Анализ и синтез маршевых тестов запоминающих устройств

Д. В. Деменковец, email: demenkovets@bsuir.by [1]
В. А. Леванцевич, email: lvn@bsuir.by [2]

[1,2] Белорусский государственный университет информатики и радиоэлектроники

Аннотация. *В статье показывается актуальность тестирования запоминающих устройств современных вычислительных систем. Представляются математические модели сложных кодочувствительных неисправностей типа PNPSFk, на базе классических маршевых тестов. Определяется понятие примитива, обеспечивающего условия активизации и обнаружения различных видов PNPSFk. Синтезируется маршевый тест March OP характеризующийся максимальной полнотой покрытия неисправностей PNPSFk по сравнению с известными маршевыми тестами, обеспечивающими такую же полноту покрытия сложных неисправностей запоминающих устройств.*

Ключевые слова: *тестирование вычислительных систем, неисправности памяти, кодочувствительные неисправности, маршевые тест.*

Введение

Проблема тестирования запоминающих устройств современных вычислительных систем, таких как встроенные системы (embedded systems), системы на кристалле (systems-on-a-chip) и сети на кристалле (nets-on-a-chip) является весьма актуальной задачей [1–3]. Для подобных вычислительных систем характерным является неуклонное увеличение емкости их запоминающих устройств, удельный вес которых достигает 94%, занимаемой системой площади кристалла [4, 5]. Наряду с увеличением емкости запоминающих устройств возрастают требования к их надежности, что достигается применением эффективных методов тестового диагностирования [6].

Тестирование запоминающих устройств с целью обнаружения различных их неисправностей предполагает запись в запоминающее устройство всевозможных его состояний и их считывание, что приводит к большой сложности теста, так как сложность самой тестовой процедуры пропорциональна величине 2^N, где N – емкость памяти в

битах. Поэтому в настоящее время широко применяются на практике и по-прежнему разрабатываются тесты, которые имеют существенно меньшую сложность, как правило, линейно зависящую от емкости памяти N [6]. Подобные тесты имеют общее название маршевые тесты (march tests) и доминируют в практических приложениях [7, 8].

В общем случае маршевый тест состоит из конечного числа маршевых элементов (march elements) [6–11]. В свою очередь, каждый маршевый элемент содержит символ, определяющий порядок формирования адресной последовательности (address sequence) для тестируемого запоминающего устройства, где символ $\Uparrow$ определяет последовательный перебор адресов памяти по возрастанию, символ $\Downarrow$ определяет последовательный перебор адресов по убыванию и сочетание двух символов $\Uparrow\Downarrow$ означает формирование адресов по убыванию либо по возрастанию. Следует отметить, что убывающая последовательность адресов $(\Downarrow)$ представляет собой последовательность, которая формируется в обратном порядке по сравнению с возрастающей последовательностью $(\Uparrow)$. Маршевый элемент содержит последовательность операций чтения $(read - r)$ и записи $(write - w)$, заключенных в круглые скобки и разделяемых точкой с запятой. Каждая операция представляет собой элемент из следующего набора: $r0-$ операция чтения содержимого запоминающего элемента (ячейки) с ожидаемым значением 0, $r1$ – операция чтения с ожидаемым значением 1, $w0$ – операция записи 0 в ячейку памяти, $w1$ – операция записи 1 в ячейку. Переход к следующей ячейке осуществляется только после выполнения всех операций в текущем маршевом элементе [6–11]. Для примера рассмотрим простейший маршевый тест Scan, который имеет следующий вид: $\{\Uparrow\Downarrow(w0);\Uparrow\Downarrow(r0);\Uparrow\Downarrow(w1);\Uparrow\Downarrow(r1)\}$. Данный тест состоит из четырех маршевых элементов и имеет сложность $4N$. Первый маршевый элемент $\Uparrow\Downarrow(w0)$ называется фазой инициализации (initialization phase), применяемой для записи начального состояния (background) запоминающего устройства. Остальные три фазы теста $\Uparrow\Downarrow(r0), \Uparrow\Downarrow(w1), \Uparrow\Downarrow(r1)$ являются основными фазами, обеспечивающими обнаруживающую способность данного теста. Таким образом, элементы маршевого теста, а именно, их вид, количество и порядок использования в тесте и определяют эффективность теста обнаруживать различные неисправности памяти.

Целью данной работы является анализ эффективности маршевых тестов обнаруживать сложные кодочувствительные неисправности памяти, а также разработка нового маршевого теста минимальной сложности.

1. Математическая модель неисправностей запоминающих устройств

Неисправности, затрагивающие несколько ячеек памяти, называются кодочувствительными неисправностями (pattern sensitive faults (PSF)) [6]. Для подобных неисправностей логическое состояние одной ячейки ЗУ, называемой базовой (base cell), может зависеть от содержимого (0 или 1) или от логических переходов из 1 в 0 или из 0 в 1 соседних ячейках (neighborhood cells) ЗУ. Различают два вида кодочувствительных неисправностей: неограниченные (unrestricted) и ограниченные (restricted), или граничные (neighborhood (NPSF)) кодочувствительные неисправности. Среди граничных неисправностей выделяют пассивные кодочувствительные неисправности (passive NPSF (PNPSF)). Для данных неисправностей состояние базовой ячейки не может быть изменено для определенного кода в $k-1$ соседних ячейках [3, 6, 8]. В качестве объекта исследования, чаще всего, рассматриваются пассивные кодочувствительные неисправности (PNPSFk), где k обозначает количество произвольных ячеек ЗУ емкостью N бит участвующих в конкретной неисправности.

Выделяют k различных классов PNPSFk в зависимости от местоположения в адресном пространстве памяти базовой ячейки (b) по отношению к соседним ячейкам (n). Например, для $k=4$ существует четыре класса PNPSF4: $b_{i0}n_{i1}n_{i2}n_{i3}$; $n_{i0}b_{i1}n_{i2}n_{i3}$; $n_{i0}n_{i1}b_{i2}n_{i3}$; $n_{i0}n_{i1}n_{i2}b_{i3}$, где адреса ячеек в адресном пространстве имеют следующее соотношение $i0 < i1 < i2 < i3$. Каждый класс включает две PNPSFk в зависимости от состояния базового элемента, которое не может быть изменено. Например, классу $n_{i0}n_{i1}b_{i2}n_{i3}$ неисправности PNPSF4, наряду с другими неисправностями, принадлежат следующие их конкретные виды: $\langle 1,1,\uparrow,0\rangle, \langle 0,0,\uparrow,1\rangle, \langle 1,1,\downarrow,0\rangle, \langle 0,0,\downarrow,1\rangle$. В соседних ячейках возможны любые из $2k-1$ двоичных наборов, каждый из которых определяет конкретные два неисправных поведения базовой ячейки. Символы $\uparrow$ и $\downarrow$ в PNPSFk неисправности означают невозможность

выполнения соответствующего перехода из нуля в единицу $(\uparrow)$ и наоборот $(\downarrow)$. Общее количество PNPSFk относящихся к k ячейкам ЗУ определяется формулой:

$$2k \times 2^{k-1} = k \times 2^{k} \quad (1)$$

2. Эффективность обнаружения кодочувствительных неисправностей маршевыми тестами

Развитие методов многократного применения маршевых тестов, с изменяемыми начальными состояниями ячеек памяти и адресной последовательностью, привели к появлению псевдоисчерпывающих тестов памяти [12]. Сущность подобных тестов заключается в формировании в произвольных k из N ячейках памяти всевозможных $2k$ двоичных комбинаций. Основой эффективности таких тестов является формирования различных видов орбит, представляющих собой набор двоичных комбинаций в произвольных k из N ячейках памяти. В рамках маршевых тестов возможным является формирование 8 видов орбит, представленных в табл. 1 и 2.

При описании предыдущей и текущей фаз теста, приведенных в табл. 1 и 2, показана последняя операция записи в фазе, после которой могут быть только операции чтения. В текущей фазе также показана первая операция записи, перед которой возможно использование только операций чтения. Отметим, что орбиты O_0, O_1, O_2 и O_3 формируются фазами, которые инвертируют содержимое запоминающего устройства. Приведенные орбиты хорошо описывают основные свойства тестов и широко представлено в классических маршевых тестах [12].

Таблица 1

Орбиты, формируемые тестами типа MATS ++ и March C–

Орбита	O_0	O_1	O_2	O_3
Предыдущая фаза теста	(…, *w*0, …)	(…, *w*0, …)	(…, *w*1, …)	(…, *w*1, …)
Фаза теста	⇑(…, *w*1, …)	⇓(…, *w*1, …)	⇑(…, *w*0, …)	⇓(…, *w*0, …)
P_0	000…00	000…00	111…11	111…11
P_1	000…01	100…00	111…10	011…11
P_2	000…11	110…00	111…00	001…11
…	…	…	…	…
P_{k-1}	011…11	111…10	100…00	000…01
P_k	111…11	111…11	000…0	000…00

Таблица 2

Орбиты, формируемые тестами типа March A

Орбита	Q0	Q1	Q2	Q3
Предыдущая фаза теста	(…, w0, …)	(…, w0, …)	(…, w1, …)	(…, w1, …)
Фаза теста	⇑(…, w1, …, w0, …)	⇓(…, w1, …, w0, …)	⇑(…, w0, …, w1, …)	⇓(…, w0, …, w1, …)
P0	000…00	000…00	111…11	111…11
P1	000…01	100…00	111…10	011…11
P2	000…10	010…00	111…01	101…11
…	…	…	…	…
Pk−1	010…00	000…10	101…11	111…01
Pk	100…00	000…01	011…11	111…10

Формирование маршевыми тестами различного рода орбит обеспечивает условие активизации многообразных неисправных состояний памяти. Причем выполнение условия генерирования псевдоисчерпывающих комбинаций в произвольных k из N ячейках памяти, с последующей их проверкой, гарантирует 100%-ое обнаружение различных подмножеств неисправностей [3, 12]. Отметим, что для определения конкретной неисправности необходимо выполнение условия ее активизации (sensitization) и ее обнаружения (detection) [6].

В общем случае, для кодочувствительных неисправностей условие активизации будет состоять из операции записи 1 в базовую ячейку, выполняющую переход из 0 в 1, и, наоборот записи 0, осуществляющей переход из 1 в 0, для всех возможных 2^{k-1} состояний в $k-1$ соседних ячейках. После выполнения каждого из переходов необходимо выполнение операции чтения из базовой ячейки, что и будет являться условием обнаружения конкретной неисправности. Причем операция чтения может быть, как в текущей фазе, так и в последующей, важным является то, что между изменением содержимого базовой ячейки (выполнением перехода) и чтением ее содержимого не было промежуточных операций записи. Для случая активных кодочувствительных неисправностей необходимо для базовой ячейки осуществить операции $r0$ и $r1$ при всевозможных изменениях состояний соседних ячеек.

Маршевые тесты MATS+: $\{\Uparrow\Downarrow (w0); \Uparrow (r0, w1); \Downarrow (r1, w0)\}$ и MATS++: $\{\Uparrow\Downarrow (w0); \Uparrow (r0, w1); \Downarrow (r1, w0, r0)\}$ формируют по две одинаковые орбиты O_0 и O_3 обеспечивающие активизацию k неисправностей PNPSFk

вида $\langle 0,0,0,\ldots,\ 0,\uparrow,1,1,1,\ldots,\ 1\rangle$ и k неисправностей вида $\langle 0,0,0,\ldots\ 0,\downarrow,1,1,1,\ldots,\ 1\rangle$. Однако, только MATS++ обеспечивает обнаружение $2k$ неисправностей PNPSFk, так как, в отличие от MATS+ в данном тесте обеспечены условия обнаружения активизированных неисправностей. Операция чтения $r1$ в третьей фазе $\Downarrow(r1, w0, r0)$ обеспечивает обнаружение неисправностей $\langle 0,0,0,\ldots,\ 0,\uparrow,1,1,1,\ldots,\ 1\rangle$ активизированных во второй фазе $\Uparrow(r0, w1)$, а операция $r0$ в третьей фазе обнаруживает неисправности $\langle 0,0,0,\ldots\ 0,\downarrow,1,1,1,\ldots,\ 1\rangle$ активизированные в этой же фазе.

Каждая из орбит, представленных в табл. 1, обеспечивает условие активизации конкретного вида PNPSFk, а именно O_0: $\langle 0,0,0,\ldots,\ 0,\uparrow,1,1,1,\ldots,\ 1\rangle$, O_1: $\langle 1,1,1,\ldots,\ 1,\uparrow,0,0,0,\ldots,\ 0\rangle$, O_2: $\langle 1,1,1,\ldots,\ 1,\downarrow,0,0,0,\ldots,\ 0\rangle$, O_3: $\langle 0,0,0,\ldots,\ 0,\downarrow,1,1,1,\ldots,\ 1\rangle$. В тоже время, орбиты Q_0, Q_1, Q_2 и Q_3 обеспечивают обнаружение двух видов неисправностей, каждая, а именно орбиты Q_0, Q_1 формируют условие активизации неисправностей $\langle 0,0,0,\ldots,\ 0,\uparrow,0,0,0,\ldots,\ 0\rangle$ и $\langle 0,0,0,\ldots,\ 0,\downarrow,0,0,0,\ldots,\ 0\rangle$ а Q_2, Q_3 $\langle 1,1,1,\ldots,\ 1,\uparrow,1,1,1,\ldots,\ 1\rangle$ и $\langle 1,1,1,\ldots,\ 1,\downarrow,1,1,1,\ldots,\ 1\rangle$. Анализ орбит, представленных в табл. 1 и 2 и обнаруживаемых с их помощью PNPSFk позволяет сформулировать следующее утверждение.

Утверждение. К множеству обнаруживаемых маршевыми тестами неисправностей PNPSFk относятся, такие их разновидности для которых, в соседних ячейках до базовой ячейки находятся значения все 0, либо все 1, после базовой также находятся одинаковые значения, причем независимо от значений в соседних ячейках до базовой ячейки.

Для случая $k = 4$, к обнаруживаемым неисправностям PNPSF4 относятся неисправности четырех различных классов, представленных в табл. 3. Количество неисправностей, обнаруживаемых данными классами составляет 24, а общее количество возможных неисправностей для данного k согласно (1) равно 64. Остальные 40 неисправностей PNPSF4: $\langle\uparrow,0,0,1\rangle$, $\langle\uparrow,0,1,0\rangle$, $\langle\uparrow,0,1,1\rangle$, … , $\langle 1,1,0,\downarrow\rangle$ относятся к подмножеству не обнаруживаемых однократными маршевыми тестами неисправностей.

Таблица 3

Обнаруживаемые неисправности PNPSF4

Классы *PNPSF*4	Неисправности *PNPSF*4
$b_{i0}\, n_{i1}\, n_{i2}\, n_{i3}$:	$\langle\uparrow,0,0,0\rangle$, $\langle\downarrow,0,0,0\rangle$, $\langle\uparrow,1,1,1\rangle$, $\langle\downarrow,1,1,1\rangle$
$n_{i0}\, b_{i1}\, n_{i2}\, n_{i3}$:	$\langle 0,\uparrow,0,0\rangle$, $\langle 0,\downarrow,0,0\rangle$, $\langle 1,\uparrow,1,1\rangle$, $\langle 1,\downarrow,1,1\rangle$, $\langle 0,\uparrow,1,1\rangle$, $\langle 0,\downarrow,1,1\rangle$, $\langle 1,\uparrow,0,0\rangle$, $\langle 1,\downarrow,0,0\rangle$
$n_{i0}\, n_{i1}\, b_{i2}\, n_{i3}$:	$\langle 0,0,\uparrow,0\rangle$, $\langle 0,0.\downarrow,0\rangle$, $\langle 1,1,\uparrow,1\rangle$, $\langle 1,1,\downarrow,1\rangle$, $\langle 0,0,\uparrow,1\rangle$, $\langle 0,0,\downarrow,1\rangle$, $\langle 1,1,\uparrow,0\rangle$, $\langle 1,1,\downarrow,0\rangle$
$n_{i0}\, n_{i1}\, n_{i2}\, b_{i3}$:	$\langle 0,0,0,\uparrow\rangle$, $\langle 0,0,0,\downarrow\rangle$, $\langle 1,1,1,\uparrow\rangle$, $\langle 1,1,1,\downarrow\rangle$

В качестве примера рассмотрим случай неисправности $\langle 0,0,0,\ldots,\ 0,\uparrow,1,1,1,\ldots,\ 1\rangle$. Как было показано выше, для обнаружения данной неисправности необходимо, чтобы предыдущая фаза вида $\Uparrow\Downarrow\ (\ldots,\ w0\ldots)$ обеспечила запись во все ячейки нулевых начальных значений. Также, как и в случае орбит, приведенная фаза может иметь произвольный вид при одном ограничении, что последняя операция записи должна быть $w0$, после которой могут применяться только операции чтения. Текущая фаза $\Uparrow(r0, w1)$ обеспечивает условие активизации k неисправностей PNPSFk вида $\langle 0,0,0,\ldots,\ 0,\uparrow,1,1,1,\ldots,\ 1\rangle$. В последующей фазе $\Uparrow\Downarrow\ (r1,\ldots)$ операция чтения $r1$ обеспечивает условие обнаружения всех k неисправностей $\langle 0,0,0,\ldots,\ 0,\uparrow,1,1,1,\ldots,\ 1\rangle$. Применение такого примитива состоящего из трех последовательных фаз $\Uparrow\Downarrow\ (\ldots,\ w0\ldots)$, $\Uparrow(r0, w1)$ и $\Uparrow\Downarrow\ (r1,\ldots)$ гарантирует обнаружение всех k неисправностей вида $\langle 0,0,0,\ldots,\ 0,\uparrow,1,1,1,\ldots,\ 1\rangle$. В частном случае, приведенный примитив, представляет собой маршевый тест MATS:

$\{\Uparrow\Downarrow\ (w0);\ \Uparrow(r0, w1);\ \Downarrow(r1)\}$.

Следует отметить, что конкретная фаза маршевого теста может участвовать в различных примитивах, однако не более чем в трех. Например, в тесте March C−: $\{\Uparrow\Downarrow\ (w0);\ \Uparrow(r0, w1);\ \Uparrow(r1, w0);\ \Downarrow(r0, w1);\ \Downarrow(r1, w0);\ \Uparrow\Downarrow\ (r0)\}$ вторая фаза $\Uparrow(r0, w1)$ является предыдущей фазой, третья фаза $\Uparrow(r1, w0)$ – текущей и четвертая $\Downarrow(r0, w1)$ последующей фазой примитива обеспечивающего активизацию и обнаружение неисправности $\langle 1,1,1,\ldots,\ 1,\downarrow,0,0,0,\ldots,\ 0\rangle$. Та же четвертая фаза $\Downarrow(r0, w1)$ является текущей фазой примитива,

описывающего обнаружение неисправности $\langle 1,1,1,\ldots,1,\uparrow,0,0,0,\ldots,0\rangle$ и предыдущей в примитиве, соответствующем неисправности $\langle 0,0,0,\ldots,0,\downarrow,1,1,1,\ldots,1\rangle$. Все множество примитивов и описываемые ими обнаруживаемые PNPSFk приведены в табл. 4.

Таблица 4

Примитивы, формируемые тестом March C−

Примитив	⇑⇓(*w*0), ⇑(*r*0,*w*1), ⇑(*r*1,*w*0)	⇑(*r*0,*w*1), ⇑(*r*1,*w*0), ⇓(*r*0,*w*1)	⇑(*r*1,*w*0), ⇓(*r*0,*w*1), ⇓(*r*1,*w*0)	⇓(*r*0,*w*1), ⇓(*r*1,*w*0), ⇑⇓(*r*0)
PNPSFk	⟨0,0,0,…,0,↑, 1, 1,1,…,1⟩	⟨1,1,1,…,1,↓, 0, 0,0,…,0⟩	⟨1,1,1,…,1,↑, 0, 0,0,…,0⟩	⟨0,0,0,…,0,↓, 1, 1,1,…,1⟩

Для конкретного вида PNPSFk неисправностей $\langle 0,0,0,\ldots,0,\uparrow,1,1,1,\ldots,1\rangle$, примитив, обеспечивающий их обнаружение, может состоять из двух фаз, предыдущей фазы $\Uparrow\Downarrow(w0)$ и текущей $\Uparrow(r0,w1,r1)$ в которой обеспечивается и активизация и обнаружение указанных неисправностей. В обоих случаях минимальная временная сложность примитивов равняется $4N$. Однако с учетом того, что конкретная фаза маршевого теста может участвовать в нескольких примитивах, средняя и суммарная сложности примитивов, описывающая обнаружение более чем одной неисправности PNPSFk, существенно меньше, как это видно на примере теста March C−. Временная сложность данного теста равняется $10N$, а сам тест обнаруживает четыре PNPSFk.

Более сложные примитивы могут обеспечить условия активизации и обнаружения не более чем двух видов PNPSFk. Это следует из того, что текущая фаза может сформировать только два перехода $(\uparrow,\downarrow)$ в базовой ячейке для каждого из четырех состояний в соседних ячейках (см. Утверждение 1). Примером подобного примитива является случай обнаружения двух видов PNPSFk, а именно $\langle 0,0,0,\ldots,0,\uparrow,1,1,1,\ldots,1\rangle$ и $\langle 0,0,0,\ldots,0,\downarrow,1,1,1,\ldots,1\rangle$ состоящим из предыдущей фазы $\Uparrow\Downarrow(\ldots,w0,\ldots)$ и текущей вида $\Uparrow(r0,w1,r1,w0,r0,w1)$. Примитив, представленный фазами $\Uparrow\Downarrow(\ldots,w1,\ldots)$ и $\Downarrow(r1,w0,r0,w1,r1,w0)$, описывает обнаружение этих же неисправностей $\langle 0,0,0,\ldots,0,\uparrow,1,1,1,\ldots,1\rangle$ и $\langle 0,0,0,\ldots,0,\downarrow,1,1,1,\ldots,1\rangle$. В тоже время два примитива $\Uparrow\Downarrow(\ldots,w0,\ldots)$, $\Downarrow(r0,w1,r1,w0,r0,w1)$ и

$\Uparrow\Downarrow(\ldots, w1, \ldots)$ и $\Uparrow(r1, w0, r0, w1, r1, w0)$ обеспечивают обнаружение следующих двух неисправностей $\langle 1,1,1,\ldots, 1, \uparrow, 0,0,0,\ldots, 0\rangle$ и $\langle 1,1,1,\ldots, 1, \downarrow, 0,0,0,\ldots, 0\rangle$.

Используя два из четырех приведенных примитивов, оказывается возможным построение маршевого теста, имеющего такую же покрывающую способность неисправности PNPSFk, как и тест March C−. Примером такого теста может быть тест $\{\Uparrow\Downarrow(w0); \Uparrow(r0, w1, r1, w0, r0, w1); \Uparrow(r1, w0, r0, w1, r1, w0)\}$, обнаруживающий, так же, как и March C−, четыре неисправности PNPSFk.

Для получения максимально возможной полноты покрытия достижимой в рамках однократного маршевого теста необходимо использовать примитивы, обеспечивающие покрытие PNPSFk соответствующих орбитам Q_0, Q_1, Q_2 и Q_3. Аналогично, как и для случая неисправностей, описываемых орбитами O_0, O_1, O_2 и O_3 получим примитивы, состоящие из двух фаз. Примитив, состоящий из предыдущей фазы $\Uparrow\Downarrow(\ldots, w0, \ldots)$ и текущей $\Uparrow\Downarrow(r0, w1, r1, w0, r0)$ фаз, описывает условия обнаружения неисправностей $\langle 0,0,0,\ldots, 0, \uparrow, 0,0,0,\ldots, 0\rangle$, и $\langle 0,0,0,\ldots, 0, \downarrow, 0,0,0,\ldots, 0\rangle$, а примитив $\Uparrow\Downarrow(\ldots, w1, \ldots)$ и $\Uparrow\Downarrow(r1, w0, r0, w1, r1)$ неисправностей $\langle 1,1,1,\ldots, 1, \uparrow, 1,1,1,\ldots, 1\rangle$ и $\langle 1,1,1,\ldots, 1, \downarrow, 1,1,1,\ldots, 1\rangle$. Отметим, что в двух предыдущих примитивах обнаружение неисправностей обеспечивается в текущей фазе каждого из них за счет соответствующих операций чтения. В случае первого примитива, состоящего из предыдущей фазы $\Uparrow\Downarrow(\ldots, w0, \ldots)$ и текущей $\Uparrow\Downarrow(r0, w1, r1, w0, r0)$, факт наличия неисправности $\langle 0,0,0,\ldots, 0, \uparrow, 0,0,0,\ldots, 0\rangle$ определяет операция чтения $r1$, а наличие неисправности $\langle 0,0,0,\ldots, 0, \downarrow, 0,0,0,\ldots, 0\rangle$, вторая операция чтения $r0$ текущей фазы. Присутствие неисправности $\langle 0,0,0,\ldots, 0, \downarrow, 0,0,0,\ldots, 0\rangle$ может быть определено и в последующей фазе, которая должна начинаться с операции чтения $r0$. Соответственно, можно показать, что примитив, состоящий из предыдущей фазы $\Uparrow\Downarrow(\ldots, w0, \ldots)$, текущей фазы $\Uparrow\Downarrow(r0, w1, r1, w0)$ и последующей $\Uparrow\Downarrow(r0)$, обеспечивает ее

обнаружение. В тоже время обнаружение неисправностей $\langle 1,1,1,\ldots,\ 1,\uparrow,1,1,1,\ldots,\ 1\rangle$ и $\langle 1,1,1,\ldots,\ 1,\downarrow,1,1,1,\ldots,\ 1\rangle$ также может быть обеспечено примитивом, состоящим из трех фаз, а именно, предыдущей фазы $\Uparrow\Downarrow\ (\ldots,\ w1,\ldots)$, текущей фазы $\Uparrow\Downarrow\ (r1, w0, r0, w1)$ и последующей фазы $\Uparrow\Downarrow\ (r1,\ldots)$.

Описанные ранее процедуры построения примитивов и их конкретные виды, приведенные выше, позволяют синтезировать маршевые тесты, однократное применение которых позволяет обеспечить максимальную полноту покрытия PNPSFk. Примером результата такого синтеза может быть тест $\{\Uparrow\Downarrow\ (w0); \Uparrow (r0, w1, r1, w0, r0); \Downarrow (r0, w1, r1, w0, r0, w1);$ $\Downarrow (r1, w0, r0, w1, r1); \Uparrow (r1, w0, r0, w1, r1, w0)\}$, сложность которого равняется сложности $23N$ известного теста March PS имеющего максимальную полноту покрытия в классе кодочувствительных неисправностей PNPSFk [20].

Объединение примитивов различной сложности позволяет синтезировать более простой маршевый тест March OP, имеющий меньшую временную сложность равную $18N$. Это тест имеет вид (2), а его примитивы, обеспечивающие обнаружение всевозможных неисправностей PNPSFk приведены в табл.5.

$$\{\Uparrow\Downarrow\ (w0); \Uparrow (r0, w1); \Uparrow (r1, w0); \Downarrow (r0, w1) \Downarrow (r1, w0, w0, w1, r1); \Downarrow (r1, w0); \Uparrow (r0, w1, r1, w0, r0)\} \quad (2)$$

Таблица 5

Примитивы, формируемые тестом March OP

Предыдущая фаза	$\Uparrow\Downarrow (w0)$	$\Uparrow(r0,w1)$	$\Uparrow(r1,w0)$	$\Downarrow(r0,w1)$	$\Downarrow(r1,w0,r0,w1,r1)$	$\Downarrow(r1,w0)$
Текущая фаза	$\Uparrow(r0,w1)$	$\Uparrow(r1,w0)$	$\Downarrow(r0,w1)$	$\Downarrow(r1,w0,r0,w1,r1)$	$\Downarrow(r1,w0)$	$\Uparrow(r0,w1,r1,w0,r0)$
Последующая фаза	$\Uparrow(r1,w0)$	$\Downarrow(r0,w1)$	$\Downarrow(r1,w0,r0,w1,r1)$	-	$\Uparrow(r0,w0,r1,w0,r0)$	-
Обнаруживаемые *PNPSFk*	$\langle 0,0,0,\ldots,0,\uparrow,1,1,1,\ldots,1\rangle$	$\langle 1,1,1,\ldots,1,\downarrow,0,0,0,\ldots,0\rangle$	$\langle 1,1,1,\ldots,1,\uparrow,0,0,0,\ldots,0\rangle$	$\langle 1,1,1,\ldots,1,\downarrow,1,1,1,\ldots,1\rangle$; $\langle 1,1,1,\ldots,1,\uparrow,1,1,1,\ldots,1\rangle$	$\langle 0,0,0,\ldots,0,\downarrow,1,1,1,\ldots,1\rangle$	$\langle 0,0,0,\ldots,0,\uparrow,0,0,0,\ldots,0\rangle$ $\langle 0,0,0,\ldots,0,\downarrow,0,0,0,\ldots,0\rangle$

Результатом применения различных комбинаций примитивов могут быть тесты запоминающих устройств, обладающие требуемой полнотой, как правило, максимальной, в классе PNPSFk и имеющие

различную временную сложность. Однако, тестом с максимальной полнотой покрытия PNPSFk и имеющим минимальную временную сложность, по мнению авторов, является тест March OP (2). Следует отметить, что, используя понятия примитивов, можно синтезировать тесты запоминающих устройств, позволяющих обнаруживать и другие разновидности сложных неисправностей.

Заключение

В представленной статье показана взаимосвязь между орбитами, формируемыми маршевыми тестами, и их способностью обнаруживать сложные кодочувствительные неисправности PNPSFk. Показано, что наличие минимального подмножества из восьми типов орбит при реализации маршевого теста является необходимым условием достижения максимальной полноты покрытия тестом PNPSFk неисправностей. Определены понятия примитивов и показаны их примеры, а также результаты синтеза маршевых тестов обеспечивающих требуемую полноту покрытия заданных видов PNPSFk. Приведен пример маршевого теста с максимальной полнотой покрытия и имеющий минимальную временную сложность.

Список литературы

1. Bushnell, M. L. Essentials of Electronic Testing for Digital, Memory & Mixed-Signal VLSI Circuits / M. L. Bushnell, V. D. Agrawal. – New York: Kluwer Academic Publishers, 2000. – 690 p.
2. Wang, L. T. VLSI Test Principles and Architectures: Design for Testability / L. T. Wang, C. W. Wu, X. Wen. – Amsterdam: Elsevier, 2006. – 808 p.
3. Ярмолик, В. Н. Контроль и диагностика вычислительных систем / В. Н. Ярмолик. – Минск: Бестпринт, 2019. – 387 с.
4. The International Technology Roadmap for Semiconductors: 2003 Edition (ITRS'2003). – San Jose: Semiconductor Industry Association, 2003. – 65 p.
5. Sharma, A. K. Advanced Semiconductor Memories: Architectures, Designs, and Applications / A. K. Sharma. – London: John Wiley & Sons, 2003. – 652 p.
6. Goor, A. J. Testing Semiconductor Memories: Theory and Practice / A. J. Goor. – Chichester: John Wiley & Sons, 1991.
7. Ярмолик, В. Н. Неразрушающее тестирование запоминающих устройств / В. Н. Ярмолик [и др.]. – Минск: Бестпринт, 2005. – 230 с.
8. Ярмолик, С. В. Маршевые тесты для самотестирования ОЗУ / С. В. Ярмолик, А. П. Занкович, А. А. Иванюк. – Минск: Бестпринт, 2009. – 270 с.

9. Marinescu, M. Simple and Efficient Algorithms for Functional RAM Testing: IEEE Int. Test Conf. / M. Marinescu. – IEEE Computer Society Press, 1982. – P. 236–239.

10. Nair, C. Efficient Algorithms for Testing Semiconductor Random-Access Memories: IEEE Int. Test Conf. / C. Nair, S. Thatte, J. Abraham. – IEEE Transactions on Computers, 1978. – vol. 27, no. 6. – P. 572-576.

11. Suk, D. S. A March Test for Functional Faults in Semiconductor Random-Access Memories: IEEE Trans. on Computers / D. S. Suk, S. M. Reddy. – IEEE Transactions on Computers, 1981. – vol. 30, no. 12. – P. 982–985.

12. Ярмолик, В. Н. Псевдоисчерпывающее тестирование запоминающих устройств на базе маршевых тестов типа March A / В. Н. Ярмолик, И. Мрозек, С. В. Ярмолик. // Информатика. – Минск, 2020. – № 2 (17). – С. 54–70.

13. Yarmolik, V. N. March ps(23n) test for DRAM pattern-sensitive faults: Proc. of the 7th AsianTest Symposium / V. N. Yarmolik, Y. Klimets, S. Demidenko. – IEEE Computer Society, 1998. – P. 354–357.