Hard-to-Detect Fault Analysis in FinFET SRAMs

Cardoso Medeiros, G.; Fieback, M.; Wu, L.; Taouil, M.; Bolzani Poehls, L. M.; Hamdioui, S.

DOI
10.1109/TVLSI.2021.3071940

Publication date
2021

Document Version
Accepted author manuscript

Published in
IEEE Transactions on Very Large Scale Integration (VLSI) Systems

Hard-to-Detect Fault Analysis in FinFET SRAMs

Guilherme Cardoso Medeiros, *Student Member, IEEE,* Moritz Fieback, *Student Member, IEEE,* Lizhou Wu, *Student Member, IEEE,* Mottaqiallah Taouil, *Member, IEEE,* Leticia Bolzani Poehls, *Member, IEEE,* and Said Hamdioui, *Senior Member, IEEE*

***Abstract*—Manufacturing defects can cause Hard-to-Detect (HTD) faults in FinFET SRAMs. Detection of these faults, such as random read outputs and out-of-spec parametric deviations, is essential when testing FinFET SRAMs. Undetected HTD faults result in test escapes, which lead to early in-field failures. This paper presents a detailed analysis of HTD faults in FinFET SRAMs by exploring their sensitization and discussing solutions to improve HTD fault coverage during manufacturing testing. We first define the fault space for SRAMs and classify all faults in the space. Following this, we perform a systematic fault analysis based on injecting resistive defects in a memory cell, inspecting its behavior, and identifying HTD faults. Furthermore, we survey existing test solutions and discuss their HTD fault coverage and limitations. Based on our analysis, it is clear that no single test solution can fully detect all HTD faults, thus leading to test escapes. Hence, there is a need for new and more efficient test solutions. Improved detection of HTD faults could be achieved by using parametric test solutions, proposing solutions that cover yet-untargeted HTD faults, combining multiple test approaches into a single solution, and further exploring stress conditions. These new approaches would reduce test escapes and therefore improve the quality of FinFET SRAMs.**

***Index Terms*—Hard-to-Detect Faults, SRAM, FinFET, Fault Analysis, Testing, Test Solutions**

I. Introduction

The semiconductor industry has used the FinFET technology to continue the down-scaling of technological nodes [1] as it has improved short-channel behavior and overcomes the growing sub-threshold leakage problem of planar CMOS technology [2]. Despite their benefits, FinFETs also present challenges; one of them is small manufacturing defects. Although these defects may not impact the transistor's functionality, they can cause parametric deviations such as increased leakage current and small delays [3]. In FinFET *Static Random Access Memories* (SRAMs), these defects cause *Hard-to-Detect* (HTD) faults, such as random read and parametric faults [4]. HTD faults do not always lead to incorrect behavior; nevertheless, they compromise the device's quality as they may significantly impact memory parameters, such as *Bitline Swing* (BLS) and *Static Noise Margin* (SNM). Furthermore, undetected HTD faults result in test escapes, a known cause of early in-field failures, and no-trouble-found components [5]. To avoid these scenarios and achieve the quality demanded by applications using FinFET devices, such as automotive and aerospace [6, 7], new high-quality test solutions are necessary [8].

To develop new test solutions for SRAMs, it is necessary first to investigate faulty memories' behavior. Traditionally, this has been carried out by following a well-established methodology based on defect injection and electrical simulations. Works by Van de Goor et al. [9], Hamdioui et al. [4], and Dillilo et al. [10] are examples of such methodology. More recently, researchers have also started to analyze faulty behavior in FinFET SRAMs [11–14]. This methodology led to many different March algorithms [10, 15–18]. However, using only March tests is not a suitable solution to detect HTD faults [19] as these test solutions require faulty behaviors at the logic level. HTD faults (e.g., random read faults, undefined state faults) do not always lead to incorrect functional behavior; hence, they may not be detected by March tests, resulting in test escapes. A solution to reduce test escapes is using *Design-for-Testability* (DFT) circuits and stress tests. Examples of DFT methodologies that can enhance HTD faults' detection are schemes that perform memory operations differently [20–22] or schemes that monitor some parameters of the memory [23–25]. Even though there are different test solutions, it is still unclear if they can efficiently cover HTD faults. Thus, to develop new and high-quality test solutions that can fully cover HTD faults, it is necessary to obtain better insight into the occurrence of HTD faults, investigate existing test solutions, and provide strategies to improve these solutions with minimum test overhead (e.g., no additional test time and low hardware complexity).

This paper presents a systematic analysis of HTD faults and test solutions for FinFET SRAMs. In particular, we focus on analyzing the occurrence of HTD faults on FinFET SRAM cells affected by resistive defects, investigating the existing test solutions, and discussing new solutions for SRAM testing; to the best of our knowledge, this is the first study that focuses on the occurrence of HTD faults, such as random reads and undefined states, in FinFET SRAMs. First, we present a fault modeling approach that includes the fault space definition, a fault classification between *Easy-to-Detect* (ETD) and HTD faults, and a methodology to validate the proposed fault space. We then perform fault analysis by injecting resistive defects in the memory and identifying faulty behavior. This type of analysis has many benefits: it helps gain insight into HTD faults' occurrence, and it provides knowledge to improve and optimize manufacturing tests and diagnosis methodologies for yield learning and process optimization. Following the HTD fault analysis, we investigate existing SRAM test solutions and classify them based on their target, fault observation and

G. C. Medeiros, M. Fieback, L. Wu, M. Taouil, and S. Hamdioui are with the Computer Engineering Lab, Delft University of Technology, Delft, The Netherlands.

L. M. B. Poehls is with EASE, Pontifical Catholic University of Rio Grande do Sul, Brazil, and IDS, RWTH Aachen University, Germany.

Manuscript received XXXXXX; revised XXXX.

This work has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 722325.

identification methods, and stress conditions. Furthermore, we analyze their fault coverage, discuss their limitations, and propose strategies that the industry could follow to develop new test solutions with enhanced HTD fault coverage (i.e., reduce test escapes) and improve the quality of FinFET SRAMs. The main contributions of this paper are as follows:

- The extension of the Fault Primitive concept to better define, classify, and identify the FinFET SRAM fault space, including the subspace of HTD faults.
- A *systematic* fault analysis approach for HTD faults in FinFET SRAMs based on electrical-level simulations.
- The analysis of existing SRAM test solutions and the evaluation of their fault coverage regarding HTD faults.
- A discussion on how to improve SRAM test solutions' HTD fault coverage, including concepts and directions.

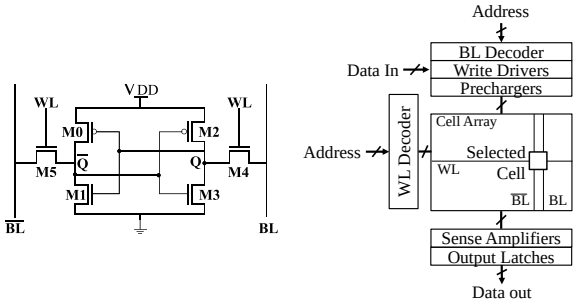
The rest of this paper is organized as follows. Section II provides a brief background on FinFET SRAMs. Section III introduces the fault modeling methodology, including the definition of the fault space, fault classification, and fault space validation. Section IV presents the fault modeling results for FinFET SRAMs based on defect injection and circuit simulation. Section V summarizes the existing tests for FinFET SRAMs, a discussion on their limitations, and recommendations for future test solutions. Finally, Section VI concludes this paper.

II. FinFET SRAMs

Static Random-Access Memories (SRAMs) are volatile memories consisting of a cell array and peripheral circuitry [26]. The cell array comprises 6T SRAM cells (Fig. 1a); the logic value stored in each cell corresponds to the voltage on node Q ('1' for V_{DD}, '0' for GND). Each 6T cell contains six transistors. Transistors M0, M1, M2, and M3 form two cross-coupled inverters; this enforces a '1' or '0' at Q, and the complementary value of Q at $\overline{Q}$. M4 and M5 are pass gates. Controlled by the Wordline (WL) signal WL, these transistors connect Q and $\overline{Q}$ to the Bitline (BL) signals BL and $\overline{BL}$.

The peripheral circuitry comprises address decoders, sense amplifiers (SAs), write drivers, prechargers, and output latches (Fig. 1b). An operation is performed by selecting an SRAM cell through the row and column address decoders; the first activates the appropriate WL while the latter activates the appropriate BL and $\overline{BL}$The operation can be either a read or a write. During a write operation, the write driver (dis)charges BL based on the value ('0' or '1') to be written, while $\overline{BL}$ is set to the complementary value. The WL signal is then asserted to match the voltage on the node Q ($\overline{Q}$) to the voltage on BL ($\overline{BL}$). During a read operation, both BL and $\overline{BL}$ are first pre-charged to V_{DD}. The WL signal is then activated; the node containing '0' will then discharge its respective BL. Subsequently, the SA is triggered by an SA enable signal SAE. The SA is connected to both BLs; it senses the voltage difference between BL and $\overline{BL}$ (i.e., the BL swing), amplifies this difference (amplification phase), and outputs the logic value '0' or '1'.

The memory used in this work is designed using $14\,\mathrm{nm}$ *Fin Field-Effect Transistors* (FinFETs). FinFETs are multi-gate transistors consisting of vertical silicon fins covered by a gate structure [2]. The dimensions of these fins, such as fin height (H_{FIN}) and fin thickness (T_{FIN}), define the transistor's driving strength. The effective width (W_{eff}) of a FinFET is defined as $W_{\mathrm{eff}} = 2H_{\mathrm{FIN}} + T_{\mathrm{FIN}}$. During the manufacturing process, inaccurate processing may affect these dimensions [27], resulting in a FinFET device bigger or smaller than designed. A smaller H_{FIN} and T_{FIN} lead to a reduced transistor's effective width, thus weakening the device. Furthermore, small shorts and fin cuts can lead to impaired FinFETs [28]. Although these defects may not lead to incorrect logic behavior, they will lead to small delays or increased leakage current [3]. Therefore, detecting these defective devices is of great significance to ensure high-quality FinFET SRAMs.



(a) 6T SRAM cell. (b) Memory organization.

Fig. 1. Memory model.

III. Fault Modeling

Manufacturing defects may lead to unexpected and undesired faulty behaviors in SRAMs. These faults can only be detected by using appropriate test solutions, which requires accurate fault modeling. In SRAMs, fault modeling is composed of three steps. It starts with defining a fault space, i.e., (all) possible faults that can occur in a given defective circuit, followed by classifying and grouping these faults. The fault space is then validated by following a fault analysis methodology to correlate faulty behaviors and manufacturing defects. These three steps (fault space definition, classification, and validation) are explained next.

A. Fault Space Definition

A fault space is defined by modeling possible faults that can occur in a given circuit; it is complete if it contains *all* faults that could occur in this circuit [29]. A fault can be either functional (i.e., related to incorrect logic behavior) or parametric (i.e., related to incorrect parametric behavior). We first discuss faulty functional behavior.

1) Functional Faults: Traditionally, *Fault Primitives* (FPs) [9] have been used to describe all faults that might lead to incorrect functional behavior. An FP is denoted by the three-tuple notation $\langle S/F/R \rangle$ as follows.

- ***S*** denotes the sequence of operations that sensitizes the fault. If S describes a fault in which only one cell is involved, the fault is defined as a single-cell fault. If S contains more than one cell, then the fault is defined as

a multi-cell fault. Due to the complexity of multi-cell faults (as they represent the combination of all possible operations and logic values for all cells involved), we limit our analysis to single-cell faults only. For such case, S takes the form of $S = x_0O_1x_1...O_nx_n$, where $x_i \in \{0,1\}$, $i \in \{0,1,...,n\}$, and $O \in \{r,w\}$. '0' and '1' denote logic values, while 'r' and 'w' denote a read and a write operation, respectively. n represents the number of operations necessary to trigger the fault. If $n \leq 1$, the fault is defined as a static fault. If $n \geq 2$, the fault is defined as a dynamic fault.

- ***F*** denotes the stored value in the cell after the fault takes place. While there are only two logic values that a cell can store ('0' and '1'), the voltage in nodes Q and $\overline{Q}$ may be different from V_{DD} or GND in the presence of a manufacturing defect. If the voltage difference between the two nodes is too small (i.e., the voltages on both nodes are almost the same), the cell may be storing an undefined value ('U') [4]. Thus, $F \in \{0,U,1\}$. Additionally, a subscript may specify the faulty effect's nature; '$_i$' and '$_t$' have been used to denote intermittent and transient faults in STT-MRAM devices [30]. We use the subscript '$_r$' to denote retention faults [31, 32], i.e., the cell's value flips some time (at least longer than the period of one memory operation) after the cell was stressed.
- ***R*** represents the read output if the last operation in S is 'r'. Here, $R \in \{0,1,?,-\}$, where '0' and '1' are logic values, '?' denotes a random readout value in case the BL swing is too small for the SA to sense the cell's content correctly, and '$-$' denotes the case where the last operation in S is not 'r'.

For example, the fault $\langle 1r1/0/? \rangle$ denotes a failed read '1' operation that flips the cell's content to '0' and returns a random readout value. Similarly, both the FPs $\langle 0w1/0_r/- \rangle$ and $\langle 0w1/0/- \rangle$ represent a failed write transition from '0' to '1'. The first FP is modified with a temporal component to indicate that the cell's impact occurs after a time interval longer than one memory operation, i.e., the cell's content goes back to '0' after some time following the write operation. On the other hand, the second fault does not include a temporal component; the cell's content is still '0' right after the write operation.

The name of an FP is defined by the combination of S, F, and R. Additionally, it also considers the number of operations in S (i.e., n). For $n = 0$, FPs are named as

$$\{ini\}F\{fin\}_{\{nat\}}, \tag{1}$$

while for $n = 1$ FP are described as

$$\{out\}\{opn\}\{opd\}\{eff\}F\{fin\}_{\{nat\}}. \tag{2}$$

Each field in the name template (1) and (2) describes a specific fault characteristic as follows:

- ini describes the cell's initial state; $ini \in \{0,1\}$.
- fin describes the cell's final state; $fin \in \{0,U,1\}$.
- nat is an optional modifier based on the fault's nature. For this work, the subscript '$_r$' is used to specify retention faults, i.e., the fault's impact on the cell's final state fin occurs after a time interval longer than one memory operation.
- out describes the read operation's output; $out \in \{i,r,d\}$, where 'i' means an incorrect read output, 'r' a random read output, and 'd' a deceptive read output (i.e., $ini \neq fin$). This field is disregarded if a write operation is the last operation in S.
- opn describes the last operations in S; $opn \in \{W,R\}$, where 'W' stands for a write operation while 'R' stands for a read operation.
- opd describes the operand of the operation opn; $opd \in \{0,1\}$.
- eff describes the operation's effect on the faulty cell; $eff \in \{T,D,N\}$, where 'T' denotes a transition operation (i.e., S=0w1 or S=1w0), 'D' denotes a destructive operation, and 'N' denotes a non-destructive operation.

TABLE I
COMPLETE SPACE OF FUNCTIONAL FAULTS.

#	S	F	R	FP Notation	FP Name	Functional Fault Model
1	0	1	-	⟨0/1/-⟩	S0F1	State Fault
2	0	U	-	⟨0/U/-⟩	S0FU	
3	1	0	-	⟨1/0/-⟩	S1F0	
4	1	U	-	⟨1/U/-⟩	S1FU	
5	0w1	0	-	⟨0w1/0/-⟩	W1TF0	Write Transistion Fault
6	0w1	U	-	⟨0w1/U/-⟩	W1TFU	
7	1w0	1	-	⟨1w0/1/-⟩	W0TF1	
8	1w0	U	-	⟨1w0/U/-⟩	W0TFU	
9	0w0	1	-	⟨0w0/1/-⟩	W0DF1	Write Disturb Fault
10	0w0	U	-	⟨0w0/U/-⟩	W0DFU	
11	1w1	0	-	⟨1w1/0/-⟩	W1DF0	
12	1w1	U	-	⟨1w1/U/-⟩	W1DFU	
13	0r0	0	1	⟨0r0/0/1⟩	iR0NF0	Incorrect Read Non-Destructive Fault
14	1r1	1	0	⟨1r1/1/0⟩	iR1NF1	
15	0r0	0	?	⟨0r0/0/?⟩	rR0NF0	Random Read Non-Destructive Fault
16	1r1	1	?	⟨1r1/1/?⟩	rR1NF1	
17	0r0	1	1	⟨0r0/1/1⟩	iR0DF1	Incorrect Read Destructive Fault
18	0r0	U	1	⟨0r0/U/1⟩	iR0DFU	
19	1r1	0	0	⟨1r1/0/0⟩	iR1DF0	
20	1r1	U	0	⟨1r1/U/0⟩	iR1DFU	
21	0r0	1	0	⟨0r0/1/0⟩	dR0DF1	Deceptive Read Destructive Fault
22	0r0	U	0	⟨0r0/U/0⟩	dR0DFU	
23	1r1	0	1	⟨1r1/0/1⟩	dR1DF0	
24	1r1	U	1	⟨1r1/U/1⟩	dR1DFU	
25	0r0	1	?	⟨0r0/1/?⟩	rR0DF1	Random Read Destructive Fault
26	0r0	U	?	⟨0r0/U/?⟩	rR0DFU	
27	1r1	0	?	⟨1r1/0/?⟩	rR1DF0	
28	1r1	U	?	⟨1r1/U/?⟩	rR1DFU	

In case $n \geqslant 2$, an additional field $\{n$d-$\}$ is prefixed to name template (2), indicating that the dynamic FP requires n operations to be sensitized. Based on this nomenclature, we draw a list with all single-cell static FPs followed by their notation and names, as shown in Table I; we do not include the nat modifier as it does not change the name of a fault. The list also groups the FPs into functional fault models [4]; for example, the functional fault model Write Transition Faults comprises the FPs W1TF0, W1TFU, W0TF1, and W0TFU. Additionally, this table could be expanded for dynamic faults by increasing the number of operations in S.

2) Parametric Faults: While it is important to understand functional faults' behavior, it is equally important to understand that manufacturing defects may also lead to parametric faulty behavior. This type of faulty behavior has been

TABLE II
FAULT SPACE OF PARAMETRIC FAULTS.

#	Parameter	Fault Model
29	BLS	Reduced Bitline Swing
30	SNM	Reduced Noise Margin
31	RNM	
32	PCH	Increased Power Consumption
33	PCR	
34	PCTW	
35	PCNW	

discussed in recent years in modeling and simulation of analog faults [33]. For SRAMs, a parametric faulty behavior means that a cell's performance characteristics are outside the expected range due to a small defect or extreme *process variation* (PV), thus failing one of its parametric specifications. Despite no functional impact, parametric faults may lead to early in-field failure and must be considered, especially for task-critical applications such as aerospace and automotive.

Unlike functional faults, it is impossible to define a complete fault space for parametric faults in SRAMs due to the number of parameters involved in such a circuit. Therefore, we have selected a set of critical circuit parameters to define parametric faults. Specifically, when one of the selected parameters is out of the pre-defined spec, we refer to it as a parametric fault. These parameters are listed in Table II. We do not define names for parametric faults. Instead, we list the parameter and group them into bigger parameter groups. BLS stands for *Bitline Swing*, i.e., the voltage difference between BL and $\overline{BL}$ during a read operation. SNM and RNM stand for *Static Noise Margin* and *Read Noise Margin* of the cell, which is the minimum noise voltage able to flip the cell's state during hold mode or a read operation, respectively. Finally, PCH, PCR, PCTW, PCNW denote the *Power Consumption* (PC) during four scenarios: hold mode (H), a read operation (R), a transition write operation (TW), and a non-transition write operation (NW). While we have limited our work to these specific parameters, any parameter could be included in the fault space as long as it has a performance specification.

B. Fault Classification

We propose to classify memory faults based on their impact and detection requirements, as shown in Fig. 2. The first-level classification considers the fault impact, i.e., whether the fault impacts the memory's functionality or parameters. This distinction is the same used to define the fault space. Therefore, the $\langle S/F/R \rangle$ notation can describe any functional fault. Functional faults have also been referred to as strong faults [31]. Contrarily, parametric faults cannot be described by the $\langle S/F/R \rangle$ notation; they are only parametric deviations. Parametric faults have also been referred to as weak faults [31].

The second level of this classification consists of their detection conditions. Faults whose detection is guaranteed using only read and write operations are classified as functional *Easy-to-Detect* (ETD) faults. They have deterministic behavior, and therefore will **always** lead to a logic faulty behavior that can be detected by writing or reading the memory cell. Contrarily, faults whose detection is not guaranteed using only read and write operations are classified as *Hard-to-Detect* (HTD) faults; they can be either functional HTD or parametric HTD. Table III shows the ETD and HTD classification of the previously-introduced faults. Next, we briefly explain each of them.

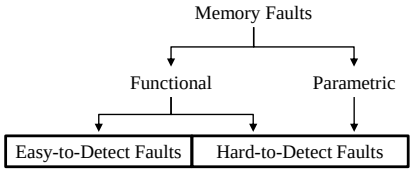


Fig. 2. Two-level fault classification.

TABLE III
SET OF ETD AND HTD FAULTS.

ETD Faults	HTD Faults	
Functional ETD	Functional HTD	Parametric HTD
S0F1, S0F1	S0FU, S1FU	BLS
W0TF1, W1TF0	W0TFU, W1TFU	SNM, RNM
W0DF1, W1DF0	W0DFU, W1DFU	PCH, PCR, PCTW, PCNW
iR0NF0, iR1NF1	rR0NF0, rR1NF1	
dR0DF1, dR1DF0	dR0DFU, dR1DFU	
rR0DF1, rR1DF0	rR0DFU, rR1DFU	
iR0DF1, iR0DFU, iR1DF0, iR1DFU		

1) Functional ETD Faults: Functional ETD faults are faults that always impact the functionality of the memory. Therefore, test approaches that rely on logic fault observation can easily detect them. For example, W1TF0 = $\langle 0w1/0/- \rangle$ can be sensitized by writing '1' on a cell containing '0' and be detected by immediately reading this cell. Due to its high impact in the functionality of the memory, functional ETD faults have been extensively analyzed in the literature [10, 17, 31, 34–37]. This has led to the development of many well-known March algorithms that guarantee to detect ETD faults, such as March SS [38], March C- [39], March AB [16], and more recently the FinFET-specific March FFDD [18].

2) Functional HTD Faults: Functional HTD faults are faults whose detection is not guaranteed by performing read and write operations on the cell. Unlike Functional ETD faults, Functional HTD faults do not always impact the memory's functionality; they are related to random read outputs and undefined cell state. Considering that random effects such as PV impact the outcome of functional HTD faults, it is statistically expected that March tests will detect only part of these faults due to incorrect logic behavior. The remaining faults that do not cause incorrect logic behavior will result in test escapes and compromise the circuit's reliability. Therefore, only special testing solutions can guarantee the detection of functional HTD faults. Although HTD faults may not lead to functional errors at time zero, they may cause reliability issues. Undetected HTD fault may cause random faulty behaviors if the memory is used in a harsh environment or conditions, leading to soft errors. Furthermore, HTD faults' impact may worsen once the memory ages, thus leading to a higher failure rate and a shorter lifetime [40]. Thus, detecting HTD faults

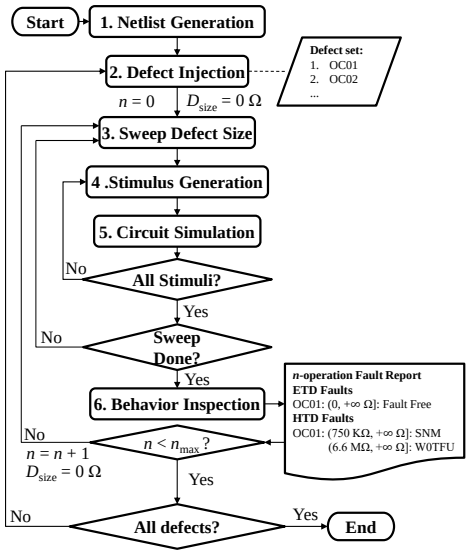


Fig. 3. Fault space validation methodology.

and avoiding test escapes is of high importance.

3) Parametric HTD Faults: Parametric HTD faults are severe parametric deviations that cause the memory cell to fail one or more of its specifications. As they do not impact memory cells' functionality, their detection is not guaranteed by performing read and write operations. From the logic functionality point of view, these faults are undetectable, as all operations pass correctly. Consequently, only special test solutions can guarantee the detection of parametric HTD faults.

C. Fault Space Validation Methodology

Once the fault space is defined, a systematic circuit simulation approach validates it. This validation is necessary to establish whether the faults defined in the fault space are realistic or not, i.e., if manufacturing defects can sensitize the faults defined in the fault space. As shown in Fig. 3, our approach consists of six steps:

1) Netlist Generation: a SPICE-level netlist of a memory (array and peripherals) is generated; this is a one-step procedure.
2) Defect Injection: a defect is introduced in the memory cell, one at a time, to sensitize faults. For this work, we have modeled the inject defects as resistor components. Additionally, the number of operations in S and the defect size D_{size} are (re)set (i.e., n = 0, D_{size} = 0 Ω).
3) Defect Size Sweep: The resistor component's resistance is swept within the specified defect range. If $n \leq 1$, this range is [0,+∞), i.e., all possible defect sizes. If $n \geq 2$, this range consists of the defect sizes in which no ETD faults were observed in previous iterations. For each iteration of sweep defect size, D_{size} is increased.
4) Stimuli Generation: based on n, the stimuli S is generated. It includes an initial condition and n memory operations. For $n = 0$, S can be $\{0,1\}$; for $n = 1$, S can be $\{0w0, 0w1, 1w0, 1w1, 0r0, 1r1\}$, and so forth.
5) Circuit Simulation: the netlist including a defect, a defect size D_{size}, and stimuli S is simulated using a Spice simulator.
6) Behavior Inspection: once all defect sizes and stimuli for a given defect and n are simulated, an (automated) behavior inspection is carried out. This step consists of analyzing all measurements from the circuit simulation step and identifying faults. The result is a report containing all observed faults alongside their respective defect size ranges.

All steps are performed sequentially. A loop between steps 2 and 6 guarantees that the methodology covers all combinations of defects, defect sizes, and stimuli.

The Behavior Inspection is the methodology's most critical step, as it translates the measured electrical behaviors to the faulty behaviors defined in the fault space. While it is easy to identify when the defect causes an incorrect logic behavior in the memory, the occurrence of random reads, undefined states, and parametric faults due to severe parametric deviation is strongly dependent on the memory's specifications. Thus, it is mandatory first to define the memory's specifications to identify the occurrence of such faults correctly:

- Random Read Fault: a SA circuit has a specification for minimal input, which is the minimum BLS that guarantees that the SA will correctly output the cell's value. A Random Read occurs when the SA's input is smaller than its specification. It is important to note that this fault is related to the SA's specification and how it is affected by PV effects rather than the BLS specification.
- Undefined State Fault: this fault occurs when the voltage difference between nodes Q and $\overline{Q}$ is smaller than a predefined threshold. Previous works that have dealt with undefined state faults in older SRAM technologies have not defined this threshold [4, 31]. The acceptable ΔV between true nodes changes from application to application; non-critical applications may tolerate a small ΔV between true nodes, while critical applications (e.g., automotive, aerospace) may require a greater ΔV. For this work, we aim at identifying HTD faults from the perspective of a critical application. Therefore, we specify this threshold as half V_{DD}. Thus, a cell is in an Undefined State if $|V(Q) - V(\overline{Q})| \leq V_{DD}/2$.
- Parametric Fault: this fault occurs when one of the memory's parameters (e.g., BLS, noise margin, power consumption) fails its specification, i.e., its performance characteristics are outside its expected range. In the context of analog faults, it has been reported [33] that this range is typically three standard deviations (σ) of the mean (μ) of a given parameter. Nevertheless, it is common for memory engineers to design SRAMs that withstand parameter variations of 5σ or even 6σ [41]. In our work, we only want to detect *extrinsic* variations (i.e., variations caused by manufacturing defects) that surpass the typical range of *intrinsic* variations (i.e., variations caused by process variation). Therefore, we define the threshold between intrinsic and extrinsic variation (i.e., PV-induced variation and defect-induced variation, respectively) as

$\pm$ 6σ. Accordingly, deviations within the $\mu\pm6\sigma$ range are considered intrinsic variations; parametric deviations outside this range are considered extrinsic variations and consequently parametric faults. A too relaxed threshold causes intrinsic variations to be classified as extrinsic (i.e., a defect-free device is signaled as defective), leading to further yield loss. Note that the definition of this range is flexible and can change based on the application; critical applications may not tolerate a variation of 3σ, and thus variations greater than 3σ are already flagged as faulty. Notwithstanding, changing this range modifies only the resistance boundaries in which a fault is sensitized; the overall fault trends are expected to remain the same.

Once the behavior inspection for a given defect (including defect size sweep and all n-operation stimuli) has finished, steps 3 to 6 are repeated for $n+1$. The defect size D_{size} is reset to 0 Ω, and the defect size range and S are adjusted. If $n < 2$, the defect size range is set to $[0, +\infty)$. If $n \geq 2$, the defect size range is adjusted based on previous iterations' reports; only defect sizes in which no ETD faults were observed are analyzed. The lower bound of the defect size range is the smallest defect size possible, and the upper bound is the smallest defect size that triggered an ETD fault. For example, consider that for a given defect, ETD faults were observed in the range $[50\ k\Omega, +\infty)$ when $n = 1$. For $n = 2$, the defect size range investigated will be then limited to $[0,\ 50\ k\Omega]$.

Results from previous iterations are also used to limit the stimuli. Since it is infeasible to investigate all possible combinations of memory operations for large values of n, S must be limited. Thus, only operations that triggered unexpected behaviors are used as the base of S. For example, consider that for a given defect, no faults were observed when $n = 1$. However, two unexpected behaviors were observed: a prolonged transition when applying the sequence $0w1$ and a slightly reduced BL swing when applying the sequence $0r0$. Such behaviors can be an indication of dynamic faulty behavior. Thus, for $n = 2$, $0w1$ and $0r0$ will be used as the base of S. Only the sensitizing sequences $\{0w1w1,\ 0w1w0,\ 0w1r1,\ 0r0w1,\ 0r0w0,\ 0r0r0\}$ are applied to the defective cell; remaining combinations of S for $n = 2$ are excluded from the analysis. This process is repeated until n reaches a user-defined maximum number of operations (n_{max}).

The aim of increasing the number of sensitizing operations and limiting the defect size range and S is to reduce the defect size ranges that cause HTD faults while enlarging the ranges that lead to ETD faults. Enlarging the range in which ETD faults are sensitized can optimize and cheapen testing costs since ETD faults can be detected by March tests (i.e., cheap test solutions), while HTD faults require *Design-for-Testability* (DFT) designs or stress tests to detect them (i.e., expensive test solutions).

This fault analysis methodology is useful to optimize the trade-off between the test quality and the overheads related to developing a new test solution (e.g., silicon area, time, costs). In the next section, we will apply this methodology to validate the previously proposed fault space. In summary, the methodology consists of injecting defects in a memory array, performing electrical simulations, inspecting the memory's behavior, and identifying faults. This way, we can determine which faults in the fault space are realistic, i.e., actually occur in a faulty memory.

IV. Simulation Setup and Results

In this section, we validate the fault space through SPICE-based circuit simulations. First, we introduce our simulation setup, including details about the simulated circuits, the Monte Carlo analysis, and the injected defects. We then present the simulation results from the Monte Carlo analysis and the results obtained following the 6-step methodology presented previously.

A. Simulation Setup

The memory netlist is described in SPICE using the Predictive Technology Model (PTM) $14\,\mathrm{nm}$ FinFET library [42]. The memory array comprises 128 rows and 64 columns where each logical word contains 32 bits; hence, two neighboring columns share a single write driver, SA, and prechargers. Capacitive loads are applied to BLs and WLs to emulate a $1\,\mathrm{kB}$ memory. The memory works on a nominal supply voltage of $0.8\,\mathrm{V}$ and a clock frequency of $2\,\mathrm{GHz}$. Additional timing circuits are used to generate control signals. This memory model is used in both memory spec identification and fault modeling analysis. The first comprises thousands of simulations to estimate the impact of PV on the memory's parameter, while the second comprises injecting defects and inspecting the electrical behavior under different sensitizing sequences. These analyses are further described below:

1) Monte Carlo Setup: To accurately distinguish parametric deviations caused by defects from parametric deviations caused by PV, we must first perform Monte Carlo simulations to estimate PV's impact on the memory's parameters and define the memory's specs. In this work, we have introduced a voltage source on the gate terminal of transistors [43] to emulate the PV's impact on $\mathrm{V_{TH}}$ variation at time zero [44]. We simulated the memory operating under four distinct scenarios: hold mode (i.e., idle), read operation, non-transition write operation, and transition write operation; each scenario is simulated 10,000 times. We measured different operating parameters in each scenario, such as power consumption, BLS, and noise margin. The mean value derived from measurements is defined as the specification, and the calculated standard deviation is then used to define the parametric deviation threshold.

2) Defect Injection Setup: The injected defects consist of twenty-eight single-cell resistive defects, as shown in Fig. 4. They are either *Resistive-Open* (RO), *Resistive-Short* (RS), or *Resistive-Bridge* (RB) defects [4]. RO defects are unintended series resistances within an existing connection. They are labeled as *Open Connections* (OC) 01 to 12. RSs and RBs are unintended resistive connections between two nodes. In more detail, RSs are shorts to power nodes ($\mathrm{V_{DD}}$ or GND); they are named *Short in the Cell* (SC) 01 and 02, *Short in the Bitline* (SB) 01 and 02, and *Short in the Wordline* (SW) 01 and 02. In contrast, RBs are shorts between any two other nodes of the cell. They are identified as *Bridges Connections*

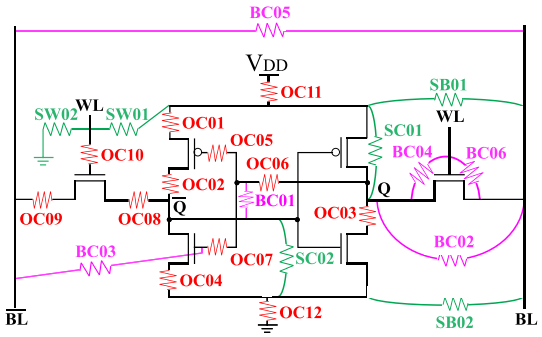


Fig. 4. Injected resistive defects: opens, shorts, bridges.

TABLE IV
RESULTS OF THE MONTE CARLO ANALYSIS.

Parameter	Mean (μ)	Standard Deviation (σ)	Condition for Parametric Fault
BLS	168 mV	7.58 mV	BLS $<$ 122.5 mV
SNM	347.2 mV	5.41 mV	SNM $<$ 341.74 mV
RNM	191 mV	8.01 mV	RNM $<$ 141.92 mV
PCH	43.19 pW	4.4 pW	PCH $>$ 70.4 pW
PCR	63.0 µW	2.1 µW	PCR $>$ 75.9 µW
PCTW	102.2 µW	2.7 µW	PCTW $>$ 118.5 µW
PCNW	27.2 µW	6.2 µW	PCNW $>$ 64.1 µW

(BC) 01 to 03. Due to the symmetry of 6T cells, all defects have symmetrical opposites, e.g., OC01 can be injected on the pull-up of either $\overline{Q}$ or Q; defect BC01 is an exceptional case to this rule. Symmetrical opposite defects will lead to a similar faulty behavior to their counterparts and are therefore neglected in our analysis.

Each simulation comprises one defect, one defect size, and one sensitizing sequence S of n operations. We have set an n_{max} of 30 operations, i.e., S contains at most 30 operations. Each combination of a resistive defect and a sensitizing sequence was simulated with (at most) 100 different defect sizes by sweeping the resistance value from 0 Ω to 100 GΩ (representing $+\infty$), logarithmically spaced.

B. Memory Spec Identification

This section presents the results obtained from the Monte Carlo analyses' measurements; we aim to obtain the mean and standard deviation of the operating parameters (e.g., BLS, SNM) to define the memory's spec. Table IV shows a summary of the measured parameters, their mean (μ), their standard variation (σ), and the condition to determine a parametric fault. The last represents the threshold between a parameter deviated by (extreme) PV and a deviation caused by a manufacturing defect; it is defined as $\mu \pm 6\sigma$. It is worth noting that the Monte Carlo analysis did not lead to functional faults; only parametric deviations were observed. Furthermore, no parametric fault was observed within the $\pm 6\sigma$ range. Therefore, it is safe to assume that the parametric faults identified during fault modeling are caused by manufacturing defects rather than PV. Reducing the $\pm 6\sigma$ range could lead to the incorrect indication of PV-induced variation as defect-induced.

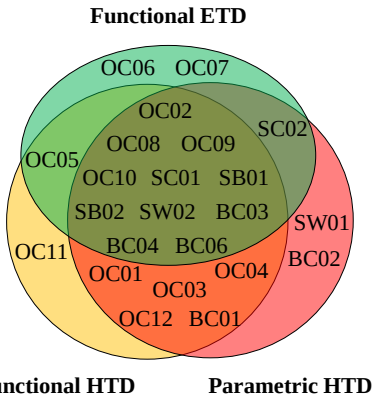


Fig. 5. Distribution of resistive defects based on sensitized faults.

C. Fault Modeling Results

This section presents the results obtained from the reports generated on the Behavior Inspection step of the fault space validation methodology. We analyze all defects assuming wide size ranges and identify which faults were sensitized by each defect. Table V shows a summary of the observed static faults by each defect. All the functional fault models previously listed in Table I have been observed, with at least one FP of each functional fault model being sensitized. FPs that were not sensitized are not included in Table V; in total, 24 of all the 35 faults investigated in this work were sensitized and are therefore listed in Table V. Nevertheless, it is worth mentioning that some faults could have been sensitized by changing the simulation setup in two manners: 1. including symmetrical opposite defects (if a defect sensitizes an R0DF1 fault, its symmetrically opposed defect sensitizes an R1DF0 fault); and 2. reducing the defect size sweep step (a correct read may become a deceptive read, and finally a destructive read as the defect strength increases; if the sweep step is too big, the intermediate behavior is not observed).

Table V is divided into three segments representing the three different fault classes. The top segment focuses on functional ETD faults; the middle segment focuses on functional HTD faults, while the bottom segment focuses on parametric HTD faults. Traditional March tests, which use only write and read operations, can guarantee to detect the faults listed only in the top segment. Thus, defects that do not lead to functional ETD faults cannot be detected by March tests. Fig. 5 further explores this limitation by showing the distribution of defects among sensitized faults. March tests can detect defects only on the functional ETD space (green circle). However, March tests do not guarantee the detection of defects on the functional HTD or parametric HTD space (yellow and red circle, respectively) or their intersection. Furthermore, it is also possible that a weaker version of a defect that can cause a functional ETD fault is only sensitizing an HTD fault, therefore limiting the efficacy of March tests.

Based on results presented on Table V and Fig. 5, we can conclude the following:

- Covering only functional ETD faults is not enough as most defects will lead to some form of HTD faulty behavior. From the set of 24 resistive defects investigated,

TABLE V
SINGLE-CELL STATIC FAULT MODELING RESULTS OF ALL INJECTED DEFECTS.

Fault Type	Fault Prim.	Defect OC01	OC02	OC03	OC04	OC05	OC06	OC07	OC08	OC09	OC10	OC11	OC12	SC01	SC02	SB01	SB02	SW01	SW02	BC01	BC02	BC03	BC04	BC05	BC06
Functional ETD	S0F1													X	X						X	X			
	S1F0																						X		
	W0DF1																X					X			
	W0TF1					X	X	X	X	X	X					X	X		X			X			X
	W0TF1$_r$		X			X																			
	W1TF0						X												X						
	iR0NF0																X					X	X		
	iR0DF1														X		X								
	rR0DF1													X	X										
Functional HTD	S0FU																			X					
	S1FU																			X					
	W0TFU	X	X			X						X													
	W1TFU											X													
	rR0NF0			X	X				X	X	X		X	X		X	X		X			X	X	X	
	rR1NF1												X						X					X	X
	rR0DFU																			X					
	rR1DFU																			X					
	dR0DFU																			X					
Parametric HTD	BLS			X	X				X	X	X		X	X	X	X	X		X	X		X	X	X	X
	SNM	X	X	X	X									X	X			X		X	X	X	X		
	RNM			X	X									X	X					X	X	X	X		
	PCH													X	X	X	X	X		X	X	X	X		X
	PCR													X	X		X	X	X		X		X		X
	PCTW													X		X	X	X	X					X	X
	PCNW													X		X	X	X	X		X	X	X	X	X

22 led to HTD faults. Furthermore, 8 out of 24 defects led to HTD faults only, parametric or functional. Without test solutions that also target HTD faults, detecting these defects is not guaranteed. Undetected HTD faults become test escapes, leading to early in-field failures and no-trouble-found components.

- Some defects will lead to both functional and parametric HTD faults. In those cases, test engineers have the flexibility to choose between test solutions that focus on functional faults or parametric faults.
- There are a few defects that only trigger one type of fault; the majority of defects will sensitize a combination of functional ETD, functional HTD, and parametric HTD faults. In general terms, weaker defects lead to parametric HTD faults, while stronger defects lead to functional HTD and ETD faults.

To better analyze the occurrence of static HTD faults, we explore the occurrence of these faults in five example defects, namely OC01, OC05, OC08, SW02, and BC01. Table VI lists the results. We categorize the faults sensitized by each defect based on defect size ranges. We define these ranges as *fault classes* (FCs). Each FC contains a set of faults that are sensitized for a given defect size range. Furthermore, an FC may be a subset of another FC, i.e., if FC-1 $\subset$ FC-2, FC-2 contains all the faults in FC-1, plus other faults not in FC-1. The expressive occurrence of HTD faults illustrates the importance of having additional DFT designs or stress tests to detect HTD faults, even when functional ETD faults are sensitized.

Since detecting most of the faults caused by resistive defects is not guaranteed using single operations, we increased the number of consecutive operations (i.e., $n + 1$) to investigate if static HTD faults can lead to dynamic ETD faults. Only resistive-open defects led to dynamic faults; Table VII outlines the results for 3 exemplary defects. Defect OC01 sensitizes dynamic parametric HTD faults for defect sizes much smaller than the HTD faults sensitized for $n \leq 1$; this is due to the disturbance generated when writing the defective memory cell. Nevertheless, this disturbance only generates parametric deviations. Therefore, dynamic stress does not lead to ETD faulty behavior for this defect. On the other hand, defect OC03 led to dynamic ETD faulty behavior for $n \geq 2$. While smaller defects may require up to 27 consecutive operations to sensitize a functional ETD fault, bigger defects require a maximum of 4 consecutive operations to sensitize a functional ETD fault. Even though dynamic functional ETD faults are sensitized, the high number of consecutive operations necessary to trigger them is a significant limiting factor in their detection. It

TABLE VI
SINGLE-CELL STATIC FAULT MODELING RESULTS OF RESISTIVE DEFECTS.

Defect Model	Resistance (Ω)	Fault Class	Faults Observed
OC01	(0, 474 K]	-	Fault Free
	(474 K, 3.8 M]	1	SNM
	(3.8 M, +∞)	2	FC-1 + W0TFU
OC05	(0, 3.8 M]	-	Fault Free
	(3.8 M, 6.6 M]	1	W0TF1
	(6.6 M, +∞)	2	W0TFU, W0TF1$_r$
OC08	(0, 2 K]	-	Fault Free
	(2 K, 20 K]	1	BLS
	(20 K, 29.3 K]	2	FC-1 + rR0NF0
	(29.3 K, +∞)	3	FC-2 + W0TF1
SW02	(0, 200.9]	6	FC-5 + rR0NF0, rR1NF1
	(200.9, 473.6]	5	FC-4 + W0TF1, W1TF0
	(473.6 , 1.1 K]	4	FC-3 + BLS
	(1.1 K, 10.2 K]	3	FC-2 + PCNW
	(10.2 K, 29.3 K]	2	FC-1 + PCTW
	(29.3 K, 38.4 K]	1	PCR
	(38.4 K, +∞)	-	Fault Free
BC01	(0, 11 K]	6	FC-3 + S0FU, S1FU
	(11 K, 20.2 K]	5	FC-4 + rR0DFU, rR1DFU
	(20.2 K, 29.3 K]	4	FC-3 + dR0DFU, dR1DFU
	(29.3 K, 202 K K]	3	FC-2 + RNM
	(202 K, 1.1 M]	2	FC-1 + SNM
	(1.1 M, 22.5 G]	1	PCH
	(22.5 G, +∞)	-	Fault Free

TABLE VII
SINGLE-CELL DYNAMIC FAULT MODELING RESULTS OF RESISTIVE OPENS.

Defect Model	Resistance (Ω)	Fault Class	Faults Observed
OC01 & OC02	(0, 112.5 K]	-	-
	(112.5 K, 150 K]	FC-1	2-PCH
	(150 K, +∞)	FC-2	FC-1 + 2-SNM, 2-PCH
OC03	(0, 15 M]	-	-
	(15 M, 16 M]	FC-1	27-rR0DF1
	(16 M, 17 M]	FC-2	16-rR0DF1
	(17 M, 20 M]	FC-3	9-rR0DF1
	(20 M, 50 M]	FC-4	5-rR0DF1
	(50 M, +∞)	FC-5	4-rR0DF1

may not be viable to perform that many instructions to each memory cell, considering time and cost constraints in modern memory testing. Furthermore, defects smaller than 15 MΩ do not sensitize any functional ETD faults. Therefore, a more effective way to detect defects OC03 and OC04 is by using special testing.

Using a test solution that only targets static functional ETD faults will lead to test escapes as it does not guarantee HTD fault detection. Furthermore, the occurrence of static HTD faults does not necessarily mean the occurrence of dynamic ETD faults. Thus, special test solutions are necessary to guarantee the detection of HTD faults. In the next section, we discuss the existing test solutions for SRAMs, detailing aspects related to their stressing conditions, fault coverage, and limitations.

V. TEST SOLUTIONS

When developing a memory test solution, test engineers first define a fault space. We can verify which faults manufacturing defects will cause in the circuit under test by validating this fault space. Then, the test solution's targeted faults are defined, and a method to detect such faults is developed. This section proposes a classification for test solutions based on the methods they use for testing. We classify existing test solutions for SRAMs and discuss their fault coverage and limitations. Lastly, we examine possible opportunities for test engineers to develop new test solutions with improved fault coverage based on what is still missing from previously proposed solutions.

A. Classification

Memory test solutions represent efforts for testing a memory circuit. The main goal of these solutions is to provide the detection of faulty behaviors. Test solutions use many different methods to test a circuit, from observing and identifying a fault to stressing the memory. To analyze well-known memory test solutions, we first propose to classify them, as shown in Fig. 6. The first classification level is based on what they are testing: functionality or memory parameters. A functional test solution is a procedure that focuses on the logic functionality of the memory, i.e., if read and write operations are performed as expected. Differently, a parametric test solution focuses on identifying parametric deviations and if parameters are within the tolerated performance ran

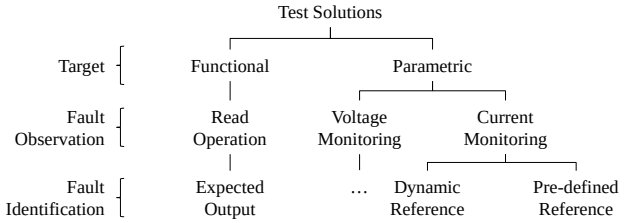


Fig. 6. Classification of memory test solutions.

Once the testing target is defined, the second classification level is based on the fault observation method. Functional test solutions can only observe faults by performing read operations, as they test for correct functionality. However, parametric test solutions observe faults by monitoring the memory's electrical parameters, such as voltage or current.

The third and final classification level is based on fault identification, i.e., defining if the observed behavior is indeed a fault. Functional test solutions can only identify faults by verifying the memory's output, i.e., comparing the read output with an expected output. Alternatively, parametric test solutions rely on comparing the measured electrical parameter with a reference value. The reference can be dynamic (e.g., measurements from other components in the same circuit) or pre-defined (e.g., a threshold defined during the design phase).

Additionally, every test solution uses *Stressing Conditions* (SCs) to trigger the targeted faults; they may use a single SC or a combination of different SCs. We split SCs between two types: algorithm-related stress and environment-related stress [45]. Algorithm-related stress specifies the algorithm that will be applied to the memory cells. It includes all types of SCs derived from using only write and read operations. Examples of algorithm-related SCs are as follows:

- **Base Test (BT)**: A BT is a sequence of operations (reads and writes) applied to a memory cell.
- **Address Order (AO)**: The AO indicates in which sequence the algorithm accesses addresses. It can be increasing (⇑), decreasing (⇓), or irrelevant (⇕).
- **Address Direction (AD)**: The AD indicates how the one-dimensional AO is applied in the two-dimensional cell array. It is generated by addressing methods such as fast *x* [45] and H2/H3/HN1 [46].
- **Data Background (DB)**: DB is the pattern of ones and zeros, as seen in the memory array. The most known DBs are Solid, Checkerboard, and Row/Column Stripe [45].

In contrast, environment-related SCs use additional stress sources to create unrealistic SCs under nominal operating conditions. These include [45], but are not limited to:

- **Voltage Stress**: An additional source applies voltage stress to the circuit. It can apply additional stress in the entire circuit (e.g., changing the supply voltage) or specific components (e.g., write driver, SA).
- **Timing Stress**: An additional source applies timing stress to the circuit. It can apply the additional stress in the entire circuit (e.g., changing the circuit's frequency) or specific components (e.g., write driver, SA) to change memory operations' timing.

- **Temperature Stress**: The temperature is either increased or reduced from its nominal value during testing.

B. Previously Proposed Test Solutions

Table VIII summarizes the most relevant memory test solutions found in literature, categorized following the classification previously presented. Furthermore, the table identifies each test solution's stressing conditions, their HTD fault coverage, and their limitations. We do not discuss Fault coverage for functional ETD faults since any test solution using appropriate algorithm-related stress can guarantee the detection of these faults.

We first discuss functional test solutions, which use read operation as the fault observation method, and compare the obtained output to an expected output value to identify faults. **Ad-hoc tests** are one of the oldest forms of structured memory test solutions [50]. Examples of ad-hoc tests include GalPat [50], SCAN [51], and Walking 1/0 [29]. **March tests** are the evolution of ad-hoc tests; they are based on FPs and have a linear time complexity. Examples of march tests include March SS [38], March C- [39], March AB [16], and March FFDD [18]. Besides algorithm stress, ad-hoc and march tests have also used voltage stress [47]. Both tests can only Partially (P in the table) cover random read functional HTD faults due to the random nature of this kind of fault; they do not detect any of the other HTD faults. On the other hand, they do not lead to yield loss, as they will not indicate fault-free cells as faulty and do not require any post-silicon calibration. Furthermore, they do not require any Hardware Modification (HW Mod. in the table) as they do not introduce additional hardware in the memory.

The remaining three functional test solutions rely on environment-related stress to change the execution of operations. **Self-timed circuits** [21] uses time stress to change the timing of the WL signal, effectively increasing or decreasing the time in which the cell can discharge the BLs during a read operation, or the write driver can write the new value into the cell. Additionally, voltage stress is also applied by changing the supply voltage of the circuit. Compared to ad-hoc and march tests, this test solution improves the coverage of functional HTD faults. Nevertheless, it cannot Fully (F in the table) cover this fault as it does not force an incorrect behavior. It also leads to yield loss, as this solution creates unrealistic operating conditions that will cause fault-free cells to fail; built-in post-silicon calibration can alleviate this drawback.

Weak-read test mode [22] applies voltage stress by creating a mismatch on the SA and biasing the amplification phase during a read operation. Thus, this kind of test fully covers random read functional HTD faults as it forces an incorrect behavior; other HTD faults are not targeted and therefore are not covered. This test solution does not lead to yield loss, as it uses post-silicon calibration strategies to adjust the stress applied to the SA. This strategy guarantees that only cells affected by random read HTD faults trigger an incorrect read output.

Lastly, **weak-write test mode** [20] reduces the voltage stress applied by the write driver during write operations, causing the failure of transition write operations in fault-free cells. However, cells that are in an undefined state are still flipped by the weaker write driver. Therefore, this solution Fully covers the undefined state functional HTD faults. It also partially covers random read functional HTD faults due to their random nature. A more recent version of weak-write test mode has been developed for RRAMS. [52]. Analogous to weak-read test mode, the weak-write test mode does not lead to yield loss since only cells in an undefined state are flipped and identified. No post-silicon calibration strategies have been proposed for this test solution. All three previously described solutions require additional hardware on the memory and hence require some hardware modification. Nevertheless, they all marginally modify the hardware, as they only require some extra gates or transistors on the peripherals to generate the environment-related stress.

The parametric test solutions listed in Table VIII exemplify the versatility of parametric testing. **On-chip voltage sensors** (OCVSs) [25] focus on monitoring the voltage on the BLs when applying a BT sequence of read and write operations. A neighborhood-comparison logic circuit compares the measured voltage to a dynamic reference obtained through measurements from neighboring cells. OCVSs fully cover all BL-related HTD faults (i.e., random read functional HTD and BLS parametric HTD faults) as they compare the behavior of all BLs in the array, detecting any possible deviation. This test solution leads to yield loss as it relies on measurements and comparison methodologies profoundly impacted by PV effects. Additionally, no post-silicon calibration strategies for this test solution proved to be efficient in reducing yield loss. This test solution also shows a severe HW modification as it introduces operational amplifiers and many logic gates into the memory array.

BL swing monitoring [24] focuses on monitoring the voltage on BLs during read operations and comparing the obtained value to a pre-defined reference (i.e., a threshold). Like OCVS, this test solution fully covers all HTD faults related to BLs, as any deviation that surpasses the pre-defined threshold is considered faulty. However, the solution also leads to yield loss because it does not use post-silicon calibration strategies to adjust the threshold. Furthermore, the required HW modification is significantly less than the one related to OCVS, as it only uses a couple of logic gates for monitoring and detecting possible deviations.

On-Chip Current Sensors (OCCSs) [23], alternatively, monitor the current flow by inserting a current-to-voltage converter in each memory column. An algorithm applies a read and write operation BT, while the sensors monitor the current throughout each operation. This solution fully covers PC parametric HTD faults as it compares the current flow among all columns in the array. OCCSs also partially cover random read HTD faults due to the random nature of these faults. Regarding their limitations, OCCSs present analogous limitations to OCVSs, i.e., lead to yield loss, no post-silicon calibration, and severe HW modification.

Finally, $\mathbf{I_{DDQ}}$ [48, 49] uses external measuring approaches to monitor the die current flow and compare the obtained value to a pre-defined reference. Thus, this type of test solution Fully

TABLE VIII
SRAM TEST SOLUTIONS

Test Target	Fault Observation	Fault Identification	Examples	Stressing Conditions: Algorithm-Related				Stressing Conditions: Environment-Related			HTD Faults Coverage: Func.		HTD Faults Coverage: Param.			Limitations		
				BT	AO	AD	DB	Volt.	Time	Temp.	R Read	U State	BLS	PC	SNM RNM	Yield Loss	Post-Silicon Calibration	HW Mod.
Func.	Read Op.	Expected Output	Ad-hoc Tests [47]	✓	✓	✓	✓	✓			P					No	N.A.	None
			March Tests [47]	✓	✓	✓	✓	✓			P					No	N.A.	None
			Self-timed Circuits [21]	✓	✓	✓	✓	✓	✓		P	P				Yes	Yes	Marginal
			Weak-Read Test Mode [22]	✓				✓			F					No	Yes	Marginal
			Weak-Write Test Mode [20]	✓				✓			P	F				No	No	Marginal
Param.	Voltage Mon.	Dynamic Ref.	On-Chip Voltage Sensors [25]	✓							F		F			Yes	No	Severe
		Pre-Defined Ref.	Bitline Swing Monitoring [24]	✓							F		F			Yes	No	Marginal
	Current Mon.	Dynamic Ref.	On-Chip Current Sensors [23]	✓							P			F		Yes	No	Severe
		Pre-Defined Ref.	I_{DDQ} [48, 49]	✓							P			F		No	N.A.	Marginal

detects PC parametric HTD faults and Partially detects random read HTD faults due to the reasons previously explained. This solution does not lead to yield loss as it flags faulty circuits and does not require post-silicon calibration. Furthermore, the required hardware modification depends on the I_{DDQ} methodology used. If the test solution uses built-in sensors, then the necessary hardware modification is marginal. However, if measurements are performed using external equipment only, then no modifications are required.

Based on these analyses, one can conclude that there is no single optimal test solution covering all HTD faults; each test solution has a specific target. Fig. 7 illustrates this lack of appropriate test solutions. The figure categorizes the considered test solutions among their algorithm-related stress (BT, BT+AO+AD+AB), their environmental-related stress (No Environmental Stress, Voltage, Time, Temperate), and their faults coverage of Functional HTD (R Read, U State) and Parametric HTD (BL Swin, Power Consum., SNM/RNM) faults. A green background denotes full detection; yellow background denotes partial detection, while a gray background denotes that no test solutions are using that combination of stresses to target a specific fault. Ad-hoc tests, march tests, and self-timed circuits focus on functional ETD faults; they only partially detect functional HTD faults. Weak-read test mode, BLS monitoring, and OCVSs focus on random read functional HTD faults. Additionally, the last two also focus on BLS parametric HTD faults. Weak-write test mode focuses on undefined state functional HTD faults and partially detects random read functional HTD faults. OCCSs and I_{DDQ} focus on PC parametric HTD faults but may also detect random read functional HTD faults if these eventually cause incorrect logic behavior. No test solution focuses on SNM/RNM parametric HTD faults. Considering the limitations of existing test solutions and the number of possible combinations shown in Fig. 7 that are still gray, it is safe to assume that a test solution that can efficiently detect HTD faults (i.e., with a high fault coverage and as minimal limitations as possible) is yet to be developed.

C. Test Solutions Outlook

Following the evaluation of existing test solutions' limitations (see Table VIII), and what they are missing (see Fig. 7), we identify a handful of approaches that memory test engineers can explore to improve HTD fault coverage during manufacturing tests.

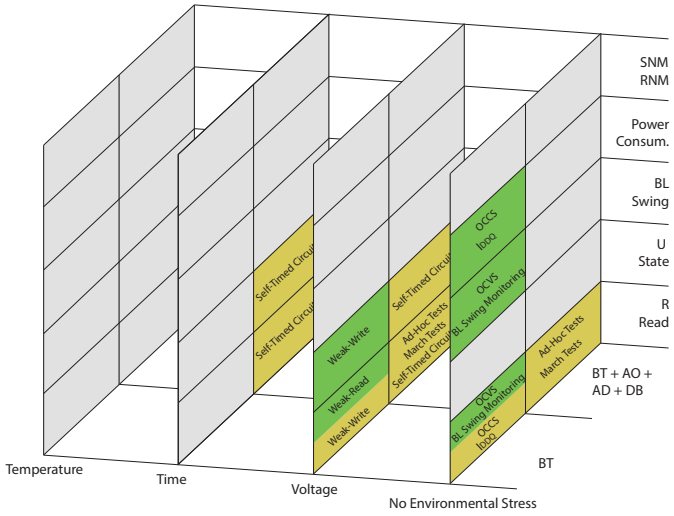


Fig. 7. Space of existing test solutions.

Parametric test solutions provide the highest HTD fault coverage: using additional hardware to monitor the memory's electrical parameters yields the best HTD fault coverage as they delve into the impact of manufacturing defects in these parameters. Hence, this type of test solution can detect defects that do not sensitize functional faults at logic level, i.e., no functional impact. Parametric test solutions should be employed in scenarios where test escapes are not tolerable (e.g., 0 defective parts per million is mandatory). Nevertheless, engineers must be aware of the costs of using parametric test solutions, such as the additional hardware and the yield loss. When employed in less critical applications, these test solutions may prove too expensive, and their use may not be justified.

Further exploration of SCs' impact: the previously discussed SCs can be further explored to improve HTD faults coverage. It is known that the industry uses voltage, time, and temperature during manufacturing tests [47] and that these SCs significantly impact the detection of functional ETD faults [11, 53] in FinFET SRAM. Additionally, it has recently been shown through electrical simulations that increasing supply voltage and temperature improves the detection of random read HTD faults [54]. Nonetheless, the authors enforce that only changing operating conditions is not enough to guarantee their detection; additional hardware applying more stress is also necessary. Unfortunately, no published works used exper-

imental data to relate SCs to HTD faults. Thus, analyses with actual data are still needed to explore further SC's impact on HTD fault detection. New types of stressing conditions might also be identified as efficient ways for testing SRAMs throughout such experiments.

Public experimental data is still lacking: the design and test of SRAMs is critical information for companies. Therefore, it is understandable that there is no publicly available data about the FinFET (SRAM) technology's most critical defects and the relation between SCs and fault coverage. If this type of data were publicly available, test engineers could develop realistic defect models and, consequently, accurate fault modeling and test solutions using appropriate SCs. Only then would it be possible to design FinFET-specific, high-quality test solutions for SRAMs.

Relation between functional and parametric HTD faults: it is straightforward to connect some functional HTD faults to other parametric HTD faults. From Table V, we see that all defects that led to a BLS parametric fault also led to a functional random read fault, either ETD or HTD. Nevertheless, the link between some faulty behaviors may be less straightforward. A reduced SNM/RNM could be an indicator of an undefined state functional HTD fault; yet, it is unknown the extent to which this statement is valid. This relation might exist for some defects, but not all. Thus, a more extensive analysis of the correlation between functional and parametric HTD faulty behavior is still necessary.

Test solutions for noise margin faults are missing: as shown in Fig. 7, no test solution guarantees the detection of the SNM/RNM parametric HTD faults. The lack of appropriate test solutions can negatively impact applications that require a high-reliability level, such as the aerospace and the automotive market [8]. SNM faults could be (Partially or Fully) detected by using test solutions that directly apply stress in every cell of the array; yet, such an approach's high costs make it unfeasible. Additionally, it might be possible to improve the detection of RNM faults by using test solutions similar to weak-write test mode (see Table VIII), in which reduced stress is applied to the cell during write operations. A more detailed analysis of these faults' characteristics is still necessary to develop efficient test solutions that target SNM/RNM parametric HTD faults.

Combining test solutions to improve overall coverage: as shown in Table VIII and Fig.7, dedicated DFT circuits have been developed targeting HTD faults; some of them use the same SCs to detect different faults, e.g., voltage is used by the DFT circuit in [22] to detect random read faults, and in the DFT circuit in [20] to detect undefined state faults. Combining somewhat similar test solutions would increase the overall HTD faults coverage. However, combining test solutions could also mean combining their limitations and overheads; ideally, these combined test solutions would maximize HTD fault coverage while minimizing overheads.

VI. Conclusion

This paper demonstrated a systematic analysis of *Hard-to-Detect* (HTD) faults and the existing test solutions for SRAMs. Based on resistive defects injection, we have shown that manufacturing defects will lead to *Easy-to-Detect* (ETD) faults and HTD faults, including functional HTD faults and parametric HTD faults. Without detection, these faults may lead to test escapes, a known cause of early in-field failures and no-trouble-found components. While ETD faults can be detected by March test solutions meeting all detection conditions, HTD faults require additional *Design-for-Testability* (DFT) circuits and environment-related stressing conditions. Analyzing the existing test solutions for (FinFET) SRAMs, it is safe to conclude that there is currently no single test solution that guarantees the detection of all HTD faults. The majority focuses on one type of fault and uses different stress conditions and extra hardware to perform the test. As the industry strives for high-quality FinFET SRAMs, new and efficient test solutions must be developed. Improved detection of HTD faults could be achieved by using parametric test solutions, proposing solutions that cover yet untargeted HTD faults, combining multiple test approaches into a single solution, and further exploring stress conditions. These new approaches would reduce test escapes and therefore improve the quality of FinFET SRAMs.

References

[1] A. Heinig *et al.*, "System integration — the bridge between more than moore and more moore," in *2014 Des., Automat., Test in Europe Conf. & Exhib. (DATE)*, Mar 2014, doi:10.7873/DATE.2014.145.

[2] D. Bhattacharya and N. K. Jha, "FinFETs: From Devices to Architectures," *Advances in Electronics*, vol. 2014, Sep 2014, doi:10.1155/2014/365689.

[3] Y. Liu and Q. Xu, "On modeling faults in FinFET logic circuits," in *2012 IEEE Int. Test Conf. (ITC)*, Nov 2012, doi:10.1109/TEST.2012.6401565.

[4] S. Hamdioui, *Testing Static Random Access Memories: Defects, Fault Models and Test Patterns*, ser. Frontiers in Electronic Testing. Boston, MA: Springer US, 2004, vol. 26, doi:10.1007/978-1-4757-6706-3.

[5] S. Davidson, "Towards an understanding of no trouble found devices," in *23rd IEEE VLSI Test Symp. (VTS)*, May 2005, pp. 147–152, doi:10.1109/VTS.2005.86.

[6] C. Acero *et al.*, "Embedded deterministic test points for compact cell-aware tests," in *2015 IEEE Int. Test Conf. (ITC)*, Oct 2015, doi:10.1109/TEST.2015.7342383.

[7] H. Tang, A. Jain, S. K. Pillai, D. Joshi, and S. Rao, "Using cell aware diagnostic patterns to improve diagnosis resolution for cell internal defects," in *2017 IEEE 26th Asian Test Symp. (ATS)*, Nov 2017, pp. 231–236, doi:10.1109/ATS.2017.51.

[8] M. Shah, S. Ghosh, and S. Martin, "A quality and reliability driven DFT and DFR strategy for automotive and industrial markets," in *2019 IEEE 37th VLSI Test Symp. (VTS)*, Apr 2019, doi:10.1109/VTS.2019.8758640.

[9] A. J. van de Goor and Z. Al-Ars, "Functional memory faults: a formal notation and a taxonomy," in *Proc. 18th IEEE VLSI Test Symp. (VTS)*, Apr 2000, pp. 281–289, doi:10.1109/VTEST.2000.843856.

[10] L. Dilillo, P. Girard, S. Pravossoudovitch, A. Virazel, S. Borri, and M. Hage-Hassan, "Resistive-open defects in embedded-SRAM core cells: Analysis and march test solution," in *13th Asian Test Symp. (ATS)*, Nov 2004, pp. 266–271, doi:10.1109/ATS.2004.75.

[11] G. Harutyunyan, G. Tshagharyan, and Y. Zorian, "Test and repair methodology for FinFET-based memories," *IEEE Tran. on Device and Mat. Reliability*, vol. 15, no. 1, pp. 3–9, Mar 2015, doi:10.1109/TDMR.2015.2397032.

[12] T. S. Copetti, T. R. Balen, G. C. Medeiros, and L. M. B. Poehls, "Analyzing the behavior of FinFET SRAMs with resistive defects," in *2017 IFIP/IEEE Int. Conf. on Very Large Scale Integration (VLSI-SoC)*, Oct 2017, doi:10.1109/VLSI-SoC.2017.8203483.

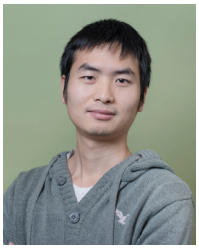
[13] G. Medeiros, E. Brum, L. B. Poehls, T. Copetti, and T. Balen, "Influence of temperature on dynamic fault behavior due to resistive defects in FinFET-based SRAMs," in *2018 IEEE 19th Latin-Amer. Test Symp. (LATS)*, Mar 2018, doi:10.1109/LATW.2018.8349697.

[14] T. S. Copetti, G. C. Medeiros, L. M. Poehls, and T. R. Balen, "Evaluating the impact of resistive defects on FinFET-based SRAMs," in *IFIP*

Advances in Inf. and Commun. Technol., vol. 500. Springer New York LLC, May 2019, pp. 22–45, doi:10.1007/978-3-030-15663-3_2.
[15] S. Hamdioui, Z. Al-Ars, and A. van de Goor, "Testing static and dynamic faults in random access memories," in *Proc. 20th IEEE VLSI Test Symp. (VTS)*, Apr 2002, pp. 395–400, doi:10.1109/VTS.2002.1011170.
[16] A. Benso, A. Bosio, S. Di Carlo, G. Di Natale, and P. Prinetto, "March AB, march AB1: new march tests for unlinked dynamic memory faults," in *IEEE Int. Conf. on Test*, Oct 2005, pp. 834–841, doi:10.1109/TEST.2005.1584047.
[17] S. Borri, M. Hage-Hassan, L. Dilillo, P. Girard, S. Pravossoudovitch, and A. Virazel, "Analysis of dynamic faults in embedded-SRAMs: implications for memory test," *Journal of Electron. Testing*, vol. 21, no. 2, pp. 169–179, Apr 2005, doi:10.1007/s10836-005-6146-1.
[18] G. Harutyunyan, S. Martirosyan, S. Shoukourian, and Y. Zorian, "Memory physical aware multi-level fault diagnosis flow," *IEEE Trans. on Emerg. Topics on Comput.*, Jan 2018, doi:10.1109/TETC.2018.2789818.
[19] M. Fieback *et al.*, "Device-aware test: a new test approach towards DPPB level," in *2019 IEEE Int. Test Conf. (ITC)*, Nov 2019, doi:10.1109/ITC44170.2019.9000134.
[20] A. Meixner and J. Banik, "Weak write test mode: an SRAM cell stability design for test technique," in *Proc. Int. Test Conf. (ITC)*, Nov 1997, pp. 1043–1052, doi:10.1109/TEST.1997.639732.
[21] J. Kinseher, L. B. Zordan, I. Polian, and A. Leininger, "Improving SRAM test quality by leveraging self-timed circuits," in *2016 Des., Automat., Test in Europe Conf. & Exhib. (DATE)*, Mar 2016, pp. 984–989, doi:10.3850/9783981537079_0840.
[22] G. C. Medeiros *et al.*, "A DFT scheme to improve coverage of hard-to-detect faults in FinFET SRAMs," in *2020 Des., Automat., Test in Europe Conf. & Exhib. (DATE)*, Mar 2020, pp. 792–797, doi:10.23919/DATE48585.2020.9116278.
[23] G. Medeiros, L. Bolzani Poehls, M. Taouil, F. Luis Vargas, and S. Hamdioui, "A defect-oriented test approach using on-chip current sensors for resistive defects in FinFET SRAMs," *Microelectronics Reliability*, vol. 88-90, pp. 355–359, Sep 2018, doi:10.1016/j.microrel.2018.07.092.
[24] M.-C. Chen, T.-H. Wu, and C.-W. Wu, "A built-in self-test scheme for detecting defects in FinFET-based SRAM circuit," in *2018 IEEE 27th Asian Test Symp. (ATS)*, Oct 2018, pp. 19–24, doi:10.1109/ATS.2018.00015.
[25] G. C. Medeiros, M. Taouil, M. Fieback, L. B. Poehls, and S. Hamdioui, "DFT scheme for hard-to-detect faults in FinFET SRAMs," in *2019 IEEE Eur. Test Symp. (ETS)*, May 2019, doi:10.1109/ETS.2019.8791517.
[26] A. Pavlov and M. Sachdev, *CMOS SRAM Circuit Design and Parametric Test in Nano-Scaled Technologies*. Springer Netherlands, 2008, vol. 40, doi:10.1007/978-1-4020-8363-1.
[27] T. Matsukawa *et al.*, "Comprehensive analysis of variability sources of FinFET characteristics," in *2009 Symp. on VLSI Technol.*, Jun 2009, pp. 118–119.
[28] M. O. Simsir, A. Bhoj, and N. K. Jha, "Fault modeling for FinFET circuits," in *2010 IEEE/ACM Int. Symp. on Nanoscale Architectures*, Jun 2010, pp. 41–46, doi:10.1109/NANOARCH.2010.5510927.
[29] A. van de Goor, *Testing Semiconductor Memories: Theory and Practice*. J. Wiley & Sons, 1991.
[30] L. Wu, S. Rao, M. Taouil, E. J. Marinissen, G. S. Kar, and S. Hamdioui, "Characterization, Modeling and Test of Synthetic Anti-Ferromagnet Flip Defect in STT-MRAMs," in *2020 IEEE International Test Conference (ITC)*, 2020, pp. 1–10, doi:10.1109/ITC44778.2020.9325258.
[31] S. Hamdioui and A. J. Van De Goor, "An experimental analysis of spot defects in SRAMs: realistic fault models and tests," in *Proc. of the Ninth Asian Test Symp. (ATS)*, Dec 2000, pp. 131–138, doi:10.1109/ATS.2000.893615.
[32] L. Dilillo, P. Girard, S. Pravossoudovitch, A. Virazel, and M. B. Hage-Hassan, "Data retention fault in SRAM memories: analysis and detection procedures," in *23rd IEEE VLSI Test Symposium (VTS'05)*, 2005, pp. 183–188, doi:10.1109/VTS.2005.37.
[33] S. Sunter, "Analog fault simulation - a hot topic!" in *2020 IEEE Eur. Test Symp. (ETS)*, May 2020, doi:10.1109/ETS48528.2020.9131581.
[34] L. Dilillo, P. Girard, S. Pravossoudovitch, A. Virazel, and M. Bastian, "Resistive-open defect injection in SRAM core-cell," in *Proc. 42nd Design Automat. Conf. (DAC)*. NY, USA: ACM Press, June 2005, p. 857–862, doi:10.1145/1065579.1065804.
[35] R. A. Fonseca *et al.*, "Analysis of resistive-bridging defects in SRAM core-cells: A comparative study from 90nm down to 40nm technology nodes," in *2010 15th IEEE Eur. Test Symp. (ETS)*, May 2010, pp. 132–137, doi:10.1109/ETSYM.2010.5512768.
[36] E. I. Vatajelu *et al.*, "Analyzing resistive-open defects in SRAM core-cell under the effect of process variability," in *2013 18th IEEE Eur. Test Symp. (ETS)*, May 2013, doi:10.1109/ETS.2013.6569373.
[37] G. C. Medeiros, L. B. Poehls, and F. F. Vargas, "Analyzing the impact of SEUs on SRAMs with resistive-bridge defects," in *2016 29th Int. Conf. on VLSI Design (VLSID)*, Jan 2016, pp. 487–492, doi:10.1109/VLSID.2016.146.
[38] S. Hamdioui, A. J. van de Goor, and M. Rodgers, "March SS: a test for all static simple RAM faults," in *Proc. 2002 IEEE Int. Workshop on Memory Technol., Design and Testing (MTDT)*, Jul 2002, pp. 95–100, doi:10.1109/MTDT.2002.1029769.
[39] L. Dilillo, P. Girard, S. Pravossoudovitch, A. Virazel, S. Borri, and M. Hage-Hassan, "Dynamic read destructive fault in embedded-SRAMs: analysis and march test solution," in *Proc. 9th IEEE Eur. Test Symp. (ETS)*, May 2004, pp. 140–145, doi:10.1109/ETSYM.2004.1347645.
[40] G. Tshagharyan *et al.*, "Modeling and Testing of Aging Faults in FinFET Memories for Automotive Applications," in *2018 IEEE International Test Conference (ITC)*, 2018, pp. 1–10, doi:10.1109/TEST.2018.8624890.
[41] D. Kraak, M. Taouil, S. Hamdioui, P. Weckx, S. Cosemans, and F. Catthoor, "eSRAM Reliability: Why is it still not optimally solved?" in *2020 15th Design Technology of Integrated Systems in Nanoscale Era (DTIS)*, 2020, pp. 1–6, doi:10.1109/DTIS48698.2020.9081145.
[42] Nanoscale Integration and Modeling (NIMO), "Predictive technology model (PTM)," 2012. [Online]. Available: http://ptm.asu.edu/
[43] I. Agbo *et al.*, "Integral Impact of BTI, PVT Variation, and Workload on SRAM Sense Amplifier," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, no. 4, pp. 1444–1454, 2017, doi:10.1109/TVLSI.2016.2643618.
[44] S. S. Sapatnekar, "Overcoming Variations in Nanometer-Scale Technologies," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 1, no. 1, pp. 5–18, 2011, doi:10.1109/JETCAS.2011.2138250.
[45] A. J. van de Goor, S. Hamdioui, and R. Wadsworth, "Detecting faults in the peripheral circuits and an evaluation of SRAM tests," in *2004 Int. Conf. on Test*, Oct 2004, pp. 114–123, doi:10.1109/TEST.2004.1386943.
[46] S. Hamdioui, Z. Al-Ars, G. N. Gaydadjiev, and A. J. van de Goor, "An investigation on capacitive coupling in RAM address decoders," in *2007 2nd Int. Design and Test Workshop*, Dec 2007, pp. 9–14, doi:10.1109/IDT.2007.4437418.
[47] S. Hamdioui, A. J. van de Goor, J. D. Reyes, and M. Rodgers, "Memory test experiment: industrial results and data," *IEE Proc. - Comput. and Digital Techn.*, vol. 153, no. 1, Jan 2006, doi:10.1049/ip-cdt:20050104.
[48] H. Balachandran and D. M. H. Walker, "Improvement of SRAM-based failure analysis using calibrated Iddq testing," in *Proc. of 14th VLSI Test Symp. (VTS)*, Apr 1996, pp. 130–136, doi:10.1109/VTEST.1996.510847.
[49] M. Hashizume, T. Tamesada, T. Koyama, and A. J. van de Goor, "CMOS SRAM functional test with quiescent write supply current," in *Proc. 1998 IEEE Int. Workshop on IDDQ Testing*, Nov 1998, pp. 4–8, doi:10.1109/IDDQ.1998.730724.
[50] M. A. Breuer and A. D. Friedman, *Diagnosis & reliable design of digital syst.*, ser. Digital syst. design series. Computer Science Press, 1976.
[51] M. S. Abadir and H. K. Reghbati, "Functional testing of semiconductor random access memories," *ACM Comput. Surv.*, vol. 15, pp. 175–198, Sep 1983, doi:10.1145/356914.356916.
[52] S. Hamdioui, M. Taouil, and N. Z. Haron, "Testing Open Defects in Memristor-Based Memories," *IEEE Transactions on Computers*, vol. 64, no. 1, pp. 247–259, 2015, doi:10.1109/TC.2013.206.
[53] T. Santos Copetti, T. Balen, E. Brum, C. Aquistapace, and L. Poehls, "Comparing the impact of power supply voltage on CMOS- and FinFET-based SRAMs in the presence of resistive defects," *Journal of Electron. Testing*, May 2020, doi:10.1007/s10836-020-05869-2.
[54] G. C. Medeiros, M. Fieback, M. Taouil, L. B. Poehls, and S. Hamdioui, "Detecting random read faults to reduce test escapes in FinFET SRAMs," in *2021 IEEE Eur. Test Symp. (ETS)*, May 2021 (in press).

Guilherme Cardoso Medeiros (Student Member, IEEE) received his B.Sc. degree from the Pontifical Catholic University of Rio Grande do Sul (PUCRS), Porto Alegre, Brazil, in 2015. He received his M.Sc. degree from the same university in 2017. He is currently a Ph.D. candidate at Delft University of Technology, the Netherlands. His main areas of interest are test strategies for SRAMs, defect and fault modelling for FinFET devices, and emerging memory technologies.

Moritz Fieback (Student Member, IEEE) received his B.Sc. and M.Sc. in 2015 and 2017, respectively, from Delft University of Technology, the Netherlands. Currently, he is a Ph.D. candidate in the same university. His research interests include device modeling, test and reliability of emerging memories.



Lizhou Wu (Student Member, IEEE) received his B.Sc. degree in electronic science and engineering from Nanjing University, China, in 2013, and M.Sc. degree in computer science and engineering from National University of Defense Technology, China, in 2016. Currently, he is pursuing a Ph.D. degree at the Computer Engineering Laboratory, Delft University of Technology, the Netherlands. His research focuses on MTJ characterization and modeling, STT-MRAM test and reliability.

Mottaqiallah Taouil (Member, IEEE) received his M.Sc. and Ph.D. degrees (both with honors) in computer engineering from Delft University of Technology, the Netherlands. He is currently an Assistant Professor at the Computer Engineering Laboratory, Delft University of Technology. His current research interests include hardware security, embedded systems, 3D stacked integrated circuits, VLSI design and test, built-in-self-test, design for testability, yield analysis, and memory test structures.

Leticia Maria Bolzani Poehls (Member, IEEE) received her M.Sc. in Electrical Engineering from the Pontifical Catholic University of Rio Grande do Sul (PUCRS), Brazil, in 2004 and her Ph.D. in Computer Engineering from the Politecnico di Torino, Italy, in 2004. She is an Associate Professor at the School of Engineering of PUCRS. Currently, she is a guest researcher at the Chair of Integrated Digital Systems and Circuit Design - RWTH Aachen University, Germany, working on developing test strategies for Resistive RAMs (RRAMs). Her fields of interest include Testing and Fault Tolerance of Integrated Systems and Emerging Technologies, Power-, Aging- and Temperature-Aware IC Design, and Development of EDA tools for IC optimization. She is a technical committee member in several IEEE-sponsored conferences, is a member of the Latin American Test Symposium (LATS) steering committee. Finally, she is a coordinating editor of the Journal of Electronic Testing: Theory and Application (JETTA).

Said Hamdioui (Senior Member, IEEE) is currently a chair professor and the head of the Quantum & Computer Engineering Department at TU Delft. He received the MSEE and PhD degrees (both with honors) from TU Delft. Prior to joining TU Delft as a professor, Hamdioui spent about seven years in the industry including Intel Corporation (California, USA), Philips Semiconductors R&D (Crolles, France), Philips/NXP Semiconductors (Nijmegen, The Netherlands). His research focuses on two domains: Dependable CMOS nano-computing (including Reliability, Testability, Hardware Security) and emerging technologies and computing paradigms (including 3D stacked ICs, in-memory-computing).

Hamdioui owns two patents, has published one book and contributed to two others, and has co-authored over 180 conference and journal papers. He delivered dozens of keynote speeches, distinguished lectures, and invited presentations and tutorials at major international forums/conferences/schools and at leading semiconductor companies. Hamdioui is an Associate Editor of IEEE Transactions on VLSI Systems (TVLSI), and he serves on the editorial board of IEEE Design & Test, Elsevier Microelectronic Reliability Journal, and of the Journal of Electronic Testing: Theory and Applications (JETTA).