. However, the number of different possible Ss grows exponentially with respect to $\#O$ according to the relation $\#S = 2 \cdot 3^{\#O}$. Therefore, the more operations are performed, the more time it takes to carry out the analysis, which makes it important to limit the number of used operations. For example, if sequences of only two operations $(\#O = 2)$ are considered to be performed on a single memory cell, then $2 \cdot 3^{\#O} = 18$ different sequences of $w0, w1$ and r are possible. Each of these sequences has to be performed at each point in the analysis space.

Fig. 4 shows the fault analysis results performed for the open shown in Fig. 3. The results were generated using Ss of at most two operations [4]. The results are organized as fault regions in the analysis space, and they change gradually (i.e., continuously) with respect to V_c and $R_{\rm op}$. For example, Region B5 in Fig. 4 contains the fault TF↑ $(\langle 0w1/0/-\rangle)$, while Region A1 contains the static fault TF↓ $(\langle 1w0/1/-\rangle)$ and the dynamic fault RDF $_{10}(\langle 1w0r0/1/1\rangle)$

B. Fault Analysis Time

In this section, we will try to estimate the time needed to perform the fault analysis using the precise simulation approach. The time needed can be described by the following relation:

$$T_{psim} = \#P \cdot \#S \cdot T_s$$

where $\#P$ is the number of points in the analysis space, $\#S$ is the number of Ss to be performed for each point (equals $2{\cdot}3^{\#O}$), and T_s is the time needed to simulate each S. Furthermore, $\#P$ can be further decomposed into $\#P = \#X \cdot \#Y$, where $\#X$ is the number of points taken along the x axis of the analysis space, and $\#Y$ is the number of points taken along the y axis of the analysis space. T_s can also be further decomposed as $T_s = T_o \cdot \#O$, where T_o is the simulation time needed for a single memory operation. In summary, the simulation time needed for the precise analysis can be written as

$$T_{psim} = \#X \cdot \#Y \cdot 2 \cdot 3^{\#O} \cdot T_o \cdot \#O.$$

We use the analysis performed in Fig. 4 as an example, where V_c is taken to be the x axis and $R_{\rm op}$ is taken to be the y axis.

1) $\#X = 10$ points (10 V_c values on a linear scale, GND $\le V_c \le$ 2.4 V)
2) $\#Y = 15$ points (2 $R_{\rm op}$ values per decade on a logarithmic scale, $1\Omega \le R_{\rm op} \le 10\,{\rm M}\Omega$)
3) $\#S = 18$ (two-operation Ss)
4) $T_o = 10$ s of simulation time
5) $\#O = 2$

This adds up to $T_{\rm psim} = \#X \cdot \#Y \cdot \#S \cdot T_o \cdot \#O = 10 \cdot 15 \cdot 18 \cdot 10 \cdot 2 = 54\,000$ s $= 15$ h. Note that despite the restriction of the analyzed $\#O$ to two, the simulation still takes a long time to perform.

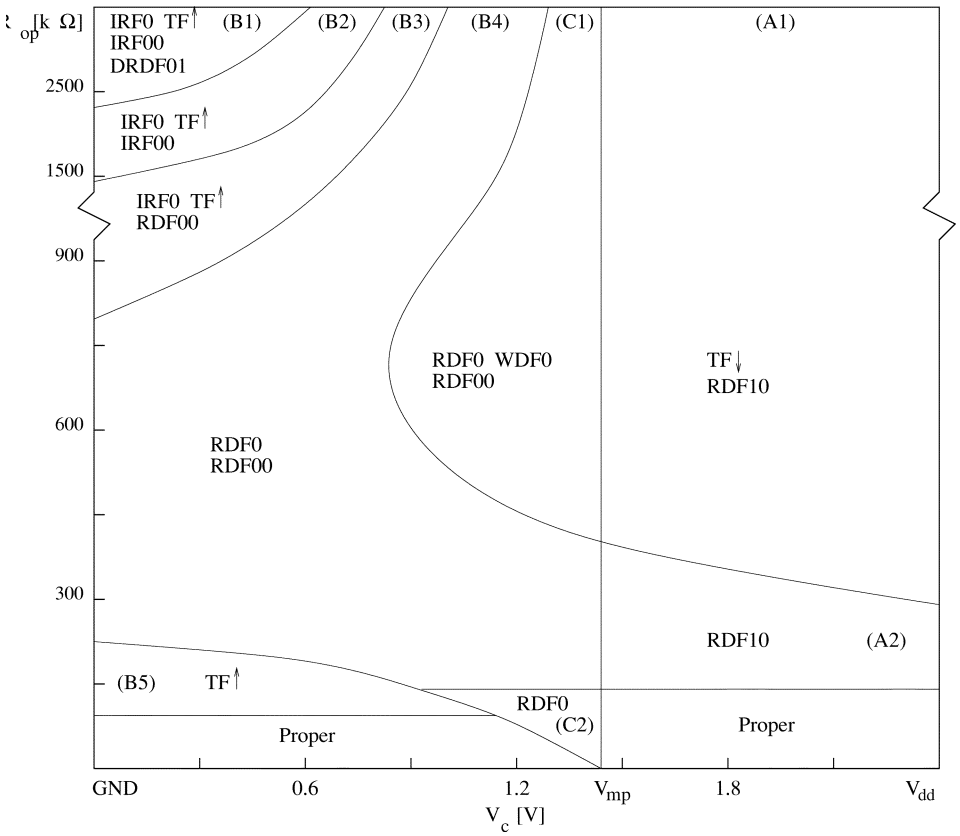

Fig. 4. Conventional fault analysis results of the open in Fig. 3 in the $(V_c, R_{\rm op})$ analysis space.

IV. New Analysis

In this section, the new fault analysis approach, called the *approximate simulation* [9], is discussed. The section starts with an example, then the properties of the approximate simulation approach are presented.

A. Example of Analysis

The new approximate simulation approach is different from precise simulation in that it enables investigating the analysis space for operation sequences with *any* $\#O$, but with a limited amount of analysis time. It achieves this by compromising the accuracy of the results.

Consider the defective DRAM cell shown in Fig. 3, where an open $(R_{\rm op})$ makes the voltage across the cell capacitor (V_c) relatively floating. The analysis takes a range of possible open resistances $(1\ \Omega \leq R_{\rm op} \leq 10\ {\rm M}\Omega)$ and possible cell voltages $({\rm GND} \leq V_c \leq V_{\rm dd})$ into consideration.

Next, a number of $R_{\rm op}$ values are selected for which the analysis is to be performed. In this approach, three different $(V_c, R_{\rm op})$ result planes are generated, one for each memory operation ($w0, w1$, and r). Each result plane describes the impact of successive $w0$, successive $w1$, or successive r operations on V_c for a given value of $R_{\rm op}$. Fig. 5 shows the three result planes for the three memory operations performed for the open shown in Fig. 3.

Plane of $w0$*:* This result plane is shown in Fig. 5(a). To generate this figure, the floating cell voltage V_c is initialized to $V_{\rm dd}$ (because a $w0$ operation is performed) and then the operation sequence $1w0w0\ldots w0$ is applied to the cell. The net result of this sequence is the gradual decrease (depending on the value of $R_{\rm op}$) of V_c toward GND. The voltage level after each $w0$ operation is recorded on the result plane, resulting in a number of curves. Each curve is indicated by an arrow pointing in the direction of the voltage change. The arrows are numbered as $(n)w0$, where n is the number of $w0$ operations needed to get to the indicated curve. We stop performing the $w0$ sequence when the voltage change δV_c as a result of $w0$ operations becomes $\delta V_c \leq 0.24$ V, a value that is arbitrarily selected at first, but can afterwards be reduced if it turns out that more $w0$ operations are needed to describe the faulty behavior. This selection of δV_c results in identifying up to four different $w0$ curves in the plane. The midpoint voltage (v_{mp}) (the cell voltage that makes up the border between a stored zero and one) is also indicated in the figure with a solid vertical line. The sense amplifier threshold voltage $(V_{\rm sa})$ is shown in the figure as a dotted line. $V_{\rm sa}$ is the cell voltage above which the sense amplifier reads a one, and below which the sense amplifier reads a zero.

Plane of $w1$*:* This result plane is shown in Fig. 5(b). To generate this figure, V_c is initialized to GND and then the operation sequence $0w1w1\ldots w1$ is applied to the cell. The result is a gradual increase of V_c toward $V_{\rm dd}$. The voltage level after each $w1$ operation is recorded on the result plane, which gives a number of curves in the plane. The curves are indicated in the same way as the curves in the plane of $w0$. We stop the $w1$ sequence when δV_c becomes less than some arbitrarily selected small value (0.24 V in this example). It is interesting to note the bump in the curve [1] $w1$ of Fig. 5(b) at about $R_{\rm op} = 300\ {\rm k}\Omega$.

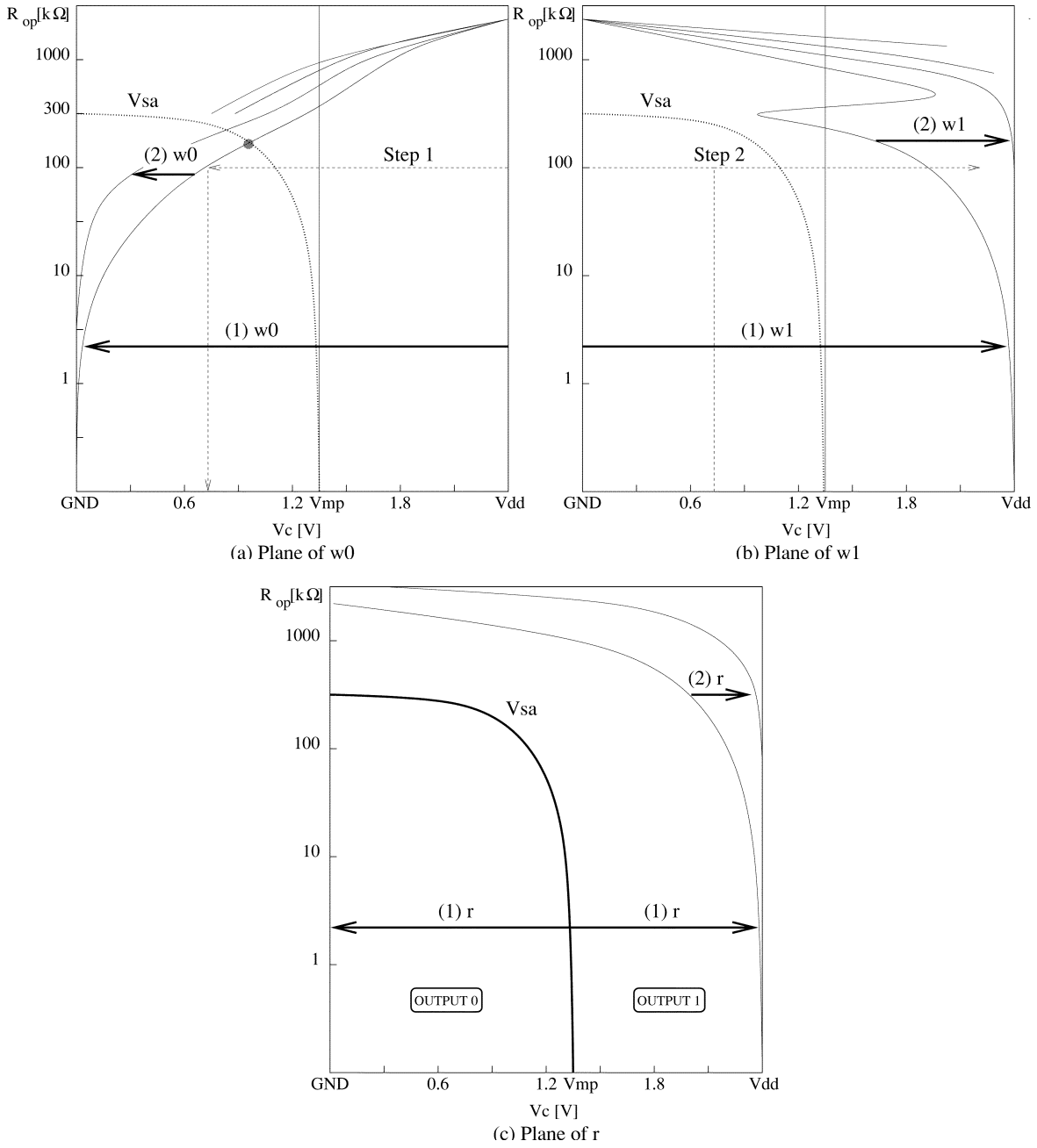

Fig. 5. Result planes of the approximate simulation for the operations (a) $w0$, (b) $w1$, and (c) r.

Remember that any memory operation starts with the sense amplifier sensing the voltage stored in the cell and amplifying it. Above $R_{\rm op} = 300$ kΩ and for any stored cell voltage V_c, the sense amplifier fails in sensing the stored zero, and senses a one instead, which helps the $w1$ operation in charging up the cell to a higher V_c. v_{mp} is also indicated in the figure using a solid vertical line. $v_{\rm sa}$ is shown in the figure as a dotted line.

Plane of r: This result plane is shown in Fig. 5(c). To generate this figure, first $V_{\rm sa}$ is established and indicated on the result plane (shown as a bold curve in the figure). This is done by performing a read operation for a number of V_c values and recursively identifying the V_c border above which the sense amplifier detects a one and below which the sense amplifier detects a zero. As $R_{\rm op}$ increases, $V_{\rm sa}$ turns closer to GND, which means that it gets easier to detect a one and more difficult to detect a zero.[1]. Then the sequence $rrr\ldots r$ is applied twice: first for V_c that is marginally lower than $V_{\rm sa}$ (0.12 V lower in this example), and a second time for V_c that is marginally higher than $V_{\rm sa}$ (0.12 V higher). The voltage level after each r operation in both read sequences is recorded on the result plane, which generally results in two sets of curves on the plane. Each set of curves is indicated in the same way as for the curves in the plane of $w0$. Note that

[1]This is caused by the fact that the precharge cycle sets the bit line voltage to $V_{\rm dd}$. Therefore, as $R_{\rm op}$ increases, a zero stored in the cell fails to pull the bit line voltage down during a read operation, and the sense amplifer detects a one instead of a zero

with $V_c < V_{\rm sa}$ (the part below the bold curve in the figure), only one r operation is enough to set V_c to GNDp; therefore, there are no curves in this part of the plane.

B. Approximating the Behavior

It is possible to use the result planes of Fig. 5 to analyze a number of aspects of the faulty behavior. We mention four aspects here and show how to derive them from the figure.

- Predict faults in any fault region of the conventional precise simulation.
- Approximate the behavior resulting from any operation sequence performed on the defective memory.
- Indicate the *border resistance (BR)*, which is the $R_{\rm op}$ value where the cell *starts* to cause faults on the output for any sequence of operations.
- Generate a test that detects the faulty behavior of the defect for any resistance value and any initial floating voltage.

1) It is possible to predict any fault region observed by the precise simulation shown in Fig. 4. For example, Region B5 in Fig. 4 with the fault TF↑ can be derived from Fig. 5(b) that describes the impact of a sequence of $w1$ operations on V_c. The curve of $(1)w1$ starts at $V_{\rm dd}$ for low $R_{\rm op}$ values, then it decreases rapidly and becomes lower than V_{mp} around $R_{\rm op} \approx 200$ kΩ, only for a small range of $R_{\rm op}$ values. At this point, the $w1$ operation fails to set a high enough voltage within the cell and TF↑ is sensitized. This shows that it is possible use the approximate analysis to derive the faulty behavior of the precise analysis.
2) The three result planes can also be used to approximate the faulty behavior of *any sequence* of memory operations. For example, the results shown in Fig. 5 can be used to find out the behavior of, say, $1w0w1r1$ for $R_{\rm op} = 100$ kΩ. The behavior is evaluated as follows
   - Starting with an initial $V_c = V_{\rm dd} = 2.4$ V, check the value of V_c after performing one $w0$ operation, $(V_c = V_{\rm dd}) \xrightarrow{w0} (V_c = 0.7\ {\rm V})$; see dashed line of Step 1 in Fig. 5(a).
   - Using the new $V_c = 0.7$ V, check the value of V_c after performing one $w1$ operation, $(V_c = 0.7\ {\rm V}) \xrightarrow{w1} (V_c > 1.9\ {\rm V})$; see dashed line of Step 2 in Fig. 5(b). The figure shows that starting with $V_c =$ GND, one $w1$ operation pulls V_c up to 1.9 V. This means that starting with $0.7\ {\rm V} >$ GND in the cell, a $w1$ operation should pull V_c up to at least 1.9 V.
   - Using $V_c > 1.9$ V, check the behavior of the read operation, $(V_c > 1.9\ {\rm V}) \xrightarrow{r} V_c > 2.2\ {\rm V})$, output $= 1$.

   This means that the memory behaves properly and no fault is detected using the sequence $1w0w1r1$ for $R_{\rm op} = 100$ kΩ.
3) The approximate simulation can also be used to state the border resistance, which is the $R_{\rm op}$ value below which the memory behaves properly for *any* possible operation sequence. For the fault analysis shown in Fig. 5, the memory would behave properly for any operation sequence as long as $R_{\rm op} < 200$ kΩ. To understand why, note that a fault would only be detected when a $w1$ operation fails to charge V_c up above $V_{\rm sa}$, or a $w0$ fails to discharge V_c to below $V_{\rm sa}$, where $V_{\rm sa}$ is indicated by the dotted curve in Fig. 5(a) and (b) and the bold curve in Fig. 5(c). In both cases, performing a r after the w detects the faulty behavior. This situation takes place on the result planes at the intersection between the first write operation curves, $(1)w0$ or $(1)w1$, and the $V_{\rm sa}$ curve. The $(1)w0$ curve intersects $V_{\rm sa}$ curve at $R_{\rm op} = 200$ kΩ, as indicated by the dot in Fig. 5(a). Note that the curve $(1)w1$ in Fig. 5(b) does not intersect the $V_{\rm sa}$ curve, which means that $w1$ operations can never result in detecting a fault.
4) The approximate simulation can also be used to generate a test that detects the faulty behavior caused by *any* defect resistance $R_{\rm op}$ for *any* initial floating voltage V_c, in case a fault can be detected. In the case of Fig. 5, faults can be detected with $R_{\rm op} \geq 200$ kΩ. Inspecting the figure shows that with $R_{\rm op} \geq 200$ kΩ, and with any voltage V_c, the sequence $w1w1w0r0$ will detect a fault and result in the destruction of the written zero in the cell. This, in turn, means that the faulty behavior can be represented by FP $= \langle w1w1w0r0/1/1\rangle$. For $R_{\rm op} = 200$ kΩ, this can be validated by noting that performing two $w1$ operations charges V_c up from any voltage (GND or higher) to $V_{\rm dd}$. With $V_c = V_{\rm dd}$, the sequence $w0r0$ detects a fault as discussed in point (2) above. As $R_{\rm op}$ increases, the faulty behavior becomes more prominent and easier to detect, since $V_{\rm sa}$ decreases rapidly toward GND. With $R_{\rm op} \geq 300$, any read operation with any initial V_c results in one on the output, which means that the sequence $w0r0$ fails. Therefore, the detection condition $\Updownarrow(\ldots, w1, w1, w0, r0, \ldots)$ detects any faulty behavior for $R_{\rm op}$. Note that the faulty behavior as discussed here does not take idle time into consideration, since this is analyzed later in detail in Section IV-D

C. Fault Analysis Time

The approximate simulation is much less time consuming than the precise simulation. The time needed can be described by the following relation:

$$T_{\rm asim} = \#P \cdot \#S \cdot T_s$$

where $\#P$ is the number of points in the analysis space, $\#S$ is the number of Ss to be performed for each point, and T_s is the time needed to simulate each S. In the approximate simulation, $\#S = 3$, since we use only 3 Ss, a sequence of $w0$, $w1$, and r. Furthermore, $\#P = \#Y$, where $\#Y$ is the number of points taken along the y axis of the analysis space ($\#X$ is dropped, since we do not take any point on the x axis). T_s can be further decomposed as $T_s = T_o \cdot \#O$, where T_o is the simulation time needed for a single memory operation.

The $\#O$ for the approximate simulation depends on how fast a given S charges the memory cell. However, in order to keep a simulation accuracy along the x axis that is approximately as good as that of the precise simulation, we need at most $\#X$ points along the x axis, which means that we need at most $\#O = \#X$. Operations, however, usually charge V_c fast for most of the $R_{\rm op}$ range, thereby reducing the average $\#O$ needed. In summary, the worst case simulation time needed for the approximate analysis can be written as

$$T_{\rm asim} = \#X \cdot \#Y \cdot T_o.$$

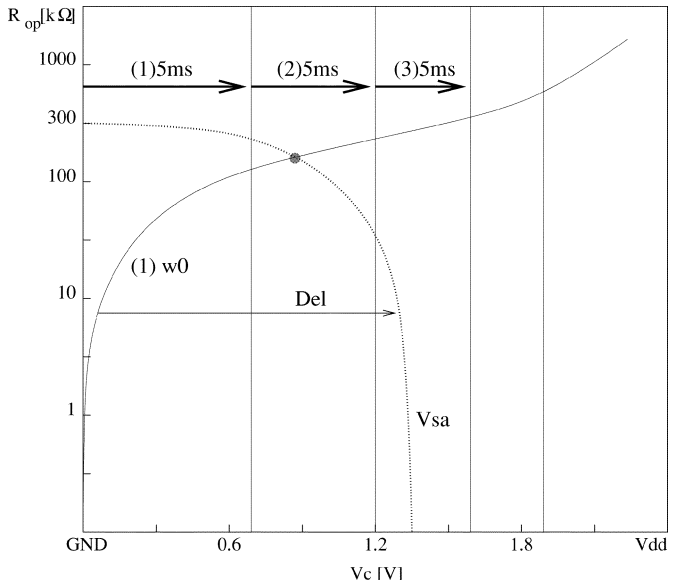

Fig. 6. Result plane when no operations are performed.

We use the analysis performed in Fig. 5 as an example, where V_c is taken to be the x axis while $R_{\rm op}$ is taken to be the y axis.

1) $\#Y = 15$ points (2 $R_{\rm op}$ values per decade on a logarithmic scale).
2) $\#S$ = 3 ($w0$, $w1$, and r Ss).
3) T_o = 10 s of simulation time.
4) $\#O \approx 4$ (this is the average over the $R_{\rm op}$ range).

This adds up to $T_{\rm asim} = \#Y \cdot \#S \cdot T_o \cdot \#O = 15 \cdot 3 \cdot 10 \cdot 4 = 1800$ s $= 0.5$ h, which is about 30 times faster than precise simulation. The difference can be much higher if the number of operations increases. The general theoretical worst case speedup of the approximate simulation approach can be given by

$$\boxed{\text{Speedup} = \frac{T_{\rm psim}}{T_{\rm asim}} = 2 \cdot \#O \cdot 3^{\#O}}$$

which is an exponential speedup with respect to $\#O$.

D. Analysis of Idle Time

The approximate analysis described in Section IV-B is able to approximate the faulty behavior of any sequence of the memory operations $w0$, $w1$ and r, yet it is not able to account for the faulty behavior resulting from idle time in the memory. In order to account for idle time, an additional result plane is needed to identify the faulty behavior of the cell when no operations are performed. Fig. 6 shows such a result plane, where the impact of waiting on V_c is presented for a cell with the defect shown in Fig. 3.

The figure shows that when no operations are performed, the voltage within the cell increases gradually, but very slowly, as a result of naturally occurring leakage currents through the pMOS pass transistor of the cell. After about 5 ms of idle time, the voltage in the cell V_c increases from 0 V to about 0.7 V, as indicated by the first vertical line from the left in the figure. Note that idle time curves have the form of vertical lines in the figure, which means that the impact of leakage current on V_c is rather independent from the value of $R_{\rm op}$. This can be attributed to the relatively long time needed for leakage to develop, compared to the very short time the cell needs to be charged or discharged.

The information provided in the figure regarding the way V_c behaves as a result of idle time makes it possible to evaluate the time dependency of the observed faulty behavior. As discussed in Section IV-B, without considering idle time, the faulty behavior resulting from the open defect shown in Fig. 3 can be described by FP $= \langle w1w1w0r0/1/1\rangle$. There are three different time dependency aspects to be evaluated for each observed FP (see Section II-C): [1] including optional idle time, [2] including necessary idle time, and [3] transient fault considerations.

1) It is important to consider the possibility of including optional idle time in an FP, since it may result in relaxing (i.e., reducing the complexity of) the test generated to detect the faulty behavior. For example, the most relaxed version of FP $= \langle w1w1w0r0/1/1\rangle$ is represented in the form $\langle w1 \ldots w1 \ldots w0 \ldots r0/1/1\rangle$, where any optional idle time can be included between any two memory operations in S. Since, according to Fig. 6, any idle time results in the slow and gradual increase of V_c, then optional idle time may be added after any of the two $w1$ operations (idle time works with the $w1$ effect), yet it is not possible to add idle time after the $w0$ operation (idle time works against the $w0$ effect). In conclusion, the FP description most suitable to describe the faulty behavior is FP $= \langle w1 \ldots w1 \ldots w0r0/1/1\rangle$
2) Necessary idle time is needed when a fault can only be sensitized by waiting for some time (Del), while a sensitizing operation sequence is performed. In the case of DRAMs, a memory cell is expected to fail when it is left idle for a specific period defined in the memory specification, called *retention time* $(t_{\rm ret})$. Therefore, from a DRAM point of view, loss of information as a result of idle time is only considered the result of a fault when $Del < t_{\rm ret}$. When $R_{\rm op} < 200$ kΩ, Fig. 6 shows that as $R_{\rm op}$ increases, the time delay Del between $(1)w0$ and $V_{\rm sa}$ curves gradually decreases, since the voltage difference needed to cause a fault decreases as $R_{\rm op}$ increases. Therefore, the faulty behavior in the cell is more accurately described by FP $= \langle w1 \ldots w1 \ldots w0Delr0/1/1\rangle$, where $0 < Del < t_{\rm ret}$.
3) Transient faults take place when the fault effect sensitized by S remains only temporarily sensitized, and is soon eliminated as a result of leakage currents in the cell. Since the fault considered here results from a faulty read operation that is directly detected on the output, it is not possible for this fault to be transient [8].

In conclusion, the faulty behavior resulting from the defect shown in Fig. 3 can be represented using FP $= \langle w1 \ldots w1 \ldots w0Delr0/1/1\rangle$, where $0 < Del < t_{\rm ret}$. The least value for Del (i.e., zero) should be used in the detection condition to ensure that the faulty behavior is detected, which means that the $r0$ operation should be performed directly after $w0$ operation. The detection condition needed to detect this FP is, therefore, $\Updownarrow (\ldots, w1, \ldots, w1, \ldots, w0, r1)$.

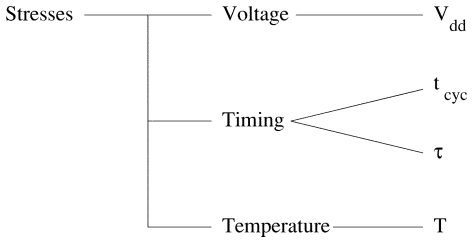

Fig. 7. Total space of STs.

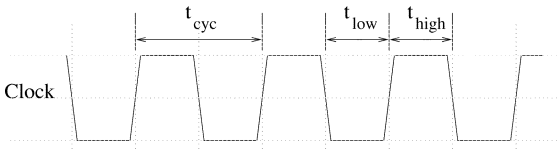

Fig. 8. Clock signal as a stress.

V. Stress Specification

In this section, the specific parameters used to optimize memory tests in general are discussed for each type of stress (ST): voltage, timing, and temperature. Fig. 7 shows the total space of STs to be discussed in this section.

A. Types of Stress

The exact specification of the used STs depends on the device being tested and the amount of control we have on the internal behavior of the memory. In general, there is at least one voltage supply $(V_{\rm dd})$ for the memory, the voltage level of which can be controlled at test time. Some more complex memory devices may have more than one supply voltage ($V_{\rm dd1}$ and $V_{\rm dd2}$) to power different parts of the memory. In this paper, we assume one supply voltage $V_{\rm dd}$, which has control on the voltages of the cell array. According to memory specification, there is a range within which this voltage may vary ($\pm 10\%$, for example).

In addition to voltage, there is timing. Almost all recent memory devices are so-called synchronous memories, referring to the fact that all events that take place in the memory are governed by a global clock signal (an input signal to any synchronous memory). For the use of timing as a ST, this clock signal can be modified in two different ways: by changing the period of the clock (also called the cycle time, $t_{\rm cyc}$) or by changing the duty cycle time (τ). The cycle time is the time the clock takes to cycle back to the same voltage level, while the duty cycle is calculated as the ratio between the time the clock spends at voltage high and the time the clock spends at voltage low ($\tau = t_{\rm high}/t_{\rm cyc}$). Fig. 8 graphically depicts $t_{\rm cyc}$, $t_{\rm high}$ and $t_{\rm low}$.

Temperature may also be used as a ST to optimize testing. Temperature has proven to be a very effective ST to bring devices closer to failure. In general, a higher testing temperature results in a higher fault coverage for many tests. For some specific defects, however, a higher temperature is less effective in testing and may result in hiding faults [10]. Testing at different temperatures is expensive and, therefore, only a limited number of temperature changes are practically acceptable in the test flow.

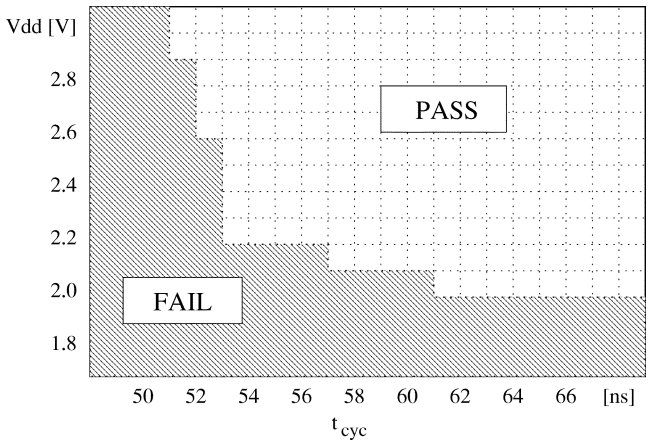

Fig. 9. Shmoo plot to optimize the cycle time and supply voltage.

Section VI presents the approach proposed by this paper to optimize these three different stresses with respect to a given memory test as a specific defect is targeted.

B. Shmoo Plotting

A Shmoo plot is an important method used to optimize STs for a given memory test [11]. Two STs (S1 and S2) are chosen to be optimized in a given range. A test is then applied to the memory and, for each combination of S1 and S2, the pass/fail outcome of the test is registered on the Shmoo plot. This creates a two dimensional graphical representation of the pass/fail behavior of the memory under the applied test. Fig. 9 shows an example of a Shmoo plot, where the x axis represents the clock cycle time and the y axis represents the supply voltage $V_{\rm dd}$. The figure shows, for example, that a lower voltage and a shorter cycle time are the most stressful conditions for the applied test.

Shmoo plotting has the advantage of direct optimization of a pair of STs for a given test on a chip, in case the chip is known to have the targeted defect. Shmoo plotting suffers, however, from the following disadvantages

1) Depending on the length of the test, generating a Shmoo plot may take large amounts of time, since the test has to be repeated for each (x, y) combination in the plot [12].
2) The tester provides only a restricted controllability and observability of internal parts of the circuit under test [13].
3) It is not always clear how the externally observed failure of the memory relates to the internal faulty behavior caused by the targeted defect.
4) Since only a limited number of memory devices are investigated, the resulting STs may not be the most optimal for the investigated test and targeted defect.

For a test designer attempting to optimize a given test for a specific defect using Shmoo plots, the above-mentioned problems make optimization a rather difficult and challenging task. The work presented in this paper targets these problems and provides more insight into the faulty behavior, an insight that guides a test designer through the process of test optimization, so that test development time can be reduced.

VI. Test Optimization

In this section, a method is introduced to optimized memory tests using electrical simulation with respect to timing, temperature, and voltage. First the optimization methodology is introduced, which is then applied for each of the three STs.

A. Optimization Methodology

The fault analysis concept that enables simulation-based optimization of STs is the ability to state the border resistance (BR) of a defect [14]. BR is the resistive value of a defect at which the memory starts to show faulty behavior. Using this important piece of information, the criterion to optimize any ST can be stated as

> A change in a given ST should modify the value of the border resistance in that direction which maximizes the resistance range that results in a detectable functional fault.

Since it is still only possible to identify the border resistance of defects within a DRAM memory cell, we can optimize STs for tests designed to detect cell defects only. In this section, we describe the approach used to identify the border resistance of cell defects.

Optimizing any ST can generally be done by performing a full fault analysis (generating the three result planes as shown in Fig. 5) for each ST value of interest, and by inspecting the impact of each ST value on BR. This method is both labor intensive and time consuming. Fortunately, it is sometimes possible to deduce the impact of different STs on the value of BR by performing a limited number of simulations only. Below, this method is outlined in an example to optimize STs for the detection condition derived for the open in Fig. 3 with respect to $t_{\rm cyc}$, T, and $V_{\rm dd}$.

The result planes in Fig. 5 have been generated for $t_{\rm cyc} =$ 60 ns, $T = +27$ °C and $V_{\rm dd} = 2.4$ V. The planes show that, for nominal STs, the border resistance has a value of about $R_{\rm op} =$ 200 kΩ. This value is determined by the intersection point of the $(1)w0$ curve and the $V_{\rm sa}$ curve as shown in Fig. 5(a). Therefore, increasing the range of the failing $R_{\rm op}$ can be done in two ways.

- By reducing the ability of $1w0$ to write a low voltage into the cell. This stresses the $1w0$ operation and results in shifting the $(1)w0$ curve to higher V_c voltages.
- By reducing the range of cell voltages in which r detects a zero. This stresses the r operation and results in shifting the $V_{\rm sa}$ curve to lower V_c voltages.

These two conditions can be easily inspected using a limited number of simulations of the $1w0$ and the r operation, as the ST in question is modified. In the following, this inspection process is shown for each ST.

B. Optimizing Timing

Fig. 10 shows the simulation results of reducing $t_{\rm cyc}$ from 60 ns to 55 ns. The figure has two panels: the top is for applying a $1w0$ operation and the bottom for applying an r. The x axis in the figure represents the time axis, while the y axis gives the stored cell voltage V_c.

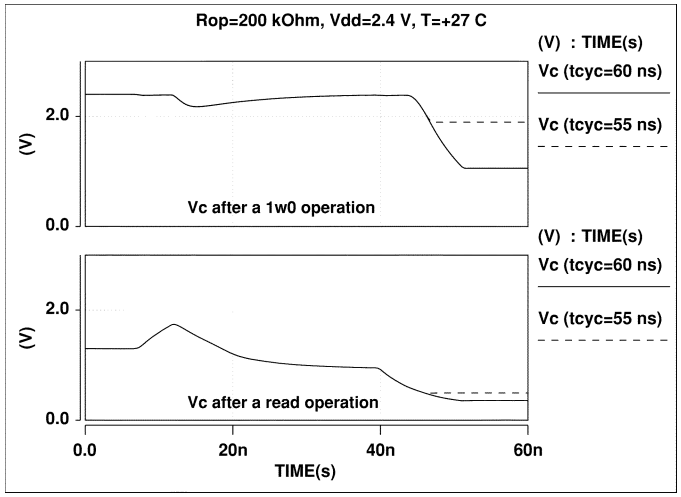

Fig. 10. Simulation of reducing $t_{\rm cyc}$ from 60 ns to 55 ns with $V_{\rm dd} = 2.4$ V, $R_{\rm op} = 200$ kΩ and $T = +27$ °C.

Applying $1w0$*:* The top panel outlines the cell voltage V_c while performing a $1w0$ operation with $t_{\rm cyc} = 60$ ns and 55 ns. In the simulation, the initial cell voltage $(V_{\rm ini})$ is $V_{\rm dd}$ (logic 1), $R_{\rm op} = 200$ kΩ, and $T = +27$ °C. By the end of the write operation, the value of V_c is 1.0 V for $t_{\rm cyc} = 60$ ns, while $V_c = 1.9$ V for $t_{\rm cyc} = 55$ ns. This indicates that reducing the cycle time reduces the ability of $1w0$ to write a zero into the cell. As a result, reducing $t_{\rm cyc}$ is considered more stressful that increasing $t_{\rm cyc}$ for the $1w0$ operation.

Applying r*:* The bottom panel outlines the cell voltage V_c while performing a r operation with $t_{\rm cyc} = 60$ ns and 55 ns. In the simulation, $V_{\rm ini} = 1.1$ V, which is slightly below $V_{\rm sa}$, $R_{\rm op} = 200\Omega$, and $T = +27$ °C The figure shows that after about $t = 13$ ns, V_c is pulled low and a zero is written back to the cell, which means the sense amplifier senses a zero for both values of $t_{\rm cyc}$. This indicates that the ability of the sense amplifier to detect a zero or a one does not change as a result of changes in timing. This means that $t_{\rm cyc}$ has no impact on $V_{\rm sa}$.

In conclusion, decreasing $t_{\rm cyc}$ is more stressful for the $1w0$ operation than increasing it, and it has no impact on the detected value of the r. Therefore, the cycle time should be reduced to increase the stress on the performed memory test.

C. Optimizing Temperature

Fig. 11 shows the simulation results with $T = -33$ °C + 27 °C and +87 °C. The figure has two panels: the top is for applying a $1w0$ operation and the bottom for applying an r

Applying $1w0$*:* The top panel outlines the cell voltage V_c while performing a $1w0$ operation with $T = -33$ °C, +27 °C and +87 °C. The simulation used $V_{\rm ini} = V_{\rm dd}$ (logic 1), $R_{\rm op} =$ 200 kΩ, and $t_{\rm cyc} = 60$ ns. By the end of the write operation (at $t = 60$ ns), the value of V_c is 1.1 V for $T = +87$ °C, $V_c =$ 1.05 V for $T = +27$ °C, while $V_c = 1.0$ V for $T = -33$ °C. This indicates that increasing the temperature reduces the ability of $1w0$ to write a zero into the cell. This behavior can be attributed to the gradual decrease in drain current as temperature increases, which is in turn caused by the decreasing mobility of charge carriers with increasing T. As a result, increasing T is

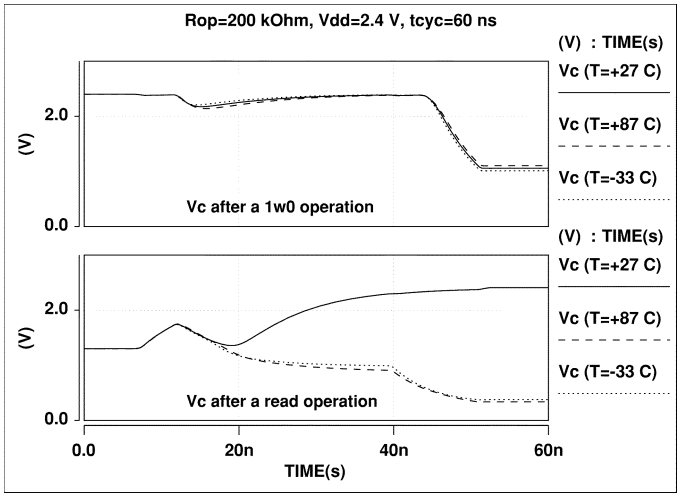

Fig. 11. Simulation with $T = -33$ °C, $+27$ °C and $+87$ °C, $V_{\rm dd} = 2.4V$, $R_{\rm op} = 200$ kΩ and $t_{\rm cyc} = 60$ ns.

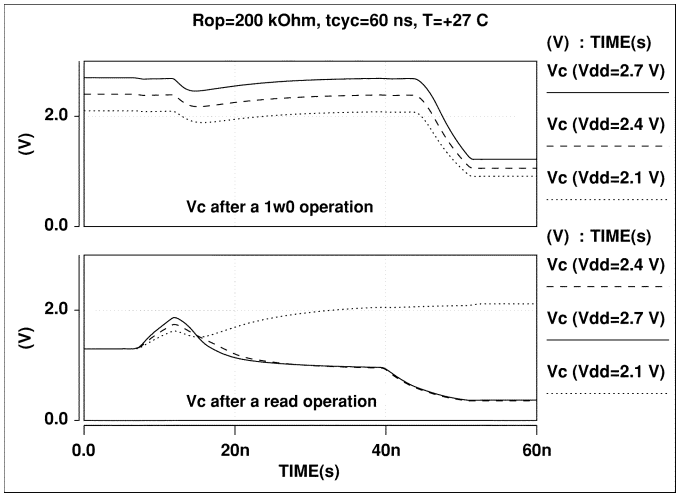

Fig. 12. Simulation with $V_{\rm dd} = 2.1$ V, 2.4 V, and 2.7 V, $t_{\rm cyc} = 60$ ns, $R_{\rm op} = 200$ kΩ, and $T = +27$ °C.

considered more stressful than decreasing T for the $1w0$ operation.

Applying r: The bottom panel outlines V_c while performing an r operation with $T = -33$ °C, $+27$ °C, and $+87$ °C. The simulation used an initial cell voltage $V_{\rm ini} = 1.3$ V, which is slightly above $V_{\rm sa}$, and $R_{\rm op} = 200$ kΩ. The sense amplifier detects a one with $T = +27$ °C, while it detects a zero with both -33 °C and $+87$ °C. This behavior is caused by a number of temperature-related mechanisms with opposing effects on the faulty behavior, such as the increased transistor threshold voltage (promotes detecting one), the increased drain current (promotes detecting zero), and the decreased leakage current (promotes detecting zero) with decreasing T. This indicates that increasing or decreasing temperature from $+27$ °C shifts the $V_{\rm sa}$ curve to the right. As a result, $+27^{\circ}$ is considered as a more stressful condition than both -33 °C and $+87$ °C for the r operation.

In conclusion, the most stressful T can either be at room temperature or high temperature. To specify which of these should be selected, the border resistance has to be identified for high T and compared with the border resistance for room T. The border resistance can be identified by performing a number of simulations to construct the $(1)w0$ curve and the $V_{\rm sa}$ curve. This has been done, and the results indicate that high temperature is more effective than room temperature since it reduces the border resistance by 5 kΩ.

D. Optimizing Voltage

Fig. 12 shows the simulation results with $V_{\rm dd} = 2.1$ V, 2.4 V, and 2.7 V. The figure has two panels: the top is for applying a $1w0$ operation and the bottom for applying an r.

Applying $1w0$: The top panel outlines the cell voltage V_c while performing a $1w0$ operation with $V_{\rm dd} = 2.1$ V, 2.4 V, and 2.7 V. The simulation used $V_{\rm ini} = V_{\rm dd}$ (logic 1), $R_{\rm op} = 200$ kΩ, and $T = +27$ °C. By the end of the write operation (at $t = 60$ ns), the value of V_c is 1.0 V for $V_{\rm dd} = 2.4$ V, $V_c = 1.2$ V for $V_{\rm dd} = 2.7$ V, while $V_c = 0.9$ V for $V_{\rm dd} = 2.1$ V. This indicates that increasing the supply voltage reduces the ability of $1w0$ to write a zero into the cell. As a result, increasing $V_{\rm dd}$ is considered more stressful than reducing $V_{\rm dd}$ for the $1w0$ operation.

Applying r: The bottom panel outlines the cell voltage V_c while performing an r operation with $V_{\rm dd} = 2.1$ V, 2.4 V, and 2.7 V. In the simulation, $V_{\rm ini} = 1.1$ V, which is slightly below $V_{\rm sa}$, $R_{\rm op} = 200$ kΩ, and $T = +27$ °C. The figure shows that after about $t = 13$ ns, V_c is discharged for $V_{\rm dd} = 2.4$ V and 2.7 V, which means that the sense amplifier detects a zero with these voltages. On the other hand, V_c is charged up for $V_{\rm dd} = 2.1$ V, which means that the sense amplifier detects a one. This indicates that increasing the supply voltage increases the range of V_c values that result in detecting a zero. As a result, increasing $V_{\rm dd}$ is considered less stressful than reducing $V_{\rm dd}$ for the r operation.

In conclusion, increasing $V_{\rm dd}$ is more stressful for the $1w0$ and less stressful for the r. This provides no information on the way $V_{\rm dd}$ stresses the test. Therefore, the border resistance should be identified by performing a number of simulations to construct the $(1)w0$ curve and the $V_{\rm sa}$ curve with $V_{\rm dd} = 2.7$ V and 2.1 V. This has been performed and the results indicate that the border resistance is 170 kΩ for $V_{\rm dd} = 2.1$ V, 200 kΩ for $V_{\rm dd} = 2.4$ V and 220 kΩ for $V_{\rm dd} = 2.7$ V. This means that $V_{\rm dd} = 2.1$ V is the most effective voltage, since it gives the lowest border resistance.

E. SC Evaluation

After identifying most stressful values of each ST, it is important to apply the resulting SC and construct the fault analysis planes of $w0$, $w1$, and r again to see whether new detection conditions are needed to detect the faulty behavior. Fig. 13 shows these result planes using the SC: $V_{\rm dd} = 2.1$ V, $t_{\rm cyc} = 55$ ns, and $T = +87$ °C.

The figure shows a number of interesting changes in the behavior as compared to Fig. 5, as listed below.

1) The border resistance represented by the intersection point of the $(1)w0$ curve and the $V_{\rm sa}$ curve is reduced to about 50 kΩ [see the dot in Fig. 5(a)].

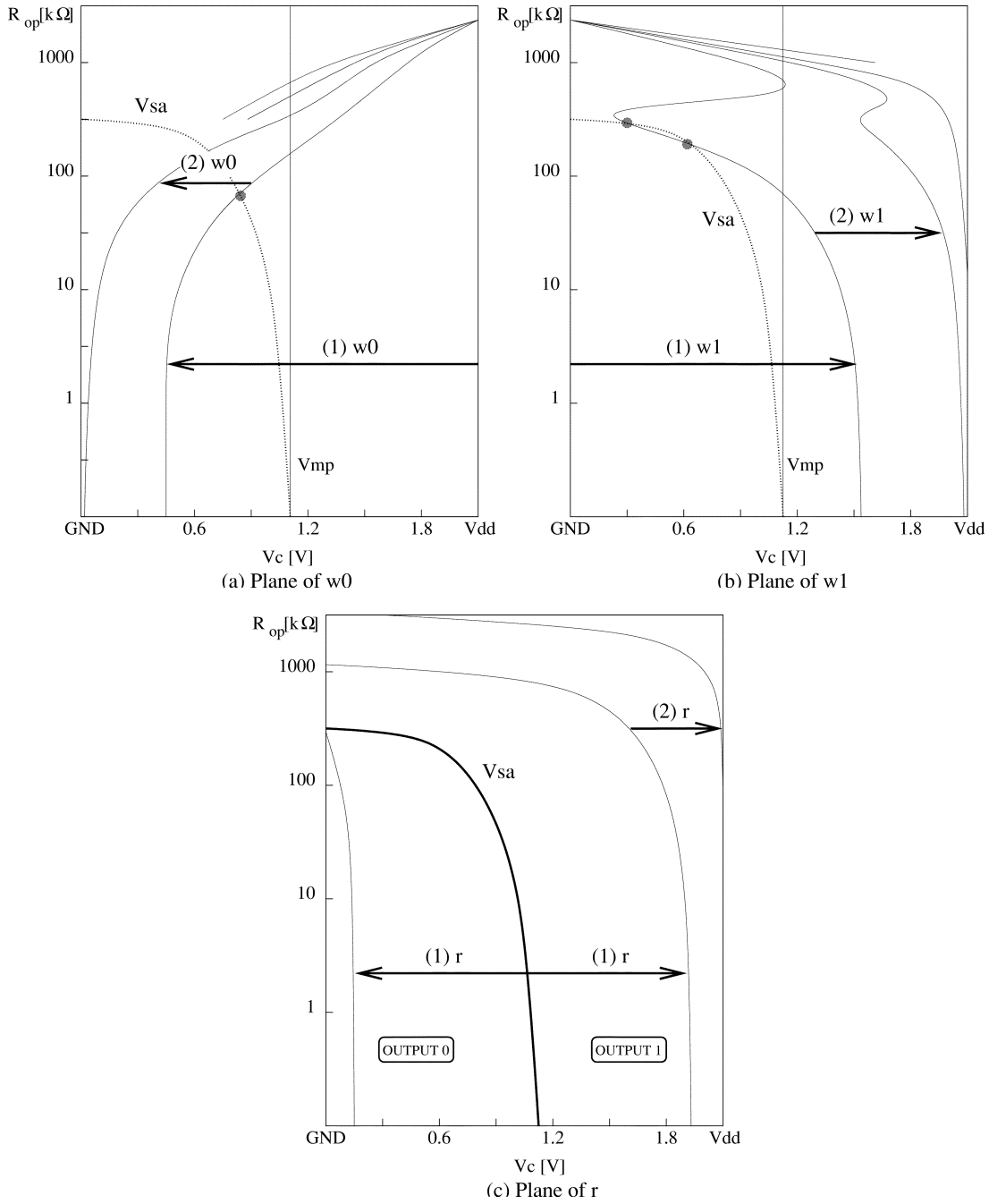

Fig. 13. Result planes with $V_{\rm dd} = 2.1$ V, $t_{\rm cyc} = 55$ ns, and $T = +87\,^{\circ}$C, for the operations (a) $w0$, (b) $w1$, and (c) r.

2) The reduction in the border resistance has been achieved by limiting the effect of a single $1w0$ and increasing the range of V_c values that gives a one on an r operation.
3) With the used SC, a new detection condition should be used that includes more $w1$ operations to charge the cell to a high enough voltage. The detection condition is $\Updownarrow (\ldots, w1, w1, w1, w0, r0, \ldots)$.
4) The applied SC induces a fail in the $0w1$ operation for the $R_{\rm op}$ range of 150 kΩ to 200 kΩ [see the two dots in Fig. 5(b)]. But this $R_{\rm op}$ value does not represent a border resistance, since $1w0$ fails at a lower $R_{\rm op}$.
5) The used SC is very stressful, since (even with $R_{\rm op} = 0\ \Omega$) a $w0$ operation cannot discharge V_c from $V_{\rm dd}$ to GND, and $w1$ cannot charge V_c up from GND to $V_{\rm dd}$.

VII. Analysis Results

The optimization method outlined in Section VI-A has been applied to optimize tests to detect the faulty behavior of a number of DRAM cell defects. This section presents the simulation methodology first, then the analysis results are discussed.

A. Simulation Methodology

The used electrical simulation model is a simplified design-validation model of a real DRAM manufactured in 0.35-μm technology. The simplified model includes one folded cell array column (2×2 memory cells, two reference cells, precharge devices, and a sense amplifier), one write driver,and one data

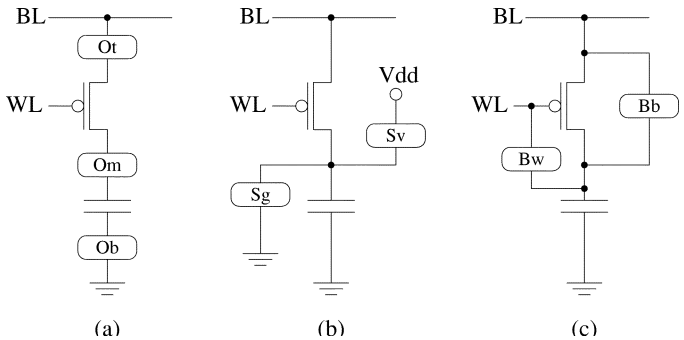

Fig. 14. Simulated cell defects. (a) Opens. (b) Shorts. (c) Bridges.

TABLE II
SIMULATION BASED OPTIMIZATION RESULTS FOR DEFECTS SHOWN IN FIG. 14

Defect	Nom. border R	V_{dd}	t_{cyc}	T	Str. border R	Str. detection condition
Ot,m,b (true)	$R \geq 200$ kΩ	↓	↓	↑	$R \geq 50$ kΩ	$\Updownarrow(..., w1, ..., w1, ..., w1, ..., w0, Del, r0, ...)$
Ot,m,b (comp.)	$R \geq 200$ kΩ	↓	↓	↑	$R \geq 50$ kΩ	$\Updownarrow(..., w0, ..., w0, ..., w0, ..., w1, Del, r1, ...)$
Sg (true)	$R \leq 1$ MΩ	↑	↓	↑	$R \leq 10$ GΩ	$\Updownarrow(..., w1, ..., r1, ...)$
Sg (comp.)	$R \leq 1$ MΩ	↑	↓	↑	$R \leq 10$ GΩ	$\Updownarrow(..., w0, ..., r0, ...)$
Sv (true)	$R \leq 400$ kΩ	↓	↓	↑	$R \leq 1$ GΩ	$\Updownarrow(..., w0, ..., r0, ...)$
Sv (comp.)	$R \leq 400$ kΩ	↓	↓	↑	$R \leq 1$ GΩ	$\Updownarrow(..., w1, ..., r1, ...)$
Bb (true)	$R \leq 200$ kΩ	↓	↓	↑	$R \leq 100$ kΩ	$\Updownarrow(..., w0, ..., r0, ...)$
Bb (comp.)	$R \leq 200$ kΩ	↓	↓	↑	$R \leq 100$ kΩ	$\Updownarrow(..., w1, ..., r1, ...)$
Bw (true)	$R \leq 200$ kΩ	↑	↓	↑	$R \leq 100$ kΩ	$\Updownarrow(..., w0, ..., r0, ...)$
Bw (comp.)	$R \leq 200$ kΩ	↑	↓	↑	$R \leq 100$ kΩ	$\Updownarrow(..., w1, ..., r1, ...)$

output buffer. The used simulation tool is the electrical Spice-based simulator Titan, which is a proprietary simulator developed by Siemens/Infineon.

Fig. 14 shows the seven analyzed defects: three opens, two shorts, and two bridges. Opens are added resistive components on signal lines within memory cells. Shorts are resistive connections to $V_{\rm dd}$ or GND. Bridges are resistive connections between nodes within the memory cell.

For all defects, the cell voltage (V_c) has been used as the floating node voltage in the analysis. For defects within cells, all array voltages other than V_c are initialized to their precharge voltage at the beginning of each memory operation.

B. Simulation Results

Table II summarizes the simulation results. The first column lists the analyzed defects as shown in Fig. 14. Defects described by "true" are simulated on the true bit line, while defects described by "comp." are simulated on the complementary bit line. The column "Nom. border R" gives the value of the border R at a nominal SC. The columns with the STs give the direction in which these STs should be modified in order to stress the memory test. The table also lists the stressed value of the border R and the corresponding detection condition.

Note that the border R value as well as the direction of ST optimization are the same for true and comp. defects in the table. In addition, the detection conditions for the comp. entries have the same structure as their true counterparts, but with ones and zeros interchanged. This is due to the fact that the physical voltages stored within the cell are the same for the true and complementary defects.

When a higher stress is applied to a defective memory, the objective is to increase the resistance range in which the memory fails. This means that the border resistance should decrease under stress for opens (Ot, Om, and Ob), and should increase under stress for shorts (Sg and Sv) and bridges (Bb and Bw). The table shows that the applied SCs are very effective in increasing the range of the failing R. In terms of testing, this means that the applied SCs increase the coverage of a given test. For example, the border resistance of cell opens (Ot, Om, and Ob) have been reduced from 200 to 50 kΩ.

For all analyzed defects, reducing the clock cycle time has proven to be more stressful than relaxing the clock. This can be explained by noting that reducing $t_{\rm cyc}$ reduces the time the memory has to charge or discharge the cell, which affects the write operation and not the read operation. Since the more stressful situation occurs when we limit the ability of a write to influence V_c, it follows directly that reducing $t_{\rm cyc}$ is the more stressful condition.

For all analyzed defects, increasing the temperature has proven to be more stressful than reducing the temperature.

This can be attributed to the fact that all simulated defects are modeled using regular ohmic resistances, the value of which does not change in the simulation. Modeling the defects to increase their R with decreasing T (which is the case with silicon-based defects) may result in a different stress value for T.

If the impact on the border resistance of the three STs used in the analysis is compared, it shows that $t_{\rm cyc}$ is by far the most effective ST. This is followed by $V_{\rm dd}$, and finally by T which has the least effect on the border resistance. This can be explained by noting that the value used for low $t_{\rm cyc} = 55$ ns is very aggressive, since it is less than the lowest limit in the memory specifications. For $V_{\rm dd}$ and T, however, the used values are within the memory specification.

The table shows that all defects start to fail in the resistance range 200 k$\Omega \leq R \leq$ 1MΩ. Open resistances, in particular, start to cause faults above a value of 200 kΩ, which is a relatively high value when compared to the open-channel resistance of 4.5 kΩ for the pass transistor. This indicates, for example, that the signal lines within cells are rather insensitive to process variations, and that strong opens are needed to cause a failure in the cell.

The table also shows that, in order to detect any faulty behavior caused by cell shorts and bridges, detection conditions are needed with only two operations. For faults caused by opens, a detection condition with four memory operations is needed. This indicates that mostly simple sequences are needed to detect the faulty behavior of cell defects. This conclusion supports the long held assumption that most important memory faults have a simple dynamic behavior with a small $\#O$.

VIII. Conclusion

This paper presented a new approach for memory test generation and stress optimization of cell defects, using defect injection and electrical Spice simulation of a memory model. The three main contributions of the paper are the following.

1) A new Spice-based test generation approach shown to provide a significant speedup in the analysis time, as compared to more conventional approaches.
2) A method to use defect simulation to optimize stresses for memory tests. The method provides more insight into the optimization process, since it internally studies the impact of each stress for the targeted defect.
3) A way to analyze the impact of idle time on the faulty behavior.

The paper presented the results of a study performed to verify the newly proposed test generation approach. The results show that the new analysis method reduces the analysis time by a factor of 30 compared to the conventional analysis, and that stresses (timing, temperature, and voltage) are effective in bringing defective devices closer to failure.

References

[1] S. S. Iyer and H. L. Kalter, "Embedded DRAM technology: Opportunities and challenges," *IEEE Spectr.*, vol. 36, pp. 56–64, Apr. 1999.

[2] Z. Al-Ars, "Analysis of the space of functional fault models and its application to embedded DRAM's," M.S. thesis, CARDIT, Delft Univ. Technology, Delft, The Netherlands, 1999.

[3] A. J. van de Goor. (1998) *Testing Semiconductor Memories, Theory and Practice*

[4] Z. Al-Ars and A. J. van de Goor, "Static and dynamic behavior of memory cell array opens and shorts in embedded DRAM's," in *Proc. Design, Automation and Test Eur.*, 2001, pp. 496–503.

[5] T. Falter and D. Richter, "Overview of status and challenges of system testing on chip with embedded DRAM's," *Solid-State Electron.*, no. 44, pp. 761–766, 2000.

[6] A. J. van de Goor and Z. Al-Ars, "Functional memory faults: A formal notation and a taxonomy," *Proc. IEEE VLSI Test Symp.*, pp. 281–289, 2000.

[7] R. D. Adams and E. S. Cooley, "Analysis of a deceptive destructive read memory fault model and recommended testing," presented at the IEEE North Atlantic Test Workshop, Hanover, NH, 1996.

[8] Z. Al-Ars and A. J. van de Goor, "Transient faults in DRAMs: Concept, analysis and impact on tests," in *Proc. IEEE Int. Workshop Memory Technology, Design and Testing*, 2001, pp. 59–64.

[9] ——, "Approximating infinite dynamic behavior for DRAM cell defects," in *Proc. IEEE VLSI Test Symp.*, 2002, pp. 401–406.

[10] R. McConnell, U. Möller, and D. Richter, "How we test siemens' embedded DRAM cores," in *Proc. IEEE Int. Test Conf.*, 1998, pp. 1120–1125.

[11] K. Baker and J. van Beers, "Shmoo plotting: The black art of IC testing," *IEEE Des. Test Comput.*, vol. 14, pp. 90–97, July–Sept. 1997.

[12] M. Hamada *et al.*, "A high-speed boundary search SHMOO PLOT for ULSI memories," in *IEEE Workshop Memory Testing*, 1993, pp. 4–9.

[13] D. Niggemeyer and M. Rüffer, "Parametric built-in self-test of VLSI systems," in *Proc. Design, Automation and Test Eur.*, 1999, pp. 376–380.

[14] Z. Al-Ars *et al.*, "Optimizing stresses for testing DRAM cell defects using electrical simulation," in *Proc. Design, Automation and Test Eur.*, 2003, pp. 484–489.

Zaid Al-Ars (S'03) received the M.S. degree in electrical engineering with honors from the Delft University of Technology, Delft, the Netherlands, in 2000. He is working toward the Ph.D. degree in electrical engineering at the same university in cooperation with Infineon Technologies, Munich, Germany, where he is currently based.

He has published numerous papers in the field of electrical defect simulation, fault modeling, and test generation in memory devices. His research project involves systematic fault analysis, and test generation and optimization for commodity as well as embedded DRAM products.

Ad J. van de Goor (M'87–SM'98–F'00) received the M.S.E.E. degree from the Delft University of Technology, Delft, the Netherlands, in 1965 and the M.S.E.E. and Ph.D. degrees from Carnegie-Mellon University, Pittsburgh, PA, in 1970.

He worked with Digital Equipment Corporation, Maynard, MA, as the chief architect of the PDP-11/45 computer. He also worked for IBM in the Netherlands and in the United States, being responsible for the architecture of embedded systems. He is currently a Professor of Computer Engineering at the Delft University of Technology. He has written two books and over 150 papers in the areas of computer architecture and testing. He is on the Editorial Board of the *Journal of Electronic Testing: Theory and Applications*. His main research interests are in testing memories and logic.