УДК 681.327

МОДЕЛИРОВАНИЕ ФУНКЦИОНАЛЬНЫХ НЕИСПРАВНОСТЕЙ ОПЕРАТИВНЫХ ЗАПОМИНАЮЩИХ УСТРОЙСТВ

А.А. ИВАНЮК, А.В. СТЕПАНОВ

Белорусский государственный университет информатики и радиоэлектроники
П. Бровки,6, Минск, 220013, Беларусь

Поступила в редакцию 7 апреля 2009

Создана детализированная VHDL-модель статического бит-ориентированного оперативного запоминающего устройства (ОЗУ) с целью проверки гипотезы об адекватности внедрения моделей неисправностей проводящих линий для отображения доминирующих типов физических дефектов ОЗУ. Спроектированная VHDL-модель позволяет исследовать природу неисправностей ОЗУ, оценить поведение цифрового устройства при наличии в нем дефектов, а также может быть применена для верификации существующих и при разработке новых алгоритмов тестирования ОЗУ.

Ключевые слова: бит-ориентированное ОЗУ, функциональные неисправности, маршевые тесты, моделирование, VHDL.

Введение

Современные достижения в области разработки средств вычислительной техники привели к появлению интегральных схем (ИС) со сверхбольшой степенью интеграции цифровых компонент. Стремительное усложнение структуры цифровых устройств определяет одну из основных проблем – проблему обеспечения высокой надежности функционирования проектируемых цифровых устройств. ОЗУ являются одними из основных компонентов разрабатываемых цифровых устройств и в то же время относятся к наименее надежным компонентам [1]. Вследствие этого большое значение имеет обеспечение корректного функционирования ОЗУ. Особенно это актуально для критических приложений, некорректное функционирование которых может нанести вред человеку. Примером являются системы автомобильной, авиационной и медицинской направленности, а также системы ядерной энергетики.

Основным фактором, который негативно влияет на достоверность работы цифровых устройств, является наличие физических дефектов в полупроводниковых кристаллах [2]. Тип дефекта зависит как от технологии изготовления устройства, так и от условий его эксплуатации. В качестве моделей, которые позволяют понять поведение цифровой схемы при условии наличия в ней определенного дефекта, используются функциональные неисправности. При формальном описании физических дефектов в виде математических моделей особое внимание уделяется адекватности отображения многообразия неисправных состояний цифрового устройства.

В настоящей работе выдвигается и проверяется следующая гипотеза: мы предполагаем, что внедрение функциональных неисправностей проводящих линий в проектные описания цифровых устройств позволяет адекватно отображать доминирующие типы физических дефектов ОЗУ.

Теоретический анализ

К доминирующим типам функциональных неисправностей проводящих линий относят: мостиковые неисправности и неисправности типа обрыв [3].

Мостиковой неисправностью принято называть неисправность, которая предполагает наличие нежелательного проводящего пути между двумя полюсами схемы (рис. 1). Причинами возникновения неисправностей могут служить различные физические дефекты, которые возникают как на стадии проектирования, так и во время эксплуатации устройства по назначению.

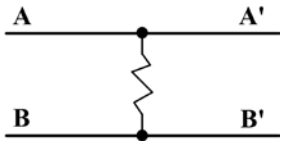


Рис. 1. Модель мостиковой неисправности

В настоящее время выделяют следующие модели мостиковых неисправностей, которые наиболее часто используются на практике: модель мостиковой неисправности типа «монтажное-И»/«монтажное-ИЛИ», модель доминантной мостиковой неисправности и модель мостиковой неисправности типа «доминантное-И»/«доминантное-ИЛИ» [3, 4]. Данные модели неисправностей позволяют описать такие дефекты, как замыкание сигнальной линии с линией питания/земли и замыкание двух сигнальных линий.

Неисправности типа обрыв обычно описываются наличием высокоомной нагрузки на входной или выходной линии компоненты, что эквивалентно отключению порта от сигнальной линии [5].

Рассмотрим существующие методы анализа функционирования цифровых устройств с учетом имеющихся неисправностей и возникших сбоев.

Развитие систем автоматизированного проектирования (САПР) цифровых устройств позволило разработчикам оценивать поведение устройства еще на начальных стадиях проектирования. Среди методов, нацеленных на определение корректного функционирования, выделяют параметрическое и функциональное моделирование. Параметрическое моделирование позволяет определять отклонение реальных физических параметров устройства (временные и электрические характеристики) от нормативных. Для параметрического моделирования применяются SPICE-подобные системы, которые используют схемотехническое описание устройств в базисе таких элементов, как транзисторы, резисторы, конденсаторы и проводящие линии [6]. С помощью подобных систем можно детально проанализировать такие параметры, как потребляемая мощность, токи утечки, переключательная активность и т.д.

Применительно к ОЗУ использование SPICE-систем затруднено высокой сложностью схемотехнической модели и значительными временными затратами на выполнение моделирования. Поэтому среди специалистов в области тестирования ОЗУ широкое распространение получила функциональная модель, которая используется для выявления факта алгоритмической работоспособности/неработоспособности устройства. Часто в качестве средств моделирования используются программные средства, имитирующие поведение цифровых устройств с учетом их временных характеристик. При этом устройства представляются в виде набора RTL-примитивов (RTL, Register Transfer Level), поведение которых хорошо известно и для которых существует предопределенная программная модель [7]. При функциональном моделировании компонента ОЗУ, как правило, описывается в виде массива запоминающих ячеек, а такие функциональные блоки, как дешифратор адреса и усилитель чтения/записи как отдельные компоненты не рассматриваются. Данный подход не позволяет адекватно отобразить реальные дефекты микросхем ОЗУ.

В данной статье предложен подход, основанный на использовании HDL-языка (HDL, Hardware Description Language) и современных САПР цифровых устройств, который позволяет не только выполнять детализированные описания устройств в базисе RTL, но и анализировать их функционирование с учетом имеющихся неисправностей.

Методика

В рамках данной работы мы создали детализированную модель бит-ориентированного статического ОЗУ (СОЗУ) посредством языка VHDL [8], которая включает в себя описание ос-

новных функциональных блоков и реальных проводящих линий, участвующих в соединении внутренних компонент устройства.

Основной целью являлось воспроизведение структуры и логики функционирования СОЗУ, что предоставило возможность оценивать поведение цифрового устройства при наличии в нем физических дефектов. В качестве средств разработки выбран язык VHDL, который является международным стандартом в области автоматизации проектирования цифровых систем. С помощью VHDL можно точно описать проектируемые системы и выполнить функциональное моделирование. Кроме того, VHDL позволяет адекватно описать реальные проводящие линии, которые будут участвовать в соединении внутренних компонент законченного устройства, используя языковые объекты типа signal [9]. Именно благодаря этому для VHDL-описаний цифровых устройств существует много решений по внедрению неисправностей на сигнальные линии [9-11].

На рис. 2 представлен пример структуры модуля бит-ориентированного СОЗУ с точки зрения основных функциональных блоков и проводящих линий.

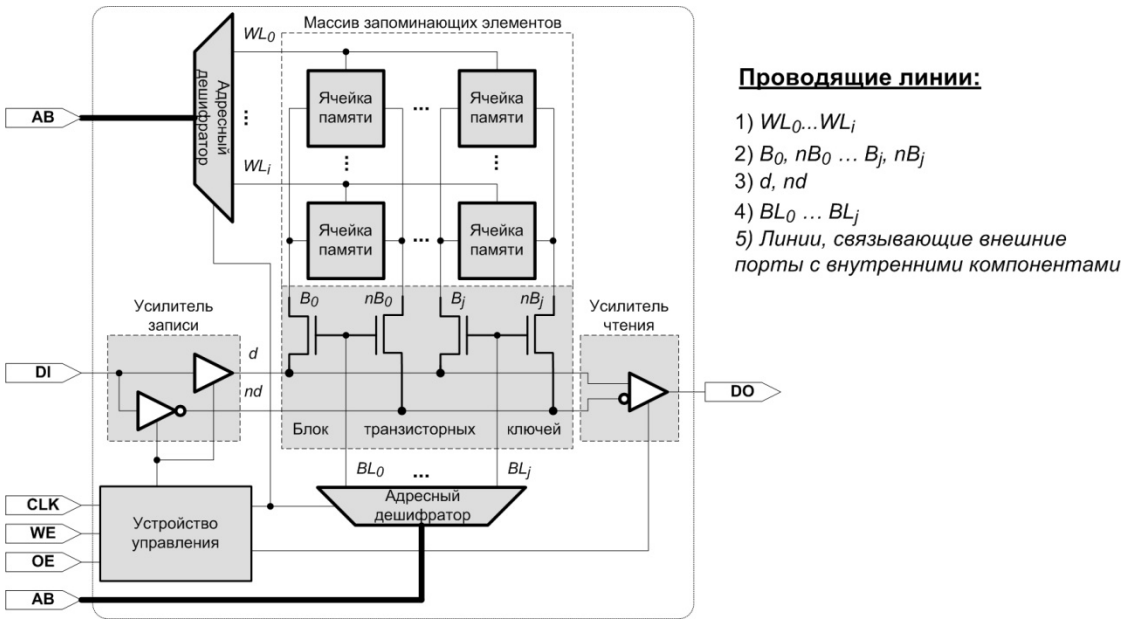


Рис. 2. Пример структуры модуля бит-ориентированного СОЗУ

Блок управления, адресные дешифраторы и проводящие линии могут быть адекватно описаны на синтезируемом подмножестве языка VHDL. Однако VHDL не обладает лингвистическими конструкциями, которые бы описывали транзисторную логику и воспринимались средствами синтеза. Тем не менее, такие компоненты, как ячейка памяти, усилитель чтения/записи и блок транзисторных ключей, могут быть описаны на несинтезируемом подмножестве языка VHDL, используя объекты signal типа std_logic, и являются пригодными для функционального моделирования.

Рассмотрим подробнее предлагаемый подход к описанию функционирования данных компонент на поведенческом несинтезируемом подмножестве языка VHDL.

Каждая ячейка памяти $CELL_{ij}$ подключена к соответствующей строке слова WL_i и двум битовым строкам B_j и nB_j. Соответственно, компонент $CELL_{ij}$ имеет один входной порт *WL* и два двунаправленных порта *B* и *nB*. Каждый запоминающий элемент СОЗУ может находиться в трех состояниях.

1) *Состояние хранения*: на входной порт *WL* подано значение '0', порты *B* и *nB* находятся в высокоимпедансном состоянии (значение ‘Z’) (рис. 3, *а*);

2) *Чтение информации*: когда соответствующий элемент $CELL_{ij}$ выбран, на порт *WL* подается значение '1', которое подключает транзисторы хранения к выходным портам *B* и *nB*. Прямое и инверсное значение хранимого бита через соответствующую пару транзисторных ключей TK_j, линии *d* и *nd*, поступает на блок усилителя чтения (рис. 3, *б*).

3) *Запись информации*: логическое значение (‘0’ или ‘1’) записываемого бита через усилитель записи, линии *d* и *nd*, соответствующие открытые транзисторные ключи, поступает в прямом и инверсном виде на порты *B* и *nB* запоминающего элемента. Если при этом на порт *WL* подано значение '1', то элемент $CELL_{ij}$ начнет хранить соответствующее значение (рис. 3, *в*).

Во избежание конфликтов при моделировании записываемые и считываемые значения сигналов на портах *B* и *nB* кодируются различными символами типа std_logic. Так, хранимое значение элементом CELL может принимать два устойчивых значения ‘L’ и ‘H’, что соответствует понятиям слабого нуля и слабой единицы. При чтении хранимого значения нуля на выходные порты *B* и *nB* записываются значения ‘L’ и ‘H’ соответственно. Аналогично, при чтении хранимого значения единицы выходные порты принимают значения ‘L’ и ‘H’. Если запоминающий элемент выбран WL_i='1', а на входных портах *BL* и *nBL* появились комплементарные значения сильных сигналов, то это означает процесс записи информации, при котором значения с портов *BL* и *nBL* поступают на хранящую ячейку. Для отделения процесса чтения от процесса записи условимся, что чтение информации для выбранного элемента $CELL_{ij}$ будет происходить при условии установки значения ‘Z’ на обоих портах *B* и *nB*.

Аналогичным образом определим поведение одной пары транзисторных ключей TK_j. Оба МОП-транзистора закрыты, если соответствующая линия BL_j принимает логическое значение ‘0’. При этом на линии B_j и nB_j поступает значение ‘Z’ (рис. 3, *а*). Если транзисторы открыты, а на линиях *d* и *nd* находится значение ‘Z’ (линии отключены от выходов усилителя записи), то на входы усилителя чтения поступают слабые значения парафазных сигналов от адресованного запоминающего элемента (рис. 3, *б*). При записи информации по линиям *d* и *nd* передаются сильные значения сигналов, которые через пару открытых транзисторов и соответствующие линии B_j и nB_j поступают на входные порты адресуемого элемента памяти (рис. 3, *в*).

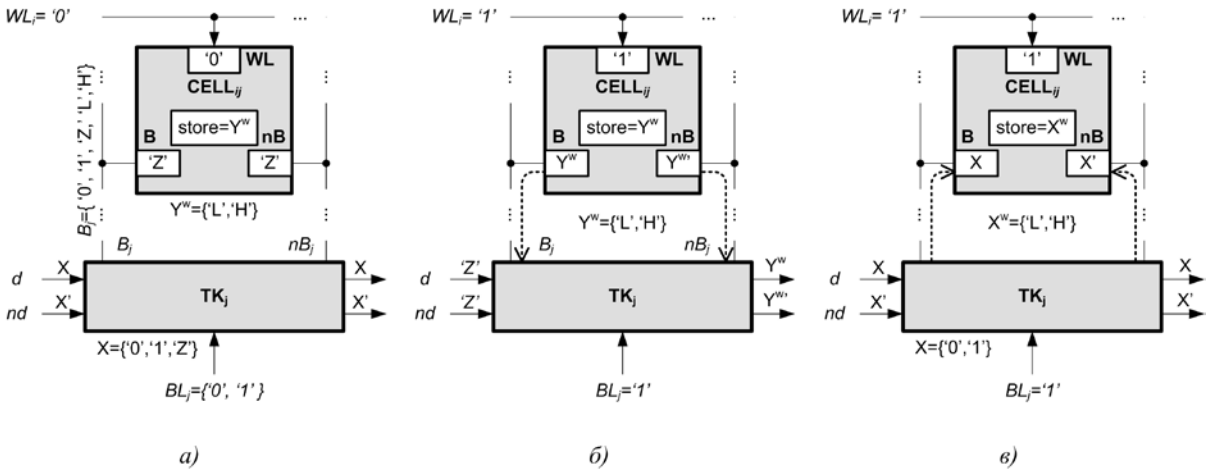


Рис. 3. Режимы функционирования блоков CELL и ТК: *а*) хранение; *б*) чтение; *в*) запись

Усилитель чтения производит сравнение значений двух сигналов на линиях *d* и *nd*. Если значение напряжения на первом входе не превышает значение напряжения на втором входе, то усилитель чтения вырабатывает на своем выходе сигнал уровня логического нуля. В противном случае, если значение напряжения на первом входе превышает значение напряжения на втором, то усилитель выработает сигнал уровня логической единицы. Кроме усиления выходного сигнала компонента играет роль буфера с тремя состояниями, управляемого устройством управления.

Проведенная верификация VHDL-описания показала, что разработанная модель корректно воспроизводит функционирование модуля бит-ориентированного СОЗУ.

Экспериментальная часть

С целью проверки гипотезы об адекватности внедрения моделей неисправностей проводящих линий для отображения доминирующих типов физических дефектов ОЗУ были проведены экспериментальные исследования.

Определим основные условия проведения экспериментов. В качестве методики внедрения неисправностей ОЗУ применяется подход внедрения неисправностей проводящих линий, который заключается в реализации подмены сигналов VHDL-описаний [12]. При моделировании работы ОЗУ вероятность возникновения той или иной неисправности проводящих линий принимается равновероятной. Внедряемые мостиковые неисправности затрагивают две сигнальные линии. Метод оценки поведения ОЗУ базируется на анализе получаемых синдромов для каждой ячейки памяти при использовании классического разрушающего маршевого теста March-*17N* [13]. В настоящее время маршевые тесты широко используются в качестве диагностирующих алгоритмов для цифровых устройств [14–16]. Маршевый тест March-*17N* имеет следующую запись на универсальном языке описания тестов MTL [17]:

{↕(*w*0); ↑(*r*0, *w*1, *r*1); ↕(*r*1); ↑(*r*1, *w*0, *r*0); ↕(*r*0); ↓(*r*0, *w*1, *r*1); ↕(*r*1); ↓(*r*1, *w*0, *r*0); ↕(*r*0)}.

Синдром *S* для произвольного разрушающего маршевого теста представляет собой информационный пакет следующего вида:

$$S = E_0E_1E_2E_3E_4\ldots E_{k-1},$$

где $E_i \in \{0, 1\}$, $i = 0, 1, 2, \ldots, k\text{-}1$, есть *i*-ый разряд сигнатуры маршевого теста, *k* - количество операций чтения в маршевом тесте. Разряд сигнатуры $E_i = 0(1)$ означает, что *i*-ая операция чтения вернула ожидаемое (некорректное) значение ячейки памяти.

В работе [13] представлен словарь синдромов, полученный для различных типов неисправностей массива ячеек ОЗУ при использовании маршевого теста March-*17N*. Было показано, что каждая неисправность из словаря имеет свой уникальный маршевый синдром.

Пример. Синдром неисправности константного нуля (*stuck-at 0 (SAF0)*) для маршевого теста March-*17N* равен «011100011100» ($S = E_0E_1E_2E_3E_4E_5E_6E_7E_8E_9E_{10}E_{11}$).

На рис. 4 представлена упрощенная схема проводимых экспериментальных исследований. Результаты эксперимента представляют собой информацию следующего вида: тип внедряемой неисправности проводящих линий; пара сигнальных линий, на которую внедряется неисправность; синдром маршевого теста для каждой ячейки памяти.

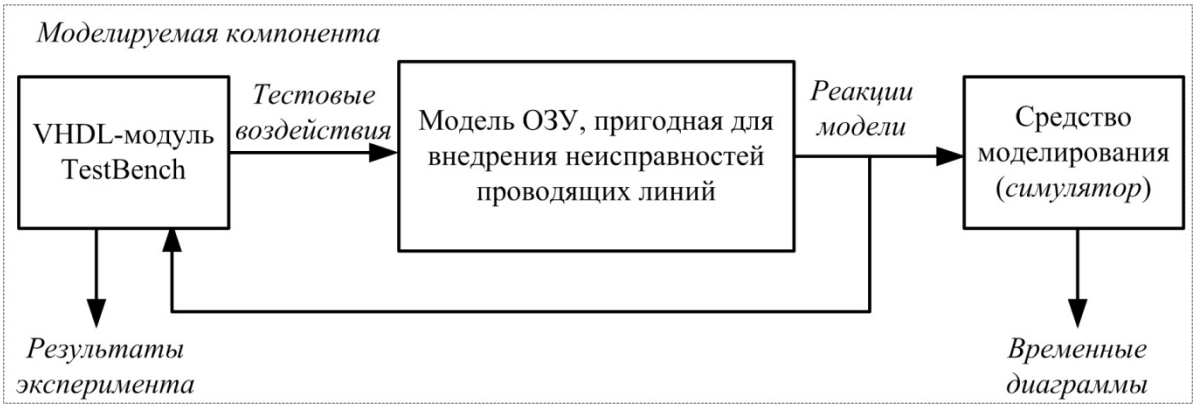


Рис. 4. Упрощенная схема проведения экспериментальных исследований

Моделирование выполнялось на базе пакета программных средств ISE WebPACK 9.1i [18] компании Xilinx.

Результаты и их обсуждение

В табл. 1 представлены функциональные неисправности ОЗУ и их <*S* / *F* / *R*> нотация [19], которые были обнаружены при внедрении неисправностей проводящих линий, затрагивающих всего две сигнальные линии.

Нотация <*S* / *F* / *R*> широко используется для описания простейших неисправностей и неисправностей взаимного влияния. Так, согласно предложенной нотации, функциональная модель неисправностей ОЗУ, затрагивающих одну ячейку памяти, может быть представлена следующим образом:

$$< S / F / R >,$$

где S – последовательность операций, выполняемая над массивом запоминающих ячеек, которая спровоцирует искажение локальных данных отказавшего элемента памяти, т.е. функционирование запоминающего устройства не будет соответствовать ожидаемому поведению; F – хранимое значение неисправной ячейки памяти, $F \in \{0, 1\}$; R – значение, возвращаемое при выполнении операции чтения, $R \in \{0, 1, –\}$. При выполнении операции записи это значение не определено (R = '–').

В свою очередь, последовательность операций S содержит текущее значение ячейки памяти, тип выполняемой операции над ячейкой (чтение/запись) и значение, которое будет хранить ячейка памяти после выполнения данной операции.

Подобным образом описывается функциональная модель неисправностей взаимного влияния:

$$< Sa, Sv / F / R >,$$

где Sa – последовательность операций, выполняемая над ячейкой-агрессором, а Sv – последовательность операций, выполняемая над ячейкой-жертвой.

Таблица 1. Обнаруженные функциональные неисправности ОЗУ

Функциональная неисправность ОЗУ	<*S / F / R*> нотация
Неисправность константного нуля	(('-', '-', '-'), '0', '-')
Неисправность константной единицы	(('-', '-', '-'), '1', '-')
Неисправность прямого действия	(('1', 'w', '0'), ('-', '-', '-'), '0', '-')
Неисправность прямого действия	(('0', 'w', '1'), ('-', '-', '-'), '1', '-')

Результаты экспериментальных исследований показали, что неисправности проводящих линий, затрагивающие две линии, не позволяют описать все типы функциональных неисправностей ОЗУ. Так, например, инверсная неисправность взаимного влияния (CFin = {(('0', 'w', '1'), ('1', '-', '-'), '0', '-')}) не может быть описана мостиковой неисправностью, затрагивающей две линии. Однако данную неисправность можно описать при помощи мостиковой неисправности более высокой кратности (минимум 6).

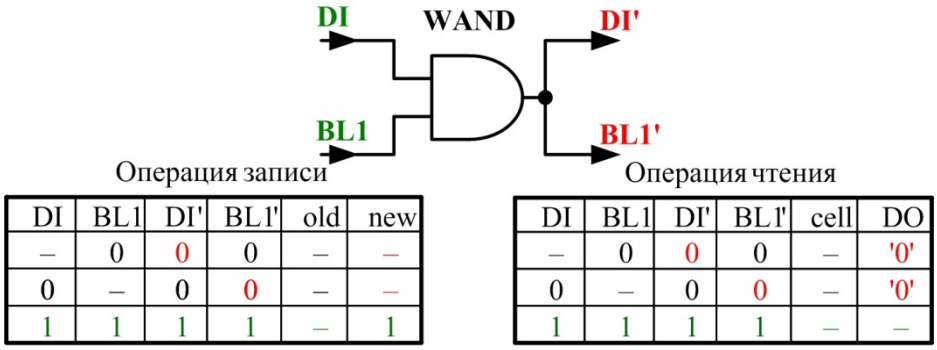
Во время проведения исследований были обнаружены неисправности, которые отсутствовали в выбранном словаре неисправностей для маршевого теста March-17*N* (табл. 2).

Таблица 2. Обнаруженные неизвестные функциональные неисправности ОЗУ

Пара линий	Модель мостиковой неисправности	Синдром неисправности
DI и *BL*1	«монтажное-И»	000100100100
*WL*1 и *BL*0	«монтажное-ИЛИ»	100100000000
*WL*1 и *BL*1	«монтажное-ИЛИ»	000100000100

Пример. Рассмотрим одну из неисправностей, тип которой не был определен с помощью маршевого теста. Данная неисправность была обнаружена при внедрении мостиковой неисправности типа «монтажное-И» на пару линий *DI* (линия данных) и *BL*1 (линия блока транзисторных ключей). Маршевый синдром неисправности – «000100100100». Данная неисправность представляет собой симбиоз переходной неисправности и неисправности некорректного чтения (рис. 5).

Для обнаружения переходных неисправностей необходимо, чтобы маршевый тест содержал в себе операции записи, которые осуществляют всевозможные логические переходы в каждой ячейке памяти и операции чтения содержимого ячейки после каждого перехода. Примером маршевого теста, который позволяет обнаруживать все переходные неисправности, является тест *MATS++*: {↕(*w*0); ↑(*r*0, *w*1); ↓(*r*1, *w*0, *r*0)}. Сложность теста, которая оценивается как совокупность всех элементарных операций с ОЗУ, равна 6*N*, где N – информационная емкость памяти в битах.



Операция записи

DI	BL1	DI'	BL1'	old	new
–	0	0	0	–	–
0	–	0	0	–	–
1	1	1	1	–	1

Операция чтения

DI	BL1	DI'	BL1'	cell	DO
–	0	0	0	–	'0'
0	–	0	0	–	'0'
1	1	1	1	–	–

Рис. 5. Неисправность TF (transition fault) + IRF (incorrect read fault)

Анализ обнаруженной неисправности показал, что при наличии данной неисправности в ОЗУ значение операции чтения из выбранной ячейки памяти и итог операции записи зависит от значения на входной линии данных. Как следствие, маршевый тест *MATS++* не позволяет обнаруживать данную неисправность в граничных ячейках ОЗУ.

На рис. 6 представлен маршевый тест *March C–*, позволяющий обнаруживать неисправность данного типа вне зависимости от ее местоположения. Сложность теста равна $10N$.

Рис. 6. Условия активизации и обнаружения неисправности

На рис. 7 представлен сравнительный анализ полученных процентных долей функциональных неисправностей ОЗУ в результате эксперимента с данными, полученными на производстве [13] . Как видно, наиболее часто встречаемыми являются константные неисправности, которые могут быть вызваны различными физическими дефектами.

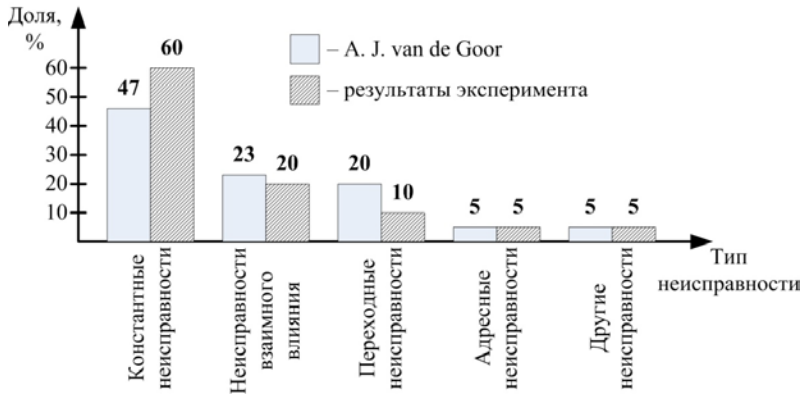


Рис. 7. Процентные доли функциональных неисправностей ОЗУ

Представленные результаты проведенных исследований позволяют сделать следующий вывод: функциональные модели неисправностей проводящих линий позволяют адекватно отображать доминирующие типы физических дефектов ОЗУ.

Результаты сравнительной оценки временных затрат на выполнение процедуры тестирования для классической модели ОЗУ и предложенной модели показывают, что время моделирования возросло не более чем на 10%.

Заключение

Предложен новый подход к моделированию функциональных неисправностей ОЗУ. Спроектирована VHDL-модель ОЗУ, которая позволяет оценивать поведение цифрового

устройства при наличии в нем дефектов, а также может быть применена для верификации существующих и разрабатываемых алгоритмов тестирования ОЗУ. Проведенный анализ временных затрат на моделирование свидетельствует о приемлемости использования данной модели на практике. Полученные результаты могут быть использованы для решения задач в области диагностики неисправностей ОЗУ, нацеленных на определение типов неисправностей и их местоположение.

RANDOM ACCESS MEMORIES FAULTS SIMULATION

A.A. IVANIUK, A.V. STEPANOV

Abstract

A new method of random access memory functional faults simulation using VHDL language is described. Detailed VHDL-model of bit-oriented static RAM is proposed. It helps to discover the nature of faults, to verify the system's behavior in a case of the faults and as result – helps to design more reliable devices.

Литература

1. *Rajski J., Kuyl M.* // Design & Diagnostics of Electronic Circuits & Systems. 1998. P. 203–209.
2. *Lala P.K.* Digital Circuits Testing and Testability. New York, 1997.
3. *Stroud C.E.* A Designer's Guide to Built-In Self-Test. Norwell, 2002.
4. *Ma S., Shaik I., Fetherston R.* // Proc. of IEEE International Test Conf. 1999. P. 587–595.
5. *Ярмолик В.Н., Иванюк А.А.* // Проектирование контролепригодных цифровых устройств. Минск, 2006.
6. *Kielkowski R.W.* SPICE, Practical Device Modeling. New York, 1995.
7. IEEE Standard VITAL ASIC Modeling Specification, IEEE Std 1076.4. 1995.
8. *Бибило П.Н.* Синтез логических схем с использованием языка VHDL. М., 2002.
9. *Ярмолик В.Н., Иванюк А.А.* // Автоматика и Вычислительная Техника. 2007. N 3. С. 3–12.
10. *Jenn E., Arlat J., Rimen M., Ohlsson J., Karlsson J.* // Proc. of the 24th International Symposium on Fault Tolerant Computing. Austin. 1994. P. 66–75.
11. *Sieh V., Tschache O., Balbach F.* // Proc. of 27th International Symposium on Fault-Tolerant Computing. Seattle. 1997. P. 32–36.
12. *Золоторевич Л.А.* // Информатика. 2005. N 3. С. 135–144.
13. *Li J.F., Wu C.W.* // Proc. Ninth IEEE Asian Test Symposium (ATS). 2008. P. 45–50.
14. *Yarmolik V.N., Kilimets Y.V., Goor A.J.* // Proc. of IEEE Int. Workshop on Memory Technology, Design and Testing. Paris, 1996. P. 100–102.
15. *Harutunyan G., Vardanian V.A.* // Proc. of IEEE Int. Workshop EWDTW. Sochi, 2006. P. 68–71.
16. *Goor A.J.* // Proc. of European Design and Test Conference. Paris, 1999. P. 420.
17. *Goor A.J.* // IEEE Design and Test of Computers. 1993. V. 10(1). P. 8–14.
18. ISE WebPACK 9.1i [Electronic resource]. 2008. Mode of access: http://www.xilinx.com.
19. *Goor A.J.* // 18th IEEE VLSI Test Symposium. 2000. P. 281–289.