A Systematic Method for Modifying March Tests for Bit-Oriented Memories into Tests for Word-Oriented Memories

Ad J. van de Goor, *Fellow*, *IEEE*, and Issam B.S. Tlili

Abstract—Most memory test algorithms are optimized for a particular memory technology and a particular set of fault models, under the assumption that the memory is bit-oriented, i.e., read and write operations affect only a single bit in the memory. Traditionally, word-oriented memories have been tested by repeated application of a test for bit-oriented memories, whereby a different data background is used during each application. This results in time inefficiencies and limited fault coverage. A systematic way of converting tests for bit-oriented memories into tests for word-oriented memories is presented, distinguishing between interword and intraword faults. The conversion consists of concatenating to the test for interword faults, a test for intraword faults. This approach results in more efficient tests with complete coverage of the targeted faults. Word-oriented memory tests are very important, because most memories have an external data path which is wider than one bit.

Index Terms—Bit-oriented memories, word-oriented memories, march tests, data backgrounds, fault models.

1 Introduction

Word-oriented memories (WOMs) contain more than one bit per word, i.e., $B \geq 2$, whereby B represents the number of bits per word and usually is a power of two. Read operations read the B bits simultaneously; write operations write data into the B bits, whereby the data to be written into each bit can be specified independently of the data for the other bits. The value of a B-bit word is referred to as the data background (*DB*) [1].

The *traditional* method for testing WOMs consists of the repeated application of a test for *bit-oriented memories (BOMs)*, whereby a different DB is used during each application. The tests for WOMs to detect *coupling faults (CFs)* [2], [3], [1] used different types of DBs, such as the Walking 1/O and the Bridging Fault patterns. The disadvantages of the repeated application of the BOM test, together with these types of DBs, are test time inefficiency and limited fault coverage for intraword CFs.

This paper presents an *improved*, systematic method to convert march tests for BOMs into march tests for WOMs; it is based on [4]. Subsets of this problem have been addressed in [5], [6]. The conversion consists of concatenating to the march test for single-cell faults, and possible *interword* faults (i.e., faults between words), a march test designed for *intraword faults* (i.e., faults within words). This changes the test time from the *product* of "(BOM test time) * (number of DBs/2)" to their *sum*.

- *A.J. van de Goor is with the Department of Information Technology and Systems, Section Computer Engineering, Delft University of Technology, Mekelweg 4, 2628 CD Delft, The Netherlands. E-mail: A.J.vandeGoor@its.tudelft.nl.*
- *I.B.S. Tlili is with the Department of Radiography/Fluoroscopy, Philips Medical Systems, Best, The Netherlands.*

Manuscript received 20 Jan. 1998; revised 12 Aug. 2002; accepted 13 Aug. 2002.
For information on obtaining reprints of this article, please send e-mail to: tc@computer.org, and reference IEEECS Log Number 106178.

The organization of this paper is as follows: Section 2 describes how a BOM test can be converted into a WOM test for single-cell and interword faults; Section 3 derives the DBs required for the different unrestricted intraword CF types and shows their use in a WOM test. Section 4 shows the relationships between the newly introduced tests for intraword CFs and describes the influence the memory organization has on the to-be-used intraword tests. Section 5 derives the DBs and tests for restricted intraword CFs, while Section 6 ends with conclusions.

2 Tests for Single-Cell and Interword Faults

The fault models for WOMs can be divided into the following classes:

1. *Single-cell faults*. These are the classical *stuck-at faults (SAFs), transition faults (TFs), data retention faults (DRFs)*, and *read disturb faults (RDFs)* [7]. They will be detected by classical BOM tests, such as MATS+, March C-, etc.
2. *Faults between memory cells*. This class of faults consists of coupling faults (CFs). They can be further divided into the subclasses:
   a. *Interword faults*. These faults are the classical CFs whereby the aggressor and victim cells belong to *different words*. They will be detected by properly designed BOM tests.
   b. *Intraword faults*. These are CFs whereby the aggressor and victim cells belong to the *same word*. A special *intraword test* has to be designed to detect this class of faults.

Depending on the dominance of the write operation on the CF, intraword CFs may or may not be detectable:

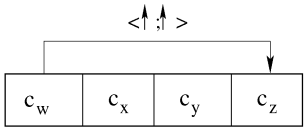


Fig. 1. Intraword CFid.

1. If the write operation dominates the CF (i.e., the value specified in the write operation will be stored in the cells), the CF will have no effect and therefore appears not to be present. A test to detect this nondominating CF does not have to be performed because the CF does not manifest itself.
2. If the CF dominates the write operation such that the CF will manifest itself, it is required to detect the CF. In the case of *idempotent CFs (CFids)*, *disturb CFs (CFdsts)* [8], and *state CFs (CFsts)* [1], the fault will not be detectable with algorithms for BOMs, as shown below.

Any BOM test can be converted into a WOM test to detect single-cell and interword faults, as follows:

1. The "w0" operation should be replaced with a "*w-data background*" denoted as "w_D," whereby *any* DB is acceptable. For example: "w0...0" (an all 0s DB), "w01...01" (a DB pattern of a repeated sequence of "01"), etc., whereby the length of the bitstring is B bits. The "w1" operation should be replaced with a write operation which writes the *inverted data background*, i.e., "$w_{\bar{D}}$."
2. The "r0" and "r1" operations should be replaced with the operations "*r-data background*" (r_D) and "*r-inverted-data background*" ($r_{\bar{D}}$).
3. In the equations expressing the required number of operations for a test, n (representing the number of bits in the chip) has to be replaced by n/B (representing the number of words in the chip).

Converting the BOM MATS+ test $\{\Updownarrow (w0); \Uparrow (r0, w1); \Downarrow (r1, w0)\}$ into a WOM test with $B = 4$, using the DB "0101" results in the test:

$$\{\Updownarrow (w0101); \Uparrow (r0101, w1010); \Downarrow (r1010, w0101)\}.$$

The DB has been chosen to be "0101" because it also will detect certain coupling faults between adjacent data lines and adjacent data bits in the same row. The test length changes from $5 * n$ to $5 * n/B$. Other DBs are possible, e.g., a solid DB, which maximizes ground bounce: $\{\Updownarrow (w0000); \Uparrow (r0000, w1111); \Downarrow (r1111, w0000)\}$.

Converting the BOM March C− test $\{\Updownarrow (w0); \Uparrow (r0, w1); \Uparrow (r1, w0); \Downarrow (r0, w1); \Downarrow (r1, w0); \Updownarrow (r0)\}$ into a WOM test with $B = 4$, using the DB "0000," results in the test:

$$\{\Updownarrow (w0000); \Uparrow (r0000, w1111); \Uparrow (r1111, w0000); \Downarrow (r0000, w1111); \Downarrow (r1111, w0000); \Updownarrow (r0000)\}.$$

Note that, for this WOM test, the set of interword fault models consists of AFs (address decoder faults), SAFs, TFs, RDFs, CFsts, CFids [2].

$$\mathbf{M} = \begin{vmatrix} c_{0,0} & c_{0,1} & \cdots & c_{0,B-1} \\ c_{1,0} & c_{1,1} & \cdots & c_{1,B-1} \\ \cdots & \cdots & \cdots & \cdots \\ c_{d-1,0} & c_{d-1,1} & \cdots & c_{d-1,B-1} \end{vmatrix}$$

Fig. 2. Matrix M.

However, the above BOM-to-WOM test conversion method does not guarantee the detection of all intraword CFs. For example, Fig. 1 shows a memory word consisting of four cells: $W_4 = (C_w, C_x, C_y, C_z)$ and the CFid, whereby C_z is $<\uparrow;\uparrow>$ coupled to C_w. (Note: The notation for the CFid $<\uparrow;\uparrow>$ means that an $\uparrow$ transition write operation applied to the aggressor cell causes an $\uparrow$ transition in the victim cell.) Assuming that the memory word contains a "0...0" value, a "w1111" operation could be used to sensitize the CFid; however, this would mask the CFid. The detection of this fault requires that the CFid is sensitized by the following write "w1*xy*0," whereby x and y may have arbitrary values such that the CFid is not masked.

3 Tests for Unrestricted Intraword Faults

The DBs to be used for intraword faults are determined by the used intraword coupling fault models. In this section, it is assumed that the layout of the memory cells in a row of the memory is not known; the fault models reflecting this assumption are the *unrestricted CF models* (*uCFs*). Section 3.1 presents the *set of DBs* to be used for detecting unrestricted intraword CFsts (uCFsts), Section 3.2 derives the *DB sequence* to be used for detecting unrestricted intraword CFids (uCFids), Section 3.3 shows the *DB operation sequence* to be used for unrestricted intraword CFdsts (uCFdsts), while Section 3.4 shows how to design march tests for uCFs.

3.1 Set of Data Backgrounds for Unrestricted Intraword CFsts (uCFsts)

In order to be able to detect CFsts between cells in a word, all states of two arbitrary cells c_i and c_j should be checked, i.e., the states of $(c_i, c_j) \in \{(0,0), (0,1), (1,0), (1,1)\}$. This fault model is referred to as the *unrestricted intraword CFst model* (*uCFst model*). Initially, only the states (0, 0) and (0, 1) have to be checked; the other two states will be checked when the algorithm is executed with the inverse data backgrounds. The *number of DBs*, d, required for checking the states (0, 0) and (0, 1) is [1]: $d = \lceil \log_2 B \rceil + 1$; if B is a power of 2, this will become: $d = \log_2 B + 1$.

The required DBs can be represented by the matrix **M** with d rows and B columns (see Fig. 2): A column i of M will be denoted by vector $\mathbf{V_i}$; it represents the DB values for cell i in the word because the values of $\mathbf{V_i}$ will be succesively written into and read from cell i.

For a minimal set of DBs, the following requirements should be satisfied:

1. No two vectors $\mathbf{V_i}$ and $\mathbf{V_j}$ may be equal. If they are equal, the corresponding cells c_i and c_j will only be checked for the states (0, 0) and (1, 1). If no two vectors may be equal, the maximum possible number of different vectors of d bits is determined by the word length as follows: $2^d = B$.

TABLE 1
DBs fof CFsts ($B = 8$)

#	Normal	#	Inverse
0	00000000	1	11111111
2	01010101	3	10101010
4	00110011	5	11001100
6	00001111	7	11110000

2. No two vectors $\mathbf{V_i}$ and $\mathbf{V_j}$ may be each other's inverse. If they are each other's inverse, cells c_i and c_j will only be checked for the states (0, 1) and (1, 0). The additional requirement that no two vectors may be each other's inverse halves the maximum number of vectors, i.e., $2^{d-1} = B$ or $d = \log_2 B + 1$.

The *actual set of DBs* is twice as large because the d inverse DBs also have to be taken into account such that $d = 2 * \lceil \log_2 B \rceil + 2$. The nature of uCFsts is such that only the *states* of the cells are relevant for sensitizing and detecting a fault; therefore, the DBs of the set can be applied in any order. For a memory with $B = 8$ (a byte-wide memory) the DBs of Table 1 could be used. Columns 2 and 3 of Table 2 list the required d and the *Test Lenght* (*TL*) for uCFsts, as a function of B; TL represents the number of read and write operations required to apply the d DBs, assuming that all 0s and all 1s DBs have already been verified by the BOM test.

3.2 Data Background Sequence for Unrestricted CFids (uCFids)

Using the uCFid fault type for intraword faults, the intraword test has to use a set of DBs which have to be applied to the memory in a particular sequence, i.e., a *sequence of DBs*, also called the *data background sequence* (*DBS*), is required [5]. This sequence is derived below.

In order to be able to detect uCFids, for the case that the uCFid dominates the write operation, the algorithms for BOMs have to be modified using a DBS. For the uCFids within a B-bit memory word, four subtypes exist: $<\uparrow;\downarrow>_{c_a,c_v}$ (Note: c_a denotes that cell-a is the aggressor cell and c_v denotes that cell-v is the victim cell), $<\uparrow;\uparrow>_{c_a,c_v}$, $<\downarrow;\downarrow>_{c_a,c_v}$, and $<\downarrow;\uparrow>_{c_a,c_v}$, where $a, v \in \{0, 1, \ldots, B-1\}$. Each subtype has $C_B^2 = B * (B-1)$ possible cases because any of the B cells can be the aggressor, while any of the $B-1$ nonaggressor cells can be the victim; the total number of uCFids is therefore $4 * B * (B-1)$. The purpose of this section is to find the minimal DBS which can sensitize all $4 * B * (B-1)$ uCFids.

TABLE 2
Number of DBs, d, and Test Length, TL, as a Function of B

B	uCFst		uCFid		uCFdst	
	d	TL	d	TL	d	TL
2	4	4	6	10	6	13
3-4	6	8	9	16	9	20
5-8	8	12	12	22	12	27
9-16	10	16	15	28	15	34
17-32	12	20	18	34	18	41
...	...	...	...	...	...	...
B	$2 * \beta$	$4 * \beta$	$3 * \beta$	$6 * \beta$	$3 * \beta$	$7 * \beta$
	$+2$		$+3$	$+4$	$+3$	$+6$

Note: $\beta = \lceil \log_2 B \rceil$.

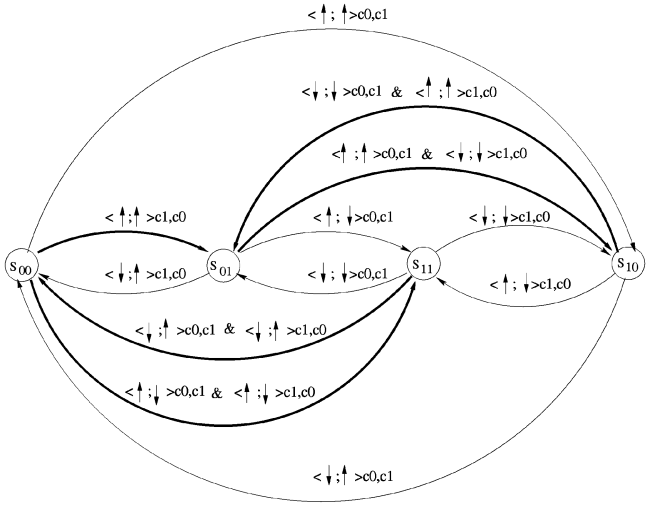


Fig. 3. State diagram for sensitizing uCFids ($B = 2$).

Fig. 3 shows the state diagram for *sensitizing* the uCFids in WOMs with $B = 2$, denoted as W_2. The states (nodes) are numbered according to the value of the two cells c_0 and c_1 in the word; the arcs (which represent the transition write operations) are labeled with the sensitized faults. From Fig. 3, we can see that:

1. A given uCFid can be sensitized by different arcs (transition write operations), e.g., the uCFid $<\uparrow;\uparrow>_{c_1,c_0}$ is sensitized by both arcs (S_{00}, S_{01}) and (S_{10}, S_{01}).
2. Each of the arcs formed by the states which are each others inverse, i.e., (S_{00}, S_{11}), (S_{11}, S_{00}), (S_{01}, S_{10}), and (S_{10}, S_{01}), is labeled with two uCFids; furthermore, those four arcs can sensitize all eight uCFids.

The above means that the state diagram of Fig. 3 can be reduced to contain only the four *bold* arcs. The fifth bold arc, (S_{00}, S_{01}), is needed to connect the two pairs of arcs. The DBS for a 2-bit word is: $\mathbf{S}_2 = 00, 11, 00, 01, 10, 01$. The DBS $\mathbf{S}_2$ can be split into: $\mathbf{S}'_2 = 00, 11, 00$ and $\mathbf{S}''_2 = 01, 10, 01$.

Extending the DBS to WOMs with 8-bit words, $W_8 = \{c_0, c_1, c_2, c_3, c_4, c_5, c_6, c_7\}$, requires the following steps:

1. *Level 0*: For each cell-pair (c_i, c_{i+1}), we apply the DBS $\mathbf{S}_2$ for 2-bit words, replicated four times to fill the 8-bit word; see Table 3. This table shows that all uCFids between $(c_i, c_{i+1+k*2})$ are sensitized, where $k \in \{0, 1 \ldots, \lfloor((B-1)-(i+1))/2\rfloor\}$; this is also shown in Fig. 4a, where the arc between two cells c_i and c_j implies that all uCFids between cells c_i and c_j are sensitized.
2. *Level 1*: For each cell-pair (c_i, c_{i+2}), we apply only the DBS $\mathbf{S}''_2 = 01, 10, 01$; this is sufficient because the DBS $\mathbf{S}'_2 = 00, 11, 00$ has already been applied in

TABLE 3
DBS $\mathbf{S}_8$ for uCFids ($B = 8$)

#	Data background $c_0c_1c_2c_3c_4c_5c_6c_7$	Lev.	Faults detected
0	0 0 0 0 0 0 0 0		
1	1 1 1 1 1 1 1 1	0	$<\uparrow;\downarrow>_{c_i,c_j}$ & $<\uparrow;\downarrow>_{c_j,c_i}$
2	0 0 0 0 0 0 0 0	0	$<\downarrow;\uparrow>_{c_i,c_j}$ & $<\downarrow;\uparrow>_{c_j,c_i}$
3	0 1 0 1 0 1 0 1	0	
4a	1 0 1 0 1 0 1 0	0	$<\uparrow;\uparrow>_{c_i,c_j}$ & $<\downarrow;\downarrow>_{c_j,c_i}$
4b			$<\downarrow;\downarrow>_{c_i,c_j}$ & $<\uparrow;\uparrow>_{c_j,c_i}$
5a	0 1 0 1 0 1 0 1	0	$<\downarrow;\downarrow>_{c_i,c_j}$ & $<\uparrow;\uparrow>_{c_j,c_i}$
5b			$<\uparrow;\uparrow>_{c_i,c_j}$ & $<\downarrow;\downarrow>_{c_j,c_i}$
6	0 0 1 1 0 0 1 1	1	
7a	1 1 0 0 1 1 0 0	1	$<\uparrow;\uparrow>_{c_i,c_j}$ & $<\downarrow;\downarrow>_{c_j,c_i}$
7b			$<\downarrow;\downarrow>_{c_i,c_j}$ & $<\uparrow;\uparrow>_{c_j,c_i}$
8a	0 0 1 1 0 0 1 1	1	$<\downarrow;\downarrow>_{c_i,c_j}$ & $<\uparrow;\uparrow>_{c_j,c_i}$
8b			$<\uparrow;\uparrow>_{c_i,c_j}$ & $<\downarrow;\downarrow>_{c_j,c_i}$
9	0 0 0 0 1 1 1 1	2	
10a	1 1 1 1 0 0 0 0	2	$<\uparrow;\uparrow>_{c_i,c_j}$ & $<\downarrow;\downarrow>_{c_j,c_i}$
10b			$<\downarrow;\downarrow>_{c_i,c_j}$ & $<\uparrow;\uparrow>_{c_j,c_i}$
11a	0 0 0 0 1 1 1 1	2	$<\downarrow;\downarrow>_{c_i,c_j}$ & $<\uparrow;\uparrow>_{c_j,c_i}$
11b			$<\uparrow;\uparrow>_{c_i,c_j}$ & $<\downarrow;\downarrow>_{c_j,c_i}$
#1,#2:		$j = i + 1 + k * 2$	
#4a, #5a:		i = even, $j = i + 1 + k * 2$	
#4b,#5b:		i = odd, $j = i + 1 + k * 2$	
#7a, #8a:		i = even, $j = i + 1 + k * 4$	
#7b,#8b:		i = odd, $j = i + 1 + k * 4$	
#10a, #11a:		i = even, $j = i + 1 + k * 8$	
#10b,#11b:		i = odd, $j = i + 1 + k * 8$	

Level 0. As we can see from Table 3 and Fig. 4b, all uCFids between $(c_i, c_{i+2+k*4})$ are sensitized, where $k \in \{0, 1 \ldots, (\lfloor((B-1)-(i+2))/4\rfloor)\}$.

3. *Level 2*: For each cell-pair (c_i, c_{i+4}), we apply the DBS $\mathbf{S}''_2$. From Table 3 and Fig. 4c, we can see that all uCFids between $(c_i, c_{i+4+k*8})$ are sensitized, where $k \in \{0, 1 \ldots, (\lfloor((B-1)-(i+4))/8\rfloor)\}$.

After *Level 2*, all uCFids for an 8-bit WOM are sensitized. Table 3 shows the DBS $\mathbf{S}_8$ for 8-bit WOMs; it consists of concatenating the DBSs of *Level 0, Level 1*, and *Level 2*. Note that six different versions of $\mathbf{S}_8$ are possible because the *Level 0, Level 1*, and *Level 2* DB subsequences can be concatenated in six different ways (e.g., one other way is: *Level 2 - Level 0 - Level 1*). The method of generating a DBS for 8-bit WOMs can be generalized for B-bit WOMs, whereby d becomes: $d = 6$ {the d used in *Level 0*} $+3$ {the d used in each additional level} $*(\lceil \log_2 B \rceil - 1)$ {the number of levels} $= 3 + 3 * \lceil \log_2 B \rceil$. Table 2 gives an overview of the required d and the test length, as a function of B. Note: TL now represents the number of read and write operations

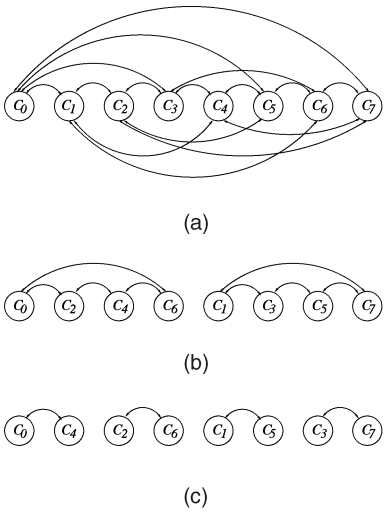


Fig. 4. Graph for sensitizing uCFids at different levels: (a) Level 0, (b) Level 1, (c) Level 2.

required to apply the DBS, excluding the all 0s value, which has alreday been verified by the BOM test.

3.3 Data Background Operation Sequence for Unrestricted CFdsts (uCFdsts)

An intraword test for uCFdsts, for the case that the uCFdst dominates the write operation, uses a sequence of data background operations; referred to as the *DB operation sequence* (*DBOS*). Eight uCFdsts subtypes exist [4], [5]: $< r0; \uparrow >_{c_a,c_v}$, $< r0; \downarrow >_{c_a,c_v}$, $< r1; \uparrow >_{c_a,c_v}$, $< r1; \downarrow >_{c_a,c_v}$, $< w0; \uparrow >_{c_a,c_v}$, $< w0; \downarrow >_{c_a,c_v}$, $< w1; \uparrow >_{c_a,c_v}$, and $< w1; \downarrow >_{c_a,c_v}$, where $a, v \in \{0, 1, \ldots, B-1\}$. Each subtype has $C_B^2 = B * (B-1)$ possible cases; the total number of uCFdsts is therefore $8 * B * (B-1)$. Similarly to the uCFid case, we have to find the minimal DBS which can sensitize the $8 * B * (B-1)$ uCFdsts; next, the minimal *number of operations using this DBS* (i.e., the minimal *DBOS*) has to be established.

Fig. 5 shows the state diagram for *sensitizing* the uCFdsts within a 2-bit WOM. The states (nodes) are numbered according to the value of the two cells c_0 and c_1 in the word; the arcs (which represent read, transition, and nontransition

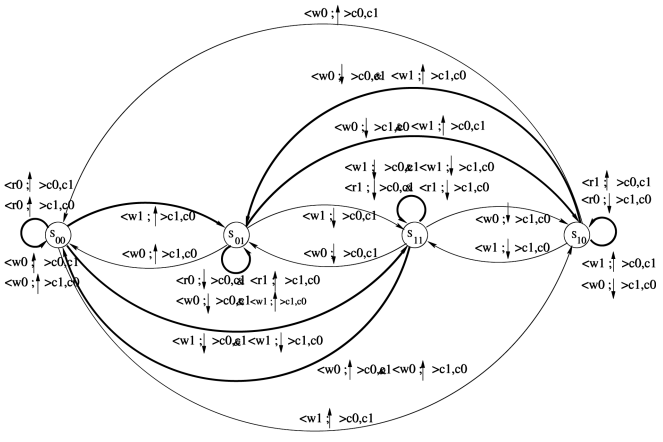


Fig. 5. State diagram for sensitizing uCFdsts ($B = 2$).

write operations) are labeled with the sensitized faults. From Fig. 5, we can see that:

1. A given uCFdst can be sensitized by different arcs. E.g., the uCFdst $< w0; \uparrow >_{c_0,c_1}$ can be sensitized by three arcs: (S_{00}, S_{00}), (S_{10}, S_{00}), or (S_{11}, S_{00}).
2. Each of the arcs formed by the states which are each other's inverse (i.e., (S_{00}, S_{11}), (S_{11}, S_{00}), (S_{01}, S_{10}), and (S_{10}, S_{01})) is labeled with two uCFdsts.
3. All arcs begining and ending in the same node are labeled with four uCFdsts.

From the above, it follows that all faults can be sensitized in more than one way, which means that the state diagram depicted in Fig. 5 can be reduced to contain the *bold* arcs only; requiring only nine arcs to sensitize all uCFdsts (for read and transition write operations). We have shown in [4] that other solutions exist; however, they are less efficient.

From Fig. 5, we can construct the DBS $\mathbf{S}_{t2}$, which depends only on *transition write* operations for W_2: $\mathbf{S}_{t2} = 00, 11, 00, 01, 10, 01$, which, similarly to the uCFds case, can be split as follows: $\mathbf{S}_{t2} = \mathbf{S}'_{t2}, \mathbf{S}''_{t2}$. Using this DBS sequence of *read and transition write operations* (Ω_{t2}, also called the *DBOS*) can be generated, assuming initial state S_{00}: $\Omega_{t2} = w11, r11, w00, r00, w01, w10, r10, w01, r01$. Thus, the number of operations needed to *sensitize* all uCFdsts using the DBS $\mathbf{S}_{t2}$ is nine. Note that the DBOS Ω_{t2} can also sensitize *all uCFids* because it is based on transition writes.

To *detect* all uCFdsts using the DBOS Ω_{t2}, each of the above sensitizing write *and* read operations has to be followed by a read operation; this read operation detects the uCFdst sensitized by the preceding write or read operation. The DBOS now becomes:

$$\Omega_{t2} = w11, r11, r11, r11, w00, r00, r00, r00, w01, \mathbf{r01}, w10, r10, r10, r10, w01, r01, r01, r01.$$

Ω_{t2} now contains sequences of three identical read operations. The first read operation is required to detect faults sensitized by the preceding write operation, the second read operation is required to detect the faults sensitized by the preceding read operation; hence, the third read operation is redundant and can be removed. The ninth operation $w01$ is needed only to connect two parts of the DBS (see Fig. 5), thus there is no need to check this write operation; therefore, the operation **r01** following this $w01$ operation can be removed. The DBOS Ω_{t2} can now be simplified to:

$$\Omega_{t2} = w11, r11, r11, w00, r00, r00, w01, w10, r10, r10, w01, r01, r01.$$

The number of operations needed to sensitize and detect all uCFdsts for a 2-bit WOM is now 13.

Extending the 2-bit DBS to a DBS for *B*-bit words, we can use the steps used for the uCFids:

1. *Level 0*: For each cell-pair (c_i, c_{i+1}), we apply the DBS found for 2-bit words. This can be done by applying the DBS $\mathbf{S}_{t2}$, replicated for *B*-bit words. All uCFdsts between $(c_i, c_{i+1+k*2})$ are sensitized, where $k \in \{0, 1 \ldots, \lfloor((B-1)-(i+1))/2\rfloor\}$.
2. *Level 1*: For each cell-pair (c_i, c_{i+2}), we apply only the DBS $\mathbf{S}''_{t2} = 01, 10, 01$; this is sufficient because the DBS $\mathbf{S}'_{t2} = 00, 11, 00$ has already been applied in *Level 0*. All uCFdsts between $(c_i, c_{i+2+k*4})$ are sensitized, where $k \in \{0, 1 \ldots, (\lfloor((B-1)-(i+2))/4\rfloor)\}$.
3. *Level 2*: For each cell-pair (c_i, c_{i+4}), we apply the DBS $\mathbf{S}''_{t2}$. All uCFdsts between $(c_i, c_{i+4+k*8})$ are sensitized, where $k \in \{0, 1 \ldots, (\lfloor((B-1)-(i+4))/8\rfloor)\}$.
4. $\log_2 B$. *Level* $\log_2 B - 1$: For each cell-pair $(c_i, c_{i+B/2})$, we apply the DBS $\mathbf{S}''_{t2}$. All uCFdsts between $(c_i, c_{i+(B/2)+k*B})$ are sensitized, where $k \in \{0, 1 \ldots, (\lfloor((B-1)-(i+B/2))/B\rfloor)\}$.

Extending the 2-bit DBOS to a DBOS for *B*-bit words requires the following steps:

1. *Level 0*: For each cell-pair (c_i, c_{i+1}), we generate the DBOS found for 2-bit words:

   $$\Omega_{t2} = w11, r11, r11, w00, r00, r00, w01, w10, r10, r10, w01, r01, r01.$$

   All uCFdsts between $(c_i, c_{i+1+k*2})$ are detected, where $k \in \{0, 1 \ldots, \lfloor((B-1)-(i+1))/2\rfloor\}$.
2. *Level 1*: For each cell-pair (c_i, c_{i+2}), we apply only the DBOS: $\Omega''_{t2} = w01, w10, r10, r10, w01, r01, r01$; this is sufficient because the DBOS: $\Omega'_{t2} = w11, r11, r11, w00, r00, r00$ has already been applied in *Level 0*. The first operation $w01$ in Ω''_{t2} does not have to be followed by a read operation because it is used only to connect the DBOSs of *Level 0* and *Level 1*. All uCFdsts between $(c_i, c_{i+2+k*4})$ are detected, where $k \in \{0, 1 \ldots, (\lfloor((B-1)-(i+2))/4\rfloor)\}$.
3. *Level 2*: For each cell-pair (c_i, c_{i+4}), we apply the DBOS Ω''_{t2} . All uCFdsts between $(c_i, c_{i+4+k*8})$ are detected, where $k \in \{0, 1 \ldots, (\lfloor((B-1)-(i+4))/8\rfloor)\}$.
4. $\log_2 B$. *Level* $\log_2 B - 1$: For each cell-pair $(c_i, c_{i+B/2})$, we apply the DBOS Ω''_{t2}. All uCFdsts between $(c_i, c_{i+(B/2)+k*B})$ are detected, where $k \in \{0, 1 \ldots, (\lfloor((B-1)-(i+B/2))/B\rfloor)\}$.

TABLE 4
DBS for CFdsts ($B = 4$)

#	DB $c_0c_1c_2c_3$	Level
0	0 0 0 0	0
1	1 1 1 1	0
2	0 0 0 0	0
3	0 1 0 1	0
4	1 0 1 0	0
5	0 1 0 1	0
6	0 0 1 1	1
7	1 1 0 0	1
8	0 0 1 1	1

Table 4 and Table 5 show, respectively, the DBS and the DBOS for 4-bit words, requiring only *Level 0* and *Level 1*, as discussed above. The value of d for *B*-bit words is: $d = 6$ {the d used in *Level 0*} $+ 3$ {the d used in the other levels} $*(\lceil \log_2 B \rceil - 1)$ {the number of levels} $= 3 + 3 * \lceil \log_2 B \rceil$, which is identical to the d of uCFids. The test length (TL) for

TABLE 5
DBOS for CFdsts ($B = 4$)

#	Op.	DB $c_0c_1c_2c_3$	Lev.	#	Op.	DB $c_0c_1c_2c_3$	Lev.
0		0 0 0 0	0	11	w	0 1 0 1	0
1	w	1 1 1 1	0	12	r	0 1 0 1	0
2	r	1 1 1 1	0	13	r	0 1 0 1	0
3	r	1 1 1 1	0	14	w	0 0 1 1	1
4	w	0 0 0 0	0	15	w	1 1 0 0	1
5	r	0 0 0 0	0	16	r	1 1 0 0	1
6	r	0 0 0 0	0	17	r	1 1 0 0	1
7	w	0 1 0 1	0	18	w	0 0 1 1	1
8	w	1 0 1 0	0	19	r	0 0 1 1	1
9	r	1 0 1 0	0	20	r	0 0 1 1	1
10	r	1 0 1 0	0				

B-bit words is: 13 {the number of operations used in *Level 0*} + 7 {the number of operations used in the other levels} $*(\lceil \log_2 B \rceil - 1) = 6 + 7 * \lceil \log_2 B \rceil$, assuming the all 0s initial state of the memory. Table 2 gives an overview of the required number of DBs (d) and the TL as a function of B.

3.4 WOM March Tests for Intraword Coupling Faults

Any given BOM march test can be converted into a WOM test which additionally covers intraword CFs (uCFsts, uCFidsm and/or uCFdsts). Such a WOM march test is a concatenation of two march tests: {*interword* march test} {*intraword* march test}. The *interword* march test consists of a traditional BOM test (such as MATS+ or March C–), modified such that the bit-operations "r0," "r1," "w0," and "w1" are replaced with the word-operations "r_D," "$r_{\bar{D}}$," "w_D," and "$w_{\bar{D}}$," whereby the all 0s DB value has been chosen for D to optimize ground bounce. The *intraword* march test is used to detect the intraword CFs. It consists of a single march element of the following form:

1. For uCFsts: $\Updownarrow (w_{D_0}, r_{D_0}, \ldots, w_{D_{d-1}}, r_{D_{d-1}})$, whereby D_0 through D_{d-1} are taken from the set of DBs of Table 1 (for $B = 8$) in such a way that both the normal and inverse values are covered. Note that the DBs can be applied in any order.
2. For uCFids:

$$\Updownarrow (w_{D_0}, r_{D_0}, w_{D_1}, r_{D_1}, \ldots, r_{D_{d-2}}, w_{D_{d-1}}, r_{D_{d-1}}),$$

   whereby D_0 through D_{d-1} represent the DBS of Table 3 (for $B = 8$).
3. For uCFdsts: $\Updownarrow (w_{D_0}, r_{D_0}, r_{D_0}, ..., w_{D_{d-1}}, r_{D_{d-1}}, r_{D_{d-1}})$, whereby the DBOS (consisting of the operations together with the DBs) is taken from Table 5 (for $B = 4$).

The above intraword test may be modified as follows, without any impact on the fault coverage:

$$\{\Updownarrow_0 (w0000); \Uparrow_1 (r0000, w11111); \Uparrow_2 (r1111, w0000);$$
$$\Downarrow_3 (r0000, w1111); \Downarrow_4 (r1111, w0000); \Updownarrow_5 (r0000);$$
$$\Downarrow_6 (\mathbf{r0000}, w1111); \Downarrow_7 (r1111, w0000);$$
$$\Downarrow_8 (r0000, w0101); \Uparrow_9 (r0101, w1010);$$
$$\Downarrow_{10} (r1010, w0101); \Uparrow_{11} (r0101, w0011);$$
$$\Downarrow_{12} (r0011, w1100); \Uparrow_{13} (r1100, w0011); \Updownarrow_{14} (r0011)\}$$

Fig. 6. WOM march test for uCFids for $B = 4$, based on March C–.

1. Extra read operations may be added, for example, to make the test more symmetric and/or to detect possible faults of other fault models.
2. The single march element may be divided into any number of march elements and, for each march element, the address order can be chosen freely.

The above freedom to modify the intraword test allows for:

1. Test time reduction. If march elements of the intraword test can be made identical to march elements of the interword test, those intraword march elements can be removed.
2. Extra fault coverage for unanticipated faults. This can be optimized when the intraword march test has the following properties:
   a. It consists of several march elements because each march element performs a sweep over the memory.
   b. All march elements start with a read; this allows for the detection of CFs.
   c. The adress orders of the march elements of the intraword march test should vary as much as possible. This maximizes the probability of detecting dynamic faults such as write recovery faults [2].

Two examples are given to show how BOM march tests can be converted into optimized WOM march tests.

Fig. 6 shows how to convert March C– $\{\Updownarrow (w0); \Uparrow (r0, w1); \Uparrow (r1, w0); \Downarrow (r0, w1); \Downarrow (r1, w0); \Updownarrow (r0)\}$ such that uCFids are detected for a 4-bit WOM. Note: The subscript of the address order denotes the march element number. The interword march test consists of the first six march elements (M_0 through M_5); while the intraword march test is designed such that it consists of nine march elements whereby the similarity with the march elements of the interword test has been maximized. An extra initial read operation has been included in M_6 of the intraword test, while M_{14} only consists of a single operation to verify the write operation of M_{13}. Because $B = 4$, only the cells c_0 through c_3 and a DBS of nine DBs (numbered 0 through 8; representing Level 0 and Level 1) of Table 3 are used. From Fig. 6, one can see that the march elements M_6 and M_7 are redundant ($M_6 = M_3$ and $M_7 = M_4$) and march element M_5 can be deleted because the "r0000" in M_8 can be used to verify the "w0000" in M_4. The removal of march elements M_5, M_6, and M_7 does not change the fault coverage of the intraword test because the DBS of Table 3 is still applied correctly. The optimized version of the WOM march test based on

$$\{\Updownarrow_0\ (w0000); \Uparrow_1\ (r0000, w11111); \Uparrow_2\ (r1111, w0000);$$
$$\Downarrow_3\ (r0000, w1111); \Downarrow_4\ (r1111, w0000);$$
$$\Downarrow_5\ (r0000, w0101); \Uparrow_6\ (r0101, w1010); \Downarrow_7\ (r1010, w0101);$$
$$\Uparrow_8\ (r0101, w0011); \Downarrow_9\ (r0011, w1100);$$
$$\Uparrow_{10}\ (r1100, w0011); \Updownarrow_{11}\ (r0011)\}$$

Fig. 7. Optimized WOM march test for uCFids for $B = 4$, based on March C–.

March C– is shown in Fig. 7. TL changes from $10 * n$ (for the BOM test) to $22 * n/4$ for a 4-bit WOM. In general, the TL for a B-bit memory will be: $(10 + 6 * \lceil \log_2 B \rceil) * n/B$. Note that the optimization reduces TL from $(10 * n/4) + (20 * n/4)$ to $22 * n/4$, which is by 26.7 percent.

Fig. 8 shows how to convert March LR

$$\{\Updownarrow\ (w0); \Downarrow\ (r0, w1); \Uparrow\ (r1, w0, r0, w1); \Uparrow\ (r1, w0);$$
$$\Uparrow\ (r0, w1, r1, w0); \Updownarrow\ (r0)\}$$

[8] such that uCFdsts will be detected for a 4-bit WOM. The conversion can be made as follows:

1. The read operation **r1** is added to march element M_4; the result is the march test:

$$\{\Updownarrow\ (w0); \Downarrow\ (r0, w1); \Uparrow\ (r1, w0, r0, w1); \Uparrow\ (r1, w0);$$
$$\Uparrow\ (r0, w1, r1, \mathbf{r1}, w0); \Updownarrow\ (r0)\}.$$

   This extra read operation does not affect the fault coverage of the BOM march test.
2. Convert the obtained BOM march test into a 4-bit interword march test (M_0 through M_5).
3. Concatenate the interword march with the intraword march test. The latter is obtained from Fig. 8, while the operations are divided over the march elements in such a way that M_6 and M_7 are compatible with the BOM march test. In addition, an extra read operation has been added to M_6.
4. Optimize the resulting WOM march test by deleting the redundant march elements of the intraword test (i.e., the sequence of operations in M_6 and M_7 has already been used in M_4 and M_5). The optimized version of the WOM march test based on March LR is shown in Fig. 9. The optimization reduces the TL from $(14 * n/4 + 20 * n/4)$ to $30 * n/4$, which is by 11.7 percent. In general, for a B-bit memory, the TL will be: $(16 + 7 * \lceil \log_2 B \rceil) * n/B$.

$$\{\Updownarrow_0\ (w0000); \Downarrow_1\ (r0000, w11111);$$
$$\Uparrow_2\ (r1111, w0000, r0000, w1111); \Uparrow_3\ (r1111, w0000);$$
$$\Uparrow_4\ (r0000, w1111, r1111, \mathbf{r1111}, w0000); \Updownarrow_5\ (r0000);$$
$$\Uparrow_6\ (\mathbf{r0000}, w1111, r1111, r1111, w0000); \Downarrow_7\ (r0000);$$
$$\Uparrow_8\ (r0000, w0101, w1010, r1010); \Downarrow_9\ (r1010, w0101, r0101);$$
$$\Uparrow_{10}\ (r0101, w0011, w1100, r1100);$$
$$\Downarrow_{11}\ (r1100, w0011, r0011); \Uparrow_{12}\ (r0011)\}$$

Fig. 8. WOM march test for uCFdsts and $B = 4$, based on March LR.

4 Relationships of WOM March Tests and the Impact of the Memory Organization

As has already been shown for BOM tests [2], WOM tests can also be designed such that a test for a more complex fault type covers the faults of the simpler fault types. Fig. 10 shows the hierarchy in fault coverage of the discussed WOM tests (Note: The notation "*WTSAFs*" denotes a *W*OM *T*est for an *SAFs*, etc.):

1. WOM tests for uCFsts, uCFids, and uCFdsts cover SAFs, TFs, and RDFs. This is due to the fact that the use of any DB value (D) and its inverse ($\bar{D}$) value will detect those faults. The set of DBs for uCFsts (see Table 1), the DBS for uCFids (see Table 3), and the DBOS for uCFdsts (see Table 5) all contain a DB value D and its inverse $\bar{D}$.
2. WOM tests for uCFids cover all uCFsts. Sensitizing uCFids requires $\uparrow$ and $\downarrow$ transition write operations to an aggressor cell a, while the victim cell v is in state 0 and 1. This means that all four states will occur, i.e., the states: $(a, v) \in \{(0,0), (0,1), (1,0), (1,1)\}$. This guarantees the detection of all uCFsts. Also, by inspection, one can see that the set of DBs of Table 1 is a subset of the DBS of Table 3.
3. WOM tests for uCFdsts cover all uCFids. The DBS $\mathbf{S}_t$ (of Section 3.3), based on the transition write operations (see Fig. 5 and Table 5), will automatically sensitize all uCFids. This also can be verified by inspecting Table 3, which contains the same DBS as Table 5. Hence, WOM tests for uCFdsts cover all uCFids, uCFsts, RDFs, TFs, and SAFs.

Generally, a memory chip with a size of n bits, denoted by M_n, may consist of s identical two-dimensional subarrays of memory cells. Each subarray M_m contains m bits; $m = q * r$, where q is the number of columns and r the number of rows (i.e., $n = s * m = s * q * r$). Multiple subarrays are used instead of one single array to shorten the word and bit lines and thereby reduce the access time. WOMs can be

$$\{\Updownarrow_0\ (w0000); \Downarrow_1\ (r0000, w11111);$$
$$\Uparrow_2\ (r1111, w0000, r0000, r0000, w1111); \Uparrow_3\ (r1111, w0000); \Uparrow_4\ (r0000, w1111, r1111, r1111, w0000);$$
$$\Uparrow_5\ (r0000, w0101, w1010, r1010); \Downarrow_6\ (r1010, w0101, r0101); \Uparrow_7\ (r0101, w0011, w1100, r1100);$$
$$\Downarrow_8\ (r1100, w0011, r0011); \Uparrow_9\ (r0011)\}$$

Fig. 9. Optimized WOM march test for uCFdsts and $B = 4$, based on March LR.

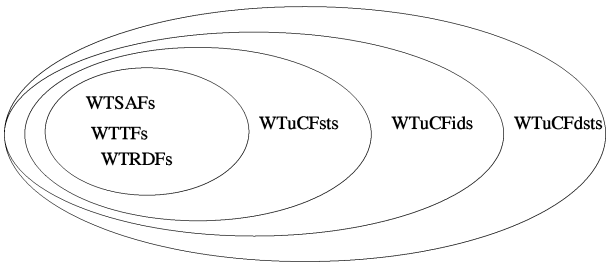


Fig. 10. Relationships between the uCF tests.

organized internally in many different ways (depending on where the B bits of a word are physically located within a row of the memory cell array):

1. *Adjacent*: A *q*-bit row in a subarray contains $w * B$ bits. The B bits of a word are adjacent and, therefore, the proposed WOM tests for uCFsts, uCFids, and uCFdsts have to be applied.
2. *Interleaved* (also called: *folded*): A *q*-bit row in a subarray contains $w * B$ bits. The B bits of a word are spread across B groups [9], [10] in such a way that the bits of a *B*-bit word are interleaved with (i.e., separated by) $w - 1$ bits of the other *B*-bit words in that row. Therefore, only the test for uCFsts has to be applied to one word of each fold in order to verify the I/O data path of each fold. Note: The uCFst fault model is adequate for checking CFs in data paths.
3. *Subarrays*: Each bit of a *B*-bit word is taken from a different subarray, while all B bits have the same address in each subarray [11]. Similarly to the interleaved case, only a test for uCFsts has to be applied to one word in each subarray, in order to verify the I/O data path.

5 Tests for Restricted Intraword Coupling Faults

The proposed tests for uCFs assume that any single a-cell in a word can influence a v-cell in the same word. Therefore, those tests do not require knowledge about the layout of the cells in the *row*. This section proposes the following fault models for CFs, assuming that the layout of the cells in a row is known: *restricted CFs (rCFs)*, whereby the v-cell can only be influenced by *a single a-cell*, which is the physical neighbor of the v-cell, and *concurrent-restricted CFs (crCFs)*, whereby the v-cell can only be influenced by the concurrent action or state of *two a-cells*, which are the physical neighbors of the v-cell. The use of the rCF and crCF fault models results in more time-efficient tests.

5.1 Restricted CFs (rCFs)

One reasonable restriction of intraword CFs (similar to the restriction used in [10]) is that an a-cell can only influence its left or its right physical neighbor. These *restricted* CFs, denoted as *rCFs*, apply to the following fault types: rCFsts, rCFids, and rCFdsts. The rCFs naturally require knowledge of the physical layout of the bits in a row. The way BOM tests can be converted into WOM tests for rCFs depends on the physical memory organization (see Section 4):

TABLE 6
Set of 8-bit DBs for rCFsts

#	Normal	#	Inverse
0	00000000	1	11111111
2	01010101	3	10101010

1. **Adjacent**
   rCFsts: In case of rCFsts, which dominate the write operation, all states of adjacent cells should be checked, i.e., the states $(i, i+1) \in \{(0,0),(0,1),(1,0),(1,1)\}$. The set of DBs which satisfy this condition is given in Table 6; this reduces the size of the set of DBs from $2 * (\lceil \log_2 B \rceil + 1)$ to 4, i.e, $d = 4$ and does not depend on B. $TL = 4$, assuming that the all 0s and all 1s DBs have already been verified by the BOM test.
   rCFids: In case of rCFids, which dominate the write operation, the minimal DBS (for $B = 8$) needed to sensitize all rCFids is the DBS of Table 3 for *Level 0*; it can sensitize all CFids between adjacent cells (see Fig. 4a) and, thus, can sensitize all rCFids. The rCFid intraword fault model reduces the size of the DBS from $3 * (1 + \lceil \log_2 B \rceil)$ to $d = 6$ and TL to 10; they do not depend on B.
   rCFdsts: In order to be able to sensitize all rCFdsts, only the DBOS for *Level 0* (see Table 5) should be applied. This is sufficient because this DBOS can sensitize all CFdsts between adjacent cells. This reduces d to 6 and TL to 13, independent of B.
2. **Interleaved (folded)**
   In the case of the interleaved organization, the B bits of a word are spread across B groups (see Section 4) such that there are no neighboring cells belonging to the same word. This means that there are no adjacent cells within a word. One only has to check the data I/O paths of each fold by applying the test for rCFsts to one word in each fold.

5.2 Concurrent-Restricted CFs (crCFs)

WOMs are read and written B bits at a time, i.e., B bits concurrently. Conceptually, $B - 1$ bits in a word may concurrently act as a-cells for a given v-cell. When the physical layout of the cells in a row is known, tests can be simplified considerably when the a-cells are restricted to be the two physical neighbors of the v-cell, which are the most likely a-cells. This coupling fault model will be called the *concurrent-restricted CF model* (*crCF model*). A crCF may occur in case of a read or write operation to three adjacent cells of the same word (i.e., c_{a1}, c_v, and c_{a2}), see Fig. 11. These read or write operations to the two a-cells can modify the value of the in-between v-cell [10]. The crCF model has to be analyzed for the fault types: the *concurrent-restricted CFst (crCFst)*, the *concurrent-restricted CFid (crCFid)*, and the *concurrent-restricted CFdst (crCFdst)*.

In order to detect crCFsts, all states of three adjacent cells ($i - 1$, i, and $i + 1$) should be checked, i.e., the states

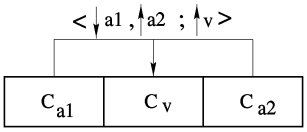


Fig. 11. crCFid between adjacent cells.

$$(i-1,i,i+1) \in \{(0,0,0),(0,0,1),(0,1,0),(0,1,1),\\ (1,0,0),(1,0,1),(1,1,0),(1,1,1)\}.$$

This set of DBs sensitizes all crCFsts between three adjacent cells; it has to be applied to each group of three adjacent cells of a B-bit word. A set of 8-bit DBs for crCFsts is given in Table 7. The size of the set is 8 ($d = 8$) and $TL = 12$ (assuming the all 0s and all 1s DBs have already been verified by the BOM test), they are independent of B.

For the crCFid fault type, eight subtypes exist: $<\uparrow,\uparrow;\uparrow>_{c_{a1},c_{a2},c_v}$ (note: c_{a1}, c_{a2} denote a-cells and c_v denotes the v-cell), $<\uparrow,\uparrow;\downarrow>_{c_{a1},c_{a2},c_v}$, $<\uparrow,\downarrow;\uparrow>_{c_{a1},c_{a2},c_v}$, $<\uparrow,\downarrow;\downarrow>_{c_{a1},c_{a2},c_v}$, $<\downarrow,\uparrow;\uparrow>_{c_{a1},c_{a2},c_v}$, $<\downarrow,\uparrow;\downarrow>_{c_{a1},c_{a2},c_v}$, $<\downarrow,\downarrow;\uparrow>_{c_{a1},c_{a2},c_v}$, and $<\downarrow,\downarrow;\downarrow>_{c_{a1},c_{a2},c_v}$, where $a1 = i-1$, $a2 = i+1$, $v = i$, and $i \in \{1,2,\ldots,B-2\}$. Fig. 12 shows a state diagram for *sensitizing* the crCFids for three adjacent cells c_0, c_1, and c_2. The states (nodes) are numbered according the values of the cells c_0, c_1, and c_2; the arcs (which represent the transitions from one state to another) are labeled with the sensitized faults (note: a sensitized fault whereby an a-cell does not change state is denoted by the "-" symbol for that a-cell). From Fig. 12, we can see that all eight crCFids are sensitized. Furthermore, the arcs (S_{000}, S_{001}), (S_{001}, S_{010}), and (S_{010}, S_{011}) do not sensitize crCFids; however, they are used to construct the DBS needed to sensitize all crCFids. Other state diagrams are possible to construct the DBS, but one has to take into account that the write operations applied to all the three cells are transition write operations because, for a B-bit word, each cell may act as a-cell as well as v-cell. A DBS for *three* adjacent cells which can sensitize all crCFids is given in Table 8. To sensitize all crCFids within a B-bit word, the DBS of Table 8 has to be applied to each group of three adjacent bits of a B-bit word. Extending Table 8 for $B > 3$ is done using the state diagram of Fig. 12; this does not increase the number of rows in the table. Table 9 shows an 8-bit DBS for crCFids, together with the operations to be performed; the write operations, w, change the state of the memory word, while read operations, r, are only required for those write operations which sensitize faults (i.e., for the vertical arcs in Fig. 12). The number of the DBs, d, is 12 and TL is 19 (assuming the all 0s initial state); independent of B.

TABLE 7
Set of 8-Bit DBs for crCFsts

#	Normal	#	Inverse
0	00000000	1	11111111
2	00100100	3	11011011
4	01001001	5	10110110
6	01101101	7	10010010

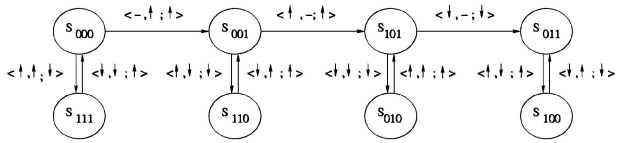


Fig. 12. State diagram for sensitizing crCFids for a 3-bit word.

For the crCFdst fault type, 16 subtypes exist: $< w_x, w_y; \updownarrow >_{c_{a1},c_{a2},c_v}$ and $< r_x, r_y; \updownarrow >_{c_{a1},c_{a2},c_v}$, where $x, y \in \{0,1\}$ and $\updownarrow$ is an $\uparrow$ or a $\downarrow$ transition. Fig. 13 shows a state diagram for sensitizing the crCFdsts, based on transition write operations for three adjacent cells c_0, c_1, c_2. The DBS for sensitizing the crCFdsts is derived from Fig. 13 and, similarly to Section 3.3, we can construct the DBOS to sensitize and detect all rcCFdsts (see Table 10). The number of the DBs, d, is 12 and TL is 27 (assuming the all 0s initial state); both are independent of B.

For an adjacent memory organization, the proposed tests for crCFsts, crCFids, and/or crCFdsts have to be applied. For the interleaved (folded) and the subarray organizations, only the data I/O paths of each fold/subarray have to be verified, using a test for crCFsts. *Note*: Because of the complex wiring of the I/O data paths and/or because of possible repair (by replacing bad rows and/or columns), usually one does not use (concurrent) restricted fault models for the I/O data paths in a fold/subarray, but only the uCFst fault model.

5.3 Examples of WOM March Tests for rCFs and crCFs

The method to convert a BOM march test into a WOM march test to detect rCFs and crCFs is the same as used in Section 3.4; below, four examples are given for $B = 4$.

1. WOM march tests, based on March C–, for rCFids and for crCFids. Fig. 14 shows the WOM version of March C– to detect rCFids. March elements M_0 through M_4, together with the $r0000$ operation of M_5,

TABLE 8
3-Bit DBS for crCFids

#	DB $c_{a1}c_vc_{a2}$	Faults detected
0	0 0 0	
1	1 1 1	$<\uparrow,\uparrow;\downarrow>_{c_{a1},c_{a2};c_v}$
2	0 0 0	$<\downarrow,\downarrow;\uparrow>_{c_{a1},c_{a2};c_v}$
3	0 0 1	$<-,\uparrow;\uparrow>_{c_{a1},c_{a2};c_v}$
4	1 1 0	$<\uparrow,\downarrow;\downarrow>_{c_{a1},c_{a2};c_v}$
5	0 0 1	$<\downarrow,\uparrow;\uparrow>_{c_{a1},c_{a2};c_v}$
6	1 0 1	$<\uparrow,-;\uparrow>_{c_{a1},c_{a2};c_v}$
7	0 1 0	$<\downarrow,\downarrow;\downarrow>_{c_{a1},c_{a2};c_v}$
8	1 0 1	$<\uparrow,\uparrow;\uparrow>_{c_{a1},c_{a2};c_v}$
9	0 1 1	$<\downarrow,-;\downarrow>_{c_{a1},c_{a2};c_v}$
10	1 0 0	$<\uparrow,\downarrow;\uparrow>_{c_{a1},c_{a2};c_v}$
11	0 1 1	$<\downarrow,\uparrow;\downarrow>_{c_{a1},c_{a2};c_v}$

TABLE 9
8-Bit DBS for crCFids

#	DB	Operation
	$c_0c_1c_2c_3c_4c_5c_6c_7$	
0	0 0 0 0 0 0 0 0	-
1	1 1 1 1 1 1 1 1	w, r
2	0 0 0 0 0 0 0 0	w, r
3	0 0 1 0 0 1 0 0	w
4	1 1 0 1 1 0 1 1	w, r
5	0 0 1 0 0 1 0 0	w, r
6	1 0 1 1 0 1 1 0	w
7	0 1 0 0 1 0 0 1	w, r
8	1 0 1 1 0 1 1 0	w, r
9	0 1 1 0 1 1 0 1	w
10	1 0 0 1 0 0 1 0	w, r
11	0 1 1 0 1 1 0 1	w, r

represent the interword test. M_5 through M_{10} represent the optimized intraword test for rCFids. It applies the Level 0 DBS of Table 3 for $B = 4$. The total test length, TL, is $20 * n/B$. Fig. 15 shows the WOM version of March C– to detect crCFids. M_5 through M_{10} represent the intraword test for crCFids. It is based on the DBS of Table 9 for $B = 4$. The total TL is $29 * n/B$.

2. WOM march tests, based on March LR, for rCFdsts and for crCFdsts. Fig. 16 shows the WOM version of March LR to detect rCFdsts. March element M_0 through M_3, together with the $r0000$ operation of M_4, represent the interword test. M_4 through M_{10} represent the optimized test for intraword rCFdsts. It applies the Level 0 DBOS of Table 5 for $B = 4$. The total TL is $36 * n/B$. Fig. 17 shows the WOM version of March LR to detect crCFdsts. M_5 through M_{13} represent the intraword test for crCFdsts. It is based on the DBOS of Table 10. The total test length, TL, is $42 * n/B$.

6 Conclusions

This paper has presented a systematic set of intraword coupling fault models, based on CFsts, CFids, and CFdsts. Depending on the restrictions applied to the a-cell(s), they consist of the following subtypes: uCFs (no restrictions), rCFs (a single a-cell, which is the physical neighbor of the v-cell), and crCFs (two a-cells, which are physically neighboring the v-cell).

TABLE 10
DBOS for crCFdsts, $B = 4$

#	Op.	DB	#	Op.	DB
		$c_0c_1c_2c_3$			$c_0c_1c_2c_3$
0		0 0 0 0	14	w	0 1 0 0
1	w	1 1 1 1	15	w	1 0 1 1
2	r	1 1 1 1	16	r	1 0 1 1
3	r	1 1 1 1	17	r	1 0 1 1
4	w	0 0 0 0	18	w	0 1 0 0
5	r	0 0 0 0	19	r	0 1 0 0
6	r	0 0 0 0	20	r	0 1 0 0
7	w	0 0 1 0	21	w	0 1 1 0
8	w	1 1 0 1	22	w	1 0 0 1
9	r	1 1 0 1	23	r	1 0 0 1
10	r	1 1 0 1	24	r	1 0 0 1
11	w	0 0 1 0	25	w	0 1 1 0
12	r	0 0 1 0	26	r	0 1 1 0
13	r	0 0 1 0	27	r	0 1 1 0

Table 11 lists the equations for d and the TL and the value pair "d;TL" for a memory with $B = 4$ and $B = 32$, as a function of the intraword CF type. For the values of TL, it is assumed the the BOM test already has written and read the all 0s and all 1s DB values.

The *traditional* method of converting a BOM test into a WOM test [1], [2], [3], by repeating the BOM test for each complementary pair of DB values, can only be applied using the uCFst fault model. In addition, it was inefficient because the BOM test, designed for single-cell faults and interword faults, was repeated $d/2$ times such that the total test time became the *product* of the "(BOM test time) * $(d/2)$." In addition, the traditional method cannot be used for detecting intraword CFids and CFdsts because that requires the use of a sequence of DBs (i.e., a DBS) or a sequence of DB operations (i.e., a DBOS), respectively.

The new, *improved* method solves the problems of test time inefficiency and intraword CF fault coverage by concatenating to the test for interword faults, a test for intraword faults. This results in a test time which is the *sum* of the "(BOM test time) + (test time for intraword CFs)"; while, in addition, it allows for the independent selection of

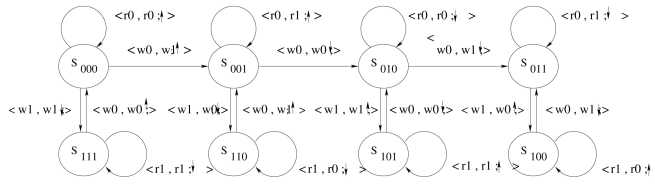


Fig. 13. State diagram for sensitizing crCFdsts for 3-bit word.

$$\{\Updownarrow_0\ (w0000); \Uparrow_1\ (r0000, w11111); \Uparrow_2\ (r1111, w0000);$$
$$\Downarrow_3\ (r0000, w1111); \Downarrow_4\ (r1111, w0000);$$
$$\Downarrow_5\ (r0000, w1111); \Uparrow_6\ (r1111, w0000); \Downarrow_7\ (r0000, w0101);$$
$$\Uparrow_8\ (r0101, w1010); \Downarrow_9\ (r1010, w0101);$$
$$\Updownarrow_{10}\ (r0101)\}$$

Fig. 14. WOM march test for rCFids, based on March C–.

$$\{\Updownarrow_0\ (w0000); \Uparrow_1\ (r0000, w11111); \Uparrow_2\ (r1111, w0000);$$
$$\Downarrow_3\ (r0000, w1111); \Downarrow_4\ (r1111, w0000); \Downarrow_5\ (r0000, w1111); \Uparrow_6\ (r1111, w0000);$$
$$\Downarrow_7\ (r0000, w0010, w1101); \Uparrow_8\ (r1101, w0010); \Downarrow_9\ (r0010, w1011, w0100);$$
$$\Uparrow_{10}\ (r0100, w1011); \Updownarrow_{11}\ (r1011, w0110, w1001);$$
$$\Uparrow_{12}\ (r1001, w0110); \Updownarrow_{13}\ (r0110)\}$$

Fig. 15. WOM march test for crCFids, based on March C–.

$$\{\Updownarrow_0\ (w0000); \Downarrow_1\ (r0000, w11111);$$
$$\Uparrow_2\ (r1111, w0000, r0000, r0000, w1111); \Uparrow_3\ (r1111, w0000); \Uparrow_4\ (r0000, w1111, r1111, r1111, w0000);$$
$$\Uparrow_5\ (r0000, w1111, r1111); \Downarrow_6\ (r1111, w0000, r0000); \Uparrow_7\ (r0000, w0101, w1010, r1010);$$
$$\Downarrow_8\ (r1010, w0101, r0101); \Uparrow_9\ (r0101, w0011, w1100, r1100); \Downarrow_{10}\ (r1100, w0011, r0011); \Updownarrow_{11}\ (r0011)\}$$

Fig. 16. WOM march test for rCFdsts, based on March LR.

$$\{\Updownarrow_0\ (w0000); \Downarrow_1\ (r0000, w11111);$$
$$\Uparrow_2\ (r1111, w0000, r0000, r0000, w1111); \Uparrow_3\ (r1111, w0000); \Uparrow_4\ (r0000, w1111, r1111, r1111, w0000);$$
$$\Uparrow_5\ (r0000, w1111, r1111); \Downarrow_6\ (r1111, w0000, r0000);$$
$$\Uparrow_7\ (r0000, w0010, w1101, r1101); \Downarrow_8\ (r1101, w0010, r0010);$$
$$\Uparrow_9\ (r0010, w0100, w1011, r1011); \Downarrow_{10}\ (r1011, w0100, r0100);$$
$$\Uparrow_{11}\ (r0100, w1001, r1001); \Downarrow_{12}\ (r1001, w0110, r0110);$$
$$\Updownarrow_{13}\ (r0110)\}$$

Fig. 17. WOM march test for crCFdsts, based on March LR.

single-cell and interword CFs and intraword CFs (which may now also consist of CFids and/or CFdsts).

Fig. 18 shows the WOM test lengths using the uCFst fault model for memories with $B = 8$, $B = 32$, and $B = 128$ and using March C– (this is a $10 * n$ BOM test) and March LA [12] (this is a $22 * n$ BOM test). The *traditional* method using March C– and March LA is indicated by "TraC–" and "TraLA," while the *improved* method using these two tests is indicated by "ImpC–" and "ImpLA." The figure shows that the test length increases with increasing values of B; the traditional method rapidly increases because its test length is proportional to the product: ($log_2 B$) * (the BOM test time). The *improved* method is *much more time efficient* because its test length is proportional to the sum: ($log_2 B$) + (the BOM test time)!

TABLE 11
Number of DBs and TLs for Different CF Types

CF type	# of DBs:d	TL	d;TL	
			$B = 4$	$B = 32$
uCFst	$2 * \beta + 2$	$4 * \beta$	6;8	12;20
rCFst	4	4	4;4	4;4
crCFst	8	12	8;12	8;12
uCFid	$3 * \beta + 3$	$6 * \beta + 4$	9;16	18;34
rCFid	6	10	6;10	6;10
crCFid	12	22	12;22	12;22
uCFdst	$3 * \beta + 3$	$7 * \beta + 6$	9;20	18;41
rCFdst	6	13	6;13	6;13
crCFdst	12	27	12;27	12;27

Note: $\beta = \lceil \log_2 B \rceil$.

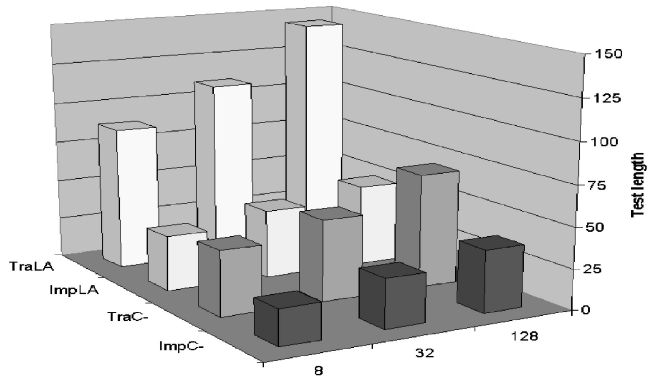


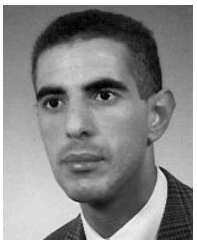
Fig. 18. Test length comparison of traditional and improved method.

REFERENCES

[1] R. Dekker, "A Realistic Fault Model and Test Algorithms for Static Random Access Memories," *IEEE Trans. Computers*, vol. 39, no. 6, pp. 567-572, June 1990.
[2] A.J. van de Goor, *Testing Semiconductor Memories, Theory and Practice.* Gouda, The Netherlands: ComTex Publishing, http://ce.et.tudelft.nl/~vdgoor/, 1998.
[3] R.P. Treuer and V.K. Agarwal, "Fault Location Algorithms for Repairable Embedded RAMs," *Proc. IEEE Int'l Test Conf.*, pp. 825-834, 1993.

[4] I.B.S. Tlili and A.J. van de Goor, "Tests for Word-Oriented Memories," Technical Report No. 1-68340-44(1997)08, Dept. of Electrical Eng., Delft Univ. of Technology, Delft, The Netherlands, 1997.
[5] A.J. van de Goor and I.B.S. Tlili, "March Tests for Word-Oriented Memories," *Proc. Design Automation and Test in Europe,* pp. 501-508, 1998.
[6] A.J. van de Goor, I.B.S. Tlili, and S. Hamdioui, "Converting March Tests for Bit-Oriented Memories into Tests for Word-Oriented Memories," *Records IEEE Int'l Workshop Memory Technology, Design and Testing,* pp. 46-52, 1998.
[7] R.D. Adams, "Extentions of Static Random Access Memory Fault Modeling and Examination of Patterns for Fault Detection," master's thesis, Thayer School of Eng., May 1996.
[8] A.J. van de Goor, G.N. Gaydadjiev, V.N. Yarmolik, and V.G. Mikitjuk, "March LR: A Test for Realistic Linked Faults," *Proc. 14th VLSI Test Symp.,* pp. 272-280, 1996.
[9] P. Mazumder and J.H. Patel, "Parallel Testing for Pattern-Sensitive Faults in Semiconductor Random-Access Memories," *IEEE Trans. Computers,* vol. 38, no. 3, pp. 394-407, Mar. 1989.
[10] M. Nicolaidis, V. Castro Alves, and H. Bederr, "Testing Complex Coupling Faults in Multiport Memories," *IEEE Trans. VLSI Systems,* vol. 3, no. 1, pp. 59-71, Mar. 1995.
[11] B. Prince, *Semiconductor Memories.* Chichester, U.K.: John Wiley & Sons, 1991.
[12] A.J. van de Goor, G.N. Gaydadjiev, V.N. Yarmolik, and V.G. Mikitjuk, "March LA: A Test for Linked Memory Faults," *Proc. European Design and Test Conf.,* pp. 627-627, 1997.

Ad J. van de Goor received the MSEE degree from the Delft University of Technology, Delft, The Netherlands, in 1965. He received another MSEE degree and the PhD degree from Carnegie-Mellon University, Pittsburgh, Pennsylvania. He worked with Digital Equipment Corporation, Maynard, Massachusetts, as the chief architect of the PDP-11/45 computer and for IBM in The Netherlands and in the USA, where he was reponsible for the architecture of embedded systems. In 1979, he became a professor of computer architecture at the Delft University of Technology, from which position he retired in 1999. His main interest is in testing logic and memories. He has written two books and more than 150 papers. He is a fellow of the IEEE and he is on the editorial board of *JETTA*.



Issam B.S. Tlili received the MSEE degree in 1998 from the Delft University of Technology, Delft, The Netherlands. In 1998, he joined Lucent Technologies in Hilversum, The Netherlands, where he was involved in the design and development of telephony switching systems. Currently, he is a software engineer in the Department of Radiography/Fluoroscopy of Philips Medical Systems, Best, The Netherlands.