КОМПЬЮТЕРНАЯ ИНЖЕНЕРИЯ И ТЕХНИЧЕСКАЯ ДИАГНОСТИКА

УДК 519.713:681.326

МОДЕЛИРОВАНИЕ И СИНТЕЗ ТЕСТОВ ДЛЯ ЦИФРОВЫХ СХЕМ С ГРАНИЧНЫМ СКАНИРОВАНИЕМ

ХАХАНОВ В.И., КОЛЕСНИКОВ К.В., ПАРФЕНТИЙ А.Н., ХАХАНОВА И.В.

Предлагаются быстродействующие алгоритмы и процедуры синтеза тестов и моделирования неисправностей, ориентированные на обработку сложных цифровых систем, включающих сотни тысяч эквивалентных вентилей на стадии проектирования. Основаны на использовании новых технологий структурного анализа, позволяющего реконфигурировать модели объектов в процессе синтеза тестов и моделирования неисправностей. Структуры данных ориентируются на компилятивно-интерпретативную реализацию алгоритмов, что делает систему гибкой по отношению к оперативной модификации моделей и быстродействующей в части компилятивного исполнения процедуры транспортирования списков дефектов.

1. Введение

Цель предлагаемого исследования – существенное, на порядок, уменьшение времени синтеза тестов для структурно- и функционально-сложных проектируемых цифровых систем.

Объект тестирования – проектируемые цифровые системы, представленные в виде булевых уравнений на языке VHDL, имплементируемые в кристаллы ПЛИС и содержащие сотни тысяч эквивалентных вентилей.

Предмет исследования – методы моделирования неисправностей и синтеза тестов на структурно-функциональных моделях.

Актуальность. Практически все методы синтеза тестов для цифровых устройств и систем используют для верификации проверяющих последовательностей средства моделирования неисправностей. Они позволяют оценить качество покрытия неисправностей заданного класса генерируемыми тестовыми наборами. Такое оценивание требует значительных временных затрат, иногда неприемлемых, если объект тестирования содержит сотни тысяч вентилей. Подтверждением сказанному могут служить формулы предельной оценки вычислительной сложности параллельного, дедуктивного и совместного алгоритмов анализа неисправностей:

$$C_p = (b^2 / W) \times G^3;$$

$$C_d = b^2 \times q \times G^3;$$

$$C_c = a_0 nV + bnayT,$$

где b – число неэквивалентных неисправностей; W – длина компьютерного слова; G – количество эквивалентных вентилей; q – среднее число вентилей, входы которых активизированы неисправностями на моделируемом тест-наборе; a_0 – фактор активности n модулей схемы, изменивших свои значения на входах при подаче очередного тест-вектора, общее число которых равно V; b – среднее число неисправностей в модуле; a – фактор активности для общего числа дефектов bn; y – фактор топологии, определяющий количество задействованных модулей при выполнении импликации дефекта от места возникновения к выходам устройства; T – общее число тестовых наборов.

Из приведенных оценок видно, что дедуктивный алгоритм менее предпочтителен, если значение q соизмеримо с G. Стоимость совместного алгоритма определяется двумя составляющими: обработкой исправной и неисправной схемы, что и формирует два слагаемых упомянутой оценки. Поэтому вычислительная сложность совместного моделирования для линейных схем равна n^2, для квадратной топологии объекта – $n^{3/2}$. Таким образом, стоимость выполнения совместного алгоритма зависит в большей степени от топологии схемы, а точнее от количества линий сходящихся разветвлений.

Можно выделить наиболее существенные факторы – структурные коллизии, негативно влияющие на быстродействие методов моделирования неисправностей, а значит на эффективность процедур синтеза тестов: 1) наличие в схеме сходящихся разветвлений; 2) глобальные обратные связи; 3) схемная избыточность.

Усилия исследователей на пути повышения быстродействия методов тестирования, как правило, были направлены на изобретение новых методов эффективной обработки упомянутых структурных коллизий. Этому посвящены тысячи научных работ, среди которых можно выделить наиболее результативные:

– классические методы моделирования неисправностей (дедуктивный, параллельный, совместный) [1,2,8,11,14] и синтеза тестов [1,2,8,11,14], а также модели, используемые для транспортирования дефектов и моделирования исправного поведения [3,4-7,10];

– предварительный анализ цифровых систем в целях определения структурных коллизий в комбинационных и последовательностных схемах [8,14,16];

– технологии реконфигурирования цифровых проектов для повышения их тестопригодности в процессе проектирования дискретных объектов [9,10,12].

Современные технологии автоматизированного проектирования допускают средства тестопригодности в процесс синтеза и имплементации, в целях существенной модификации и реконфигурации цифровой системы. Это означает, что можно разрабатывать и имплементировать средства реконфигурирования проектируемых цифровых систем и их моделей в целях уменьшения влияния структурных коллизий, их числа в разрабатываемых изделиях, а в пределе – их устранения из проектов.

В рамках решения упомянутой проблемы предлагается разработать новые технологии:

1) структурного анализа цифровых проектов в целях определения структурных коллизий;

2) реконфигурирования проектируемых изделий и их моделей до- и в процессе обработки;

3) разработки дедуктивно-параллельных методов обратного моделирования константных неисправностей и синтеза тестов;

4) верификации разработанных технологий на примерах структурного, функционального и объемного многообразия проектируемых цифровых систем.

2. Процедура обратной суперпозиции

Прежде всего следует учитывать негативное влияние сходящихся разветвлений на адекватность моделирования неисправностей. Здесь речь идет о возможности рассмотрения одномерных путей активизации неисправностей в схеме. При отсутствии сходящихся разветвлений такая возможность дает повышение быстродействия моделирования в десятки раз. Это связано с тем, что операции над списками транспортируемых дефектов, имеющие вычислительную сложность $b^2 \times q \times G^3$, модифицируются. Взамен приходит процедура обратной суперпозиции (модификация П-алгоритма) на одномерных путях активизации неисправностей древовидных подграфов, вычислительная сложность которой – $q \times G^2$:

1. Моделирование собственных неисправностей примитивов в целях определения входных дефектов, которые проверяются на выходе при заданном тестовом наборе. Результатом такого анализа является вектор проверяемых дефектов элемента

$$L_i = (L_{i1}, L_{i2}, ...L_{ij}, ...L_{in_i}, L_i),$$

входные и выходная координаты которого определены в алфавите $A = \{0,1,X,\varnothing\}$. После обработки всех примитивов получается упорядоченный по номерам выходов список векторов $L = (L_1, L_2, ...L_i, ..., L_n)$, число которых равно количеству примитивов в схеме.

2. Начиная с последнего элемента, выполняется процедура суперпозиции, которая заключается в последовательном объединении векторов из списка L с обобщенным вектором проверяемых дефектов схемы L*, мощность которого равна числу линий схемы n. Априорно значения всех его координат устанавливаются в состояния, определяемые в соответствии с выражением:

$$\overset{n}{\underset{j=1}{\forall}} L_j^* = \varnothing .$$

Затем выполняется объединение всех векторов примитивов, которые имеют внешние выходы на разъем: $L = L \cup L_i^Y$.

Очередной кандидат из списка L должен быть: во-первых, предшественником для вектора уже обработанного примитива, во-вторых, пересечение кандидата по его выходной координате с вектором L* должно иметь непустое пересечение:

$$L^* = L^* \cup L_i \leftarrow L_j^* \cap L_{ij}^Y \neq \varnothing .$$

Если приводить процедуру к параллельным булевым операциям, то необходимо прежде всего выполнить кодирование символов алфавита двумя разрядами $K(A) = \{0 = 01, 1 = 10, X = 11, \varnothing = 00\}$. Затем, учитывая изоморфизм алгебр логики и множеств, упомянутая формула обратной суперпозиции преобразуется к следующему виду:

$$L^* = L^* \vee L_i \leftarrow L_j^* \wedge L_{ij}^Y \neq \varnothing .$$

3. Анализ структурных коллизий

Однако использованию описанной процедуры обратной суперпозиции препятствует невозможность выполнения одномерной активизации, которая может быть проиллюстрирована следующим примером.

Пример 1. Выполнить анализ схем (рис. 1) в целях определения множества проверяемых дефектов на заданных тест-векторах с помощью процедуры обратной суперпозиции.

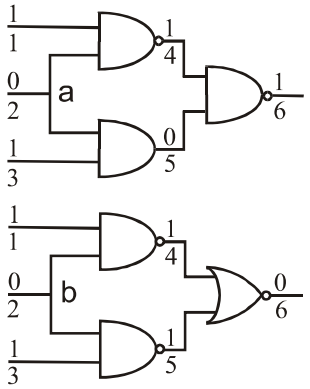

Рис. 1. Схемы ложной проверки и не проверки дефектов

Для обеих схем активизация неисправностей на линиях a и b дает некорректные результаты их проверки.

В верхней схеме константная неисправность 2^1 "проверяется" на одномерном пути 2-5-6, используя процедуру обратного прослеживания. Однако при этом не учитывается тот факт, что этот дефект изменяет состояние линии 4 с 1 на 0, что создает условия запрета транспортирования неисправности на выход схемы. Поэтому неисправность на линии $a - 2^1$ является ложно проверяемой на наборе (101) при рассмотрении ее транспортирования по одномерному пути активизации.

Нижняя схема является примером альтернативной ситуации – к выходу 6 нет одномерного пути транспортирования дефектов с внешних входов и в частности от линии b. Тем не менее неисправность на упомянутой линии – 2^1 проверяется, поскольку ее присутствие создает "кратный" дефект на линиях 4 и 5, который изменяет состояние выходной линии схемы.

В обоих случаях имеется существенная некорректность, которая не позволяет использовать только идею одномерности при отслеживании пути транспортирования неисправностей от выходов ко входам схемы.

Тем не менее, учитывая, что некорректность связана исключительно со сходящимися разветвлениями, необходимо исключить их из модели схемы, сделав ее древовидной. Поскольку такую модификацию сделать невозможно, не нарушая функциональные свойства проектируемого устройства, то следует разделить анализ цифрового устройства на две процедуры. Первая ориентирована на обработку только сходящихся разветвлений, количество которых значительно меньше остальных линий. Вторая – на моделирование неисправностей линий, относящихся к древовидным подграфам.

Фиксированное и незначительное число (порядка 20%) сходящихся разветвлений дает основания для их моделирования дедуктивно-параллельным методом. Предпочтение его использования перед остальными связано с фиксированным количеством СР для конкретной схемы, которое укладывается в целое число форматов машинного слова, ориентированного на технологичные регистровые операции булевой алгебры. Кроме того, параллелизм можно применять и для построения аппаратурных акселераторов моделирования неисправностей, где число разрядов регистров для параллельных операций можно увеличить до нескольких тысяч. Это дает возможность повысить быстродействие моделирования в десятки и сотни раз.

Второй тип структурной коллизии связан с возникновением генераторного режима анализа неисправностей на линиях, принадлежащих контуру обратной связи. Исправное моделирование определяет состояния всех линий устойчивыми и двоичными. Характерным примером возникновения такой коллизии является следующая ситуация.

Пример 2. Выполнить моделирование неисправностей для схемы, представленной на рис. 2.

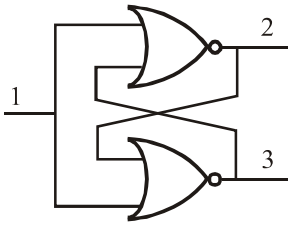

Рис. 2. Схема с обратными связями

Входной набор – 0, предыдущий тест-вектор равен 1. Вектор исправного моделирования схемы в формате переменных (1,2,3) определяется в виде (100).

Дедуктивно-параллельная схема для данного вектора реконфигурируется к следующей структуре (рис. 3).

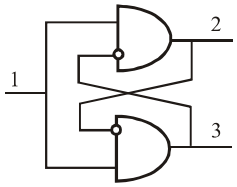

Рис. 3. Дедуктивно-параллельная схема

Результат моделирования неисправностей для набора 100 представлен в виде итеративной матрицы вычисления списков дефектов:

L_j	I_1	I_2	I_3	I_4
$L_2 = L_1 - L_3 + 2^1$	$1^0 2^1$	2^1	$1^0 2^1$	2^1
$L_3 = L_1 - L_2 + 3^1$	$1^0 3^1$	3^1	$1^0 3^1$	3^1

где $I_j (j = \overline{1,4})$ – столбцы итераций, каждый из них содержит списки проверяемых константных неисправностей по выходам 2 и 3, которые вычисляются по формулам, заданным в столбце L_j.

Здесь на четырех итерациях зафиксирован генераторный режим для дефекта 1^0 на выходах 2 и 3, заключающийся в регулярности его появления и исчезновения.

В этом случае необходимо корректировать списки проверяемых неисправностей по следующему правилу: неисправность, однажды исчезнувшая на некоторой итерации моделирования, должна быть исключена из проверяемых на заданном тесте.

Для рассматриваемого примера уже на второй итерации следует исключить из рассмотрения дефект 1^0. Тогда выполнение итерации I_3 дает окончательный список проверяемых на тест-векторе 100 неисправностей: $L = \{2^1, 3^1\}$.

Физическая сущность генераторного режима заключается в отсутствии выходных условий проверки 1^0, поскольку состояния выходов 2 и 3 при отсутствии и наличии данной неисправности неразличимы:

$$[(T = 100) \Rightarrow (Y_2 = 0) \,\&\, (Y_3 = 0)] \cap$$

$$\cap [(T(1^0) = 0XX) \Rightarrow (Y_2 = X) \,\&\, (Y_3 = X)] \neq \varnothing.$$

Пример 3. Выполнить дедуктивно-параллельное моделирование неисправностей на входном наборе 01 для триггерной схемы, изображенной на рис. 4.

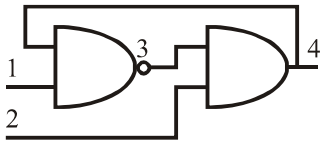

Рис. 4. Триггерная схема

Дедуктивно-параллельная схема для набора 0111 имеет вид, представленный на рис. 5.

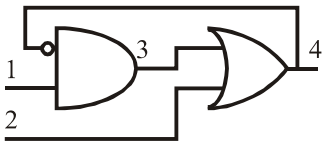

Рис. 5. Дедуктивно-параллельная схема

Она получена путем реконфигурирования триггерной структуры (см. рис.4) на тест-векторе 0111. На данном наборе также возникает генераторный режим при наличии дефекта 1^1:

L_j	I_1	I_2	I_3	I_4
$L_3 = L_1 - L_4$	$1^1 3^0$	3^0	$1^1, 3^0$	3^0
$L_4 = L_3 + L_2$	$1^1 2^0 3^0 4^0$	$2^0 3^0 4^0$	$1^1 2^0 3^0 4^0$	$2^0 3^0 4^0$

Окончательный список проверяемых на тест-векторе 0111 дефектов с учетом корректировки имеет вид $L_4 = \{2^0, 3^0, 4^0\}$.

Но здесь уже физическая причина исключения 1^1 состоит в инициировании сохранения состояния на линиях 3 и 4 при возникновении упомянутого дефекта, что означает неразличимость проявления неисправности на внешних выходах схемы. В самом деле, вектор исправного поведения равен 0111, а наличие дефекта приводит к состоянию линий схемы в виде 1111. Иначе, на линиях 3 и 4 не наблюдается проявление неисправности в форме различных сигналов при отсутствии и наличии дефекта.

Таким образом, существуют две структурные коллизии, проявляющиеся в виде генераторного режима списков неисправностей на двух соседних итерациях, в контурах обратных связей, которые являются следствием наличия фиктивно проверяемой неисправности. Последняя должна быть исключена из списка обнаруживаемых на рассматриваемом тест-векторе, поскольку:

1) состояния наблюдаемых выходов при отсутствии и наличии данной неисправности неразличимы;

2) возникает генераторный режим исправного поведения, приводящий к неопределенному статическому состоянию наблюдаемых выходов.

4. Реконфигурирование модели для синтеза тестов

Использование технологии граничного сканирования [17] дает возможность повысить качество проектируемого изделия с точки зрения упрощения процедур синтеза тестов. Для этого необходимо расширить идею граничного сканирования внешних контактов (рис. 6) микросхемы (чипа) на линии сходящихся разветвлений.

В этом случае они будут не только наблюдаемы относительно выхода сканирования данных TDO, но и доступны для подачи на них тестовых последовательностей со входа TDI. В результате такого дополнения комбинационная схема будет представлять совокупность древовидных структур, имеющих корневыми вершинами внешние выходы или наблюдаемые линии сходящихся разветвлений.

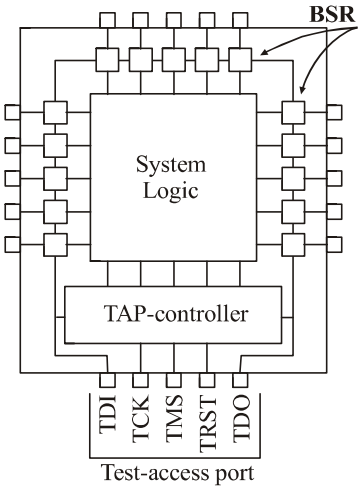

Рис. 6. Ячейка BSR-регистра

Однако здесь самое интересное то, что для таких подсхем существуют эффективные алгоритмы детерминированной генерации тестов [18] и их анализа методом обратного прослеживания [7], которые отличаются от классических по быстродействию как минимум на порядок. Тем не менее имеется плата за столь существенные преимущества применения граничного сканирования к реконфигурированию проектируемой цифровой системы. Ее следует отнести к недостаткам использования упомянутой технологии, которые определяются:

1) отсутствием поддержки расширения BSR (Boundary Scan Register) существующим стандартом IEEE [17];

2) увеличением времени тестирования реального цифрового устройства или его модели в nt раз, где n и t – число сходящихся разветвлений и тест-векторов;

3) расширением длины регистра BSR на количество сходящихся разветвлений путем синтеза дополнительных ячеек;

4) увеличением времени проверки межсоединений на плате цифрового устройства на nt раз, где n и t – число сходящихся разветвлений и тестовых наборов для верификации печатного монтажа;

5) некоторым несущественным, но увеличением длины межсоединения сходящегося разветвления со входами последующих элементов, что является следствием установки ячейки BSR (рис. 7), приводящей к уменьшению быстродействия системной логики проектируемого устройства на величину задержки цепи X/Y–Y/X.

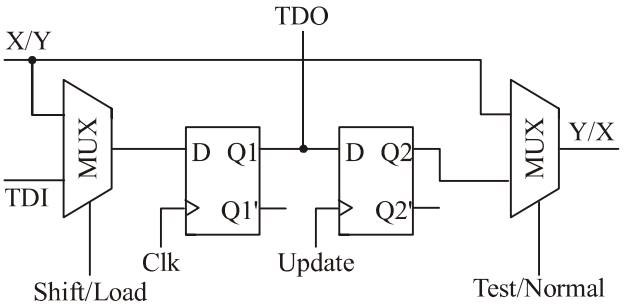

Рис. 7. Ячейка BSR-регистра

Учитывая, что время тестирования проекта или устройства пропорционально длине проверяющих последовательностей, актуальной представляется задача не только синтеза полного относительно класса ОКН теста, но и его минимизация. Поэтому возникает новая проблема адаптации существующих алгоритмов детерминированной генерации тестов, которые дают наилучший результат в смысле минимальности длины входных наборов. Далее рассматривается кубический метод синтеза тестов, основанный на использовании кубического исчисления и сокращенных таблиц истинности примитивов, входящих в состав цифрового проекта. Его основными положениями являются:

1. Синтез тестов для функционально законченных фрагментов комбинационных схем, заданных кубическими покрытиями.

2. Генерация тестов по структуре комбинационного устройства методом синтеза минимального множества путей одномерной активизации, покрывающих все переменные или линии в схеме.

3. Активизация не дефектов, а линий, входящих в состав логического пути, на паре соседних тестовых последовательностей.

4. Использование символов неопределенности X при выполнении фазы активизации (Forward Propagation) входов примитивов в целях предоставления возможности доопределения последних символами 0 и 1 в фазе обеспечения (Backward Implication). Это дает существенное уменьшение противоречий, возникающих на обратном ходе алгоритма активизации одномерного пути. Иногда это приводит к построению многомерного пути активизации, если одномерного не существует. Иллюстрацией сказанному может служить пример, представленный на рис. 8.

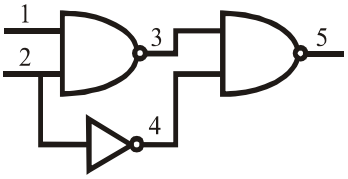

Рис. 8. Создание двумерной активизации

Здесь активизация пути 1-3-5 (01-10-01) обеспечивается сигналами X1 и 1X на линиях 2 и 4 для первого и второго элементов, которые не являются противоречивыми после выполнения обратной импликации:

1	2	3	4	5
0	X	1	1	0
1	1	0	X	1

.

Доопределение линии 4 (1X) через вход 2 (X1) создает активность для входа 2 (10), которая является следствием невозможности активизации переменной 1 ввиду ее избыточности. Поэтому кубический алгоритм на обратном ходе создает дополнительный путь для активизации выхода 4 изменением единственно возможного, в данном случае, входа 2:

1	2	3	4	5
0	0	1	1	0
1	1	0	0	1

.

При задании жестких условий активизации [], исключающих наличие символов X, процедура построения одномерного пути 1-3-5 имеет неудовлетворительный результат, поскольку на прямой фазе состояния линий 2 и 4, связанные инверсным отношением, получаются противоречивыми:

1	2	3	4	5
0	1	1	1	0
1	1	0	1	1

.

5. Разрешимость коллизий путем выполнения активизации – прямой фазы – от выхода ко входу схемы. В данном случае убедительным примером может служить невозможность построения заказанного пути активизации, построенного по стратегии от входа схемы к его внешнему выходу (рис. 9).

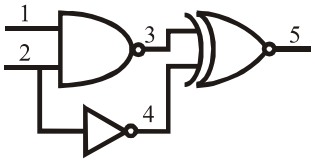

Рис. 9. Невозможность активизации линии 2

Для активизации линии 2 не существует условий ее обеспечения по входам 1 и 2:

1	2	3	4	5
X	0	1	1	0
1	1	0	0	0

.

Это связано с противоречием, возникающим на прямой фазе алгоритма активизации, что в свою очередь является следствием избыточности схемы:

$$Y_5 = \overline{X_1 X_2} \oplus \overline{X}_2 = \overline{X_1 X_2} X_2 \vee X_1 X_2 \overline{X}_2 = \overline{X_1 X_2} X_2 = (\overline{X}_1 \vee \overline{X}_2) X_2 = \overline{X}_1 X_2 \vee \overline{X}_2 X_2 = \overline{X}_1 X_2.$$

Тем не менее, если выполнять фазу активизации от выходов ко входам, то удалось бы избежать пустого решения, поскольку относительно линии 3 существует путь, активизирующий выход схемы:

1	2	3	4	5
0	X	1	0	1
1	1	0	0	0

→

1	2	3	4	5
0	1	1	0	1
1	1	0	0	0

Данный пример аргументирует необходимость параллельного (совместного) выполнения фаз активизации и обеспечения методом обратного прослеживания.

5. Функциональный К-алгоритм синтеза тестов

Он инвариантен по отношению к сходящимся разветвлениям, поскольку последние находятся внутри функционального элемента, представленного кубическим покрытием. Основная идея синтеза тестов заключается в использовании уравнения тестирования [2]: $T \oplus F = L$.

Другими словами, кубическое покрытие списков неисправностей L для вектора T и покрытия примитива C вычисляется на основе использования бинарной координатной операции XOR ($\oplus$), определяющей взаимодействие компонентов T, C, L в троичном алфавите: $T_j \oplus F_{ij} =$

$\oplus$	0	1	X
0	0	1	X
1	1	0	X
X	X	X	X

.

Алгоритм синтеза теста проверки одиночных константных неисправностей одновыходового примитива, задаваемых регулярным диагональным L^1 -покрытием списка дефектов всех переменных функционального элемента (единица на входной и выходной координате каждого куба обозначает активизацию выхода Z при наличии неисправности любого знака на входе):

$$L^1 = \left|\begin{array}{ccccc} X_1 & X_2 & ...X_j... & X_m & Z \\ \hline 1 & 0 & ...0... & 0 & 1 \\ 0 & 1 & ...0... & 0 & 1 \\ ... & ... & ... & ... & ... \\ 0 & 0 & ...1... & 1 & 1 \\ ... & ... & ... & ... & ... \\ 0 & 0 & ...0... & 1 & 1 \end{array}\right|,$$

содержит следующие пункты:

1. Формирование кандидатов в тест $T_t^k \in T^k$ путем выполнения операций над векторами из L^1, F:

$$T^k = L^1 \oplus F. \qquad (1)$$

2. Получение теста T ($T_t \in T$) из множества кандидатов T^k на основе анализа выражения

$$T = T \cup T_t = F_i \cap T_t^k \leftarrow \exists i(F_i \cap T_t^k \neq \varnothing). \qquad (2)$$

3. Минимизация множества тест-векторов T путем применения операции поглощения:

$$T = T \setminus T_r \leftarrow T_i \cap T_r = T_r. \qquad (3)$$

4. Для получения еще более минимального теста необходимо выполнить процедуру моделирования неисправностей на функциональной или структурной модели устройства и на полученной многозначной таблице неисправностей [19] решить задачу получения минимального покрытия неисправностей построенным тестом.

Следующий пример иллюстрирует основные шаги описанного алгоритма.

Пример 4. Построить минимальный тест для функционального трехвходового элемента, представленного кубическим покрытием:

$$F = \left|\begin{array}{cccc} X_1 & X_2 & X_3 & Y \\ \hline X & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ X & 0 & 1 & 0 \\ X & 1 & 0 & 0 \\ 0 & 1 & X & 0 \end{array}\right|. \qquad (4)$$

На первом шаге выполняется $\oplus$ -операция между кубическим покрытием и векторами из множества L^1 :

$$F\left|\begin{array}{cccc} X_1 & X_2 & X_3 & Y \\ \hline X & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ X & 0 & 1 & 0 \\ X & 1 & 0 & 0 \\ 0 & 1 & X & 0 \end{array}\right| \oplus L\left|\begin{array}{cccc} X_1 & X_2 & X_3 & Y \\ \hline 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{array}\right| = \{F \oplus L_1, F \oplus L_2, F \oplus L_3\}.$$

Данная операция приводит к получению трех подмножеств кубов, которые являются кандидатами в тест:

$$T^k = \left|\begin{array}{cccc} X_1 & X_2 & X_3 & Y \\ \hline X & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ X & 0 & 1 & 1 \\ X & 1 & 0 & 1 \\ 1 & 1 & X & 1 \end{array}\right| \cup \left|\begin{array}{cccc} X_1 & X_2 & X_3 & Y \\ \hline X & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ X & 1 & 1 & 1 \\ X & 0 & 0 & 1 \\ 0 & 0 & X & 1 \end{array}\right| \cup \left|\begin{array}{cccc} X_1 & X_2 & X_3 & Y \\ \hline X & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ X & 0 & 0 & 1 \\ X & 1 & 1 & 1 \\ 0 & 1 & X & 1 \end{array}\right|.$$

Каждый куб из полученных подмножеств проверяется на его соответствие правилу (2). Кубы, удовлетворяющие упомянутому критерию, формируют тест:

$$T = \left|\begin{array}{cccc} X_1 & X_2 & X_3 & Y \\ \hline 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{array}\right| \cup \left|\begin{array}{cccc} X_1 & X_2 & X_3 & Y \\ \hline X & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ X & 1 & 1 & 1 \\ X & 0 & 0 & 1 \end{array}\right| \cup \left|\begin{array}{cccc} X_1 & X_2 & X_3 & Y \\ \hline X & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ X & 0 & 0 & 1 \\ X & 1 & 1 & 1 \end{array}\right|.$$

Первая стадия минимизации заключается в исключении дублирующих векторов в соответствии с (3). Затем выполняется доопределение символов X путем их замены таким двоичным значением, которое позволит:

1) исключить полученный вектор из теста по правилу избыточности (3);

2) минимизировать кодовое расстояние между соседними тестовыми наборами;

3) повысить качество теста – количество неисправностей, проверяемых рассматриваемым вектором.

Описанная фаза минимизации дает результат:

$$T_{min} = \left|\begin{array}{cccc} X_1 & X_2 & X_3 & Y \\ \hline 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \\ X & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ X & 1 & 1 & 1 \\ X & 0 & 0 & 1 \\ X & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \end{array}\right| = \left|\begin{array}{cccc} X_1 & X_2 & X_3 & Y \\ \hline 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{array}\right|.$$

Далее можно уменьшить длину теста T_{min} путем построения многозначной таблицы неисправностей с помощью программы анализа дефектов и решения задачи получения минимального покрытия неисправностей тестовыми наборами. Здесь процесс моделирования выполняется на вентильной реализации кубического покрытия (рис.10), представленной в виде дизъюнктивной нормальной формы, записанной по конституентам единицы таблицы (4).

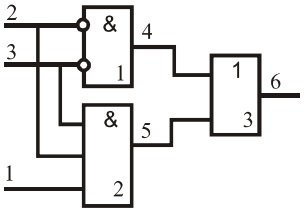

Рис. 10. Структурная реализация функции

Для рассматриваемого примера такие действия приводят к следующему результату (таблица):

1 2 3 4 5 6	1 2 3 4 5 6	Fault list
0 1 1 0 0 0	1 – – 1 1 1	$L_6 = \{4^1 5^1 6^1\}, L_4 = \{4^1\}, L_5 = \{1^1 5^1\}$
1 1 1 0 1 1	0 0 0 – 0 0	$L_6 = \{5^0 6^0\}, L_4 = \{4^1\}, L_5 = \{1^0 2^0 3^0 5^0\}$
1 1 0 0 0 0	– 1 1 1 1 1	$L_6 = \{4^1 5^1 6^1\}, L_4 = \{2^1 3^1 4^1\}, L_5 = \{3^1 5^1\}$
1 0 1 0 0 0	– 1 0 1 1 1	$L_6 = \{4^1 5^1 6^1\}, L_4 = \{3^0 4^1\}, L_5 = \{2^1 5^1\}$
1 0 0 1 0 1	– 1 1 0 – 0	$L_6 = \{4^0 6^0\}, L_4 = \{2^1 3^1 4^0\}, L_5 = \{5^1\}$

В правой части таблицы представлены результаты моделирования неисправностей отдельных примитивов. В среднем столбце показаны проверяемые тест-векторами неисправности константного типа, которые получены с помощью процедуры обратной суперпозиции.

Далее получение минимального теста основывается на выборе таких наборов, которые в совокупности проверят на каждой линии неисправности обоих знаков. Таким тестом здесь является подмножество наборов: $T_{min} = \{1,2,5\}$.

В общем случае существует формальная процедура синтеза минимального теста [20], основанная на преобразовании конъюнктивной формы записи покрытия неисправностей по столбцам в дизъюнктивную форму в целях последующего получения минимальных термов минимальной функции.

Пусть имеется тест, содержащий 6 наборов, покрывающий все неисправности, представленные в схеме:

Test	F_1	F_2	F_3	F_4	F_5	F_6
t_1	1	1	.	.	1	.
t_2	.	1	1	.	.	.
t_3	1	.	.	1	.	.
t_4	.	.	.	.	1	1
t_5	.	.	1	.	.	1
t_6	1	.	1	.	.	.

Необходимо получить минимальное множество наборов, не уменьшающее покрытие неисправностей. Для этого записывается произведение дизъюнкций в соответствии со столбцами таблицы и выполняется преобразование к минимальной дизъюнктивной форме на основе правила поглощения ($ab \vee a = a$):

$$T = (t_1 \vee t_3 \vee t_6)(t_1 \vee t_2)(t_2 \vee t_5 \vee t_6)(t_3)(t_1 \vee t_4)(t_4 \vee t_5) =$$
$$= (t_1 \vee t_1t_3 \vee t_1t_6 \vee t_1t_2 \vee t_2t_3 \vee t_2t_6)(t_2t_3 \vee t_3t_5 \vee t_3t_6) \wedge$$
$$\wedge (t_1t_4 \vee t_1t_5 \vee t_4 \vee t_4t_5) =$$
$$= (t_1 \vee t_2t_3 \vee t_2t_6)(t_2t_3 \vee t_3t_5 \vee t_3t_6)(t_1t_5 \vee t_4) =$$
$$= (t_1t_2t_3 \vee t_1t_3t_5 \vee t_1t_3t_6 \vee t_2t_3 \vee t_2t_3t_5 \vee t_2t_3t_6 \vee$$
$$\vee t_2t_3t_6 \vee t_2t_3t_5 \vee t_2t_3t_6)(t_1t_5 \vee t_4) =$$
$$= (t_1t_3t_5 \vee t_1t_3t_6 \vee t_2t_3)(t_1t_5 \vee t_4) =$$
$$= (t_1t_3t_5 \vee t_1t_3t_5t_6 \vee t_1t_2t_3t_5 \vee t_1t_3t_4t_5 \vee t_1t_3t_4t_6 \vee t_2t_3t_4) =$$
$$= (t_1t_3t_5 \vee t_1t_3t_4t_6 \vee t_2t_3t_4).$$

Полученный результат формирует все возможные минимальные покрытия в виде трех конъюнктивных термов. Лучшим решением является терм с минимальным числом букв, здесь таких два: $T = t_1t_3t_5 \vee t_2t_3t_4$.

6. Совместный К-алгоритм синтеза тестов

Ориентирован на генерацию тестов для комбинационных и последовательностных схем, где фазы активизации и обеспечения выполняются совместно и в направлении от выходов схемы к ее входам. Стратегия алгоритма заключается в построении множества одномерных путей активизации, покрывающих все линии схемы. Основные шаги алгоритма [18]:

1. Выбор очередного пути активизации от внешнего выхода ко входу, который покрывает еще не обработанные линии.

2. Выполнение фазы продвижения активности от выходов ко входам путем определения для каждого примитива пары кубов, обеспечивающих изменение выхода при наличии активного входа.

3. Доопределение неактивных координат примитивов путем выполнения обратной импликации на всех элементах, участвующих в формировании условий активизации на внешних входах схемы.

4. Минимизация полученного теста путем применения правил и процедур, рассмотренных ранее в подразд. 5, при описании функционального К-алгоритма.

Шаги 2 и 3 выполняются совместно во времени, что возможно благодаря предварительному ранжированию линий и элементов схемы.

Рассмотрим предложенный алгоритм на примере построения теста для функционального модуля анализа дефектов, представленного на рис. 11.

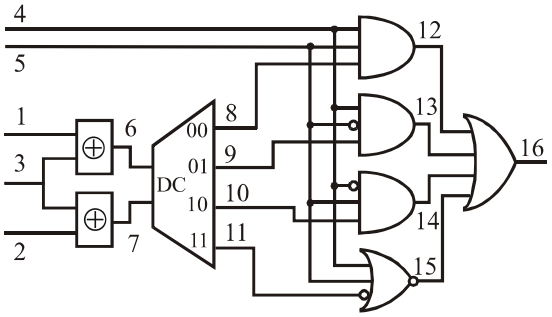

Рис. 11. Структурная схема симулятора

Результат выполнения активизации входа 4 до внешнего выхода представлен в форме пары векторов, определенных в троичном алфавите на всех 16 линиях схемы функционального примитива:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0	0	0	0	X	0	0	X	0	0	0	0	0	0	0	0
X	X	X	1	1	X	X	1	X	X	X	1	X	X	X	1

Доопределение символов X по входным координатам дает пару двоичных наборов, которые необходимо верифицировать путем выполнения исправного моделирования, что дает следующий результат:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0
0	0	0	1	1	0	0	1	0	0	0	1	0	0	0	1

Аналогично выполнив активизацию остальных линий, получим следующий результат в виде пар наборов:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0	0	0	X	0	0	0	X	0	0	0	0	0	0	0	0
X	X	X	1	1	X	X	1	X	X	X	1	X	X	X	1
0	0	0	X	X	0	0	1	0	0	0	0	0	0	0	0
1	0	0	0	1	1	0	0	X	1	X	X	X	1	X	1
0	0	1	0	0	1	1	0	0	1	0	0	0	0	0	0
0	1	1	X	X	1	0	X	X	0	1	X	X	X	1	1
1	0	0	X	X	1	0	0	0	1	0	0	0	0	0	0
1	0	1	1	0	0	1	0	1	0	0	X	1	X	X	1

Выполним доопределение входных символов X по описанным правилам, затем – моделирование дво-

ичных векторов, которые объединим с ранее полученной парой наборов для активизации переменной 4. В результате получим следующий тест:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0
0	0	0	1	1	0	0	1	0	0	0	1	0	0	0	1
0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0
1	0	0	0	1	1	0	0	0	1	0	0	0	1	0	1
0	0	1	0	0	1	1	0	0	1	0	0	0	0	0	0
0	1	1	0	0	1	0	0	0	0	1	0	0	0	1	1
1	0	0	1	0	1	0	0	0	1	0	0	0	0	0	0
1	0	1	1	0	0	1	0	1	0	0	0	1	0	0	1
0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0
0	0	0	1	1	0	0	1	0	0	0	1	0	0	0	1

Моделирование неисправностей подтвердило 100% качество сгенерированного теста, но обладающего некоторой избыточностью (векторы 9, 10).

Предложенные методы и алгоритмы реализованы в программу синтеза тестов и интегрированы в автоматическую систему тестирования SIGETEST.

7. Заключение

Представлена стратегия тестирования сложных проектов цифровых систем, основанная на использовании технологии граничного сканирования сходящихся разветвлений, доставляющих существенные неудобства в процессе синтеза тестов и моделирования неисправностей константного типа. Стратегия позволяет на порядок уменьшить затраты времени при генерации детерминированных тестов и анализе их качества. В целях повышения быстродействия алгоритмов тестирования предложены решения, имеющие научную новизну:

1. Процедура обратной суперпозиции для древовидных структур, позволяющая в десятки раз повысить быстродействие моделирования неисправностей для сложных цифровых проектов.

2. Анализ структурных коллизий, связанных с обработкой сходящихся разветвлений и обратных связей в целях нахождения лучшего решения для их устранения.

3. Реконфигурирование модели цифрового устройства для моделирования и синтеза тестов на основе расширения длины регистра сканирования данных, что позволяет в несколько раз сократить время синтеза детерминированных тестов методом одномерной активизации.

4. Функциональный К-алгоритм синтеза тестов, эффективно обрабатывающий цифровые устройства, представленные в форме кубического покрытия. Вычислительная сложность алгоритма пропорциональна квадрату числа переменных.

5. Совместный К-алгоритм синтеза тестов для цифровых структур, отличающийся от алгоритма Рота меньшим числом переборов за счет введения неопределенности в условия активизации одномерных путей.

Литература: **1.** *Hahanov V.I., Babich A.V., Hyduke S.M.* Test Generation and Fault Simulation Methods on the Basis of Cubic Algebra for Digital Devices. Proceedings of the Euromicro Symposium on Digital Systems Design DSD2001. Warsaw, Poland. September, 4-6, 2001. P. 228-235. **2.** *Хаханов В.И., Хак Х.М. Джахирул, Масуд М.Д. Мехеди.* Модели анализа неисправностей цифровых систем на основе FPGA, CPLD // Технология и конструирование в электронной аппаратуре. 2001. № 2. С. 3-11. **3.** *Хаханов В.И., Сысенко И.Ю., Хак Х.М. Джахирул, Масуд М.Д. Мехеди.* Кубическое моделирование неисправностей цифровых проектов на основе FPGA, CPLD // Радиоэлектороника, информатика, управление. 2001. № 1. С. 123-129. **4.** *Levendel Y.H., Menon P.R.* Comparison of fault simulation methods – Treatment of unknown signal values // Journal of digital systems. Vol. 4. 1980. P. 443-459. **5.** *Abramovici M., Breuer M.A. and Friedman A.D.* Digital System Testing and Testable Design, Computer Science Press, 1998. 652 p. **6.** *Хаханов В.И.* Техническая диагностика элементов и узлов персональных компьютеров. К.: ІЗМН. 1997.308 с. **7.** *Убар Р.Р.* Анализ диагностических тестов для комбинационных цифровых схем методом обратного прослеживания неисправностей // Автоматика и телемеханика. 1977. №8. С.168-176. **8.** *Семенец В.В., Хаханова И.В., Хаханов В.И.* Проектирование цифровых систем с использованием языка VHDL. Харьков: ХНУРЭ, 2003. 492 с. **9.** *Хаханов В.И., Соколов А.Ю., Чамян А.Л.* Детерминированный метод генетических алгоритмов для синтеза тестов верификации цифровых систем // Радиоэлектороника и информатика. 2001. № 4. С. 125-129. **10.** *Хаханов В.И., Сысенко И.Ю., Колесников К.В.* Дедуктивно-параллельный метод моделирования неисправностей на реконфигурируемых моделях цифровых систем // Радиоэлектороника и информатика. 2002. № 1. С. 98-105. **11.** *Хаханов В.И., Колесников К.В., Хаханова А.В.* BDP-метод моделирования неисправностей для синтеза тестов цифровых проектов // Радиоэлектороника и информатика. 2002. № 2. С. 60-66. **12.** *Бондаренко М.Ф., Кривуля Г.Ф., Рябцев В.Г., Фрадков С.А., Хаханов В.И.* Проектирование и диагностика компьютерных систем и сетей. К.: НМЦ ВО. 2000. 306 с. 13. *Active*-VHDL Series. Book #1 - #4. Reference Guide. ALDEC Inc. 1998. 206 p. **14.** *Хаханов В.И., Убар Р.-Й.Р.* Технологии проектирования систем на кристаллах. Моделирование неисправностей сверхбольших цифровых проектов // АСУ и приборы автоматики. 2002. Вып.122. С. 16-35. **15.** *Jutman A., Ubar R., Hahanov V., Skvortsova O.* Practical works for on-line teaching design and test of digital circuits. Proceedings of the 9-th IEEE International Conference on Electronics, Circuits and System ICECS 2002.Dubrovnik, Croatia, September 15-18, 2002. P. 1223-1227. **16.** *Hahanov V.I., Skvortsova O.B., Sysenko I.Y., Chamyan H.L.* ATPG System and Test Generation Methods for Digital Devices. 8-th Bienuual Baltic Electronics Conference. Tallinn, Estonia. October 6-9, 2002. P. 299-302. 17. *IEEE* Std 1149.1- 2001, IEEE Standard Test Access Port and Boundary Scan Architecture, IEEE. 2001. 82p. **18.** *Хаханов В.И., Сысенко И.Ю., И.М. Абу Занунех Халиль.* Проектирование тестов для структурно-функциональных моделей цифровых схем // Радиоэлектроника и информатика. 1999. №.3. С. 51-59. **19.** *Хаханов В.И., Ханько В.В, Бедратый Р.А.* Структурный анализ многозначных таблиц неисправностей для диагностирования цифровых устройств // АСУ и приборы автоматики, 1998. Вып. 107. С. 35-43. **20.** *Автоматизация* диагностирования электронных устройств / Ю.В. Малышенко и др./ Под ред.В.П.Чипулиса. М.: Энергоатомиздат, 1986. 216с.

Поступила в редколлегию 16.01.2003

Рецензент: д-р техн. наук, проф. Кривуля Г.Ф.

Хаханов Владимир Иванович, д-р техн. наук, профессор кафедры АПВТ ХНУРЭ. Научные интересы: техническая диагностика вычислительных устройств, систем, сетей и программных продуктов. Увлечения: баскетбол, футбол, горные лыжи. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел. 70-21-326. E-mail: hahanov@kture.kharkov.ua

Колесников Константин Васильевич, ст. преподаватель кафедры КС Черкасского государственного технологического университета. Научные интересы: тестирование цифровых систем и сетей. Увлечения: живопись – постимпрессионизм, поэзия – серебряный век, символизм. Адрес: Украина, 18006, Черкассы, бул. Шевченко, 460, тел. 43-74-28, e-mail: kvvk601@mail.ru

Парфентий Александрдр Николаевич, студент ХНУРЭ. Научные интересы: техническая диагностика вычислительных устройств. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел. 70-21-326.

Хаханова Ирина Витальевна, канд. техн. наук, доцент кафедры АПВТ ХНУРЭ. Научные интересы: техническая диагностика и проектирование вычислительных устройств. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел. 70-21-326.