A Fault Coverage Evaluation of Linked Neighborhood Pattern-Sensitive Faults in Random-Access Memories

Cristina Huzum*, Petru Caşcaval*

** Faculty of Automatic Control and Computer Engineering, "Gheorghe Asachi", Technical University of Iaşi, Romania (e-mail: cristinahuzum@yahoo.com, cascaval@cs.tuiasi.ro).*

Abstract: A fault coverage evaluation concerning a linked neighborhood pattern sensitive faults model (NPSFs) in $N \times 1$ random-access memories is presented. For the simulation study, the most important published tests dedicated to the NPSF model have been considered. Simulation results show that these tests cover the entire model of simple NPSFs, but only the longer of them cover the linked NPSF model. In addition, a fault coverage evaluation of linked faults involving a single cell (LF1s), two cells (LF2s), and three cells (LF3s) is also presented.

Keywords: Memory Testing, Functional Fault Model, Static Faults, Linked Neighborhood Pattern-Sensitive Faults, March Tests.

1. INTRODUCTION

Rapid increase of density in the integrated circuits has an immediate effect upon memory testing. On one hand, the capacity of random-access memories chips enhances, thus increasing the test time and cost; on the other hand, the density of memory circuits grows, therefore more failure modes and faults need to be taken into account in order to obtain a good quality product. Accordingly, there are two conflicting constraints that need to be dealt with when considering a test algorithm: reducing the number of memory operations in order to permit large capacity memories to be tested in an appropriate period of time and covering a larger variety of memory faults (Adams 2003; Hamdioui 2004).

As a result of the increasing coupling effect triggered by the growing density of memory circuits, the pattern-sensitive fault (PSF) is becoming an important fault model (Hayes 1980; Kang & Cho 2000; Cockburn 1995). The PSF model is a type of coupling fault, with several aggressor cells (4, 9 etc.) and only one victim cell. In this work, the neighborhood PSF (NPSF) has been considered. This is a particular PSF, in which the aggressor cells are located in the physical neighborhood of the victim cell. The NPSF model was first defined by J.P. Hayes in 1980. He also devised a memory test for this model (Hayes 1980). Soon after that, D.S. Suk and M. Reddy have proposed a new memory test (Suk & Reddy 1980) based on a bipartite method. This test divides the memory cells into two partitions and applies a sequence of transitions to cover all possible victim-aggressor combinations. Unfortunately, for the memory chips currently used, the test proposed by Suk and Reddy needs a long time to perform. In 2002, other more efficient march tests were given by Cheng, Tsai, and Wu, namely: CM-79N and March-100N. In another paper, written in 2008, Julie, Wan Zuha and Sidek use a modified version of March-100N for diagnosis of SRAM. In all these papers the authors have limited themselves only to the class of simple faults. In this work we focused on the problem of testing the linked neighborhood pattern-sensitive faults. Also, the most common linked faults composed of simple single-cell and simple two-cell faults (LFs from here on) have been considered for this simulation study.

The remainder of this paper is organized as follows. Section 2 introduces some notations and definitions, and Section 3 defines the set of fault primitives for the neighborhood pattern-sensitive fault model. Section 4 presents the memory tests we have considered for the simulation study. Section 5 presents experimental results regarding the ability of these important published tests dedicated to the NPSF model to cover linked NPSFs. Section 6 introduces some notations concerning the LFs model, and reports simulation results regarding the fault coverage of this model by the tests presented in Section 4. Some conclusions concerning this work are drawn in Section 6.

2. NOTATIONS, DEFINITIONS AND FAULT CLASSIFICATIONS

An operation sequence that results in a difference between the observed and the expected memory behaviour is called a *sensitizing operation sequence* (*S*). The observed memory behaviour that deviates from the expected one is called *faulty behaviour* (*F*). In order to specify a certain fault, one has to specify the *S*, together with the corresponding faulty behaviour *F*, and the read result (*R*) of *S* in case it is a read operation. The combination of *S*, *F* and *R* for a given memory failure is called a *Fault Primitive* (FP), and is usually denoted as $< S / F / R >$. The concept of FPs allows for establishing a complete framework of all memory faults. Some classifications of FPs can be made based on different and independent factors of *S* (Hamdioui, van de Goor & Rodgers 2002).

a) Depending on the number of simultaneous operations required in the *S*, FPs are classified into *single-port* and *multi-port* faults.

- *Single-ports faults:* These are FPs that require at the most one port in order to sensitize a fault. Note that single-port faults can be sensitized in single-port as well as in multi-port memories.
- *Multi-port faults:* These are FPs that can only sensitize a fault by performing two or more operations simultaneously via different ports.

b) Depending on the number of simultaneous operations required in the *S*, FPs are classified into *static faults* and *dynamic faults*.

- *Static faults:* These are FPs which sensitize a fault by performing at most one operation in the memory (#O=0 or #O=1);
- *Dynamic faults*: These are FPs that perform more than one operation sequentially in order to sensitize a fault (#O > 1).

c) Depending on the way FPs manifest themselves, they can be divided into *simple faults* and *linked faults*.

- *Simple faults:* These are faults which cannot be influenced by another fault. That means that the behaviour of a simple fault cannot change the behaviour of another one; therefore masking cannot occur.
- *Linked faults:* These are faults that do influence the behaviour of each other. That means that the behaviour of a certain fault can change the behaviour of another one such that masking can occur. Note that linked faults consist of two or more simple faults.

In this work, single-port, static faults are considered. From here on, the term 'fault' refers to a single-port, static, simple fault and 'linked fault' means single-port, static, linked fault.

The following notations are usually used to describe operations on RAMs :

- $\uparrow$ denotes an up transition due to a certain sensitizing operation.
- $\downarrow$ denotes a down transition due to a certain sensitizing operation.

3. THE LINKED NEIGHBORHOOD PATTERN-SENSITIVE FAULT MODEL

RAM faults can also be divided into *single-cell* and *multi-cell* faults. Single-cell faults consist of FPs involving a single cell, while multi-cell faults consists of FPs involving more than one cell. In this work, we consider a particular class of multi-cell faults (also called *coupling faults*), namely the pattern sensitive faults (PSF). The PSF is a coupling fault, which affects the content of a memory cell (called the *victim cell* or the *base cell*), or the ability to change its content, when other memory cells (called *aggressor cells*) have certain patterns. It is unnecessary and unrealistic to consider all possible patterns of all the memory cells, therefore simplified models of neighborhood pattern sensitive faults (NPSF) were introduced. In these models, the aggressor cells are limited to the physical neighborhood of the victim cell. Depending on the number of aggressor cells, NPSF can be divided into several types, but only two of those are more common: Type-1 NPSF that has four aggressor cells and Type-2 NPSF that has eight aggressor cells as illustrated in Fig. 1a and Fig. 1b, respectively (van de Goor 1991).

Like in the most previous works, in this paper only Type-1 NPSF is studied, because it is more practical for the type of memory we have considered and less concerning when it comes to complexity.

Due to the features of the NPSF model, the general notation for a FP is particularized, thus in the rest of this paper a FP is denoted as $< N\ W\ E\ S; B / Bf >$ (Cheng, Tsai & Wu 2002), where:

- *N*, *W*, *E*, *S* describes the sensitizing value or operation in the aggressor cells (placed as presented in Fig. 1a);
- *B* describes the correct value or transition in the base cell;
- *Bf* shows the faulty value or transition of the base cell.

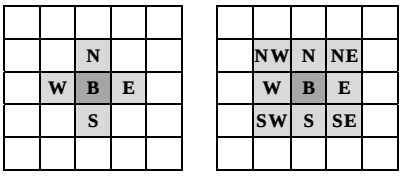
Note that *N*, *W*, *E*, *S*, *B* and $Bf \in \{0, 1, \uparrow, \downarrow\}$.

Depending on the behaviour of the fault, the NPSF can be divided into three classes (van de Goor 1991), namely:

- Static NPSF (SNPSF): the base cell is forced to a certain value when the aggressor cells have a certain pattern. An example of a static NPSF is FP_1=<0100; 0/1>, where the base cell is forced to 1 when the aggressor cells have the pattern 0100.
- Passive NPSF (PNPSF) reflects the impossibility of the base cell to execute a transition due to the appearance of a certain pattern in the aggressor cells. An example of a PNPSFs is FP_2=<1100; $\downarrow$/1>, where the base cell cannot switch from 1 to 0 because the aggressor cells have the pattern 1100.
- Active NPSF (ANPSF): a certain transition in one of the aggressor cells forces the victim cell to change its state when the other aggressor cells (also called *enabling cells*) have a certain pattern. An example of this class of faults is FP_3=<10$\uparrow$0; 0/1>, where a transition in the *E* cell causes the base cell to flip from 0 to 1 when the *N*, *W* and *S* cells have the pattern 100.

The model of NPSF we have considered can be entirely described by the set of FPs presented in Table 1. There are 192 fault primitives: 32 SNPSFs, 32 PNPSFs, and 128 ANPSFs.

The linked neighborhood pattern-sensitive faults are NPSFs that influence the behaviour of each other, such as masking can occur. Therefore, they are more difficult to detect. A linked fault consists of two or more FPs with contrary effects on the same victim (base) cell. For example, take a NPSF fault in which an up transition into cell *W* changes the state of cell *B* from 1 to 0, when the



a *b*

Fig. 1 – Common types of neighborhood pattern sensitive faults:
a – Type-1 NPSF; *b* – Type-2 NPSF.

enabling cells have the pattern 100, whereas an up transition into cell *S* changes the state of cell *B* from 0 to 1, when the enabling cells have the pattern 011. This is a linked fault that can be modeled by two FPs,

FP_1=<1↑00;1/ 0>, and FP_2=<011↑;0/1>.

4. MEMORY TESTS FOR NPSFS

In order to describe the memory tests we have considered for the simulation study, first some notations regarding the march tests are given. Usually, a complete march test is delimited by '{…}' bracket pair, while a march element is delimited by the '(…)' bracket pair. March elements are separated by semicolons, and the operations within a march element are separated by commas. Note that all operations of a march element are performed at a certain address, before proceeding to the next address. The whole memory is checked homogeneously in either one of two orders: ascending address order (⇑) or descending address order (⇓). When the address order is not relevant, the symbol ⇕ is used. Multibackground march tests are march tests that run under several different data backgrounds (Cheng, Tsai & Wu 2001; Yarmolik & Mrozek 2000). In this case, the *w*0 and *r*0 operations are substituted with the *wa* and *ra* operations, respectively, where *a* is the value in the background. Also, *w*1 and *r*1 are substituted with *wb* and *rb*, respectively, where *b* is the complement of *a*.

Table 1. List of NPSF primitives

FFM	Fault primitives	
SNPSF	<*x y z t*; 0 / 1>	*x*, *y*, *z*, *t* ∈ {0, 1}
	<*x y z t*; 1 / 0>	
PNPSF	<*x y z t*; ↑ / 0>	*x*, *y*, *z*, *t* ∈ {0, 1}
	<*x y z t*; ↓ / 1>	
ANPSF	<*x y z* ↑; 0 / 1>	*x*, *y*, *z* ∈ {0, 1}
	<*x y z* ↓; 0 / 1>	
	<*x y z* ↑; 1 / 0>	
	<*x y z* ↓; 1 / 0>	
	<*x y* ↑ *z*; 0 / 1>	
	<*x y* ↓ *z*; 0 / 1>	
	<*x y* ↑ *z*; 1 / 0>	
	<*x y* ↓ *z*; 1 / 0>	
	<*x* ↑ *y z*; 0 / 1>	
	<*x* ↓ *y z*; 0 / 1>	
	<*x* ↑ *y z*; 1 / 0>	
	<*x* ↓ *y z*; 1 / 0>	
	<↑ *x y z*; 0 / 1>	
	<↓ *x y z*; 0 / 1>	
	<↑ *x y z*; 1 / 0>	
	<↓ *x y z*; 1 / 0>	

The most important published test algorithms dedicated to the classical model of NPSF are presented as follows:

a) The test given by Suk and Reddy (Suk & Reddy 1980), *SR* from here on, is a non-march test algorithm that divides the memory cells into two partitions and applies a sequence of transitions to cover all possible victim-aggressor combinations. The length of this test is 165*N*.

b) March-100N: This march memory test, given by (Cheng, Tsai &Wu 2002), uses eight different data backgrounds and for each of them applies the following march sequence: { ⇕ (*wa*); ⇑ (*ra*, *wb*, *wa*); ⇑ (*ra*, *wb*); ⇑ (*rb*, *wa*, *wb*); ⇑ (*rb*, *wa*); ⇕ (*ra*) } .

The data backgrounds used for the memory initialisation (denoted by BG1, BG2, ..., BG8) are presented in Fig. 2.

Additionally, only for BG1 the test applies the march sequence { ⇓ (*ra*, *wb*); ⇓ (*rb*, *wa*)}.

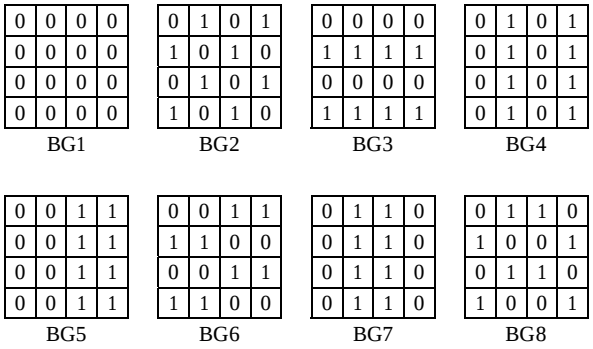


Fig. 2 – The 8 backgrounds for March-100N.

c) CM-79N: This memory test, also given by Cheng, Tsai, and Wu (2002), uses sixteen different data backgrounds and for each of them applies the following march sequence: { ⇕(*wa*); ⇑ (*ra*, *wb*, *rb*, *wa*); ⇕(*ra*) } .

The data backgrounds used for the memory initialisation (denoted by BG1, BG2, ..., BG16) are presented in Fig. 3.

As specified in (Cheng, Thai & Wu 2002), some redundant operations can be removed out of the test. Thus, the initial write operation of the march test is applied only on the cells that must be changed (note that every two successive backgrounds have exactly four different cells, so instead of writing nine cells, the test will write only the four cells that are different). Also, the last read operation skips over the cells that are not changed, because the first read operation for the next background can do that. This happens with every background change. Therefore, the test length (96*N*) is reduced with 15 × (5*N*/9 + 5*N*/9) = 50*N*/3 operations. Consequently, the length of test CM-79N is 79⅓*N*.

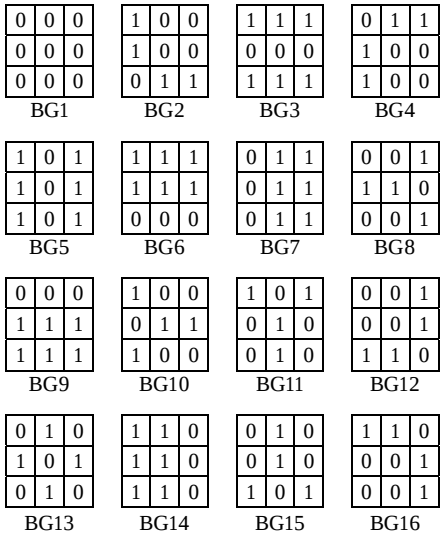


Fig. 3 – The 16 backgrounds for CM-96N march test.

5. FAULT COVERAGE EVALUATION FOR LINKED NPSFS

For the simulation study, the linked faults consisting of two simple faults have been considered. There are 96 NPSFs that flip the base cell from 0 to 1 and 96 that flip it from 1 to 0. Consequently, a total of 96 × 96 = 9216 linked faults have been considered for the coverage evaluation by the memory tests presented in the previous section.

The simulation results on the linked NPSFs show that the tests *SR* and CM-100N are able to cover entirely the model of linked NPSFs. Concerning the CM-79N test, our study demonstrates that a lot of linked NPSFs cannot be detected by this march memory test. The linked faults undetected by CM-79N are presented as follows.

Taking into account that the patterns of the backgrounds used by this test are composed of 3×3 cells, for the simulation study, the memory cells have been divided into nine mutually disjoint subclasses. These are denoted B1, B2, …, B9, depending on their location. Let *r* and *c* be the row address and the column address, respectively, of a memory cell. The cell belongs to a certain subclass according to the following formulas:

- B1 – *c* % 3 = 0 and *r* % 3 = 0
- B2 – *c* % 3 = 0 and *r* % 3 = 1
- B3 – *c* % 3 = 0 and *r* % 3 = 2
- B4 – *c* % 3 = 1 and *r* % 3 = 0
- B5 – *c* % 3 = 1 and *r* % 3 = 1
- B6 – *c* % 3 = 1 and *r* % 3 = 2
- B7 – *c* % 3 = 2 and *r* % 3 = 0
- B8 – *c* % 3 = 2 and *r* % 3 = 1
- B9 – *c* % 3 = 2 and *r* % 3 = 2.

For a memory array with 8 rows and 8 columns, these nine subclasses of cells are illustrated in Fig. 4.

	0	1	2	3	4	5	6	7
0	B1	B4	B7	B1	B4	B7	B1	B4
1	B2	B5	B8	B2	B5	B8	B2	B5
2	B3	B6	B9	B3	B6	B9	B3	B6
3	B1	B4	B7	B1	B4	B7	B1	B4
4	B2	B5	B8	B2	B5	B8	B2	B5
5	B3	B6	B9	B3	B6	B9	B3	B6
6	B1	B4	B7	B1	B4	B7	B1	B4
7	B2	B5	B8	B2	B5	B8	B2	B5

Fig. 4 – The cell subclasses for an 8×8 memory chip array.

Due to the shape and the dimensions of the CM-79N backgrounds, every memory cell that belongs to the same subclass will support the same opperations during the test. Moreover, if two base cells belong to the same subclass, their agressor cells will support the same initializations. Hence, for the simulation study only nine locations (one for each subclass) have been considered for the base cell.

Table 2 presents all the linked NPSFs that are not detected by the test CM-79N. To simplify the writing in Table 2, the '<' and '>' symbols usually used to denote a fault primitive have been neglected.

Table 2. List of undetected linked NPSFs for CM-79N

Sub class	Undetected linked NPSFs			
B1	↑010;1/0 ↓010;0/1	↑111;1/0 ↓111;0/1	↑011;0/1 ↓011;1/0	↑110;0/1 ↓110;1/0
	0↑01;1/0 0↓01;0/1	0↑10;1/0 0↓10;0/1	1↑00;1/0 1↓00;0/1	1↑11;1/0 1↓11;0/1
	0↑00;0/1 0↓00;1/0	0↑11;0/1 0↓11;1/0	1↑01;0/1 1↓01;1/0	1↑10;0/1 1↓10;1/0
	10↑0;1/0 10↓0;0/1	01↑1;0/1 01↓1;1/0	10↑1;0/1 10↓1;1/0	11↑0;0/1 11↓0;1/0
	00↑1;1/0 00↓1;0/1	010↑;1/0 010↓;0/1	100↑;1/0 100↓;0/1	111↑;1/0 111↓;0/1
	000↑;0/1 000↓;1/0	011↑;0/1 011↓;1/0	101↑;0/1 101↓;1/0	110↑;0/1 110↓;1/0
	001↑;1/0 001↓;0/1			
B2	↑000;0/1 ↓000;1/0	↑101;0/1 ↓101;1/0	↑110;0/1 ↓110;1/0	↑111;1/0 ↓111;0/1
	0↑00;0/1 0↓00;1/0	0↑11;0/1 0↓11;1/0	1↑01;0/1 1↓01;1/0	1↑10;0/1 1↓10;1/0
	0↑01;1/0 0↓01;0/1	1↑00;1/0 1↓00;0/1	00↑0;0/1 00↓0;1/0	10↑0;0/1 01↓0;1/0
	10↑0;0/1 010↑;1/0	10↑1;0/1 10↓1;1/0	01↑1;0/1 01↓1;1/0	11↑0;0/1 11↓0;1/0
	00↑1;1/0 00↓1;0/1	01↓0;1/0 101↑;0/1	000↑;0/1 000↓;1/0	101↑;0/1 010↑;1/0
	101↑;0/1 101↓;1/0	110↑;0/1 110↓;1/0	011↑;0/1 011↓;1/0	100↑;1/0 100↓;0/1
	001↑;1/0 001↓;0/1	010↑;1/0 010↓;0/1	111↑;1/0 111↓;0/1	
B3	↑101;0/1 ↓101;1/0	↑010;1/0 ↓010;0/1	↑100;1/0 ↓100;0/1	0↑00;0/1 0↓00;1/0
	0↑11;0/1 0↓11;1/0	1↑01;0/1 1↓01;1/0	1↑10;0/1 1↓10;1/0	0↑01;1/0 0↓01;0/1
	0↑10;1/0 0↓10;0/1	1↑11;1/0 1↓11;0/1	00↑1;1/0 00↓1;0/1	10↑0;1/0 10↓0;0/1
	01↑0;1/0 01↓0;0/1	01↑1;1/0 10↓1;0/1	01↑1;1/0 100↓;0/1	11↑1;1/0 11↓1;0/1
	00↓0;1/0 111↑;1/0	10↓1;0/1 011↓;1/0	000↓;0/1 111↑;1/0	000↑;0/1 000↓;1/0
	101↑;0/1 101↓;1/0	110↑;0/1 110↓;1/0	011↑;0/1 011↓;1/0	100↑;1/0 100↓;0/1
	001↑;1/0 001↓;0/1	010↑;1/0 010↓;0/1	111↑;1/0 111↓;0/1	100↓;0/1 011↓;1/0

Table 2 (continued).
List of undetected linked NPSFs for CM-79N

Sub class	Undetected linked NPSFs			
B4	↑010;1/0 ↓010;0/1	↑100;1/0 ↓100;0/1	0↑00;0/1 0↓00;1/0	0↑11;0/1 0↓11;1/0
	1↑01;0/1 1↓01;1/0	1↑10;0/1 1↓10;1/0	0↑10;1/0 0↓10;0/1	00↑0;0/1 00↓0;1/0
	10↑1;0/1 10↓1;1/0	01↑1;0/1 01↓1;1/0	11↑0;0/1 11↓0;1/0	00↑1;1/0 00↓1;0/1
	10↑0;1/0 10↓0;0/1	01↑0;1/0 01↓0;0/1	11↑1;1/0 11↓1;0/1	000↑;0/1 000↓;1/0
	101↑;0/1 101↓;1/0	110↑;0/1 110↓;1/0	011↑;0/1 011↓;1/0	100↑;1/0 100↓;0/1
	001↑;1/0 001↓;0/1	010↑;1/0 010↓;0/1	111↑;1/0 111↓;0/1	
B5	↑000;0/1 ↓000;1/0	↑110;0/1 ↓110;1/0	↑001;1/0 ↓001;0/1	↑010;1/0 ↓010;0/1
	↑111;1/0 ↓111;0/1	0↑01;1/0 0↓01;0/1	0↑10;1/0 0↓10;0/1	1↑00;1/0 1↓00;0/1
	1↑11;1/0 1↓11;0/1	10↑1;0/1 10↓1;1/0	11↑0;0/1 11↓0;1/0	00↑1;1/0 00↓1;0/1
	10↑0;1/0 10↓0;0/1	11↑1;1/0 11↓1;0/1	000↑;0/1 000↓;1/0	101↑;0/1 101↓;1/0
	110↑;0/1 110↓;1/0	011↑;0/1 011↓;1/0	100↑;1/0 100↓;0/1	001↑;1/0 001↓;0/1
	010↑;1/0 010↓;0/1	111↑;1/0 111↓;0/1	100↓;0/1 110↓;1/0	
B6	↑011;0/1 ↓011;1/0	↑101;0/1 ↓101;1/0	↑010;1/0 ↓010;0/1	↑100;1/0 ↓100;0/1
	0↑11;0/1 0↓11;1/0	1↑01;0/1 1↓01;1/0	1↑10;0/1 1↓10;1/0	0↑01;1/0 0↓01;0/1
	1↑00;1/0 1↓00;0/1	1↑11;1/0 1↓11;0/1	00↑0;0/1 00↓0;1/0	10↑1;0/1 10↓1;1/0
	11↑0;0/1 11↓0;1/0	00↑1;1/0 00↓1;0/1	01↑0;1/0 01↓0;0/1	11↑1;1/0 11↓1;0/1
	000↑;0/1 000↓;1/0	101↑;0/1 101↓;1/0	110↑;0/1 110↓;1/0	011↑;0/1 011↓;1/0
	100↑;1/0 100↓;0/1	001↑;1/0 001↓;0/1	010↑;1/0 010↓;0/1	111↑;1/0 111↓;0/1
B7	↑101;0/1 ↓101;1/0	↑010;1/0 ↓010;0/1	0↑00;0/1 0↓00;1/0	0↑11;0/1 0↓11;1/0
	1↑10;0/1 1↓10;1/0	0↑01;1/0 0↓01;0/1	0↑10;1/0 0↓10;0/1	1↑00;1/0 1↓00;0/1
	1↑11;1/0 1↓11;0/1	10↑1;0/1 10↓1;1/0	00↑1;1/0 00↓1;0/1	01↑0;1/0 01↓0;0/1
	11↑1;1/0 11↓1;0/1	01↓0;0/1 11↓0;1/0	000↑;0/1 000↓;1/0	101↑;0/1 101↓;1/0
	110↑;0/1 110↓;1/0	011↑;0/1 011↓;1/0	100↑;1/0 100↓;0/1	001↑;1/0 001↓;0/1
	010↑;1/0 010↓;0/1	111↑;1/0 111↓;0/1		
B8	↑101;0/1 ↓101;1/0	↑110;0/1 ↓110;1/0	↑001;1/0 ↓001;0/1	↑010;1/0 ↓010;0/1
	↑100;1/0 ↓100;0/1	↑111;1/0 ↓111;0/1	0↑00;0/1 0↓00;1/0	0↑11;0/1 0↓11;1/0
	1↑01;0/1 1↓01;1/0	1↑10;0/1 1↓10;1/0	0↑01;1/0 0↓01;0/1	0↑10;1/0 0↓10;0/1
	1↑00;1/0 1↓00;0/1	00↑1;1/0 00↓1;0/1	10↑0;1/0 10↓0;0/1	11↑1;1/0 11↓1;0/1
	000↑;0/1 000↓;1/0	101↑;0/1 101↓;1/0	110↑;0/1 110↓;1/0	011↑;0/1 011↓;1/0
	100↑;1/0 100↓;0/1	001↑;1/0 001↓;0/1	010↑;1/0 010↓;0/1	111↑;1/0 111↓;0/1
B9	↑000;0/1 ↓000;1/0	↑011;0/1 ↓011;1/0	↑110;0/1 ↓110;1/0	↑111;1/0 ↓111;0/1
	0↑00;0/1 0↓00;1/0	0↑11;0/1 0↓11;1/0	1↑01;0/1 1↓01;1/0	1↑10;0/1 1↓10;1/0
	0↑01;1/0 0↓01;0/1	0↑10;1/0 0↓10;0/1	1↑00;1/0 1↓00;0/1	1↑11;1/0 1↓11;0/1
	00↑0;0/1 00↓0;1/0	10↑1;0/1 10↓1;1/0	01↑1;0/1 01↓1;1/0	11↑0;0/1 11↓0;1/0
	00↑1;1/0 00↓1;0/1	10↑0;1/0 10↓0;0/1	000↑;0/1 000↓;1/0	101↑;0/1 101↓;1/0
	110↑;0/1 110↓;1/0	011↑;0/1 011↓;1/0	100↑;1/0 100↓;0/1	001↑;1/0 001↓;0/1
	010↑;1/0 010↓;0/1	111↑;1/0 111↓;0/1		

6. FAULT COVERAGE EVALUATION FOR LFS

Because of the fact that single-cell and two-cell coupling faults are the vast majority of faults observed in practice [Al-Ars & van de Goor 2000], the linked faults based on a combination of these kinds of FPs (LFs) are the most frequent linked faults in random-access memories. Therefore, the coverage of LFs by the tests introduced in section 4 has also been evaluated.

As presented in [Hamdioui 2004], depending on the number and nature of the faults involved, LFs can be categorized into three types, as illustrated in Fig. 5. These LFs are:

1) LFs involving a single cell (LF1s), that are based on a combination of two single-cell FPs, both with the same aggressor cell which is also the victim cell.

2) LFs involving two cells (LF2s), based on a combination of two two-cell FPs, or on a combination of a single-cell FP and a two-cell FP. Consequently, they are divided into three subtypes:

a) $LF2_{aa}$, is based on a combination of two two-cell LFs, both having the same aggressor cell and the same victim cell,

b) $LF2_{av}$, based on a combination of one two-cell FP (sensitized first) and one single-cell FP,

c) $LF2_{va}$, similar to $LF2_{av}$, is based on a combination of one two-cell FP and one single-cell FP (sensitized first).

3) LFs involving three cells (LF3s) are based on a combination of two two-cell FPs with the same victim cell, but different aggressor cells.

Not all combinations between these faults form a valid LF, so for every of the types defined above, the valid combinations of FPs were selected [Hamdioui 2004]. There are 12 LF1s, 24 $LF2_{aa}$s, 16 $LF2_{av}$s, 18 $LF2_{va}$s and 24 LF3s.

Because of the nature of the tests considered for the study, the memory cells analyzed in our simulation are different from one test to another. For example, regarding CM-79N, 9 subclasses of memory cells were defined (see Section 5), and every memory cell that belongs to the same subclass supports the same operations during the test. Thus, for the simulation study there have been considered 9 cells for the LF1s (one from every subclass), 81 combinations for LF2s (all possible combinations of two cells from the 9 subclasses), and 729 combinations for LF3s (all possible combinations of three cells from the 9 subclasses).

The simulation results are presented in Table 3. Note that March-100N has the best fault coverage of LFs (91,7%), because two march elements ($\Downarrow(ra, wb)$, and $\Downarrow(rb, wa)$) are applied in a descending address order. This sensitizes coupling faults in which the aggressor cell is localized at a higher address than the victim cell, so more two-cell coupling faults are detected.

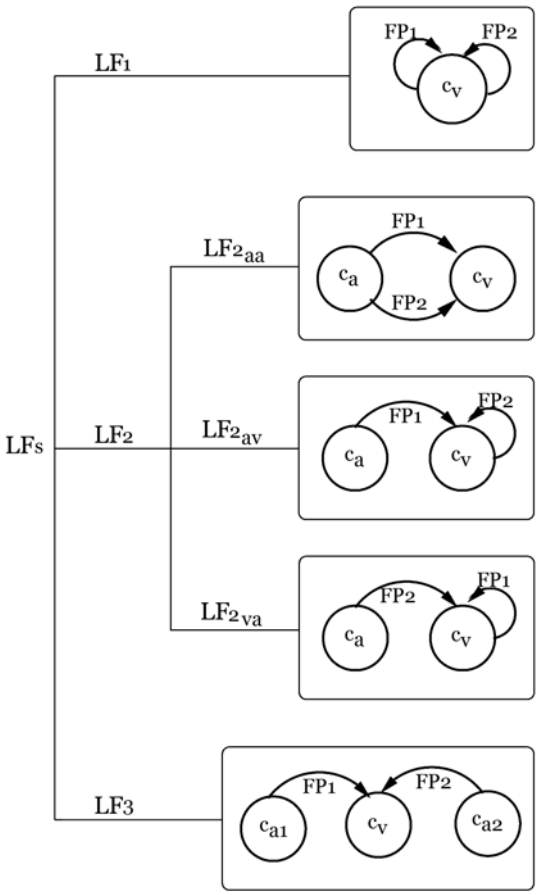


Fig. 5 – Classification of LFs.

Table 3
Fault coverage of NPSF dedicated tests on LFs

Memory test	Length	Fault coverage (%)			
		LF1	LF2	LF3	Total
SR	165,5*N*	83,33	70,43	68,75	69,02
March-100N	100*N*	100	91,74	91,71	91,72
CM-79N	79,33*N*	83,33	80,54	83,15	82,79

7. CONCLUSIONS

Many of the works dedicated to the NPSF model take into account only the simple fault model. However, a more realistic model that needs to be considered is the linked NPSF. This paper presents the fault coverage of the linked faults by the most important published tests dedicated to the NPSF model. The results show that only the longer algorithms are able to cover the whole model of linked NPSFs. Also, the simulation study leads to a new opportunity: to create a new memory test, shorter than March-100N, able to detect all NPSFs as well as linked NPSFs. This memory test will be the subject of an upcoming paper.

REFERENCES

Al-Ars, Z., van de Goor, A.J. (2001). Static and Dynamic behaviour of memory cell array opens and shorts in embedded DRAMs, *Proc. Design Automation Test Eur.*, pp. 496-503.

Adams, R.D. (2003). *High performance memory testing: design principles, fault modelling and self-test*. Kluwer Academic Publishers, Norwell, USA.

Cheng, K.L., Tsai M.F., and Wu C.W. (2001). Efficient neighborhood pattern-sensitive fault test algorithms for semiconductor memories. *Proc. IEEE VLSI Test Symp (VTS)*, Maria Del Rey, CA, pp. 225-237.

Cheng, K.L., Tsai M.F., and Wu C.W. (2002). Neighborhood pattern-sensitive fault testing and diagnostics for random-access memories. ", *IEEE Trans. On CAD*, vol. 21, no. 11, pp. 1328-1336.

Cockburn, B.F. (1995). Deterministic tests for detecting scrambled pattern-sensitive faults in RAMs. *Proc. IEEE Int. Workshop Memory Technology, Design and Testing (MTDT)*, San Jose, CA, pp. 117-122.

Hamdioui, S., van de Goor, A.J., and Rodgers, M. (2002). March SS: A test for all static simple RAM faults. *Proc. of IEEE Int'l Workshop on Memory Technology, Design and Testing*, Isle of Bendor, France, pp. 95-10.

Hamdioui, S. (2004). *Testing static random access memories: defects, fault models and test patterns*. Kluwer Academic Publishers, Norwell, USA.

Hamdioui, S., Al-Ars, Z., van de Goor, A.J. (2004). Linked faults in Random Access Memories: concept, fault models, test algorithms, and Industrial Results, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 23, no. 5, pp. 737-757.

Hayes, J.P. (1980). Testing memories for single-cell pattern-sensitive fault, *IEEE Trans. Comput.*, vol. 29, pp. 249-254.

Julie, R.R., Wan Zuha, W.H., Sidek, R.M. (2008) 12N test procedure for NPSF testing and diagnosis for SRAMs, *Proc. IEEE Int. Conf. on Semiconductor Electronics*, pp. 430-435.

Kang, D.C., Cho, S.B. (2000). An efficient build-in self-test algorithm for neighborhood pattern sensitive faults in high-density memories, *Proc. 4th Korea-Russia Int. Symp. Science and Technology*, vol. 2, pp. 218-223.

Suk, D.S., Reddy, M. (1980). Test procedures for a class of pattern-sensitive faults in semiconductor random-access memories, *IEEE Trans. Comput.*, vol. 29, pp. 419–429.

Van de Goor, A.J. (1991). *Testing semiconductor memories: theory and practice*, Wiley, Chichester, U.K.

Yarmolik, V., Mrozek, I. (2000). MultiBackground memory testing, in *Proc. MIXDES 14th International Conference*, Ciechocinek, Poland.