BIST Enhancement for Detecting Bit/Byte Write Enable Faults in SOC SRAMs

Said Hamdioui Zaid Al-Ars
Delft University of Technology
Computer Engineering Laboratory
Mekelweg 4, 2628 CD Delft, The Netherlands
{S.Hamdioui, Z.Alars}@ewi.tudelft.nl

Javier Jimenez Jose Calero
Design of Systems on Silicon (DS2)
Calle Charles Robert Darwin, 2
46980, Paterna, Valencia, Spain
{Javier.Jimenez,Jose.Calero}@ds2.es

***Abstract*—The continued increase of the integration density of systems on chip (SoCs) and the number of embedded memory blocks in them, together with the continued technology scaling, increases their sensitivity to a variety of potential manufacturing (new) defects. Standard march tests are usually used to achieve a good fault/defect coverage. This paper presents an experiment in diagnosing defects in the circuitry responsible for the realization of bit, byte or group write enable in memories. First defects in such circuitry are analyzed, and fault models together with an appropriate test algorithm are presented. Subsequently, the test is added to an existing BIST engine to target the bit/byte write enable faults. The preliminary silicon results of two experiments are presented. They validate some of the targeted fault models and show the importance of considering bit/byte write enable faults for high outgoing product quality.**

I. Introduction

Todays ASICs and SoCs have become increasingly embedded memory intensive. It is very common to have tens if not hundreds of embedded SRAMs with different sizes and types (e.g., single-port, multi-port), performing different functions on a single chip. The wide use of SRAMs in different applications means that new functionalities have to be added to such memories in order to satisfy the design requirements. One of such functionalities is the ability of the SRAM to perform bit and byte write operations. In addition, words may be arranged as groups of bits having different number of data bits than a standard byte. The memory may also have the ability to write such groups of data. This is realized by using specific *write enable control* circuitry that enables or disables the writing circuitry of the associated bits in the data words. Such circuitry needs to be tested for different kinds of defects as compared with the memory array, and using specific test sequences. This paper deals with such a topic.

The paper is organized as follows. Section 2 presents some background information about bit/byte enable control circuit. Section 3 discusses the targeted defects and fault models. Section 4 presents the test algorithm detecting such faults. Section 5 gives an industrial evaluation. Section 6 analyzes the test results while focusing on the impact of bit/byte write enable faults. Section 7 concludes the paper.

II. Bit/Byte write enable circuitry

There are two possible locations to implement the bit/byte write enable circuits, either in the beginning of input data path or at its end [1]. Figure 1 shows a typical implementation of bit write enable circuits when implemented at the the beginning of input of data path; i.e., the circuit is implemented *before* the column decoder. The data word to be written is controlled by (a) a *global write enable GWE*, which enables or disables the word to be written, and (b) *bit write enables BWEi* which are associated one-to-one with each data bit of the word and therefore controlling whether the data bit i ($i \in \{0, 1, .., B - 1\}$; B is the word size) to be written. A data bit is written into memory if both GWE and its corresponding BWEi are both high.

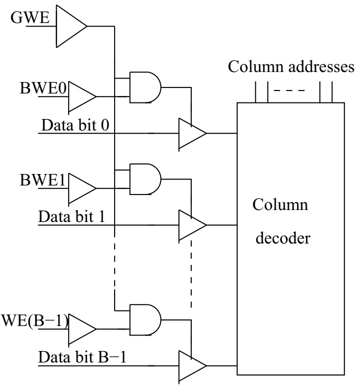

Fig. 1. Typical bit write enable desig

The BWE controls/mask signals are often routed close to each other and therefore subject to manufacturing defects as is the case for the memory cell array. Examples are shorts to Vdd and GND, and bridges between neighboring lines. The default way of testing memory (e.g., using a march test) is to control all mask signals (i.e., GWE's and BWE's) from the same source and therefore set up all of them to their asserted values [1], [2]. That means all data bits are enabled concurrently for writing. As consequence, if any of the mask signal suffers from short (e.g., to Vdd) and/or bridges, the fault will not be detected.

III. Bit/Byte write enable defects/fault models

The possible manufacturing defects in the BWE control circuits can be modeled as opens, shorts and bridges. In the presence of opens in such circuits, the whole writing path

978-1-4244-2628-7/08/$25.00 ©2008 IEEE

will be impacted and therefore such defects can be detected during the test of address decoder (delay) faults [3], [4], [5], [6], [7]. For this reason, this paper will deal only with shorts and bridges.

Let us define a *short* as undesired resistive path between a control line (e.g., BWE) and Vdd or GND. On the other hand, let us define a *bridge* as an undesired resistive path between two control lines both different from Vdd and GND (e.g., BWE0 bridged with BWE1; see Figure 1). Only bridges involving two lines will be considered as their occurrence probability is very high as compared with bridges involving more than two lines. Moreover, a test targeting a bridge involving two lines also covers some multi-line bridges.

Consider a BWE control circuit with B BWE control signals. As the topology of the circuit and the position of the lines are usually not known, all possible fault models (shorts and bridges) have to be considered for testing. These are:

- AND-wired BWE bridges: there are $C_2^B = \frac{B!}{2!(B-2)!}$ possibilities of such bridges.
- OR-wired BWE bridges: there are $C_2^B = \frac{B!}{2!(B-2)!}$ possibilities of such bridges.
- BWE shorted to Vdd: there are B possibilities of such shorts.
- BWE shorted to GND: there are B possibilities of such shorts.

In total there are $2C_2^B + 2B = B(B+1)$ possible functional faults.

IV. Test solutions for BWE faults

As mentioned previously, the default way used to test a memory with bit/byte write enable capabilities is to enable all the control/mask signals and apply the memory tests like march tests [1], [2]. However, the faults within the bit/byte logic control circuit will not be covered.

Another way which is used to test for bit/byte write faults is by applying a *minimal test* [1], [2]; in this case the memory is written with one pattern while all BWE signals enabled, followed by writing the inverted data to the memory while all BWE signals are disabled; the memory will be then read with the original expected pattern. This approach will detect all shorts (i.e., stuck at 1 and stuck at 0), but will fail to detect the bridging faults.

A BIST algorithm for bit/byte write enable faults was reported in [1]; it uses a serial interfacing technique [8]. A *checkerboard* test was proposed in [2] to cover shorts as well as bridges. However, the proposed solution guarantees the detection of bridges of only *adjacent* lines; this is something that is not usually known for test engineers in advance.

There are two different types of BWE faults: bridges (namely, AND-wired and OR-wired bridges between BWE lines), and shorts (between different BWE lines and Vdd or GND). It is possible to detect all of these faults using a specific test algorithm that inspects the correct functionality of the BWE circuitry. The test is referred to as 'Test BWE', which has the structure shown in Figure 2. The test has a 100% Fault Coverage (FC) for all faults caused by both bridges and shorts as will be shown next. The test length is constant and does not depend on the size of the memory, since it requires only one row to be selected.

```
Select one row (R) and write 0's;
for each BWEi {
        Enable BWEi;
           write 1's in R;
           read from R;
           write 0's in R;
           disable BWEi;
}
Repeat with complementary data;
```

Fig. 2. Definition of Test BWE

First a single row is selected to perform the test. Note that only one row is needed to check for the functionality of BWE control circuitry and there is no need to go through the whole memory.

Second, for each enabled BWEi where $i \in \{0, 1, ..., B-1\}$, a sequence of operations is performed: starting with writing 1 only to the selected BWEi, followed by reading the whole row, thereafter writing 0 back to the selected BWEi, and finally disabling the selected BWEi. All shorts, OR- and AND-bridges will be sensitized when writing a solid data background 1's, and detected during the reading operation of the selected row as follows:

- In the presence of any short between GND and any BWEi, or in the presence of AND-bridges between BWEi and any other BWE, no 1 will be written in the enabled bit. These faults will be detected during the reading of the row R, as 0 will be read from the enabled bit (corresponding to BWEi) instead of 1.
- In the presence of any short between Vdd and any BWE (except BWEi), or in the presence of OR-bridges between BWEi and any other BWE, 1 will not only be written in the enabled bit, but also to the bits with these defective BWE's. These faults will be detected during the reading of the row R as the faulty written 1's will be read from non-enabled bits instead of the expected 0's.

Note that the "for loop" has to be performed for a second time but then with complementary data values. This is in order to cover possible asymmetric defects.

V. Industrial results

A large experiment was performed at "Design of Systems on Silicon (DS2). The experiment consists of adding advanced memory test algorithms to an existing BIST controller (including Test BWE) and evaluate the impact on the outgoing product quality, as well as diagnosing and validating some advanced fault models (e.g., advanced dynamic faults in the memory array, in the address decoder as well as in the peripheral circuitry). Below, first an overview of the experiment will be presented and thereafter the coverage results.

TABLE I
SET OF TESTS ADDED TO THE EXISTING BIST

Dynamic Tests				
Name	Test length	Description	DB	AD
March DFr	$22n$	$\{\Uparrow(w0);\Uparrow(r0,w0,r0,w1,r1);\Uparrow(r1,w1,r1,w0,r0);$ $\Downarrow(r0,w0,r0,w1,r1);\Downarrow(r1,w1,r1,w0,r0);\Uparrow(r0)\}$	cDB and rDB	fx and fx
March dADF	$\frac{9}{2}n+$ $9\cdot n\cdot\log_2(n)$	$\{\Updownarrow(w0);\Uparrow^{H1}(r0,w1);\Uparrow^{H1}(r1,w0);\Downarrow^{H1}(r0,w1);\Downarrow^{H1}(r1,w0)\}$	sDB	fx and fy (H1 addressing)
March dPCFw March dPCFm	$8n$ $5n$	$\{\Updownarrow(w0);{}_X\Updownarrow(w1,r1,w0);\Updownarrow(w1);{}_X\Updownarrow(w0,r0,w1)\}$ $\{\Updownarrow(w0);{}_X\Updownarrow(r0,w1);{}_X\Updownarrow(r1,w0)\}$	sBd or cDB sDB or cDB	fx fx
Special Tests				
March SAM	$[1+28\log_2(B)]\frac{n}{B}$	The description depends on the size of the B; see [23]	*Special DBs*	fx
Gal9R	$38n$	$\{\Uparrow(w0);\Uparrow_b(w1_b,\Box(r0,r1_b),w0_b);\Downarrow(w1);\Downarrow_b(w0_b,\Box(r1,r0_b),w1_b)\}$	sDB	fy
March SZ	$[1+\log_2(B)]\cdot 7\cdot\frac{n}{B}$	$\{\Downarrow(w0);\Uparrow(r0,w1,r1);\Downarrow(r1,w0,r0)\}$	*Special DBs*	fx and fy
Scan	$4n$	$\{\Updownarrow(w0);\Updownarrow(r0);\Updownarrow(w1);\Updownarrow(r1)\}$	rDB	fx
TEST BWE	$\Theta(B)$	see Figure 2	irrelevant	not applicable

A. Overview of the experiment

An *algorithmic stress* specifies the way the test is performed, and therefore it influences the sequence and/or the type of the memory operations. The used algorithmic stresses in our experiment are the *address directions (ADs)* and the *data-backgrounds (DBs)*. The used ADs consist of 'fx' and 'fy'.

- Fast x (fx): 'Fast x' addressing is simply incrementing or decrementing the address in such a way that each step goes to the next row.
- Fast y (fy): 'Fast y' addressing is simply incrementing or decrementing the address in such a way that each step goes to the next column.

A DB is defined as the pattern of ones and zeros as seen in an array of memory cells. The used DBs are:

- Solid (sDB): all 0s and all 1s.
- Checkerboard (bDB): 0101.../1010.../0101.../1010..
- Column stripe (cDB): 0101.../0101.../0101.../0101.
- Row stripe (rDB): 0000.../1111.../0000.../1111....

The test program used in this experiment consists of three test classes: the initial existing BIST, Dynamic Tests and Special Tests; they are explained next.

The existing BIST

The existing BIST implements the main classical/known memory tests that have been used in the industry such as Scan [9], MATS+ [10], MATS++[11], PMOVI [12], March C- [13], March B and March G [14]. It also implements a minimum test (discussed in Section 4) targeting BWE faults. To facilitate the diagnosis and the evaluation of the advanced tests, all classical tests were implemented using both ADs (fx and fy), but with only a sDB. These stress combinations used with the above mentioned tests are sufficient to detect all *static*/traditional faults in memory cell array [9], [10], [12], [13], [14], [15], [16], [17], [18], address decoder [3], [4], [16] as well as in the peripheral circuits [16].

Dynamic Tests

The first class of added tests to the existing BIST consists of tests targeting *dynamic* faults. With the scaling of technology, new defect mechanisms take place in addition to the known traditional ones. The new defect mechanisms cause fault behaviors that are different from the static faults. They cause mainly *time-related* faults referred to as *dynamic* faults. Dynamic faults are divided into three types: (a) Dynamic Memory Cell Array Faults [6], [19], [20] (b) Dynamic/Delay Address Decoder Faults [3], [4], [5], [7] and (c) Dynamic Peripheral Circuit Faults [21], [22].

Dynamic Tests added to the existing BIST consist of the tests shown in first block of Table I. They are given together with their test length and the stress combination they are used with; in the table n denotes the memory size and B the word size.

- Test for dynamic memory cell array faults: The framework of all possible dynamic memory cell array faults is presented in [19]. However, only a *subset* of such space has been shown to be realistic based on defect injection and circuit simulation; mainly some single-cell dynamic faults. March DFr shown in Table 1 is an effective test to target such a subset of faults [15], [20].
- Tests for dynamic/delay address decoder faults: March dADF show in Table 1 is an optimal and efficient test capable of detecting such faults [4], [7]. The test has to use *hamming addressing* with hamming distance between two addresses of H=1. H is defined as the number of bit positions in which two successive addresses (say Ax and Ay) of an address pair differ. In addition, the test has to be applied using the sDB together with AD fx (to detect dADFs in row decoder) and fy (to detect dADFs in column decoder).
- Tests for dynamic peripheral circuit fault: The minimal test set to target all such faults is given in Table 1; it consists of two tests [21]: March dPCF$_w$ (w=walking) and March dPCF$_m$ (m=marching); Both tests have to be used with fx addressing and sDB or cDB.

Special Tests

The second class of tests added to the existing BIST controller consists of some *Special Tests*. These consist of four extra tests that are added, based on our experience in the field of memory testing, to detect some specific faults or some unmodeled/non-understood faults; they are given in the second bottom block of Table I.

- March SAM [23] is developed to target intra-word coupling faults, which are coupling faults that take place between memory cells that belong to the same memory word, and faults due to the interference between I/O paths of the same word. March SAM requires the use of fx AD and *special DBs* that are different from the standard ones; see for more details [23].
- March Gal9R is a simplified form of Galpat [16] where the read actions are restricted only to the eight physical neighbors of the base cell. In the table, '$_\Box$' denotes the eight neighbor cells of any base cell and '$_b$' denotes the base cell. For example, '$_\Box(r0, r1_b)$' means apply read 0 to the 8 neighbor cells, and after each read apply read 1 from the base cell. The test has to be used with sDB and fy AD.
- March SZ is expected to detect interferences between I/O paths, timing related faults (not yet fully understood), cell stability faults and leakage currents. Such a test is repeated with $[1 + \log_2(B)]$ DBs which are generated as follows. For B=2, there are 2 DBs:00 and 01; for B=4, there are 3 DBs: 0000, 0101 and 1100; etc.
- Scan with fx AD and rDB. This test is very powerful in detecting faults in write drivers. The same test used with other stresses (e.g., sDB) will not be able to detect such faults. Therefore, the test can help in diagnosing the location of the faults (e.g., the memory array or write drivers).
- Test BWE discussed in Section IV; see Figure 2.

B. Coverage results

Two test experiments were performed, using the same non-algorithmic stresses (typical voltage, typical speed and room temperature). In both experiments, the 'device under test' is considered faulty by a test if it fails the same test twice.

- In the first experiment, *all* tests (i.e., original tests in the existing BIST and the tests of Table I) were applied two times to $0.13\mu m$ embedded SRAMs of 256KB.
- In the second experiment, another memory design is used (the same technology). First the original tests were applied; then the added tests of Table I were applied *only* to passing chips. The added tests are then applied sequentially; if a device fails a test then it will not be tested by the next test.

The representation of all the test results and their analysis is out of the scope of this paper. In this paper, we will restrict ourself to the coverage results with emphasis on the added value of the Test BWE.

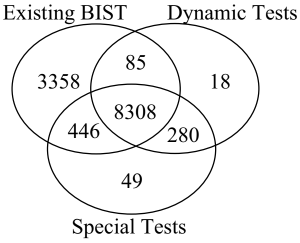

Fig. 3. Venn-diagram of first experiment

First experiment

From testing a huge number of embedded SRAMs, the total number of failing devices is 20211, from which 7667 failed each individual test (i.e., failed each test in the existing BIST, each dynamic test and each special test). Faults detected by all individual tests are supposed to be the easy-to-detect faults like stuck-at-faults. In the rest of this paper, the focus will be on the devices that did not fail all individual tests, which consists of 12544=20211-7667.

Figure 3 shows the venn-diagram of the 12544 failing devices for the three different test classes. The existing BIST detects 12197, Dynamic Tests detect 8691 and Special Tests detect 9083 faults. Note that 8308 faults are detected by each of the three test classes, but not by all individual tests. For example, a fault is detected by Dynamic Tests since March dADF detects the fault, but not by March dPCFw neither by March dPCFw. From the venn-diagram we can conclude the following:

- The fault coverage (i.e., the number of detected defective chips) achieved with existing BIST is the highest; it is about 97.2% of the total faults. This may be explained by (a) the high number of tests in the existing BIST as compared with the total number of tests of the other two test classes and (b) the occurrence probability of the traditional/static faults is larger as compared with the new targeted faults.
- The number of faults that are *only* detected with Static Tests is 3358. These faults are not modelled as static faults since March DFr (designed to target some dynamic faults), which also cover all static faults [15], [20], does not detect them. This clearly indicates the existence of faults that are not understood and modeled yet.
- The Dynamic Tests detect 298=18+280 ($\simeq$ 2.4%) faults that are not detected with Static Tests. This may indicate the importance of considering dynamic faults in order to achieve a high product quality.
- Special Tests detect 49 ($\simeq$ 0.4%) faults that are neither detected with Static Tests nor with Dynamic Tests. Special Tests target intra-word coupling faults, faults due to the interference between I/O paths of the same word, cell stability faults and byte-write enable faults. Such faults are partially understood and modeled.

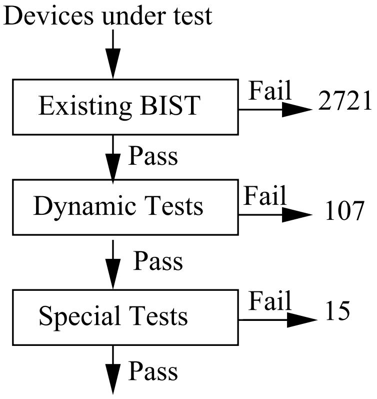

Fig. 4. Coverage results of second experiment

Second experiment

The volume of devices tested in the second experiment was about 7x less than the volume of the first experiment. Testing was done sequentially; the results are given in Figure 4. The total number of faults detected in this experiment is 2843. First, all existing BIST test algorithms were applied. The total number of failures in this phase is 2721; i.e., 96% of total faults. In the next phase, the Dynamic Tests were applied (also sequentially starting first with March DFr, then March dADF, March PCFm and March PCFw). The total faults detected is 107; i.e., 3.8% of the total faults. Finally, Special Tests were applied sequentially in the following order: Scan, March SAM, Test BWE, Sacn, Gal9R and March SZ. The total faults detected in this phase is 15; i.e., 0.5% of the total faults. It should be noted that when tests are applied sequentially, if a test detects a defective device, then the remaining tests were not applied; the device is then counted as faulty.

Based on the two experiments, one can conclude that using dynamic tests to deal with dynamic faults has a large added value, especially when considering very low PPM levels for critical applications. The added value in the first experiment is about 2.4% additional fault coverage and in the second experiment about 3.8%. On the other hand, the added value (beyond existing BIST and Dynamic Tests) of Special Tests is 0.4% and 0.5% respectively.

VI. Results analysis of BWE Test

This section deals with the analysis of the added value of the BWE test in the two test experiments.

First experiment

The total FC achieved in the first experiment is 12544 (this includes faults detected by existing BIST, Dynamic Tests and Special Tests). The FC realized by Test BWE is only 273, from which 267 faults are also detected by other tests. Note that the FC of Test BWE is very low as the test is a very simple and cheap. However, the number of *unique faults* detected by such a test is 6. Unique faults are faults that are only detected once by a single test (in this case by Test BWE). As Test BWE is designed to target bit/byte write enable faults, the detected 6 unique faults are suspected to be related to bit/byte write enable faults. Failure analysis for a couple of such failures (which have a double-bit/partial row bitmap signature) have been done in order to diagnose the root cause of such faults. First the layout has been analyzed to check the most suspected/sensitive part where a defect can appear (e.g., region with dense tracks) and thereafter cross section have been obtained. The results reveals that the faults are caused by bridges between different line tracks of the bit/byte write enable circuitry.

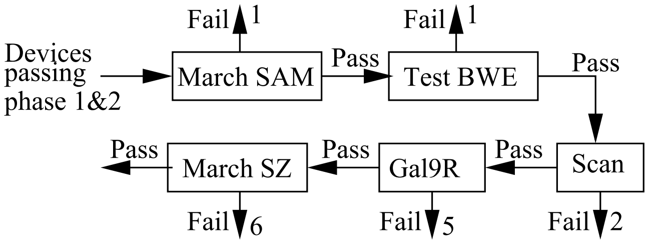

Fig. 5. Special Tests failure distribution

Second experiment

The total number of faults detected by Special Tests in the second experiment, where sequential testing was used (see Figure 4), is 15. The tests were applied in the following order: March SAM (to target intra-word faults), Test BWE (to target BWE faults), Scan, Gal9R and March SZ, respectively. Test BWE detects one unique fault. A similar diagnosis procedure has been done as in that in the first experiment; the results showed a bridging defect in the bit/byte write enable circuitry.

Note that with March SAM and Test BWE, well known and modeled faults are targeted. However, this is not 100% the case for the other three tests. Figure 5 shows the distribution of detected faults by Special Tests. Interesting enough that Scan used with a specifed addressing (Fast X) and data-background (row stripe) have the capability to detect unique faults. Scan is very powerful when used with such stresses to detect dynamic faults in write drivers [21]. Note also that Gal9R and March SZ detect many unexplained unique faults.

It should be concluded from the above that bit/byte write enable faults should also be considered for testing in order to achieve a high fault coverage. The claim that using standard march tests while enabling all bit/byte control signals will achieve a good fault coverage is wrong. Many faults within the bit/byte logic control circuit will not be covered.

VII. Conclusions

In this paper, an analysis of defects in bit/byte write enable control circuits of embedded memories has been presented. Fault models and an appropriate test detecting all possible bit/byte write enable faults has been proposed. The test solution, together with advanced memory test solutions, have been added to an existing BIST controller to industrially evaluate the added value of such test. The results of the experiments showed that the conventional way of testing bit/byte write enable faults may not always detect all the faults and therefore a specific test to target such faults have to be considered for serious test purposes.

References

[1] S.Adham, B. Nadeau-Dostie, "A BIST for Bit/Group Write Enable Faults in SRAMs"', *in Proc. IEEE Int'l Workshop on Memory Technology, Design and Testing*, 2004.

[2] T. Powell, *et. al*, "Chasing Subtle Embedded RAM Defects for Nanometer Technologies". *in Proc. IEEE Int'l Test Conf.*, paper 33.4, 2005.

[3] J. Otterstedt et al., "Detection of CMOS Address Decoder Open Faults with March and Pseudo Random Memory Tests", Proc. IEEE Int'l Test Conf., pp. 53-62, 1998.

[4] M. Sachdev, "Open Defects in CMOS RAM Address Decoders", IEEE Design and Test of Computers, pp. 26-33, Apr.-June 1997.

[5] M. Azimane and A.K. Majhi, "New Test Methodology for Resistive Open Defect Detection in Memory Address Decoders", Proc. IEEE VLSI Test Symp., pp. 123-128, 2004

[6] L. Dilillo, P. Girard, S. Pravossoudovitch, A. Virazel, S. Borri, M Hage-Hassan, "Dynamic read destructive fault in embedded-SRAMs: analysis and march test solution", In proc. Of IEEE European Test Symposium, pp. 140 - 145, 2004.

[7] S. Hamdioui, Z. Al-ars, A.J. van de Goor, "Opens and Delay Faults in CMOS RAM Address Decoders", IEEE Trans. Computers, Vol. 55, No. 12, pp. 1630-1639, Nov., 2006.

[8] B. Nadeau-Dostie, V.K. Agrawal, "Serial Interfacing for Embedded Memory Testing", *IEEE Design & Test of Computers*, pp. 52-63, 4/1990.

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

[10] R. Nair, "An Optimal Algorithm for Testing Stuck-at Faults Random Access Memories", IEEE transactions on Computers, Vol. C-28(3), pp. 258-261, 1979.

[11] M.A. Breuer and A.D. Friedman, Diagnosis and Reliable Design of Digital Systems, Computer Science Press, Woodland Hills, CA, USA, 1976.

[12] J.H. De Jonge and A.J. Smeulders, "Moving Inversions Test Pattern is Thorough, Yet Speedy", In Comp. Design, pp. 169-173, 1976.

[13] M. Marinescu, "Simple and Efficient Algorithms for Functional RAM Testing", In Proc. of International Test Conference, pp. 236-239, 1982.

[14] D.S. Suk and S.M. Reddy, "A March Test for Functional Faults in Semiconductors Random-Access Memories", IEEE Transactions on Computers, C-29(6), pp. 419-429, 1980.

[15] S. Hamdioui, A.J. van de Goor, and M. Rodgers, "March SS: A Test for All Static Simple RAM Faults", Proc. IEEE Int'l Workshop Memory Technology, Design and Testing, pp. 95-100, 2002.

[16] A.J. van de Goor, "Testing Semiconductor Memories, Theory and Practice", ComTex Publishing, Gouda, The Netherlands, (second edition), 1998.

[17] R.D. Adams and E.S. Cooley, "Analysis of Deceptive Destructive Read Memory Fault Model and Recommended Testing", Records North Atlantic Test Workshop, pp. 27-32, May 1996.

[18] S. Hamdioui, and A.J. van de Goor, "Experimental analysis of spot defects in SRAMs: realistic fault models and tests", Proc. Ninth Asian Test Symp., 2000, pp. 131-138

[19] Z. Al-Ars, A. J. van de Goor, Static and Dynamic Behavior of Memory Cell Array Spot Defects in Embedded DRAMs, IEEE Transactions on Computers, pp. 293-309, March 2003

[20] G. Harutunyan, V.A. Vardanian and Y. Zorian, "Minimal March Tests for Dynamic Faults in Random Access Memories", In Proc of ETS, pp. 43 - 48, 2006

[21] A.J. van de Goor, S. Hamdioui, and R. Wadsworth, "Detecting Faults in the Peripheral Circuits and Evaluation on SRAM Tests", Proc. IEEE Int'l Test Conf., pp. 114-123, 2004.

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

[23] S. Hamdioui and J.D. Reyes, "New Data-Background Sequences and Their Industrial Evaluation for Word-Oriented Random-Access Memories", IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems, Vol. 24, No. 6, pp. 892-904, June 2005.