ITC 94

Transparent Memory Testing for Pattern Sensitive Faults*

M. G. Karpovsky

Dep. Elec., Comp. & Syst. Eng.
College of Eng., Boston University
Boston, MA 02215, USA

V. N. Yarmolik

Computer Science Dep.
Minsk Radio Eng. Inst.,
Minsk, 220027, BELARUS

Abstract: *This paper presents a new methodology for RAM testing based on PS(n,k) fault model (the k out of n pattern sensitive fault model). According to this model the contents of any memory cell which belongs to an n-bit memory block, or ability to change the contents, is influenced by the contents of any $k-1$ cells from this block. This paper includes the investigation of memory testing approaches based on the transparent pseudoexhaustive testing and its approximations by pseudorandom circular tests, which can be used for periodic and manufacturing testing and require lower hardware and time overheads than the standard approaches.*

Key words: Memory testing, transparent memory testing, built-in self-test (BIST), pseudoexhaustive memory testing, signature analysis.

1 Introduction

Efficient techniques for transparent BIST testing were proposed by B.Koeneman and M.Nicolaidis [1]. These techniques can be applied to any RAM test patterns for manufacturing and periodic testing. Both techniques are based on addition of predicted test patterns to the memory-under-test contents which provides for preservation of initial contents of the RAM.

As the first step in development of a test for off-line RAM testing, a fault model representing physical defects is constructed [2, 3]. Typically, the test procedure is intended to detect all faults for the selected model (target faults).

On-line testing is the main tool that ensures high reliability of storage data during operational life of the memory by employing on-chip error-detecting/correcting circuits.

It is obvious, that target faults for off-line testing have the corresponding description as target errors used for on-line testing and, vice versa [4]. But nevertheless, traditionally, off-line testing and on-line testing have not been combined to minimize hardware and time overheads. Moreover, built-in self-testing approaches are not using a powerful built-in hardware, required for on-line testing. During manufacturing and field off-line testing, the built-in error-detection/correction circuits are ignored.

As an alternative to the existing development of two separate approaches for RAMs testing, namely on-line and off-line testing, we present in this paper a methodology for RAM's testing, which can be used for periodic and manufacturing testing. For periodic testing our approach ensures the preservation of the initial contents of the memory and provides for a high fault coverage for target faults.

We propose in this paper a unified approach for transparent memory BIST based on the general fault model. According to this model the contents of any cell which belongs to a n bit memory block, or ability to change the contents, is influenced by the contents of any $k-1$ cells of the block. These contents consists of a pattern of 0s and 1s or changes in these contents. We denote these faults as PS(n,k) faults.

The following problems will be explored: (a) analysis of efficiency of the selected PS(n,k) fault model, (b) design of pseudoexhaustive transparent tests for PS(n,k)s, (c) design of pseudorandom transparent circular tests, (d) BIST RAM implementations of circular tests, (e) analysis of performances (hardware and time overheads, error/fault detection probabilities, fault coverage) of the proposed tests, (f) off-line utilization of error detection/correction memory circuits.

2 Fault Models Investigation

A good survey of the commonly used fault models for RAMs can be found in [5]. Here, we review only the background required for understanding of the proposed general fault model.

By a simple fault we mean any *single fault*, which is not a *pattern sensitive fault*. The following simple faults are frequently considered:

Stuck-at faults (SAF): A permanent stuck-at 0 or

*This work was supported by the NSF under Grant MIP92-08487, the NRC under Grant "Testing of Computer Hardware" and the NATO under Grant 910411.

stuck-at 1 fault that may occur in any memory cell.

Transition faults (TF): A memory cell i in the state $a_i \in \{0,1\}$ fails to undergo an a_i to $\overline{a}_i$ transition when $\overline{a}_i$ is to be written in the cell; however, both states are possible for the cell, for instance at power-on.

Coupling idempotent faults (CF_{id}): A coupling idempotent fault is present from a cell i to a cell j if, when the cells contain a particular pair of binary values a_i and a_j, and $\overline{a}_i$ is written into cell i, then cell j, as well as cell i, changes state.

Coupling inversion faults (CF_{in}): A single coupling inversion fault is present from a cell i to a second cell j if, when cells i, j has values a_i and a_j, and $\overline{a}_i$ is written into cell i, then the state of cell j is complemented or "toggled" with respect to its previous value.

The above coupling faults, which involves two cells (*2-couplings*) is a special case of the more general *k-coupling fault*.

A *pattern sensitive fault* (PSF) is most general memory fault model defined as follows: the contents of cells, or the ability to change the contents, is influenced by the contents of all other cells in the memory. These contents consists of a pattern of 0s and 1s, or changes in these contents [5].

The common restriction on the class of PSF is the *size n of the neighborhood*, and the number k of cells involved in the fault [5]. For $k = 2$ and $n = N$, the cell can be influenced by any other cell. This is a case of coupling faults [6, 7, 8].

There is a restricted k-coupling not linked (i.e. disjoint) fault, where $k = 3$ [7] and the test complexity for this case is $N + 32 \cdot N \cdot \log_2 N$. The second known test algorithm for these faults requires $36 \cdot N + 24 \cdot N \cdot \log_2 N$ operations [8]; thus both are $O(N \log_2 N)$ tests, so they are of less practical interest. For the general k-coupling fault with $k > 3$ no deterministic test exist of this date.

Another common restriction on PSF is that the base cell is influenced by a neighborhood of $k-1$ cells which have only one position with respect to the base cell. This case represents NPSFs [9]. The efficiency of the NPSF tests depends on the technical realization of the memory. Within most memory chips the address lines, the memory rows and the columns are scrambled; so, cells with logically ascending addresses will probably not necessarily lay next to each other. When the scrambling table is not known, a test for NPSF is less useful

].

Standard memory test algorithms presented in [2, 6] an be used only when each address refers to exactly ne bit of memory. As an example Marinesku B algoithm [6] and $MATS+$ algorithm [2] are shown in Table 1. When we test a word-oriented memory, we write o and read from the memory whole words of data, not

Table 1: Memory test algorithms — one bit version

	MATS+		Marinesku B
S_0	$\Downarrow$ $(w0)$	S_0	$\Downarrow$ $(w0)$
S_1	$\Downarrow$ $(r0,\ w1)$	S_1	$\Downarrow$ $(r0,\ w1,\ w0,\ w1)$
S_2	$\Uparrow$ $(r1,\ w0)$	S_2	$\Downarrow$ $(r1,\ w0,\ r0,\ w1)$
		S_3	$\Uparrow$ $(r1,\ w0,\ w1,\ w0)$
		S_4	$\Uparrow$ $(r0,\ w1,\ r1,\ w0)$

just single bits. To achieve a high fault coverage data backgrounds to each word can be applyed [10]. For example, 0101, 0011, 0000, 1010, 1100, 1111 and 10101 010, 00110011, 00001111, 00000000, 01010101, 110011 00, 11110000, 11111111 are an appropriate primary data backgrounds for testing a RAM with 4 and 8 bits per word (a four bit and byte oriented RAM). In general, for a word with m bits number of backgrounds is $2(\lceil \log_2 m \rceil + 1)$.

Current memory BIST approaches [11, 12] based on standard memory tests have the following major drawbacks: a lower efficiency for detection a non-target fault models, the hardware and time overheads, a destructive (not transparent) test procedure, a fixed (not flexible) test mode, which usually allow to generate one or two test algorithms only. These approaches are not tailored to the most efficient combined utilization of on-line and off-line test circuitry.

As a realistic and sufficiently general fault model we propose the k out of n pattern sensitive fault (PS(n,k)) model, where $n < N$ is a size of a block (region, row or column of memory array) for memory with the size N. According to this model the contents of any memory cell which belongs to a n bit memory block, or ability to change the contents, is influenced by the contents of any $k-1$ cells in the same block. These contents consists of a pattern of 0s and 1s or changes in these contents.

This PS(n,k) fault model can be considered as an extension of the existing fault models. Depending on the values of k and n, PS(n,k) generalize the following cases considered in the literature:

For $k = n = N$, PS(n,k) represents PSF [5], when the contents of a cell, or the ability to change the contents, is influenced by the contents of all other cells in the memory.

For $k < n$ and $n = N$ we have the case of the general k-coupling faults, when the base cell is influenced by a group of $k-1$ cells which can be placed anywhere in the memory.

For $k = n < N$ PS(n,k) is reduced to NPSFs, where

the base cell influenced by the $k-1$ cells, which take on only a single position.

The case when $k < n$ and $n < N$ or $n \ll N$ is more common and realistic one, when the contents of any memory cell belongs to an n-bit memory block, or ability to change the contents, is influenced by the contents of any $k-1$ cells of the same block.

The major goal of this paper is to develop a unified approach for detection PS(n,k) faults.

In the following sections we are developed a unified methodology for memory testing based on the PS(n,k) model and will introduce a pseudoexhaustive transparent memory testing (PXT) approach for detection of PS(n,k) faults. This approach provides for a 100% fault coverage for PSFs with the complexity $C \cdot N$ where C depends on n and k. In most cases C is $O(k2^k n \log_2 n)$. The major disadvantage of PXT is related to the required overhead which in this case is also $O(k2^k n \log_2 n)$. To reduce the overhead an approximation to PXT is suggested. This approximation is based on pseudorandom circular test sequences, which can be used for periodic and manufacturing testing.

3 Pseudoexhaustive Transparent Memory Testing

Pseudoexhaustive testing (PXT) [13, 14] of combinational devices has several attractive features. In addition to the fact that test patterns can be generated quit easily, the process and its fault coverage is basically dependent neither on the fault model assumed nor on the specific circuit under test. This approach guarantees 100% coverage for all combinational faults. The major disadvantage of PXT is related to the fact that this approach is efficient only for combinational devices such that every output of the device depends on a small number of inputs.

Set $E(n,k)$ of n-bit binary vectors is a (n,k) PXT iff all 2^k vectors appear at any k positions in $E(n,k)$. For example, $E(3,2) = \{000, 011, 101, 110\}$. Techniques for construction of $E(n,k)$ and estimations on minimal numbers $f(n,k) = \mid E(n,k) \mid$ of pseudoexhaustive test patterns can be found in [13, 14, 15].

We will show in this section that PXT combined with standard March tests may be efficient for transparent memory testing for detection of PS(n,k) faults.

The problem of pseudoexhaustive transparent memory testing can be formulated as construction of a test procedure for memory testing such that:

Each memory cell within k out of n cells in the block must be read in state 0 and in state 1, for all possible changes in remaining $k-1$ cells.

Each memory cell within k out of n cells must be written and read in state 0 and in state 1, for all 2^{k-1} contents of the pattern in $k-1$ cells.

Table 2: Pseudoexhaustive test patterns $E(n,k)$

$E(3,2)$	$E(4,3)$	$E(5,3)$	$E(6,2)$	$E(6,3)$
111	0000	10000	000000	011111
100	0011	01000	000011	101111
010	0110	00100	011100	110111
001	0101	00010	101101	111011
	1100	00001	110110	111101
	1111	01111	111011	111110
	1010	10111		100000
	1001	11011		010000
		11101		001000
		11110		000100
				000010
				000001

These properties should be satisfied for any k cells in any block of n cells.

To simplify the test procedure we use the standard transparent march test MATS+ [5], which ensures 1 out of n exhaustive tests $E(n,1)$ such that in every 1 memory cell all 2^1 possible binary values 0 and 1 appear at least once. We apply MATS+ for different backgrounds $A(n,k)$ which represent a modulo two sums of $f(n,k)$ patterns of a pseudoexhaustive test, $E(n,k)$, with an initial n-bit memory block content A. The proposed test procedure PXT consists of the following main stages:

1. A new background $A_{j+1}(n,k)$ is generated as a modulo two sum of the previous value of $A_j(n,k) = A \oplus E_j(n,k)$ and $\Delta E_{j+1} = E_j(n,k) \oplus E_{j+1}(n,k)$. As the result we will get a modulo two sum the next pattern $E_{j+1}(n,k)$ of pseudoexhaustive test $E(n,k)$ with an initial n-bit memory block content $(A_{j+1}(n,k) = A \oplus E_{j+1}(n,k))$.

2. The transparent MATS+ memory test algorithm is applied.

3. The items 1 and 2 are repeated for all $f(n,k)$ backgrounds.

For example, if $E(3,2) = \{000, 011, 101, 110\}$, then $A(3,2) = \{a_0a_1a_2, a_0\overline{a}_1\overline{a}_2, \overline{a}_0a_1\overline{a}_2, \overline{a}_0\overline{a}_1a_2\}$. For $E(5,2) = \{11111, 10000, 01000, 00100, 00010, 00001\}$ we will get $A(5,2) = \{\overline{a}_0\overline{a}_1\overline{a}_2\overline{a}_3\overline{a}_4, \overline{a}_0a_1a_2a_3a_4, a_0\overline{a}_1a_2a_3a_4, a_0a_1\overline{a}_2a_3a_4, a_0a_1a_2\overline{a}_3a_4, a_0a_1a_2a_3\overline{a}_4\}$.

Some examples of $E(n,k)$ test patterns are shown in Table 2. Transparent memory test algorithm MATS+ (one-bit version) can be represented as [1]

$$\{\Downarrow [r(a_i), w(\overline{a}_i)]; \Uparrow [r(a_i), w(\overline{a}_i)]\} \qquad (1)$$

This test procedure allows to generate all of 2^{k+1} $k+1$-bit test patterns in each $k+1$ out of n memory cells for any block of n cells and all cells within a $k+1$ bits implement ↑ and ↓ transitions for all combinations of the remaining k cells which allows to detection of all PS(n,k) faults.

Two additional memory arrays are required for BIST [11] implementation of the pseudoexhaustive transparent memory testing. One of them S for storing an initial content of n-bit memory block, the second one E is the memory for $f(n,k)$ by n bit pseudoexhaustive test patterns.

The following simple procedure can be used for BIST implemented pseudoexhaustive transparent memory testing.

Procedure PXT

Input:	Memories block size n;
	Starting address w;
Loop1:	For $i=0$ to $n-1$ Do
Write:	$[a_{w+i}]$ to s_i
End:	Loop1
Loop2:	For $j=0$ to $f(n,k)-1$ Do
Loop3:	For $i=0$ to $n-1$ Do
Write:	$[a_{w+i}] \oplus [e_i^j]$ to a_{w+i}
End:	Loop3
Loop4:	For $i=0$ to $n-1$ Do
Read:	$[a_{w+n-1-i}]$ to D_0 and Compare
	with $[s_{n-1-i}] \oplus [e_{n-1-i}^j]$;
Write:	$[D_0] \oplus 1$ to $a_{w+n-1-i}$;
End:	Loop4
Loop5:	For $i=0$ to $n-1$ Do
Read:	$[a_{w+i}]$ to D_0 and Compare
	with $[s_i] \oplus [e_i^j] \oplus 1$;
Write:	$[D_0] \oplus 1$ to a_{w+i};
End:	Loop5
End:	Loop2

Here D_0 is an additional D-flip-flop; $[a_i]$ means the content of the ith memory cell; $\Delta E_j(n,k) = e_0^j e_1^j e_2^j \dots e_{n-1}^j$, $j \in \{0,1,2,\dots,f(n,k)-1\}$ is determined by the patterns $E_j(n,k)$ and $E_{j+1}(n,k)$ of the pseudoexhaustive test $E(n,k)$, where $\Delta E_0 = E_0$, and $\Delta E_c = E_c - E_{c-1}, c \in \{1,2,3,\dots,f(n,k)-1\}$.

Pseudoexhaustive transparent memory testing procedure for the case of PS(5,2), where $k=2$, $n=5$, $A = 00000$ and backgrounds generated by the test $E(5,2) = \{11111, 10000, 01000, 00100, 00010, 00001\}$ are shown in Table 3. Here $\Delta E(5,2) = \{11111, 01111, 11000, 01100, 00110, 00011\}$. (The bit we Read or Write at a given step of the procedure is printed in bold in Table 3.)

In this case first we write in the block of five 1-bit cells pseudoexhaustive pattern $11111 \in E(5,2)$ then read rightmost bit a_4, and write 0 in this cell, then read 1 in the next bit a_3 and write 0 in this cell. After zeros are written in all 5 cells we read the leftmost bit a_0 and write 1 in this cell. After ones are written in all cells we repeat the procedure for the next pseudoexhaustive pattern $10000 \in E(5,2)$. As we can see the PXT ensures detectability of the PS(5,2) faults, which for $k=2$ covered reduced functional faults i.e. SAF, TF and CF.

Table 4: The values of $f(n,k)$

n	k						
	2	3	4	5	6	7	8
4	5	8	16				
5	6	10	16	32			
6	6	12	21	32	64		
8	6	12	24	56	85	128	256
10	6	12	24	90	165	240	341
12	7	16	24	118	261	440	715
14	7	18	52	118	357	728	1365
16	8	18	54				

The complexity of the above presented memory test algorithm depends on the complexity of MATS+ algorithm and the value of $f(n,k)$. For the total amount of *Write* and *Read* operations we have

$$T[PXT(n,k)] = 5\ n \cdot f(n,k) + n, \qquad (2)$$

where $f(n,k)$ depends on k and n. Table 4 presents the values or their upper bounds for some n and k. For the overhead we have

$$L[PXT(n,k)] = n\ f(n,k) + n. \qquad (3)$$

Formula (3) does not take into account a complexity of a controller required for BIST implementation of PXT. The complexity of the controller can be estimated as

$$L_{CON}[PXT(n,k)] = O(\log_2 N) + O(\log_2 n). \qquad (4)$$

Since $f(n,3) \le 10 \log_2 n$ [14, 15], then

$$T[PXT(n,3)] \le 5\ n \cdot f(n,3) + n = 50\ n \cdot \log_2 n + n. \qquad (5)$$

Only two algorithms [7, 8] for the case of 3-coupling faults, which detect so-called 'restricted 3-coupling faults' have been proposed. Both algorithms are $O(N \cdot$

Table 3: Pseudoexhaustive memory testing

March element	MATS+ for Different Backgrounds					
	11111	10000	01000	00100	00010	00001
$\Downarrow [r(a_i), w(a_i \oplus 1)]$	11110	10001	01001	00101	00011	00000
	11100	10011	01011	00111	00001	00010
	11000	10111	01111	00011	00101	00110
	10000	11111	00111	01011	01101	01110
	00000	01111	10111	11011	11101	11110
$\Uparrow [r(a_i), w(a_i \oplus 1)]$	10000	11111	00111	01011	01101	01110
	11000	10111	01111	00011	00101	00110
	11100	10011	01011	00111	00001	00010
	11110	10001	01001	00101	00011	00000
	11111	10000	01000	00100	00010	00001

Table 5: The complexity $T[PXT(n,3)]$ in seconds (*sec.*); memory block size in bits

Size n of block	2^{10}	2^{12}	2^{14}	2^{16}
$T[PXT(n,3)]$	0.05	0.24	1.16	5.3
Size n of block	2^{18}	2^{20}	2^{22}	2^{24}
$T[PXT(n,3)]$	23.8	105.9	465.5	2030.0

$\log_2 N)$ tests. Memory test algorithms for more complex k-coupling faults are not known [5]. For $n = N$ and $k = 3$ our approach has the same complexity (see (5)) as two known algorithms [7, 8] and detects all 3-coupling faults (not restricted). As we can see from Table 5 the required time to perform the PXT for 2^{20} bit RAM and cycle time $100ns$ is $105.9sec$.

Pseudoexhaustive memory testing approach, for $n = N$ allows to detect all k-coupling faults for any k. The complexity $T[PXT(N,k)]$ is $O(N \cdot \log_2 N)$ since

$$2^{k-1} \log_2 N \leq f(N,k) \leq k \cdot 2^k (\log_2 e)^{-1} \log_2 N, \quad (6)$$

for k fixed and N large enough [14, 15].

Thus PXT approach provides for detection of all k coupling faults, as well as all memory faults covered by PS(n,k) and its complexity is only $O(N \cdot \log_2 N)$ (see Table 5). But at the same time for $n = N$ the length of pseudoexhaustive test patterns equals N, which allows to use this approach for external testing only. We need to implement an extra memory or hardware unit E for generation of $f(N,k)$ test patterns with length N and additional memory S for initial memory content.

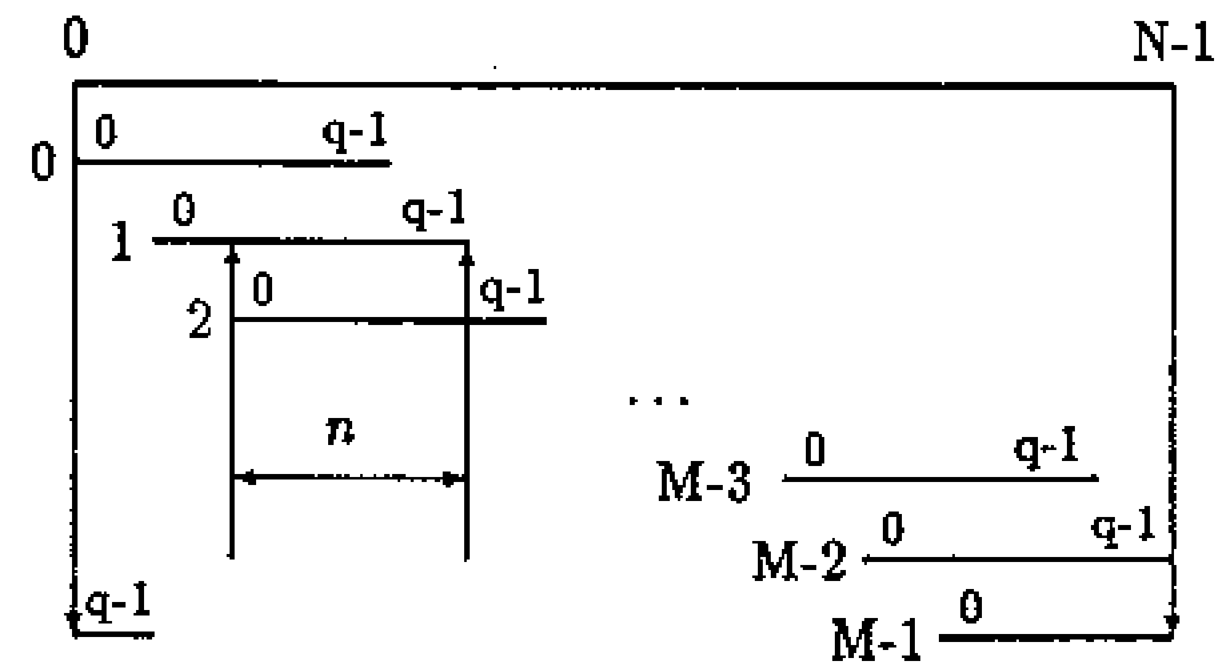

Figure 1: The timing diagram of the self-test procedure

In the following sections we describe a modification of this approach for $n \ll N$ which results in a drastic reduction of the pseudoexhaustive test patterns length and the test complexity.

4 Test Session Organization

To simplify a test generation procedure for PXT we partition the original memory into $M = M(N,q,n)$ blocks with size $q > n$. In this case every test session consists of $M(N,q,n)$ elementary PXT sessions where q represents the size of the memory block and n the number of overlapped memory bits in two consecutive elementary test sessions (see Fig. 1) to ensure pseudoexhaustive testing within any n bit block of the memory array.

For the more general case of a 2-dim memory a number of elementary sessions is determined as

$$M(N,q,n_x,n_y) = \lceil \frac{N}{q_x q_y} \rceil \approx \frac{N}{q_x q_y}, \quad (7)$$

where $q = (q_x + n_x)(q_y + n_y)$ and $n = n_x n_y$.

Then, we have for complexity $T = T[PXT(N, q, k)]$ of the modified PXT

$$T[PXT(N,q,k)] = M(N,q,n_x,n_y)\times$$
$$\times T[PXT((q_x+n_x)(q_y+n_y),k)] =$$
$$= \frac{N}{q_x q_y} 5(q_x+n_x)(q_y+n_y)\times$$
$$\times f[(q_x+n_x)(q_y+n_y),k] + \ (q_x+n_x)(q_y+n_y). \quad (8)$$

The overhead for storage of $f[(q_x+n_x)(q_y+n_y),k]$ pseudoexhaustive patterns of length $q = (q_x+n_x)(q_y+n_y)$ and initial block contents can be estimated as

$$L = (q_x+n_x)(q_y+n_y)\,[f((q_x+n_x)(q_y+n_y),k)+1]. \quad (9)$$

Equations (8) and (9) illustrate the tradeoff between a testing time and an overhead for detection of PS(n,k). For a given $n = n_x n_y$ an increase in the block size $q = (q_x+n_x)(q_y+n_y)$ results in a decrease of test time T (determined by (8)) and in an increase in overhead L (determined by (9)). It is easy to show that to minimize TL optimal values for n_x and n_y are q_x and q_y.

In the case of linear 1-dim arrays we have $q = q_x + n_x = 2n$, $n_x = n$ and $M(N, 2n, n) = N/n$. Then

$$T[PXT(N,q,k)] = \frac{N}{n}\cdot T[PXT(2n,k)] =$$
$$= (10\cdot f(2n,k)+2)\cdot N. \quad (10)$$

For the case of a $\sqrt{N} \times \sqrt{N}$ 1-bit RAM with crosstalks within rows or columns only one can take as blocks rows and columns. Since $q = \sqrt{N}, n \le 1$ and $M(N, \sqrt{N}, n) = 2\sqrt{N}$ we have $L = \sqrt{N} f(n,k) + \sqrt{N}$ and

$$T[PXT(N,\sqrt{N},k)] = 2\sqrt{N}\ T[PXT(\sqrt{N},k)] =$$
$$= 10N\cdot f(\sqrt{N},k) + 2N. \quad (11)$$

For example, if $k = 3$, then $f(\sqrt{N},3) = 10\log_2\sqrt{N} = 5\log_2 N$ and

$$L = 5\sqrt{N}\log_2 N + \sqrt{N},$$

$$T[PXT(N,\sqrt{N},3)] = 50N\log_2 N + 2N.$$

For $N = 2^{22}$ and access time $t = 100ns$ we have $L = 227,328$ bits and $T = 462,2sec$.

To detect all PS(n,k)s within any 2-dim block with the size $n = n_x n_y$ we have $q_x = q_y = n_x = n_y = \sqrt{n}$, $q = 4n$, and

$$M(N,4n,\sqrt{n},\sqrt{n}) = \frac{N}{n}.$$

Test complexity $T[PXT(N,q,k)]$ for 2-dim memory is

$$T[PXT(N,q,k)] = \frac{N}{n}\cdot T[PXT(4n,k)] =$$
$$= (20\cdot f(4n,k)+4)\cdot N. \quad (12)$$

For example, for $n = 16$, $k = 3$ for detection of PS(16,3) faults, since $f(64,3) = 37$, we have by (12) $T[PXT(N,16,3)] = (20\cdot f(64,3)+4)\cdot N = 744\cdot N$.

The required time to perform the PXT for a $N = 2^{22}$ bit RAM and cycle time $t = 100ns$ in this case is $312.0sec$.

The hardware overhead for the above mentioned example of BISTed $4Mb$ RAM consists of an extra (37×64) memory E for storing $f(64,3) = 37$ by 64 bit pseudoexhaustive patterns and 64-bit memory S, which are both required 0.059% of the original array.

5 Transparent Circular Testing

Sufficient reduction of the overheads can be achieved for a transparent test with a short length of a test cycle, which does not depend on the initial memory contents. The required fault coverage can be obtained for the test which allows the generation of as many as possible different patterns to approximate pseudoexhaustive memory test.

The method consists of the simulation of Linear Feedback Shift Register (LFSR) on a block of memory-under-test and it has a simple hardware implementation.

Let us assume that a memory-under-test contains N bits. We partition the original memory into $M = M(N,q,n)$ blocks with size $q > n$ where n represents the number of overlapped memory bits in two consecutive elementary test sessions. For every block we will organize q ($q \ll N$) neighboring cells in the testing mode as a LFSR.

The test session for our approach consists of $M \le N$ subsessions (see section 5). The implementation of a subsession requires the following three main stages.

1.Compute the signature of the memory block-under-test initial contents.

2.The memory block-under-test with the size q operates as the selected LFSR.

3.Compute the signature of the new contents of the memory block after stage 2 and compare it with the signature computed at stage 1.

The faults manifesting themselves during intermediate read and write operations would be detectable by the observation of a distortions the final memory contents. In this case we only need to verify the resulting memory contents.

For the RAM with built-in error-detection/correction circuit, stages 1 and 3 can be avoided. In

this case, a nonzero output (syndrome) of the error-detection/correction circuitry indicates the presence of a fault.

6 Pseudorandom Circular Test Patterns

A major advantage of the pseudorandom testing as an approximation of pseudoexhaustive memory testing is simplicity of its implementation, hardware implementation included. LFSRs have been widely used for generation of pseudorandom (PSR) test patterns and for compression of test responses [16].

We will describe in this section a procedure PSR for transparent testing of RAMs based on simulation of p shifts of two LFSRs with generating polynomials $\varphi(x) = 1 \oplus x^1 \oplus x^q$ and $\psi(x) = 1 \oplus x^{q-1} \oplus x^q$ on a block of q cells. Polynomials $\varphi(x)$ and $\psi(x)$ may be replaced by any pair of reciprocal polynomials. Selecting $\varphi(x) = 1 \oplus x^1 \oplus x^q$ results in only one two-input XOR gate required for simulation, minimizes address switching and allows usage of a standard counter for address generation.

When we use the polynomial and its reciprocal version sequentially in one algorithm we return back to the initial state at the end of a test session.

The pseudorandom transparent memory testing procedure consists of the following main steps.

1.Compute the signature of the memory contents $A = (a_0a_1a_2 \dots a_{q-1})$ $(A \leftarrow S)$.

2.Simulation of p shifts for the LFSR generated by $\varphi(x) = 1 \oplus x^1 \oplus x^q$ which generates the following sequence of internal states of the block $A, \alpha A, \alpha^2 A, \dots, \alpha^p A$, where $1 \oplus \alpha \oplus \alpha^q = 0$.

3.Simulation of p shifts of the LFSR generated by $\psi(x) = 1 \oplus x^{q-1} \oplus x^q$ which generates a sequence $\alpha^p A, \alpha^{p-1} A, \dots, \alpha A, A$.

4.Negate data in the block $(\overline{A} \leftarrow A)$.

5.Repeat steps 2 and 3.

6.Negate data in the block $(\overline{A} \leftarrow A)$.

7.Compute the signature S^* of resulting memory contents and compare it with the signature S of initial contents $(S = S^*?)$.

The following simple procedure PSR can be used to simulate the LFSR, described by polynomial $\varphi(x) = 1 \oplus x^1 \oplus x^q$ over q-bit memory block. Here D_0 and D_1 are the first and a second stage of an additional shift register SR; $n = 2^k$ is the memory-under-test block; $[a_i]$ – the contents of the ith memory cell.

Procedure PSR

Input:	Memory block size q;
	Starting address w;
	Number of shifts p;
Read:	$[a_{w+q-1}]$ to D_0
Shift:	SR;
Loop1:	For $j = 1$ to p Do
Read:	$[a_w]$ to D_0;
Write:	$[D_1] \oplus [D_0]$ to a_w;
Shift:	SR;
Loop2:	For $i = 1$ to $q-1$ Do
Read:	$[a_{w+i}]$ to D_0;
Write:	$[D_1]$ to a_{w+i};
Shift:	SR;
End:	Loop2;
Read:	$[a_{w+q-1}]$ to D_0
Shift:	SR;
End:	Loop1;

The *Write* and *Shift* operations in Loop1 and Loop2 can accomplished simultaneously and the complexity $T = T[PSR]$ of memory implemented LFSR simulation is determined as

$$T[PSR] = 2(qp + p + 1). \qquad (13)$$

The total complexity $T = T[PSR(q,p)]$ of pseudorandom transparent testing which includes all stages of LFSR simulation and signatures calculation for memory block with the size q has the form

$$T[PSR(q,p)] = 8qp + 4(q + 2p) + 8. \qquad (14)$$

For example for a $\sqrt{N} \times \sqrt{N}$ 1-bit RAM with crosstalks limited to rows and columns only we can take for the BIST realization of pseudorandom test $q = \sqrt{N}$, blocks are rows and columns. Then, taking into account that $M = 2\sqrt{N}$ we have for the test complexity

$$T[PSR(N,p)] = 2\sqrt{N}\, T[PSR(\sqrt{N},p)] =$$
$$= 16Np + 8N + 16\sqrt{N}p + 16\sqrt{N}, \qquad (15)$$

For $N = 2^{22}$ and $p = 32$ and cycle time $t = 100ns$ we have $T = 218.2sec$.

The number of different combinations $f(k)$ which are generated at any k bits within a memory block does not depend on the initial state of the cells. We can get a good approximation of the PXT algorithm by increasing the value of q which provides for any $f(k), k \leq q$ by increasing a test complexity for any random initial contents.

The experimental results for average values of $P(k) = f(k)/2^k$ for $q = 8$ and $q = 32$ for different

Table 6: Probability $P(k)$ for pseudorandom tests based on simulation of LFSR on $q = 8$-bit block-under-test

p	$k=2$	$k=3$	$k=4$	$k=6$	$k=8$
1	1.00	0.841	0.607	0.248	0.086
2	1.00	0.917	0.717	0.324	0.115
4	1.00	0.972	0.849	0.444	0.169
8	1.00	0.997	0.949	0.623	0.265
16	1.00	1.000	0.998	0.821	0.441
32	1.00	1.000	1.000	0.947	0.651

Table 7: Probability $P(k)$ for pseudorandom tests based on simulation of LFSR on $q = 32$-bit block-under-test

p	$k=2$	$k=4$	$k=6$	$k=8$	$k=10$
1	1.00	0.611	0.247	0.086	0.027
2	1.00	0.718	0.321	0.115	0.036
4	1.00	0.857	0.447	0.170	0.055
8	1.00	0.959	0.634	0.269	0.092
16	1.00	0.991	0.831	0.433	0.197
32	1.00	1.000	0.953	0.581	0.325

p are shown in Table 6 and Table 7. As we can see from Table 6 and Table 7, $P(k)$ is very close to 1 for $k \leq \log_2 p$. We note that $P(k)$ does not depend on selection of k positions within a block of q cells.

Thus, the pseudorandom circular test has a good fault coverage for pattern sensitive faults for a small size k, of neighborhood and a nonzero but decreasing coverage for growing k

Faults manifesting themselves during intermediate Read and Write operations are detected by observation of distortions in final states of the block-under-test.

Probabilities of missing P_{al} a fault for CF_{id} $\langle\uparrow,0\rangle$, $\langle\uparrow,1\rangle$, $\langle\downarrow,0\rangle$ and $\langle\downarrow,1\rangle$ are shown in Table 8 and for CF_{in} $\langle\uparrow,\updownarrow\rangle$ and $\langle\downarrow,\updownarrow\rangle$ in Table 9. As we can see all CFs are detectable with the probability very close to 1 for $p \geq 8$. It follows from Table 6 Table 7, Table 8 and Table 9, that by selecting $q = 32$ and $p = 16$ we have all 2^k test vectors for any $k \leq 4$ bits within the block-under-test and in addition to this the fault coverage for coupling faults is very close to 100%. For a bit-oriented 2^{22}-bit RAM with $q = 2n = 32$, $p = 16$ and $t = 100nsec.$ the required testing time is 114.2sec. (see formula (16) below) and 225.0sec. for 2-dim. (see formula (17)).

Table 8: Aliasing probability P_{al} for CF_{id} $\langle\uparrow,0\rangle$, $\langle\uparrow,1\rangle$, $\langle\downarrow,0\rangle$ and $\langle\downarrow,1\rangle$ for pseudorandom tests based on simulation of LFSR on a q-bit block-under-test

q	$p=1$	$p=2$	$p=4$	$p=6$	$p=8$
4	0.604	0.375	0.125	0.067	0.029
8	0.556	0.330	0.126	0.051	0.020
16	0.530	0.294	0.097	0.038	0.016
32	0.515	0.273	0.079	0.028	0.009

Table 9: Aliasing probability P_{al} for CF_{in} $\langle\uparrow,\updownarrow\rangle$ and $\langle\downarrow,\updownarrow\rangle$ for pseudorandom tests based on simulation of LFSR on a q-bit block-under-test

q	$p=1$	$p=2$	$p=3$	$p=4$	$p=5$
4	0.416	0.166	0.052	0.000	0.000
8	0.445	0.195	0.081	0.033	0.013
16	0.471	0.223	0.103	0.047	0.020
32	0.484	0.236	0.114	0.053	0.027

7 Hardware Implementation

Hardware implementation of the pseudorandom circular test pattern bit(word)-oriented memory testing requires two($2m$) flip-flops, one(m) multiplexers and one(m) XOR gates for the test pattern generator, one(m) multiplexers for mode control and one signature analyzer per memory unit. For a small q and $m = 1$ the signature analyzer is an q-bit shift register and for the general case it consists of an $r \leq 32$ bit signature analyzer, a r-bit register and a comparator. In many cases one can take $r = m$.

The complexity of a hardware implementation of the test controller depends on the size N of memory, as well as on the number of cells q within the memory block. For general case the complexity of test controller can be estimated as $O(\log_2 N) + O(\log_2 q)$. Details of these hardware implementations are described in following subsections.

7.1 Bit-Oriented RAM

The block diagram for the hardware implementation of the procedure PSR for bit-oriented RAM with transparent testing is shown in Fig 2.

The RAM may operate in the normal computing mode and in the transparent self-testing mode.

Transparent periodic testing of the RAM of Fig 2 is divided into $M(N, q, n)$ subsessions (see section 5), where $q \ll N$ is the size of a RAMs block used for one subsession. Every subsesssion consists of the main

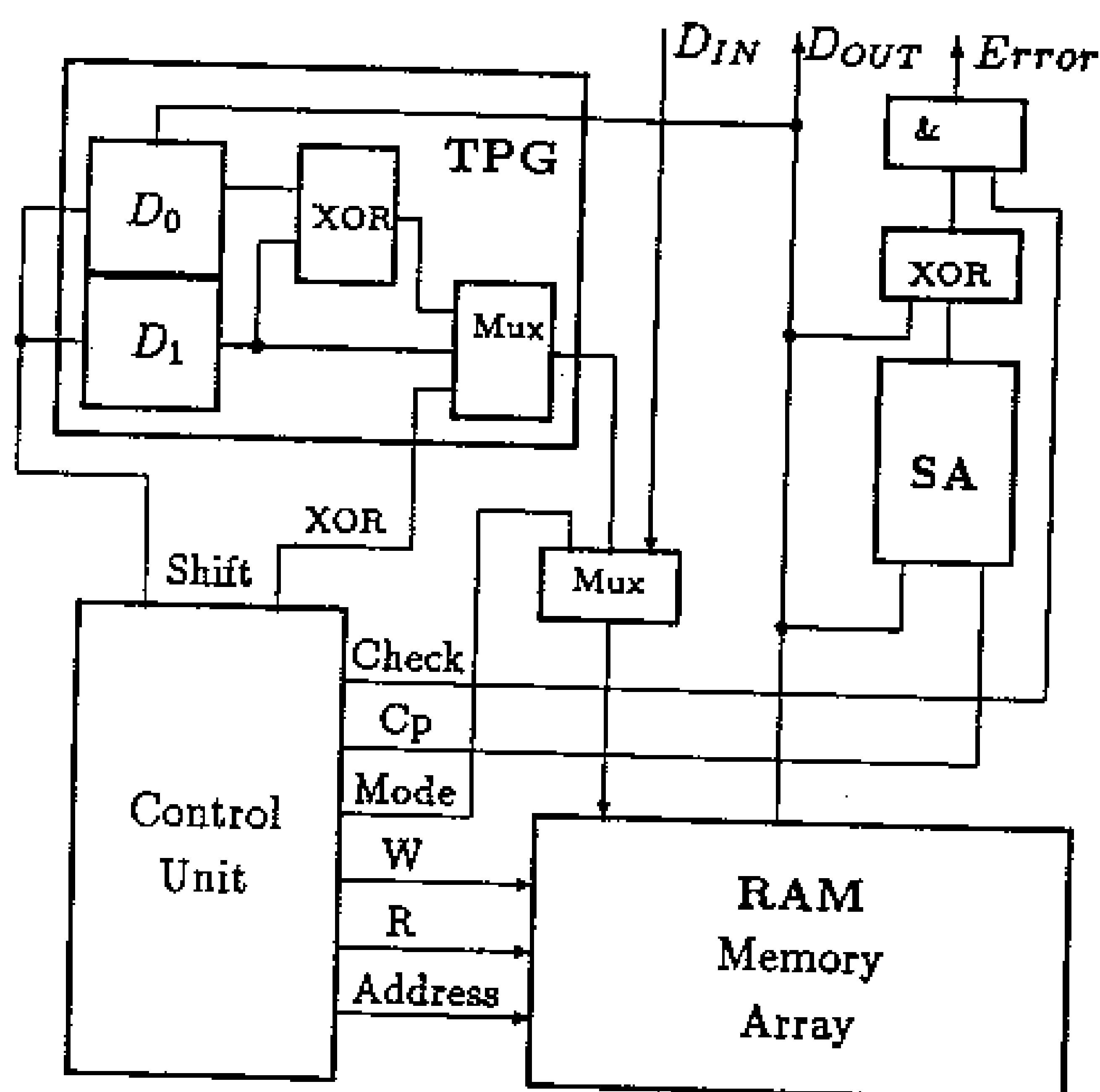

Figure 2: Block diagram of the transparent self-testing RAM

following steps.

1. Compute the signature for a initial state of the block of q memories cells. For small q (say $q < 32$) cells contents are written into a shift register (SA) (see Fig. 2) and for larger q SA is replaced by a primitive r-bit $LFSR$. This stage requires q memory cycles.

2. The block of q memory cells operate as LFSR described by primitive polynomial or reciprocal polynomial.

3. Compute the signature of a new state of the block and compare it with computed at step 1. For the RAM shown in Fig. 2 this step requires comparison of the contents of the memory block with the contents of SA (see Fig. 2). This step requires q cycles.

For the complexity $T[PSR(N, q, n)]$ of a one test session we have in a view of for 1-dim. memory

$$T[PSR(N, 2n, n)] = M(N, 2n, n)T[PSR(2n, p)] =$$

$$= (16p + \frac{8p}{n} + \frac{8}{n} + 8)N, \qquad (16)$$

and for 2-dim. memory

$$T[PSR(N, 4n, n)] = M(N, 4n, n)T[PSR(4n, p)] =$$

$$= (32p + \frac{16p}{n} + \frac{8}{n} + 16)N, \qquad (17)$$

where $T[PSR(2n, p)]$ and $T[PSR(4n, p)]$ are the complexities of PSR.

A hardware implementation of a transparent self-testing memory for a more common structure of a W words by m bits RAM consists of m stages of a test pattern generator TPG, an m-bit signature analyzer ($MISR$), an m bit register (RG), and a comparator. In this case q words with m cells in a word are tested simultaneously in time. As in the previous case the self-testing procedure requires $M(N, q, n)$ subsessions and the test complexity is determined by (16) and (17). The only difference is that during steps 1 and 2 the compaction of the contents of q memory words is implemented by a $MISR$.

7.2 Word-Oriented RAM with On-line Testing

An important feature of our approach is the ability to combine off-line and on-line testing. For PSR procedure we can use on-line circuitry for response evaluation and fault manifestation. In this case we do not need to calculate and compare the signatures.

For the case of parity check codes and even m our approach has a simple implementation shown in Fig. 3. During the off-line test mode the presented design implements the transformation of the memory word contents for fault-free case within the parity code codewords.

For a faulty RAM, a memory word will be changed to a noncode word. A manifestation of any fault through the cyclic test pattern application can be detected in the *Read* mode by the on-line parity check circuit.

Thus, if the RAM has the on-line test circuits, it can be used for the off-line transparent self-testing procedure and allow us to reduce hardware and time overheads for BIST. For the case shown in Fig 3 there is one extra column for the parity check code and an additional circuit for on-line testing. To implement the transparent off-line circular test pattern for the above mentioned RAM (Fig. 3), we have to generate the circular test patterns generator for all columns of the RAM.

Comparing with the case of a word-oriented RAM, we note that for the RAM with the on-line testing capability we do not need an m-bit $MISR$, m-bit register and comparator, which simplifies the hardware implementation.

8 Conclusions

In this paper we have presented unified approach for memory testing based on PS(n,k) fault model and pseudoexhaustive transparent memory testing and its approximations by pseudorandom circular tests. It was shown that proposed test procedures have high fault coverage for simple faults, as well as for pattern sensitive faults covered by PS(n,k) faults.

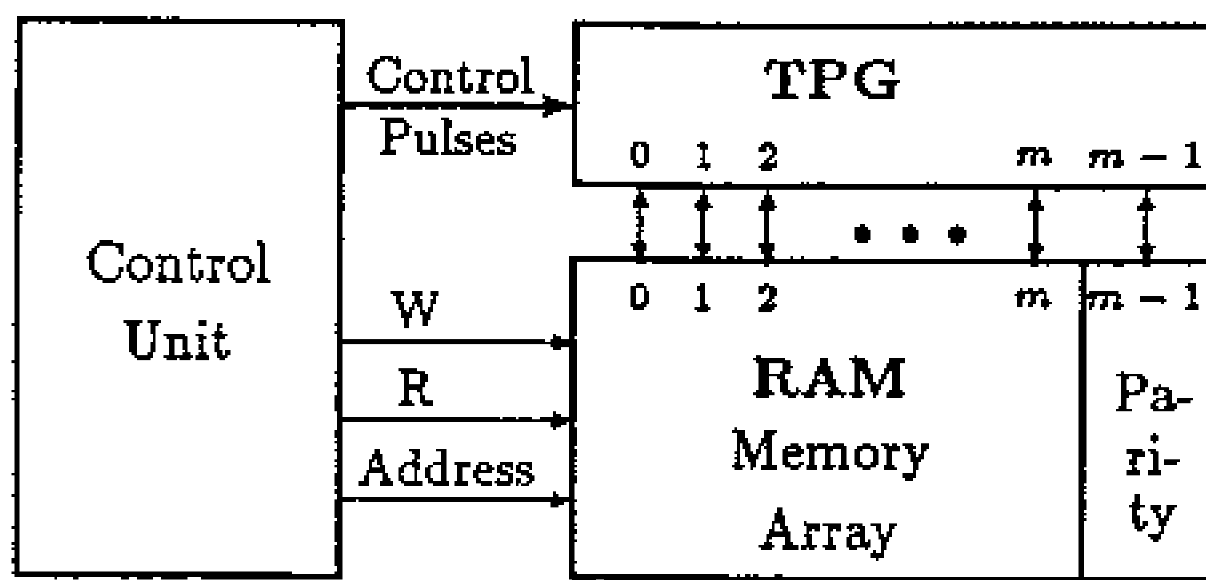

Figure 3: Block diagram of a transparent self-testing for RAM with on-line and off-line testing circuits

The results of the paper shows us the possibilities of pseudoexhaustive memory testing for relatively small n and k and transparent memory testing by the circular pseudorandom test patterns. It was shown that the BIST implementations of the circular memory testing have some attractive features. The most important among them are combined utilization of on-line and off-line circuitry, low hardware and time overhead and not destructive off-line testing.

Acknowledgment

The authors wish to thank Prof. Lev B. levitin of the Boston University, Boston, for his valuable suggestions.

References

[1] Nicolaidis M. "Transparent BIST for RAMs", *Proceedings International Test Conference*, Baltimore, September 1992, pp. 598-607.

[2] Goor A.J. "Using March Tests to Test SRAMs", *IEEE Design and Test of Computer*, Vol.10, N1, March 1993, pp.8-14.

[3] Krasniewski A., Krzyztof G. "Is There Any Future for Deterministic Self-Test of Embedded RAMs?", *Proceedings European Test Conference*, Rotterdam, April 1993, pp.159-168.

[4] Yarmolik V.N., Nicolaidis M. "Exact Aliasing Computation And/Or Aliasing free design for RAM BIST", *Proceedings of Int. Workshop on Memory Testing*, San Jose, August 1993.

[5] Goor A.J. "Testing Semiconductor Memories, Theory and Practice", *John Willey and Sons*, Chichester, 1991.

[6] Marinesku M. "Simple and efficient algorithm for functional RAM testing", *Proceedings International Test Conference*, 1982, pp.236-239.

[7] Nair C., Thatte S.M., Abraham J.A. "Efficient Algorithms for Testing Semiconductor Random-Access Memories", *Proceedings International Test Conference*, Vol.C-27, June 1978, pp.572-576.

[8] Papachristou C.A., Saghal N.B. "An Improved Method for Detecting Functional Faults in Random-Access Memories", *IEEE Transactions on Computers*, Vol.C-34, N2, 1985, pp.110-116.

[9] Suk D.S., Reddy S.M. "Test Procedure for a class of pattern sensitive faults in semiconductor random access memories", *IEEE Transactions on Computers*, Vol.C-29, N6, 1980, pp.419-429.

[10] Treuer R., Agarwal V.K. "Buit-In Self-Diagnosis for Repairable Embedded RAMs", *IEEE Design and Test of Computer*, Vol.10, N2, June 1993, pp.24-33.

[11] Jain S.K., Stroud S.H. "Built-In Self-Testing of Embedded Memories", *IEEE Design and Test of Computer*,Vol3, N5, October 1986, pp.27-37.

[12] Le K.T., Saluja K.K. "A Novel Approach for Testing Memories Using a Built-in Self Testing Technique", *Proceedings International Test Conference*, Washington, September 1986, pp.830-838.

[13] Barzilai Z., Coppersmith D., Rosenberg A. "Exhaustive Generation of Bit Pattern with with Application to VLSI Self-Testing" , *IEEE Transactions on Computers*, Vol.C-31, N2, 1983, pp.190-194.

[14] Levitin L.B., Karpovsky M.G. "Efficient Exhaustive Test Based on MDS Codes" , *Proceedings IEEE International Symposium on Information Theory* , Ann Arbor, 1986.

[15] Cohen G., Karpovsky M., Levitin L. "Exhaustive Testing of Circuits with Outputs Depending on Limited Number of Inputs" , *IEEE International Information Workshop* , Caesarea, 1984.

[16] Bardell P.H., McAnney W.H., Savir J. "Built-in Test for VLSI: *Pseudorandom Techniques*", John Wiley and Sons, New York, 1987.