A DFT Scheme to Improve Coverage of Hard-to-Detect Faults in FinFET SRAMs

Medeiros, Guilherme Cardoso; Cem Gursoy, Cemil; Wu, Lizhou; Fieback, Moritz; Jenihhin, Maksim; Taouil, Mottaqiallah; Hamdioui, Said

DOI
10.23919/DATE48585.2020.9116278

Publication date
2020

Document Version
Accepted author manuscript

Published in
Proceedings of the 2020 Design, Automation and Test in Europe Conference and Exhibition, DATE 2020

A DFT Scheme to Improve Coverage of Hard-to-Detect Faults in FinFET SRAMs

Guilherme Cardoso Medeiros[1] Cemil Cem Gürsoy[2] Lizhou Wu[1] Moritz Fieback[1]
Maksim Jenihhin[2] Mottaqiallah Taouil[1] Said Hamdioui[1]
[1]Computer Engineering Laboratory, Delft University of Technology, The Netherlands
[2]Department of Computer Systems, Tallinn University of Technology, Estonia
Email: {G.CardosoMedeiros, M.C.R.Fieback, Lizhou.Wu, M.Taouil, S.Hamdioui}@tudelft.nl
{Cemil.Gursoy, Maksim.Jenihhin}@taltech.ee

***Abstract*—Manufacturing defects can cause faults in FinFET SRAMs. Of them, easy-to-detect (ETD) faults always cause incorrect behavior, and therefore are easily detected by applying sequences of write and read operations. However, hard-to-detect (HTD) faults may not cause incorrect behavior, only parametric deviations. Detection of these faults is of major importance as they may lead to test escapes. This paper proposes a new design-for-testability (DFT) scheme for FinFET SRAMs to detect such faults by creating a mismatch in the sense amplifier (SA). This mismatch, combined with the defect in the cell, will incorrectly bias the SA and cause incorrect read outputs. Furthermore, post-silicon calibration schemes can be used to avoid over-testing or test escapes caused by process variation effects. Compared to the state of the art, this scheme introduces negligible overheads in area and test time while it significantly improves fault coverage and reduces the number of test escapes.**

***Index Terms*—FinFET, SRAM, Hard-to-Detect Faults, Defects, DFT, Memory Testing**

I. Introduction

FinFET technology has been used by the semiconductor industry to continue the downscaling of *integrated circuits* (ICs) [1]. Despite their benefits, FinFETs also present challenges; one of them is testing FinFET-based circuits such as FinFET SRAMs. Due to their complex structure, FinFETs may be affected by small manufacturing defects [2]. Although these defects may not impact functionality, they can cause parametric deviations such as increased leakage current and small delays [3]. In FinFET SRAMs, these defects cause *hard-to-detect* (HTD) faults such as random and parametric faults [4]. HTD faults do not always lead to incorrect behavior; hence, test schemes that rely on fault observation (e.g., March tests) are not suitable to detect them [5]. Using only these test schemes will lead to test escapes – which are a known cause of early in-field failures and no trouble founds [6]. In order to avoid these scenarios and achieve the required quality and reliability of FinFET devices [7, 8], new high-quality test methodologies are necessary [9].

Different *design-for-testability* (DFT) methodologies to detect defects and HTD faults have been proposed. The use of additional hardware to trigger cell stability faults was proposed in [10, 11]; random and parametric faults were not discussed. A more recent DFT scheme proposed to change the memory timing scheme to improve detection of defects [12]. However, changing the timing scheme of the memory may introduce inaccurate and unrealistic stress in the memory. Therefore, the use of this scheme may lead to over-testing and consequently yield loss. Another DFT approach consists of using monitoring hardware (on-chip sensors and comparison circuits) to perform parametric testing on memory parameters such as current flow [13] and bitline swing [14, 15]. However, detection of HTD faults using monitoring hardware can become complex and inaccurate due to PV effects [16]. Furthermore, the additional hardware introduces a large area overhead, and can also negatively impact memory performance.

This paper advances the state of the art by presenting a simple yet effective DFT scheme to detect HTD faults in FinFET SRAMs. The proposed scheme unbalances the *sense amplifier* (SA) aiming to trigger incorrect read operations when an HTD fault is present. HTD faults in the cell causing parametric deviations will be further stressed and will cause read operations to output incorrect data, enabling their detection. This scheme can be used alongside traditional March test sequences to significantly improve the coverage of HTD faults with little overhead in area and test time. Furthermore, calibration schemes can be easily integrated to overcome PV effects. The main contributions of this paper are as follows:

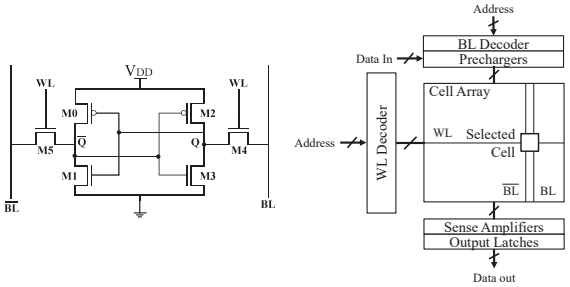
- A new low-cost DFT scheme to detect HTD faults in FinFET SRAMs.
- Validation of the scheme by injecting defects and assessing its detection capabilities.
- Analysis of the impact of PV on the DFT and a post-silicon calibration strategy to counter these variations.

This paper is organized as follows. Section II explains FinFET SRAMs. Section III explains HTD faults. Section IV presents the proposed scheme, while Section V validates it. Section VI introduces a post-silicon calibration scheme. Finally, Section VII discusses and concludes the paper.

II. FinFET SRAM

Static Random-Access Memories (SRAMs) are volatile memories consisting of a cell array and peripheral circuitry [17]. A 6T SRAM cell (Fig. 1a) is designed using six transistors; M0, M1, M2 and M3 form two cross-coupled inverters, while M4 and M5 are used as pass gates. The *wordline* (WL)

This work is supported by European Union's Horizon 2020 Research and Innovation through RESCUE project under grant 722325.



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

Fig. 1. Memory model.

controls M4 and M5, which are connected to the *bitlines* (BL and $\overline{BL}$). The digital value stored in the cell corresponds to the voltage on Q ('1' for V_{DD}, '0' for 0 V).

Additional components (decoders, sense amplifiers (SAs), write drivers, prechargers, I/O circuitry) are used to provide write and read capabilities to the SRAM. This organization is depicted in Fig. 1b. An operation can be performed by selecting a row address through the WL and a column address through a pair of BLs. During write operations, the write driver either charges or discharges the BL based on the value to be written, while $\overline{BL}$ is set to the complementary value. The WL is then activated to force the new value on the cell. During read operations, both BLs will be first pre-charged to V_{DD}. The WL is activated so that the cell can discharge one of the BLs based on the stored value. Subsequently, the SA is triggered by an SA Enable (SAE) signal to convert the voltage difference on BL and $\overline{BL}$ (i.e., the BL swing) to '0' or '1'.

The memory used in this work is designed using *Fin Field-Effect Transistors* (FinFETs). These devices are multi-gate transistors consisting of vertical stripes of silicon fins covered by a gate structure [18]. The dimensions of these fins, such as fin height (H_{FIN}) and fin thickness (T_{FIN}), define the driving strength. During manufacturing process, inaccurate processing may affect these dimensions [19]. Variations in H_{FIN} and T_{FIN} may weaken the device; small shorts and fin cuts can lead to impaired FinFETs [20]. Although these devices may not cause incorrect behavior, they will lead to small delays or increased leakage current [3]. Furthermore, such devices may also cause reliability problems (e.g., shorter lifetime, higher failure rate) once used in field [21]. Therefore, detection of these devices is of great importance to assure high-quality and reliable FinFET devices.

III. Hard-to-Detect Faults

In the context of memories, manufacturing defects may cause strong faults and weak faults [4], as shown in Fig. 2:

Strong Faults are faults that can cause incorrect logic behavior. Such faults can be expressed by *fault primitives* (FPs) using the standard <*S*/*F*/*R*> notation [22], where S denotes the sensitizing sequence, F the fault effect on the cell, and R the read value in the case of read operations. Some strong faults can have a **random** behavior [4]; for example, the cell switches to an undefined state or the BL swing at the SA's input is smaller than the required value. Other strong faults are **deterministic** and thus easy to detect.

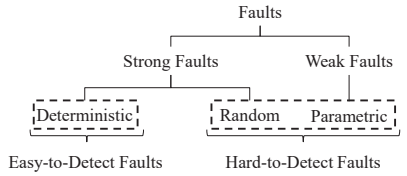


Fig. 2. Classification of memory faults.

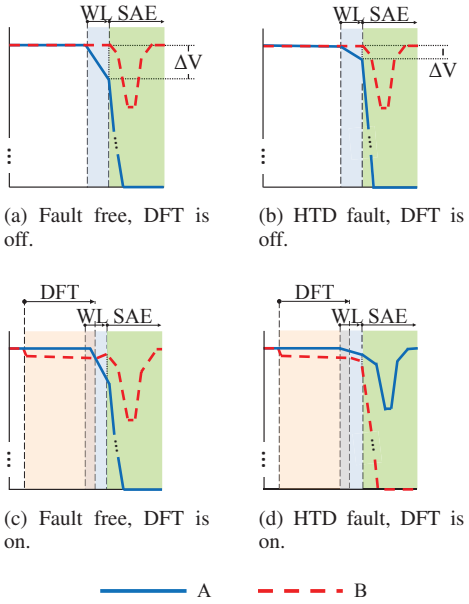
Weak Faults are faults that do not lead to incorrect logic behavior; they cause **parametric** faults. For example, a defect could create a small voltage disturbance on a node of the cell or reduce the BL swing during read operations. Although this defect does not cause any incorrect behavior, it decreases the robustness of the cell.

Different testing schemes are used to detect different faults. Test approaches that rely on fault observation (e.g., March tests) can easily detect strong deterministic faults as they always lead to incorrect behavior; thus, they are also named ***easy-to-detect*** (ETD) faults. This is not the case for random faults as they may not lead to incorrect behavior [4]. Therefore, read operations cannot guarantee their detection. Such faults can be compared to faults caused by marginal defects, which are defects that only cause faults at specific temperatures and voltage conditions [2]; it is widely known that the occurrence of these defects is increasing as the feature sizes shrink down [23]. Since random faults are impacted by random effects such as *process variation* (PV), it is statistically expected that only part of them will be detected by March tests due to incorrect logic behavior; the remaining faults that do not cause incorrect behavior will lead to test escapes.

Detection of weak faults is even more complex as they never lead to incorrect behavior, only to parametric faults, i.e., variations in the performance and in parameters such as BL swing or leakage current. From a logic point of view, parametric faults are undetectable; in the presence of a parametric fault, all operations pass correctly. Both random and parametric faults are considered ***hard-to-detect*** (HTD) faults as they do not necessarily imply incorrect behavior. Consequently, their detection is not guaranteed by approaches that use fault observation. Nevertheless, detection of such faults is of high importance as they may lead to test escapes and cause reliability problems (e.g., shorter lifetime, higher in-field failure rate) [21]. One approach to detect HTD faults is to introduce *design-for-testability* (DFT) circuitry into the memory to perform a special test (e.g., high-stress testing, parametric testing, etc.). In this work, we propose to use a novel, simple and low-cost DFT scheme to detect these faults. The proposed scheme accomplishes this by applying additional stress to HTD faults, thus forcing them to become easily detectable by read operations.

IV. DFT Scheme

This section presents the proposed DFT scheme. We first present its concept, followed by the hardware implementation.



(a) Fault free, DFT is off.

(b) HTD fault, DFT is off.

(c) Fault free, DFT is on.

(d) HTD fault, DFT is on.

Fig. 3. Concept of the proposed DFT scheme.

A. Concept

The DFT technique proposed in this work targets the detection of single-cell HTD faults by observing incorrect values during read operations. To achieve this, additional transistors are used to create a mismatch in the SA and unbalance it. Cells affected by HTD faults will fail to overcome the DFT mismatch and will output incorrect values. For example, these HTD faults may cause *Incorrect Read Faults* (IRFs) [22], which are described as $<1r1/1/0>$ or $<0r0/0/1>$. Since IRFs are ETD faults, they can be easily detected by applying traditional March sequences. Therefore, such HTD faults can be detected using simple test algorithms and fault observation.

Fig. 3 illustrates the concept behind the proposed DFT. It shows mock waveforms of the internal nodes of the SA (A and B) during *read* '0' ($r0$) operations. In order to guarantee a '0' as read output, the voltage difference $\Delta V = V_B - V_A$ (i.e., the BL swing) must be bigger than a predefined threshold; smaller BL swings may lead to random read outputs and consequently test escapes. We first discuss standard fault-free read operations (Fig. 3a) followed by faulty read operations(Fig. 3b). Before WL is activated, both nodes of the SA are precharged to V_{DD}. Activating the WL connects the cell to the BLs, which in turn triggers the discharge of node A. In the fault-free operation, the cell has no obstacles discharging node A. This leads to a BL swing that is big enough to overcome possible variations (PV, temperature, voltage fluctuations) and guarantee a correct output. Once the WL is deactivated and the SA is activated (i.e., *SAE* is asserted), the amplification phase starts: node B is charged to V_{DD}, while A is discharged to GND. However, if the cell is affected by a small defect, this discharge may be slower than expected. In HTD faults, this results in a BL swing that is not enough according to the specification, but large enough to work functionally correct (i.e., a parametric fault) or sometimes correct (i.e., a random fault). Due to its random nature, it is expected that only a part of HTD faults will cause incorrect behavior; the remaining part will lead to test escapes. Hence, the proposed DFT scheme targets to further stress these read operations in which the BL swing is too small and force an incorrect behavior, thus enabling the detection of HTD faults.

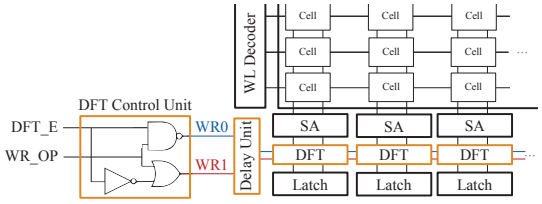


Fig. 4. Memory organization with the DFT.

Fig. 3c and 3d illustrate the same fault-free and faulty read operations, but now using the proposed DFT scheme.This circuit is activated during special read operations; we name these operations *weak read '0'* ($\tilde{r}0$) and *weak read '1'* ($\tilde{r}1$). During these operations, additional transistors are used to introduce a mismatch in the SA to unbalance it. This takes place before the WL is open; while A is kept at V_{DD}, B is slightly discharged. When WL is activated, the cell must overcome the mismatch introduced by the DFT in order to discharge A. In fault-free read operations, the cell can easily overcome this mismatch by developing enough BL swing (Fig. 3c). Once *SAE* is asserted, the SA amplifies the correct value. However, this is not the case for HTD faults. Due to defects, defective cells are not able to overcome the mismatch introduced by the DFT circuit. This leads to the SA being incorrectly biased once it is activated and consequently to an incorrect read output that can easily be detected.

B. Implementation

Fig. 4 illustrates the modified memory organization. The DFT control unit generates the signals to control the DFT. This unit has two inputs: a 1-bit signal (*DFT_E*) to enable the DFT, and a 1-bit signal (*WR_OP*) to distinguish between $\tilde{r}0$ and $\tilde{r}1$ operations, which are read operations with the DFT enabled. It generates signals *WR0* and *WR1* that are used to control the DFT (i.e., the SA mismatch) during read '0' and read '1' operations respectively. A delay unit controls the timing of these signals; this will be further discussed in Section VI.

The SA and DFT circuits are depicted in Fig. 5. The DFT circuit consists of 6 transistors divided into two symmetric groups connected to the internal nodes of the SA. During $\tilde{r}0$ operations, transistors labeled with "_R0" are turned on, while "_R1" transistors are kept off. The opposite applies for $\tilde{r}1$ operations. M0s and M1s are 8-fin and 3-fin PFETS, respectively, while M2s are 1-fin NFETs. During read operations, transistor M0 and M2 are simultaneously activated to discharge one node, while the counterpart transistor M1 is activated to prevent the discharge of the opposite node.

These transistors are sized to trigger incorrect read outputs for HTD faults, i.e., when the BL swing is smaller than a predefined threshold. We define this limit by performing PV

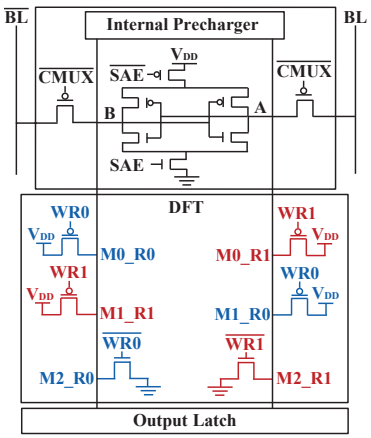


Fig. 5. The proposed DFT scheme.

Monte Carlo simulations (20,000 simulations), measuring the mismatch between cross-coupled transistors, and calculating the 6σ variation to obtain a failure rate of 10^{-9}. For the SA circuit previously described, a 6σ variation of 36 mV was observed. This is the minimum input offset required to guarantee that only 1 out of 10^9 devices fail due to process variation. As a consequence, any read operation with an absolute BL swing of 36 mV or less is considered a random fault. Furthermore, we define an additional margin of around 10% of the 6σ variation for parametric faults; this value is established based on simulations where we varied temperature and supply voltage. Read operations with an absolute BL swing between 36 mV and 40 mV are considered parametric faults. Thus, read operations in which the BL swing is smaller than 40 mV are considered HTD faults (i.e., either random or parametric) and are consequently targeted by the proposed DFT.

V. Validation

In this section, we validate the proposed DFT scheme. First, we describe the experimental setup, thereafter, the performed experiments and finally, the simulation results.

A. Experimental Setup

The memory netlist is described in SPICE using the PTM 14 nm FinFET library [24]. The memory array is composed of 128 rows and 64 columns where each logical word contains 32 bits. Hence, write drivers, SAs, and precharger circuits are shared among two neighboring columns. Furthermore, a timing circuit is used to generate intermediate control signals.

B. Performed Experiments

Experiments were carried out by SPICE simulations. 16 single-cell resistive defects [4] have been injected in the cell, one at a time as shown in Fig. 6. They are either resistive-opens (RO), resistive-short (RS), or resistive-bridge (RB) defects. RO defects are unintended series resistances within an existing connection. They are labeled as *Open Connections* (OC) 01 to 11. RSs and RBs are unintended resistive connections between two nodes. RSs are shorts to power nodes (V_{DD} or GND); they are named as *Short Connections* (SC) 01 and 02. In

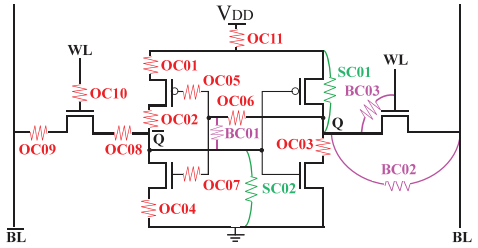
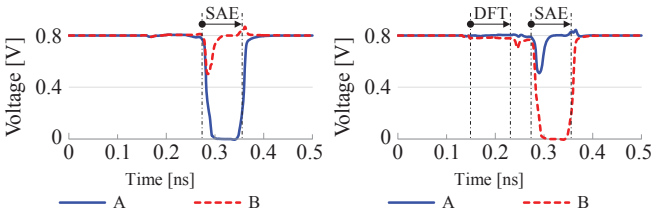


Fig. 6. Set of injected resistive defects.



(a) DFT is off. (b) DFT is on.

Fig. 7. Internal nodes of the SA during read operations.

contrast, RBs are shorts between any two other nodes of the cell. They are identified as *Bridges Connections* (BC) 01 to 03.

A simple March element of a write operation followed by a weak read operation is used to sensitize HTD faults. This element is applied to the entire memory array; subsequently, it is repeated with complimentary operations. This sequence can be described by $\Updownarrow(w1,\tilde{r}1)$; $\Updownarrow(w0,\tilde{r}0)$. Note that weak read operations should be performed directly after a write operation to the cell to maximize the detection capability. The reason for this is that writing to a defective cells requires time to stabilize and any deviations (e.g., voltage drops in the true nodes of cells) increase the detection.

C. Simulation Results

We first analyze a study case of a cell affected by a defect (OC08, 9 kΩ) that causes an HTD fault. During a standard $r0$ operation (Fig. 7a), the DFT is off and the SA is balanced. Due to the defect in the cell, a BL swing of only 34 mV is observed. When the SA is enabled, node B is charged to V_{DD} while A is discharged to GND – which represents a '0' as output. However, this small BL swing results actually in a random fault as a small noise might affect the read value. Hence, if this HTD fault is not detected, it may lead to test escapes. With the DFT enabled, the $\tilde{r}0$ operation (see Fig. 7b) successfully detects the fault, as an incorrect logic value is read.

We further analyze the detection capability of the proposed DFT by injecting all the previously discussed resistive defects, and sweeping their size. The results of these experiments are shown in Fig. 8. The figure shows for each defect its faulty behaviour (i.e., fault free, parametric fault, random fault, or ETD faults) based on defect size. The first defect size that the DFT can detect is indicated by the vertical yellow line.

From the 16 resistive defects injected, 8 caused both HTD and ETD faults, 2 caused only HTD faults, while the remaining

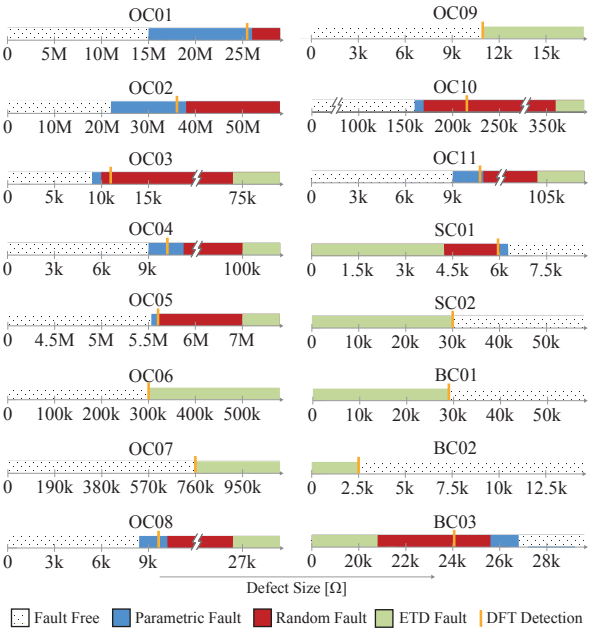
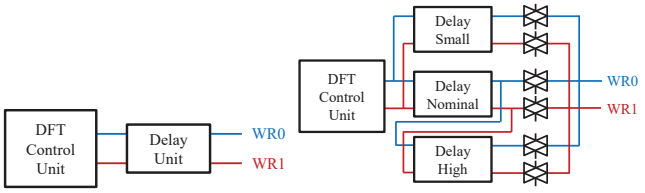


Fig. 8. Detection capability of the proposed approach.

6 only caused ETD faults. Overall, the proposed DFT was able to detect incorrect behavior in the random and parametric fault ranges. Note that none of the fault-free ranges have been identified as faulty. Moreover, only in three cases (OC03, OC10, and BC03) the DFT was not able to trigger incorrect behavior in the parametric fault range. Furthermore, in cases where only ETD faults have been sensitized, the DFT has the same detection capability as traditional March tests.

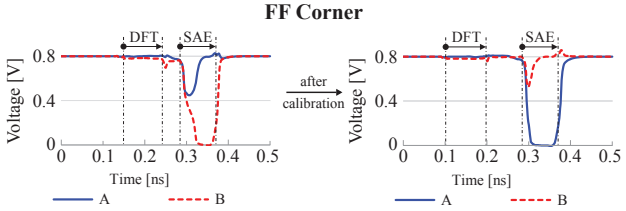
VI. Post-Silicon Calibration

Process variation (PV) is an important concern in scaled memories as it introduces random variations in transistors. PV impacts the proposed DFT scheme by altering the BL swing, the SA mismatch, and the stress introduced by the DFT. This results in two undesired scenarios: (1) the DFT may cause incorrect behavior in fault-free read operations (i.e., yield loss), or (2) the DFT does not trigger incorrect behavior during HTD faults (i.e., test escapes). A solution for both cases is using post-silicon calibration. We propose a simple calibration scheme that focuses on the stress introduced by the DFT. Fig. 9 depicts such scheme. Normally, a default delay unit (Fig. 9a) is used to delay the signals *WR0* and *WR1* that control the transistors that introduce mismatch. Changing the timing of these signals will impact the time interval between deactivating the DFT and enabling the SA, hence altering the mismatch introduced by the DFT during read operations. The additional calibration hardware is shown in Fig. 9b. In addition to the original delay path with a single delay unit, two supplementary delay paths are introduced; the top path has a lower delay, while the bottom path has a higher delay compared to the default. The delay paths are controlled by transmission gates, which are activated once the PV corner is identified. Note that this calibration is only necessary the first time the memory is tested.

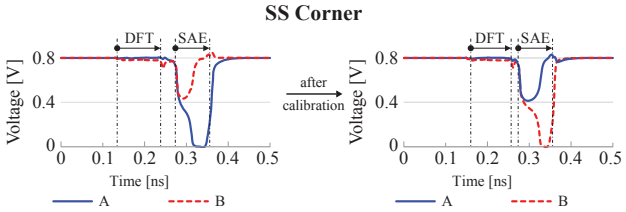


(a) Original timing scheme. (b) Modified DFT timing circuit.

Fig. 9. Timing scheme of the proposed DFT.



(a) Post-silicon calibration for FF corner.



(b) Post-silicon calibration for SS corner.

Fig. 10. Internal nodes of the SA before and after calibration.

We first describe the top delay path. The reduction in delay speeds up the DFT, thus increasing the time interval between deactivating the DFT and turning on the SA. Consequently, the SA has more time to recover from the mismatch caused by the DFT. This mode is useful when fault-free read operations are considered faulty. This scenario is depicted in Fig. 10a, which shows the internal nodes of the SA before and after calibration. Due to process variation, the circuit is in the FF corner, i.e., both PMOS and NMOS transistors are faster than designed. During standard read operations, a BL swing of 46 mV is observed. Therefore, the read output should be '0'. However, because of PV, an incorrect output is observed when activating the DFT. By calibrating the timing circuit, this is no longer observed; once enabled, the SA circuit correctly pulls node A to GND and B to V_{DD}.

On the other hand, the longer delay path delays the (de)activation of the DFT even further. Consequently, the time interval between deactivating the DFT and turning on the SA is reduced, leaving less time for the SA to recover from the DFT stress. This mode is useful when faulty read operations are considered correct as the defects have more time to recover. This is shown in Fig. 10b, where the SA resides in SS corner, i.e., both PMOS and NMOS transistors are slower than designed. In this scenario, HTD faults are triggered; during standard read operations, a BL swing of 34 mV is observed. Yet, due to PV, the stress introduced by the DFT does not trigger incorrect behavior. Calibrating the DFT to delay its activation leads to the detection of the otherwise test escape.

When the SA circuit is enabled, the stress from the DFT is still significant. Therefore, the SA pulls node A to V_{DD} and node B to GND leading to an incorrect output. Note that local PV variations are already counted for in the offset specification as explained in IV-B.

VII. Discussion and Conclusion

Based on the obtained results, we conclude the following:

Costs: the area overhead introduced by the proposed DFT scheme is minimal. The Control Unit (calibration circuit included) in Fig. 10b comprises 21 logic gates, which is negligible compared to other peripheral circuitry such as decoders. The DFT circuit in Fig. 5 introduces only 6 additional transistors into each SA. Due to the lack of a physical layout, area overhead cannot be accurately determined; we estimate it based on fin counting. The total fin count in the SA increases 15% due to the DFT circuit. However, the overhead with an 128-row memory column alongside all its peripherals (SA, write drive, precharge) is only 1.57%. This overhead is lower for larger memory columns and can be further reduced by sharing SAs among multiple columns.

The test time overhead of our scheme is also negligible. When using traditional March algorithms, only two additional operations ($\tilde{r}0$ and $\tilde{r}1$) are necessary to detect HTD faults. By doing so, we combine the fault coverage of these algorithms (i.e., ETD faults) with the fault coverage of the proposed DFT (i.e., HTD faults) with the same effort.

Comparison with state of the art: compared to traditional March tests, our DFT provides significant improvements in fault coverage as it enables the detection of HTD faults, which is not guaranteed by the former. Compared to monitoring hardware schemes [13, 15], our scheme significantly reduces hardware complexity and area overhead. The scheme proposed in [14] compares the bitline discharge pace to a static reference. However, this reference may not always indicate HTD or ETD faults. Therefore, this methodology may lead to yield loss. Our scheme prevents this by carefully designing the DFT so that only HTD faults are detected. Finally, the scheme in [12] is an interesting approach to detect HTD faults as it does not use additional hardware; it only changes the timing scheme of the memory. However, according to the authors, it may increase over-testing by up to 30%. Although it detects defects that might lead to early in-field failures, it also leads to an expressive yield loss as it considers many fault-free devices as faulty. Furthermore, none of the previously works mentioned the impact of PV, which may compromise their testing schemes. We have demonstrated that PV may lead to test escapes and yield loss, and have accordingly proposed a scheme to avoid these scenarios.

Drawbacks and limitations: while our scheme greatly improves detection of HTD faults, it also has drawbacks and limitations. The main drawback is the capacitance load introduced into the SA. Even when not active, the additional transistors used to unbalance the SA will impact the BL swing, although the impact is small. Namely, simulations of fault-free read operations have shown a BL swing decrease of 3 mV when the DFT circuit is added, which represents an impact of only 2.5% on the BL swing. Furthermore, the main limitation of our scheme is the defect and fault space. As it focuses on single-cell defects and faults, the proposed DFT scheme may not be able to detect HTD faults due to coupling faults or defects in WLs or BLs. Additional validation for multi-cell defects, faulty peripheral circuitry, and coupling faults is still necessary for a more complete solution.

Overall, our DFT scheme presents significant gains on fault coverage with negligible overheads in testing time and area. Clearly, the DFT scheme proposed in this work is a suitable methodology to improve the coverage of HTD faults and consequently reduce test escapes.

References

[1] A. Heinig *et al.*, "System integration - The bridge between More than Moore and More Moore," in *DATE 2014*. IEEE, 2014.

[2] P. G. Ryan *et al.*, "Process defect trends and strategic test gaps," in *2014 International Test Conference*, Oct 2014.

[3] Y. Liu *et al.*, "On modeling faults in FinFET logic circuits," in *IEEE 2012 International Test Conference*. IEEE, nov 2012.

[4] S. Hamdioui, *Testing Static Random Access Memories*, ser. Frontiers in Electronic Testing. Boston, MA: Springer US, 2004, vol. 26.

[5] M. Fieback *et al.*, "Device-aware test: A new test approach towards DPPB," in *International Test Conference*, 2019.

[6] S. Davidson, "Towards an understanding of no trouble found devices," in *IEEE VTS 2005*, May 2005.

[7] C. Acero *et al.*, "Embedded deterministic test points for compact cell-aware tests," in *2015 IEEE ITC*, Oct 2015.

[8] H. Tang *et al.*, "Using Cell Aware Diagnostic Patterns to Improve Diagnosis Resolution for Cell Internal Defects," in *2017 IEEE 26th Asian Test Symposium (ATS)*, Nov 2017.

[9] M. Shah *et al.*, "A Quality and Reliability Driven DFT and DFR Strategy for Automotive and Industrial Markets," in *2019 IEEE 37th VLSI Test Symposium (VTS)*. IEEE, apr 2019.

[10] A. Pavlov *et al.*, "Word line pulsing technique for stability fault detection in SRAM cells," in *IEEE ITC, 2005.*, Nov 2005.

[11] A. Ney *et al.*, "A new design-for-test technique for SRAM core-cell stability faults," in *2009 DATE*, April 2009.

[12] J. Kinseher *et al.*, "Improving SRAM test quality by leveraging self-timed circuits," in *2016 DATE*, March 2016.

[13] G. Medeiros *et al.*, "A defect-oriented test approach using on-Chip current sensors for resistive defects in FinFET SRAMs," *Microelectronics Reliability*, sep 2018.

[14] M.-C. Chen *et al.*, "A Built-in Self-Test Scheme for Detecting Defects in FinFET-Based SRAM Circuit," in *IEEE ATS 2018*. IEEE, oct 2018.

[15] G. Cardoso Medeiros *et al.*, "DFT Scheme for Hard-to-Detect Faults in FinFET SRAMs," in *IEEE European Test Symposium*, 2019.

[16] A. Gomez *et al.*, "Effectiveness of a hardware-based approach to detect resistive-open defects in SRAM cells under process variations," *Microelectronics Reliability*, 2016.

[17] A. Pavlov *et al.*, *CMOS SRAM Circuit Design and Parametric Test in Nano-Scaled Technologies*. Springer Netherlands, 2008, vol. 40.

[18] D. Bhattacharya *et al.*, "FinFETs: From Devices to Architectures," *Advances in Electronics*, sep 2014.

[19] T. Matsukawa *et al.*, "Comprehensive analysis of variability sources of FinFET characteristics," *2009 Symposium on VLSI Technology*, 2009.

[20] M. O. Simsir *et al.*, "Fault modeling for FinFET circuits," in *2010 IEEE/ACM International Symposium on Nanoscale Architectures*, 2010.

[21] H. Villacorta *et al.*, "FinFET SRAM hardening through design and technology parameters considering process variations," in *RADECS 2013*. IEEE, sep 2013.

[22] A. J. van de Goor *et al.*, "Functional memory faults: a formal notation and a taxonomy," in *18th IEEE VTS*, April 2000.

[23] W. Needham *et al.*, "High volume microprocessor test escapes, an analysis of defects our tests are missing," in *ITC 1998*, Oct 1998.

[24] Nanoscale Integration and Modeling (NIMO), "Predictive Technology Model (PTM)," 2012. [Online]. Available: http://ptm.asu.edu/