The Effectiveness of Scan Test and its New Variants

Ad J. van de Goor Said Hamdioui Zaid Al-Ars
Delft University of Technology; Faculty of Electrical Engineering, Mathematics and Computer Science
Laboratory of Computer Engineering; Mekelweg 4, 2628 CD Delft, The Netherlands
E-mail: {A.J.vandeGoor, S.Hamdioui, Z.e.al-ars}@ewi.tudelft.nl

Abstract

Many industrial experiments have shown that the very simple and time-efficient Scan test detects many unique faults. This paper shines a new light on the properties of Scan test; such properties will be evaluated using industrial data. In addition, it will be shown that many faults in a memory, which are not in the cell array, are detectable using the appropriate read-write sequences. The traditional version of Scan test performs 'some' of such read-write sequences, but lacks the capability of performing all of them for both the 'up' and the 'down' address orders and the '0' and the '1' data values. Therefore a new set of Scan based tests will be proposed to fill that vacuum.

1. Introduction

In order to have a high fault coverage, and therefore also a low PPM level, a *set of tests* is usually applied for screening out defective SRAM or DRAM parts. It is interesting to note that one particular test, the *Scan test* [1, 2], almost always belongs to the set, and always detects *unique faults* that are not detected with any other test. This paper gives a description of Scan test; it shows its effectiveness based on industrial results and explains why Scan test always detect unique faults. Because of its industrial popularity, the *Scan test* has many aliases; e.g., the test is also known as *MSCAN test* and as *Zero-One test* [2].

However, Scan test also has its shortcomings: Scan lacks the capability of applying the '0' and the '1' data values, for all *Read-Write Sequences 'RWSs'*, for both 'up' and 'down' address orders. The importance of using the appropriate RWS and both addressing orders will be shown for faults in the peripheral circuits.

Naturally, one may question whether variants of Scan test would be possible, and if so, whether they could be even more effective. Therefore, the concept of Scan test will be extended in two orthogonal ways: based on the use of different RWSs, and based on the use of the '0' and the '1' data values for both the 'up' ad the 'down' address orders. The organization of the paper is as follows. Section 2 introduces the notions of algorithms, tests and stresses, together the notation of march tests. Section 3 describes the results of two experiments, involving the use of Scan test, applied to SRAMs and to DRAMs. It shows the unique fault detection capability of Scan. Section 4 introduces the Read-Write Sequences 'RWSs', which are very important for detecting faults in the peripheral circuits; it also shows the effectiveness of the *Partial Moving Inversion 'PMOVI'* test. This will be used to show the shortcomings of the traditional Scan test. The section ends with a set of four new Scan based tests. Last, Section 5 ends with 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+ [3]. 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 are some form of march test, below the notation of march tests will be given, followed by a description of the stresses of interest to this paper.

Notation of march tests: A *march test* is a sequence of *March Elements 'MEs'*. A 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'*: an *increasing AO* '$\Uparrow$', from cell 0 to cell $n-1$, or a *decreasing AO* '$\Downarrow$', 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 [3]. It consists of three MEs: M0, M1 and M2. The notation '$\Uparrow(r0, w1)$' of M1 means 'for $i = 0$ to $n-1$ do {read $A[i]$ with expected value 0; $A[i] := 1$}'.

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

Records of the 2004 International Workshop on Memory Technology, Design and Testing (MTDT'04)
1087-4852/04 $20.00 © 2004 **IEEE**

Table 1. Algorithms used in the experiment

#	BT	TL	Algorithm, also named BT
1	Scan [1]	$4n$	$\{\Uparrow(w0); \Uparrow(r0); \Uparrow(w1); \Uparrow(r1)\}$
2	MATS+ [3]	$5n$	$\{\Uparrow(w0); \Uparrow(r0, w1); \Downarrow(r1, w0)\}$
3	MATS++ [2]	$6n$	$\{\Updownarrow(w0); \Uparrow(r0, w1); \Downarrow(r1, w0, r0)\}$
4	March B [5, 2]	$17n$	$\{\Updownarrow(w0); \Uparrow(r0, w1, r1, w0, r0, w1); \Uparrow(r1, w0, w1); \Downarrow(r1, w0, w1, w0); \Downarrow(r0, w1, w0)\}$
5	March C- [2]	$10n$	$\{\Updownarrow(w0); \Uparrow(r0, w1); \Uparrow(r1, w0); \Downarrow(r0, w1); \Downarrow(r1, w0); \Updownarrow(r0)\}$
6	March G [6]	$*$	$\{\Updownarrow(w0); \Uparrow(r0, w1, r1, w0, r0, w1); \Uparrow(r1, w0, w1); \Downarrow(r1, w0, w1, w0); \Downarrow(r0, w1, w0); D; \Updownarrow(r0, w1, r1); D; \Updownarrow(r1, w0, r0)\}$
7	March LA [7]	$22n$	$\{\Updownarrow(w0); \Uparrow(r0, w1, w0, w1, r1); \Uparrow(r1, w0, w1, w0, r0); \Downarrow(r0, w1, w0, w1, r1); Dn(r1, w0, w1, w0, r0); \Downarrow(r0)\}$
8	March LR [8]	$14n$	$\{\Updownarrow(w0); \Downarrow(r0, w1); \Uparrow(r1, w0, r0, w1); \Uparrow(r1, w0); \Uparrow(r0, w1, r1, w0); \Downarrow(r0)\}$
9	March RAW[9]	$26n$	$\{\Updownarrow(w0); \Uparrow(r0, w0, r0, r0, w1, r1); \Uparrow(r1, w1, r1, r1, w0, r0); \Downarrow(r0, w0, r0, r0, w1, r1); \Downarrow(r1, w1, r1, r1, w0, r0); \Updownarrow(r0)\}$
10	March SR [12]	$14n$	$\{\Downarrow(w0); \Uparrow(r0, w1, r1, w0); \Uparrow(r0, r0); \Uparrow(w1); \Downarrow(r1, w0, r0, w1); \Downarrow(r1, r1)\}$
11	March SS [13]	$22n$	$\{\Updownarrow(w0);\Uparrow(r0, r0, w0, r0, w1); \Uparrow(r1, r1, w1, r1, w0); \Downarrow(r0, r0, w0, r0, w1); \Downarrow(r1, r1, w1, r1, w0); \Updownarrow(r0)\}$
12	March SL [14]	$41n$	$\{\Updownarrow(w0);\Uparrow(r0, r0, w1, w1, r1, r1, w0, w0, r0, w1); \Uparrow(r1, r1, w0, w0, r0, r0, w1, w1, r1, w0); \Downarrow(r0, r0, w1, w1, r1, r1, w0, w0, r0, w1); \Downarrow(r1, r1, w0, w0, r0, r0, w1, w1, r1, w0)\}$
13	March U [15]	$13n$	$\{\Updownarrow(w0);\Uparrow(r0, w1, r1, w0); \Uparrow(r0, w1); \Downarrow(r1, w0, r0, w1); \Downarrow(r1, w0)\}$
14	Algorithm B [5, 2]	$17n$	$\{\Updownarrow(w0); \Uparrow(r0, w1, r1, w0, r0, w1); \Uparrow(r1, w0, w1); \Downarrow(r1, w0, w1, w0); \Downarrow(r0, w1, w0)\}$
15	PMOVI [16, 17, 18]	$13n$	$\{\Downarrow(w0); \Uparrow(r0, w1, r1); \Uparrow(r1, w0, r0); \Downarrow(r0, w1, r1); \Downarrow(r1, w0, r0)\}$

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

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 the data-background stress.

Address Direction '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: Fast-X and Fast-Y. With *Fast X 'fX'* addressing, each address increment or decrement operation causes another row to be accessed; with *Fast Y 'fY'* addressing, each address increment or decrement operation causes another column to be accessed.

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 and all 1s;

Checkerboard (bDB):0101.../1010...;

Column Stripe (cDB): 0101...; and

Row Stripe (rDB): 0000.../1111....

3. Industrial evaluation

This section presents results of two experiments, involving the use of Scan test, applied to SRAMs and DRAMs

3.1 Experimental results for SRAMs

A large experiment has recently been performed [4], consisting of a set of 39 BTs, applied to a large set of SRAM chips, using the following algorithm stresses: fX and fY ADs, together with the sDB, the bDB, the cDB, and the rDB. The used environmental stresses were: high (+*V*) and low (-*V*) **V**oltage; and high (+*S*) and low (-*S*) **S**peed. 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 remaining $39 - 25 = 15$ BTs, which means $15 \times 32 = 480$ tests, will be discussed.

Table 1 shows this set of 15 BTs. The left column shows the number '#' of the BT, the column 'BT' lists the name of the BT, the column 'TL' lists the *Test Length*; while the complete algorithm is given in the last column. Note that Scan has a TL of *only* $4n$!

Table 2 summarizes the results of the 15 BTs of Table 1; only the most interesting BTs have been listed for the environmental SCs +S and +V; which was the most effective [4]. The column pair 'Highest FC' lists the highest *Fault Coverage 'FC'* of the corresponding BT, together with the algorithm Stress Combination 'SC' producing that FC. For example, Scan BT has the highest FC of 92 for the SC fY and bDB, or the SC fY and cDB. The column pair 'Lowest FC' is similar. The column pair 'Highest Union' lists the combined FC of two applications of the corresponding BT, using different SCs. For example, the highest FC of two ap-

Table 2. Influence of algorithm SCs

BT	Highest FC		Lowest FC		Highest Union	
	FC	SC[1]	FC	SC[1]	FC	SC[2]
Scan	92	Yb;Yc	79	Xs	102	Xs+Yc
March G	109	Ys;Yc	101	Xb;Xr	111	Many[3]
March LR	110	Ys;Yc	97	Xr	113	Many[3]
March SL	107	Yr	96	Xr	112	Xs+Yb
March U	112	Yr	99	Xs	115	Many[3]
PMOVI	105	Ys	98	Xr	108	Xc+Ys

1: The entries 'Pq' represent the algorithm SC of the BT
'P' specifies the AD; P $\in$ {X=fX, Y=fY}
'q' specifies the DB; q $\in$ {s=sDB, b=bDB, r=rDB, c=cDB}
'Yb;Yc' means that Yb *or* Yc can be used
2: 'Xs+Yc' means the use of fX with sDB *and* fY with cDB
3: Many also includes Xs

plications of Scan (which is 102) will occur with the SCs fX and sDB, and fY and cDB. Based on Table 2 the following conclusions can be made:

- The highest FC is systematically obtained with the fY AD, and the lowest FC with the fX AD. This could indicate that the SRAMs being tested are more sensitive to delays in the column decoder path.
- The fY AD, together with the cDB or the sDB, are the most effective; while the fX AD, together with the rDB, is the least effective.
- The 'Highest Union' almost consistently involves a test using the fY AD, together with another test using the fX AD; with the fX AD typically the sDB is used.

Table 3 summarizes the effect of the environmental SCs. The column 'SC' lists the environmental SC; the next two column pairs list the 'Highest FC' and the 'Highest Union'. For example, for SC=+S+V, the highest FC is 120, which is produced by the BTs March G and also by March SL. Note 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. The column 'Highest Union' lists the the two tests, which together, have the highest FC; note that the second BT is *always Scan BT*. *This means that Scan BT detects more of another class of faults than any of the other BTs, which makes it a very special BT!*

It is important to note here that most of the tests of Table 1 have been designed to detect address decoder faults and faults in the memory cell array (for more details, see the references with each BT in Table 1), but not for detecting faults in the peripheral circuits.

Table 3. Summary of environmental SCs

SC	Highest FC		Highest Union	
	FC	BT	FC	BT
+S+V	120	March G, March SL	124	March G, Scan
+S-V	106	March U	109	March U, Scan
-S+V	102	March LR	105	March LR, Scan
-S-V	99	March B, March U	103	March U, Scan

Table 4. The fault coverage of Scan for +S+V

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

Table 4 shows the FC of Scan BT, for the environmental SC=+S+V. The column '#' lists the number of the algorithm SC, listed in column 'SC'. The column 'UF' lists the number of *Unique Faults 'UFs'* detected by the corresponding test. E.g., Scan test using fX and cDB detects *one* UF; this means that none of the other 8 tests of the table do detect that fault. Industrial application of Scan test usually results in UFs, which means that none of the other applied BTs did detect those faults; this will be explained in Section 4.

The 8×8 matrix lists the FC of Scan test. The *diagonal entries* list the FC of the particular SC; e.g., the FC for the SC fX and cDB is 89. The entries *below the diagonal* list the *Union* of the FCs of the two corresponding SCs; e.g., the joint FC of Scan test, applied using fX, together with the sDB, and fY, together with cDB is 102; see intersection of column 1 and row 8. The entries *above the diagonal* list the *Intersection* of the FCs of the corresponding two tests; that is the number of faults they both detect. The table shows that the FC of Scan depends on the used DB and the AD.

3.2 Experimental results for DRAMs

Figure 1 shows the results of applying Scan BT to a set of 800 DRAM chips [19]. A total of 21 BTs were applied, using 168 SCs at room temperature. The total FC of all tests was 116; the total FC for Scan was 28. The figure shows the FC as a function of the AO-DB (i.e., address order - data backgrounds) combination. The highest FC is obtained for fX addressing, using the bDB or the rDB (and for fY using the bDB). This means that applying a sequence of write and read operations in a column (i.e., using the fX AD), using an alternating 0101... pattern produces the highest FC. *This is the most stressy pattern for the write drives, the precharge circuits and the sense amplifiers*, as will be explained in the next section.

4. Analysis of Scan

The traditional Scan test, see test #1 in Table 1, is a very simple test; though very effective in terms of detecting unique faults. The questions which naturally arise are: why is is the case for Scan? and whether other tests, based on the concept of Scan test are possible? In order to show

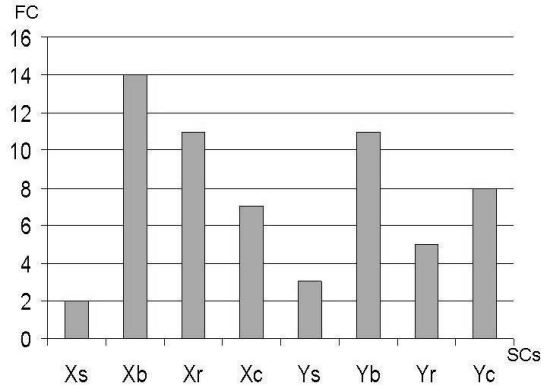

Figure 1. Effect of DB and AO on FC of Scan

these, we will first analyze *Read-Write Sequences 'RWSs'* and another very well-known test, the *Partial Moving Inversion 'PMOVI'* test. Thereafter, the shortcomings of Scan will be shown, followed by a new set of Scan based tests.

4.1 Read-Write Sequences 'RWSs'

A way to express the capability of a test of detecting faults in the peripheral circuits and faults due to address decoder delays is in terms of required Read-Write Sequences 'RWSs' [4]. A RWS is a property of a march element and is defined by the last operation applied to an address and the first operation applied to the next address. E.g., the march element '$\Uparrow(r0, w1, r1, w1)$' performs a *Read-after-Write 'RaW'* RWS, because the last operation to an address is the '$w1$' operation, and the first operation applied to the next address is the '$r0$' operation. The set of RWSs consists of:

RaR: Read-after-Read,
WaW: Write-after-Write,
WaR: Write-after-Read, and
RaW: Read-after-Write.

Below, a summary will be given of the required RWSs and DBs for detecting *Peripheral circuit Faults 'PFs'* [4]. The PFs consist of faults due to: (a) a *Slow Write Driver 'SWDF'*, (b) a *Slow Sense Amplifier 'SSAF'*, and (c) a *Slow PRecharge circuit 'SPRF'*. Tests for these PFs should use the fX AD, because they all deal with faults in a column of the cell array.

- **SWDF**: The Write driver may be too slow such that the differential voltage on the bit lines during the write operation is reduced; this may cause the cell not to be written. The WaW RWS is the most stressy RWS, using alternating data values of '0' and '1'; i.e., using the bDB or the rDB.

Table 5. Detection capabilities of PMOVI

Addressing	RWS	ME
$\Uparrow$	$r0ar1$	ME #1: $\Uparrow(r0, w1, r1)$
$\Uparrow$	$r1ar0$	ME #2: $\Uparrow(r1, w0, r0)$
$\Downarrow$	$r0ar1$	ME #3: $\Downarrow(r0, w1, r1)$
$\Downarrow$	$r1ar0$	ME #4: $\Downarrow(r1, w0, r0)$

- **SSAF**: The *Sense Amplifier 'SA'* may be too slow, or is asymmetric (because of some offset voltage) such that read operations produce incorrect results. The RaW RWS is the most stressy RWS, using the bDB or rDB.
- **SPRF**: The *Prechage Circuit 'PC'* may be too slow, or it may not precharge both bit lines to the same voltage level; such that especially read operations will produce incorrect results, because they are most sensitive to bit line voltage offset errors. The RaW RWS is the most stressy RWS, using the bDB or rDB.

The above peripheral circuit faults are only detectable use the fX AD, together with the bDB or the rDB. It shows he importance of the use of the proper RWSs and DBs. Note that Scan is special because it is the *only BT* which has a WaW and a RaR RWS, without other operations in between, and therefore it has the capability to detect e.g., the SWDF. For example, the WaW RWS, using the bDB, performs the following sequence of write operations: w0, w1, w0, w1, etc. When performed using the fX AD, the write drivers, as well as the row decoder, are stressed the most: each write operation has to write the inverse data, while at the same time it has to apply the operation to the next address!

4.2 An analysis of the PMOVI test

The Partial Moving Inversion 'PMOVI' BT is listed as test #15 in Table 1. This test is very popular in industry, because it detects unique faults; i.e., faults not detected by any of the other BTs. The algorithm of PMOVI consists of an initialization ME#0 '$\Downarrow(w0)$', together with 4 march elements 'MEs' which all have the structure '$\Updownarrow(rx, w\overline{x}, r\overline{x})$'; they perform a RaR RWS; see Table 5. ME#1 (i.e., $\Updownarrow(r0, w1, r1)$') and ME#3 perform a $r0ar1$ (this means a $r0$ *after* a $r1$) RWS, while MEs#2 and ME#4 perform a $r1ar0$ RWS. In addition, the $rxar\overline{x}$ RWSs are performed such that the address to which the rx operation is applied is *higher* than the address to which the $r\overline{x}$ operation is applied (in ME#1 and ME#2); or *lower*, in ME#3 and ME#4. The result is that *speed related faults* are detected [2, 18]; see Table 5. E.g., the first entry in the table shows that the $r0ar1$ RWS, whereby the $r0$ operation is applied to a higher address than the $r1$ operation (denoted by the '$\Uparrow$' symbol in the column 'Addressing'), is detected by ME #1 '$\Uparrow(r0, w1, r1)$'; etc.

The message is that the RaR RWS '$rxar\overline{x}$' is applied using *both AOs*, i.e., the $\Uparrow$ and the $\Downarrow$ AO (for covering worst

Table 6. Shortcomings of traditional Scan test

Address	$\Uparrow(w0/w1)$	$\Uparrow(r0/r1)$	$\Uparrow(w1/w0)$	$\Uparrow(r1/r0)$
0	0:w0	0:r0	0:w1	0:r1
1	1:w1	1:r1	1:w0	1:r0
2	2:w0	2:r0	2:w1	2:r1
3	3:w1	3:r1	3:w0	3:r0
Result	$\Uparrow w1aw0$	$\Uparrow r1ar0$	$\Uparrow w0aw1$	$\Uparrow r0ar1$
Missing	$\Downarrow w1aw0$	$\Downarrow r1ar0$	$\Downarrow w0aw1$	$\Downarrow r0ar1$

case address transitions), and for both data values, i.e., $x = 0$ and $x = 1$ (for covering asymmetric sensitivities in the '0' and '1' values) [2, 16, 17, 18].

4.3 Shortcomings of the traditional Scan test

Table 6, designed for a memory with 4 words, illustrates the shortcomings of the traditional Scan test. It is assumed that the bDB or the rDB is used, together with the fX AD. The result will be that the pattern '0101' will be written by the '$\Uparrow(w0/w1)$' ME, or the '1010' pattern by the $\Uparrow(w1/w0)$' ME. The notation '$wx/w\overline{x}$' denotes that alternating the value x and $\overline{x}$ will be written to adjacent addresses.

The column 'Address' lists the physical memory address, the remaining column headers list the MEs of Scan test. Each table entry in those four columns has the form '$\#$:Ox', where '#' denotes the Step # in the ME, for the operation 'Ox'; $O\in\{r, w\}$, $x\in\{0, 1\}$. The entry 'Result' shows the RWS (e.g., '$r1ar0$') which is performed (using the '$\Uparrow$' AO); however, the entry 'Missing' shows the RWS (e.g.,'$r1ar0$') which has *not* been performed (using the '$\Downarrow$' AO). *Hence, Scan BT does not apply its operations using both AOs!* Note also that Scan does not apply all RWS (e.g., RaW which is the most stressy for detecting SSAF and SPRF).

4.4 New Scan based tests

Table 7 lists the four new variants of Scan BT, one for each RWS. The test length 'TL' is given in the first line of the header of each sub-table; the second line describes the algorithm, using the following notation for the operations of the MEs '$(Op1/Op2)$', where $Op1, Op2\in\{-, wx, rx\}$. '$-$' denotes no-operation, '/' denotes that the two operations '$Op1$ and $Op2$' are applied to two successive addresses; i.e., $Op1$ is applied to all *even* addresses, while $Op2$ to all *odd* addresses. Note that the TL of the '$\Uparrow\ (-/w0)$' ME is $\frac{1}{2}n$.

As an example, consider the $8n$ WaW Scan test which consists of 8 MEs; see third sub-table. In case a ME uses the '$\Downarrow$' AO, Steps #0 through #3 proceed from the highest address (Address 3) to address 0. The row 'Result' shows that the the following WaW RWSs are performed: ME#0 performs the '$\Uparrow w1aw0$', ME#2' the '$\Downarrow w1aw0$', ME#4 the '$\Uparrow w0aw1$', and ME#6 the '$\Downarrow w0aw1$'; i.e., that all four combinations of AOs and data values are covered.

5. Conclusions

The importance of Scan test has been demonstrated, based on industrial results. Its unique fault detection capabilities come from the fact that WaW (ie., Write-after-Write) and RaR (i.e., Read-after-Read) operation sequences are applied without any intervening other operations. This causes the most stressy condition for some of the periphery circuits (e.g., the write drives), provided that alternating data values are used. The paper also illustrates the shortcomings of the traditional Scan test, and proposes four new tests; one for each read-write sequence. It is the expectation that the fault coverage of the new tests will be superior. Plans are currently in place for industrial evaluation of the new tests.

References

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

[2] 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/

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

[4] A.J. van de Goor and S. Hamdioui, 'Detecting Faults in Peripheral Circuits and and Evaluation of SRAM Tests', Submitted to the IEEE Int. Test Conf., Charlotte NC, Oct. 26-28, 2004.

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

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

[7] 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

[8] A.J. van de Goor and G.N. Gaydadjiev, 'March LR: A Memory Test for Realistic Linked Faults', In Proc. IEEE VLSI Test Symposium (VTS'96), 1996, pp. 272-280

[9] S. Hamdioui, Z. Al-Ars and A.J. van de Goor, 'Testing Static and Dynamic Faults in Random Access Memories', In Proc. of the 20^{th} IEEE VLSI Test Symposium (VTS'2002), Monterey, CA, pp. 395-400

[10] 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

[11] A. J. van de Goor and J. de Neef, 'Industrial Analysis of DRAM Tests', In Proc. Design Automation and Test in Europe (Date99), March 9 -12, Munich, 1999, pp. 623-630

[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

1087-4852/04 $20.00 © 2004 **IEEE**

Table 7. New Scan based tests

The RaW Scan test: TL=$5n$

$\{\Uparrow(-/w0); \Uparrow(w1/r0); \Downarrow(r0/w1); \Uparrow(-/w1); \Uparrow(w0/r1); \Downarrow(r1/w0)\}$

Address	$\Uparrow(-/w0)$ ME#0	$\Uparrow(w1/r0)$ ME#1	$\Downarrow(r0/w1))$ ME#2	$\Uparrow(-/w1)$ ME#3	$\Uparrow(w0/r1)$ ME#4	$\Downarrow(r1/w0)$ ME#5
0	0:−	0:w1	3:w1	0:−	0:w0	3:w0
1	1:w0	1:r0	2:r0	1:w1	1:r1	2:r1
2	2:−	2:w1	1:w1	2:−	2:w0	1:w0
3	3:w0	3:r0	0:r0	3:w1	3:r1	0:r1
Result	−	$\Uparrow r0aw1$	$\Downarrow r0aw1$	−	$\Uparrow r1aw0$	$\Downarrow r1aw0$

The RaR Scan test: TL=$6n$

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

Address	$\Uparrow(w0/w1)$ ME#0	$\Uparrow(r0/r1)$ ME#1	$\Downarrow(r1/r0)$ ME#2	$\Uparrow(w1/w0)$ ME#3	$\Uparrow(r1/r0)$ ME#4	$\Downarrow(r0/r1)$ ME#5
0	0:w0	0:r0	3:r0	0:w1	0:r1	3:r1
1	1:w1	1:r1	2:r1	1:w0	1:r0	2:r0
2	2:w0	2:r0	1:r0	2:w1	2:r1	1:r1
3	3:w1	3:r1	0:r1	3:w0	3:r0	0:r0
Result	$\Uparrow w1aw0$	$\Uparrow r1ar0$	$\Downarrow r1ar0$	$\Uparrow w0aw1$	$\Uparrow r0ar1$	$\Downarrow r0ar1$

The WaW Scan test: TL=$8n$

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

Address	$\Uparrow(w0/w1)$ ME#0	$\Uparrow(r0/r1)$ ME#1	$\Downarrow(w1/w0)$ ME#2	$\Downarrow(r1/r0)$ ME#3	$\Uparrow(w1/w0)$ ME#4	$\Uparrow(r1/r0)$ ME#5	$\Downarrow(w0/w1))$ ME#6	$\Downarrow(r0/r1)$ ME#7
0	0:w0	0:r0	3:w0	3:r0	0:w1	0:r1	3:w1	3:r1
1	1:w1	1:r1	2:w1	2:r1	1:w0	1:r0	2:w0	2:r0
2	2:w0	2:r0	1:w0	1:r0	2:w1	2:r1	1:w1	1:r1
3	3:w1	3:r1	0:w1	0:r1	3:w0	3:r0	0:w0	0:r0
Result	$\Uparrow w1aw0$	$\Uparrow r1ar0$	$\Downarrow w1aw0$	$\Downarrow r1ar0$	$\Uparrow w0aw1$	$\Uparrow r0ar1$	$\Downarrow w0aw1$	$\Uparrow r0ar1$

The WaR Scan test: TL=$5n$

$\{\Uparrow(w0/-); \Uparrow(r0/w1); \Downarrow(w1/r0); \Uparrow(w1/-); \Uparrow(r1/w0); \Downarrow(w0/r1)\}$

Address	$\Uparrow(w0/-)$ ME#0	$\Uparrow(r0/w1)$ ME#1	$\Downarrow(w1/r0)$ ME#2	$\Uparrow(w1/-)$ ME#3	$\Uparrow(r1/w0)$ ME#4	$\Downarrow(w0/r1)$ ME#5
0	0:w0	0:r0	3:r0	0:w1	0:r1	3:r1
1	1:−	1:w1	2:w1	1:−	1:w0	2:w0
2	2:w0	2:r0	1:r0	2:w1	2:r1	1:r1
3	3:−	3:w1	0:w1	3:−	3:w0	0:w0
Result	−	$\Uparrow w1ar0$	$\Downarrow w1ar0$	−	$\Uparrow w0ar1$	$\Downarrow w0ar1$

[13] 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 (MTDT'2002), Bendor, France, pp. 95-100

[14] S. Hamdioui, Z. Al-Ars, A.J. van de Goor and Mike Rodgers, 'March SL: A Test for All Static Linked Memory Faults', In Proc. of the Twelfth Asian Test Symposium (ATS'2003), November 16-19, Xi'an, China, pp. 372-377

[15] 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

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

[17] Matthias Klaus and Ad 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

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

[19] A. J. van de Goor and A. Paalvast, 'Industrial Evaluation of DRAM SIMM Tests', In Proc. IEEE Int. Test Conf., (ITC'00), 2000, pp. 426-435