Proceedings of Design and Diagnostics of Electronic Circuits and Systems Workshop (DDECS 98), Szczyrk, Poland, September 2-4, 1998, pp. 27-33

New Transparent RAM BIST Based on Self-Adjusting Output Data Compression

V. Yarmolik, Yu. Klimets
Belarussian State University of Informatics and Radioelectronics, Minsk, Belarus,

S. Hellebrand, H.-J. Wunderlich
University of Stuttgart, Germany

Abstract

The new memory transparent BIST technique is proposed in this paper. It has more higher fault coverage compare to classical transparent technique. Also this technique decreases the test complexity up to 50% for the most of march tests.

1. Introduction

The modern trend in IC technology toward merging logic and memory on the same chip has lead to the arising of the number devices with embedded RAM [1-4]. For example all modern microprocessors contain up to 1Mb embedded cache-memory of first and second levels. Since much of the circuitry of such devices is memory [1,2], the importance of memory testing increases. To solve this problem, and also to reduce the long testing times inherent to conventional external testing, a number of theoretical and practical built-in self-test (BIST) techniques have been proposed in the past [5-7]. But for many applications it is important to perform periodic testing of RAM [7]. In such case we need to test the memory parts of the chip between two cycles of the system operation. Besides the memory contents has to be restored after the completion of the test. In the traditional approach the contents of the memory-under-test is saved in some other data storage prior beginning the test. The contents is restored back when the testing is completed. However, such an approach can not be employed to test an embedded memory where the spare data storage is not available, and where a time delay due to the save-restore procedure is not acceptable.

An efficient way to solve this problem is to use transparent test technique [7-8]. The most significant result in transparent testing area was introduced by M. Nicolaidis in [7]. Unfortunately this technique has the following drawbacks:

- the first drawback is that this technique significantly increases test complexity due to necessity to calculate the reference signature before testing;
- this technique does not ensure 100% fault coverage even for single fault due to signature aliasing;
- the third drawback deals with decreasing of diagnostic ability of transparent test technique, because it is difficult to calculate the placement of faulty cell from the values of reference and working signatures.

Therefore in this paper we will present a new transparent RAM BIST technique which saves advantages of classical transparent technique and allows to eliminate above drawbacks.

2. Terminology and fault models

We use the following notations in this paper:

- **0 (1)** denotes that a cell is in logical state zero (one);
- **a**, $a \in (0,1)$ denotes that a cell is in logical state **x**, i.e. either 0 or 1;
- **w0 (w1)** denotes a write **0 (1)** operation to a cell;
- **r0 (r1)** denotes a read operation to a cell where a **0 (1)** is expected;

- w_a denotes a write a operation to a cell, where a is a value which is stored in data register;
- $w_{\bar{a}}$ denotes a write $\bar{a}$ operation to a cell, where a is a value which is stored in data register;
- r^* denotes a read operation to a memory cell, store value in data register followed by signature update operation;
- r^{-*} denotes a read operation to a memory cell and compact inverse value to the signature register;
- r_a denotes a read operation to a memory cell where the value stored in the data register is expected;
- $r_{\bar{a}}$ denotes a read operation to a memory cell where the inverse value to the stored in the data register is expected;
- $\Uparrow$ denotes an up addressing order;
- $\Downarrow$ denotes a down addressing order;
- $\Updownarrow$ denotes a don't care addressing order;
- $\oplus$ denotes an XOR operation;
- A(B) denotes a march tests phase, where A denotes an addressing order $A \in \{\Uparrow, \Downarrow, \Updownarrow\}$, B denotes the set of operation which are applied to a cell.

Following fault models are considered in this paper [9].

- *Stuck-at fault (SAF).* The logic value of a stuck-at cell is always 0 (SAF0) or always 1 (SAF1).
- *Transition fault (TF).* A cell fails to undergo a $0 \rightarrow 1$ (TF$\uparrow$), or a $1 \rightarrow 0$ (TF$\downarrow$).

A *coupling fault (CF)* involves two cells, when one cell (cell j) sensitizing the fault caused in another cell (cell i). Cell j is said to be the *coupling cell*, whereas cell i is said to be the *coupled cell*. Several types of CFs can be distinguished.

- An *inversion coupling fault (CFin)* is defined as follows: an $\uparrow$ or $\downarrow$ transition in the coupling cell causes an inversion in the coupled cell. This results in the following two CFin subtypes: $<\uparrow, \updownarrow>$ and $<\downarrow, \updownarrow>$.
- An *idempotent coupling fault (CFid)* is caused by an $\uparrow$ or $\downarrow$ transition write operation in the coupling cell which forces a certain value (0 or 1) in the coupled cell. The following four CFid subtypes exist: $<\uparrow,0>$, $<\downarrow,0>$, $<\uparrow,1>$, $<\downarrow,1>$.

Address decoder and read/write logic faults can be modeled as memory cell array faults, therefore they are not considered explicitly.

3. Classical transparent RAM testing

In this section we will describe the basic principles of classical transparent RAM testing. More detailed information can be founded in [7].

Every march test algorithm is composed by several march phases. Individual march phase traverses all RAM addresses and perform a specified combination of read and write operations. Usually the march test algorithms include an initialization phase. It is used only for setting the memory to a known state and is not useful for fault excitation. Usually, it is only one phase which is begin from the write operation.

The general rules for transforming any march test algorithm to a transparent one are presented below [7].

Let AL_0 be the initial algorithm.

Step 1. Delete the initialization phase from the algorithm AL_0. Let AL_1 be the algorithm obtained from step 1.

Step 2. On this step we substitute every write operations in algorithm AL_1 for transparent one. Let x be the value resulting by the preceding read operation and y be the data of the write operation we want to replace. If x=y then substitute this write operations for w_a operation, otherwise - for $w_{\bar{a}}$ operation. Let AL_2 be the algorithm representation resulting by the step 2.

Step 3. Substitute all read operations r0 and r1 for r^* operation in the algorithm AL_2. Let AL_3 be the algorithm obtained from step 3.

The algorithm AL_3 is a transparent test algorithm. It is called the basic transparent test algorithm. Step 4 is used to perform the signature prediction. The resulting algorithm AL_4 will be called the signature prediction algorithm.

Step 4. Delete all write operations from algorithm AL_1 and substitute all r0 read operations for r^* operation, all r1 operation - for $r\text{-}^*$ operation.

Example: let's consider all these step for March B algorithm transformation.

The initial record of March B is the follow:

AL_0: ⇕ (w0); ⇑(r0,w1,r1,w0,r0,w1);⇑ (r1,w0,w1);⇓ (r1,w0,w1,w0); ⇓ (r0,w1,w0).

On the first step we delete initialization phase and obtain the following algorithm:

AL_1: ⇑(r0,w1,r1,w0,r0,w1);⇑ (r1,w0,w1);⇓ (r1,w0,w1,w0); ⇓ (r0,w1,w0).

Then on the second step we substitute all write operations. As a result:

AL_2: ⇑(r0, $w_{\neg a}$,r1, $w_{\neg a}$,r0, $w_{\neg a}$);⇑ (r1, $w_{\neg a}$, w_a);⇓ (r1, $w_{\neg a}$, w_a, $w_{\neg a}$); ⇓ (r0, $w_{\neg a}$, w_a).

On the third step we substitute all read operations. As a result:

AL_3: ⇑(r^*, $w_{\neg a}$, r^*, $w_{\neg a}$, r^*, $w_{\neg a}$);⇑ (r^*, $w_{\neg a}$, w_a);⇓ (r^*, $w_{\neg a}$, w_a, $w_{\neg a}$); ⇓ (r^*, $w_{\neg a}$, w_a). This is a transparent test algorithm.

On the fourth step we obtain a signature prediction algorithm:

AL_4: ⇑(r^*, r^*, r^*);⇑ (r^*);⇓ (r^*); ⇓ (r^*).

Now we will describe the overall procedure of memory testing.

1. During the first step the reference signature C_{REF} has to be determined by running a signature prediction algorithm AL_4. The obtained signature is written to the register of reference signature.
2. During the second step of testing the test working signature is determined by running a transparent test algorithm AL_3.
3. On the third step both signatures C_{REF} and C_{TEST} have be compared to provide the test result.

It is easy to note from the above description that transparent RAM testing technique has the following drawbacks:

- A first drawback is that this technique significantly increases test complexity (about 40%-50% for most march tests) due to necessity to calculate the reference signature before testing. Our approach preserves original complexity of the test procedure.
- A second drawback is that this technique does not ensure 100% fault coverage even for single fault due to signature aliasing. As we show later our technique allows to 100% detect all fault of single and double quantity.

A third drawback deals with decreasing of diagnostic ability of transparent test technique, because it is difficult to calculate the placement of faulty cell from the values of reference and working signatures. Our technique give the exact faulty cell address by performing simple XOR operation on the reference and working signature.

4. The basic principles of self-adjusting output data compression

Our transparent memory RAM BIST is based on self-adjusting output data compression. In this section we give a short survey into the basic principles and peculiarities of self-adjusting output data compression technique.

In [10] was introduced a new scheme for output data compression which is based on a new memory characteristic derived as the modulo-2 sum of all addresses pointing to nonzero cells. This characteristic can be adjusted concurrently with write operations by simple EXOR-operations on the initial characteristic and on the addresses affected by the change.

As illustrated in Figure 1, the proposed reference characteristic C_{REF} of the correct initial memory contents is defined as the modulo-2 sum of all memory addresses pointing to cells containing a 1.

The compressor circuit simply has to perform bitwise EXOR-operations on addresses controlled by the value stored in the data register.

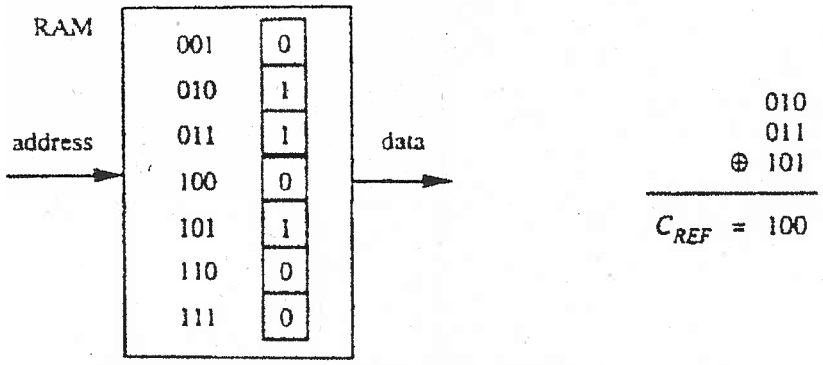

Figure 1. Modulo-2 address characteristic for bit-oriented RAMs

Besides the simple mechanism of mapping changes in memory to changes in the characteristic the proposed scheme has a number of additional advantageous properties.

Property 1. All single errors are detectable and diagnosable, since if only single errors are assumed, the expression $C_{REF} \oplus C_{TEST}$ provides the address of the faulty memory cell.

Property 2. All double errors are detectable since in this case the expression $C_{REF} \oplus C_{TEST}$ corresponds to the modulo-2 sum of two faulty cells addresses ADR_1 and ADR_2.

Property 3. The signatures for normal and inversion memory contents will be identical. Indeed, the signature for memory, where all cells contain 1's, is equal to 000... 0 (because it is the modulo-2 sum of all binary numbers from 0 up to 2m -1, where m is the address length of the memory).

Property 4. The final values of signature does not depend from the direction of march test phases.

In the next section we will describe the overall procedure of testing in the our new transparent memory BIST.

5. General rules for transforming march test into transparent one

Our technique is also based on the idea of using the RAM's contents in order to test it. We use the following steps to transform classical march test into transparent one.

Let AL_0 be the initial algorithm.

Step 1. Delete the initialization phase from the algorithm AL_0. Let AL_1 be the algorithm obtained from step 1.

Step 2. On this step we substitute every write operations in algorithm AL_1 for transparent one. Let x be the value resulting by the first read operation in phase and y be the data of the write operation we want to replace. If x=y then substitute this write operations for w_a operation, otherwise - for $w_{\bar{a}}$ operation. Let AL_2 be the algorithm representation resulting by the step 2.

Step 3. Substitute all first read operation in every phase for r^* operation in the algorithm AL_2. Let AL_3 be the algorithm obtained from step 3.

Step 4. On this step we substitute every read operations in algorithm AL_3 for transparent one. Let x be the value resulting by the preceding write operation in phase. If x=a then substitute this read operations for r_a operation, otherwise - for $r_{\bar{a}}$ operation. Let AL_4 be the algorithm representation resulting by the step 4.

The algorithm AL_4 is a transparent test algorithm. As a signature prediction algorithm we use simple one-phase algorithm $\Updownarrow (r^*)$ for any march test.

Example: let's consider all these step for March B algorithm transformation.

The initial record of March B is the follow:

AL_0: $\Updownarrow (w0)$; $\Uparrow(r0,w1,r1,w0,r0,w1)$;$\Uparrow (r1,w0,w1)$;$\Downarrow (r1,w0,w1,w0)$; $\Downarrow (r0,w1,w0)$.

On the first step we delete initialization phase and obtain the following algorithm:

AL_1: $\Uparrow(r0,w1,r1,w0,r0,w1)$;$\Uparrow (r1,w0,w1)$;$\Downarrow (r1,w0,w1,w0)$; $\Downarrow (r0,w1,w0)$.

Then on the second step we substitute all write operations. As a result:

AL_2: $\Uparrow(r0, w_{\bar{a}},r1, w_a,r0, w_{\bar{a}})$;$\Uparrow (r1, w_{\bar{a}}, w_a)$;$\Downarrow (r1, w_{\bar{a}}, w_a, w_{\bar{a}})$; $\Downarrow (r0, w_{\bar{a}}, w_a)$.

On the third step we substitute first read operations in all phases. As a result:

AL_3: $\Uparrow(r^a, w_{-a}, r1, w_a, r0, w_{-a}); \Uparrow (r^a, w_{-a}, w_a); \Downarrow (r^a, w_{-a}, w_a, w_{-a}); \Downarrow (r^a, w_{-a}, w_a)$.

On the fourth step we substitute all the rest read operations. As a result:

AL_4: $\Uparrow(r^a, w_{-a}, r_{-a}, w_a, r_a, w_{-a}); \Uparrow (r^a, w_{-a}, w_a); \Downarrow (r^a, w_{-a}, w_a, w_{-a}); \Downarrow (r^a, w_{-a}, w_a)$. This is a transparent test algorithm. The signature prediction algorithm is the same for any march test: $\Uparrow(r^a)$.

Now we will describe the overall testing procedure of memory BIST based on the self-adjusting output data compression.

1. During the first step the reference signature C_{REF} has to be determined by running a signature prediction algorithm $\Uparrow(r^a)$. The obtained signature is written to the register of reference signature. This step can be eliminated for all following test executions by self-adjusting reference signature concurrently with normal memory operations as it will be shown in next section.
2. During the second step of testing the test working signature is determined by running a first phase of the transparent test algorithm AL_4.
3. On the third step both signatures C_{REF} and C_{TEST} have be compared to provide the first phase result. They are have to be equal for fault-free memory.
4. Repeat steps 2-3 for all phases of algorithm AL_4.

Such testing algorithm has one advantages compare to classical transparent algorithm. It allows to achieve more better fault coverage due to decreasing the length of compacting sequence. We compare working and reference signatures after every test phase. Therefore the length of compacting sequence decreases in k times compare to classical transparent technique, where k is a number of phases in march test. It allows to detect 100% of all single and double fault in memory (of course if these faults are detected by march test).

It is impossible to compare reference and working signatures after every test phase in classical transparent technique because in this case the value of signature depends from the direction of test phase, number of read operation in test phase and inverse or non-inverse memory content to the initial state of the memory before testing.

6. Test complexity decreasing by signature prediction algorithm elimination

Signature prediction algorithm can be eliminated for all test executions except the first time due to possibility of reference signature self-adjusting.

During memory operation C_{REF} must be updated after every write operation. As explained is Section 4.1 this implies that for each write operation the "difference", i.e. the bitwise EXOR of the old and the new memory entry has to be determined. A key issue in implementing of the such BIST approach is to ensure a backup of the old memory entry at a low hardware and performance penalty. The best method to achieve this goal, of course, depends on the memory organization. For example a refreshment algorithm for the dynamic RAM can be used which writes the complete row containing the word targeted by a write request to the refreshment register before loading the new word from data register. In this case the old memory entry can be transferred from the refreshment register to the test register via the switching matrix.

In such a way we can decrease test complexity for all test executions except the first one.

7. Example of transparent memory BIST based on self-adjusting output data compression

In this section we present an example of the new transparent version of test algorithm March C- and the corresponding BIST implementation. We have choose March C- because it is wide-spread algorithm with excellent coverage ability and small number of operations.

The test March C- is presented as following:

March C-(10n): $\Updownarrow (w0); \Uparrow(r0,w1); \Uparrow (r1,w0); \Downarrow (r0, w1); \Downarrow (r1,w0); \Updownarrow (r0)$.

It performs 10n read and write operations and detects all stuck-at, transition and coupling faults in memory. By using above steps this algorithm is transformed into a transparent test algorithm (9n): $\Uparrow(r^a,w_a); \Uparrow(r^a,w_a); \Downarrow(r^a,w_{-a}); \Downarrow(r^a,w_{-a})$. The signature prediction algorithm is the same for any march test (1n): $\Uparrow(r^a)$.

The diagram in Figure 2 shows the BIST architecture.

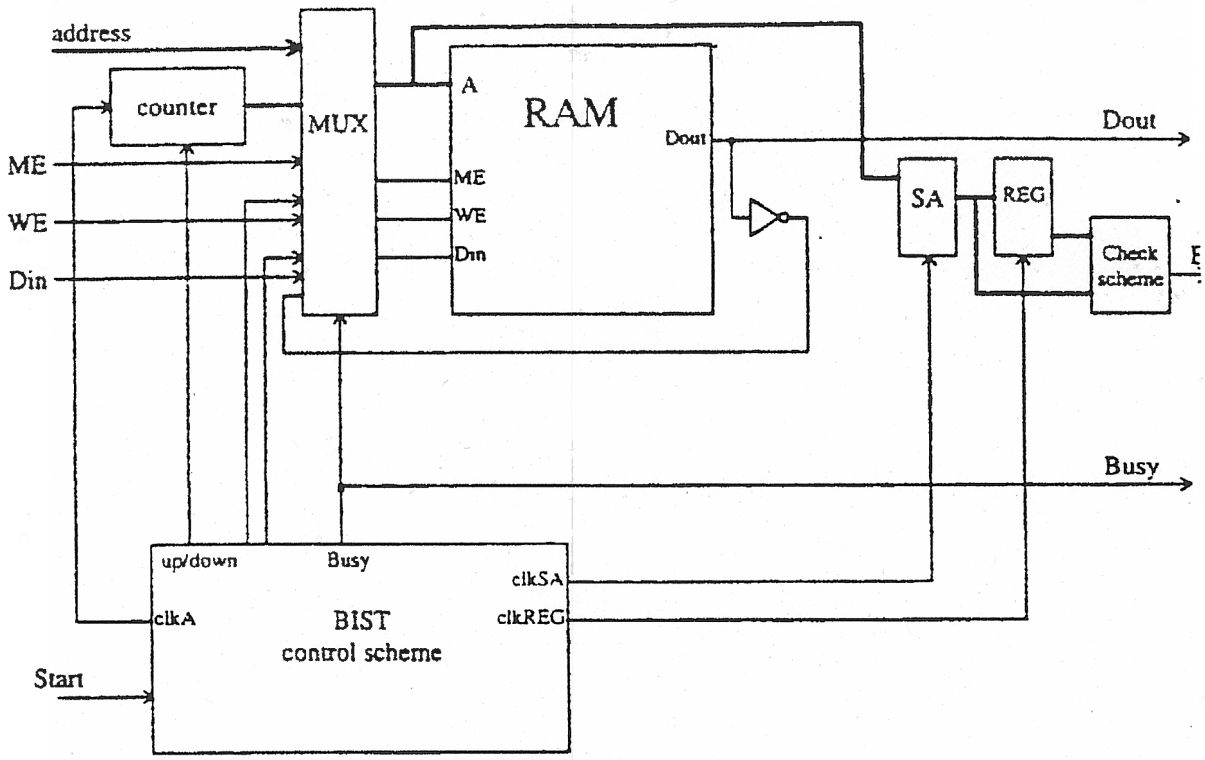

Figure 2. Memory BIST functional diagram

1) The high level of the signal **Start** starts the BIST operation. Once a memory chip is in self-test mode, the BIST circuit takes over control of all signals such as **ME, WE, A, Din** by turn on the signal **Busy**. At the conclusion of self-test signal **Busy** falls down and signal **Err** indicate the result of self-test.

2) During the first step of testing procedure the reference signature C_{REF} has to be determined by running the signature prediction algorithm $\Uparrow(r^*)$. This signature is stored in the register **REG** by signal **clkREG**.

3) During the second step we execute transparent test algorithm March C-:

$\Updownarrow(w0)$; $\Uparrow(r0,w1)$;$\Uparrow(r1,w0)$;$\Downarrow(r0,w1)$; $\Downarrow(r1,w0)$; $\Updownarrow(r0)$. In our case we used static RAM memory therefore we don't need in flip-flop to store out data from memory. Due to peculiarity of March C- we used inverter scheme instead of. After every test phase the reference and working signatures are compared by **Check scheme**. Functional diagram of signature analyzer is presented on Figure 3. It consists from **m** synchronous T-flip-flops (where **m** is a number of address bit).

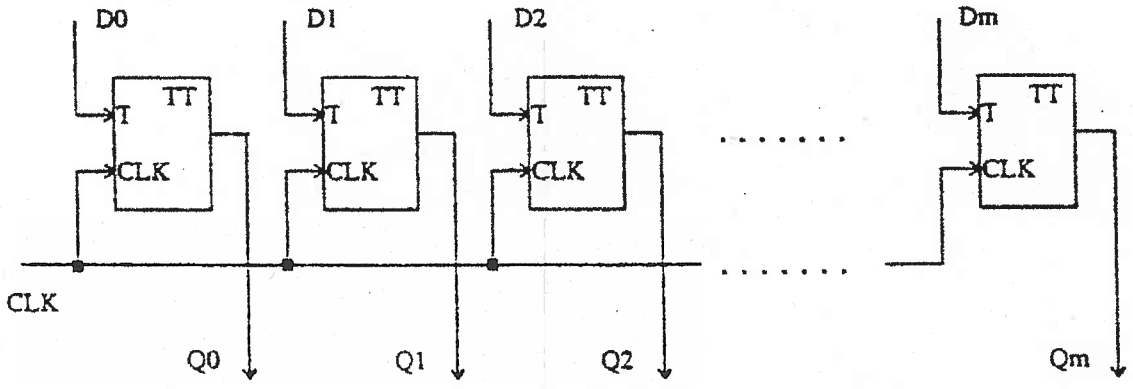

Figure 3. Signature analyzer for new transparent memory BIST

The overall complexity of tests is **9n+1n=10n** while for classical transparent test technique we have to perform **9n+5n=14n** operations. For BIST with self-adjusting of reference signature the overall complexity of test will b **9n+1n=10n** for first time of test execution and **9n** for all following times due to self-adjusting of the referenc signature concurrently with memory operations.

Fault coverage for classical and new transparent test technique is presented in the Table 1.

We used for simulation 32Kbit DRAM and BIST based on March C- with 15-bit signature analyzer. It's easy to note that all single and double faults are detected by new scheme while the classical transparent technique is failed due to fault masking.

Fault type	SAF			TF			CFin			CFid		
Fault multiplicity	1	2	3	1	2	3	1	2	3	1	2	3
Classical transparent BIST, %	99.1	99.2	99.2	99.3	99.2	99.1	99.8	99.3	99.1	99.6	99.2	99.1
New transparent BIST, %	100	100	99.8	100	100	99.7	100	100	99.9	100	100	99.9

Table 1. Fault coverage for classical and new transparent techniques

The more higher fault coverage for new transparent technique is achieved due to decreasing of the compacted sequence as it is shown in the Table 2.

Memory size	Classical transparent BIST	New transparent BIST
32Kbit	163.840	32.768
1Mbit	5.242.880	1.048.576
16Mbit	83.866.080	16.777.216

Table 2. Compacted sequence length for transparent BIST based on March C-

For classical transparent technique the length of the compacted sequence is equal to the product of number of read operations in march test and memory size. For our technique the length of compacted sequence is equal to the memory size. Therefore for equal lengths of signature analyzer our technique provides more higher fault coverage.

8. Conclusion

In this paper we have proposed the new memory transparent BIST technique. Our technique has some advantages compare to classical transparent technique because it has more higher fault coverage. Also our technique decreases the test complexity up to 50% for the most of march tests.

References

1. L. Youngs, S. Paramanandam, "Mapping and Repairing Embedded-Memory Defects", *IEEE Design & Test of Computers*, January-March 1997, pp. 18-24.
2. R. Wu, et. al., "Testing Logic-Intensive Memory ICs on Memory Testers", *IEEE Design & Test of Computers*, January-March 1997, pp. 50-54.
3. M. Sachdev, M. Verstraelen, "Development of a Fault Model and Test Algorithms for Embedded DRAMs", *IEEE International Test Conference*, 1993, pp. 815-824.
4. S. Barbagallo, F. Corno, P. Prinetto, M. Sonza Reorda, "Testing a Switching Memory in a Telecommunication System", *IEEE International Test Conference*, Washington, DC, Oct. 1995, pp. 947-953.
5. R. Dekker, F. Beenker, and L. Thijssen, "Realistic Built-In Self-Test for Static RAMs", *IEEE Design & Test of Computers*, Vol. 6, No. 1, Feb. 1989, pp.26-34.
6. K. T. Le, K. K. Saluja, "A Novel Approach for Testing Memories Using a Built-In Self-Testing Technique", *IEEE International Test Conference*, Washington, DC, 1986, pp. 830-839.
7. M. Nicolaidis, "Transparent BIST for RAMs", *IEEE International Test Conference*, Baltimore, MD, Oct. 1992, pp. 598-607.
8. Yarmolik V., Karpovski M., "Transparent Memory BIST", *IEEE International Workshop on Memory Technology, Design and Testing*, 1994, pp.106-111.
9. A. J. van de Goor, *Testing Semiconductor Memories. Theory and Practice*. Wiley, Chichester, 1991.
10. V. Yarmolik, H.-J. Wunderlich, S. Hellebrand, "Self-Adjusting Output Data Compression: An Efficient BIST Technique for RAMs", *DATE*, 1998, pp. 173-179.