BIST for Word-Oriented DRAM ♣

L. Zakrevski, M. Karpovsky, S. H. Yang♣♣

Research Lab. on Reliable Computing,
Boston University, Department of Computer Engineering,
8 Saint Mary's St., Boston, MA 02215, USA
zakr@bu.edu

Abstract

The problem of exhaustive test generation for detection of coupling faults between cells in word-oriented memories is considered. According to this fault model, contents of any w-bit memory word in a memory with n words, or ability to change this contents, is influenced by the contents of any other s-1 words in the memory. A near optimal iterative method for construction of test patterns is proposed. The systematic structure of the proposed test results in simple BIST implementations.

***Key words**: Memory testing, word-oriented memory, pattern sensitive faults, pseudo-exhaustive memory testing.*

1. Introduction

In this paper the problem of test generation for word-oriented memories is considered. Most of the existing test generation methods are developed for bit-oriented memories. In the case when each cell can be in more than 2 states, the number of faults increases, so new methods for test generation are needed. There are several approaches to the problem of testing of word-oriented memories [4,9,13]. One of the most attractive solutions for the test cost reduction problem is based on built-in self-test (BIST) [15].

We assume that there are n memory cells, each of them can be in q different states, and s-coupling fault model is used. A set of s cells is said to be s-coupled when a *Write* operation in one cell produces a change in the contents of another cell, subject to a particular data pattern in the remaining s-2 cells, which may be anywhere in the memory [1]. This model includes most classical fault models for word-oriented memories. Typically $q=2^w$, (w is the number of bits in the memory word) in practice cases w=2, w=4 and w=16 are most common.

This fault model applies especially to DRAMs, where in addition to the traditional faults for SRAM chips [7,8], neighborhood pattern sensitive faults ('NPSFs') [11] have to be considered. Several tests for NPSFs detection were proposed in [11]. Their main restriction is that the physical topology of the cells in the memory cell array has to be known. In practice, it is usually not the case. Also, most of the existing methods are oriented for fault detection in bit-oriented memories.

The word-oriented memory test algorithms can be constructed, using the backgrounds, by replicating the single-bit memory test algorithm ($\lceil \log_2 w \rceil$+1) times [12]. This approach (with some modifications [13]) allows to detect the 2-coupling faults only and does not guarantee the detection of the k-couplings as well as pattern sensitive faults [4].

Pseudo-random memory tests [1,3,10,14] do not require knowledge of the physical topology of the memory cell array and can be applied to word-oriented memories; however, they have the disadvantage that their fault coverage is probabilistic.

In this paper a unified approach for word-oriented memory BIST is proposed. This approach is based on the results for bit-oriented memories (q=2), which were presented in [1,4]. In [4], some of these results were generalized for word-oriented DRAMs with small numbers of cells.

This paper is organized in a following way:

In Section 2 the general mathematical model of s-surjective matrices for test generation is proposed. Matrix $\boldsymbol{A}_q(n,s)$ with elements from {0,1,…,q-1} is called s-surjective ((n,s) exhaustive), if in each s columns all q^s q-ary s-tuples can be found as rows.

♣ This work was supported by the NSF under Grant MIP 9630096
♣♣ Dr. Yang is currently with Dept of Electrical Engineering, Kwangwoon Univ., Seoul, Korea

In Section 3 we discuss the problem of building s-surjective q-ary matrices $\boldsymbol{A}_q(n,s)$ with minimal numbers of rows. A general method for construction of near-optimal s-surjective matrices is developed. It is based on an iterative procedure and small s-surjective matrices, called seeds. The problem of construction of optimal seeds is solved, upperbounds on minimal numbers of rows in s-surjective matrices and estimations on test times are presented. Also, special cases q=4 and q=16 are analyzed (they are important from the practical point of view).

In Section 4 the hardware realization of the memory BIST based on the proposed approach is considered and required overheads are estimated.

2. Mathematical model

Definition 1. Matrix $\boldsymbol{A}_q(n,s)$ with n columns and elements from {0,...,q-1} is called s-surjective ((n,s)-exhaustive [1,2]), if in each s-tuple of columns all q^s q-ary possible s-tuples can be found as rows.

To detect and locate all possible unlinked s-coupling faults in n cells, it is necessary (but may not be sufficient) to generate all possible q^s combinations for every s cells, thus q-ary test matrix $\boldsymbol{A}_q(n,s)$ with n columns must be s-surjective [1]. (Rows of $\boldsymbol{A}_q(n,s)$ are test patterns.)

To provide for test generation for transition faults, the concept of strong surjectivity can be used.

Definition 2. Matrix $\boldsymbol{A}_q(n,s)$ with elements $a(i,j)$ is called strongly s-surjective, if $\boldsymbol{A}_q(n,s)$ is s-surjective and for any $x_1,x_2,\ldots,x_s \in \{0,\ldots,q\text{-}1\}$ and any s-1 tuple of columns $\{j_1,\ldots,j_{s-1}\}$ there exists a row i such that $a(i,j_1) = x_1,\ldots, a(i,j_{s-1}) = x_{s-1}$ and $a(i+1,j_1) = x_s$.

At the testing stage a MARCH test is used - each row is considered as a test background and is loaded to the memory using the MARCH procedure [1,2,4,7]. After loading a background, the state of all memory cells is checked.

To generate a minimal test, we need to construct a strongly s-surjective q-ary matrix with n columns and minimal number of rows. This number is represented by function $f_q(n,s)$.

Since each row (background) of a test matrix corresponds to n *Read* and n *Write* operations, the resulting test length (number of *Read* and *Write* instructions) can be estimated as $2n\, f_q(n,s)$.

An obvious low bound on $f_q(n,s)$ is given by the following statement:

Statement 1. If $n \geq s$, then $f_q(n,s) \geq q^s$.

The proposed methods can be used for both transparent and non-transparent memory testing. For transparent testing the state of all memory cells must be restored after the test (at least, at the absence of faults). It means, that the i-th background is loaded as the modulo-two sum of the present background (or initial memory state) and the i-th row of the test matrix $\boldsymbol{A}_q(n,s)$. To provide the restoring of the initial memory state in the absence of faults, the number of ones in each column of test matrix should be even.

We will describe below an iterative method for construction of strongly s-surjective matrices, which aims to minimize the number $f_q(n,s)$ of rows in these matrices for given n,q,s. This method is based on the method for construction of s-surjective matrices for the binary case suggested in [1].

We note that even for the binary case (q=2) the problem of construction of s-surjective (but not necessary strongly s-surjective) matrices with minimal numbers of rows is still open. Some results in this direction can be found in [1,5].

3. Construction of non-binary strongly s-surjective matrices

3.1 Iterative construction of surjective matrices

If we have a strongly s-surjective matrix $\boldsymbol{A}_q(b,s)$ with $f_q(b,s)$ rows and b columns, we can expand it using a special matrix $\boldsymbol{M}_b(n,s)$ with n columns. Elements of $\boldsymbol{M}_b(n,s)$ should be integers from {0,..,b-1}. The following condition must be satisfied: for each s columns there exist a row in $\boldsymbol{M}_b(n,s)$ with all different elements in the selected columns (we assume b ³ s). An example of matrix $\boldsymbol{M}_5(10,4)$ for s=4, n=10 and b=5 is given below.

$$\boldsymbol{M}_5(10,4) = \begin{bmatrix} 0\,0\,1\,1\,2\,2\,3\,3\,4\,4 \\ 0\,1\,1\,2\,2\,3\,3\,4\,4\,0 \\ 0\,1\,2\,3\,4\,0\,1\,2\,3\,4 \\ 0\,1\,2\,3\,4\,1\,2\,3\,4\,0 \\ 0\,1\,2\,3\,4\,3\,4\,5\,0\,1 \end{bmatrix} .$$

We can construct a strongly s-surjective matrix $\boldsymbol{A}_q(n,s)$ with n columns from matrix $\boldsymbol{M}_b(n,s)$ in the following way: instead of every element i in $\boldsymbol{M}_b(n,s)$ we substitute the i-th column of $\boldsymbol{A}_q(b,s)$. The resulting matrix will be strongly s-surjective and have $f_q(b,s)$ $m_b(n,s)$ columns, where $m_b(n,s)$ is the number of rows in $\boldsymbol{M}_b(n,s)$.

Thus we have the following result

<u>**Statement 2**</u>. For any $s \leq b$ $f_q(n,s) \leq f_q(b,s)\, m_b(n,s)$.

We note also that by deleting columns in strongly s-surjective matrix $\boldsymbol{A}_q(b,s)$ we can obtain seeds with numbers of columns less than b.

Let us suppose that elements of raw i of matrix $\boldsymbol{M}_b(n,s)$ are upperbounded by t_i. Then we can use for substitution of elements in the i-th row of $\boldsymbol{M}_b(n,s)$ seed matrices with t_i columns.

Statement 3. If matrix $\boldsymbol{M}_b(n,s)=(m(i,j))$ and $\max_j m(i,j)=t_i$, then $f_q(n,s)\le\sum_i f_q(t_i,s)$

In view of Statement 3, the procedure for construction of strongly s-surjective matrices can be divided into two parts - construction of good seeds and construction of matrices $\boldsymbol{M}_b(n,s)$ with small numbers $m_b(n,s)$ of rows.

3.2 Construction of seed matrices

It has been shown in [4] that we can construct optimal strongly s-surjective matrices $\boldsymbol{A}_q(q+1,s)$ with q^s+1 rows. (For s=3 and $q=2^w$ s-surjective matrices $\boldsymbol{A}_q(q+2,s)$ with q^s+1 rows can be constructed.) These matrices are based on extended Reed-Solomon codes over GF(q) [6]. The extended (q+1,q+1-s,s+1) Reed-Solomon code over GF(q) is defined by the check matrix:

$$\boldsymbol{H}=\begin{bmatrix} 1 & 0 & 1 & 1 & 1 & \dots & 1 \\ 0 & 0 & 1 & \alpha & \alpha^2 & \dots & \alpha^{q-2} \\ 0 & 0 & 1 & \alpha^2 & \alpha^4 & \dots & \alpha^{2(q-2)} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 1 & 1 & \alpha^{s-1} & \alpha^{2(s-1)} & & \alpha^{(q-2)(s-1)} \end{bmatrix}$$

Here α is a primitive element of GF(q) ($\alpha^i\neq\alpha^j$ for $i\neq j\in\{0,1,\dots,q-2\}$) [6].

It was shown in [4] that if all q^s linear combinations of rows of $\boldsymbol{H}$ ordered in a special way are selected as a rows of matrix $\boldsymbol{A}$ then $\boldsymbol{A}=\boldsymbol{A}_q(q+1,s)$ is a strongly s-surjective matrix with q^s+1 rows. (The second row is repeated at the end.) The main idea consists in using another primitive polynomial over GF(q^s) to order the rows of $\boldsymbol{A}_q(q+1,s)$.

For example, let us consider $q=2^w=4$, s=2 and construct strongly 2-surjective matrix $\boldsymbol{A}_4(5,2)$. Then GF(2^2)=$\{0,1,\alpha,\alpha^2\}$, where α is a root of polynomial $\varphi(x)=x^2+x+1$ ($\alpha^3=1$). The operations of addition and multiplication in the field GF(2^2) are described by the following tables:

	Addition				Multiplication			
	0	1	α	α^2	0	1	α	α^2
0	0	1	α	α^2	0	0	0	0
1	1	0	α^2	α	0	1	α	α^2
α	0	α^2	0	1	0	α	α^2	1
α^2	α^2	α	1	0	0	α^2	1	α

where 0 = 00, 1 = 10, α= 01 and α^2= 11.

For the construction of the optimal 2-pseudoexhaustive backgrounds over GF(2^2) we use the following check matrix:

$$\mathbf{H}=\begin{bmatrix} 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & a & a^2 \end{bmatrix}$$

To order the resulting backgrounds, primitive polynomial in GF(4^2) should be considered, such as $\lambda(x)=x^2+x+\alpha$. Let $\lambda(b)=0$, $b\in$GF(4^2). If $b^i=(v_1(i), v_2(i))$, where $v_1(i), v_2(i)\in$GF(2^2), then i-th row of $\boldsymbol{A}_4(5,2)$ can be obtained as $(v_1(i), v_2(i))\,\boldsymbol{H}$.

The rows of the resulting seed matrix $\boldsymbol{A}_4(5,2)$ with 17 rows are given in Table 1.

For the matrix $\boldsymbol{A}_4(5,2)$ in any two columns all 16 pairs (x_1,x_2) ($x_1,x_2\in$GF(2^2)) can be found as rows and in each column all 16 pairs (x_1,x_2) can be found in two consecutive rows.

Table 1. Construction of strongly 2-surjective matrix $\boldsymbol{A}_4(5,2)$

i	$v_1(i)$	$v_2(i)$	i-th row of $A_4(5,2)$
0	0	0	0 0 0 0 0
1	1	0	1 0 1 1 1
2	0	1	0 1 1 α α^2
3	α	1	α 1 α^2 0 1
4	α	α^2	α α^2 1 α^2 0
5	1	1	1 1 0 α^2 α
6	α	0	α 0 α α α
7	0	α	0 α α α^2 1
8	α^2	α	α^2 α 1 0 α
9	α^2	1	α^2 1 α 1 0
10	α	α	α α 0 1 α^2
11	α^2	0	α^2 0 α^2 α^2 α^2
12	0	α^2	0 α^2 α^2 1 α
13	1	α^2	1 α^2 α 0 α^2
14	1	α	1 α α^2 α 0
15	α^2	α^2	α^2 α^2 0 α 1
16	1	0	1 0 1 1 1

The elements of this matrix can be easily generated using an embedded BIST test generator, based on 2 LFSR's [4] (see Section 4).

The resulting test can be constructed by combining the generated backgrounds with the MARCH test (MATS+), as illustrated in Table 2 for n=5, w=2 [4]. Here $a_0,\dots,a_4$ are the initial states of the cells $W_1,\dots,W_4$. *Read/Write* operations to cell W_j are denoted as $r(W_j)$ and $w(W_j)$. Each 5 clock cycles next test background (printed in bold in Table 2) is generated.

Table 2. Construction of MARCH test to detect all 2-couplings in word-oriented memory for n=5, w=2

t	$r(W_j)$	$w(W_j)$	W_0	W_1	W_2	W_3	W_4
0			a_0	a_1	a_2	a_3	a_4
1		$w(W_0)$	$\underline{0}$	a_1	a_2	a_3	a_4
2		$w(W_1)$	0	$\underline{0}$	a_2	a_3	a_4
3		$w(W_2)$	0	0	$\underline{0}$	a_3	a_4
4		$w(W_3)$	0	0	0	$\underline{0}$	a_4
5		$w(W_4)$	**0**	**0**	**0**	**0**	$\underline{\mathbf{0}}$
6	$r(W_4)$	$w(W_4)$	0	0	0	0	$\underline{1}$
7	$r(W_3)$	$w(W_3)$	0	0	0	$\underline{1}$	1
8	$r(W_2)$	$w(W_2)$	0	0	$\underline{1}$	1	1
9	$r(W_1)$	$w(W_1)$	0	$\underline{0}$	1	1	1
10	$r(W_0)$	$w(W_0)$	$\underline{\mathbf{1}}$	**0**	**1**	**1**	**1**
11	$r(W_0)$	$w(W_0)$	$\underline{0}$	0	1	1	1
12	$r(W_1)$	$w(W_1)$	0	$\underline{1}$	1	1	1
13	$r(W_2)$	$w(W_2)$	0	1	$\underline{1}$	1	1
14	$r(W_3)$	$w(W_3)$	0	1	1	$\underline{\alpha}$	1
15	$r(W_4)$	$w(W_4)$	**0**	**1**	**1**	$\boldsymbol{\alpha}$	$\underline{\boldsymbol{\alpha}^2}$
...	...	...	...	...	...	...	...
81	$r(W_0)$	$w(W_0)$	$\underline{1}$	α^2	0	α	1
82	$r(W_1)$	$w(W_1)$	1	$\underline{0}$	0	α	1
83	$r(W_2)$	$w(W_2)$	1	0	$\underline{1}$	α	1
84	$r(W_3)$	$w(W_3)$	1	0	1	$\underline{1}$	1
85	$r(W_4)$	$w(W_4)$	**1**	**0**	**1**	**1**	$\underline{\mathbf{1}}$
86	$r(W_4)$		1	0	1	1	$\underline{1}$
87	$r(W_3)$		1	0	1	$\underline{1}$	1
88	$r(W_2)$		1	0	$\underline{1}$	1	1
89	$r(W_1)$		1	$\underline{0}$	1	1	1
90	$r(W_0)$		$\underline{\mathbf{1}}$	**0**	**1**	**1**	**1**

3.3 Construction of matrices $\boldsymbol{M}_b(n,s)$

For s=1, matrices $\boldsymbol{M}_b(n,s)$ contain only one row, since we can use the same column to construct surjective matrix. Thus

$$\boldsymbol{M}_1(n,1) = [1\ 1\ \dots\ 1],$$

$$\boldsymbol{A}_q(n,1) = \begin{bmatrix} 0 & \dots & 0 \\ 1 & \dots & 1 \\ \alpha & \dots & \alpha \\ \dots & \dots & \dots \\ \alpha^2 & \dots & \alpha^2 \end{bmatrix} \quad \text{and } f_q(n,1)=q.$$

Now, let us consider the case s=2. In this case, it is necessary and sufficient that all columns of $\boldsymbol{M}_b(n,s)$ should be different. So, we can simply write numbers 0,...,n-1 in the (q+1)-ary system as the columns of $\boldsymbol{M}_{q+1}(n,s)$.

For example, let us assume that we have a strongly 2-surjective seed matrix $\boldsymbol{A}_4(5,2)$ with 5 columns and 4^2+1=17 rows (as it was constructed in the previous Section). We can use the following matrix $\boldsymbol{M}_5(25,2)$ with 25 columns:

$$\boldsymbol{M}_5(25,2) = \begin{bmatrix} 0\,1\,2\,3\,4\,0\,1 \dots 4\,0\,1\,2\,3\,2 \\ 0\,0\,0\,0\,0\,1\,1 \dots 3\,4\,4\,4\,4\,4 \end{bmatrix}.$$

After substitution, the resulting test matrix $\boldsymbol{A}_4(25,2)$ will have 33 rows.

In the general case, using this approach we receive the following formula for the number of rows in the resulting strong 2-surjective matrices:

Statement 4. $f_q(n,2) \le q^2 \lceil \log_{q+1} n \rceil + 1$.

For example $f_{16}(10^6,2)$=1,281 for q=16.

For s>2, procedures for construction of matrices $\boldsymbol{M}_b(n,s)$ are more complex. Several different procedures can be used. For example, we can combine shift and concatenation operation, as it is shown by the following example (s=3, b=6):

$$\boldsymbol{M}_6(14,3) = \begin{bmatrix} 0\,1\,2\,3\,4\,5\,5\,5\,5\,5\,5\,5\,5\,5 \\ 0\,0\,0\,0\,0\,1\,2\,3\,4\,5\,5\,5\,5\,5 \\ 0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,2\,3\,4\,5 \end{bmatrix}.$$

Using this approach we can construct matrices $\boldsymbol{M}_b(n,s)$ with $\lceil (n\text{-}2)/(b\text{-}2) \rceil$ rows.

Another procedure, which will be used in this paper, was proposed in [1]. The main idea of this construction is the following: for each iteration r mutually prime numbers $p_1,\dots, p_r$ are selected and matrix $\boldsymbol{M}_b(n,s)$ is constructed, where $m_{ij} = i \bmod p_j$ ($j \in \{1,\dots,r\}$) .

An example of matrix $\boldsymbol{M}_5(8,3)$ constructed in such a way, is given below:

$$\boldsymbol{M}_5(8,3) = \begin{bmatrix} 0\,1\,2\,3\,4\,0\,1\,2 \\ 0\,1\,2\,3\,0\,1\,2\,3 \\ 0\,1\,2\,0\,1\,2\,0\,1 \end{bmatrix} \begin{matrix} (p_1=5) \\ (p_2=4) \\ (p_3=3). \end{matrix}$$

Here the seed matrix had 5 columns and the resulting matrix $\boldsymbol{A}_4(8,3)$ - 8 columns.

The following result was proven in [1] and can be generalized to any q.

Statement 5. Let $\boldsymbol{M}=(m_{ij})$ and $m_{ij}=i$ (mod p_j) and there exist strongly s-surjective matrices $\boldsymbol{A}_q(p_j,s)$ with $f_q(p_j,s)$ rows. If $p_1,\dots,p_r$ are mutually prime such that $\prod_{i=1}^{r} p_i \ge n^{\binom{s}{2}}$ then there exists a strongly s-surjective matrix $\boldsymbol{A}_q(n,s)$ with $f_q(n,s) = \sum f_q(p_j,s)$ rows.

For example, for s=3, b=18 and r=4 we can select numbers 13,15,16,17. By statement 5, matrix $\boldsymbol{M}_{18}(n,3)$ with $n=\lceil(13\bullet15\bullet16\bullet17)^{1/3}\rceil = 38$ columns and 4 rows can be constructed.

Numbers of rows in matrices $\boldsymbol{M}_b(n,s)$ can be further decreased using the following statement:

Statement 6. For mutually prime numbers $p_1,...,p_r$, p_1£ p_2£...£ p_r£ b, matrix $\boldsymbol{M}_b(n,s)$ with r rows and $n = p_1p_2...p_k$, columns can be constructed, where $k =\lceil 2r/(s\bullet(s\text{-}1)\rceil$.

Using this statement for the example above will result in n = 195 - so we can construct matrix $\boldsymbol{M}_{18}(195,3)$ with 195 columns and 4 rows.

For large n several iterations will be used, with different seed matrices at every iteration. For example, for s=3 and q=16 we can matrices $\boldsymbol{M}_{18}(72,3)$ with 3 rows and $\boldsymbol{M}_{18}(195, 3)$ with 4 rows construct using the seed matrix $\boldsymbol{A}_{16}(18,3)$. Resulting 3-surjective $(72\times(3\bullet16^3+1))$ and $(195\times(4\bullet16^3+1))$ matrices can be combined using mutually prime numbers 71,191,193,195 to construct strongly 3-surjective matrix with 71•191=13,561 columns and $(3+4\bullet3)\bullet16^3+1$=61,441 rows. As a rule, at each subsequent iteration more numbers should be used, and the increase in the number of columns is exponential. To find an optimum solution a computer search can be needed.

The last remark is that sometimes it is better to select non-mutually prime numbers. For example, instead of selecting 7,11,13,15,16 (q=16, s=3) it is better to select 11,13,14,15,16.

3.4 Example and experimental results

To illustrate the proposed method, let us consider two important cases q=4 and q=16.

For q=4 s-couplings with s=3 and s=4 can be detected in practice for large n. In the first case seed matrix $\boldsymbol{A}_4(6,3)$ with 4^3+1=65 rows is used. Using matrix $\boldsymbol{M}_6(14,3)$ (see above), we receive strongly 3-surjective matrix $\boldsymbol{A}_4(16,3)$ with $3\bullet4^3+1$ = 193 rows. For the next iteration mutually prime numbers p_1=9, p_2=11, p_3=13, p_4=14 can be used. Then we obtain matrix $\boldsymbol{M}_{14}(117,3)$ with 4 rows. One more iteration, using 7 mutually prime numbers allows to generate the test matrix for testing 1-Mbit memory with $3\bullet4\bullet7\bullet64+1 \approx 5.4 * 10^3$ rows. Thus $f_4(2.5*10^5,3) \leq 5.4 \bullet 10^3$.

In a similar way strongly 4-surjective matrices are constructed, based on seed matrix $\boldsymbol{A}_4(5,4)$ with 4^4=256 rows.

Now we consider another practical example with q=16, s=3, $n=10^6$. (4-Mbit memory) Seed matrix $\boldsymbol{A}_{16}(18,3)$ with 16^3=4096 rows can be used in this case. Taking p_1=13, p_2=15, p_3=16, p_4=17 we receive matrix $\boldsymbol{M}_{18}(195,3)$ with 4 rows. Using 7 mutually prime numbers at the second step, we receive matrix $\boldsymbol{M}_{143}(10^6,3)$ with 7 rows. So, the resulting 3-surjective matrix will have 10^6 columns and $4\bullet7\bullet4096 \approx 1.15\bullet10^5$ rows. Assuming that each cell *Read/Write* operation will take 50 ns, resulting test time will be t(4-Mbit, 16, 3) $\approx 1.15\bullet10^4$ sec ≈3 hours (see Table 3). This result is better than one achieved by the existing methods. Some other results for 1-Mbit word-oriented memory are the following: $t_4(5\bullet10^5,2) \approx 12$ sec, $t_{16}(2.5\bullet10^5, 2) \approx 2$ min, $t_4(5\bullet10^5,3) \approx 10$ min, $t_{16}(2.5\bullet10^5,3) \approx 50$ min.

Table 3. Mutual prime numbers selection for construction of $\boldsymbol{M}_{18}(n,3)$ (q=16, s=3).

Number of cells n	Number of iterations	Number of columns in a seed matrix	Mutually prime numbers selected	Number of rows in M $m_{18}(n,3)$
72	1	18	11,17,18	3
195	1	18	13,15,16,17	4
330	1	18	11,13,14,15,16	5
$3.6\bullet10^4$	2	195	191,193,194,195	16
$7\bullet10^6$	2	195	181,183,187,191, 193,194,195	28

More results on application of the proposed methods are given at Tables 4, 5 and 6. In Table 4 numbers of rows in matrices $\boldsymbol{M}_b(n,s)$ are given, where b=q+1 for s=2 and s=4 and b=q+2 for s=3. In Table 5 numbers of test backgrounds, and in Table 6 resulting test times for 1 Mbit memory are given. Also, in Table 6 results for the bit-oriented memories are presented [1]. Access time is assumed to be $5\bullet10^{-8}$ sec.

Table 4. Numbers of rows $m_b(n,s)$ for a 1Mbit memory

s \ w	2	4	8
1	1	1	1
2	12	5	3

Table 6. Test times for 1Mbit memory

s \ w	1	2	4	8
1	0.2 sec	0.2sec	0.4 sec	3.2 sec
2	0.4 sec	8.6 sec	32 sec	40 min
3	8.2 sec	4.5 min	36 min	405 hours
4	55 sec	13.8 h.	89 hours	$1.8{\bullet}10^5$ h.

We note that numbers of rows in matrices $\boldsymbol{M}_b(n,s)$ can be further reduced, if we allow that the fraction of s-tuples of columns such that the probability that there exists a row with all different elements in the selected columns is at least 1-ε for small ε>0 (in the previous sections we assumed ε>0).

Statement 7. Let $\boldsymbol{M}=(m_{ij})$, where $m_{ij}=i \pmod{p_j}$, $p_1,...,p_r$ are mutually prime numbers, $s<p_1<p_2<...<p_r$, and $\prod_{j-1}^{r} p_j / n \to 0$. Then we have for the probability $q(p_1,...,p_r, s)$ that for a randomly selected s-tuple of columns there exists a row with all different entries in these columns:

$$q(p_1,...,p_r,s) \geq 1-\prod_{i=1}^{r}(1-\prod_{j=1}^{s-1}(1-j/p_i)).$$

For example, if s=4, q=16, p_1=13, p_2=15, p_3=16, p_4=17 then ε ≈0.002.

4. Hardware realization

The hardware overhead needed to realize the proposed test procedure for built-in self testing consists of hardware required for generation of elements of seed matrix $\boldsymbol{A}_q(b,s)$ (2 LFSRs, see Fig.1 and 2), a hardware to generate elements of $\boldsymbol{M}_b(n,s)$, a comparator and a control unit.

An example of the network generating rows of 2-surjective seed matrix $\boldsymbol{A}_q(q+1,3)$ with q+1 columns is given by Fig.1. It is assumed that the initial memory state is "all zeroes" [4].

The given network contain two LFSRs. First LFSR (LFSR-b) is implementing the primitive polynomial $l(x)=x^2+x+a$ (on GF(4)) and is used to order the backgrounds and consists of 4 flip-flops, second (LFSR-a) generates a^i. This LFSR if implementing polynomial $j(x)=x^2+x+1$ and it consists of 2 flip-flops. The realization of these LFSRs is shown in Fig.2.

The clock pulse input C_o is used for loading initial LFSR-b state $(v_0(i), v_1(i))=(1,0)$ Clock pulses C_1 and C_2 enable all flip-flops to make the shift operations (see Fig.1). LFSR-a is the binary w-bit LFSR over GF(q), which operate as ordinary LFSR when C_3=0 and as register with parallel load when C_3=1. Detailed description of control unit operation is given at [4].

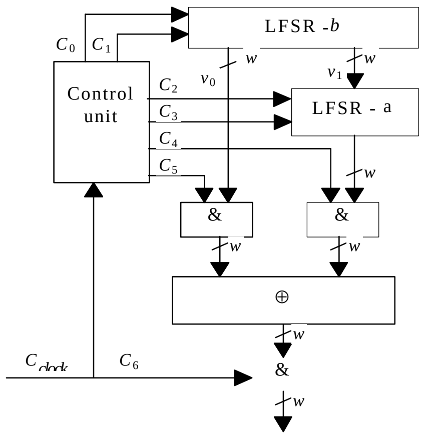
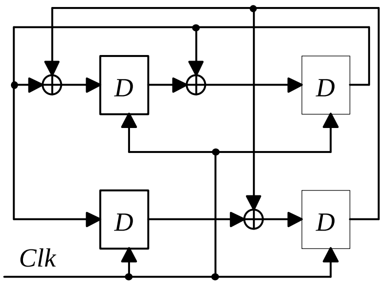
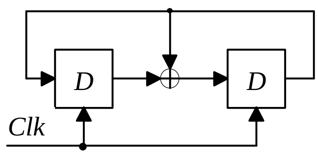


Fig. 1. Network generating rows of seed matrix $\boldsymbol{A}_4(5,2)$



a) LFSR-b



b) LFSR-a

Fig. 2. Hardware realization of LFSR-a and LFSR-b for q=4, s=2.

A hardware implementation of the network generating rows of seed matrix $\boldsymbol{A}_q(n,2)$ requires $3w+(w+2)\lceil \log_{q+1} n \rceil$ flip-flops and 12w gates.

In the general case (s>2) s-1 LFSRs, corresponding to $\alpha,\alpha^2,\alpha^{s-1}$ are required, each with w flip-flops, along with one LFSR-b with ws flip-flops.

To generate elements of matrices $\boldsymbol{M}_b(n,s)$, it is needed to store all prime numbers p_i, used for the row generation.

We note that all used matrices $\boldsymbol{M}_b(n,s)$ have the following properties:

1) $m(1,j) = 0$;

2) $m(i+1,j) \in \{0, m(i,j), m(i,j)+1\}$.

These properties can be used to generate the resulting test.

The total hardware overhead to detect 2- and 3-couplings is about 3% in terms of gates and 5% in terms of the area for a 1-Mbit DRAM. This system was implemented in hardware for q=2 and results confirm the estimations [1].

5. Conclusions

In this paper we develop an efficient approach for detection and location of unlinked couplings between cells in word-oriented memories. The proposed approach is a generalization of the approach suggested in [1] for the binary case. The proposed method requires 36 min for detection of 100% of 3-couplings between words for 1Mbit DRAM with 4 bits in each word and 50 ns access time. The required overhead for BIST implementation is less than 5%. Test time can be drastically reduced, if a small percentage of faults will be not detected.

References

1. Das D., Karpovsky M. "Exhaustive and Near-Exhaustive Memory Testing Techniques and their BIST implementation", *Journal of Electronic Testing*, 1997, 10, pp.215-229.

2. Cockburn B.F., "Deterministic Tests for Detecting Single V-Coupling Faults in RAMs", *Journal of Electronic Testing*, 1994, 1, pp.91-113.

3. Savir J., McAnney W.H., Vecchio S.R., "Testing for Coupled Cells in Random Access Memories", *IEEE Trans. on Computers,* 1991, 10, pp.1177-1180.

4. Karpovsky M.G., Yarmolik V.N., van de Goor A.G., "Pseudoexhaustive Word-Oriented DRAM Testing", *Proc. European Design and Test Conf.*, 1995, pp.24-38.

5. Levitin L.B., Karpovsky M.G., "Efficient Exhaustive Test Based on MDS Codes", *Proc. of IEEE Int. Symp. on Information Theory*, 1986, p.64.

6. MacWilliams J.F., Sloane N.J., "The Theory of Error-Correcting Codes", North-Holland Publishing Company, Amsterdam, 1977.

7. Goor A.G. "Using March Tests to Test SRAMs", *IEEE Design and Test of Computers*, 1975, 1, pp.8-14.

8. Karpovsky M.G.,Das D., Vardhan H. "Built in Self Testing for Detection of Coupling Faults in Semiconductor Memories",

9. Nordholz P., Otterstedt J., Niggemeyer D., "A Defect-Tolerant Word-Oriented Static RAM with Built-In Self-Test and Self-Reconfiguration", *Proc. of Int. Conf. on Innovative Systems in Silicon*, Austin, Texas, USA, pp. 124 - 132, 1996.

10. Bardell P.N. et al. "Built-in Test for VLSI: Pseudo-Random Techniques", John Wiley & Sons, New York, N.Y, 1987.

11. Suk D.S., Reddy S.M. "Test Procedures for a Class of Pattern-Sensitive Faults in Semiconductor Random-Access Memories", *IEEE Trans. on Computers*, 1980, 6, pp.419-429.

12. Treuer R.P., Agarwal V.K. "Built-in Self-Diagnosis for Repairable Embedded RAMs", IEEE Design and Test of Computers, 1993, 1, pp. 23-33.

13. Treuer R.P., Agarwal V.K. "Fault Location Algorithms for Repairable Embedded RAMs", *Proc. of Int. Test Conf.*, 1993, pp.825-834.

14. Yarmolik V.N., Demidenko S.N. "Generation and Application of Pseudorandom Sequences for Random Testing", John Wiley & Sons, New York, N.Y, 1988.

15. Jain S.K., Stroud S.H. "Buit-in Self-Testing of Embedded Memories", *IEEE Design and Test of Computers*, 1986, 5, pp.27-37.