DETECTING FAULTS IN THE PERIPHERAL CIRCUITS AND AN EVALUATION OF SRAM TESTS

Ad J. van de Goor[1]

[1]Delft University of Technology
Computer Engineering Laboratory
Mekelweg 4, 2628 CD Delft, The Netherlands
E-mail: A.J.vandeGoor@ewi.tudelft.nl

Said Hamdioui[1,2]

[2]Philips Semiconductor R&D
860 Rue Jean Monnet, 38920 Crolles, France
E-mail: S.Hamdioui@ewi.tudelft.nl
E-mail: Said.Hamdioui@philips.com

Rob Wadsworth[3]

[3]ST Microlelctronics
1310 Electronics Drive
Carrollton, TX 75006, USA
E-mail: Rob.Wadsworth@st.com

Abstract:

Very little has been published on faults in the memory peripheral circuits, denoted as 'PFs'. This paper shows that PFs will be detected by march tests, provided that they satisfy particular properties, expressed in terms of properties of the algorithm, and of properties of the algorithm stress. The latter consists of the used data backgrounds and addressing directions. The detection capabilities of a set of well-know march algorithms will be established for the PFs. In addition, industrial results from applying this set of tests to a large number of 0.13 micron 512 Kbyte SRAM chips, combined with a variety of stress conditions, will be presented.

Keywords: *March tests, data-backgrounds, address directions, fault coverage, peripheral circuit faults.*

1 Introduction

With increasing memory densities and clock speeds, the potential for faults in the peripheral circuits, referred to as *PFs*, is on the rise. This is due to the increased sensitivity to open defects (aggravated by the use of copper wiring) and to capacitive coupling [1, 2, 3]; while the circuits are more time-critical.

Very little has been published on fault models and algorithms for PFs [4]. Much has been published on fault models and algorithms for detecting faults in the address decoders, denoted as *AFs*, and in the memory cell array, denoted as *MFs* [5, 6, 7, 8, 9, 10, 11, 12]. The question to be answered is: How capable are the latter algorithms for detecting faults in commercial SRAMs, and for detecting PFs?

This paper establishes fault models for peripheral memory circuits and derives a set of easy to use criteria march tests have to satisfy in order to detect PFs; similar to the detection criteria used for AFs. The newly established criteria will be applied to a set of well-known march tests to show that the statement [8, 10] "all faults in the peripheral circuits are covered with tests for memory cell array faults"

is not (always) correct.

Several papers have been published on the industrial evaluation of tests applied to DRAMs and SRAMs [13, 14, 15, 16]; however, the tests used in those experiments were of that generation. This paper presents the results of applying a large set of modern tests to a large number of SRAM chips, using many different stresses, such as supply voltages, data backgrounds, etc. The results show that the high voltage and high speed stresses are most effective; while March G and March SL have the highest fault coverage, followed by the very time-efficient tests March U and March LR. The Scan test has the property that it detects faults which are very different from the other tests.

The paper is organized as follows: Section 2 describes the notation for march tests and the stresses of interest to this work. Section 3 gives a classification of the traditional memory faults. Section 4 presents the new peripheral circuit faults, together with requirements march tests have to satisfy in order to detect those faults. Section 5 analyzes a set of well-known march tests, including several recently published tests, for their capability of detecting faults in the peripheral circuits, and shows the results of their application to a large number of SRAM chips. Section 6 gives the conclusions.

2 Memory test algorithms and stresses

A *test* consists of a *Base Test 'BT'*, applied using a particular *Stress Combination 'SC'*. A BT is a test algorithm, such as, e.g., MATS+ [5]. An SC consists of a combination of values for the different *stresses*; e.g., $V_{DD} = 1.8V, Temp = 70^0C$, etc.

Because most BTs have a march test format, below the notation of march tests will be given, followed by a description of the stresses of interest to this paper.

2.1 Notation of march tests

A *march test* is a sequence of march elements. A *march element (ME)* consists of a sequence of operations applied to every cell (n is the number of cells in the memory), in either one of two *Address Orders 'AOs'*: *Increasing* ($\Uparrow$) *AO*, from cell 0 to cell $n - 1$, or a *Decreasing* ($\Downarrow$) *AO*, from cell $n - 1$ to cell 0. When the AO is irrelevant the symbol '$\Updownarrow$' is used.

Example: $\{\Updownarrow(w0); \Uparrow(r0, w1); \Downarrow(r1, w0)\}$ is the MATS+ test [5]. It consists of three march elements: M0, M1 and M2. M1: $\Uparrow(r0, w1)$ means 'for $i = 0$ to $n - 1$ do {read $A[i]$ with expected value 0; $A[i] := 1$}'.

2.2 Stresses combinations

When testing, each BT is applied using several *stresses*. The stresses can be divided into algorithm stresses and non-algorithm stresses.

A *non-algorithm stress*, also referred to as an *environmental stress*, specifies the environmental values, such as the supply voltage, the temperature, the timing (the clock frequency), etc.; they are effective during the application of the test.

An *algorithm stress* specifies the way the test is performed, and therefore it influences the sequence and/or the type of the memory operations. The most known algorithm stresses are the address direction and data-background.

Address Direction 'AD' is the addressing extension of the one-dimensional address order 'AO' to the two dimensional space of the memory cell array. A real memory consists of a number or rows and columns (and thus also of a number of diagonals). The AD specifies the direction (i.e., rows, columns, or diagonals) in which the address sequence has to be performed. The commonly used ADs in the industry are described below.

1. **Fast X (fX):** With fX addressing, each address increment or decrement operation causes an adjacent physical row to be accessed.

2. **Fast Y (fY):** With fY addressing, each address increment or decrement operation causes an adjacent physical column to be accessed.

3. **Fast D (fD):** With fD addressing, each address increment or decrement operation causes an adjacent physical diagonal to be accessed. Fast D is used less frequently in industry.

Data Background 'DB' is the pattern of ones and zeros as seen in an array of memory cells. The most common types of DBs are:

Solid (sDB): all 0s (i.e., 0000.../0000...) or all 1s

Table 1. Single and two-cell faults

Single-cell faults		Two-cell faults	
Name	FFM	Name	FFM
State Fault	SF	State CF	CFst
Transition Fault	TF	Transition CF	CFtr
Write Destructive Fault	WDF	Write Destructive CF	CFwd
Read Destructive Fault	RDF	Read Destructive CF	CFrd
Deceptive RDF	DRDF	Deceptive CFrd	CFdr
Incorrect Read Fault	IRF	Incorrect Read CF	CFir
		Disturb CF	CFds

Checkerboard (bDB): 0101.../1010.../0101.../1010...
Column Stripe (cDB): 0101.../0101.../0101.../0101...
Row Stripe (rDB): 0000.../1111.../0000.../1111...

3 Classification of memory faults

Faults in memory systems are divided into *Memory cell array Faults 'MFs'*, *Address decoder Faults 'AFs'* and *Peripheral circuit Faults 'PFs'*; the latter consist of faults in the write drivers, precharge circuits, sense amplifies, etc. The first two classes of faults are traditional; they have been included for completeness.

3.1 Memory cell array Faults 'MFs'

Memory cell array faults can be divided into single-cell faults and two-cell (i.e., coupling) faults [10, 12, 17, 24].

Single-cell faults: Single-cell faults are faults involving a single cell; i.e., the cell used to sensitize the fault is the same as where the fault appears. *Functional Fault Models (FFMs)*, such as the Transition Fault (TF) and the Read Destructive Fault (RDF), belong to this class of faults. Table 1, left part, shows the single-cell FFMs.

Coupling faults: *Coupling Faults (CFs)* are faults involving two cells. The state of, or an operation applied to, one cell (called the aggressor cell), influences the state of, or the operation applied to, the victim cell. Table 1, right part, shows the two-cell FFMs.

3.2 Address decoder Faults 'AFs'

The traditional Address decoder Faults 'AFs' [10] are described below, together with their detection condition.

Traditional AFs: The following types of AFs have traditionally been the faults considered to occur in address decoders:

• **AFna:** An address does not access its cell.
• **AFmc:** An address uniquely accesses multiple cells; i.e., this is the *only* address accessing those cells.

- **AFma**: A cell is uniquely accessed by multiple addresses; i.e., these addresses *only* access that cell.
- **AFoc**: An address additionally accesses other cells.

Detection condition for traditional AFs: Any march test will detect AFna through AFoc if it satisfies *Condition AF* for $h \geq 1$ [10, 18]. It consists of the following two march elements (Note: the suffix 'u' denotes *up* for the ⇑ AO, the suffix 'd' denotes *down* for the ⇓ AO; '...' means any number of read (r) or write (w) operations, $\overline{x}$ means NOT x, and $[, r\overline{x}]^h$ ($[, rx]^h$) means h (from h*ammer*) $r\overline{x}$ (rx) operations; $h \geq 0$):

AFh-u: $\Uparrow(rx, ..., w\overline{x}[, r\overline{x}]^h)$; $x \in \{0, 1\}$
AFh-d: $\Downarrow(r\overline{x}, ..., wx[, rx]^h)$; $x \in \{0, 1\}$

4 Peripheral circuit Faults 'PFs'

Naturally, the peripheral circuits of a memory also can be faulty. The faulty behavior will manifest itself as a speed related fault, or as a fault due to excessive leakage. Speed related faults can occur in the write drivers, the sense amplifiers, and in the precharge circuits; the leakage related fault is due to leaky pass transistors [10, 19].

Below, the reason for the faulty behavior of each peripheral circuit type is given, together with the requirement for detecting that faulty behavior. However, the properties of such requirements will be discussed first.

4.1 Detection requirement properties

The requirements for detecting faults in the peripheral circuits will be expressed in two aspects:
   A. *March element properties*, and
   B. *Algorithm stresses*.

A. March element properties

March element (ME) properties consist of the required ME *type*, and the required *Read-Write Sequence 'RWS'*. The MEs have two orthogonal properties; see Table 2 (in the table O denotes any operation, $O \in \{r, w\}$):

1. **Type**: The type of a ME, denoted as the 'MEt', can be marching (denoted as 'MEma') or walking (denoted as 'MEwa').

   A ME of type ME*max* has the property that before the application of an ME*max* the cell is in state $\overline{x}$, and upon completion of the application of the ME*max* the cell is in state x.

   A ME of type ME*wax* has the property that before the application of an ME*wax* the cell is in state $\overline{x}$, and upon completion of the application of the ME*wax* the cell is

Table 2. Taxonomy of March Elements 'MEs'

RWS	ME type; $x \in \{0, 1\}$	
	MEma$x$	MEwa$x$
RaW	$\Updownarrow(r\overline{x}, ..., wx)$	$\Updownarrow(r\overline{x}, ..., Ox, ..., w\overline{x})$
RaR	$\Updownarrow(r\overline{x}, ..., wx, rx)$	$\Updownarrow(r\overline{x}, ..., Ox, ..., r\overline{x})$
WaW	$\Updownarrow(w\overline{x}, ..., wx)$	$\Updownarrow(w\overline{x}, ..., Ox, ..., w\overline{x})$
WaR	$\Updownarrow(w\overline{x}, ..., wx, rx)$	$\Updownarrow(w\overline{x}, ..., Ox, ..., r\overline{x})$

again in state $\overline{x}$; however, during the application of the MEwa$x$ the cell has been changed to in state x.

2. **Read Write Sequence 'RWS'**: The RWS is defined by the last operation applied to a given address, together with the first operation applied to the next address.

Examples: the ME '$\Uparrow(r0, w1, r1, w1)$' applies 4 operations to each cell; the last operation to a cell is the $w1$ operation, which is followed by a $r0$ operation applied to the next cell; therefore this is a ME of type MEma1, which applies the *Read-after-Write 'RaW'* RWS.

The MEma$x$ '$\Updownarrow(r\overline{x}, ..., wx, rx)$', see Table 2, performs the *Read-after-Read 'RaR'* RWS.

The ME '$\Uparrow(r0, w1, w0)$' is of type MEwa1, and applies a RaW RWS. It has the property that during the application of the MEwa1 to a 4-bit memory, a '1' walks through the memory, as follows: {0000=initial state, 1000=during the application of the ME to cell-0, 0100=during the application of the ME to cell-1, 0010, 0001, 0000=final state}.

B. Algorithm stresses

Algorithm stresses consist of the required address direction 'AD' (e.g., fX or fY) and the used data background 'DB' (e.g., sDB or cDB).

4.2 PFs and their detection conditions

The PFs consist of the following fault models:

1. **Slow Write Driver Fault (SWDF)**.
2. **Slow Sense Amplifier Fault (SSAF)**.
3. **Slow PRecharge circuit Fault (SPRF)** [4].
4. **Bit Line Imbalance Fault (BLIF)** [10, 19].

Below, the above PFs will be discussed in detail, together with their detection conditions. Table 3 summarizes the march element requirements for detecting each of the above faults; e.g., to detect 'SSAF' fault, a test has to perform a RaW MEma$x$ or a RaW MEwa$x$.

Table 4 shows the most efficient test for each of the PFs, together with the properties of that test; the *Test Length 'TL'*,

which is the total number of read and write operations required for performing the test, varies from $4n$ for March WDm (for the test for SWDF) to $8n$ for March WDw and March BLI (for BLIF). The SWDF, the SSAF and the SPRF are detectable by two efficient tests; one based on MEma (therefore e.g., the name of the test 'March SAm' has the suffix **m**), and another based on the use of MEwa, the name of that test has the suffix **w**.

Table 3. ME requirements for detecting PFs

ME properties		Peripheral circuit fault			
RWS	MEt	SWDF	SSAF	SPRF	BLIF
RaW	MEmax	-	+	+	-
RaR	MEmax	-	-	-	-
WaW	MEmax	+	-	-	-
WaR	MEmax	-	-	-	-
RaW	MEwax	-	+	+	-
RaR	MEwax	-	-	-	-
WaW	MEwax	+	-	-	+
WaR	MEwax	-	-	-	-
Note: The RWSs have to be applied using **Fast-X addressing**, and for both data values, because of asymmetric sensitivities					

SWDF: Slow Write Driver Fault

The Write driver may be too slow due to a defect in the driver circuit and/or due to resistive defects (such as partial open vias) in its path to the to-be-written cells. The result will be that the differential voltage on the bit lines (BLs) during the write operation is reduced. This may cause the cell not to be written. The SWDF will be sensitized by a MEma type ME, using a WaW RWS (see Table 3) with the fX AD and with the bDB or the rDB, see Table 4.

Explanation: the operation 'wx' of the WaW MEma, see Table 2, forces the BLs into one state, while the immediately following '$w\overline{x}$' operation, applied to the next cell in the same column (i.e., to the same BLs), using the fX AD, has to force the BLs into the opposite state. This is the worst stress for the SWDF.

The ME which is the most effective in sensitizing the SWDF is the WaW MEmax '$\updownarrow(w\overline{x}, wx)$' (see Table 2), because the two write operations are performed back-to-back; i.e., no other operations in between the wx followed with the $w\overline{x}$ operation occur. A possible 'in between operation' could reduce the effect of the first sensitizing wx operation. The test has to be repeated for $x = 0$ and $x = 1$, because of possible asymmetric sensitivities for the '0' and '1' value of the cell.

Table 4 shows the most efficient test using MEma for the SWDF; in the table, $_x\updownarrow$ denotes the AO $\updownarrow$ using fX AD. The MEma $_x\updownarrow(wD)$, where $D = bDB$ or $D = rDB$, writes a sequence of '0, 1, 0, 1, 0, 1, ...' values using fX AD, and therefore equivalent to WaW RWS. The fault is detected with $_x\updownarrow(rD)$.

A variation of this test, based on the use of the MEwa, can be designed: it uses WaW MEwa type '$\updownarrow(wx, w\overline{x}, r\overline{x}, wx)$' (see Table 2), together with the algorithm stresses fX addressing and the sDB or the cDB. The MEwa '$\updownarrow(wx, w\overline{x}, r\overline{x}, wx)$' can be divided into two MEs $\updownarrow(wx)$ and $\updownarrow(w\overline{x}, r\overline{x}, wx)$ without having any impact on the ME type. The resulting test based on WaW MEwa for SWDF is given in Table 4. The first ME in the test '$\updownarrow(wD)$' initializes the columns; the next ME '$_x\updownarrow(w\overline{D}, r\overline{D}, wD)$' (where $D\in\{sDB, cDB\}$) performs the required WaW sequences such that the w operations, with opposite data value, are applied back-to-back to sensitize the fault. That fault is detected by the read operation.

Note: The SWDF would also be sensitized by any ME of the form '$_x\updownarrow(..., wD, w\overline{D}, r\overline{D}, ...)$', where $D\in\{bDB, rDB\}$. However, then the sensitization would take place by applying the operation sequence '$wx, w\overline{x}$' to the same cell, such that the delay of the row decoder, due to switching from a cell to the next cell, is not included; this will be a less stressy sensitization.

SSAF: Slow Sense Amplifier Fault

The *Sense Amplifier 'SA'* may be too slow, or is asymmetric (because of some offset voltage) due to a defect in the SA circuits and/or due to resistive defects in the path from the cell to the SA. This may cause read operations to produce incorrect results. The sensitization and detection of the SSAF requires the application of the MEma, using the RaW RWS, (i.e., the ME '$\updownarrow(r\overline{x}, ..., wx)$'); for $x = 0$ and $x = 1$, together with the algorithm stresses fX addressing and the sDB or cDB. Hence the MEma $_x\uparrow\uparrow(rD, w\overline{D})$ and the MEma $_x\uparrow\uparrow(r\overline{D}, wD)$, where $D \in \{sDB, cDB\}$ are required for the detection of SSAF; see Table 4.

E.g., the operation '$w0$' brings the BLs in the worst case state for the following '$r1$' operation, applied to the next cell in the same column; hence the use of fX addressing and the sDB or the cDB.

A variation of this test, based on the use of the MEwa, can be designed: it uses MEwa type MEs with the RaW RWS, together with the algorithm stresses fX addressing and the bDB or the rDB; see Table 4. The first ME in the test '$\updownarrow(wD)$' writes the sequence '0,1,0,1,...'; the next ME '$_x\updownarrow(rD, wD)$' (where $D\in\{bDB, rDB\}$) performs the required RaW sequence such that the r operation and the w operation operate on opposite data values. E.g., a read 1 operation is applied to the next cell c_2 after a write 0 operation is applied to cell c_1, which is adjacent and belongs to the same column as cell c_2.

Note: The SPRF would also be sensitized and detected with any ME of the form '$\updownarrow(..., rx, w\overline{x}, r\overline{x}, ...)$'. However, then the sensitization would take place by applying the operation sequence '$rx, r\overline{x}$' to the same cell, such that the delay

Table 4. Efficient tests for PFs

#	Name	Fault	TL	MEt	RWS	DB*	Test; $D = DB$
1	March WDm	SWDF	$4n$	MEma	WaW	bDB, rDB	$\{x\Updownarrow(wD);x\Updownarrow(rD);x\Updownarrow(w\overline{D});x\Updownarrow(r\overline{D})\}$
2	March WDw	SWDF	$8n$	MEwa	WaW	sDB, cDB	$\{\Updownarrow(wD);x\Updownarrow(w\overline{D},r\overline{D},wD);\Updownarrow(w\overline{D});x\Updownarrow(wD,rD,w\overline{D})\}$
3	March SAm	SSAF	$5n$	MEma	RaW	sDB, cDB	$\{\Updownarrow(wD);x\Updownarrow(rD,w\overline{D});x\Updownarrow(r\overline{D},wD)\}$
4	March SAw	SSAF	$6n$	MEwa	RaW	bDB, rDB	$\{\Updownarrow(wD);x\Updownarrow(rD,w\overline{D});x\Updownarrow(w\overline{D});x\Updownarrow(r\overline{D},wD)\}$
5	March PRm	SPRF	$5n$	MEma	RaW	sDB, cDB	$\{\Updownarrow(wD);x\Updownarrow(rD,w\overline{D});x\Updownarrow(r\overline{D},wD)\}$
6	March PRw	SPRF	$5n$	MEwa	RaW	bDB, rDB	$\{\Updownarrow(wD);x\Updownarrow(rD,wD);x\Updownarrow(w\overline{D});x\Updownarrow(r\overline{D},w\overline{D})\}$
7	March BLI	BLIF	$8n$	MEwa	WaW	sDB, cDB	$\{\Updownarrow(wD);x\Updownarrow(w\overline{D},r\overline{D},wD);\Updownarrow(w\overline{D});x\Updownarrow(wD,rD,w\overline{D})\}$

*: The notation 'bDB, rDB' means the use of bDB or rDB

of the row decoder is not included; this will be a less stressy sensitization.

SPRF: Slow PRecharge circuit Fault

The *Prechage Circuit 'PC'* may be too slow, or it may not precharge both BLs to the same voltage level, due to defects in the PC circuitry and/or due to resistive defects in the BLs [4]. The result may be that especially read operations will produce incorrect results, because they are most sensitive to BL voltage offset errors.

The sensitization and detection of the SPRF require the application of the MEma '$\Updownarrow(r\overline{x},...,wx)$'; for $x = 0$ and $x = 1$, together with fX addressing and the sDB or the cDB. Hence the MEma $x\Uparrow(rD,w\overline{D})$ and the MEma $x\Uparrow(r\overline{D},wD)$, where $D\in\{sDB,cDB\}$ are required for the detection of SPRF; see Table 4.

A variation of this test, based on the use of the MEwa, can be designed: it uses MEwa type MEs with the RaW RWS, together with the algorithm stresses fX addressing and the bDB or the rDB; see Table 4. The explanation is similar to that given for the detection condition of SSAF. E.g., the operation '$w0$' brings the BLs in the worst case state for the following '$r1$' operation, applied to the next cell in the same column.

Note: This fault would also be sensitized and detected with any ME of the form '$\Updownarrow(...,rx,w\overline{x},r\overline{x},...)$'. However, then the sensitization would take place by applying the operation sequence '$rx,r\overline{x}$' to the same cell, such that the delay of the row decoder is not included; this will be a less stressy sensitization. Alternatively, the SPRF would be sensitized and detected with a RaR RWS of the form '$\Updownarrow(r\overline{x},...,rx)$'; however, the stress caused by the 'rx' operation will be much less than that of the 'wx' operation of the RaW RWS.

It is important to note here that the requirements for detecting SPRF are the same as those required for detecting SSAF. Therefore they can be detected using the same tests; see Table 4, where one can clearly see that March SAm is the same as March PRm, and March SAw is the same as March PRw.

BLIF: Bit Line Imbalance Fault

With decreasing feature widths, the transistors increasingly draw more current in the off-state. This also applies to the pass transistors, which may impact the possibility of reading the correct value of a cell (write operations are less sensitive). The worst case situation occurs when a 'rx' operation is applied to a cell, which is the only cell in that column containing the value 'x'. Then, while reading the cell with the 'x' value, the leaky pass transistors of the cells with the '$\overline{x}$' value will neutralize the read result, to the extent that the incorrect '$\overline{x}$' value may be read. The sensitization and detection of the BLIF requires the application of the WaW MEwa '$\Updownarrow(wx,w\overline{x},r\overline{x},wx)$' (see Table 2); for $x = 0$ and $x = 1$, together with fX addressing and the sDB or the cDB. The MEwa '$\Updownarrow(wx,w\overline{x},r\overline{x},wx)$' can be divided into two MEs $\Updownarrow(wx)$ and $\Updownarrow(w\overline{x},r\overline{x},wx)$ without impacting the MEwa type. The resulting test for BLIF based on WaW MEw is given in Table 4.

Explanation: e.g., the MEwa operation '$w1$' brings a single cell in state '1', after which it is read; while all other cells in the same column contain the value '0'.

Note that the required WaW MEwa for BLIF is the same as that required for SWDF based on MEwa; see Table 4.

Minimal test set for PFs

Based on the above, one can conclude that in order to detect all targeted PFs at least two march tests have to be used:

- March WDw and March SAm, or
- March WDw and March SAw.

Using the first two tests is recommended since they combine a test based on MEwa and a test based on MEma; each test has to be used with differen DB; e.g., one with sDB and one with cDB. From industrial point of view, it has been shown that using different DBs has a large impact on the fault coverage and can detect some non-modeled (non-known) faults [14, 15, 16]. The minimal test set for PFs is given in Table 5; the tests are renamed as March BLI-WDw (for BLIF and SWDF) and March SAPRm and March SAPRw (for SSAF and SPRF).

Table 5. Minimal test set for PFs

#	Name	TL	DB	Description; D=DB	FC
1	March BLIWDw	8n	sBd or cDB	$\{\Updownarrow(wD);_x\Updownarrow(w\bar{D},r\bar{D},wD);\Updownarrow(w\bar{D});_x\Updownarrow(wD,rD,w\bar{D})\}$	BLIF, SWDF
2*	March SAPRm	5n	sDB or cDB	$\{\Updownarrow(wD);_x\Updownarrow(rD,w\bar{D});_x\Updownarrow(r\bar{D},wD)\}$	SSAF, SPRF
3*	March SAPRw	6n	bDB or rDB	$\{\Updownarrow(wD);_x\Updownarrow(rD,wD);_x\Updownarrow(w\bar{D});_x\Updownarrow(r\bar{D},w\bar{D})\}$	SSAF, SPRF

*: Only test 3 or test 2 has to be used with test 1

Table 6. Algorithms used in the experiment

#	BT	TL	Algorithm, also named BT
1	Scan [8]	4n	{⇑(w0); ⇑(r0); ⇑(w1); ⇑(r1)}
2	MATS+ [5]	5n	{⇑(w0); ⇑(r0,w1); ⇓(r1,w0)}
3	MATS++ [10]	6n	{⇕(w0); ⇑(r0,w1); ⇓(r1,w0,r0)}
4	March B [6]	17n	{⇕(w0); ⇑(r0,w1,r1,w0,r0,w1); ⇑(r1,w0,w1); ⇓(r1,w0,w1,w0); ⇓(r0,w1,w0)}
5	March C- [10]	10n	{⇕(w0); ⇑(r0,w1); ⇑(r1,w0); ⇓(r0,w1); ⇓(r1,w0); ⇕(r0)}
6	March G [20]	*	{⇕(w0); ⇑(r0,w1,r1,w0,r0,w1); ⇑(r1,w0,w1); ⇓(r1,w0,w1,w0); ⇓(r0,w1,w0); Del; ⇕(r0,w1,r1); Del; ⇕(r1,w0,r0)}
7	March LA [21]	22n	{⇕(w0); ⇑(r0,w1,w0,w1,r1); ⇑(r1,w0,w1,w0,r0); ⇓(r0,w1,w0,w1,r1); ⇓(r1,w0,w1,w0,r0); ⇓(r0)}
8	March LR [22]	14n	{⇕(w0); ⇓(r0,w1); ⇑(r1,w0,r0,w1); ⇑(r1,w0); ⇑(r0,w1,r1,w0); ⇓(r0)}
9	March RAW[23]	26n	{⇕(w0); ⇑(r0,w0,r0,r0,w1,r1); ⇑(r1,w1,r1,r1,w0,r0); ⇓(r0,w0,r0,r0,w1,r1); ⇓(r1,w1,r1,r1,w0,r0); ⇕(r0)}
10	March SR [12]	14n	{⇓(w0); ⇑(r0,w1,r1,w0); ⇑(r0,r0); ⇑(w1); ⇓(r1,w0,r0,w1); ⇓(r1,r1)}
11	March SS [12, 24]	22n	{⇕(w0); ⇑(r0,r0,w0,r0,w1); ⇑(r1,r1,w1,r1,w0); ⇓(r0,r0,w0,r0,w1); ⇓(r1,r1,w1,r1,w0); ⇕(r0)}
12	March SL [25]	41n	{⇕(w0); ⇑(r0,r0,w1,w1,r1,r1,w0,w0,r0,w1); ⇑(r1,r1,w0,w0,r0,r0,w1,w1,r1,w0); ⇓(r0,r0,w1,w1,r1,r1,w0,w0,r0,w1); ⇓(r1,r1,w0,w0,r0,r0,w1,w1,r1,w0)}
13	March U [26]	13n	{⇕(w0); ⇑(r0,w1,r1,w0); ⇑(r0,w1); ⇓(r1,w0,r0,w1); ⇓(r1,w0)}
14	Algorithm B [7]	17n	{⇕(w0); ⇑(r0,w1,w0,w1); ⇑(r1,w0,r0,w1); ⇓(r1,w0,w1,w0); ⇓(r0,w1,r1,w0)}
15	PMOVI [27, 28, 29]	13n	{⇓(w0); ⇑(r0,w1,r1); ⇑(r1,w0,r0); ⇓(r0,w1,r1); ⇓(r1,w0,r0)}

*: The TL is $23n + 2Del$; where Del is the delay time required for detecting Data Retention Faults

5 Test evaluation

A set of 39 BTs has been applied to a large set of 0.13 micron 512 Kbyte SRAM chips, using the following stresses:

- Algorithm stresses
  - Addressing: Fast X 'fX' and Fast Y 'fY'.
  - Data-backgrounds: Solid (sDB), Checkerboard (bDB), Column Stripe (cDB), and Row Stripe (rDB).

- Environmental stresses
  - Voltage: high Voltage $(+V)$ and low Voltage $(-V)$
  - Speed: high Speed $(+S)$ and low Speed $(-S)$

The consequence is that each BT is applied 2(Voltage) ×2(Speed) ×2(Addressing) × 4(DB)=32 times. Therefore, a total of $39 \times 32 = 1248$ tests have been applied. However, 24 BTs produced results which were not very interesting; therefore, the results of the interesting 15 BTs, which means $15 \times 32 = 480$ tests, will be discussed.

Table 6 shows this set of BTs. The left column shows the number of the BT, the column 'BT' lists the name of the BT; the last column gives the description of the algorithm.

The results of the evaluation are presented in two parts: an analytical evaluation of the capability of the tests of Table 6, in terms of being able to detect Peripheral circuit Faults 'PFs', and an industrial evaluation by applying the tests of Table 6 to a large number of SRAM chips, using the above SCs.

5.1 Analytical analysis

The requirements for detecting PFs, see Table 3 and Table 4, can be applied to the BTs of Table 6; the result is shown in Table 7. The column 'MEt' describes the March Element type of the corresponding BT, as defined in Section 4. Similarly, the column 'RWS' lists the Read-Write Sequence, as specified in the corresponding BT. The columns 'SWDF', 'SSAF', and 'SPRF' list the required algorithm stress for detecting those PFs. Note that the detection of BLIFs requires a test with MEs of type MEwa, using the WaW RWS; see Table 3. None of the tests of Table 7 have that property; therefore the column 'BLIF' is absent in Table 7. Similarly, the column 'AD' is absent, because all PFs require the '$_x\Updownarrow$' AD. The required Data Backgrounds 'DBs' are fault specific; they are given under the columns 'SWDF', 'SSAF', and 'SPRF'.

Table 7. Algorithm stress for detecting PFs

Algorithm (BT) and its properties			Required algorithm stress		
BT	MEt	RWS	SWDF	SSAF	SPRF
Scan	ma	WaW,RaR	b,r	-	-
MATS+	ma	RaW	-	s,c	s,c
MATS++	ma	RaW,RaR	-	-	-
March B*	wa; ma	RaW	-	s,c; b,r	s,c; b,r
March C-	ma	RaW	-	s,c	s,c
March G*	wa; ma	RaR;RaW	-	s,c; b,r	s,c; b,r
March LA	ma	RaR	-	-	-
March LR*	wa;ma	RaR; RaW	-	s,c; b,r	s,c; b,r
March RAW	ma	RaR	-	-	-
March SR	wa;ma	RaW,RaR;WaW	b,r	b,r	b,r
March SS	ma	RaW	-	s,c	s,c
March SL	ma	RaW	-	s,c	s,c
March U*	wa;ma	RaW	-	s,c;b,r	s,c;b,r
Algor. B*	wa;ma	RaW	-	s,c;b,r	s,c;b,r
PMOVI	ma	RaR	-	-	-

*: The DBs (e.g., for March B) is 's,c' for 'ma' MEt and 'b,r' for 'wa' MEt
  's,c' means sDB or cDB; 'b,r' means bDB or rDB
The required AD is 'x ⇕'; it is the same for all PFs
Note: None of the BTs satisfy the requirements for detecting BLIFs

From Table 7 it can be concluded that the Scan and March SR are the only BTs capable of detecting SWDFs. MATS++, March LA, March RAW and PMOVI cannot detect any PFs; while March B, March G, March LR, March U and Algorithm B can detect SSAFs and SPRFs in more than one way: using MEs of type MEma or of type MEwa. It should be noted that a PF is considered detected by a march test if the test satisfies the MEt with the corresponding RWS together with the algorithm stress for *both* $x=0$ *and* $x=1$; see Section 4.2.

5.2 Experimental results

The application of the 15 BTs, using 32 SCs, results in 480 tests. These will be discussed in terms of the following subjects:

- Influence of environmental stresses
- Influence of algorithm stresses

Influence of environmental stresses The influence of the environmental stresses will be analyzed using Table 8. The table consists of four subtables; one for each combination of the values of the Speed (+S and -S) and Voltage (+V and -V) stresses. Each subtable has the following structure: The column '#', as listed in Table 6, shows the BT# of the BT listed in column 'Base Test'. In order to save space, 8 out of the 15 BTs of Table 6 have been selected. The set of 8 BTs consists of 6 BTs with the highest *Fault Coverage 'FC'*; the other two BTs are Scan and PMOVI. They are selected because of their special properties, as will follow later.

The column 'FC' lists the Fault Coverage (FC) of the corresponding BT, while the column 'UF' lists the number of *Unique Faults 'UFs'* detected with the BT. A *UF* is a fault which is only detected with a single BT; i.e., none of the other BTs, or tests, in that table do detect that fault. The

columns '1, 4, 6, ..., 15' represent the same BTs as listed in column '#'; they form a matrix of FCs.

The **bold-face** diagonal entries of the matrix list the FC of the corresponding BT; for ease of reference, these entries are also listed in the column 'FC'.

The entries *below* the bold-face diagonal list the FC of the *union* of the FCs of the corresponding two tests. For example, in the +S+V subtable, the FC of March G is 120 and of Scan is 113; the *union* of the FCs is 124 (printed in *italics* in the +S+V subtable); this is the number of faults Scan and March G detect *together*.

The entries *above* the bold-face diagonal list the FC of the *intersection* of the FCs of the corresponding two tests. For example, the *intersection* of the FCs of March G and Scan is 109 (printed in italics in the +S+V subtable); this is the number *common* faults detected with both March G and Scan.

Table 10 summarizes the influence of the environmental SCs on the FC of the BTs; the entries have been derived from Table 8. The table shows that the stress +S results in a higher FC than -S, and +V in a higher FC than -V. The SC +S+V has the highest FC, and -S-V the lowest.

Table 10 also shows that different BTs obtain their highest FC with different environmental SCs; e.g., the very simple $13n$ BT 'March U' has the highest FC for the SCs +S-V and -S-V; which is very interesting. Note that March U, March LR and March SR are BTs which contain MEs of type MEwa, see Table 6 and Table 7; this makes them very different from the more traditional march tests based on MEs of type MEma, such that they also may detect different types of faults. March U and March LR appear 3 out of 6 times in column 'Highest FC', and 3 out of 4 times in column 'Highest Union'; while they do not appear in the column 'Lowest FC'!

Note that Algorithm B [7] has been designed for detecting the same faults as March B [7], except for the fact that it is more symmetric, which facilitates its implementation. March B has been extended to become March G[20], in order to cover data retention faults; see Table 6. Therefore, the potential presence of Data Retention Faults 'DRFs' may explain the high FC of March G.

From Table 8 one can conclude that the BTs Scan and PMOVI have the lowest FC; however, they have been included for other purposes. Scan is the only test capable of detecting SWDFs, such that in many situations it does detect unique faults 'UFs' [14, 15]. MOVI has been included because, although it typically has a lower FC than March C-, which is of the same complexity, it detects speed related faults [10], address decoder delay faults [29], and many unique faults in DRAMs [14, 15]. However, in this experiment, MOVI did not outperform the other BTs.

Table 8 shows that the highest Union values in each of the subtables always includes the Scan test, together with an-

Table 8. Influence of environmental stresses

		high Speed (+S) Testing																				
		high Voltage (+V); Subtable +S+V										low Voltage (-V); Subtable +S-V									
#	Base Test	FC	UF	1	4	6	8	12	13	14	15	#	FC	UF	1	4	6	8	12	13	14	15
1	Scan	113	0	113	108	109	109	111	110	109	109	1	96	1	96	91	91	93	90	93	91	89
4	March B	116	0	121	116	116	115	116	116	116	110	4	103	0	108	103	101	101	99	101	99	96
6	March G	120	1	124	120	120	116	118	118	117	112	6	103	0	108	105	103	102	99	102	99	95
8	March LR	117	0	121	118	121	117	116	116	115	111	8	105	0	108	107	106	105	99	104	101	96
12	March SL	120	0	122	120	122	121	120	119	117	113	12	100	0	106	104	104	106	100	99	98	94
13	March U	119	0	122	119	121	120	120	119	117	113	13	106	0	109	108	107	107	107	106	102	95
14	Algorithm B	117	0	121	117	120	119	120	119	117	111	14	102	0	107	106	106	106	104	106	102	94
15	PMOVI	113	0	117	119	121	119	120	119	119	113	15	96	0	103	103	104	105	102	107	104	96

		low Speed (-S) Testing																				
		high Voltage (+V); Subtable -S+V										low Voltage (-V); Subtable -S-V									
#	Base Test	FC	UF	1	4	6	8	12	13	14	15	#	FC	UF	1	4	8	12	13	14	15	8
1	Scan	97	0	97	94	94	94	94	95	96	94	1	85	0	85	82	82	82	81	81	80	80
4	March B	100	0	103	100	99	100	99	99	97	96	4	99	0	102	99	96	95	92	97	93	91
6	March G	99	0	102	100	99	99	99	99	97	96	6	97	0	100	100	97	95	92	95	94	91
8	March LR	102	0	105	102	102	102	100	99	98	96	8	97	0	100	101	99	97	93	96	93	92
12	March SL	100	0	103	101	100	102	100	99	97	96	12	94	0	98	101	99	98	94	93	91	91
13	March U	100	0	102	101	100	103	101	100	98	96	13	99	0	103	101	101	100	100	99	93	92
14	Algorithm B	100	0	101	103	102	104	103	102	100	96	14	95	0	100	101	98	99	98	101	95	91
15	PMOVI	96	0	99	100	99	102	100	100	100	96	15	92	0	97	100	98	97	95	99	96	92

Table 9. Fault coverage of some algorithms for High voltage and fast timing

#	Algorithm	FC	UF	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	Scan	113	0	113	110	109	108	110	109	109	109	109	110	111	111	110	109	109
2	MATS+	115	0	118	115	113	111	114	112	111	112	111	110	113	113	113	112	111
3	MATS++	115	0	119	117	115	112	114	113	113	114	113	112	114	113	113	112	111
4	March B	116	0	121	120	119	116	112	116	113	115	112	113	113	116	116	116	110
5	March C-	116	0	119	117	117	120	116	114	113	114	113	112	115	114	114	113	112
6	March G	120	1	124	123	122	120	122	120	114	116	113	114	114	118	118	117	112
7	March LA	115	0	119	119	117	118	118	121	115	115	114	113	114	114	114	113	111
8	March LR	117	0	121	120	118	118	119	121	117	117	114	114	115	116	116	115	111
9	March RAW	114	0	118	118	116	118	117	121	115	117	114	112	114	113	113	112	111
10	March SR	116	0	119	121	119	119	120	122	118	119	118	116	114	115	114	113	109
11	March SS	118	0	120	120	119	121	119	124	119	120	118	120	118	115	114	113	111
12	March SL	120	0	122	122	122	120	122	122	121	121	121	121	123	120	119	117	113
13	March U	119	0	122	121	121	119	121	121	120	120	120	121	123	120	119	117	113
14	Algorithm B	117	0	121	120	120	117	120	120	119	119	119	120	122	120	119	117	111
15	PMOVI	113	0	117	117	117	119	117	121	117	119	116	120	120	120	119	119	113

other BT; this is summarized in the column 'Highest Union' of Table 10. This means that the Scan BT detects more of another class of faults than any of the other BTs. In three of the four cases, the second test is March U or March LR; both contain MEs of type MEwa. Note that the March G (for SC= +S+V) and Scan (for SC= +S-V) detect unique faults 'UFs'.

Tables 8 and 10 show that BTs performed with the environmental SC +S+V have the highest FC. Therefore the subtable +S+V of Table 8 has been extended to include all of the 15 selected tests; see Table 9. This table shows that the FCs of the 15 BTs are very close.

Table 10. Summary of environmental SCs

SC	Highest FC		Lowest FC		Highest Union	
	FC	BTs*	FC	BTs	FC	BTs**
+S+V	120	March G, SL	116	March B	124	March G
+S-V	106	March U	102	Alg. B	109	March U
-S+V	102	March LR	99	March G	105	March LR
-S-V	99	March B, U	84	March SL	103	March U

*: The notation 'March B, U' means 'March B and March U'
**: The highest union always includes Scan as one of the two BTs

Influence of algorithm stresses

The consequences of the algorithm stresses (AD=fX or fY, and DB=sDB, bDB, rDB or cDB) for the environmental SC +S+V is shown in Table 11. The table consists of 6 subta-

Table 11. Algorithm stresses

High Speed (+S) and High Voltage Testing(+V)

Scan

#	SC	FC	UF	1	2	3	4	5	6	7	8
1	fX-sDB	79	0	79	72	72	70	75	71	73	69
2	fX-bDB	89	0	96	89	76	81	74	84	75	82
3	fX-rDB	82	0	89	95	82	76	74	77	75	76
4	fX-cDB	89	1	98	97	95	89	73	81	73	82
5	fY-sDB	83	0	87	98	91	99	83	76	78	75
6	fY-bDB	92	0	100	97	97	100	99	92	76	85
7	fY-rDB	82	1	88	96	89	98	87	98	82	76
8	fY-cDB	92	0	102	99	98	99	100	99	98	92

March G

#	FC	UF	1	2	3	4	5	6	7	8
1	104	0	104	100	98	102	102	99	101	102
2	101	0	105	101	97	100	100	98	99	100
3	101	1	107	105	101	98	99	96	98	99
4	105	0	107	106	108	105	103	100	102	103
5	109	0	111	110	111	111	109	106	108	109
6	106	0	111	109	111	111	109	106	106	106
7	108	0	111	110	111	111	109	108	108	108
8	109	0	111	110	111	111	109	109	109	109

March LR

#	SC	FC	UF	1	2	3	4	5	6	7	8
1	fX-sDB	102	0	102	100	96	100	100	98	98	100
2	fX-bDB	103	0	105	103	96	99	100	98	98	100
3	fX-rDB	97	0	103	104	97	97	97	97	96	97
4	fX-cDB	102	0	104	106	102	102	101	99	98	100
5	fY-sDB	110	0	112	113	110	111	110	108	107	109
6	fY-bDB	108	0	112	113	108	111	110	108	106	108
7	fY-rDB	109	1	113	114	110	113	112	111	109	108
8	fYcDB	110	0	112	113	110	112	111	110	110	110

March SL

#	FC	UF	1	2	3	4	5	6	7	8
1	101	0	101	97	95	94	96	94	98	92
2	100	0	104	100	96	93	97	97	98	95
3	96	0	102	100	96	93	94	95	96	93
4	97	0	104	104	100	97	93	92	95	91
5	105	0	110	108	107	109	105	100	102	100
6	105	1	112	108	106	110	110	105	101	101
7	107	0	110	109	107	109	110	111	107	100
8	102	0	111	107	105	108	107	106	109	102

March U

#	SC	FC	UF	1	2	3	4	5	6	7	8
1	fX-sDB	99	0	99	97	95	97	96	96	97	96
2	fX-bDB	103	0	105	103	99	100	99	100	101	98
3	fX-rDB	100	0	104	104	100	99	98	97	98	96
4	fX-cDB	103	0	105	106	104	103	98	98	100	98
5	fY-sDB	108	0	111	112	110	113	108	106	106	105
6	fY-bDB	110	1	113	113	113	115	112	110	109	105
7	fY-rDB	112	1	114	114	114	115	114	113	112	106
8	fY-cDB	108	0	111	113	112	113	111	113	114	108

PMOVI

#	FC	UF	1	2	3	4	5	6	7	8
1	99	0	99	95	94	96	97	94	98	95
2	99	0	103	99	94	96	97	94	97	96
3	98	0	103	103	98	95	96	94	95	95
4	100	1	103	103	103	100	97	94	97	97
5	105	0	107	107	107	108	105	98	102	100
6	100	0	105	105	104	106	107	100	99	99
7	104	0	105	106	107	107	107	105	104	100
8	102	0	106	105	105	105	107	103	106	102

bles, one for each of the selected BTs. Table 12 summarizes the result.

It is interesting to note that the highest FCs are always obtained for the AD=fY, and the lowest for AD=fX; without exception. This denotes that the particular SRAMs being tested are more sensitive to delays in the column decoder, which could indicate that accessing columns is the most timing critical operation.

The fY addressing, together with the cDB or the sDB, are the most effective; while fX addressing, together with rDB, are the least effective.

The 'Highest Union' almost consistently involves a test using the fY AD, and another test using the fX AD; while with the fX AD typically the sDB is used.

6 Conclusions

In this paper the space of faults in the memory peripheral circuits, denoted as 'PFs', has been explored. The space consists of the Slow Write Driver Fault 'SWDF', the Slow Sense Amplifier Fault 'SSAF', the Slow PRecharge circuit Fault 'SPRF', and the Bit Line Imbalance Fault 'BLIF'. Similar to the detection conditions for address decoder faults, it has been shown that PFs can be detected with march tests, provided that they satisfy particular combinations of the following properties: properties of the algorithm

Table 12. Influence of algorithm SCs

BT	Highest FC		Lowest FC		Highest Union	
	FC	SC*	FC	SC*	FC	SC**
Scan	92	Yb;Yc	79	Xs	102	Xs+Yc
March G	109	Ys;Yc	101	Xb;Xr	111	Many*
March LR	110	Ys;Yc	97	Xr	113	Many*
March SL	107	Yr	96	Xr	112	Xs+Yb
March U	112	Yr	99	Xs	115	Many
PMOVI	105	Ys	98	Xr	108	Xc+Ys

*: The entries 'Pq' represent the SC of the BT
  'P' specifi es the AD; P ∈ {X=fX, Y=fY}
  'q' specifi es the DB; q ∈ {s=sDB, b=bDB, r=rDB, c=cDB}
  'Yb;Yc' means that Yb or Yc can be used
**: Many also includes Xs

(the 'march element type' and the 'read-write sequence'), together with properties of the algorithm stress (the address direction and the data background).

The above properties have been analyzed for a large set of well-know tests, while these tests also have been applied to a large number of SRAM chips. The results show that:

- Environmental stresses: High voltage is the most stressful, followed by High Speed; while High Voltage and High Speed is the most stressful combination. The least stressful combination is Low Voltage and Low Speed.

- Algorithm stress: Fast Y addressing is the most stress-

ful Addressing Direction 'AD', and the Solid and the Column Stripe Data Backgrounds are the most stressful DBs.

- *Algorithm impact*: Scan and March SR are the only algorithms capable of detecting SWDFs; March B, March G, March LR, March SR, March U and Algorithm B are the algorithms containing march elements of type Marching and of type Walking; which makes them very effective in detecting SSAFs and SPRFs. None of the evaluated tests is capable of detecting BLIFs.

In addition, a set of new, efficient, march tests for detecting PFs has been designed, with test lengths between $4n$ and $8n$. The analysis done showed that all targeted PFs can be covered with only two march tests with a total test length of at most $14n$.

The above results are, of course design and layout dependent. Evaluating the same memory tests with the same stresses, using a different memory implementation may produce different results.

References

[1] P. Nigh and A. Gattiker, 'Test Method Evaluation Experiments & Data'. In Proc. of the IEEE Int. Test Conf., 2000, pp.454-463

[2] S. Kundu et al., 'Test Challenges in Nanometer Technologies', Journal of Electronic Testing: Theory and Applications, Vol. 17, No. 3/4, June/August 2001, pp. 209-218

[3] A.D. Sathe et al., 'Analog Macromodeling of Capacitive Coupling Faults in Digital Circuit Interconnects'. In Proc. of the IEEE Int. Test Conf., 2002, pp.375-383

[4] R.D. Adams and E.S. Cooley, "False Write Through and Un-Restored Write Electrical Level Faults Models for SRAMs", *In Proc. of IEEE Int. Workshop on Memory Technology, Design and Test*, pp. 27-32, 1997.

[5] R. Nair et al., 'Effi cient Algorithms for Testing Semiconductor Random-Access Memories', IEEE Trans. on Comp., C-27(6), 1978, pp. 572-576

[6] D.S. Suk and S.M. Reddy, 'A March Test for Functional Faults in Semiconductor Random Access Memories', IEEE Trans. on Comp., C-30(12), 1981, pp. 982-982

[7] M. Marinescu, 'Simple and Effi cient Algorithms for Functional RAM Testing', In Proc. IEEE Int. Test Conf., 1982, pp. 236-239

[8] M.S. Abadir and J.K. Reghbati, 'Functional Testing of Semiconductor Random Access Memories', ACM Computing Surveys, 15(3), 1983, pp. 175-198

[9] R. Dekker et al., 'Fault Modeling and Test Algorithm Development for Static Random Access Memories'. In Proc. of the IEEE Int. Test Conf. (ITC'1988), 1988, pp.343-352

[10] A.J. van de Goor, 'Testing Semiconductor Memories: Theory and Practice', *ComTex Publishing, Gouda, The Netherlands*, 1998; ISBN 90-804276-1-6. http://ce.et.tudelf.nl/~vdgoor/

[11] R.D. Adams, 'High Performance Memory Testing', Kluwer Academic Publishers, Norwell, MA, USA, 2003

[12] S. Hamdioui, 'Testing Static Random Access Memories: Defects, Fault Models and Test Patterns', Kluwer Academic Publishers, Boston, MA; ISBN 1-4020-7752-1, 2004

[13] S.N.H. Goto and K. Iwasaki, 'Experiment fault analysis of 2Mb SRAM chips', Proc. of European Design and Test Conf., 1999, pp. 623 -630

[14] A. J. van de Goor and J. de Neef, 'Industrial Analysis of DRAM Tests', In Proc. Design Automation and Test in Europe, March 9 -12, Munich, 1999, pp. 623-630

[15] A. J. van de Goor and A. Paalvast, 'Industrial Evaluation of DRAM SIMM Tests', In Proc. IEEE Int. Test Conf., 2000, pp. 426-435

[16] I. Schanstra and A.J. van de Goor, 'Industrial Evaluation of Stress Combinations for March Tests Applied to SRAMs', Proc. of the IEEE Int. Test Conf., 1999, pp. 983-992

[17] A.J. van de Goor and Z. Al-Ars, 'Functional Memory Faults: A Formal Notation and a Taxonomy', In Proc. of the 18th IEEE VLSI Test Symposium, Montreal, pp. 281-289

[18] A.J. van de Goor and C.A. Verruijt, 'An Overview of Deterministic Functional RAM Chip Testing', ACM Computing Surveys, Vol. 22, No. 1, 1990, pp. 5-33

[19] P. Mazumder, 'Parallel Testing of Parametric Faults in Three Dimensional Random-Access Memory', Proc. IEEE Int. Test Conf., 1988, pp. 933-941

[20] A.J. van de Goor, 'Using March Tests to Test SRAMs', IEEE Design & Test of Computers, 1993, pp. 8-14

[21] A.J. van de Goor et al., 'March LA: A Test for Linked Memory Faults', In Proc. European Design and Test Conference, 1997, pp. 627

[22] A.J. van de Goor and G.N. Gaydadjiev, 'March LR: A Memory Test for Realistic Linked Faults', In Proc. IEEE VLSI Test Symposium, 1996, pp. 272-280

[23] S. Hamdioui, Z. Al-Ars and A. J. van de Goor, 'Testing Static and Dynamic Faults in Random Access Memories', In Proc. of the 20th IEEE VLSI Test Symposium, Monterey, CA, pp. 395-400

[24] S. Hamdioui, A. J. van de Goor and M. Rodgers, 'March SS: A Test for All Static Simple RAM Faults', In Proc. of the IEEE Int. Workshop on Memory Technology, Design and Testing, Bendor, France, pp. 95-100

[25] S. Hamdioui, Z. Al-Ars, A. J. van de Goor and M. Rodgers, 'Linked Faults in Random Access Memories: Concept, Fault Models, Test Algorithms and Industrial Results", In IEEE Trans. on Computer Aided Design of Integrated Circuits and Systems, Vol. 23, No. 5, pp. 195-205, May 2004.

[26] A.J. van de Goor and G.N. Gaydadjiev, 'March U: A Test for Unlinked Memory Faults', IEE Proceedings Circuits, Devices and Systems; Vol. 144, No. 3, June 1997, pp. 155-160

[27] J.H. de Jonge and A.J. Smeulders, 'Moving Inversion Test Pattern is Thorough, Yet Speedy', Computer Design, May 1976, pp. 169-173

[28] B. Nadeau-Dostie et al., 'Serial Interfacing for Embedded-Memory Testing', IEEE Design & Test of Comp., Vol. 7, No. 2, 1990, pp. 52-63

[29] M. Klaus and A. J. van de Goor, 'Tests for Resistive and Capacitive Defects in Address Decoders', In Proc. of the Tenth Asian Test Symposium (ATS'01), Kyoto, Japan, 2001, pp. 31-36