ИНФОРМАТИКА И УПРАВЛЕНИЕ В ТЕХНИЧЕСКИХ И СОЦИАЛЬНЫХ СИСТЕМАХ

УДК 681.004.6

А.Е. Миндров[1], Н.И. Кащеев[2], Н.С. Путихин[2], О.П. Тимофеева[1]

ПОСТРОЕНИЕ ПРОВЕРЯЮЩИХ ТЕСТОВ ДИСКРЕТНЫХ СХЕМ НА ОСНОВЕ НЕПРЕРЫВНЫХ РАСШИРЕНИЙ БУЛЕВЫХ ФУНКЦИЙ

Нижегородский государственный технический университет им. Р.Е. Алексеева[1],
Национальный исследовательский университет Высшая школа экономики, Н. Новгород[2]

Посвящена разработке способа построения тестов цифровых схем с использованием непрерывных моделей дискретных устройств. Представлены алгоритмы, позволяющий решить задачу поиска тестовых наборов с помощью непрерывной оптимизации. Предложенный подход реализован программно в виде среды для поиска тестов цифровых схем. В целях апробации построена система автоматической генерации тестов для константных неисправностей комбинационных схем. Приведены результаты работы программного комплекса для ряда схем набора ISCAS'85, демонстрирующие эффективность используемых алгоритмов и методов.

Ключевые слова: автоматическое построение тестов, моделирование неисправностей, непрерывные модели, цифровые схемы, комбинационные схемы, константные неисправности.

Введение

Трудоемкость выполнения задач построения тестов значительно возрастает при увеличении сложности цифровых схем. Следовательно, требования к эффективности методов и алгоритмов становятся существенно выше. В связи с этим разработке методов и алгоритмов построения тестов дискретных устройств в последнее время уделяется много внимания. Большинство ставших классическими методов построения тестов основаны на использовании модели константной неисправности. В настоящей работе предложен метод построения тестов на основе непрерывного подхода к моделированию работы схемы. В качестве апробации изложенного подхода на базе разработанных программных модулей построена система автоматической генерации тестов для константных неисправностей комбинационных схем. Представлены результаты работы данного программного комплекса на ряде схем набора ISCAS'85.

Непрерывный подход к моделированию дискретных устройств

В большинстве традиционных методов построения тестов используются дискретные модели цифровых устройств. Наиболее распространенными дискретными математическими моделями являются правильные логические сети, таблицы истинности и различные формы аналитических выражений булевых функций, реализуемых устройствами. Как следствие, задачи поиска тестовых последовательностей, основанные на подобных моделях, имеют комбинаторный характер решения. В настоящее время большинство методов представляют собой вариации метода ветвей и границ. Общеизвестна трудоемкость таких методов.

Однако возможно описание комбинационного устройства с помощью функций от непрерывного аргумента. Данная работа описывает метод, при котором булевы функции элементов схемы заменяются их непрерывными аналогами. Подобная идея была впервые пред-

ложена Kano [1]. Настоящая работа развивает этот подход и расширяет область его практического применения.

Итак, изложим основные аспекты непрерывного подхода к моделированию дискретных устройств. Введем следующие обозначения: $x=(x_1,\ldots,x_n)$ – координаты точки в *n*-мерном пространстве, единичный гиперкуб - $T^n=\{x|0\le x_i\le 1, i=1,\ldots,n\}$, $V^n=\{x|x_i\in\{0,1\}, i=1,\ldots,n\}$, элементами множества $V^n\subset T^n$ являются вершины единичного гиперкуба T^n.

Как известно, любая комбинационная схема может быть представлена логической функцией или множеством функций, реализующих логику схемы. При этом каждому первичному выходу схемы будет соответствовать своя логическая функция от *n* переменных $x_1,\ldots,x_n; x_i\in\{0,1\}$.

Пусть $\mu(x)$ – некоторая *n*-местная функция алгебры логики, которую можно определить как отображение вершин единичного *n*-мерного гиперкуба на множество {0, 1} - $\mu(x): V^n\to V^1$.

Определение 1. Непрерывным продолжением функции алгебры логики $\mu(x)$ называется любая функция $f(x)$, отображающая *n*-мерное арифметическое пространство R^n во множество действительных чисел R^1 и совпадающая с функцией $\mu(x)$ на множестве V^n: $f(x): R^n\to R^1, f(x)=\mu(x), \forall x\in V^n$.

Как видим, единственное существенное условие, налагаемое на вид функции $f(x)$, является требование совпадения значений $f(x)$ и $\mu(x)$ в вершинах *n*-мерного гиперкуба.

Рассмотрим вопрос о соотношении булевых функций, образующих базис, с их непрерывными продолжениями. Исходя из приведенного определения, легко видеть справедливость следующего утверждения.

Пусть функция алгебры логики $\mu(x)$ представлена суперпозицией базисных функций. Тогда суперпозиция, полученная заменой базисных функций на их непрерывные продолжения, есть непрерывное продолжение функции $\mu(x)$.

Таким образом, достаточно построить лишь непрерывные продолжения функций алгебры логики, образующих базис. На их основе можно построить непрерывное продолжение любой функции алгебры логики. Возьмем за основу конъюнктивный базис Буля. Определим следующие непрерывные продолжения базисных функций: $\tilde{y}=x_1x_2$ – для конъюнкции, $\tilde{y}=1-x$ – для отрицания. На множестве вершин единичного гиперкуба значения непрерывных продолжений действительно совпадают со значениями соответствующих булевых функций. Основываясь на приведенном утверждении, построим непрерывные продолжения для всех основных функций алгебры логики:

Таблица 1

Непрерывные продолжения булевых функций

Булева функция	Непрерывное продолжение
$Y=\overline{X}$	$\tilde{y}=1-x$
$Y=X_1\wedge X_2$	$\tilde{y}=x_1x_2$
$Y=X_1\vee X_2$	$\tilde{y}=x_1+x_2-x_1x_2$
$Y=X_1\oplus X_2$	$\tilde{y}=x_1+x_2-2x_1x_2$

Построенная система функций имеет следующее свойство: для любых значений аргумента, лежащих внутри единичного гиперкуба, значения функции лежат в интервале (0, 1). Таким образом, значения 0 и 1 достигаются непрерывными продолжениями только на гранях

и в вершинах единичного гиперкуба T^n. Рассмотрим далее правильную логическую сеть, представляющую некоторое комбинационное устройство. В качестве множества возможных неисправностей рассмотрим множество одиночных константных неисправностей.

Определение 2. Непрерывной моделью комбинационного устройства назовем непрерывную сеть $A^н$, получающуюся из правильной логической сети A заменой функций алгебры логики, реализуемых элементами сети A, их непрерывными продолжениями.

Рассмотрим линию l, соединяющую выходной полюс элемента t_1 с входным полюсом элемента t_2. По аналогии с дискретной моделью определим одиночную константную неисправность l/a, $a \in \{0, 1\}$ непрерывной сети $A^н$ как неисправность, приводящую к постоянному присутствию на линии l арифметического значения a.

Пусть построена непрерывная сеть $A^н$ для комбинационного устройства A. Разорвем линию l сети и создадим дополнительный вход ω. Пусть $\tilde{\varphi}^m(\omega, x_1, x_2, ..., x_n)$ есть функция, реализуемая на линии m сети $A^н$.

Определение 3. Функцией чувствительности для сигнала ω на линии m назовем функцию

$$\tilde{W}_l^m(x_1, x_2, ..., x_n) = \tilde{\varphi}^m(1, x_1, x_2, ..., x_n) + \tilde{\varphi}^m(0, x_1, x_2, ..., x_n) - 2\tilde{\varphi}^m(1, x_1, x_2, ..., x_n)\tilde{\varphi}^m(0, x_1, x_2, ..., x_n).$$

Очевидно, что эта функция есть непрерывное продолжение функции исключающее ИЛИ.

Теорема 1

Неисправности $\omega \equiv 1$ и $\omega \equiv 0$ сети $A^н$ проверяюися теми и только теми входными наборами, которые являются решениями уравнений (1) и (2) соответственно:

$$\tilde{W}_l^m(x_1, x_2, ..., x_n)\left(1 - \tilde{\varphi}^l(x_1, x_2, ..., x_n)\right) = 1, x \in V^n; \quad (1)$$

$$\tilde{W}_l^m(x_1, x_2, ..., x_n)\tilde{\varphi}^l(x_1, x_2, ..., x_n) = 1, x \in V^n. \quad (2)$$

Если $\tilde{W}_l^m(x_1, x_2, ..., x_n) = 1$, то значение ω на линии l распространяется до линии m так, что мы наблюдаем ω или $\overline{\omega}$ на линии m. Из теоремы 1 непосредственно следует, что функции в левой части уравнений (1) и (2) являются непрерывными продолжениями различающих функций.

Определение 4. Непрерывными различающими функциями неисправностей $\omega \equiv 1$ и $\omega \equiv 0$ на линии l сети $A^н$ называются функции (3) и (4), соответственно:

$$\varphi_l^1(X) = \tilde{W}_l^m(x_1, x_2, ..., x_n)\left(1 - \tilde{\varphi}^l(x_1, x_2, ..., x_n)\right); \quad (3)$$

$$\varphi_l^0(X) = \tilde{W}_l^m(x_1, x_2, ..., x_n)\tilde{\varphi}^l(x_1, x_2, ..., x_n). \quad (4)$$

Получение непрерывных различающих функций представляет собой значительно более простую задачу, чем получение булевых различающих функций. Действительно, серьезную проблему при получении аналитических выражений булевых функций представляет операция инвертирования нормальных форм в соответствии с правилами Де Моргана. В отличие от этого, инвертирование непрерывных продолжений в соответствии с табл. 1 заключается в простой смене знака у всех слагаемых инвертируемой функции и прибавлении к ней единицы.

Построение проверяющего теста, близкого по длине к минимальному

При нахождении близких к минимальным покрытий таблицы неисправностей P можно применять достаточно простое и эффективное эмпирическое правило: в таблице P выбирается столбец с минимальны числом единиц. Среди строк, содержащих единицу в выбранном столбце, выбирается та, которая содержит максимальное количество единиц. Найденная, та-

ким образом, строка включается в покрытие, а таблица P сокращается путем вычеркивания этой строки и столбцов, которые она покрывает . Далее аналогичная процедура проводится с сокращенной таблицей P и т.д. до тех пор, пока в таблице P не будут вычеркнуты все столбцы.

Алгоритм построения проверяющих тестов, основанный на таком эмпирическом правиле нахождения почти·минимального покрытия таблицы неисправнотсей P, состоит из выполнения следующих пунктов:

1. Для каждой неисправности $s_i \in S$ определяем r_i число проверяющих ее наборов.
2. Полагаем S' подмножество всех обнаружимых неисправностей из S.
3. Во множестве S' выбираем неисправность s_i, проверяемую минимальным числом входных наборов.
4. Находим M_i – множество наборов, проверяющих неисправность s_i.
5. Во множестве M_i выбираем набор v_j, проверяющий максимальное количество неисправностей из множества S', и включаем его в тест.
6. Из множества S' исключаем неисправности, проверяемые набором v_j.
7. Если $S' = \varnothing$, тест построен, в противном случае идем на пункт 3.

Очевидно, этот алгоритм может быть эффективно реализован в том случае, если удастся, не находя таблицы P, решать следующие задачи:

Задача А: для каждой неисправности $s_i \in S$ найти r_i - количество проверяющих ее наборов.

Задача В: построить множества M_i всех наборов, проверяющих неисправность s_i.

Задача С: определить для любого входного набора множество проверяемых ей неисправностей.

Рассмотрим задачу А. Пусть $\varphi_i(x)$ – непрерывная различающая функция неисправности $s_i \in S$. Функция $\varphi_i(x)$ имеет вид:

$$\varphi_i(x) = \sum_{j=1}^{N} b_j x_{j_1} \ldots x_{j_{t_j}} + b_0,$$

где b_j, $j = \overline{0, N}$ целые числа. Перегруппировав в этом выражении слагаемые, легко представить функцию $\varphi_i(x)$ в виде

$$\varphi_i(x) = a_0 + \sum_p \alpha_p x_p + \sum_{p<j} \alpha_{pj} x_p x_j + \sum_{p<j<t} \alpha_{pjt} x_p x_j x_t + \ldots,$$

где $\alpha_0 = \varphi_i(\overset{1}{0}, \ldots, \overset{n}{0})$,

…

$\alpha_0 = \varphi_i(\overset{1}{0}, \ldots \overset{p}{1}, \ldots, \overset{n}{0}) - \alpha_0$,

…

$\alpha_0 = \varphi_i(\overset{1}{0}, \ldots \overset{p}{1}, \ldots, \overset{j}{1} \ldots, \overset{n}{0}) - (\alpha_0 + \alpha_p + \alpha_j)$, и т. д.

Решение задачи А дает следующая теорема.

Теорема 2

$r_i = 2^n \varphi(0.5, \ldots, 0.5)$,

где r_i – количество входных наборов, проверяющих неисправность s_i сети A^H, $\varphi_i(x)$ – непрерывная различающая функция неисправности s_i, n – количество входных полюсов сети A^H.

Доказательство

Определим операцию над двумя векторами $x \in T^n$ и $v_j = (v_{j_1}, \ldots, v_{j_n}) \in V^n$, $j = \overline{1, 2^n}$:

$$(x*v_j)=\prod_{m=1}^{n}\eta(x_m,v_{jm}),\text{ где }\eta(x_m,v_{jm})=\begin{cases}x_m, & \text{если } v_{jm}=1,\\ 1-x_m, & \text{если } v_{jm}=0.\end{cases}$$

Операция $(x*v_j)$ обладает следующим очевидным свойством:

$$(v_i*v_j)=\begin{cases}0, \text{ если } v_i\neq v_j,\\ 1, \text{ если } v_i=v_j.\end{cases} \tag{5}$$

Рассмотрим функцию $\varphi_i^*(x)=\sum_{j=1}^{2^n}\varphi_i(v_j)(x*v_j)$.

Раскрыв в функции $\varphi_i^*(x)$ все скобки и приведя подобные члены, представим $\varphi_i^*(x)$ в виде

$$\varphi_i^*(x)=c_0+\sum_p c_p x_p+\sum_{p<j}c_{pj}x_p x_j+\sum_{p<j<t}c_{pjt}x_p x_j x_t+\ldots,$$

где

$$c_p=\varphi_i^*\left(\overset{1}{0},\ldots,\overset{n}{0}\right)$$

$$c_p=\varphi_i^*\left(\overset{1}{0},\ldots,\overset{p}{1},\ldots,\overset{n}{0}\right)-c_0$$

$$c_{pj}=\varphi_i^*\left(\overset{1}{0},\ldots,\overset{p}{1},\ldots,\overset{j}{1},\ldots,\overset{n}{0}\right)-(c_0+c_p+c_j),\text{ и т.д.}$$

Учитывая свойство (5),

$$\varphi_i^*(\upsilon_m)=\sum_{j=1}^{2^n}\varphi_i(\upsilon_j)(\upsilon_m*\upsilon_j)=\varphi_i(\upsilon_m),m=\overline{1,2^n}.$$

Отсюда следует, что $c_0=d_0,\ldots,c_p=d_p,\ldots,c_{pj}=d_{pj},\ldots$ и т.д.

Тогда в любой точке $x\in T^n$ справедливо $\varphi_i(x)=\varphi_i^*(x)$.

Таким образом,

$$\varphi_i(0.5,\ldots,0.5)=\varphi_i^*(0.5,\ldots,0.5)=\sum_{j=1}^{2^n}\varphi_i(\upsilon_j)\big((0.5,\ldots,0.5)*\upsilon_j\big)=\sum_{j=1}^{2^n}\varphi_i(\upsilon_j)(0.5)^n=2^{-n}\sum_{j=1}^{2^n}\varphi_i(\upsilon_j)=2^{-n}r_i,$$

что и требовалось доказать.

Рассмотрим задачу В.

Из теоремы 2 имеем $0\le\varphi_i(0.5,\ldots,0.5)=2^{-n}r_i$

Случай $\varphi_i(0.5,\ldots,0.5)=0$ соответствует необнаружимой неисправности s_i, тогда $M_i=\varnothing$.

Рассмотрим случай $\varphi_i(0.5,\ldots,0.5)>0$.Как было показано выше, справедливо представление $\varphi_i(x_1,0.5,\ldots,0.5)=f_{i_1}(0.5,\ldots,0.5)x_1+g_{i_1}(0.5,\ldots,0.5).$

Из точки $a_0=(0.5,\ldots,0.5)$ перейдем в точку $a_1=(\delta_1,0.5,\ldots,0.5)$, где

$$\delta_1=\begin{cases}1, \text{ если } f_{i_1}(a_0)=\dfrac{\partial\varphi_i}{\partial x_1(a_0)}>0,\\ 0, \text{ если } f_{i_1}(a_0)=\dfrac{\partial\varphi_i}{\partial x_1(a_0)}<0,\\ 0 \text{ или } 1, \text{ если } f_{i_1}(a_0)=\dfrac{\partial\varphi_i}{\partial x_1(a_0)}=0.\end{cases}$$

Очевидно, $\varphi_i(a_0) \le \varphi_i(a_1)$. Аналогично совершим переход из точки a_1 в точку $a_2 = (\delta_1, \delta_2, ..., 0.5, ..., 0.5)$ и т.д. до тех пор, пока не придем в точку $a_n = \delta^1 \in V^n$. Поскольку $\varphi_i(\delta^1) \ge \varphi_i(a_0) > 0$ и $\varphi_i(x) \in \{0,1\}$ при $x \in V^n$, для значения $\varphi_i(\delta^1)$ остается единственная возможность - $\varphi_i(\delta^1) = 1$.

Таким образом, получен набор δ^1, проверяющий неисправность s_i. Набор δ^1 включаем в множество M_i и рассматриваем далее функцию $\varphi_{i_1}(x) = \varphi_{i_1}(x) - (x * \delta^1)$.

Функция $\varphi_{i_1}(x)$ совпадает с функцией $\varphi_i(x)$ во всех точках множества V^n за исключением точки δ^1. Действительно, $\varphi_{i_1}(\delta^1) = \varphi_i(\delta^1) - (\delta^1 * \delta^1) = 1 - 1 = 0$. С другой стороны, если, $\upsilon_i \ne \delta^1$ то $\varphi_{i_1}(\upsilon_j) = \varphi_i(\upsilon_j) - (\upsilon_j * \delta^1) = \varphi_i(\upsilon_j)$. Способом, описанным ранее, найдем точку δ^2 такую, что $\varphi_i(\delta^2) = \varphi_{i_1}(\delta^2) = 1$. Набор δ^2 включаем в множество M_i и рассматриваем далее функцию $\varphi_{i_2}(x) = \varphi_{i_1}(x) - (x * \delta^2)$. и т.д. до тех пор, пока не будет получен последний набор $\delta^{ri}, r_i \ge 1$, проверяющий неисправность s_i.

Рассмотрим задачу С. Решение ее очевидно. Пусть υ_i - некоторый входной набор, $S' \subset S$ - некоторое подмножество множества S рассматриваемых неисправностей. Набор υ_i проверяет неисправность s_i тогда и только тогда, когда $\upsilon_i(\upsilon_j) = 1$, а количество неисправностей из S', проверяемых набором υ_i, определяется по формуле $R_j = \sum_{s_i \in S'} \varphi_i(\upsilon_j)$.

В качестве примера применения алгоритма рассмотрим построение теста для схемы, изображенной на рис. 1.

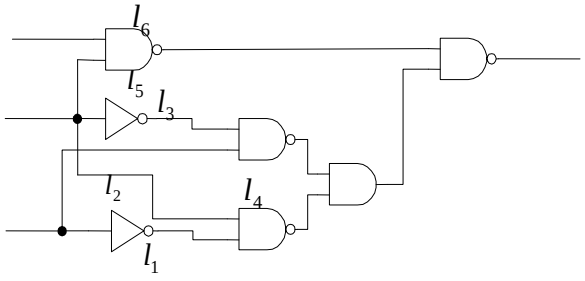

Рис. 1. Комбинационная схема

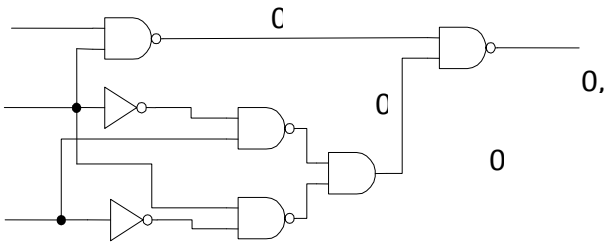

Рис. 2. Вычисление значений на каждой линии схемы

В табл. 2 для каждой рассматриваемой неисправности приведены непрерывные различающие функции $\varphi_i(x)$, а также $r_i = 8\varphi_i(0.5,0.5,0.5)$ - количество наборов, проверяющих каждую из рассматриваемых неисправностей. На первом шаге построения теста $S' = S$. Из табл. 2 видно, что множество S' содержит 8 неисправностей, каждая из которых проверяется единственным набором. Выберем любую из них, например s_1. Для того чтобы найти набор, проверяющий неисправность s_1, рассмотрим функцию $\varphi_1(x_1, x_2, x_3) = x_1 x_2 - x_1 x_2 x_3$.

Производные функции $\varphi_1(x)$ равны $\partial\varphi_1 / \partial x_1 = x_2 - x_2 x_3$, $\partial\varphi_1 / \partial x_2 = x_1 - x_1 x_3$, $\partial\varphi_1 / \partial x_3 = -x_1 x_2$. Поскольку $\partial\varphi_1 / \partial x_1(0.5,0.5,0.5) = 0.25 > 0$, из этой точки $a_0 = (0.5,0.5,0.5)$ переходим в точку $a_1 = (1,0.5,0.5)$; $\partial\varphi_1 / \partial x_2(a_1) = 0.5 > 0$, переходим в точку $a_2 = (1,1,0.5)$; $\partial\varphi_1 / \partial x_3(a_2) = -1 < 0$, переходим в точку $a_3 = (1,1,0)$. Таким образом, найден набор 110, проверяющий неисправность s_i. Поскольку $\varphi_1(1,1,0) = 1$ для $i \in \{1,1,12\}$, удалив неисправности s_1, s_5, s_{12} из множества S', получим $S' = \{s_2, s_3, s_4, s_6, s_7, s_8, s_9, s_{10}, s_{11}\}$. Для дальнейшего рассмотрения из множества S' выберем неисправность s_2.

Аналогично найдем набор 010, проверяющий эту неисправность. После удаления неисправностей, проверяемых набором 010, множество S' примет вид $S' = \{s_3, s_4, s_6, s_8, s_9, s_{10}, s_{11}\}$. Выберем далее неисправность s_9. Ее проверяет набор 111. После сокращения множества S' получаем: $S' = \{s_3, s_4, s_6, s_8, s_{10}\}$. В множестве S' находится единственная неисправность s_{10}, для которой $r_i = 1$. Ее проверяет набор 001. После сокращения множества S' получим $S' = \{s_3, s_6\}$.

Из табл. 2 определяем $r_3 = r_6 = 2$. Рассмотрим любую неисправность из S', например s_3. Поскольку $r_3 = 2$, множеством M_3 будет состоять из двух наборов. Используя функцию $\varphi_3(x) = x_1 - x_1x_2$, найдем первый проверяющий набор 100. Рассматривая функцию $\varphi_{3_1}(x) = \varphi_3(x) - ((x_1, x, x_3) * (1,0,0)) = x_1 - x_1x_2 - x_1(1-x_2)(1-x_3)$ найдем второй набор 101.

Таким образом, получено множество $M_3 = \{100,101\}$. Поскольку оба найденных набора проверяют по две неисправности из S', выбираем любой из них, например 100. После этого шага имеем $S' = \varnothing$. Таким образом, получен тест, состоящий из пяти наборов 110, 010, 111, 000, 100.

Таблица 2

Таблица непрерывных различающих функций

i	s_i	$\varphi_i(x_1, x_2, x_3)$	r_i
1	$l_1 \equiv 0$	x_1x_2-$x_1x_2x_3$	1
2	$l_1 \equiv 1$	x_2-x_1x_2-x_2x_3+$x_1x_2x_3$	1
3	$l_2 \equiv 0$	x_1-x_1x_2	2
4	$l_2 \equiv 1$	1-x_1-x_2+x_1x_2	2
5	$l_3 \equiv 0$	x_1x_2-$x_1x_2x_3$	1
6	$l_3 \equiv 1$	x_1-x_1x_2	2
7	$l_4 \equiv 0$	x_2-x_1x_2-x_2x_3+$x_1x_2x_3$	1
8	$l_4 \equiv 1$	1-x_1-x_2+x_1x_2	2
9	$l_5 \equiv 0$	$x_1x_2x_3$	1
10	$l_5 \equiv 1$	x_3-x_2x_3-x_1x_3+$x_1x_2x_3$	1
11	$l_5 \equiv 0$	$x_1x_2x_3$	1
12	$l_6 \equiv 1$	x_1x_2-$x_1x_2x_3$	1

Континуальный метод построения проверяющих тестов

Рассмотренный ранее алгоритм можно отнести к аналитическим методам построения проверяющих тестов комбинационных устройств и, будучи таковым, он имеют ограничение на применение, присущее всей группе этих методов. Ограничения эти связаны с необходимостью получения функций (в данном случае – непрерывных различающих функций) в аналитическом виде. В рамках традиционных моделей комбинационных устройств этот недостаток аналитических методов непреодолим. Непрерывная же модель, каковой является непрерывная сеть A^H, дает возможность вычислять значение реализуемой ей функции, а, значит, и непрерывных различающих функций, в любой точке $x \in R^n$. Присвоим входным полюсам $x_1x_2x_3$ сети A^H значения 0.5,0.5,0.5. На рис. 2 показана последовательность вычисления значений всех линий схемы. На выходе получаем значение 0,578125.

Основным преимуществом непрерывных методов моделирования схемы является другой принцип работы с моделью схемы по сравнению с традиционными подходами. Схема моделируется как обычная непрерывная функция нескольких переменных. Задача алгоритмов построения тестов на основе непрерывного подхода схемы – найти глобальный максимум целевой функции. Таким образом, открывается возможность использовать алгоритмы глобальной оптимизации применительно к задачам поиска тестов для логических схем.

Построение тестов для цифровых схем будем осуществлять в процессе максимизации специально построенной непрерывной целевой функции, описывающей поведение тестируемой схемы, как в присутствии, так и в отсутствии неисправности. В данной работе рассмотрено применение алгоритма поиска максимума целевой функции на основе метода покоординатного подъема. Будем использовать систему, проверяющую эквивалентность двух схем: исправной и неисправной (рис 3). В общем случае, данная система описывается следующей функцией:

$$G(x) = \bigcup_{k=1}^{s} (F_k(x) \oplus F_k^{'}(x)).$$

Выход *G* принимает значение 1 тогда и только тогда, когда две схемы имеют различные значения выходных линий при подаче определенной комбинации сигналов на входы $x_1,...,x_p$.

Действительно, как можно видеть из рис. 3, значение функции *G* будет равно 1, если выходы исправной и неисправной схем различаются, и нулю – в противном случае.

$$G(X) = 1. \tag{6}$$

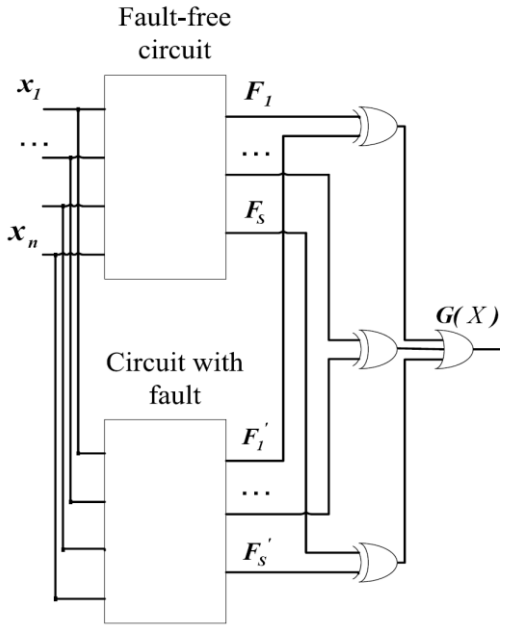

Рис. 3. Система, реализующая целевую функцию

Задачу дискретной оптимизации (6) можно заменить на задачу непрерывной оптимизации (7), используя непрерывную модель дискретного устройства путем замены всех булевых функций на соответсвующие непрерывные продолжения булевых функций.

$$\tilde{G}(X) = 1. \tag{7}$$

Функция $\tilde{G}(X)$ есть непрерывный аналог логической функции $G(X)$.

Итак, для поиска тестов на основе непрерывного подхода необходимо:

- получить непрерывную модель схемы;

- внедрить в схему неисправность и получить непрерывную модель схемы с неисправностью;
- построить непрерывную целевую функцию;
- найти входной вектор, максимизирующий целевую функцию Полученный вектор и есть тест для рассматриваемой неисправности.

Задача поиска максимума целевой функции – это задача глобальной оптимизации, поскольку нас интересует именно глобальный максимум исследуемой функции на всей области определения данной функции. Заметим, что хотя глобальных максимумов может быть и несколько, во всех точках экстремума значение целевой функции будет равно 1, т.е. максимум заранее известен, это упрощает решение задачи. Координаты каждого глобального максимума будут представлять собой один из возможных тестовых наборов.

В качестве первоначального приближения тестового набора выбирается точка $x^1 = (x_1^1, x_2^1, ..., x_n^1) = (0.5, 0.5, ..., 0.5)$ - центр единичного гиперкуба. Далее следуют одна или более итераций, каждая из которых заключается в следующем. Относительно текущей точки вычисляется оценка для частной производной $\partial G(x)/\partial x_i, i = 1,2,...,n$ по каждой координате, значение которой не равно 0 или 1. Оценка определяется как модуль разности между значением целевой функции в текущей точке и значением в точке, имеющей приращение по одной из координат. Выбирается та координата, для которой вычисленная оценка максимальна. Далее, вычисленная разность используется как критерий установления значения для выбранной координаты. Если значение разности неотрицательно, то устанавливается 1. Для отрицательных значений берется 0. Если для вновь полученной точки функция достигает единичного значения, то тестовый набор найден, и работа алгоритма завершается. В противном случае начинается следующая итерация. Количество итераций ограничено, если за заданное количество итераций решение не найдено, то такая несисправность считается необнаружимой. Очевидно, что ограничение накладываемое на количество итераций является компромиссом между быстродействием алгоритма и процентом покрытия несиправностей.

Результаты

Описанные алгоритмы и методы были реализованы в программном комплексе генерации тестовых наборов для константных неисправностей комбинационных схем. Разработанный программный продукт, названный CONTINENT, был апробирован на некоторых комбинационных схемах набора ISCAS'85. В табл. 3 приведены полученные результаты.

Таблица 3

Результаты работы программы генерации тестов

Схема	CONTINENT		ATALANTA [5]		ATPG [4]	
	Покрытие	Время (с)	Покрытие	Время (с)	Покрытие	Время (с)
c432	520 / 524	0.52	520 / 524	1.742	519 / 524	2
c499	750 / 758	0.901	750 / 758	0.11	749 / 758	2
c880	942 / 942	0.63	942 / 942	0.1	922 / 942	7
c1355	1566 / 1574	2.313	1566 / 1574	0.3	1554 / 1574	39
c1908	1870 / 1879	2.323	1870 / 1879	3.194	1852 / 1879	74
c2670	2630 / 2747	18.486	2630 / 2747	68.298	2473 / 2747	3614
c3540	3291 / 3428	14.841	3291 / 3428	1.662	3274 / 3428	1250
c5315	5291 / 5350	10.505	5291 / 5350	1.662	5285 / 5350	320
c6288	7710 / 7744	13.359	7710 / 7744	1.732	7710 / 7744	1100
c7552	7419 / 7550	28.861	7414 / 7550	415.116	7150 / 7550	3600
Всего	31989 / 32496	92.739	31984 / 32496	493.916	31488 / 32496	10008

Выводы

В данной работе предлагается метод построения тестов, основанный на использовании непрерывной модели цифровой схемы. Полученные алгоритмы применены при построении и программой реализации автоматической системы поиска тестовых наборов для константных неисправностей комбинационных схем. Представлены результаты апробации разработанного программного комплекса на некоторых схемах набора ISCAS'85.

Библиографический список

1. **Kano, H.** Test pattern generation for logic networks by real number logic simulation // AUTOTESTCONF'79. 1979. P. 168–178.
2. **Миндров, А.Е.** Использование непрерывной модели схемы для генерации тестов / А.Е. Миндров, Н.И. Кащеев // Simulation and CAD systems. 1989. P. 47–50.
3. **Кащеев, Н.И.** Использование непрерывной оптимизации для генерации тестовых наборов / Н.И. Кащеев, В.В. Белобородов // Системы обработки информации и управления. – Н. Новгород. 2001. Вып. 7. С. 21–25.
4. **Goel, P.** An implicit enumeration algorithm to generate tests for combinational logic circuits // IEEE Trans. on Computers. vol.C-31, P. 215–222, March 1981.
5. H. K. Lee and D. S. Ha. On the Generation of Test Patterns for Combinational Circuits, Technical Report No. 12_93, Department of Electrical Engineering, Virginia Polytechnic Institute and State University, 1993
6. I. Rivin and S.T. Chakradhar. Discrete Test Generation by Continuous Methods // in Proc. 12th IEEE VLSI Test Symposium. P. 100–105, April 1994.

Дата поступления
в редакцию 25.10.2015

A. Mindrov[1], N. Kascheev[2], N. Putikhin[2], O.Timopheeva[1]

TEST PATTERN GENERATION FOR DISCRETE CIRCUITS USING CONTINUOUS EXTENSIONS OF BOOLEAN FUNCTIONS

Nizhny Novgorod state technical university n.a. R.E. Alexeev[1],
National Investigate University Higher School of Economics, Nizhny Novgorod[2]

Purpose: Create a novel algorithm for gate-level test pattern generation for combinational circuits.
Design/methodology/approach: Test generation method for combinational circuits by means of continuous optimization is proposed. The classical ATPG methods target the problem at the logical level and use a discrete approach for simulation of the circuit behavior. This paper presents a new approach for circuit simulation using continuous set of values. In scope of this approach a continuous gate-level model of a circuit is introduced. Boolean functions of the gates are replaced with corresponding continuous analogues. The continuous objective function is constructed and then the problem of test pattern generation is solved by means of maximization of the objective function. Experimental results for ISCAS'85benchmarks are presented to demonstrate the effectiveness of the method proposed.

Key words: ATPG, test generation, combinational circuits, continuous models.