ПОСТРОЕНИЕ ТЕСТОВ КОНТРОЛЯ ЦИФРОВЫХ СИСТЕМ: ПРОБЛЕМЫ И РЕШЕНИЯ

Л. А. Золоторевич, А. В. Ильинкова

Белорусский государственный университет
Минск, Беларусь
E-mail: zolotorevichla@bsu.by

Дается анализ состояния проблемы контроля сверхбольших сложнофункциональных интегральных схем. Рассматриваются особенности построения тестов контроля цифровых устройств, описанных на языке VHDL на структурном уровне, методами случайного поиска. Предлагается метод направленного построения тестов контроля цифровых систем, представленных на уровне межрегистровых передач.

Ключевые слова: цифровая система, генерация теста, функциональная неисправность, уровень межрегистровых передач, автоматная модель.

СОСТОЯНИЕ ПРОБЛЕМЫ ПОСТРОЕНИЯ ТЕСТОВ КОНТРОЛЯ

Проблема построения тестов для сложнофункциональных электронных систем является наиболее наукоемкой во всем спектре проблем проектирования. Данная проблема на сегодняшний день еще далека от эффективного теоретического и практического решения. Задача построения тестов является NP-сложной и находится в экспоненциальной сложности от числа элементов объекта контроля. Современные цифровые системы имеют порядка миллиардов транзисторов, в переводе на простые логические элементы это сотни миллионов элементов. Учитывая, что это объекты с памятью, то объем диагностического эксперимента намного превышает 2^{100} при использовании полного теста (при n =100, где n – число внешних входов системы). Очевидна невозможность применения полного теста. В соответствии с Международными технологическими стандартами количество диагностов при проектировании сложных объектов превосходит количество разработчиков в 2–3 раза. Поэтому актуальна проблема разработки методов построения тестов для верификации проектов и оценки степени тестопригодности объекта.

Следует заметить, что бурно развивающаяся электронная отрасль выставляет все новые требования и условия к задаче построения тестов. Если в конце прошлого века рассматривалась задача эффективного построения теста контроля на уровне заданной структуры объекта, то в настоящее время кроме данной задачи сформулирована и ведется поиск решения задачи построения тестов для систем, представленных в разных системах идентификации [1–4].

В работе решается задача построения теста контроля на начальном этапе проектирования, когда отсутствует структура объекта, а объект представлен на уровне межрегистровых передач. Такая постановка задачи возникла из естественного желания пораньше узнать способность проектируемой системы к контролю, т. е. оценить контролепригодность проекта, снизить размерность задачи и сократить сроки проектирования.

МАТЕМАТИЧЕСКАЯ ПЛАТФОРМА

В указанной выше постановке задачи структура объекта отсутствует. Имеется описание поведения системы на уровне RTL, который является промежуточным между системным уровнем и уровнем структурного представления. Здесь весьма ответственным моментом является выбор моделей рассматриваемых функциональных неисправностей. При этом целесообразно работать в более широком диапазоне моделей, если нет доказательной базы для выбора некоторой модели из известного диапазона. В литературе известны предложения по выбору неисправностей как при тестировании программ. Рассматриваются модели выпавшего оператора, замена условного перехода безусловным, замена одной операции некоторой другой, применение неисправности константного типа к некоторой переменной, сигналу, к некоторому разряду переменной и т. д. В данной работе кроме указанных моделей будет использоваться и модель функциональной неисправности, которая аргументированно соответствует неисправности константного типа реального объекта.

В теории тестового диагностирования известны методы направленного и случайного построения тестов. Если рассматривать возможность их реализации в указанной выше постановке, то следует отметить весьма существенную особенность. Если методы построения теста случайным образом можно пытаться реализовать на основе имеющихся фирменных компиляторов языка VHDL, то для направленного построения тестов необходимо знание внутреннего представления программного кода, которое, к сожалению, закрыто для использования извне.

С учетом сложившейся традиции цифровая система рассматривается на уровне RTL как две подсистемы – операционная, выполняющая преобразование данных в соответствии с заданными алгоритмами, и управляющая, реализующая управление операционной частью. Поэтому в качестве математической платформы для описания цифровой системы будем использовать описание в виде графов потоков данных и потока управления. Граф потока управления – это ориентированный граф с узлами, соответствующими операторам программного кода, и ребрами, указывающими порядок выполнения операторов. Граф потока данных – это ориентированный граф с узлами, соответствующими выполняемым операциям и ребрами, указывающими последовательность операций по преобразованию некоторой переменной или сигнала. При этом строится один граф потока управления и столько графов потока данных, сколько переменных описывают полное состояние моделируемой системы. Структура графов приведена в работе [4]. Построение графов осуществляется при статическом анализе программного кода, который осуществляется при его синтаксическом анализе.

Научная гипотеза, используемая при разработке метода генерации тестов на RTL-уровне, формулируется так: для того чтобы найти входные данные, которые позволят определить по выходным данным наличие или отсутствие некоторой неисправности в системе, представленной в рамках любой системы идентификации, необходимо активизировать неисправность, т. е. заставить ее проявиться на выходе некоторого элемента системы, затем заставить ее проявиться хотя бы на одном из выходов системы, после чего необходимо вычислить все входные данные, которые позволят сохранить условия, полученные при решении данной задачи. Гипотеза сформулирована на основе идеи Рота, реализованной при построении тестов контроля объекта на структурном уровне.

МЕТОД НАПРАВЛЕННОГО ПОСТРОЕНИЯ ТЕСТОВ

Предлагается метод направленного построения тестов контроля цифровых систем, описанных на уровне RTL на языке VHDL. Общая идея метода заключается

- в переходе от системы арифметических уравнений, описывающих поведение объекта, к построению системы КНФ булевых функций разрешения;
- конъюнктивном объединении функций разрешения;
- решении задачи выполнимости результирующей КНФ разрешения объекта.

Рассмотрим фрагмент некоторого программного кода, приведенный на рис. 1. Предположим, что все входные переменные являются целочисленными размерностью n бит (mod 2^n). Для генерации тестов необходимо:

1) На основе программного кода объекта составить систему арифметических уравнений, описывающих функционирование объекта;

2) Выполнить корректировку системы с учетом внесения неисправностей соответствующего оператора;

3) Составить систему КНФ булевых функций, соответствующих системе уравнений, полученных в п. 2.

```
if   S = '0' then
          D =  A;
     else  D =  B;
          G = B + C;
          E = D + C;
          F = E * G;
          L = D < G;
endif;
```

Рис. 1. Фрагмент программного кода

```
D = /S*A + S*B;
G = B + C;
E = D + C;
F = E * G;
L = D < G;
```

Рис. 2. Система арифметических уравнений (mod 2^n)

На рис. 2 приведена система арифметических уравнений, описывающих функционирование объекта, представленного программным кодом, приведенным на рис. 1. Положим, что необходимо построить тест, проверяющий оператор целочисленного сложения G = B + C. Положим также, что нам известен тест (последовательность входных наборов) для контроля сумматора по модулю 2^n, и нами выбран один набор теста, который задает B = 15, C = 4, G' = 5 по модулю 2^n. Для построе-

ния теста контроля рассматриваемого объекта система уравнений (рис. 2) должна быть скорректирована, как показано на рис. 3, чтобы обеспечить распространение неисправности к выходам объекта и определение входных переменных.

D = /S*A + S*B;	G'≠ G;
G = B + C;	F' = E * G';
E = D + C;	F ≠ F' ;
F = E * G;	L' = D < G';
L = D < G;	L ≠ L' ;

Рис. 3. Система арифметических уравнений (mod 2^n) с неисправностью

Идея решения подобных уравнений состоит в том, что каждой целочисленной переменной $m <= 2^n$ ставится в соответствие определенный логический вектор размерности n. Значение каждого бита результата арифметической операции вычисляется рекурсивно следующим образом. Для уравнения $w=a+b$ вначале вычисляется значение младшего бита результата $w_0=a_0+b_0$. После вычисления w_0 его результат используется при формировании равенств по модулю 2 для следующих битов. Ниже определен метод вычисления битов высшего порядка из битов более низкого порядка.

Лемма 1. Если x, y, z – решение $f(x,y,z) \equiv 0(\mathrm{mod}\, 2^n)$ и $x_n, y_n, z_n \in \{0,1\}$, то $f(x + x_n 2^n, y + y_n 2^n, z + z_n 2^n) \equiv 0(\mathrm{mod}\, 2^{n+1})$ тогда и только тогда, если $ax_n + by_n + cz_n \equiv f(x,y,z)/2^n (\mathrm{mod}\, 2)$.

Уравнения, соответствующие операции умножения, описываются следующим форматом общего вида: $f(x,y,z) = a*x*y + b*z + c = 0(\mathrm{mod}\, 2^n)$. Для этой формы уравнения мы определяем лемму 2:

Лемма 2. Если x, y, z – решение $f(x,y,z) \equiv 0(\mathrm{mod}\, 2n)$ и $x_n, y_n, z_n \in \{0,1\}$, то $f(x + x_n 2^n, y + y_n 2^n, z + z_n 2^n) \equiv 0(\mathrm{mod}\, 2^{n+1})$ тогда и только тогда, если $axy_n + ax_n y + cz_n \equiv f(x,y,z)/2^n (\mathrm{mod}\, 2)$.

Уравнения, описывающие функцию мультиплексирования данных, описываются следующим форматом: $f(s,x,y) = \bar{s}*x + s*y$. В этом уравнении s имеет размерность 1 бит, а переменные x, y, f – n бит. В первой итерации мы находим решение для s и во второй итерации используем это значение. Так для всех итераций кроме первой. Уравнение мультиплексора является упрощенным случаем, рассмотренным в лемме 1.

Уравнения, соответствующие операторам сравнения данных, описываются следующим общим форматом: $lt_n - (x < y) \equiv 0(\mathrm{mod}\, 2)$, $gt_n - (x > y) \equiv 0(\mathrm{mod}\, 2)$, $eq_n - (x = y) \equiv 0(\mathrm{mod}\, 2)$.

Лемма 3. Если x, y, lt_{n-1} решения $f(x,y,lt_{n-1}) \equiv 0(\mathrm{mod}\, 2)$ и $x_n, y_n, lt_n \in \{0,1\}$, то $f(x + x_n 2^n, y + y_n 2^n, lt_n) \equiv 0(\mathrm{mod}\, 2)$ тогда и только тогда, если $lt_n - ((x_n < y_n) OR (x_n == y_n) lt_{n-1}) \equiv 0(\mathrm{mod}\, 2)$.

Лемма 4. Если x, y, gt_{n-1} решения $f(x, y, gt_{n-1}) \equiv 0(\mathrm{mod}\,2)$ и $x_n, y_n, gt_n \in \{0,1\}$, то $f(x + x_n 2^n, y + y_n 2^n, gt_n) \equiv 0(\mathrm{mod}\,2)$ тогда и только тогда, если gt_n - $((x_n < y_n) OR (x_n == y_n) gt_{n-1}) \equiv 0(\mathrm{mod}\,2)$.

Лемма 5. Если x, y, eq_{n-1} решения $f(x, y, eq_{n-1}) \equiv 0(\mathrm{mod}\,2)$ и $x_n, y_n, eq_n \in \{0,1\}$, то $f(x + x_n 2^n, y + y_n 2^n, eq_n) \equiv 0(\mathrm{mod}\,2)$ тогда и только тогда, если eq_n - $(x_n == y_n) eq_{n-1} \equiv 0(\mathrm{mod}\,2)$.

Отметим, что для рекурсивного вычисления неравенства необходимо учитывать, что неравенство может быть определено только в старшем *i*-м бите (*i* от 0 до *n*-1).

ПОСТРОЕНИЕ ФУНКЦИЙ РАЗРЕШЕНИЯ

Для вычисления определенного бита арифметических операций используются эквивалентные логические уравнения, представляющие функции разрешения уравнений соответствующей системы булевых функций.

Можно показать, что для операции однобитового сложения f = b + c можно получить следующее выражение для функции разрешения $(\bar{b} \vee f)(\bar{c} \vee f)(b \vee c \vee \overline{f})$. В докладе приводятся правила получения КНФ разрешения. В таблице приведены однобитовые функции и соответствующие функции разрешения в виде КНФ.

Функции разрешения

Однобитовые арифметические уравнения	КНФ функций разрешения
$f = b + c$	$(\bar{b} \vee f)(\bar{c} \vee f)(b \vee c \vee \overline{f})$
$f = b * c$	$(b \vee \overline{f})(c \vee \overline{f})(\bar{b} \vee \bar{c} \vee f)$
$f = \bar{s} * a +$	$(s \vee a \vee \overline{f})(a \vee b \vee \overline{f})(\bar{s} \vee b \vee$ $* (s \vee \bar{a} \vee f)(\bar{s} \vee \bar{b} \vee f)(\bar{a} \vee \bar{b}$
$f = b < c$	$(\bar{b} \vee \overline{f})(c \vee \overline{f})(b \vee \bar{c} \vee f)$
$f = 1$	f
$f = 0$	$\overline{f}$
$f \neq f'$	$(f \vee f' \vee \overline{f''})(f \vee \overline{f'} \vee f'') *$ $* (\overline{f} \vee f' \vee f'')(\overline{f} \vee \overline{f'} \vee \overline{f''})$

Чтобы одновременно решить равенства по модулю 2, мы объединяем все КНФ разрешения логических функций вместе, используя логический элемент И. Чтобы найти второй бит решения этого уравнения, мы должны найти рекурсивное уравнение по модулю 2, используя приведенные выше леммы.

ЛИТЕРАТУРА

1. *Jervan, G*. High level and hierarchical test sequence generation / G. Jervan, Z. Peng, O. Goloubeva, M. S. Reorda // Workshop of High-Level Design Validation and Test, 2002. P. 196–174.

2. I*noue, M.* Test synthesis for datapath using datapath-controller functions / M. Inoue, K. Suzuki, H. Okamoto, H. Fujiwara // Proceeding of the 12th Asian Test Symposium (ATS'03). 2003. P. 294–299.

3. *Goloubeva, O.* High-level test generation for hardware testing and software validation / O. Goloubeva, M. Sonza Reorda, M. Violante // Workshop of High-Level Design Validation and Test. 2003. P. 143–148.

4. *Золоторевич, Л. А*. Построение моделей цифровых систем для направленного построения тестов контроля / Л. А. Золоторевич, А. В. Ильинкова // The International Conference Computer-Aided Design of Diskrete Devices (CAD-DD'10) . Minsk. 2010. P. 279–286.