

МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ

УДК 004.33.054

С.В. Ярмолик, В.Н. Ярмолик

ИТЕРАТИВНЫЕ ПОЧТИ ПСЕВДОИЩЕРПЫВАЮЩИЕ
ВЕРОЯТНОСТНЫЕ ТЕСТЫ

Предлагается метод построения почти псевдоисчерпывающих вероятностных тестов, характеризующихся большей полнотой покрытия в сравнении с вероятностными и детерминированными тестами при такой же их временной сложности. Данные тесты позволяют генерировать всевозможные двоичные комбинации на любых k из N входов тестируемого цифрового устройства. Показывается их эффективность для малых значений k и небольшого числа итераций. Приводятся экспериментальные исследования для случая тестирования ОЗУ.

Введение

Псевдоисчерпывающие тесты (*Pseudoexhaustive Testing*) [1–4], которые являются развитием исчерпывающих тестов (*Exhaustive Testing*) [5], первоначально рассматривались только для случая тестирования комбинационных цифровых устройств и оперативных запоминающих устройств (*Hardware Testing*). Они являются реальной альтернативой исчерпывающим тестам и основываются на формировании множества тестовых наборов $B(N, k)$, обеспечивающих 2^k возможных двоичных комбинаций на любых k из N входов тестируемого комбинационного устройства или в любых k из N ячеек запоминающего устройства. Характерной особенностью псевдоисчерпывающих тестов $B(N, k)$ является то, что их сложность $O(B(N, k))$ существенно меньше по сравнению с исчерпывающими тестами. Так, для теста $B(6, 2) = \{000000, 000011, 011100, 101101, 110110, 111011\}$, приведенного в [6], сложность (количество тестовых наборов) $O(B(6, 2))$ псевдоисчерпывающего теста $B(6, 2)$ будет равна 6, что заметно меньше, чем сложность исчерпывающего теста для $N = 6$, равная $2^6 = 2^6 = 64$. Для общего случая сложность $O(B(N, k))$ псевдоисчерпывающего теста $B(N, k)$ оценивается неравенством $2^k \leq O(B(N, k)) \leq 2^N$.

Дальнейшее развитие псевдоисчерпывающие тесты получили в рамках задачи комбинаторных стратегий тестирования сложных вычислительных систем, включая тестирование программного обеспечения [7, 8]. В общем случае при отсутствии дополнительной информации о тестируемой вычислительной системе эта задача решается разбиением возможных значений параметров на конечное число групп и генерацией всевозможных комбинаций значений параметров для всех групп. Для построения этих комбинаций используются покрывающие массивы, дающие минимально возможные множества тестовых векторов, которые перебирают все сочетания пар, троек или другого числа k из N значений отдельных параметров [9].

В случае тестов запоминающих устройств типа ОЗУ в качестве покрывающих массивов рассматриваются множества векторов начальных состояний ОЗУ, позволяющие эффективно обнаруживать сложные обобщенные модели неисправностей для заданных, как правило маршевых, тестов ОЗУ [10, 11]. Чаще всего как обобщенные модели неисправностей ОЗУ рассматриваются кодочувствительные неисправности (*Pattern Sensitive Faults – PSF*), затрагивающие несколько ячеек ОЗУ [10, 11].

Получение псевдоисчерпывающих тестов с минимальной сложностью $O(B(N, k))$ и их генерация представляют собой весьма трудоемкую задачу, которая зависит от величин N и k [1, 2]. Для малых величин N и k значения $O_{\min}(B(N, k))$ или их оценки приведены в [4]. Для фиксированных значений k и больших величин N известны только граничные оценки величины $O_{\min}(B(N, k))$, удовлетворяющие неравенству $2^{k-1} \log_2 N \leq O_{\min}(B(N, k)) \leq k 2^k (\log_2 e)^{-1} \log_2 N$ [4]. Несмотря на большое многообразие методов формирования псевдоисчерпывающих тестов, до настоящего времени не существует радикального решения получения тестов $B(N, k)$ для современных вычислительных систем и программных приложений с реальной временной сложностью $O(B(N, k))$ [7–10].

Применяя тест $B_S(N, k)$, состоящий из $\lceil \log_2 N \rceil + 1$ пары векторов $(B_0, B_1), (B_2, B_3), \dots, (B_{2l}, B_{2l+1})$, и предполагая, что $k \ll N$ и $N - k \approx N$, будет достигнута максимальная полнота покрытия теста $C(B_S(N, k))$, которая будет вычисляться как

$$C(B_S(N, k)) = \left(1 - \left(\frac{2^{k-1} - 1}{2^{k-1}} \right)^{\lceil \log_2 N \rceil + 1} \right) 100 \% . \quad (1)$$

Для $N = 2^m$ и $k \in \{2, 3, 4, 5\}$ полнота покрытия $C(B_S(N, k))$ теста $B_S(N, k)$ приведена в табл. 2.

Таблица 2

Полнота покрытия $C(B_S(N, k))$ для различных значений N и k , %

| $N = 2^m$ | 2^5 | 2^{10} | 2^{15} | 2^{20} | 2^{25} | 2^{30} | 2^{35} |
|-----------|-------|----------|----------|----------|----------|----------|----------|
| $k = 2$ | 98,44 | 99,95 | 99,99 | 99,99 | 99,99 | 99,99 | 99,99 |
| $k = 3$ | 82,20 | 95,77 | 98,99 | 99,76 | 99,92 | 99,98 | 99,99 |
| $k = 4$ | 63,87 | 76,98 | 88,19 | 93,94 | 96,89 | 98,40 | 99,18 |
| $k = 5$ | 32,10 | 49,16 | 64,39 | 74,21 | 81,33 | 86,47 | 90,20 |

Из табл. 2 следует, что, например, при $q = 2(m + 1) = 2(30 + 1) = 62$ итерациях теста $B_S(N, k)$ оказывается возможным достигнуть почти псевдоисчерпывающей полноты покрытия $C(B_S(N, k)) = 99,98\%$ для $k = 3$. Более детальный анализ данных, приведенных в табл. 2, позволяет сделать следующие выводы. Значение полноты покрытия $C(B_S(N, k))$ теста $B_S(N, k)$, которое при фиксированном значении $N = 2^m$ является предельно достижимым для $B_S(N, k)$, с ростом $N = 2^m$ также увеличивается. С другой стороны, максимальная полнота покрытия является убывающей функцией по отношению к величине k . В то же время сложность $O(B_S(N, k))$ теста $B_S(N, k)$ при $N = 2^m$ равняется $2(m + 1)$ и является минимальной для подобных тестов [1, 4, 12].

В сравнении с детерминированными псевдоисчерпывающими тестами вероятностные тесты позволяют достичь полноты покрытия $C(B_R(N, k))$, сколь угодно близкой к величине 100 %, для заданного k за счет заметного увеличения сложности теста. Использование l произвольных (случайных) векторов позволяет достичь полноты покрытия, вычисляемой согласно выражению

$$C(B_R(N, k)) = \left(1 - \left(1 - \frac{1}{2^k} \right)^l \right) 100 \% . \quad (2)$$

Очевидно, что максимальная полнота покрытия, равная 100 %, достижима только для $l \rightarrow \infty$, а реальные ее величины, близкие к 100 %, – для больших величин l . Это следует из результатов, представленных в табл. 3 для тех же значений N и $l = 2(m + 1)$, что и для $B_S(N, k)$ (см. табл. 2).

Таблица 3

Полнота покрытия $C(B_R(N, k))$ для различных значений N и k , %

| $N = 2^m$ | 2^5 | 2^{10} | 2^{15} | 2^{20} | 2^{25} | 2^{30} | 2^{35} |
|-----------|-------|----------|----------|----------|----------|----------|----------|
| $k = 2$ | 96,83 | 99,82 | 99,98 | 99,99 | 99,99 | 99,99 | 99,99 |
| $k = 3$ | 79,85 | 94,70 | 98,60 | 99,63 | 99,90 | 99,97 | 99,99 |
| $k = 4$ | 53,90 | 75,82 | 87,32 | 93,35 | 96,51 | 98,40 | 99,18 |
| $k = 5$ | 31,68 | 50,26 | 63,79 | 73,63 | 80,81 | 86,03 | 89,83 |

Сравнительный анализ результатов, представленных в табл. 2 и 3, показывает большую эффективность детерминированных псевдоисчерпывающих тестов при том же фиксированном количестве тестовых векторов l . В то же время с увеличением их числа различие в эффективности уменьшается. Однако применение вероятностного подхода позволяет достичь любой полноты покрытия $C(B_R(N, k))$, сколь угодно близкой к 100 %, для заданного k за счет увеличения количества l тестовых векторов.

2. Итеративные почти псевдоисчерпывающие вероятностные тесты

Известные вероятностные тесты, приведенные в предыдущем разделе, можно рассматривать как повторяющиеся тесты $B_R(N, k) = \{B_0, B_1, B_2, \dots, B_{q-1}\}$, состоящие из q итераций. В каждой итерации используется один случайный вектор $B_i = b_0b_1b_2\dots b_{N-2}b_{N-1}$, где $b_s \in \{0, 1\}$ для $s \in \{0, 1, \dots, N-1\}$ представляют собой равновероятную двоичную цифру.

В качестве основной идеи итеративных почти псевдоисчерпывающих тестов $B_I(N, k, r)$ предлагается применять вероятностные тесты, в которых в каждой итерации случайный вектор используется более чем один раз ($r > 1$). В простейшем случае итерация будет состоять в применении случайного вектора B_i и его инверсного значения $\bar{B}_i = \bar{b}_0\bar{b}_1\bar{b}_2\dots\bar{b}_{N-2}\bar{b}_{N-1}$. Исходя из утверждения 1, последующее использование инверсных значений бит исходного случайного вектора является оптимальным с точки зрения обеспечения максимальной полноты покрытия [10]. В общем случае каждая итерация состоит из $r \geq 2$ тестовых векторов, первый из которых формируется случайным образом, а $r-1$ остальных – в соответствии с некоторым детерминированным алгоритмом. Предполагая, что $rq = l$ и эффективность одной итерации оценивается величиной $0 < p(N, k, r) < 1$, полнота покрытия $C(B_I(N, k, r))$ произвольного итерационного теста оценивается соотношением

$$C(B_I(N, k, r)) = (1 - (1 - p(N, k, r))^q) 100 \%. \quad (3)$$

Тогда эффективность итеративного псевдоисчерпывающего вероятностного теста $B_I(N, k, r)$ по сравнению с вероятностным тестом $B_R(N, k)$ оценивается выражением

$$\Delta = C(B_I(N, k, r)) - C(B_R(N, k)) = \left(\left(\left(1 - \frac{1}{2^k} \right)^r \right)^q - (1 - p(N, k, r))^q \right) 100 \%. \quad (4)$$

Величина $((1 - 1/2^k)^r)^q - (1 - p(N, k, r))^q$ с ростом q стремится к нулю, так как $(1 - 1/2^k)^r < 1$ и $(1 - p(N, k, r)) < 1$. Таким образом, эффективность произвольного итерационного псевдоисчерпывающего вероятностного теста $B_I(N, k, r)$ в сравнении с вероятностным для малых значений q будем максимальной. В то же время увеличение значения r приводит к снижению эффективности одной итерации в сравнении с вероятностным тестом, состоящим из r случайных двоичных векторов. В качестве оптимальных значений предлагается использовать $r = 2, 3$ и 4 , для которых известны оптимальные сочетания, состоящие из двух, трех и четырех последовательных векторов, удовлетворяющих утверждению 1 [10]. Для $r = 2$ оптимальными векторами B_i и B_j являются инверсные векторы $B_j = \bar{B}_i$, для которых $HD(B_i, B_j) = N$. При $r = 3$ и 4 максимальное минимальное расстояние Хэмминга $HD(B_i, B_j) = 2N/3$ между любыми двумя векторами B_i, B_j из трех ($r = 3$) или из четырех ($r = 4$) [10].

Значение $p(N, k, r)$ для $r = 3$ и 4 при больших величинах N и $k \ll N$ не зависит от величины N ($p(N, k, r) = p(k, r)$) и может быть оценено следующими соотношениями $p(k, r)$:

$$p(k, 3) \approx \frac{1}{2^k} + \frac{2}{3^k} - \frac{1}{2^k 3^k} + \frac{2}{2^k 3^k} \sum_{i=1}^{k-1} 2^{k-i} \binom{k}{i};$$

$$p(k, 4) \approx \frac{1}{2^k} + \frac{1}{3^{k-1}} - \frac{1}{2^k 3^{k-1}} + \frac{1}{2^k 3^{k-1}} \sum_{i=1}^{k-1} 2^{k-i} \binom{k}{i}. \quad (5)$$

Значения $C(B_I(N, k, 4))$ для $r = 4$, одной итерации ($q = 1$) и конкретных значений k приведены в табл. 4. Для сравнения в этой таблице также даются значения эффективности $C(B_S(N, k))$ стандартного теста $B_S(N, k)$, состоящего только из первых четырех векторов (см. табл. 1). В последней строке приведена оценка эффективности $C(B_R(N, k))$ вероятност-

ного теста $B_R(N, k)$, состоящего также из четырех тестовых векторов. Таким образом, $O(B_I(N, k, 4)) = O(B_S(N, k)) = O(B_R(N, k)) = 4$.

Таблица 4

Оценка полноты покрытия тестов $B_I(N, k, 4)$, $B_S(N, k)$ и $B_R(N, k)$ для $q = 1$ и различных k , %

| k | 2 | 3 | 4 | 5 | 6 |
|-------------------|-------|-------|-------|-------|------|
| $C(B_I(N, k, 4))$ | 83,33 | 47,22 | 25,00 | 12,42 | 6,23 |
| $C(B_S(N, k))$ | 75,00 | 43,75 | 23,43 | 12,10 | 6,15 |
| $C(B_R(N, k))$ | 68,32 | 41,38 | 22,75 | 11,92 | 6,11 |

В табл. 5 приведены значения $C(B_I(N, k, 4))$, $C(B_S(N, k))$ и $C(B_R(N, k))$ для $k = 2, 3$ и 4.

Таблица 5

Оценка полноты покрытия тестов $B_I(N, k, 4)$, $B_S(N, k)$ и $B_R(N, k)$ для $k = 2, 3$ и 4, %

| q | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-------------------|------|------|------|------|------|------|------|------|
| $C(B_I(N, 2, 4))$ | 83,3 | 97,2 | 99,5 | 99,9 | 99,9 | 99,9 | 99,9 | 99,9 |
| $C(B_S(N, 2))$ | 75,0 | 93,7 | 98,4 | 99,6 | 99,9 | 99,9 | 99,9 | 99,9 |
| $C(B_R(N, 2))$ | 68,3 | 89,9 | 96,8 | 98,9 | 99,6 | 99,8 | 99,9 | 99,9 |
| $C(B_I(N, 3, 4))$ | 47,2 | 72,1 | 85,3 | 92,2 | 95,9 | 97,8 | 98,8 | 99,4 |
| $C(B_S(N, 3))$ | 43,7 | 68,3 | 82,2 | 89,6 | 94,4 | 96,8 | 98,2 | 98,9 |
| $C(B_R(N, 3))$ | 41,3 | 65,6 | 79,8 | 88,9 | 93,1 | 95,9 | 97,6 | 98,5 |
| $C(B_I(N, 4, 4))$ | 25,0 | 43,7 | 57,8 | 68,4 | 76,3 | 82,2 | 86,6 | 89,9 |
| $C(B_S(N, 4))$ | 23,4 | 41,4 | 55,1 | 65,6 | 73,7 | 79,8 | 84,6 | 88,2 |
| $C(B_R(N, 4))$ | 22,7 | 40,3 | 53,9 | 64,4 | 72,5 | 78,7 | 83,6 | 87,3 |

Анализ приведенных результатов свидетельствует об эффективности применения итерационных почти псевдоисчерпывающих вероятностных тестов $B_I(N, k, r)$ для малых значений k ($k = 2, 3$ и 4). Для больших величин k эффективность почти псевдоисчерпывающих вероятностных тестов также является более высокой в сравнении с детерминированными $B_S(N, k)$ и вероятностными $B_R(N, k)$ тестами, однако она не настолько выше, как для малых значений k . Заметное преимущество почти псевдоисчерпывающих вероятностных тестов $B_I(N, k, r)$ достигается и для малого количества итераций q , а с их ростом преимущество таких тестов снижается.

3. Применение почти псевдоисчерпывающих тестов для тестирования ОЗУ

Как показано в разд. 2, каждая итерация почти псевдоисчерпывающего вероятностного теста $B_I(N, k, r)$ состоит из $r \geq 2$ тестовых векторов, первый из которых формируется случайным образом, а $r - 1$ остальных – в соответствии с некоторым алгоритмом, который обеспечивает выполнение условия, сформулированного в утверждении 1. Для $r = 3$ (4) на основании исходного случайного вектора B_i необходимо сформировать два (три) новых вектора таким образом, чтобы максимальное минимальное расстояние Хэмминга $HD(B_i, B_j)$ между любыми двумя векторами B_i и B_j из трех (четырех) равнялось $2N/3$ [10]. Согласно утверждению 1 для $r = 3$ множество таких векторов B_i, B_j и B_l имеет вид

$$\begin{aligned}
 B_i &= b_0 b_1 b_2 \dots b_{N/3-1} b_{N/3} b_{N/3+1} \dots b_{2N/3-1} b_{2N/3} b_{2N/3+1} \dots b_{N-1}; \\
 B_j &= \bar{b}_0 \bar{b}_1 \bar{b}_2 \dots \bar{b}_{N/3-1} \bar{b}_{N/3} \bar{b}_{N/3+1} \dots \bar{b}_{2N/3-1} b_{2N/3} b_{2N/3+1} \dots b_{N-1}; \\
 B_l &= b_0 b_1 b_2 \dots b_{N/3-1} \bar{b}_{N/3} \bar{b}_{N/3+1} \dots \bar{b}_{2N/3-1} \bar{b}_{2N/3} \bar{b}_{2N/3+1} \dots \bar{b}_{N-1}.
 \end{aligned} \tag{6}$$

Исходный случайный вектор $B_i = b_0 b_1 b_2 \dots b_{N-2} b_{N-1}$, где $b_s \in \{0, 1\}$ представляет собой равновероятную двоичную цифру, с применением операции инвертирования преобразуется в B_j и B_l . Для $r = 4$ множество таких векторов B_i, B_j, B_l и B_o получается аналогичным образом [10]:

$$\begin{aligned}
 B_i &= b_0 b_1 b_2 \dots b_{N/3-1} b_{N/3} b_{N/3+1} \dots b_{2N/3-1} b_{2N/3} b_{2N/3+1} \dots b_{N-1}; \\
 B_j &= \bar{b}_0 \bar{b}_1 \bar{b}_2 \dots \bar{b}_{N/3-1} \bar{b}_{N/3} \bar{b}_{N/3+1} \dots \bar{b}_{2N/3-1} b_{2N/3} b_{2N/3+1} \dots b_{N-1}; \\
 B_l &= b_0 b_1 b_2 \dots b_{N/3-1} \bar{b}_{N/3} \bar{b}_{N/3+1} \dots \bar{b}_{2N/3-1} \bar{b}_{2N/3} \bar{b}_{2N/3+1} \dots \bar{b}_{N-1}; \\
 B_o &= \bar{b}_0 \bar{b}_1 \bar{b}_2 \dots \bar{b}_{N/3-1} b_{N/3} b_{N/3+1} \dots b_{2N/3-1} \bar{b}_{2N/3} \bar{b}_{2N/3+1} \dots \bar{b}_{N-1}.
 \end{aligned} \tag{7}$$

Например, для $N=9$ и $B_i=000000000$ получим, что $B_j=111111000$, $B_l=000111111$ и $B_o=111000111$.

В случае тестирования ОЗУ рассматриваются модели неисправностей, для обнаружения которых необходимо формирование всевозможных двоичных комбинаций в произвольных $k-1$ из N двоичных ячеек ОЗУ [3, 6, 10–13]. При этом значение k , как правило, не превышает 5 [14]. Объемы современных ОЗУ неуклонно растут и превышают 10^9 , что накладывает жесткие временные ограничения на время их тестирования. Очевидным решением для тестирования ОЗУ является использование почти псевдоисчерпывающих вероятностных тестов $B_i(N, k, r)$, которые, как было показано в предыдущем разделе, характеризуются высокой эффективностью для малых значений k и $N \gg k$. Почти псевдоисчерпывающие вероятностные тесты целесообразно использовать для формирования оптимальных начальных состояний ячеек ОЗУ в рамках реализации неразрушающих маршевых тестов [15].

В качестве начальных состояний ОЗУ при реализации неразрушающих маршевых тестов с целью уменьшения сложности процедуры тестирования используется их текущее состояние $B_i = b_0 b_1 b_2 \dots b_{N-2} b_{N-1}$, что позволяет уменьшить сложность процедуры тестирования на N циклов обращения к памяти, необходимых для записи случайного вектора B_i . Тогда для $r=3$ сложность процедур перехода от текущего состояния B_i к B_j , от B_j к B_l и возврат к исходному состоянию (переход от B_l к B_i) во всех случаях равняется $4N/3$ циклов обращения к памяти для операций инвертирования в соответствии с (6). Здесь инвертирование содержимого ячейки памяти состоит из операций чтения и записи инверсного значения. Суммарная сложность формирования векторов (6) равняется $4N$, а векторов (7) – $16N/3$ циклов обращения к ОЗУ. Очевидными недостатками рассмотренной процедуры применения почти псевдоисчерпывающих вероятностных тестов $B_i(N, k, r)$ являются снижение обнаруживающей способности тестов ОЗУ из-за их зависимости от текущего состояния ОЗУ $B_i = b_0 b_1 b_2 \dots b_{N-2} b_{N-1}$ и временная сложность итераций, равная $4N$ для $r=3$ и $16N/3$ для $r=4$. Снижение эффективности теста связано с тем, что в среднем только 80 % содержимого ОЗУ ЭВМ изменяется при изменении решаемой задачи [16]. При неизменном начальном состоянии B_i для двух последовательных итераций использование процедур (6) и (7) не приведет к формированию новых двоичных комбинаций.

Уменьшение временной сложности почти псевдоисчерпывающих вероятностных тестов $B_i(N, k, r)$ достигается путем их применения для фиксированных блоков ОЗУ меньшей размерности, равной $M = 2^m - 1 < N$. Для упрощения реализации процедур (6) и (7) предположим, что $m = 2^w$, где w – целое число. Таким образом, $m = 2, 4, 16, 256, \dots$, а значение $2^m - 1$ можно представить в виде

$$2^m - 1 = 2^{2^w} - 1 = (2^1 - 1)(2^1 + 1)(2^2 + 1)(2^4 + 1)(2^{16} + 1) \dots (2^{2^{w-1}} + 1). \tag{8}$$

Тогда для любого $M = 2^m - 1$ значения $M/3$ и $2M/3$ определяются как

$$M/3 = \prod_{i=1}^{w-1} (2^{2^i} + 1); \quad 2M/3 = 2 \prod_{i=1}^{w-1} (2^{2^i} + 1). \tag{9}$$

Так, например, для $M = 2^m - 1 = 2^4 - 1 = 15$ в соответствии с (9) получим $M/3 = 5$ и $2M/3 = 10$.

Уменьшение размерности блока ОЗУ позволяет уменьшить сложность его тестирования и обеспечить режим прерываемого тестирования с использованием маршевых тестов [15, 16]. Следует, однако, отметить, что суммарная сложность тестирования всего ОЗУ останется неизменной.

Для уменьшения влияния текущего состояния блока ОЗУ $B_i = b_0b_1b_2 \dots b_{M-2}b_{M-1}$ на последующие итерации теста целесообразно модифицировать процедуры (6) и (7) таким образом, чтобы при переходе от вектора B_i к вектору B_j инвертировалось содержимое $b_s \in \{0, 1\}$, $s \in \{0, 1, \dots, M-1\}$, произвольных $2M/3$ ячеек блока ОЗУ размерности M бит. При переходе от B_j к B_l половина из ранее измененных значений b_s принимала свое исходное состояние, в то время как оставшиеся $M/3$ бит исходного блока изменили свое состояние на противоположное, и так далее для всех переходов в соответствии с (6) и (7). Основой подобной модификации может быть разбиение блока $B_i = b_0b_1b_2 \dots b_{M-2}b_{M-1}$ из M ячеек ОЗУ на три произвольных блока B_{i1} , B_{i2} и B_{i3} , определяемых индексами (адресами) $s \in \{0, 1, \dots, M-1\}$ ячеек b_s , каждый из которых состоит из $M/3$ ячеек. Для этих целей целесообразно использование генераторов M -последовательностей, формируемых на основании примитивных порождающих полиномов $\varphi(x) = 1 \oplus \alpha_1x^1 \oplus \alpha_2x^2 \oplus \dots \oplus \alpha_{m-1}x^{m-1} \oplus \alpha_mx^m$, где $\alpha_i \in \{0, 1\}$ и зависит от вида порождающего полинома.

В общем случае для формирования почти псевдоисчерпывающих вероятностных тестов $B_f(N, k, r)$ первоначально необходимо задание размерности $M = 2^m - 1 < N$ блока ОЗУ и его начального адреса M_0 . В соответствии с (9) вычисляются значения $M/3$ и $2M/3$. Задаются q примитивных порождающих полиномов $\varphi(x)$, $\deg \varphi(x) = m$. Необходимо отметить, что для заданного m существует $\Phi(2^m - 1)/m$ примитивных полиномов, где Φ – функция Эйлера [15]. С учетом принятых допущений процедура генерирования почти псевдоисчерпывающих вероятностных тестов $B_f(N, k, r)$ для $r = 3$ имеет следующий вид.

Входные данные: размерность $M = 2^m - 1$ блока ОЗУ и его начальный адрес M_0 , значение величин $M/3$ и $2M/3$, количество q итераций теста, примитивные полиномы $\varphi(x) \in \{\varphi_1(x), \varphi_2(x), \dots, \varphi_q(x)\}$ степени m .

1. Задается $i = 1$.

2. В качестве первого вектора $B_i = b_0b_1b_2 \dots b_{M-2}b_{M-1}$ теста $B_f(N, k, 3)$ принимается текущее состояние выбранного блока ОЗУ, где b_0 является содержимым ячейки ОЗУ с адресом M_0 , b_1 – ячейки $M_0 + 1$ и b_{M-1} – ячейки $M_0 + M - 1$.

3. В соответствии с порождающим полиномом $\varphi_i(x)$ строится генератор M -последовательности. Задается его начальное состояние A_1 .

4. Осуществляется переход к вектору B_j путем инвертирования $2M/3$ бит исходного вектора B_i . Адреса ячеек выбранного блока определяются как $M_0 - 1 + A_s$, где $A_s \in \{1, 2, 3, \dots, 2^m - 1\}$, $s = 1, 2, 3, \dots, 2M/3$, – текущее состояние генератора M -последовательности.

5. Аналогично, как и в п. 4, выполняется переход от вектора B_j к вектору B_l , при этом начальным состоянием генератора будет следующее его состояние – $A_{2M/3+1}$.

6. Выполняется возврат к текущему состоянию блока ОЗУ путем инвертирования его содержимого в $2M/3$ ячейках, начиная от адреса $M_0 - 1 + A_{M/3+1}$.

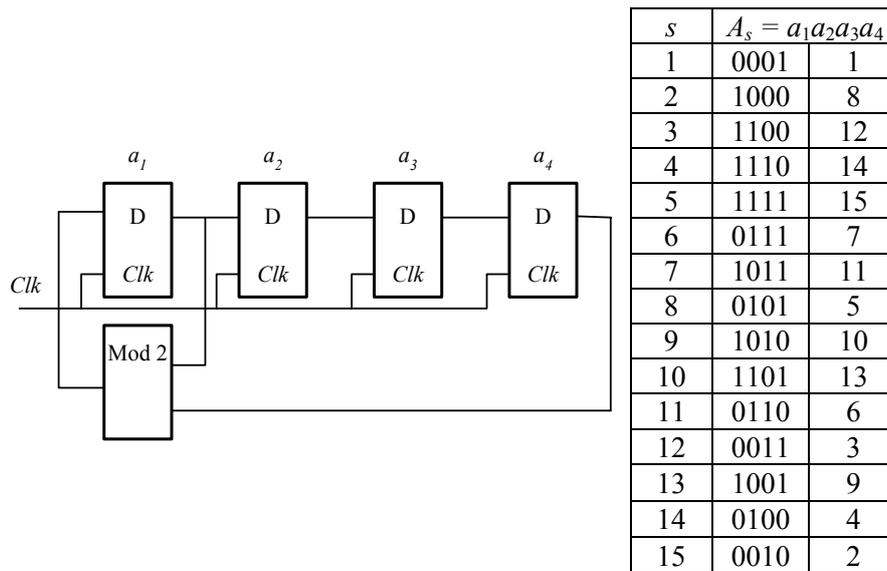
7. $i = i + 1$.

8. Если $i < q$, данная процедура повторяется, начиная со второго этапа.

Процедура генерирования $B_f(N, k, r)$ для $r = 4$ отличается только одним дополнительным переходом от вектора B_l к вектору B_o . Начальное состояние A_1 генератора M -последовательности может принимать произвольное значение из множества $\{1, 2, 3, \dots, 2^m - 1\}$.

4. Практическая реализация почти псевдоисчерпывающих тестов ОЗУ

Для $m = 4$ порождающий полином может иметь вид $\varphi(x) = 1 \oplus x^1 \oplus x^4$. Применение генератора M -последовательности на основе данного полинома (рисунок) позволяет разделить блок ОЗУ $B_i = b_0b_1b_2 \dots b_{14}b_{15}$, состоящий из $M = 15$ ячеек, на три блока: $B_{i1} = b_1b_8b_{12}b_{14}b_{15}$, $B_{i2} = b_7b_{11}b_5b_{10}b_{13}$ и $B_{i3} = b_6b_3b_9b_4b_2$, каждый из которых содержит $M/3 = 15/3 = 5$ ячеек.



Блок-схема генератора М-последовательности, формируемой в соответствии с полиномом $\varphi(x) = 1 \oplus x^1 \oplus x^4$, и временная диаграмма его состояний в двоичном и десятичном кодах

Рассмотрим процедуру генерирования $B_f(N, k, 3)$ для следующих входных данных: $m = 4$; $M = 2^m - 1 = 15$; $M_0 = 000 \dots 0$; $M/3 = 5$; $2M/3 = 10$; $q = 1$ и $\varphi_1(x) = 1 \oplus x^1 \oplus x^4$. Согласно ранее описанной процедуре получим:

1. Задается $i = 1$.
 2. Принимается текущее состояние выбранного блока ОЗУ $B_i = b_0b_1b_2 \dots b_{M-2}b_{M-1} = 000 \dots 00$.
 3. В соответствии с порождающим полиномом $\varphi_1(x) = 1 \oplus x^1 \oplus x^4$ строится генератор М-последовательности (см. рисунок) с начальным состоянием $A_1 = 0001$.
 4. Осуществляется переход к следующему вектору B_j путем инвертирования 10 бит исходного вектора $B_i = b_0b_1b_2b_3b_4b_5b_6b_7b_8b_9b_{10}b_{11}b_{12}b_{13}b_{14} = 0000000000000000$, $2M/3$ адреса ячеек задаются генератором М-последовательности и принимают значения 0, 7, 11, 13, 14, 6, 10, 4, 9, 12. Соответственно получим $B_j = 100010110111111$.
 5. Аналогично, как и в п. 4, выполняется переход от вектора $B_j = 100010110111111$ к вектору B_l , при этом начальным состоянием генератора будет $A_{2M/3+1} = 0110_{(2)} = 6_{(10)}$. Очередные $2M/3$ адреса имеют значения 5, 2, 8, 3, 1, 0, 7, 11, 13, 14. Соответственно получим $B_l = 011111101110100$.
 6. Выполняется возврат к текущему состоянию $B_i = 0000000000000000$ блока ОЗУ путем инвертирования его содержимого в $2M/3$ ячейках с адресами 6, 10, 4, 9, 12, 5, 2, 8, 3, 1.
 7. $i = q$, что свидетельствует об окончании теста $B_f(N, k, 3)$.
- Для тех же начальных условий $B_f(15, k, 4)$ будет состоять из векторов, представленных в табл. 6, в этой же таблице приведены тесты $B_S(15, k)$ и $B_R(15, k)$.

Таблица 6

Тесты $B_f(15, k, 4)$, $B_S(15, k)$ и $B_R(15, k)$

| $B_f(15, k, 4)$ | $B_S(15, k)$ | $B_R(15, k)$ |
|------------------|------------------|-----------------|
| 0000000000000000 | 0000000000000000 | 011110010110100 |
| 1000101101111111 | 1111111111111111 | 001011010001111 |
| 011111101110100 | 1111111000000000 | 110100011110010 |
| 111101011001011 | 0000000111111111 | 000111100101101 |

В табл. 7 представлены результаты моделирования тестов из табл. 6.

Таблица 7

Оценка полноты покрытия тестов $B_f(15, k, 4)$, $B_S(15, k)$ и $B_R(15, k)$ для $q = 1$ и различных k , %

| k | 2 | 3 | 4 | 5 | 6 |
|--------------------|-------|-------|-------|-------|------|
| $C(B_f(15, k, 4))$ | 85,71 | 48,35 | 24,86 | 12,49 | 6,25 |
| $C(B_S(15, k))$ | 76,66 | 45,00 | 24,03 | 12,33 | 6,23 |
| $C(B_R(15, k))$ | 72,33 | 43,51 | 23,53 | 12,19 | 6,19 |

Полученные результаты полностью согласуются с рассмотренными ранее теоретическими результатами. Несколько большие значения полноты покрытия во всех случаях объясняются малым значением N ($N = 15$), для которого невыполнение условия $k \ll N$ приводит к погрешностям в соотношениях (1)–(5).

Заключение

Представленные в работе процедуры формирования почти псевдоисчерпывающих вероятностных тестов $B_f(N, k, r)$ для $r = 2, 3$ и 4 могут быть применены к ОЗУ с целью выявления сложных кодочувствительных неисправностей. Теоретически и экспериментально было показано, что данные тесты обладают более высокой покрывающей способностью для PPSFk-неисправностей по сравнению с детерминированным $B_S(15, k)$ и вероятностными тестами $B_R(15, k)$ для небольших значений k и небольшого числа итераций q при одинаковых временных затратах и сложности теста. Таким образом, применение псевдоисчерпывающих вероятностных тестов $B_f(N, k, r)$ целесообразно для бит-ориентированных ОЗУ емкостью N бит при значениях $k \ll N$. При этом следует отметить достаточно легкую генерацию векторов псевдоисчерпывающих вероятностных тестов $B_f(N, k, r)$ с помощью генератора М-последовательностей.

Список литературы

1. Tang, D.T. Exhaustive Test Pattern Generation with Constant Weight Vectors / D.T. Tang, L.S. Woo // IEEE Transactions on Computers. – 1983. – Vol. C–32, № 12. – P. 1145–1150.
2. Tang, D.T. Iterative Exhaustive Pattern Generation for Logic Testing / D.T. Tang, C.L. Chen // IBM Journal Res. Develop. – 1984. – Vol. 28, № 2. – P. 212–219.
3. Karpovsky, M.G. Pseudo-Exhaustive Word-Oriented DRAM Testing / M.G. Karpovsky, V.N. Yarmolik, A.J. Goor // EUROTEST Conference : proc. Int. Conf., Paris, France, 6–9 March 1995 / IEEE Computer Society. – Washington, USA, 1995. – P. 126–132.
4. Levitin, L.B. Efficient Exhaustive Test Based on MDS Codes // L.B. Levitin, M.G. Karpovsky // IEEE Int. Symposium on Information Theory : proc. Int. Conf. / IEEE Computer Society. – Ann Arbor, USA, 1986. – P. 64.
5. Barzilai, Z. Exhaustive Generation of Bit Pattern with Application to VLSI Self-Testing / Z. Barzilai, D. Coppersmith, A. Rozenberg // IEEE Transactions on Computers. – 1983. – Vol. C–31, № 2. – P. 190–194.
6. Karpovsky, M.G. Transparent Memory BIST / M.G. Karpovsky, V.N. Yarmolik // IEEE Workshop on Memory Technology Design and Testing : proc. Int. Conf., San Jose, CA, USA, 8–9 August 1994 / IEEE Computer Society. – San Jose, USA, 1994. – P. 106–111.
7. Grindal, M. Combination Testing Strategies / M. Grindal, J. Offutt, S.F. Andler // GMU Technical Report ISE-TR-04-05. – July, 2004. – 32 p.
8. Кулямин, В.В. Комбинаторика слов и построение тестовых последовательностей / В.В. Кулямин // Труды ИСП РАН. – 2004. – № 8 (1). – С. 25–40.
9. Hartman, A. Problems and Algorithms for Covering Arrays / A. Hartman, L. Raskin // Discrete Mathematics. – 2004. – № 284. – P. 149–156.
10. Yarmolik, S.V. Multi-Background Memory Testing / S.V. Yarmolik, I. Mrozek // Mixed Design of Integrated Circuits and Systems (MIXDES'07) : proc. of the 14th Int. Conf., Ciechocinek, Poland, 21–23 June 2007 / Technical University of Lodz. – Ciechocinek, Poland, 2007. – P. 511–516.

11. Yarmolik, S.V. Optimal Memory Address Seeds for Pattern Sensitive Faults Detection / S.V. Yarmolik, B. Sokol // IEEE Workshop on Design and Diagnostics of Electronic Circuits and Systems (DDECS'2006) : proc. Int. Conf., Prague, Czech Republic, 18–21 April 2006 / Czech Technical University Publishing House. – Prague, Czech Republic, 2006 – P. 220–221.

12. Das, D. Exhaustive and Near-Exhaustive Memory Testing Techniques and their BIST Implementations / D. Das, M.G. Karpovsky // Journal of Electronic Testing: Theory and Applications. – 1997. – Vol. 10. – P. 215–229.

13. Курбацкий, А.Н. Анализ эффективности многократных маршевых тестов запоминающих устройств / А.Н. Курбацкий, С.В. Ярмолик // Весн. Гродн. дзярж. ун-та імя Я. Купалы. Сер. 2. Матэматыка. Фізіка. Інфарматыка, вылічальная тэхніка і кіраванне. Біялогія. – 2008. – № 2 (68). – С. 103–107.

14. Adams, R.D. High Performance Memory Testing: Design Principles, Fault Modeling and Self-Test / R.D. Adams. – N.Y. : Kluwer Academic Publishers, 2003. – 247 p.

15. Неразрушающее тестирование запоминающих устройств / В.Н. Ярмолик [и др.]. – Минск : Бестпринт, 2005. – 230 с.

16. Mrozek, I. Problemu Funkcjonalnego Testowania Pamieci RAM / I. Mrozek, V. Yarmolik. – Bialystok : Oficyna Wydawnicza Politechniki Bialostockiej, 2009. – 264 str.

Поступила 03.02.10

*Белорусский государственный университет
информатики и радиоэлектроники,
Минск, П. Бровки, 6
e-mail: syarmolik@gmail.com,
yarmolik10ru@yahoo.com*

S.V. Yarmolik, V.N. Yarmolik

ITERATIVE NEAR PSEUDOEXHAUSTIVE RANDOM TESTS

The method for iterative near pseudoexhaustive random tests generation have been proposed. It was shown that for equal complexity the efficiency of iterative near pseudoexhaustive random tests are sufficiently higher to compare with deterministic and random tests. These tests allow getting all binary combinations for any arbitrary k out of N digital circuit inputs with high probability. It has been proven that efficiency of these tests is growing with decreasing the value k and number of test's iterations. The experimental data for the case of RAM testing as the validation data have been obtained.