Analysis of resistive defects on a foundry 8T SRAM-based IMC architecture

Lila Ammoura, Marie-Lise Flottes, Patrick Girard, Jean-Philippe Noël, Arnaud Virazel

Analysis of Resistive Defects on a Foundry 8T SRAM-based IMC Architecture

L. Ammoura [1] M.-L. Flottes [1] P. Girard [1] J.-P. Noel [2] A. Virazel [1]

[1] LIRMM – Univ. of Montpellier / CNRS
F-34392 Montpellier
<name>@lirmm.fr

[2] Univ. Grenoble Alpes, CEA, LIST
F-38000 Grenoble
jean-philippe.noel@cea.fr

***Abstract*—A promising new alternative to efficiently solve the Von Neumann bottleneck problem is to adopt In-Memory Computing (IMC) architectures. Beyond the arithmetic operations, IMC architectures aim at integrating additional logic operators directly in the memory array or/and at the periphery in order to provide close computing abilities. However, they are subject to manufacturing defects in the same way as conventional memories. In this paper, a comprehensive model of a 128x128 bitcell array based on 28nm FD-SOI process technology has been considered to analyze the behavior of IMC 8T SRAM bitcells in the presence of resistive defects (open and short) injected in the read port. A hierarchical analysis allowing a thorough study of each defect has been carried in order to identify their impact in both memory and computing modes, locally on the defective bitcell as well as globally on the array. Experimental results show that the IMC mode offers the most effective detectability of resistive-short and resistive-open defects.**

Keywords— In-Memory Computing, 8T SRAM cell, resistive defect analysis, Test.

I. INTRODUCTION

Nowadays, computer systems dedicated for data-intensive applications are still based on the Von Neumann paradigm, which introduces major limitations such as reduced performance acceleration, increased power consumption and limited system scalability [1]. These limitations are mainly caused by the Von Neumann bottleneck that induces frequent data transfer between the main memory and the processor. Consequently, data storage and processing are the most critical challenges of the new Big Data paradigm. The design of high-performance computing systems could adopt a new approach that is data-centric, instead of the conventional model that is primarily computation-centric [2]. This approach consists in reducing data transfer by performing processing closer to where the data is stored in the system memory. Several alternative architectures have been proposed that fall into this category. One of them is called "Near-memory" which aims to process data close to where it resides. Another promising approach to efficiently solve the data-intensive applications problem is to adopt In-Memory Computing (IMC) architectures which consists of bringing the processing tasks inside the memory. Beyond conventional operations, IMC architectures aim at integrating additional logic in the memory array and at the periphery in order to provide close computing abilities and efficiently address the Von Neumann bottleneck problem [3].

IMC architectures can be built using different types of volatile or non-volatile basic memory cells. Memory computing primitives have been widely explored in CMOS-based SRAM [3-5] and DRAM [6-8] for both logic and arithmetic operations. IMC architectures based on DRAM and SRAM have also been proposed for applications such as graphic accelerators or Machine Learning (ML) [8-10]. On the other hand, data-intensive applications can be handled by IMC architectures based on memory devices, such as Resistive RAM (RRAM), Spin Transfer Torque MRAM (STT-MRAM) and even Spin Orbit Torque MRAM (SOT-RAM) [11-16]. A system named MAGIC, which stands for Memristor-Aided logic, has been presented in [14-16]. It is based on RRAM cells and allows to compute in-memory NOR operation.

A common aspect of volatile and non-volatile IMC architectures is that they are both prone to manufacturing defects, in the same way as conventional memories built with the same process technologies. In order to enable the use of this new computing paradigm in modern data processing units, the development of test solutions dedicated to IMC architectures is therefore mandatory. Two test solutions have been proposed in [17-18] to test the correct operations in computing mode of SRAM-based IMC architectures using 8T bitcells. These solutions mainly consist in modifying March test algorithms through the addition of computing operations. However, as shown by preliminary results presented in [19], these tests do not cover all potential defects that can occur in the IMC architecture. In particular, defects in the memory read port are not covered.

In this paper we present a study of resistive defects in the read port of 8T SRAM bitcells used in IMC architectures. This study has been carried out and validated on an IMC architecture based on 28nm FD-SOI process technology. The goal is to justify the addition of extra computing operations during the execution of the test procedure on an IMC architecture in order to cover all potential resistive defects in the memory array and IMC operators. The strategy is based on a qualitative study of all potential resistive-short and resistive-open defects that may affect the read port of 8T SRAM bitcells. The study has been carried out to identify the impact of each defect in both memory and computing modes, locally on the defective bitcell as well as globally on the array. Experimental results show the potential impact of the considered resistive defects on read/write/compute operations. These results also allow to extract the sensitization sequences for each defect in a first step, and then, to apply them on a realistic model to be validated in simulation in a second step.

The remainder of this paper is the following. Section II presents the structure of an 8T SRAM bitcell at the transistor level, explains the Read/Write and IMC operations and presents the model considered in this study. Section III

summarizes the existing test solutions for 8T SRAM-based IMC architectures proposed so far in the literature and discusses the detection efficiency of these solutions for resistive-short defects that may occur in the read port of the SRAM cells. In Section IV, the framework and the defect injection approach are first described. Then, a qualitative analysis of the considered defects is presented. Experimental results are reported in Section V. Finally, Section VI concludes the paper and gives future perspectives.

II. 8T SRAM-Based IMC and Array Organization

In this section, we first present the transistor level structure of an 8T SRAM bitcell, then explain its operating principles in memory mode (i.e., Read/Write) and in computing mode (i.e., NOR operation). Finally, we present the current IMC architecture used in our study.

A. IMC 8T SRAM bitcells Principle

IMC architectures allow computations to be performed directly in the memory instead of offloading the data to an external computing node. They can operate in two modes: *memory mode* and *computing mode*. In memory mode, the memory performs a read or a write operation on an addressed word. In computing mode, the memory executes an operation from at least two addressed words.

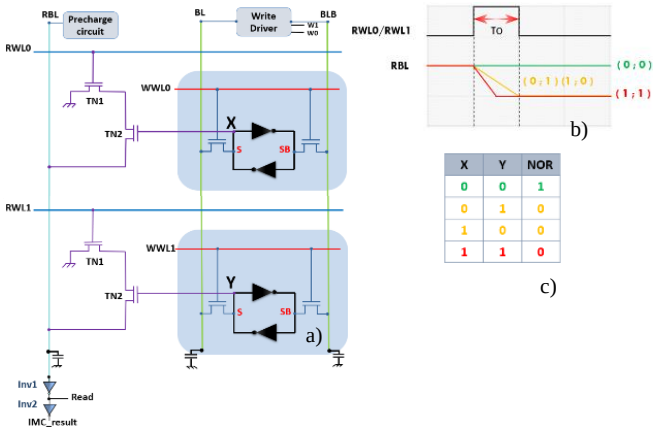

Figure 1. a) An example of 8T SRAM cells for in-memory computing, b) a waveform showing the execution of NOR operation and c) the resulting truth table

1) Memory Mode

In memory mode, the writing operation in an 8T SRAM bitcell is similar to that of the 6T SRAM one, so it is done in two steps:

- The write driver in Fig. 1.a drives the bit lines to the appropriate values by setting the state to be loaded on the Bit Line (BL) and its inverse on the Bit Line Bar (BLB).
- Then the address decoder sets the appropriate Write Word Line (WWL) to the high state. Since the access transistors are much larger than those of the bitcell inverters, the internal signals of the bitcell are forced to the values carried by the bit lines and the bistable circuit switches to the new stable configuration.

To read the content of the 8T SRAM cell, the Read Bit Line (RBL), initially precharged at Vdd, has a floating '1' at the beginning of the operation. Then, the Read Word Line (RWL) is activated. Let us consider the two following cases:

- If the considered bitcell stores a logic '0', the NMOS transistor TN2 of the read port will be in the off-state. The *RBL* will therefore be maintained at Vdd, so a logic '0' will be read on the read output port (considering the presence of an inverter at the ''Read'' output of *RBL*).
- In case the bitcell stores a logic '1' (*S* = '1'), the *RBL* will be discharged through TN1 and TN2, so that a logic '1' will be read on the read output port after inversion.

2) Computing Mode

The computing mode consists in a read operation performed between at least two (or more) 8T SRAM bitcells by simultaneously activating their *RWL* signals. The output of the read port subsequently shows a NOR behavior of the selected 8T SRAM cells.

Let us assume that we simultaneously activate *RWL0* and *RWL1* corresponding to the lines storing operand *X* and operand *Y* respectively, as shown in Fig. 1.a. The precharged *RBL* maintains its Vdd state if and only if the two operands *X* and *Y* are at logic '0'. In other words, as shown in Fig. 1.b, simply by activating simultaneously *RWL0* and *RWL1* signals for a time T0, *RBL* remains high only for *X* = '0' and *Y*= '0', and it completely discharges as soon as one of the operands at least is at logic '1'. The NOR operation is then computed with the data stored in the two selected bit cells as inputs and the result of the operation is deduced from the voltage of the *RBL*. An inverter (Inv1) is connected to the *RBL* so that the output of the inverter goes low if the *RBL* remains high. Thus, the output IMC_result of the cascaded inverter (Inv2) shows a NOR behavior (the truth table is given in Fig. 1.c). Note that if we store the complementary state in the bitcells (i.e., the node SB contains the input data to be computed) the output IMC_result of the cascaded inverter (Inv2) shows an AND behavior.

B. Considered IMC SRAM Array

To characterize the electrical behavior under realistic conditions, our study was conducted on the model presented in Fig. 2, which is a 128x128 bitcell array designed in 28 nm FD-SOI process technology [20]. The model is made of write drivers ensuring the writing operations in the bitcells and of precharge circuits, which maintain the *RBL* signals at Vdd necessary to perform the read operations and the computation at array level. The model also includes a layout extraction of the parasitic capacitances of the main signals (i.e., *BL*/*BLB*, *RBL* and *RWL*) that reinforces the realistic aspect of the model and allows achieving results that closely approximate those that can be achieved in a real circuit.

III. Background on 8T SRAM-based IMC Testing

In this section, we summarize the test solutions proposed so far in the literature for 8T SRAM-based IMC architectures. Next, we discuss the effectiveness of these solutions to detect resistive-short and resistive-open defects that may occur in the read port of 8T SRAM cells depending on their resistance values.

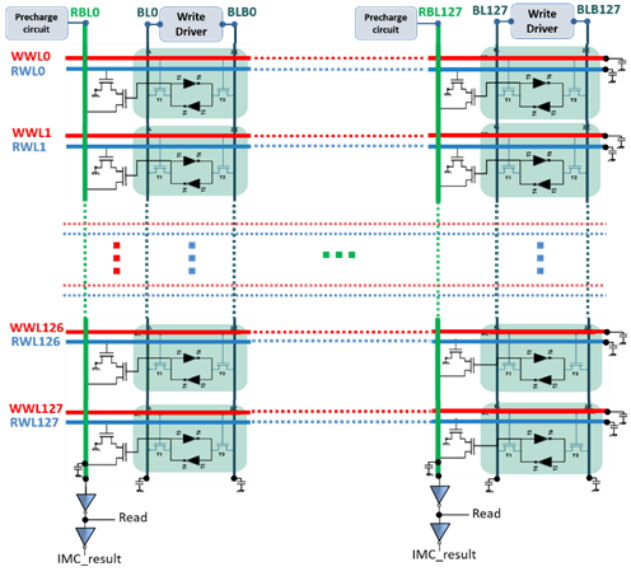

Figure 2. Considered 128x128 matrix model with layout extraction of parasitic capacities

A. Test Solutions in Memory Mode

Electrical faults in memories are generally modeled as functional faults and their detection is carried out using functional tests for a given set of Functional Fault Models (FFMs). A systematic approach is essential for increasing the performance and reliability of memories [21]. The three main steps of a typical test development methodology for SRAMs are:

1) *Defect analysis:* which can be performed using a physical model of the memory on which defect injection campaigns are performed.
2) *Fault modeling:* which consists of finding an appropriate FFM for each type of fault encountered during the defect analysis.
3) *Development of test algorithms* (e.g., March) which are developed to cover all possible FFMs that can be found in a given memory technology [22].

In [17-18], the March C- test algorithm is used to test the IMC 8T SRAM in the memory mode. The March C- test algorithm is the following {⇳ (w0); ⇑ (r0, w1); ⇑ (r1, w0); ⇓ (r0, w1); ⇓ (r1, w0); ⇳ (r0)} and has a complexity of 10N (N being the number of cells). It provides a 100% coverage of Stuck-at-Faults (SAF), Transition Faults (TF), idempotent and inversion Coupling Faults (CFid, CFin) and Address decoder Faults (AF), but does not cover all the resistive-short defects located in the read port of the IMC 8T SRAM cells [19]. In summary, functional tests and test algorithms already developed for conventionnel 6T SRAMs testing were used to test 8T SRAM-based computing architectures in memory mode in [17-18]. Unfortunately, the read port of the 8T SRAM bitcells is not fully tested. Therefore, it is not enough to use functional test and test algorithms of 6T SRAMs to fully test 8T SRAMs. Defects located in the isolated read port also need to be modeled and tested.

B. Test Solutions in Computing Mode

In computing mode, testing 8T SRAM-based IMC architecture has been addressed in [17], where the authors propose a March-like test algorithm to test the proper operation of the computation. So, to test computing operations, the two following requirements are considered:

- Requirement #1 is to perform a read operation or a NOR operation to detect the fault caused by an excessive leakage current when the data is all ‘0’.
- Requirement #2 is to perform a NAND operation in the worst cases where the operands are (0,1) or (1,0).

The first requirement is already satisfied by the first two operations that appear in the March C- test algorithm, and to satisfy the second requirement, modifications have been made to the March C- test algorithm, that consist in adding two NAND operations.

In summary, the test solutions proposed so far in the literature to cover defects in 8T SRAM IMC architectures allow to test the correct IMC operations **but do not cover all resistive defects that may occur in the read port of each SRAM bitcell of the array.**

IV. Resistive-Short & Resistive-Open Defect Analysis

This section first shows the locations of resistive-short and resistive-open defects injected in the read port of the 8T SRAM bitcell. Note that this study focuses only on the isolated read port of the 8T SRAM bitcell because defects on the remaining six transistors behave in the same way as defects in a conventional 6T SRAM cell and thus were not considered. We assume the presence of a single defect for each analysis because the occurrence of multiple defects is unlikely. Then, we detail the defect injection approach and finally present a qualitative analysis of the corresponding faulty behaviors.

A. Defect Injection Framework

As already discussed in the previous section, performing a computation in memory is ultimately equivalent to performing a Read operation on at least two bitcells of the same column. So, ensuring that the read operation operates correctly is essential for any IMC architectures. 8T SRAM bitcells are the most suitable for SRAM-based IMC, because they have a read port isolated from the write port, which ensures that the read operation does not interfere with the data content of the bitcell, even if several *RWLs* are activated simultaneously. This behavior makes these cells useful in the IMC context. Therefore, our goal is to analyze the impact of the resistive-short and resistive-open defects in the read port of 8T SRAM bitcells. Six resistive-short defects (see Fig. 3) and three resistive-open defects (see Fig. 4) are considered for each of the two transistors constituting the read port of 8T SRAM bitcell.

To proceed with the injection of resistive-short and resistive-open defects, a monitoring bitcell (*i; j*) (i.e., located at row *i* and column *j*) is targeted by a single defect injected at its read port. To analyze each injected defect, we set-up an approach to monitor each time the state of the faulty bitcell, the states of the neighboring bitcells (i.e., bitcells on the same column and row), and the computation results between a fault-free memory bitcell and the faulty bitcell. The purpose is to reveal the potential impact of each defect on the read/write/computing operations. The defect analysis is hierarchically performed as follows:

- Stand Alone Analysis (SA_Analysis): local impact on the defective bitcell itself during memory mode operations on that bitcell.
- Neighborhood Analysis (N_Analysis): It is done in two steps: i) impact on defect-free surrounding bitcells during memory mode operations on the faulty bitcell, and ii) local impact on the defective bitcell during memory mode operations performed on fault-free surrounding bitcells only.
- Computation Analysis (C_Analysis): It is done in two steps: i) impact on computing mode operations performed between the defective bitcell and at least a fault-free one in the same column i.e., NOR(c_a;c_v), and ii) impact on computing mode operations performed between at least two defect-free bitcells located in the same column than the defective one, i.e., NOR(c_v;c_v).

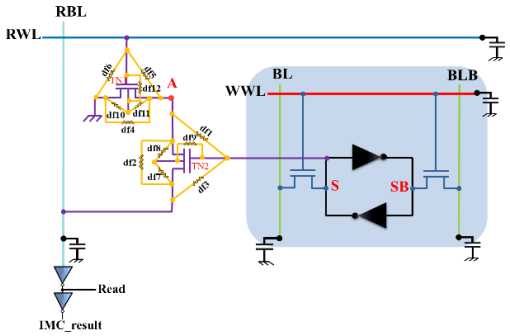

Figure 3. Resistive-short defects injection in the read port of an 8T SRAM cell

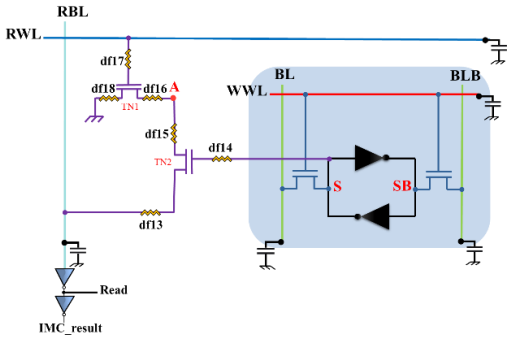

Figure 4. Resistive-open defect injection in the read port of an 8T SRAM cell

This hierarchical analysis allows a thorough study of each defect to identify their impact in both memory and computing modes locally on the defective bitcell as well as globally on the array. Moreover, it enables the definition of a Fault Primitive (FP) for each considered defect. As detailed in [21], a FP is denoted as:

- **<S/F/R>** when a single cell is involved; the cell Cv (victim cell) is used to sensitize a fault where it appears. **S** describes the Sensitizing Operation Sequence (SOS) that sensitizes the fault; S ∈ {0, 1, w0, w1, w↑, w↓, r0, r1}.
- **<Sa,Sv/F/R>** when two cells are involved; **Sa** describes the sensitizing operation or state of the aggressor cell, while **Sv** describes those of the victim cell; Si ∈ {0, 1, X, w0, w1, w↑, w↓, r0, r1} (i∈{a, v}), where X is the don't care value X∈{0, 1}.

In both notations**, F** describes the value or the behavior of the faulty cell; **F** ∈ {0, 1, ↑, ↓, -} where ↑ (resp. ↓) means the faulty cell undergoes a transition. **R** describes the logic output level of a read operation in case **S** contains read operations. Generally, it takes one of the values {0, 1, -}, where '-' is used when no read operation is required for the SOS.

B. Qualitative Resistive-Short Defects Analysis

Each injected resistive-short defect induces a faulty behavior in both memory and computing modes. As a case study, the faulty behavior produced by the resistive-short defect df1, which creates a resistive connection between the storage node S and the potential denoted A in Fig. 3, is described below.

- SA_Analysis: During a write operation, the *RWL* remains at logic '0', which means that transistor TN1 is always blocked. Consequently, node *S* will not be disturbed during any write operation on the defective bitcell. On the other hand, during a read operation, as soon as the *RWL* is activated, transistor TN1 becomes passing, then node *S* is grounded. Thus, the Read operation is destructive in the case where the defective bitcell initially stores a logic '1'. In other words, R1 operation performed on the defective cell operates as a W0 operation and returns a logic '0' to the output port (i.e., inverter of the read port). This faulty behavior is represented in Fig. 5.

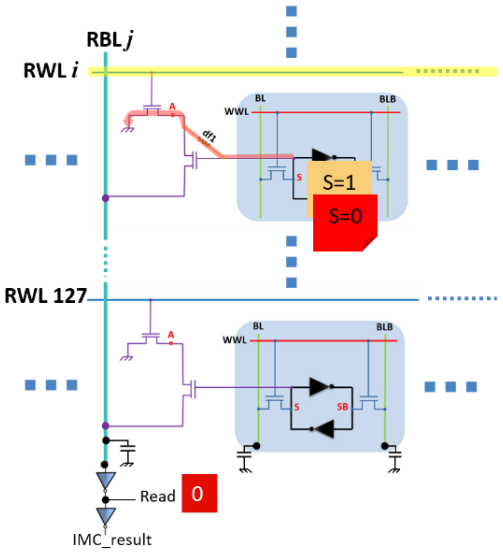

Figure 5. Faulty behavior of the bitcell in presence of df1

- N_Analysis: Since write operations on the defective bitcell are not impacted, any write operation on defect-free surrounding memory cells is correctly acted and vice versa. Faulty behaviors appear during Read operations. Let us consider the case where the faulty bitcell stores a logic '1' and a read operation on a defect-free bitcell of the same row is acted. Upon activation of the *RWL*, the content of the defective bitcell will be forced to logic '0'. In summary, when the defective bitcell stores a logic '1', activating the *RWL* signal to act any Read operation on

another bitcell of the same row forces its content to a logic '0'.

- C_Analysis: In computing mode, df1 will present the same destructive Read problem. Let us assume a computation between the defective bitcell storing a logic '1' and a defect-free bitcell on the same column storing a logic '0'. As soon as the appropriate *RWL* signals are activated, the content of the defective cell switches to a logic '0'. Consequently, instead of computing the NOR (1;0) operation that must provide a logic '0', the output IMC_result of the read port provides a logic '1' corresponding to the NOR (0;0) (see Fig. 2).

All the injected resistive-short defects have been analyzed in the same way as detailed for df1. Table 1 summarizes the results of the analysis and reports at each analysis step (SA_Analysis, N_Analysis, C_Analysis) the operations affected ("RX"/"WX"/"NOR($c_{a/v}$;c_v)" when the read/write/computing operation, respectively, is affected, "-" if no operation is affected).

Table 1. Summary of the qualitative resistive-short defect analysis

Defect	SA_Analysis	N_Analysis		C_Analysis	
	Operation	Same Row	Same Column	IMC NOR(c_a;c_v)	IMC NOR(c_v;c_v)
df1	Destructive R1	Destru-ctive Rx	-	NOR(1;0)	-
df2	R0	-	-	NOR(0;0)	-
df3	W0, R0, R1	-	-	NOR(1;0)	-
df4	-	-	R0	-	NOR(0;0)
df5	R1	-	R0	NOR(1;0)	NOR(0;0)
df6	R1	R1	-	NOR(1;0)	-
df7	R0	-	R0	NOR(0;0)	-
df8	-	-	R0	-	NOR(0;0)
df9	W1, R1	-	-	NOR(1;0)	-
df10	-	-	-	-	-
df11	-	-	R0	-	NOR(0;0)
df12	R1	R1	-	NOR(1;0)	-

C. *Qualitative Resistive-Open Defects Analysis*

Each injected resistive-open defect induces a faulty behavior during the memory mode as well as computing mode. As case study, faulty behaviors produced by the resistive-open defect df13 (see Fig. 4) is described below.

- SA_Analysis: Since the read port is isolated from the write port, the write operation is not disturbed by df13. The presence of this defect produces a delay, which is directly related to the RBL, which may disturb its discharge during a R1 operation. Let us consider the example of a defective bitcell storing a logic '1'. In order to read its contents, the RWL signal is activated. In the case of a fault-free operation, the RBL discharges through the two transistors TN1 and TN2. In presence of df13, the discharge of the RBL is delayed and, the greater the resistance value of the defect, the greater the delay produced. So, at a certain value of the resistive-open defect, the value read will not be captured at the output of the read port.
- N_Analysis: The operations performed on the defective cell do not affect the functioning of the neighboring cells (same row and same column). All read/write operations on the surrounding defect-free bitcells are performed correctly and vice-versa.
- C_Analysis: The operation in computing mode between defect-free bitcells of the same column is not affected by df13. However, the computation performed with bitcells including the defective one is affected by the delay produced by df13. Let us consider the example of a defective bitcell storing a logic '1' and another defect-free bitcell in the same column storing a logic '0' implying a NOR (1;0) operation. As seen previously, the presence of df13 produces a delay which slows down the discharge of the RBL so that the data is not captured at the output of the reading port. For high resistance values of df13, the RBL does not discharge. So, the IMC_result output of the read port provides a logic '1' instead of a logic '0'. This faulty behavior is represented in Fig. 6.

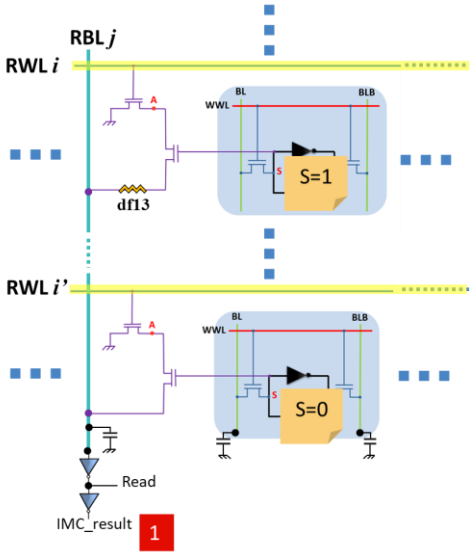

Figure 6. Faulty behavior in computing mode in presence of df13

All the injected resistive-open defects have been analyzed in the same way as detailed for df13. Table 2 summarizes the results of the analysis and reports at each analysis step (SA_Analysis, N_Analysis, C_Analysis) the operations affected ("RX"/"WX" when the read/write operation is affected, "-" if no operation is affected).

Table2. Summary of the qualitative resistive-open defect analysis

Defect	SA_Analysis	N_Analysis		C_Analysis	
	Operation	Same Row	Same Column	IMC NOR(c_a;c_v)	IMC NOR(c_v;c_v)
df13	R1	-	-	NOR(1;0)	-
df14	R1	-	-	NOR(1;0)	-
df15	R1	-	-	NOR(1;0)	-
df16	R1	-	-	NOR(1;0)	-
df17	R1	-	-	NOR(1;0)	-
df18	R1	-	-	NOR(1;0)	-

V. Experimental Results

In this section, we present the SPICE simulation results for the resistive-short defects df1 and the resistive-open defect df13 analyzed in the previous subsection. A summary of the observed faulty behaviors of the considered resistive defects is provided at the end.

A. Experimental Setup

All the electrical simulations of the injected defects have been performed using the XA simulator from Synopsys [23] considering a 128x128 bitcell matrix model (see Fig. 1) designed using a 28nm FD-SOI process technology. The simulations have been carried out by applying sequences of operations deduced from the previous subsection with the aim of sensitizing each defect.

All injected resistive defects cause a read delay that may induce faulty behaviors. This produced delay depends on the size of the injected defect resistance and the discharge time of the RBL (i.e., parasitic capacitors). Therefore, it is necessary to define a reading time T_{read}, at which we ensure that the data at the output of the read port is captured in the meantime. Thus, T_{read} is deduced from the maximum time T0, which is required by the RBL to completely discharge at the operating environment of a typical process corner, 1V supply voltage and 125°C temperature. Then, T0 is added to a margin of T0/2 as the necessary delay for the data to pass through the read inverters. Based on simulations performed at the operating environment described above, T0 is measured at about 700ps, so T_{read} is selected at 1ns for the rest of the defect injection campaign. To extract the minimum resistance value R_{min} of each resistive-open defect and the maximum resistance value R_{max} of each resistive-short defect, a threshold is defined at the level of the RBL voltage, which is 30% of Vdd when the correct output is logic '0' and 70% of Vdd when it is logic '1' (i.e., 300mv for logic '0' and 700mv for a logical '1') at the T_{read} instant.

In the following, the read operation "RX" and the IMC operation "NOR(X;Y)" are checked by the behavior of the RBL signal. The RBL behavior is considered correct if the response is provided during the Tread≈1ns. Beyond this read time, the data is not captured at the output of the read port, so the operations are not executed properly, which leads to a faulty behavior.

B. Simulation Results of Resistive-Short Defects

The simulation campaigns are based on the results reported in Table 1 obtained through the qualitative analysis of resistive-short defects. Table 3 summarizes all the results obtained for all the injected resistive-short defects. It is constituted by the three considered categories of analysis (SA_Analysis, N_Analysis and C_Analysis). In each category, the operations affected by the injected defect are reported by detailing each time the maximum size (i.e., R_{max}) of the defects that leads to the faulty behavior. For the category C_Analysis, it is divided into two sub-categories (see the fourth column of Table 3). Each sub-category is divided into 3 columns: the first column contains the assigned operation (NOR(c_a;c_v) and/or NOR (c_v;c_v)), the second column reports the maximum size of the defect which leads to a faulty behavior when the computing is carried out between two bitcells (i.e., N=1), the third column contains the maximum size of the defect when the computing is global (i.e., it is carried out on all the cells of the column N=127). In the last column, the sequence of operations allowing the sensitization of each defect is determined according to the operation that generates the maximum resistance value, i.e., the critical resistance value "$R_c = \max\{R_{max}\}$" of the resistive-short defect highlighted in bold in Table 3 for each defect, in order to cover the largest range of these resistive defects. R_c is highlighted in bold in Table 3 for each resistive-short defect.

For example, in the case of df1, the maximum defect size is achieved with the C_Analysis with N=127 (i.e., R_c=37kΩ) that corresponds to a NOR(1;0^{127}) computing operation. So, the sequence <1,0^{127} NOR(1;0^{127})/0^{127}/1> (detailed below) will be applied on all the bitcells of the column where the defective bitcell is located, as follows:

<1,0^N NOR(1;0^N)/0^N/1>

where a logic '1' is initially stored in the defective bitcell. A logic '0' is initially stored in N bitcells of the same column as the defective one. Then, a NOR(1;0^N) operation is performed between all the selected bitcells. The N bitcells

Table 3. Summary of resistive-short defect simulation results

Defect	SA_Analysis		N_Analysis				C_Analysis						<S/F/R>/ <Sa,Sv/F/R>
			Same Row		Same Column								
	Operation	R_{max} Ω	Operation	R_{max} Ω	Operation	R_{max} Ω	IMC Operation NOR(c_a;c_v)	R_{max} Ω N=1	R_{max} Ω N=127	IMC Operation NOR(c_v;c_v)	R_{max} Ω N=1	R_{max} Ω N=126	
df1	R1	36.5k	Rx	27.4k	-	-	NOR(1;0^N)	36.5k	**37k**	-	-	-	<1,0^N NOR(1;0^N)/0^N/1>
df2	R0	101k	-	-	-	-	NOR(0;0^N)	**104k**	99k	-	-	-	<0,0^N NOR(0;0^N)/0^N/1>
df3	W0	5k	-	-	-	-	NOR(1;0^N)	15.9k	15.2k	-	-	-	<0R0/0/1>
	R0	**112k**					NOR(0;0^N)	108k	100k				
	R1	15.9k											
df4	-	-	-	-	R0	**70k**	-	-	-	NOR(0;0^N)	70k	64k	<1,0R0/0/1>
df5	R1	5.4k	-	-	R0	70k	NOR(1;0^N)	5.4k	5.4k	NOR(0;0^N)	70k	**78k**	<1,0^N NOR(0;0^N)/0^N/0>
df6	R1	223	R1	**269**	-	-	NOR(1;0^N)	223	228.4	-	-	-	<1,1R1/1/0>
df7	R0	104k	-	-	R0	102k	NOR(0;0^N)	**105k**	100k	-	-	-	<0,0^N NOR(0;0^N)/0^N/0>
df8	-	-	-	-	R0	**70k**	-	-	-	NOR(0;0^N)	70k	65k	<X,0R0/0/1>
df9	W1	29k	-	-	-	-	NOR(1;0^N)	38.2k	**39.4k**	-	-	-	<1,0^N NOR(1;0^N)/0^N/1>
	R1	38.2k											
df10	-	-	-	-	-	-	-	-	-	-	-	-	-
df11	-	-	-	-	R0	**70k**	-	-	-	NOR(0;0^N)	70k	66k	<1,0R0/0/1>
df12	R1	223	R1	**269**	-	-	NOR(1;0^N)	223	229	-	-	-	<1,1R1/1/0>

remain at logic '0'. The output level of the logical operation is a logic '1'.

Waveforms in Fig. 6 present the SPICE simulation performed on the 128x128 bitcell array using the sequence of operations allowing the sensitization of the resistive-short defect df1, i.e., $<1,0^{127}$ NOR$(1;0^{127})/0^{127}/1>$ at its maximum detectable resistance with typical PVT conditions (Process Typ, Voltage 1V and Temperature 27°C). Thus, resistances above this critical value R_c=37kΩ lead to a correct computing operation (i.e., NOR$(1;0^N)$=0; N=127) and resistances lower than R_c lead to an incorrect behavior.

The aggressor bitcell initially contains a logic '1' and the victim bitcells contain a logic '0'. Then, a NOR is performed on the whole column by activating all the RWLs signal simultaneously at t=42ns. The RBL signal starts to discharge until it reaches 300mv at T_{read} time (i.e., 43ns), which are the two considered limits for extracting the maximum resistance of detectability. The red signal in Fig. 7 represents the result of the NOR operation at the output of the read port, i.e., the IMC_result signal in Fig. 2. Note that the blue dotted lines in Fig.7 represent the defect-free behavior (i.e., when no defect is injected).

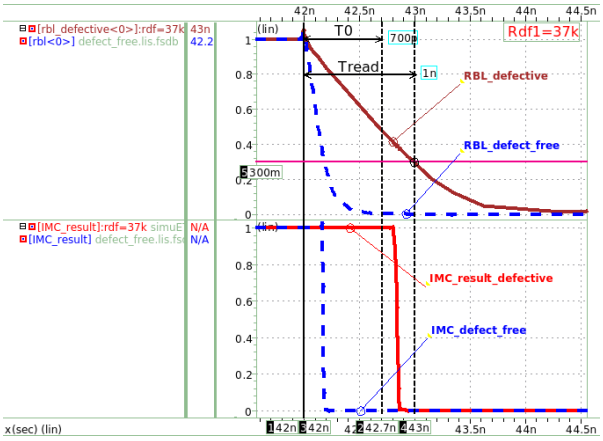

Figure 7. Waveforms of the sensitization sequence "<1,0^N NOR(1;0^N)/0^N/1>, with N=127" with df1 size set at 37kΩ.

C. Resistive-Open Defect Simulation Results

Table 4 summarizes all the results obtained for all the injected resistive-open defects. The simulation campaigns are based on the results obtained during the qualitative analysis of the defects. Table 4 shows the two categories where the operations affected by the defects appear (SA_Analysis and C_Analysis). The first category (SA_Analysis) is represented in the second column detailing the minimum size of defects that lead to this faulty behavior (i.e., R_{min}). For the category C_Analysis, it is divided into 3 groups (cf. the third column of Table 2), the computation is performed between 2 bitcells (i.e., N=1), then 16 bitcells (i.e., N=15) and between all the cells of the column (i.e., N=127), while specifying each time the minimum value of the resistance of the defect which leads to this faulty behavior. For each injected defect, the sequence of operations allowing its sensitization is determined (last column in Table 2) according to the operation that generates the minimum resistance value, i.e., the critical resistance value "$R_c = \min\{R_{min}\}$" of the defect, in order to cover the largest range of these resistive defects. R_c is highlighted in bold in Table 4 for each resistive-open defect.

For example, in the case of df13, the minimum defect size is achieved with the C_Analysis with N=127 (i.e., R_c=26.8kΩ) that corresponds to a NOR$(1;0^{127})$ computing operation. So, the sequence $<1,0^{127}$ NOR$(1;0^{127})/0^{127}/1>$ (detailed below) will be applied on all the bitcells of the column where the defective bitcell is located as explained earlier.

Waveforms in Fig. 8 present the SPICE simulation performed on the 128x128 bitcell array using the sequence of operations allowing the sensitization of the resistive-open defect df13, i.e., $<1,0^{127}$ NOR$(1;0^{127})/0^{127}/1>$ at its minimum detectable resistance. Thus, resistances below this critical value R_c=26.8kΩ lead to a correct computing operation (i.e., NOR$(1;0^N)$=0; N=127) and resistances higher than R_c lead to an incorrect behavior.

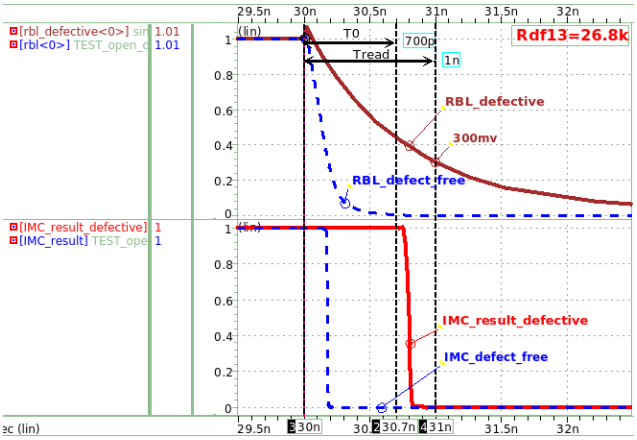

Figure 8. Waveforms of the sensitization sequence "<1,0^N NOR(1;0^N)/0^N/1>, with N=127" with df13 size set at 26.8kΩ.

Table 4. Summary of resistive-open defect simulation results

Defect	SA_Analysis		C_Analysis				<S/F/R>/ <Sa,Sv/F/R>
	Operation	R_{min} Ω	IMC Operation NOR(c_a;c_v)	R_{min} Ω N=1	R_{min} Ω N=15	R_{min} Ω N=127	
df13	R1	≈31k	NOR$(1;0^N)$	≈31k	≈29.8k	≈ **26.8k**	$<1,0^N$ NOR$(1;0^N)/0^N/1>$
df14	R1	≈16.79M	NOR$(1;0^N)$	≈16.89M	≈16.59M	≈**14.56M**	$<1,0^N$ NOR$(1;0^N)/0^N/1>$
df15	R1	≈23.2k	NOR$(1;0^N)$	≈23.2k	≈22.9k	≈**20k**	$<1,0^N$ NOR$(1;0^N)/0^N/1>$
df16	R1	≈23.2k	NOR$(1;0^N)$	≈23.2k	≈22.9k	≈**20k**	$<1,0^N$ NOR$(1;0^N)/0^N/1>$
df17	R1	≈4.39M	NOR$(1;0^N)$	≈4.44M	≈4.42M	≈**4.35M**	$<1,0^N$ NOR$(1;0^N)/0^N/1>$
df18	R1	≈21k	NOR$(1;0^N)$	≈20.6k	≈20.6k	≈**17.7k**	$<1,0^N$ NOR$(1;0^N)/0^N/1>$

The aggressor bitcell initially contains a logic '1' and the victim bitcells contain a logic '0'. Then, a NOR is performed on the whole column by activating all the RWLs signal simultaneously at t=30ns. The RBL signal starts to discharge until it reaches 300mv at T_{read} time (i.e., 31ns), which are the two considered limits for extracting the minimum resistance of detectability. The red signal in Fig. 8 represents the result of the NOR operation at the output of the read port, i.e., the IMC_result signal in Fig. 2. The dotted blue lines in Fig. 8 represent the defect-free behavior.

D. Discussion on the Defect Injection Results

According to the defect behavior presented so far for an IMC 8T SRAM bitcell with resistive-open and resistive-short defects at the read port, some detectability conditions can be deduced. As shown in the results reported in Table 3 and Table 4, the IMC mode offers a better detectability of almost 50% of the injected resistive-short defects and 100% of the injected resistive-open defects.

Let us first discuss the results obtained for the injected resistive-short defects. Global computing involving all the bitcells of the same column improves the detectability of larger sizes of df1, df5 and df9 (i.e., the more cells are involved, the more defects of larger sizes are detected). The IMC performed between two bitcells provides a better coverage of resistive defects of both types df2 and df7. On the other hand, the resistive-short defect df3 is detectable by simple memory operations (i.e., Read/Write) applied on the defective bitcell. The five resistive-short defects df4, df6, df8, df11 and df12 are detectable by memory operations applied on a victim bitcell (i.e., defect-free bitcell) of the same column or the same row as the defective one. Conversely, df10 is not detectable because it does not affect the operation in either memory or computing mode.

Let us now discuss the results obtained for the injected resistive-open defects. As shown in the results reported in Table 4, the IMC mode implying all the bitcells of the same column offers a better detectability of all the injected defects, i.e., an improvement of up to 13.8% for df13 to df16, of 15.7% for df18 and of 1% for df17. Note that the improvement in resistance values is different depending on the location of each defect. Moreover, different ranges of critical resistance have been found. The minimum resistance of defects that are connected to the gates of transistors is in the MΩ range (i.e., df14 and df17), while the minimum resistance for the other defects (i.e., df13, df15, df16, df18) is in the kΩ range.

From these results, the main conclusion is that the IMC mode improves the detectability of most of the considered resistive defects by involving all bitcells of the same column for a computing operation (for specific resistive-short defects and for all the resistive-open defects). Consequently, the row decoder must be adapted in order to allow a computing operation involving all bitcells of each column for detecting and covering smaller sizes of resistive-open defects and larger sizes of resistive-short defects.

VI. CONCLUSION

In this paper, we first detailed the operating principle of 8T SRAM bitcells in their two operation modes. Then, we presented the comprehensive memory model considered in our study (128x128 bitcell array in 28nm FD-SOI process technology). We highlighted the fact that the algorithms proposed in the literature to test 8T SRAM-based IMC architectures do not completely cover the resistive defects that can affect the read port of 8T SRAM memory bitcells. Then, we presented our analysis for a thorough study of intra-cell resistive-open and resistive-short defects injected into the read port. Impacts in both memory and computation modes were identified, both locally (on the defective bitcell), and globally (on the array). Then, we reported results obtained during the simulation campaigns based on the qualitative analysis by specifying the critical size of the defects for which they are detectable. The obtained results show that the IMC mode improves the detectability of several injected resistive defects.

Our future work will consist, in a first step, in analyzing the inter-cell resistive defects so that, in a second step, it will be possible to develop an effective test and design-for-test solutions that allow to cover all the defects that can affect the IMC 8T SRAM architectures.

REFERENCES

[1] A. Jaiswal, I. Chakraborty, A. Agrawal and K. Roy, "8T SRAM Cell as a Multibit Dot-Product Engine for Beyond Von Neumann Computing," in IEEE Transactions on Very Large Scale Integration Systems, vol. 27, no. 11, pp. 2556-2567, Nov. 2019.

[2] S. Hamdioui et al., "Memristor based computation-in-memory architecture for data-intensive applications," 2015 Design, Automation & Test in Europe Conference & Exhibition, 2015, pp. 1718-1725.

[3] A. Agrawal, A. Jaiswal, C. Lee and K. Roy, "X-SRAM: Enabling In-Memory Boolean Computations in CMOS Static Random Access Memories," in IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 65, no. 12, pp. 4219-4232, Dec. 2018.

[4] A. Jaiswal, I. Chakraborty, A. Agrawal and K. Roy, "8T SRAM cell as a multibit dot-product engine for beyond von Neumann computing," IEEE Trans. Very Large Scale Integr. Syst., vol. 27, no. 11, pp. 2556-2567, Nov 2019.

[5] A. Biswas and A. P. Chandrakasan, "CONV-SRAM: An energy-efficient SRAM with in-memory dot-product computation for low-power convolutional neural networks," IEEE J. Solid-State Circuits, vol. 54, no. 1, pp. 217-230, Jan 2018.

[6] V. Seshadri et al., "Ambit: In-memory accelerator for bulk bitwise operations using commodity DRAM technology," Proc. 50th Annu. IEEE/ACM Int. Symp. Microarchit., pp. 273-287, 2017.

[7] M. F. Ali, A. Jaiswal and K. Roy, "In-memory low-cost bit-serial addition using commodity dram technology," IEEE Trans. Circuits Syst. I Reg. Papers, vol. 67, no. 1, pp. 155-165, Jan 2020.

[8] S. Angizi and D. Fan, "GraphiDe: A Graph Processing Accelerator leveraging In-DRAM-Computing," ACM Great Lakes Symposium on VLSI, Washington, D.C. USA, May 9-11, 2019.

[9] J. Zhang, Z. Wang, and N. Verma, "In-memory computation of a machine-learning classifier in a standard 6T SRAM array," IEEE Jour. of Solid-State Circuits, vol. 52, no. 4, pp. 915–924, Apr. 2017.

[10] https://www.eejournal.com/article/in-memory-computing/, In-Memory Computing, No Fewer than Four Approaches, by Bryon Moyer, 2019.

[11] F. Parveen, S. Angizi, and D. Fan, "IMFlexCom: Energy Efficient In-memory Flexible Computing using Dual-mode SOT-MRAM," ACM Journal on Emgerging Technologies in Computing Systems, Vol. 14, no.3, Oct. 2018.

[12] Z. He, Y. Zhang, S. Angizi, B. Gong and D. Fan, "Exploring A SOT-MRAM based In-Memory Computing for Data

Processing," IEEE Transactions on Multi-Scale Computing Systems, 2018.

[13] M. Kooli, H.-P. Charles, C. Touzet, B. Giraud, and J.-P. Noel, "Smart instruction codes for in-memory computing architectures compatible with standard SRAM interfaces," In Proc. Design, Automation & Test in Europe Conference & Exhibition, 2018.

[14] S. Kvatinsky et al., "MAGIC—Memristor-Aided Logic," IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 61, nº 11, p. 895-899, nov. 2014.

[15] R. Ben Hur, N. Wald, N. Talati, et S. Kvatinsky, "Simple magic: Synthesis and in-memory Mapping of logic execution for memristor-aided logic," in 2017 IEEE/ACM International Conference on Computer-Aided Design , Irvine, CA, 2017, p. 225-232.

[16] A. Haj-Ali, R. Ben-Hur, N. Wald, R. Ronen, et S. Kvatinsky, "IMAGING-In-Memory AlGorithms for Image processiNG," IEEE Transactions on Circuits and Systems I: Regular Papers, p. 1-14, 2018.

[17] T.L. Tsai, J.F. Li, C.-L. Hsu and C.-T. Sun, "Testing of In-MemoryComputing 8T SRAMs," Proc. IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems, 2019.

[18] J.-F. Li, T.-L. Tsai, C.-L. Hsu and C.-T. Sun, "Testing of Configurable 8T SRAMs for In-Memory Computing," Proc. IEEE Asian Test Symposium, 2020.

[19] L. Ammoura, M.-L. Flottes, P. Girard and A. Virazel, "Preliminary Defect Analysis of 8T SRAM Cells for In-Memory Computing Architectures," 16th International Conference on Design & Technology of Integrated Systems in Nanoscale Era, 2021, pp. 1-4.

[20] N. Planes et al., "28nm FDSOI technology platform for high-speed low-voltage digital applications," 2012 Symposium on VLSI Technology, 2012, pp. 133-134.

[21] A.J. van de Goor and Z. Al-Ars, "Functional Memory Faults: A Formal Notation and a Taxonomy", VLSI Test Symposium, pp. 281-289, 2000.

[22] A. Bosio, L. Dilillo, P. Girard, S. Paravossoudovitch, and A. Virazel, "Advanced Test Methods for SRAMs," ISBN 978-1-4419-0938-1, Springer, 2009.

[23] "XA User Guide."https://www.synopsys.com/content/dam/synopsys/implementation&signoff/datasheets/primesim-xa-ds.pdf