Int. J. Appl. Math. Comput. Sci., 2010, Vol. 20, No. 1, 191–205
DOI: 10.2478/v10006-010-0014-6

ANALYSIS OF MULTIBACKGROUND MEMORY TESTING TECHNIQUES

IRENEUSZ MROZEK

Institute of Computer Science
Białystok Technical University, Wiejska 45A, 15–351 Białystok, Poland
e-mail: i.mrozek@pb.edu.pl

March tests are widely used in the process of RAM testing. This family of tests is very efficient in the case of simple faults such as stuck-at or transition faults. In the case of a complex fault model—such as pattern sensitive faults—their efficiency is not sufficient. Therefore we have to use other techniques to increase fault coverage for complex faults. Multibackground memory testing is one of such techniques. In this case a selected March test is run many times. Each time it is run with new initial conditions. One of the conditions which we can change is the initial memory background. In this paper we compare the efficiency of multibackground tests based on four different algorithms of background generation.

Keywords: RAM testing, pattern sensitive faults, March tests, multibackground testing.

1. Introduction

Modern semiconductor memories are among the most fundamental integrated-circuit devices and cores in digital systems (Krasniewski, 2008; Sosnowski, 2007; Zorian, 2002). Their testing is quickly becoming a more difficult issue as the rapidly increasing capacity and density of memory chips. With the advances of deep-submicron technology, more failure modes and faults need to be dealt with in order to maintain good quality and reliability of memory chips. The neighborhood pattern sensitive fault (NPSF) model is not new, but it is still widely discussed in the literature of memory testing (Goor, 1991; Cheng *et al*., 2002; Huang and Li, 2006). This model is more general and allows describing a wide spectrum of failures within modern memory chips.

Traditional March algorithms (Goor, 1991) have been widely used in memory testing because of their linear time complexity, high fault coverage, and ease in built-in self-test (BIST) implementation. It is known that traditional March algorithms do not generate all neighborhood patterns that are required for testing NPSFs. However, they can be modified to get detection abilities for NPSFs. Based on traditional March algorithms, various approaches have been proposed to detect NPSFs, such as the tiling method (Goor, 1991; Hayes, 1975), the two-group method (Goor, 1991; Hayes, 1980), the row-March algorithm (Franklin and Saluja, 1996), transparent testing (Cockburn, 1995; Karpovsky and Yarmolik, 1994; Nicolaidis, 1996; Voyiatzis, 2006), pseudo-exhaustive testing (Karpovsky *et al*., 1995), testing based on different address sequences (Sokol and Yarmolik, 2006; Yarmolik, 2008) and different address seeds (Yarmolik, 2008), and the multibackground method (Yarmolik and Mrozek, 2007).

For one execution of a March test there are no specific requirements on the address order or the memory background (Niggemeyer *et al*., 1998). For any address order and memory background, the number of detectable memory faults will be the same and can be calculated according to the structure and properties of the memory test (Niggemeyer *et al*., 1998; Yarmolik and Mrozek, 2007). In the case of multibackground memory tests, backgrounds play a very important role in the final outcome. As was shown and investigated in (Yarmolik and Mrozek, 2007), different subsets of backgrounds can give different subsets of detectable faults for a selected memory test. The selection of an optimal set of backgrounds to get the highest fault coverage is still an open issue and there are no known algorithms for optimal background generation and construction in the case of more than four iterations of the test.

The main goal of this paper is to compare the efficiency of multibackground March tests (in terms of pattern sensitive fault detection) based on standard backgrounds (Goor, 1991; Karpovsky and Yarmolik, 1994; Cheng *et al*., 2002) and the optimal one, which was proposed by the author in (Yarmolik and Mrozek, 2007;

Mrozek and Yarmolik, 2008b). The paper is organized as follows: Section 2 provides an overview of pattern sensitive faults, their types and definitions. Section 3 reviews March test abilities regarding PSF detection. This section reviews the March test structure and provides an analysis of the fault coverage of the *MATS+* test in the case of PSF faults. Moreover, this section stresses the importance of proper background selection in multibackground testing. Section 4 covers background generation techniques. This section reviews the structure of backgrounds which are used in multibackground testing. Subsections 4.1–4.3 describe and analyse standard backgrounds known from the literature. Subsection 4.4 develops in-depth optimal sets of backgrounds for two, three and four run memory testing. Section 5 presents the results of the comparison of the efficiency (in term of PSF detection) of standard backgrounds and the optimal one. This section presents the fault coverage of the multirun *MATS+* test for PNPSF*k* faults and various backgrounds.

2. Pattern sensitive faults

Several types of faults can occur in memory devices, e.g., stuck-at faults, transitions faults, coupling faults, address decoder faults, pattern sensitive faults (Goor, 1991). Some of them involve only one memory cell, some of them—more than one memory cell. It is obvious that the latter are more difficult to detect. A general model of faults belonging to the second group are pattern sensitive faults. It occurs when the content of a memory cell, or the ability to change the cell content, is influenced by a certain pattern of other cells in the memory. Considering all possible patterns, as has been shown in numerous publications, is both impractical and unnecessary.

A simplified model of the PSF, known as the neighborhood PSF (NPSF), is normally adopted. An NPSF is a special case of the general multicell coupling fault, wherein the coupling cells are the neighborhood of the coupled cell. In general, the coupled cell is called the *base cell* and the coupling cells are called the *deleted neighborhood cells*. The base cell and the deleted neighborhood cells together are called *neighborhood cells*. Among *k*-cell NPSFs (NPSF*k*), three-cell NPSFs (NPSF3), five-cell NPSFs (NPSF5) and nine-cell NPSFs (NPSF9) are most often used (see Fig. 1) (Goor, 1991). This fault model can further be categorized into three subtypes of faults (Goor, 1991; Huang and Li, 2006): static NPSFs (SNPSF), passive NPSFs (PNPSF) and active NPSFs (ANPSF).

A *static NPSF* (SNPSF) occurs if the base cell is forced to a certain state due to the appearance of a certain pattern in the deleted neighborhood. To detect SNPSF3 and SNPSF5, all the eight (for SNPSF3) and all the 32 (for SNPSF5) static neighborhood patterns must be applied, and the generation of these patterns by the test algorithm must be verified.

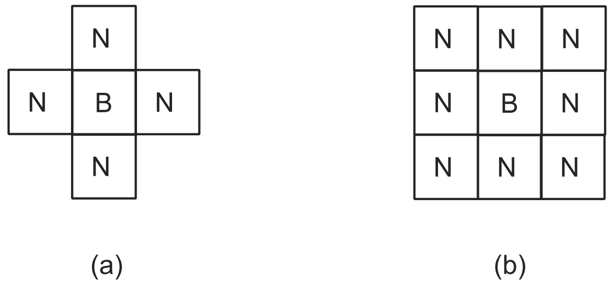

Fig. 1. Topology of NPSF faults: Type 1—five-cell NPSF (a), Type 2—nine-cell NPSF (b).

A *passive NPSF* (PNPSF) occurs if the base cell cannot change its state from 0 to 1 or from 1 to 0 due to the appearance of a certain pattern in the deleted neighborhood. To detect PNPSF3 and PNPSF5, all the eight (for PNPSF3) and all the 32 (for PNPSF5) static neighborhood patterns must be applied, and the generation of these patterns within neighborhood cells by the test algorithm must be verified.

An *active NPSF* (ANPSF) occurs if the base cell is forced to a certain state when a transition occurs in a deleted neighborhood cell, while other deleted neighborhood cells assume a certain pattern. To detect ANPSF3 and ANPSF5, all 16 (for ANPSF3) and all 128 (for ANPSF5) static neighborhood patterns must be applied, and the generation of these patterns by the test algorithm must be verified.

Let us focus our attention on the PNPSF as the most difficult fault to be detected. First of all, it should be emphasized that due to scrambling information as well as specific optimization techniques a huge amount of such types of faults that should be considered. Any arbitrary *k* memory cells out of all *N* memory cells can be involved into PNPSF*k*. There are *k* subtypes of distinct PNPSF*k* faults. This classification depends on the order in the address space and on places of all the cells within this space.

Let the memory addresses $i_0, i_1, i_2 \ldots i_{k-1}$ for particular PNPSF*k* be sorted in ascending order, in such a way that $i_0 < i_1 < i_2 < \cdots < i_{k-1}$. Then, every PNPSF*k* can be presented as the set of elements $a_{i0}, a_{i1}, a_{i2}, \ldots, a_{i(k-1)}$, $a_{ij} \in \{0, 1\}$; $j = 0, 1, 2, \ldots, (k-1)$ ordered in the address space according to the ascending order of memory cell addresses. One out of *k* cells is the base cell. This means that there are *k* separate classes of PNPSF*k* with respect to the base cell position. For example, in the case when $k = 5$, there are five separate classes of PNPSF*k*, namely, $b_{i0}n_{i1}n_{i2}n_{i3}n_{i4}$, $n_{i0}b_{i1}n_{i2}n_{i3}n_{i4}$, $n_{i0}n_{i1}b_{i2}n_{i3}n_{i4}$, $n_{i0}n_{i1}n_{i2}b_{i3}n_{i4}$ and $n_{i0}n_{i1}n_{i2}n_{i3}b_{i4}$ (b_{ij}—base cell, n_{ij}—neighborhood cell). For neighborhood patterns there are 2^{k-1} different patterns and there are two states for the base cell. Then the exact number of PNPSF*k* is determined as

$$L_k = 2 \times 2^{k-1} \times k \times \binom{N}{k} = 2^k \times k \times \binom{N}{k}. \quad (1)$$

It is quite important to emphasize that there is an equal number of the faults considered within all k classes, namely, L_k/k. For example, in the case of PNPSF3 we have 448 faults b_{i0}, n_{i1}, n_{i2}, 448 faults n_{i0}, b_{i1}, n_{i2}, and 448 faults n_{i0}, n_{i1}, b_{i2}.

3. PSF detection capabilities of March tests

Different types of algorithms have been proposed to test random access memory (RAM). Among them March tests are most often used because of their linear complexity and the number of types of fault detection. Also, it is easy to transform them from non-transparent to a transparent version. Transparent tests have the circular property to ensure the recovery of a memory content. For the fault free case, at the end of the test session a memory content will have the same value as before the session, and for faulty memory, the content at the end of the session will be different from the initial content.

A March test consists of a sequence of March elements; a March element consists of a sequence of operations which are all applied to a given cell, before proceeding to the next one. The way one proceeds to the next cell is determined by the address order which can be increasing (increasing addresses from cell 0 to $n-1$), denoted by "⇑", or decreasing, denoted by "⇓". The "⇑" address order has to be the exact inverse of the "⇓" address order (Goor, 1991). For some March elements, the address order can be chosen arbitraily—this will be indicated by the symbol "⇕". An operation applied to a cell can be 'w0' (write '0'), 'w1', "r0" (read 0), or "r1". A well-known March test is *March C-*, which has complexity equal to $(9N)$ and the following transparent form: $\{\Uparrow (ra, w\bar{a}); \Uparrow (r\bar{a}, wa); \Downarrow (ra, w\bar{a}); \Downarrow (r\bar{a}, wa); \Updownarrow (ra)\}$, where $a \in \{0, 1\}$ and $\bar{a}$ is an inverse value compared with a (Nicolaidis, 1996).

Now let us examine some memory tests in terms of their abilities to detect PNPSF*k*. As has been observed, the main part of memory tests usually has sequences of the phases. For example, the transparent *MATS+* test is constructed as $\{\Uparrow (ra, w\bar{a}); \Downarrow (r\bar{a}, wa)\}$. Suppose that we use it for testing 8-bit memory and the initial value of the memory (*background*) is $A = a_0, a_1, a_2, a_3, a_4, a_5, a_6, a_7$. For the 8-bit memory, there are $2 \times 3 \times 2^{3-1} \times 56 = 1344$ PNPSF3 and $2 \times 5 \times 2^{5-1} \times 56 = 8960$ PNPSF5 (see (1)). The consecutive states of the tested memory according to the *MATS+* procedure are shown in Table 1.

Table 1. MATS+ test implementation.

MATS+ Phases	Memory contents							
	a_0	a_1	a_2	a_3	a_4	a_5	a_6	a_7
$\Uparrow (ra, w\bar{a})$	$\mathbf{\bar{a}_0}$	a_1	a_2	a_3	a_4	a_5	a_6	a_7
	$\bar{a}_0$	$\mathbf{\bar{a}_1}$	a_2	a_3	a_4	a_5	a_6	a_7
	$\bar{a}_0$	$\bar{a}_1$	$\mathbf{\bar{a}_2}$	a_3	a_4	a_5	a_6	a_7
	$\bar{a}_0$	$\bar{a}_1$	$\bar{a}_2$	$\mathbf{\bar{a}_3}$	a_4	a_5	a_6	a_7
	$\bar{a}_0$	$\bar{a}_1$	$\bar{a}_2$	$\bar{a}_3$	$\mathbf{\bar{a}_4}$	a_5	a_6	a_7
	$\bar{a}_0$	$\bar{a}_1$	$\bar{a}_2$	$\bar{a}_3$	$\bar{a}_4$	$\mathbf{\bar{a}_5}$	a_6	a_7
	$\bar{a}_0$	$\bar{a}_1$	$\bar{a}_2$	$\bar{a}_3$	$\bar{a}_4$	$\bar{a}_5$	$\mathbf{\bar{a}_6}$	$\bar{a}_7$
	$\bar{a}_0$	$\bar{a}_1$	$\bar{a}_2$	$\bar{a}_3$	$\bar{a}_4$	$\bar{a}_5$	$\bar{a}_6$	$\mathbf{\bar{a}_7}$
$\Downarrow (r\bar{a}, wa)$	$\bar{a}_0$	$\bar{a}_1$	$\bar{a}_2$	$\bar{a}_3$	$\bar{a}_4$	$\bar{a}_5$	$\bar{a}_6$	$\mathbf{\bar{a}_7}$
	$\bar{a}_0$	$\bar{a}_1$	$\bar{a}_2$	$\bar{a}_3$	$\bar{a}_4$	$\bar{a}_5$	$\mathbf{\bar{a}_6}$	a_7
	$\bar{a}_0$	$\bar{a}_1$	$\bar{a}_2$	$\bar{a}_3$	$\bar{a}_4$	$\mathbf{\bar{a}_5}$	a_6	a_7
	$\bar{a}_0$	$\bar{a}_1$	$\bar{a}_2$	$\bar{a}_3$	$\mathbf{\bar{a}_4}$	a_5	a_6	a_7
	$\bar{a}_0$	$\bar{a}_1$	$\bar{a}_2$	$\mathbf{\bar{a}_3}$	a_4	a_5	a_6	a_7
	$\bar{a}_0$	$\bar{a}_1$	$\mathbf{\bar{a}_2}$	a_3	a_4	a_5	a_6	a_7
	$\bar{a}_0$	$\mathbf{\bar{a}_1}$	a_2	a_3	a_4	a_5	a_6	a_7
	$\mathbf{\bar{a}_0}$	a_1	a_2	a_3	a_4	a_5	a_6	a_7

It should be mentioned that as the memory address sequence, counter sequences were chosen, and the starting address was $i_0 = 0$. Now we can see the only pattern appearing, within all memory for every active cell marked in bold. Indeed, we check (read a_i and write the inverse value $\bar{a}_i$ during the first phase and read $\bar{a}_i$ and write a_i during the second phase) the cell a_i in both phases for the same background in the remaining cells. The activation of PNPSF*k* can occur during the write operation for the base cell only, as well as detection during the read operation. This means that activation for the *MATS+* test occurs during the first phase only and detection during the second one.

To summarize, we can conclude that we can detect PNPSF*k* only for one neighborhood pattern in $k-1$ cells out of 2^{k-1} possible patterns and for one transition within the base cell from state 0 to state 1 or from 1 to 0. Depending on the size k of PNPSF*k* there are k subtypes of this fault for which detectable faults will be different in terms of the pattern within the deleted neighboring cells. That is why the number Q_k of detectable faults during one *MATS+* memory test run is

$$Q_k = k \times \binom{N}{k}, \quad (2)$$

and fault coverage (FC) for the *MATS+* test is

$$FC_{M+}(PNPSFk) = \frac{Q_k}{L_k} 100\%. \quad (3)$$

As an example, according to (3) the fault coverage of the *MATS+* March test in term of PNPSF5 can be calculated as $FC_{M+}(PNPSF5) = (1/2^5)100\% = 3.125\%$. Moreover, it should be stressed that the fault coverage described by (3) is valid for every memory March test with the consecutive phases as in *MATS+*.

To investigate memory March tests, let us suppose that PNPSF*k* includes memory cells with increasing order of the addresses $i_0, i_1, i_2, \ldots, i_{k-1}$, in such a way that $i_0 < i_1 < i_2 < \ldots < i_{k-1}$ and the base cell has the address i_j , where $0 \leq j \leq k-1$. Then, due to the consecutive access to the memory cells during the March test, there are four possible patterns within deleted neighborhood cells:

1) $\bar{\alpha}_{i0}, \bar{\alpha}_{i1}, \bar{\alpha}_{i2} \cdots \bar{\alpha}_{i(j-1)}, \alpha_{i(j+1)} \cdots \alpha_{i(k-2)}, \alpha_{i(k-1)}$,
2) $\alpha_{i0}, \alpha_{i1}, \alpha_{i2} \cdots \alpha_{i(j-1)}, \bar{\alpha}_{i(j+1)} \cdots \bar{\alpha}_{i(k-2)}, \bar{\alpha}_{i(k-1)}$,
3) $\bar{\alpha}_{i0}, \bar{\alpha}_{i1}, \bar{\alpha}_{i2} \cdots \bar{\alpha}_{i(j-1)}, \bar{\alpha}_{i(j+1)} \cdots \bar{\alpha}_{i(k-2)}, \bar{\alpha}_{i(k-1)}$,
4) $\alpha_{i0}, \alpha_{i1}, \alpha_{i2} \cdots \alpha_{i(j-1)}, \alpha_{i(j+1)} \cdots \alpha_{i(k-2)}, \alpha_{i(k-1)}$.

The first pattern can be generated by the test which includes one of the following phases: $\{\ldots \Uparrow (ra, \ldots, w\bar{a}); \ldots\}$ and $\{\ldots \Downarrow (r\bar{a}, \ldots, wa); \ldots\}$. The second pattern appears for the case of $\{\ldots \Downarrow (ra, \ldots, w\bar{a}); \ldots\}$ and $\{\ldots \Uparrow (r\bar{a}, \ldots, wa); \ldots\}$. The third pattern is possible for the phases $\{\ldots \Uparrow (r\bar{a}, \ldots, w\bar{a}); \ldots\}$ and $\{\ldots \Downarrow (r\bar{a}, \ldots, w\bar{a}); \ldots\}$, and the fourth pattern can be generated in the neighborhood cells by the following phases: $\{\ldots \Uparrow (ra, \ldots, wa); \ldots\}$ and $\{\ldots \Downarrow (ra, \ldots, wa); \ldots\}$. It should be mentioned that the above March test phases are sufficient only for fault manifestation and do not necessarily guarantee detectability of PNPSF*k*. To achieve their detectability, the read operation of the base cell has to be performed, which can be done within the same phase or a consecutive one. Brief analysis of the tests *MATS+* and *March C-* allows us to make the conclusion that in the first case there is only one pattern generated in the neighborhood cells

$$\bar{\alpha}_{i0}, \bar{\alpha}_{i1}, \bar{\alpha}_{i2}, \ldots, \bar{\alpha}_{i(j-1)}, \alpha_{i(j+1)}, \ldots, \alpha_{i(k-2)}, \alpha_{i(k-1)},$$

and in the second case there are two patterns generated in the neighborhood cells:

$$\bar{\alpha}_{i0}, \bar{\alpha}_{i1}, \bar{\alpha}_{i2}, \ldots, \bar{\alpha}_{i(j-1)}, \alpha_{i(j+1)}, \ldots, \alpha_{i(k-2)}, \alpha_{i(k-1)}$$

and

$$\alpha_{i0}, \alpha_{i1}, \alpha_{i2}, \ldots, \alpha_{i(j-1)}, \bar{\alpha}_{i(j+1)}, \ldots, \bar{\alpha}_{i(k-2)}, \bar{\alpha}_{i(k-1)}.$$

The simplest March test to detect NPSF*k* is *March PS (4N)*:

$$\{\Updownarrow (wa); \Uparrow (ra, w\bar{a}, r\bar{a}); \}. \tag{4}$$

Like in the case of the *MATS+* test, the *March PS (4N)* test generates only one pattern and that is why the coverage of PNPSF*k* for one run of this test and an arbitrary background can be calculated according to (3).

As an experimental investigation, the influence of the second background of two runs of 8-bit memory testing for PNPSF3 for the test *March PS (4N)* and *March PS(23N)* is shown in Fig. 2 and Table 2. It should be mentioned that for one run of the memory test *March PS(23N)* fault coverage equals to 66.58%, and for *March PS (4N)* fault coverage equals to 12.50%. In both cases, background 00000000 has been chosen as the first one.

Table 2. Two background memory test fault coverage (March PS(4N)) for PNPSF3.

March PS(4N)	
Second Pattern $a_0a_1 \cdots a_6a_7$	FC [%]
00000001 00000010 $\cdots$ 10000000	17.19
00000011 00000101 $\cdots$ 11000000	20.54
00000111 00001011 $\cdots$ 11100000	22.77
00001111 00010111 $\cdots$ 11110000	24.11
00011111 00101111 $\cdots$ 11111000	24.78
00111111 01011111 $\cdots$ 11111100	25.00

Brief analysis of the experimental results allows making the conclusion that, depending on the second background, fault coverage takes sufficiently different values. For example, in the case of *March PS (4N)* fault coverage for the second background 00000010 increases only by 4.69%, but in the case of the second background 11111111—by 12.50% (see Table 2).

4. Background generation techniques

In this section, four different techniques of background generation will be investigated. We will focus on random backgrounds, regular backgrounds, random pairs of backgrounds, and optimal backgrounds.

4.1. Random backgrounds. One of the approaches to background generation is to use random backgrounds

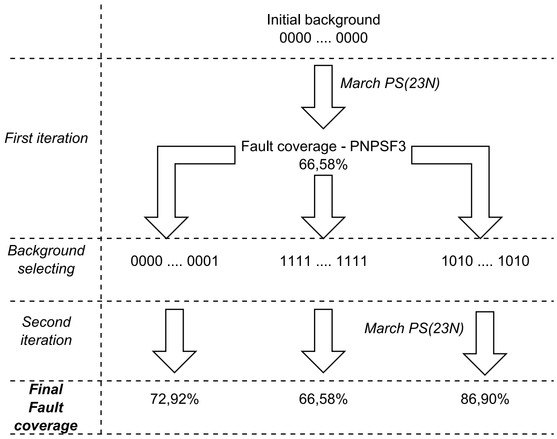

Fig. 2. Influence of the second background on PNPSF3 fault coverage (March PS(23N)).

$B_0, B_1, B_2, \ldots, B_l$. Let P be a probability of fault detection by a certain test. Then the probability of not detecting this fault by the same test is

$$P^* = 1 - P. \tag{5}$$

The probability of fault detection in the second iteration of the test (assuming a random change of the memory background between iterations) is

$$P_2 = 1 - (1 - P)(1 - P) = 1 - (1 - P)^2. \tag{6}$$

Generally speaking, it can be said that the probability of fault detection after l iterations

$$P_l = 1 - (1 - P)^l. \tag{7}$$

One run of the *MATS+* test allows getting the fault coverage $1/2^k$ (see (3)). Therefore, fault coverage for test session based on the *MATS+* test and random backgrounds can be calculated as

$$FC_{M+}(random) = 1 - \left(1 - \frac{1}{2^k}\right)^l, \tag{8}$$

where *M+* means *MATS+*.

The fault coverage of the test session based on *MATS+* test and random backgrounds for different numbers of iterations and PNPSF3, PNPSF5 and PNPSF9 are presented in Table 3.

In the case of transparent testing we can take the advantage of the fact that RAM undergoes constant changes in working computer systems. Most modern operating systems today have a virtual memory module implemented. This makes the contents of physical memory change to a high degree. To confirm this fact, a real computer system was tested (Linux system, Kernel 2.6, 512 MB RAMs). The physical memory content was read after every 15 minutes of the computer's work. Results obtained for a lightly-loaded system (without running additional software after installation) are presented in Fig. 3

Table 3. Fault coverage of a test session based on the MATS+ test and random backgrounds.

Iterations	PNPSF3	PNPSF5	PNPSF9
1	12.500 %	3.125 %	0.195 %
2	23.438 %	6.152 %	0.390 %
4	41.382 %	11.926 %	0.778 %
8	65.639 %	22.430 %	1.549 %
10	73.692 %	27.202 %	1.933 %
15	86.507 %	37.888 %	2.885 %
20	93.079 %	47.005 %	3.829 %
40	99.521 %	71.915 %	7.511 %

and for a heavy-loaded system (with running a lot of memory consuming software)—in Fig. 4. Each bar represents physical memory changes (in %) in relation to the previous state.

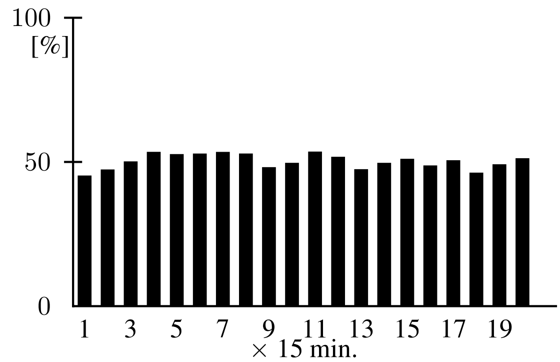

Fig. 3. Changing memory contents in time—a lightly-loaded system.

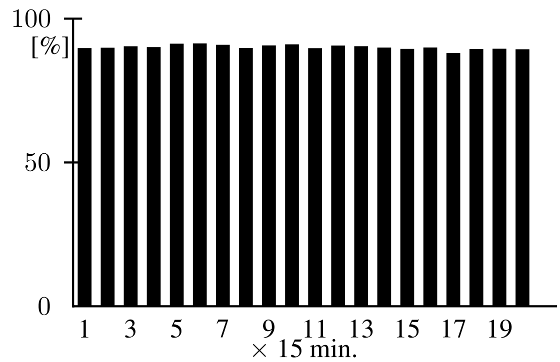

Fig. 4. Changing memory contents in time—a heavy-loaded system.

From the obtained results you can see that in a real computer system memory can be divided into two parts: the one where changes are big enough to successfully use multibackground testing based on random backgrounds, and that where changes are not big enough and where we can use another technique, for example, the one based on March address order changing (Sokol and Yarmolik, 2006).

4.2. Regular backgrounds. Another solution is the application of $q = 2(\log_2 N + 1)$ backgrounds with the standard and well-known structure (Goor, 1991). For the case of $N = 2^m$, there are $2(m+1)$ backgrounds. For example, with $m = 5$, all 12 backgrounds are shown in Table 4. The construction of this set can be done for the case of transparent testing with 1s indicating the positions of an inverted value of the original background $B_0 = b_0b_1b_2 \dots b_{N-2}b_{N-1}$. The positions of 1s for B_2 may take a random value but at the same time their number should be equal to $N/2$. The next background, B_3, is the inverted copy of the previous one and it is true for a general case when any background with an odd index is an inverted copy of the previous background with an even index. For example, B_1 is an inverted copy of B_0, B_3 is an inverted copy of B_2, and so on. More complicated is the procedure of even pattern generation. This is due to the inversion of random bits from the restricted sets of the previous even background.

Table 4. Set of regular backgrounds for $m = 5$.

B_0	00000000000000000000000000000000
B_1	11111111111111111111111111111111
B_2	00000000000000001111111111111111
B_3	11111111111111110000000000000000
B_4	00000000111111110000000011111111
B_5	11111111000000001111111100000000
B_6	00001111000011110000111100001111
B_7	11110000111100001111000011110000
B_8	00110011001100110011001100110011
B_9	11001100110011001100110011001100
B_{10}	10101010101010101010101010101010
B_{11}	01010101010101010101010101010101

The efficiency of this set of backgrounds is shown in Table 4 and can be calculated analytically (Mrozek *et al.*, 2008). To simplify the analysis, let us consider the case of PNPSF*k* detection based on *MATS+* like tests as one of representatives of simple tests, when $k \ll N$.

The first background, B_0, from the defined set (see Table 4) allows us to generate all-zeros patterns within any k out of N arbitrary memory cells. That is why on the basis of a *MATS+* like test with the background B_0 the number of $Q_{M+}(B_0)$ detectable PNPSF*k* faults can be calculated as

$$Q_{M+}(B_0) = k \times \binom{N}{k}. \tag{9}$$

The background B_1 generates completely different patterns, compared with B_0, within the arbitrary k memory cells; then $Q(B_1) = Q(B_0)$ and $Q(B_0, B_1) = Q(B_1) + Q(B_0)$. Taking into account that for real applications N is a big integer number for which $N - k \approx N$ and $N^k \gg N^{k-1}$, the last equations can be simplified to

$$Q_{M+}((B_0, B_1), k) = \frac{2N^k}{(k-1)!}. \tag{10}$$

With the same assumption for N and k, the entire amount of $L(PNPSFk)$ of all possible PNPSF*k* can be approximated by the following equation:

$$L(PNPSFk) = k \times 2^k \times \binom{N}{k} = \frac{2^k N^k}{(k-1)!}. \tag{11}$$

Then the fault coverage $FC(B_0, B_1)$ of the PNPSF*k* faults as the result of two runs of the *MATS+* like test with the backgrounds B_0, B_1 is calculated as

$$\begin{aligned} &FC_{M+}((B_0, B_1), k) \\ &\quad = \frac{Q_{M+}((B_0, B_1), k)}{L(PNPSFk)} 100\% = \frac{1}{2^{k-1}} 100\% \\ &\quad = \left(1 - \left(\frac{2^{k-1} - 1}{2^{k-1}}\right)\right) 100\%. \end{aligned} \tag{12}$$

It should be noted that only all zeros (B_0) and all ones (B_1) patterns within any k arbitrary memory cells are generated based on the backgrounds B_0 and B_1. The next pair of backgrounds, B_2 and B_3, will allow getting new patterns as combinations of 0s and 1s in some groups of k memory cells. The background B_2 and the background B_3 due to their inverse structure allow us to generate the same amount of new patterns within any k memory cells. The full amount of detectable PNPSF*k* faults after four runs of the *MATS+* like test based on the set of the backgrounds B_0, B_1, B_2 and B_3 is

$$\begin{aligned} &Q_{M+}(((B_0 \dots B_3)), k) \\ &\quad = 2k \times \left(\binom{N}{k} + \sum_{i=1}^{k-1} \binom{N/2}{k-i} \binom{N/2}{i}\right). \end{aligned} \tag{13}$$

Taking into account that N is a big integer number, $N^k \gg N^{k-1}$ and $k \ll N$, and the last equation for the case of even N can be simplified to

$$\begin{aligned} &Q_{M+}(((B_0 \dots B_3)), k) \\ &\quad = kN^k \left(\frac{2}{k!} + \frac{2}{2^k} \sum_{i=1}^{k-1} \frac{1}{(k-i)! \times i!}\right). \end{aligned} \tag{14}$$

Then the fault coverage $FC((B_0 \dots B_3))$ of the PNPSF*k* faults is calculated as

$$FC_{M+}(((B_0 \dots B_3)), k) \approx \frac{2^k - 1}{2^{2k-2}} 100\%. \tag{15}$$

The last equation can be simplified to

$$FC_{M+}(((B_0 \dots B_3)), k) \approx \left(1 - \left(\frac{2^{k-1} - 1}{2^{k-1}}\right)^2\right) 100\%. \tag{16}$$

To estimate all possible PNPSF*k* faults detectable on the basis of the backgrounds B_0, B_1, B_2, B_3, B_4 and B_5, let us examine the details of the constructions of the backgrounds B_4 and B_5 (see Fig. 5). First of all, it should be emphasized that patterns are generated on the basis of the background B_5 due to their inverse version of B_4 completely different compared with the patterns generated on the basis of the background B_4. Then the amount of the PNPSF*k* faults detectable on the basis of the background B_5 equals the amount of the faults detectable on the basis of backgrounds B_4. That is why we examine the background B_4 in terms of PNPSF*k* faults detection and multiply the number of detectable PNPSF*k* faults by 2.

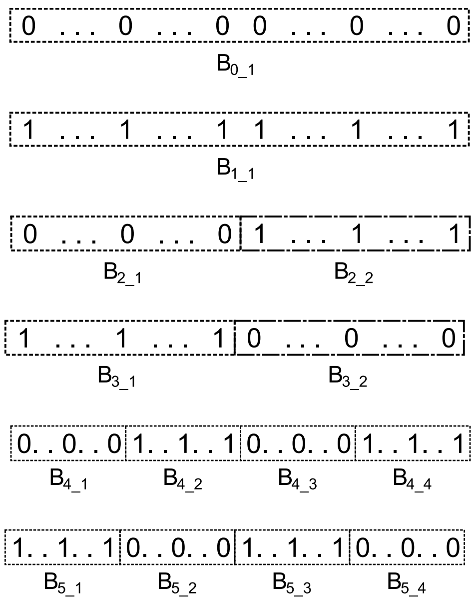

Fig. 5. Construction of regular backgrounds for $m = 3$.

Let suppose that N is divisible by 4 and as a result the background B_4 has four parts, namely, B_{4_1}, B_{4_2}, B_{4_3} and B_{4_4} (see Fig. 5). Compared with the previous backgrounds (B_0, B_1, B_2 and B_3), the background B_4 allows us to generate new patterns for any k (see Tab. 5). For $k = 4$, there are the following new patterns:

$$\begin{aligned} P_1(B_{4\_1}\_B_{4\_2}) &= \{0\_111, 00\_11, 000\_1\},\\ P_2(B_{4\_3}\_B_{4\_4}) &= \{0\_111, 00\_11, 000\_1\},\\ P_3(B_{4\_1}\_B_{4\_2}\_B_{4\_3}) &= \{00\_1\_0, 0\_11\_0, 0\_1\_00\},\\ P_4(B_{4\_1}\_B_{4\_2}\_B_{4\_4}) &= \{00\_1\_1, 0\_11\_1, 0\_1\_11\},\\ P_5(B_{4\_1}\_B_{4\_3}\_B_{4\_4}) &= \{00\_0\_1, 0\_00\_1, 0\_0\_11\},\\ P_6(B_{4\_2}\_B_{4\_3}\_B_{4\_4}) &= \{11\_0\_1, 1\_00\_1, 1\_0\_11\},\\ P_7(B_{4\_1}\_B_{4\_2}\_B_{4\_3}\_B_{4\_4}) &= \{0\_1\_0\_1\}. \end{aligned}$$

Table 5. Patterns are generated based on the background B_4.

	B_{4_1}	B_{4_2}	B_{4_3}	B_{4_4}
P_1	00…0	11…1		
P_2			00…0	11…1
P_3	00…0	11…1	00…0	
P_4	00…0	11…1		11…1
P_5	00…0		00…0	11…1
P_6		11…1	00…0	11…1
P_7	00…0	11…1	00…0	11…1

The same amount of patterns as the inverse copy of the previous one will be generated based on the background B_5. The entire number of the PNPSF*k* faults detectable on the basis of the patterns P_1 and P_2 generated by the backgrounds B_4 and B_5 is

$$Q_{M+}((P_1, P_2), k) = 4k \sum_{i=1}^{k-1} \binom{N/4}{k-i} \binom{N/4}{i}. \tag{17}$$

An additional portion of detectable PNPSF*k* faults is determined by the patterns P_3, P_4, P_5 and P_6 and can be estimated as

$$\begin{aligned} &Q_{M+}((P_3, P_4, P_5, P_6), k) \\ &= 8k \sum_{j=1}^{k-2} \binom{N/4}{j} \sum_{i=1}^{k-j-1} \binom{N/4}{k-i-j} \binom{N/4}{i}. \end{aligned} \tag{18}$$

The last set of PNPSF*k* faults detectable on the basis of the B_4 and B_5 backgrounds is described by the pattern P_7 and equals to

$$\begin{aligned} &Q_{M+}((P_7), k) \\ &= 2k \sum_{j=1}^{k-3} \binom{N/4}{j} \sum_{r=1}^{k-j-2} \binom{N/4}{r} \\ &\quad \times \sum_{i=1}^{k-j-r-1} \binom{N/4}{k-i-j-r} \binom{N/4}{i}. \end{aligned} \tag{19}$$

The full amount of detectable PNPSF*k* faults after six runs of the *MATS+* like test based on the set of the backgrounds B_0, B_1, B_2, B_3, B_4 and B_5 equals

$$\begin{aligned} &Q_{M+}(((B_0, B_1), (B_2, B_3), (B_4, B_5)), k) \\ &= 2k \Bigg(2 \sum_{i=1}^{k-1} \binom{N/4}{k-i} \binom{N/4}{i} \\ &\quad + 4 \sum_{j=1}^{k-2} \binom{N/4}{j} \sum_{i=1}^{k-j-1} \binom{N/4}{k-i-j} \binom{N/4}{i} \\ &\quad + \sum_{j=1}^{k-3} \binom{N/4}{j} \sum_{r=1}^{k-j-2} \binom{N/4}{r} \\ &\quad \times \sum_{i=1}^{k-j-r-1} \binom{N/4}{k-i-j-r} \binom{N/4}{i} \Bigg). \end{aligned} \tag{20}$$

If we simplify the last equation, then the fault coverage of the test based on the six backgrounds is

$$\begin{aligned} &FC_{M+}(((B_0,B_1),(B_2,B_3),(B_4,B_5)),k) \qquad (21)\\ &\approx \frac{Q_{M+}(((B_0,B_1),(B_2,B_3),(B_4,B_5)),k)}{L(PNPSFk)} 100\% \\ &= \left(1-\left(\frac{2^{k-1}-1}{2^{k-1}}\right)^3\right) 100\%. \end{aligned}$$

In the case of the application of the l+1 pair of the backgrounds (B_0, B_1), (B_2, B_3), (B_4, B_5),..., (B_{2l}, B_{2l+1}) for $l \in \{1, 2, 3, \ldots, m\}$, fault coverage can be calculated according to

$$\begin{aligned} &FC_{M+}(((B_0,B_1),\ldots,(B_{2l},B_{2l+1})),k) \qquad (22)\\ &\approx \frac{Q_{M+}(((B_0,B_1),\ldots,(B_{2l},B_{2l+1})),k)}{L(PNPSFk)} 100\% \\ &= \left(1-\left(\frac{2^{k-1}-1}{2^{k-1}}\right)^{l+1}\right) 100\%. \end{aligned}$$

The last equation allows us to get an estimate of the maximal possible fault coverage value $FC_{\text{MAX}} = FC((B_0, B_1), \ldots, B_{2m}, B_{2m+1})$, where $m = \lceil \log_2 N \rceil$, based on the set of regular backgrounds. This value equals

$$FC_{\text{MAX}} = \left(1-\left(\frac{2^{k-1}-1}{2^{k-1}}\right)^{\lceil \log_2 N \rceil + 1}\right). \qquad (23)$$

For some sizes N of the memory, this value is shown in Table 6.

Table 6. Value of FC_{MAX} for varying N.

$N(bit)$	10^3	10^6	10^9
$k = 3$	95.55%	99.76%	99.98%
$k = 4$	76.98%	93.94%	98.40%
$k = 5$	49,16%	74.21%	86.47%

4.3. Random pairs. In the previous section, a technique which used the pairs of backgrounds (B_{2l}, B_{2l+1}) was presented. The background B_{2l+1} is the inversion of the background B_{2l}. The background B_{2l} is generated according to the scheme presented in Table 4. However, we do not always have enough time to generate the background B_{2l}. Therefore, the approach presented in this section is based on consecutive application of the background pairs $(B_j, \overline{B_j})$, where the first background is the random one and the second is its inverted version (random pairs). In the case of transparent testing we can take the advantage of the fact that RAM undergoes constant changes in working computer systems. Consequently, in periodic testing it is possible to treat, each time we start the test procedure, the contents of RAM as a random background. In the case of one run of the *MATS+* test, one background, allows getting the fault coverage $1/2^k$ (see (3)). Therefore the fault coverage of the test session based on random pairs of the background and on the *MATS+* test can be calculated as

$$\begin{aligned} &FC_{M+}((B_0,\overline{B}_0),\ldots,(B_l,\overline{B}_l)) \\ &= 1-\left(1-\frac{1}{2^{k-1}}\right)^{l+1}. \qquad (24) \end{aligned}$$

4.4. Optimal backgrounds. To achieve high fault coverage of PNPSFk for multirun memory testing, it is quite important to choose appropriate backgrounds. Obviously, for different types of memory tests the optimal backgrounds will be different.

To select an optimal background for multi-background memory testing the Hamming distances $\text{HD}(B_k, B_j)$ in between two backgrounds (B_k, B_j) $k, j \in \{1, 2, 3, \ldots, m\}$ as a metric were proposed and experimentally analysed in (Yarmolik and Mrozek, 2007). Based on this metric, the following statement was formulated and experimentally validated (Yarmolik and Mrozek, 2007; Yarmolik, 2008):

Theorem 1. *In the case of m runs of the memory test which allow us to generate only one pattern within neighboring cells based on the backgrounds $B_1, B_2, B_3, \ldots, B_m$, an optimal set of such a type of background should have the maximal Hamming distance $\text{HD}(B_k, B_j)$ between any B_k and B_j, where $k, j \in \{0, 1, 2, \ldots, m\}$.*

This statement can be used for selecting the optimal values of background for memory tests generating only one pattern for k neighboring memory cells like the *MATS+* and *PS (4N)* tests. According to this, in the case of multi-run memory testing, memory backgrounds should have the maximal Hamming distance between all pairs of backgrounds. Now we will try to estimate this value (maximal Hamming distance).

4.4.1. Background dissimilarity measures. In the case of multirun memory testing, every consecutive background should not be similar to the previous one or, more precisely, it should be dissimilar as much as possible compared with the backgrounds applied during the previous test sessions. The memory background can be regarded as a binary vector and the set of backgrounds can be defined as a set of binary vectors $B_i = b_{i1}b_{i2}\ldots b_{iN}$, $i \in \{1, 2, \ldots, 2^N\}$, where $b_{ic} \in \{0, 1\}$, $\forall c \in \{1, 2, \ldots, N\}$, and N is the one-bit wide memory size.

There are numerous measures of binary vector dissimilarity (Tubbs, 1989; Zhang and Srihari, 2003).To measure dissimilarity between two memory backgrounds $B_1 = b_{11}b_{12}\ldots b_{1N}$ and $B_2 = b_{21}b_{22}\ldots b_{2N}$, we can define the characteristics S_{qg} as follows. Given two backgrounds B_1 and B_2, let S_{qg} $(q, g \in \{0, 1\})$ be the number of occurrences of matches with q in B_1 and g in B_2 at the corresponding positions. There are four characteristics, namely, S_{00}, S_{01}, S_{10} and S_{11} which were used to define eight measures of similarity and dissimilarity between two binary vectors (Tubbs, 1989; Zhang and Srihari, 2003). For example, in the case when $B_1 = 010110001100$ and $B_2 = 010100101011$, we have $S_{00} = 4$, $S_{01} = 3$, $S_{10} = 2$ and $S_{11} = 3$.

Based on S_{00}, S_{01}, S_{10} and S_{11}, there exist eight characteristics to evaluate similarity measures and their associated dissimilarity measures, i.e., the Jaccard-Needham, Dice, Correlation, Yule, Russell-Rao, Sokal-Michener, Rogers-Tanmoto and Kulzinsky measures. Four measures—Jaccard-Needham, Dice, Russell-Rao and Kulzinsky—are independent of S_{00} due to unequal importance of “zero” matches (S_{00}) and “one” matches (S_{11}) for different applications, especially for search algorithms and data mining (Zhang and Srihari, 2003). Only some of these measures depend on all four characteristics and can be regarded as metrics, including the Sokal-Michener measure of similarity $S(B_1, B_2) = (S_{11} + S_{00})/N$ and dissimilarity $D(B_1, B_2) = 1 - (S_{11} + S_{00})/N$. For this metric, it is easy to show that, based on the equality $N = S_{00} + S_{01} + S_{10} + S_{11}$, the dissimilarity measure can be represented as $D(B_1, B_2) = (S_{10} + S_{01})/N$.

In our case, the distance between two backgrounds has to be estimated as the Hamming distance $\mathrm{HD}(B_1, B_2) = N \times D(B_1, B_2) = S_{01} + S_{10}$. For example, in the case when $B_1 = 010110001100$ and $B_2 = 010100101011$, we have $\mathrm{HD}(B_1, B_2) = S_{01} + S_{10} = 3 + 2 = 5$.

4.4.2. Two run memory testing. In the case of two run memory testing, based on the *MATS+* and *PS (4N)* like tests, Statement 1 can mathematically be formulated as $\max\{\mathrm{HD}(B_i, B_j)\}$ for $\forall\, i \neq j \in \{1, 2, \dots, 2N\}$. To satisfy this statement, two backgrounds have to have a maximal possible Hamming distance $\mathrm{HD}(B_i, B_j) = N$. To generate the second background B_j, we only need to use the complement $\overline{B}_i = I - B_i$ of the first background B_i as the second background $B_j = \overline{B}_i$. The unit binary vector I is an N-dimensional binary vector with all elements equal to 1. For the previous example, for $B_i = 010110001100$ and $\overline{B}_i = 101001110011$, $S_{00} = S_{11} = 0$, $S_{01} = 7$ and $S_{10} = 5$; then $\mathrm{HD}(B_i, \overline{B}_i) = S_{01} + S_{10} = 7 + 5 = 12$. It is easy to show that $\mathrm{HD}(B_i, \overline{B}_i) = N$ for $i \in \{1, 2, \dots, 2^N\}$.

Consecutive application of two backgrounds B_i and $\overline{B}_i$ guarantees the maximal fault coverage of PNPSF*k* for any k. This follows from the fact that for any k arbitrary cells the background $\overline{B}_i$ provides different patterns compared with the first background B_i, and that is why during

Table 7. Two run MATS+ test fault coverage.

MIN $FC_{2rM+}(PNPSF3)$		MAX $FC_{2rM+}(PNPSF3)$	
Second pattern $B_j = b_{j1}b_{j2}\dots b_{j8}$	FC (%)	Second pattern $B_j = b_{j1}b_{j2}\dots b_{j8}$	FC (%)
00000001 00000010 00000100 00001000 00010000 00100000 01000000 10000000	17.19	00111111 01011111 ... 01111110 10011111 10101111 ... 10111110 ... 01111111 ... 11111110 11111111	25.00

the second run of the test (*MATS+* and *PS (4N)* like) new PNPSF*k* will be detected. Then, fault coverage can be estimated as

$$FC_{2rM+}(PNPSFk) = (1/2^{k-1})100\%. \tag{25}$$

The same fault coverage can be achieved for another pair of backgrounds taking into account the next observation. For any pair of backgrounds $B_i = b_{i1}b_{i2}\dots b_{iN}$ and $B_j = b_{j1}b_{j2}\dots b_{jN}$ with $\mathrm{HD}(B_i, B_j) > N - k$, there are not the same patterns for any k arbitrary b_{il} and b_{jl}. Taking into account that during the *MATS+* test the value of memory cells takes an inversion value, it is easy to show that there are not the same patterns for any k arbitrary cells during the test session. As an example for the case of 8-bit memory for the first background $B_i = b_{i1}b_{i2}\dots b_{i8} = 00000000$, there are 37 backgrounds shown in Table 7 allowing us to get $FC_{2rM+}(PNPFSk) = (1/2^{k-1})100\%$ for the $k > 2$. For $k = 3$ $FC_{2rM+}(PNPFSk) = 25\%$. For a greater value of k, the number of the second optimal background will be sufficiently high.

It is quite important to emphasise that for the second background which does not satisfy the inequality $\mathrm{HD}(B_i, B_j) > N - k$ the fault coverage $FC_{2rM+}(PNPFSk)$ is less than the maximal one. Even for the small differences, high fault coverage cannot be achieved. For example, in the case of two backgrounds $B_i = 00000000$ and $B_j = 00011111$ and $k = 3$ with $\mathrm{HD}(B_i, B_j) = N - k = 5$, the fault coverage $FC_{2rM+}(PNPFS3) = 24.78\%$.

Based on this investigation for the case of two run memory testing, it is possible to formulate the next statement (Mrozek and Yarmolik, 2008b):

Theorem 2. *In the case of two runs of the memory test which allow us to generate only one pattern within neighboring cells based on two backgrounds B_i and B_j, an optimal set of such type of background should satisfy the inequality* $\mathrm{HD}(B_i, B_j) > N - k$ *for* $i, j \in \{1, 2, 3, \dots, 2^N\}$*, where N is one bit-wide memory size.*

A more complicated problem arises for three and more runs of memory testing based on different backgrounds.

4.4.3. Three run memory testing. First of all, for the case of three-run memory testing taking into account Statement 1, we have to estimate the maximum minimal possible Hamming distance between any pair of (B_i, B_j), (B_i, B_l) and (B_l, B_j) out of three backgrounds $\{B_i, B_j, B_l\}$ $\forall$ $i \neq j \neq l \in \{1, 2, \ldots, 2^N\}$. This problem can be formulated as a maximum minimal hamming distance problem. Mathematically, it can be formulated as

$$\begin{aligned}\mathrm{MMHD}(B_i, B_j, B_l) &= \operatorname*{MAX}_{\forall i\neq j\neq l\in\{1,2,\ldots,2^N\}} \big\{\mathrm{MIN}[\mathrm{HD}(B_i, B_j), \\ &\quad \mathrm{HD}(B_i, B_l), \mathrm{HD}(B_j, B_l)]\big\}.\end{aligned} \tag{26}$$

Consider two arbitrary backgrounds B_i and B_j with four characteristics $S_{00}(B_i, B_j)$, $S_{01}(B_i, B_j)$, $S_{10}(B_i, B_j)$ and $S_{11}(B_i, B_j)$. Then, $\mathrm{HD}(B_i, B_j) = S_{01}(B_i, B_j) + S_{10}(B_i, B_j)$, and let it be the maximal one for the three-background case, which we are looking for. Now, to construct the third background B_l, we have to get as large as possible Hamming distances between B_l and the previously obtained backgrounds B_i and B_j. In this case B_l should be equally far (in terms of the Hamming distance) from both backgrounds B_i and B_j. This distance can be calculated based on four characteristics: $S_{00}(B_i, B_j)$, $S_{01}(B_i, B_j)$, $S_{10}(B_i, B_j)$ and $S_{11}(B_i, B_j)$ for B_i, B_j.

Taking into account that the backgrounds $B_i = b_{i1}b_{i2}\ldots b_{iN}$ and $B_j = b_{j1}b_{j2}\ldots b_{jN}$ have $S_{00}(B_i, B_j) + S_{11}(B_i, B_j)$ equal bits ($b_{ic} = b_{jc}$, $c \in \{1, 2, \ldots, N\}$), the background B_l should have the opposite bits in the corresponding positions. In this case, the distances $\mathrm{HD}(B_i, B_l)$ and $\mathrm{HD}(B_l, B_j)$ will satisfy the inequalities $\mathrm{HD}(B_i, B_l) \geq S_{00}(B_i, B_j) + S_{11}(B_i, B_j)$ and $\mathrm{HD}(B_l, B_j) \geq S_{00}(B_i, B_j) + S_{11}(B_i, B_j)$. Now $S_{01}(B_i, B_j) + S_{10}(B_i, B_j)$ bits for the background B_l should be chosen to maximize both values $\mathrm{HD}(B_i, B_l)$ and $\mathrm{HD}(B_l, B_j)$. To maximize these distances, the best solution can be achieved for the case of the equality of two Hamming distances $\mathrm{HD}(B_i, B_l)$ and $\mathrm{HD}(B_l, B_j)$. This means that half of bits out of $S_{01}(B_i, B_j) + S_{10}(B_i, B_j)$ different bits ($b_{ic} \neq b_{jc}, c \in \{1, 2, \ldots, N\}$) for B_i and B_j in the corresponding positions of B_l should have the same value as in B_i and the rest of bits the same value as in B_j.

The background B_l generated according to the above presented procedure has the distances $\mathrm{HD}(B_i, B_l)$ and $\mathrm{HD}(B_l, B_j)$ determined by the following equation:

$$\begin{aligned}\mathrm{HD}(B_i, B_l) &= \mathrm{HD}(B_j, B_l) \\ &= S_{00}(B_i, B_j) + S_{11}(B_i, B_j) \\ &\quad + \frac{1}{2}\big[S_{01}(B_i, B_j) + S_{10}(B_i, B_j)\big] \\ &= N - \frac{1}{2}\mathrm{HD}(B_i, B_j)\end{aligned} \tag{27}$$

when $\mathrm{HD}(B_i, B_j)$ is an even number. In the case of an odd value of $\mathrm{HD}(B_i, B_j)$, it is determined by

$$\mathrm{HD}(B_i, B_l) = N - \lfloor \frac{1}{2}\mathrm{HD}(B_i, B_j)\rfloor \tag{28}$$

$$\mathrm{HD}(B_j, B_l) = N - \lceil \frac{1}{2}\mathrm{HD}(B_i, B_j)\rceil. \tag{29}$$

It should be noted that $\lfloor \mathrm{HD}(B_i, B_j)/2 \rfloor + \lceil \mathrm{HD}(Bi, Bj)/2 \rceil = \mathrm{HD}(B_i, B_j)$.

For example (Fig. 6), in the case of two backgrounds $B_i = 011100$, $B_j = 010011$ we have $S_{00}(B_i, B_j) = 1$, $S_{01}(B_i, B_j) = 2$, $S_{10}(B_i, B_j) = 2$ and $S_{11}(B_i, B_j) = 1$ and an even value of $\mathrm{HD}(B_i, B_j) = S_{10}(B_i, B_j) + S_{01}(B_i, B_j) = 2 + 2 = 4$. Four characteristics were calculated on the basis of the values of all components of both backgrounds: $B_i = b_{i1}b_{i2}b_{i3}b_{i4}b_{i5}b_{i6}$ and $B_j = b_{j1}b_{j2}b_{j3}b_{j4}b_{j5}b_{j6}$. $S_{00}(B_i, B_j) = 1$ follows from the fact that only $b_{i1} = b_{j1} = 0$, $S_{01}(B_i, B_j) = 2$ due to $b_{i5} = b_{i6} = 0$ and $b_{j5} = b_{j6} = 1$, $S_{10}(B_i, B_j) = 2$ due to $b_{i3} = b_{i4} = 1$ and $b_{j3} = b_{j4} = 0$, and $S_{11}(B_i, B_j) = 1$ because $b_{i2} = b_{j2} = 1$. To generate a new background B_l, their first and second bits have to have an inverse value compared with B_i and B_j, namely, $b_{l1} = 1$, due to $b_{i1} = b_{j1} = 0$ and $b_{l2} = 0$, because $b_{i2} = b_{j2} = 1$. Then the first half of the bits (two bits) with an opposite value in B_i and B_j should take a value from one background. Let it be B_i (for example, $b_{l3} = b_{l4} = 1$) and the second half the values from background B_j (for example, $b_{l5} = B_{l6} = 1$). The final result is $B_l = b_{l1}b_{l2}b_{l3}b_{l4}b_{l5}b_{l6} = 101111$ and

$$\begin{aligned}\mathrm{HD}(B_i, B_l) &= \mathrm{HD}(B_l, B_j) \\ &= S_{00}(B_i, B_j) + S_{11}(B_i, B_j) \\ &\quad + \frac{1}{2}[S_{01}(B_i, B_j) + S_{10}(B_i, B_j)] \\ &= 1 + 1 + \frac{1}{2}(2 + 2) = 4.\end{aligned}$$

Now the problem of the maximum minimal Hamming distance (26) can be formulated as the solution of the following problem:

$$\begin{aligned}&\mathrm{MMHD}(B_i, B_j, B_l) \\ &= \operatorname*{MAX}_{\mathrm{HD}(B_i,B_j)} \big\{\mathrm{MIN}[\mathrm{HD}(B_i, B_j), \\ &\quad (N - \lceil \mathrm{HD}(B_i, B_j)/2\rceil), (N - \lfloor \mathrm{HD}(B_i, B_j)/2)\rfloor]\big\}.\end{aligned} \tag{30}$$

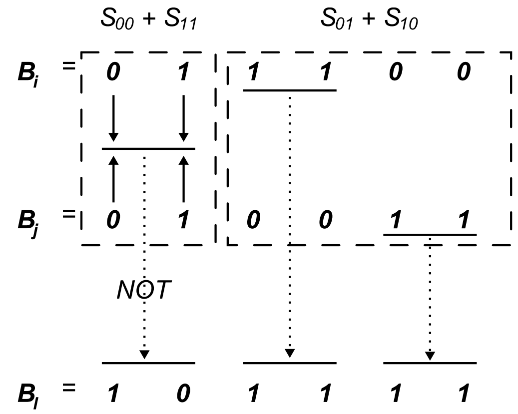

Fig. 6. Optimal third vector generation scheme.

Using the notation $X = \mathrm{HD}(B_i, B_j)$, we will get the final equality $X = N - X/2$. An unknown value of X which allows us to get the minimal differences $X-(N-X/2) = 0$ or 1 is the solution to our problem. For a small value of N, the optimal values of the Hamming distances are shown in Table 8. For the validation of the presented

Table 8. Optimal Hamming distances for small N.

N	$\mathrm{HD}(B_i, B_j)$	$\mathrm{HD}(B_i, B_l)$	$\mathrm{HD}(B_j, B_l)$
4	3	3	2
5	3	3	4
6	4	4	4
7	5	5	4
8	6	5	5
9	6	6	6
10	7	7	6
12	8	8	8

results, the experimental values of the fault coverage of PNPSF3 for $N = 8$ in the case of the *MATS+* like test are shown in Table 9.

For real large N in the case of a three run *MATS+* like memory test, the following statement is true (Mrozek and Yarmolik, 2008b):

Theorem 3. *In the case of three runs of the memory test which allows us to generate only one pattern within neighboring cells based on three backgrounds B_i, B_j and B_l ($i \neq j \neq l \in \{1, 2, \ldots, 2^N\}$ and N is one bit-wide memory size), an optimal set of such a type of background should satisfy the following equality:*

$$\mathrm{HD}(B_i, B_j) = \mathrm{HD}(B_i, B_l) = \mathrm{HD}(B_j, B_l) \approx \frac{2}{3}N. \tag{31}$$

Let us prove that for the case of three backgrounds the maximum minimal hamming distance $\mathrm{MMHD}(B_i, B_j, B_l)$ cannot be greater than $2N/3$. To simplify our investigation, suppose that $2N/3$ is an integer number and, according to the previous statement, $\mathrm{MMHD}(B_i, B_j, B_l) = \mathrm{HD}(B_i, B_j) = \mathrm{HD}(B_i, B_l) = \mathrm{HD}(B_l, B_j) = 2N/3$.

Table 9. Experimental results for $N = 8$.

Backgrounds B_i, B_j, B_l	MIN $\mathrm{HD}(B_i, B_j, B_l)$	FC_{3rM+} [%]
00000000 11111111 00000001	1	29.68
00000000 11111111 11110000	4	35.71
00000000 11110000 00001111	4	35.71
00000000 11111000 00011111	5	37.05

Now let $\mathrm{MMHD}(B_i, B_j, B_l) = 2N/3 + 1$, which is greater than $2N/3$. From this fact all distances $\mathrm{HD}(B_i, B_j)$, $\mathrm{HD}(B_i, B_l)$ and $\mathrm{HD}(B_l, B_j)$ should be greater than or equal to $2N/3 + 1$. Let the distance $\mathrm{HD}(B_i, B_j) = 2N/3 + 1$, then $S_{00}(B_i, B_j) + S_{11}(B_i, B_j) = N - (2N/3 + 1) = N/3 - 1$ and that is why $\mathrm{HD}(B_i, B_l) = N/3 - 1 + Q$, where $Q \le 2N/3 + 1$ and $\mathrm{HD}(B_l, B_j) = N/3 - 1 + (2N/3 + 1 - Q) = N - Q$. For $\mathrm{HD}(B_i, B_l) \ge 2N/3 + 1$, the value of Q should satisfy the inequality $Q \ge N/3 + 2$. Then $\mathrm{HD}(B_l, B_j) = N - Q \le N - (N/3 + 2) = 2N/3 - 2$, which is less than $2N/3 + 1$. That is why $\mathrm{MMHD}(B_i, B_j, B_l)$ cannot be greater than $2N/3$. The maximal fault coverage FC_{3rM+} for the three runs of the MATS+ test can be estimated as (Mrozek and Yarmolik, 2008a)

$$FC_{3rM+}((B_i, B_j, B_l), k) \approx \left(\frac{1}{2^k} + \frac{2}{3^k} - \frac{1}{2^k 3^k} + \frac{2}{2^k 3^k}\sum_{i=1}^{k-1} 2^{k-i}\binom{k}{i}\right)100\%. \tag{32}$$

In the case when $k = 3$, according to (32), $FC_{3rM+} = 36.10\%$.

4.4.4. Four run memory testing. Now, for the case of four run memory testing taking into account Statement 1 we have to estimate the maximum minimal possible Hamming distance between any pair (B_i, B_j), (B_i, B_l), (B_i, B_r), (B_j, B_l), (B_j, B_r) and (B_l, B_r) out of four backgrounds $\{B_i, B_j, B_l, B_r\}$ $i \neq j \neq l \neq r \in$

$\{1, 2, \ldots, 2^N\}$. Mathematically, this problem can be formulated as

$$\begin{aligned}&\mathrm{MMHD}(B_i, B_j, B_l, B_r)\\&= \underset{\forall i\neq j\neq l\neq r\in\{1,2,\ldots,2^N\}}{\mathrm{MAX}} \{\mathrm{MIN}\\&[\mathrm{HD}(B_i, B_j), \mathrm{HD}(B_i, B_l), \mathrm{HD}(B_i, B_r),\\&\mathrm{HD}(B_j, B_l), \mathrm{HD}(B_j, B_r), \mathrm{HD}(B_l, B_r)]\}.\end{aligned} \tag{33}$$

Let us have three first arbitrary backgrounds B_i, B_j and B_l with the optimal value of $\mathrm{MMHD}(B_i, B_j, B_l)$. As has been shown earlier for large N, this value can be regarded as integer number $2N/3$. Then $\mathrm{HD}(B_j, B_j) = \mathrm{HD}(B_i, B_l) = \mathrm{HD}(B_l, B_j) = 2N/3$.

We have to emphasize that for the case of three backgrounds it is impossible to get a greater value of $\mathrm{MMHD}(B_i, B_j, B_l)$. that is why for the case of four backgrounds $\mathrm{MMHD}(B_i, B_j, B_l, B_r)$ it cannot be grater than $2N/3$, either. This means that the best solution for the case of four backgrounds will be the fourth background with the distances between it and the first three backgrounds equal $\mathrm{HD}(B_r, B_i) = \mathrm{HD}(B_r, B_j) = \mathrm{HD}(B_r, B_l) = 2N/3$. Since the backgrounds B_i and B_j have $S_{01}(B_i, B_j) + S_{10}(B_i, B_j) = 2N/3$ different bits, the third background B_l was generated by the selection of part of its bits from the background B_i and the another part from B_j, as well as the inversion of all $S_{00}(B_i, B_j) + S_{11}(B_i, B_j) = N/3$ equal bits for B_i and B_j.

When we create the next background B_r as the selection of other parts of $S_{01}(B_i, B_j) + S_{10}(B_i, Bj_{)} = 2N/3$ different bits from the backgrounds B_i and B_j and the inversion of all $S_{00}(B_i, B_j) + S_{11}(B_i, B_j) = N/3$ bits, this background can be regarded as the third background compared with B_i and B_j. This follows from the conclusion that the background B_r has the same distances $\mathrm{HD}(B_r, B_i) = \mathrm{HD}(B_r, B_j) = 2N/3$ as the background B_l, $\mathrm{HD}(B_l, B_i) = \mathrm{HD}(B_l, B_j) = 2N/3$.

From the procedure of generating B_l and B_r we can conclude that in $S_{01}(B_i, B_j) + S_{10}(B_i, B_j) = 2N/3$ positions with the different bits for B_i and B_j, the backgrounds B_l and B_r have an inverse value of bits. Then $\mathrm{HD}(B_l, B_r) = 2N/3$. To summarize, it is easy to show that $\mathrm{HD}(B_i, B_j) = \mathrm{HD}(B_i, B_l) = \mathrm{HD}(B_j, B_l) = \mathrm{HD}(B_i, B_r) = \mathrm{HD}(B_j, B_r) = \mathrm{HD}(B_l, B_r) = 2N/3$.

For the previous example, in the case of two backgrounds $B_i = 011100$, $B_j = 010011$, the third background $B_l = b_{l1}b_{l2}b_{l3}b_{l4}b_{l5}b_{l6} = 101111$ was generated to satisfy the equality $\mathrm{HD}(B_i, B_j) = \mathrm{HD}(B_i, B_l) = \mathrm{HD}(B_j, B_l) = 2N/3$. To generate a new, fourth background B_r, its first and second bits have to have an inverse value compared with B_i and B_j, namely, $b_{l1} = 1$, due to $b_{i1} = b_{j1} = 0$ and $b_{l2} = 0$, because $b_{i2} = b_{j2} = 1$. Then another part (compared with the case of B_l generation) of the bits (two bits) with an opposite value in B_i and B_j should take a value from one background. Let it be B_j (e.g., $b_{l3} = b_{l4} = 0$) and the second part—the values from the background B_i (e.g., $b_{l5} = b_{l6} = 0$). The final result is $B_l = b_{l1}b_{l2}b_{l3}b_{l4}b_{l5}b_{l6} = 100000$, which satisfies the next statement (Mrozek and Yarmolik, 2008b):

Theorem 4. *In the case of four runs of the memory test which allows us to generate only one pattern within neighboring cells based on four backgrounds B_i, B_j, B_l and B_r ($i \neq j \neq l \neq r \in \{1, 2, \ldots, 2^N\}$ and N is one bit-wide memory size), an optimal set of such a type of background should satisfy the following equality:*

$$\begin{aligned}\mathrm{HD}(B_i, B_j) &= \mathrm{HD}(B_i, B_l) = \mathrm{HD}(B_j, B_l)\\&= \mathrm{HD}(B_i, B_r) = \mathrm{HD}(B_j, B_r)\\&= \mathrm{HD}(B_l, B_r) \approx 2N/3.\end{aligned} \tag{34}$$

The maximal fault coverage FC_{4rM+} for the optimal backgrounds and four runs of the *MATS+* test can be estimated as (Mrozek and Yarmolik, 2008a)

$$\begin{aligned}&FC_{4rM+}((B_i, B_j, B_l, B_r), k)\\&\approx \Bigg(\frac{1}{2^k} + \frac{1}{3^{k-1}} - \frac{1}{2^k 3^{k-1}}\\&+ \frac{1}{2^k 3^{k-1}} \sum_{i=1}^{k-1} 2^{k-i} \binom{k}{i}\Bigg) 100\%.\end{aligned} \tag{35}$$

In the case when $k = 3$, according to (35), $FC_{4rM+} \approx 47.25\%$. For the validation of the above results, the experimental values for the fault coverage of PNPSF3 for $N = 9$ in the case of the *MATS+* like test are shown in Table 10. The maximal fault coverage from Table 10 is somewhat different than the result based on (35), because in (35) there was assumption about large N (Mrozek and Yarmolik, 2008a).

5. Results

Based on (8), (22), (24), (32) and (35), we can compare the efficiency (in terms of the PNPSF*k* detection ability) of the test procedures based on the *MATS+* like test and the presented backgrounds. First, we will compare standard backgrounds (random backgrounds, random pairs and regular backgrounds). The fault coverage of the procedure which consists of multiple runs of the *MATS+* test and applications of regular backgrounds is presented in Table 11. In this table, k means the size of the PNPSF*k* fault and l—the number of the pairs of backgrounds (iterations) applied. The same results for random backgrounds are presented in Table 3.

The third test procedure is based on consecutive application of the backgrounds pairs $(B_j, \overline{B_j})$, where the first background is the random one and the second is its inverted version. The results for the procedure based on

Table 10. Experimental results for $N = 9$.

Backgrounds B_i, B_j, B_l, B_r	MIN HD(B_i, B_j, B_l, B_r)	FC_{4rM+} [%]
000000000 000001111 000000111 000000001	1	28.1
000000000 111111111 000000111 000001100	2	40.1
000000000 111111111 000000111 110000011	3	42.1
000000000 111111000 111000111 000111111	6	48.4

Table 11. Regular background efficiency.

k/l	0	1	2	3	4
3	25.00%	43.75%	57.81%	68.36%	76.27%
4	12.50%	23.44%	33.01%	41.38%	48.71%
5	6.25%	12.11%	17.60%	22.75%	27.58%
k/l	5	6	7	8	9
3	82.20%	86.65%	89.99%	92.49%	94.37%
4	55.12%	60.73%	65.64%	69.93%	73.69%
5	32.11%	36.35%	40.33%	44.06%	47.55%

the *MATS+* like test and such backgrounds are presented in Table 13.

From the presented results we can see that all the sets of backgrounds allows us to achieve very similar fault coverage. Some better results can be achieved with regular backgrounds (see Table 11) and random pairs of backgrounds (see Table 13) than with random backgrounds. Therefore it seems that, especially in periodic testing, random pairs of backgrounds allow us to achieve very good results with minimal hardware overhead.

Now let we compare the efficiency (in terms of NPSFk detection) of the test procedure based on standard backgrounds and optimal backgrounds. Unfortunately, nowadays the author is able to generate only four optimal backgrounds according to algorithm presented in (Mrozek and Yarmolik, 2008b). Algorithms for optimal background generation for more than four iterations are still an open issue. Therefore we compare the efficiency of only four iterations of the *MATS+* like test for PNPSF3 detection. The results of the comparison for various fault sizes (k) are presented in Table 14.

The background efficiency in the case when $k = 3$ and four iterations of the *MATS+* test

Table 12. Random background efficiency.

k/l	0	1	2	3	4
3	23.43%	41.38%	55.12%	65.64%	73.10%
4	12.11%	22.75%	32.11%	40.33%	47.55%
5	6.15%	11.93%	17.34%	22.43%	27.20%
k/l	5	6	7	8	9
3	79.86%	84.58%	88.19%	90.96%	93.08%
4	53.90%	59.49%	64.39%	68.70%	72.49%
5	31.68%	35.88%	39.83%	43.53%	47.01%

Table 13. Random pairs of background efficiency.

k/l	0	1	2	3	4
3	0.25%	43.75%	57.81%	68.36%	76.26%
4	12.50%	23.44%	33.01%	41.38%	48.71%
5	6.25%	12.11%	17.60%	22.75%	27.58%
k/l	5	6	7	8	9
3	82.20%	86.65%	89.98%	92.49%	94.36%
4	55.12%	60.73%	65.64%	69.93%	73.69%
5	32.11%	36.35%	40.33%	44.06%	47.55%

is presented in Fig. 7.

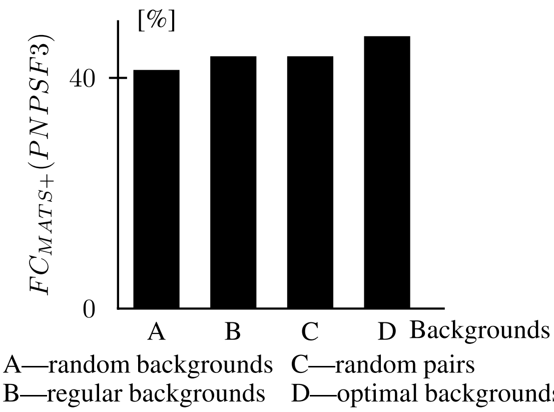

Fig. 7. Multirun test comparison.

The results in Table 14 and Fig. 7 proved that optimal backgrounds allow us to achieve the best results for four iteration of the *MATS+* like test.

6. Conclusions

There are many methods for generating backgrounds in multirun memory testing. The most popular and optimal ones are presented and compared in this paper. From the results we can conclude that for a multirun test procedure which consist up to four iterations the best results will be achieved with optimal backgrounds. If we want to run the test more than four times, the best results will be achieved with regular backgrounds or random pairs of

Table 14. Background efficiency for four iterations of the MATS+ test.

Backgrounds/k	3	4	5
Random	41.38%	22.75%	11.92%
Regular	43.75%	23.44%	12.11%
Random pairs	43.75%	23.43 %	12.10%
Optimal	47.23%	24.54%	12.42%

backgrounds (compare Tables 11, 12 and 13). Random pairs of backgrounds are especially suitable for periodic testing in systems where the contents of memory undergo constant changes.

In our investigation we focused only on simple March tests like the *MATS+* test. This is because, according to weighted fault coverage measure for March tests (Mrozek *et al.*, 2008), it is not necessary to use complex tests to achieve good results in multibackground testing. Weighted fault coverage takes into consideration not only the fault coverage of the test but its complexity, too. So in multirun memory testing the simplest tests allow us to achieve as good results (or better) as complex ones with the same total complexity of the test procedure.

References

Cheng, K.-L., Tsai, M.-F. and Wu, C.-W. (2002). Neighborhood pattern sensitive fault testing and diagnostics for random access memories, *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems* **21**(11): 1328–1336.

Cockburn, B. F. (1995). Deterministic tests for detecting scrambled pattern-sensitive faults in RAMs, *MTDT '95: Proceedings of the 1995 IEEE International Workshop on Memory Technology, Design and Testing, Washington, DC, USA*, pp. 117–122.

Franklin, M. and Saluja, K. K. (1996). Testing reconfigured RAM's and scrambled address RAM's for pattern sensitive faults, *IEEE Transactions on CAD of Integrated Circuits and Systems* **15**(9): 1081–1087.

Goor, A. J. v. d. (1991). *Testing Semiconductor Memories: Theory and Practice*, John Wiley & Sons, Chichester.

Hayes, J. P. (1975). Detection of pattern-sensitive faults in random-access memories, *IEEE Transactions on Computers* **24**(2): 150–157.

Hayes, J. P. (1980). Testing memories for single-cell pattern-sensitive faults, *IEEE Transactions on Computers* **29**(3): 249–254.

Huang, Y. and Li, J. F. (2006). Testing active neighborhood pattern-sensitive faults of ternary content addressable memories, *European Test Symposium, Southampton, UK*, pp. 55–62.

Karpovsky, M. G., Goor, A. J. v. d. and Yarmolik, V. N. (1995). Pseudo-exhaustive word-oriented DRAM testing, *EDTC '95: Proceedings of the 1995 European Conference on Design and Test, Washington, DC, USA*, p. 126.

Karpovsky, M. G. and Yarmolik, V. N. (1994). Transparent memory testing for pattern-sensitive faults, *Proceedings of the IEEE International Test Conference on TEST: The Next 25 Years, Washington, DC, USA*, pp. 860–869.

Krasniewski, A. (2008). Concurrent error detection for combinational logic blocks implemented with embedded memory blocks of FPGAs, *DDECS'08: Proceedings of the IEEE International Workshop on Design and Diagnostics of Electronic Circuits and Systems, Bratislava, Slovakia*, pp. 74–79.

Mrozek, I. and Yarmolik, V. N. (2008b). Optimal backgrounds selection for multi run memory testing, *DDECS'08: Proceedings of the IEEE International Workshop on Design and Diagnostics of Electronic Circuits and Systems, Bratislava, Slovakia*, pp. 332–338.

Mrozek, I. and Yarmolik, V. N. (2008a). MATS+ transparent memory test for pattern sensitive fault detection, *MIXDES'08: Proceedings of the 15th International Conference on Mixed Design of Integrated Circuits and Systems, Poznań, Poland*, pp. 493–498.

Mrozek, I., Yarmolik, V. N. and Buslowska, E. (2008). Multiple run memory testing for PSF detection, *EWDTS '08: Proceedings of the IEEE East-West Design and Test Symposium, Lviv, Ukraine*, pp. 125–130.

Nicolaidis, M. (1996). Theory of transparent BIST for RAMs, *IEEE Transactions on Computing* **45**(10): 1141–1156.

Niggemeyer, D., Redeker, M. and Otterstedt, J. (1998). Integration of non-classical faults in standard march tests, *MTDT '98: Proceedings of the 1998 IEEE International Workshop on Memory Technology, Design and Testing, San Jose, CA, USA*, p. 91.

Sokol, B. and Yarmolik, S. V. (2006). Address sequences for march tests to detect pattern sensitive faults, *DELTA '06: Proceedings of the Third IEEE International Workshop on Electronic Design, Test and Applications, Kuala Lumpur, Malaysia*, pp. 354–360.

Sosnowski, J. (2007). Improving software based self-testing for cache memories, *Proceedings of the 2nd International Design and Test Workshop, 2007, Cairo, Egypt*, pp. 49–54.

Tubbs, J. D. (1989). A note on binary template matching, *Pattern Recognition* **22**(4): 359–366.

Voyiatzis, I. (2006). Accumulator-based compression in symmetric transparent RAM BIST, *DTIS'06: Proceedings of the International Conference on Design and Test of Integrated Systems in Nanoscale Technology, Tunis, Tunisia*, pp. 273–278.

Yarmolik, S. (2008). Address sequences and backgrounds with different Hamming distances for multiple run March tests, *International Journal of Applied Mathematics and Computer Science* **18**(3): 329–339, DOI: 10.2478/v10006-008-0030-y.

Yarmolik, S. V. and Mrozek, I. (2007). Multi background memory testing, *MIXDES07: Proceedings of the 14th International Conference on Mixed Design of Integrated Circuits and Systems, Ciechocinek, Poland*, pp. 511–516.

Zhang, B. and Srihari, S. (2003). Binary vector dissimilarity measures for handwriting identification, *Proceedings of the SPIE, Document Recognition and Retrieval X, Santa Clara, CA, USA*, pp. 155–166.

Zorian, Y. (2002). Embedded memory test and repair: Infrastructure IP for SOC yield, *ITC '02: Proceedings of the 2002 IEEE International Test Conference, Washington, DC, USA*, p. 340.

Ireneusz Mrozek received his M.Sc. and Ph.D. degrees in computer science in 1994 and 2004, respectively. Since 1994 he has been employed at the Faculty of Computer Science of Białystok Technical University (Poland). His main research interests include the area of diagnostic testing of embedded memories. Particularly, he focuses on transparent tests for RAM as well as the application of these in the BIST or BISR schemes.

Received: 16 April 2009
Revised: 4 September 2009