УДК 004.054

Л. В. ДЕРБУНОВИЧ, д-р техн. наук, проф. НТУ "ХПИ",
Д. Г. КАРАМАН, аспирант каф. АУТС НТУ "ХПИ",
Т. Н. ПАЩЕНКО, студентка НТУ "ХПИ"

МЕТОД СИНТЕЗА ДРЕВОВИДНЫХ ЛЕГКО ТЕСТИРУЕМЫХ ЛОГИЧЕСКИХ СХЕМ

У статті запропоновано метод синтезу логічних схем, який дозволяє отримати деревовидну структуру, яка синдромно тестується, для довільної логічної функції з мінімальними апаратними затратами.

The logic schemes synthesis method is presented which makes it possible to obtain syndrome testable tree-like structure for arbitrary logic function with minimal hardware costs.

Увеличение сложности современных цифровых устройств обуславливает совершенствование существующих и разработку новых методов их тестового диагностирования. В рамках этой проблемы особо остро стоит вопрос о методах синтеза легко тестируемых схем, так как из-за специфических условий, выдвигаемых методами тестирования, не каждая произвольная логическая схема (ЛС) пригодна для тестового диагностирования по выбранному методу.

Критический анализ существующих методов синтеза ЛС показывает, что некоторые из них могут быть применены для синтеза тестопригодных ЛС для некоторых методов технической диагностики, в частности синдромного тестирования.

Известно, что древовидные схемы из элементов И, ИЛИ, НЕ, ИЛИ-НЕ, И-НЕ, XOR, не имеющие ветвления внутренних соединений являются синдромно тестируемыми. В такой схеме каждый из путей распространения сигналов от входов схемы к ее выходам представляет собой одномерную сеть (ОС) из логических элементов в базисе, приведенном выше. Такие ОС еще называют каскадом Майтра при условии, что каждый элемент сети является двухвходовым, или каскадом Мадхопадхая, если число входов у каждого из элементов может быть различным (больше двух).

В [1] подробно рассмотрены основные виды ОС и методы их реализации на частных примерах булевых функций. Было предложено расширить базис настраиваемой сети мультиплексором "2→1", что, в соответствии с методикой оценки аппаратных затрат фирмы *Synopsys Inc.* (табл. 1), позволяет более эффективно использовать площадь кристалла.

Однако приведенные методы синтеза ОС носят аналитический характер, не имеют четко сформулированного алгоритма и не пригодны для программной реализации.

В [2] был предложен метод синтеза древовидных КС, основанный на последовательном нахождении множества ОС, дизъюнкция которых является реализацией заданной ЛФ. Для нахождения условий реализуемости произвольной ЛФ одномерной сетью предложено использовать минтермные матрицы (ММ). В работе приведен алгоритм, который можно реализовать программно и применять для произвольных логических функций любого порядка, но использование операции дизъюнкции в качестве крайнего правого элемента дерева осложняет поиск функций, реализуемых в виде ОС. Этот процесс не освещен в описанном алгоритме синтеза и является достаточно трудоемкой задачей.

Таблица 1 – Оценка аппаратных затрат фирмы *Synopsys Inc*.

Инвертор	0,7 в.э.	2 на 1 мультиплексор	1,7 в.э.
2-вх. И-НЕ(ИЛИ-НЕ)	1,0 в.э.	3-вх. И- НЕ(ИЛИ-НЕ)	1,5 в.э.
2-вх. И(ИЛИ)	1,3 в.э.	3-вх. И(ИЛИ)	2,0 в.э.
2-вх. ИСКЛ.-ИЛИ	2,0 в.э.	D-триггер	3,6 в.э.

Цель статьи. Из проведенного анализа следует, что существует необходимость в разработке нового алгоритма синтеза синдромно тестопригодных схем, такого, который бы позволял синтезировать схемы, реализуемые с минимальными аппаратными затратами, и такого, который можно было бы реализовать программно для автоматизации синтеза таких схем.

Синтез древовидных, пригодных для синдромного тестирования схем можно разбить на две основные подзадачи: организация ветвления и синтез одномерных сетей для каждой из ветвей. Основными критериями синтеза являются организация минимального необходимого количества ветвей (в идеальном случае функция должна быть реализована одной ветвью в виде одномерной сети Майтра или Мадхопадхая) и минимального числа элементов в каждой из ветвей.

В качестве алгоритма синтеза ОС целесообразно использовать метод, основанный на использовании минтермных матриц (ММ).

Минтермная матрица ЛФ $f(x_1,x_2,\dots,x_n)$ – прямоугольная матрица размерностью $k\times n$, где k – число минтермов ЛФ, в которой каждый столбец отмечен входными переменными x_i, $i=\overline{1,n}$, а каждая строка соответствует минтермам ЛФ, представленным двоичными векторами.

Пусть в дальнейшем n – число переменных ЛФ, k – число минтермов, q_i – число единиц в столбцах x_i ММ.

Алгоритм синтеза ОС по произвольной логической функции n переменных выглядит следующим образом:

1) составить ММ для заданной ЛФ $f(X')$ (для первой итерации $f(X') = f(X) = f(x_1,x_2,\dots,x_n)$);

2) определить число переменных n, число минтермов k;

3) если число минтермов ЛФ $k < 2^{n-1}$ и в минтермной матрице существует по меньшей мере один столбец x_i, у которого $q_i = k$ или $q_i = 0$, то крайним правым элементом ОС, реализующим заданную ЛФ, является элемент И. Необходимо исключить переменную x_i из дальнейшего рассмотрения и перейти к шагу 1, если n > 1, или перейти к шагу 6 в противном случае;

4) если число минтермов ЛФ $k > 2^{n-1}$ и в минтермной матрице существует по меньшей мере один столбец x_i, у которого $q_i = 2^{n-1}$ или $q_i = k\text{-}2^{n-1}$, то крайним правым элементом ОС, реализующей заданную ЛФ, является элемент ИЛИ. Необходимо исключить переменную x_i из дальнейшего рассмотрения и все те минтермы, на которых $x_i = 1$, если было выполнено условие $q_i = 2^{n-1}$, или все те минтермы, на которых $x_i = 0$, если было выполнено условие $q_i = k\text{-}2^{n-1}$, и перейти к шагу 1, если $k > 1$ или перейти к шагу 6 в противном случае;

5) Если число минтермов ЛФ $k = 2^{n-1}$ и в минтермной матрице найдется по меньшей мере один столбец x_i, вычеркивание которого приводит к минтермной матрице размерности $k\times(n-1)$ с 2^{n-1} различными строками, то крайним правым элементом ОС является элемент XOR. Необходимо исключить переменную x_i из дальнейшего рассмотрения и все те минтермы, на которых $x_i = 1$, затем перейти к шагу 1;

6) если $n = 1$, то на оставшийся вход последнего левого элемента подается x_i , если в ММ $x_i = 1$, или $\overline{x_i}$, если $x_i = 0$; если $k = 1$, то на свободный вход левого элемента подается конъюнкция двух оставшихся переменных x_i и x_j , причем если значение $x_i = 0$ ($x_j = 0$), то эта переменная подается с инверсией $\overline{x_i}$ ($\overline{x_j}$), а если $x_i = 1$ ($x_j = 1$) – без инверсии.

В [2] было предложено использование дизъюнкции в качестве элемента разветвления. Однако такой подход создает определенные сложности по определению подфункций ветвей. Не существует алгоритма разбиения основной функции на подфункции для финальной дизъюнкции с тремя и более входами и, следовательно, приходится выполнять разбиение аналитическим умозрительным путем, что является трудоемкой и машинно-нереализуемой задачей.

В данной статье предложено в качестве элемента-разветвителя использовать мультиплексор "2→1" (рис. 1).

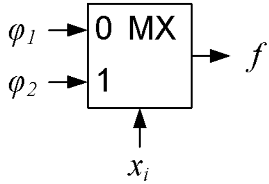

Рис. 1. Мультиплексор "2→1"

Такое решение позволит, во-первых, упростить алгоритмическую реализацию метода синтеза древовидных синдромно тестируемых схем, а во-вторых, при использовании в качестве управляющей мультиплексором одной из входных переменных, позволит сократить количество используемых в подфункциях переменных и тем самым упростить их реализацию:

$$\begin{aligned}
&\varphi_1 = f_1^*(x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_n) = f_1^*(X') \\
&\varphi_2 = f_2^*(x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_n) = f_2^*(X') \\
&f = f_{MX}(\varphi_1,\varphi_2,x_i) = \begin{cases} \varphi_1, & x_i = 0; \\ \varphi_2, & x_i = 1. \end{cases}
\end{aligned} \tag{1}$$

где $X = (x_1,\ldots,x_n)$ – полный набор входных переменных,
$X' = (x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_n)$ – сокращенный набор без x_i.

Если в качестве примера взять функцию:

$$f(x_1,x_2,x_3,x_4,x_5) = (x_1x_3x_5 + x_2) \oplus x_4 + (x_1 \oplus x_3)x_5 + x_2 \tag{2}$$

то её реализация методом, предложенным в [2] (рис. 2) может быть оценена в 9,2 в.э. согласно табл. 1, тогда как при использовании в качестве разветвителя мультиплексора "2→1" (рис. 3) – всего лишь в 7,2 в.э. (4-входовый элемент ИЛИ составляет 2,5 в.э.).

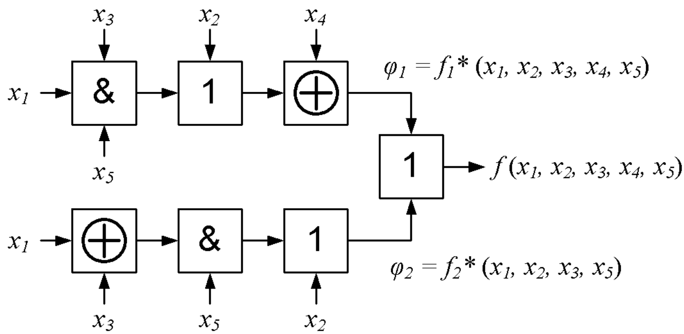

Рис. 2. Синдромно тестопригодная реализация БФ классическим методом

В процессе синтеза схемы были рассмотрены все варианты реализации подфункций φ_1 и φ_2. (табл. 2):

Таблица 2 – Реализуемость подфункций φ_1 и φ_2 в зависимости от выбранной управляющей переменной x_i

	x_1	x_2	x_3	x_4	x_5
φ_1	V	✖	V	V	V
φ_2	✖	V	✖	V	✖

Как видно из таблицы, вероятность успешной одновременной реализации обоих подфункций φ_1 и φ_2 для выбранной управляющей переменной x_i является низкой. Это объясняется большим количеством переменных, от которых зависят подфункции φ_1 и φ_2, и эта зависимость прямо пропорциональна.

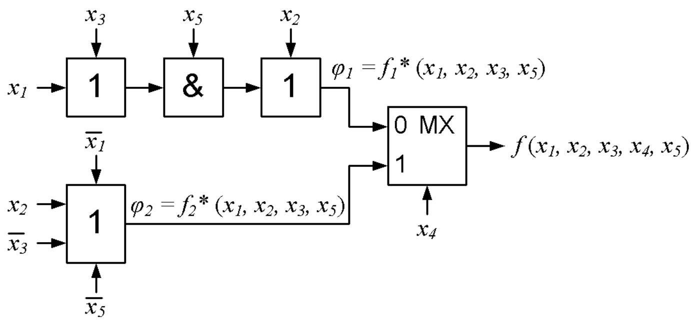

Рис. 3. Синдромно тестопригодная реализация БФ на основе мультиплексора "2→1"

Решением этой проблемы может стать многокаскадное использование мультиплексоров при организации ветвления функции. Алгоритмически это выглядит как рекурсивное применение метода разветвления на основе мультиплексоров к подфункциям φ_1 и φ_2.

Двухкаскадное разбиение, приведенное на рис. 4, может быть описано следующими выражениями:

$$\begin{gathered}
\varphi_{11} = f_1^*(x_1,\dots,x_{i-1},x_{i+1},\dots,x_{j-1},x_{j+1},\dots,x_n) = f_1^*(X'');\\
\varphi_{12} = f_2^*(x_1,\dots,x_{i-1},x_{i+1},\dots,x_{j-1},x_{j+1},\dots,x_n) = f_2^*(X'');\\
\varphi_{21} = f_3^*(x_1,\dots,x_{i-1},x_{i+1},\dots,x_{j-1},x_{j+1},\dots,x_n) = f_3^*(X'');\\
\varphi_{22} = f_4^*(x_1,\dots,x_{i-1},x_{i+1},\dots,x_{j-1},x_{j+1},\dots,x_n) = f_4^*(X'');\\
\varphi_1 = f_{MX}(\varphi_{11},\varphi_{12},x_j)\\
\varphi_2 = f_{MX}(\varphi_{21},\varphi_{22},x_j)\\
f = f_{MX}(\varphi_1,\varphi_2,x_i)
\end{gathered} \tag{2}$$

где $X'' = (x_1,\dots,x_{i-1},x_{i+1},\dots,x_{j-1},x_{j+1},\dots,x_n)$ – сокращенный набор без x_i и x_j.

Функция, предлагаемая ранее в качестве примера, была реализована и с применением двухкаскадного разбиения. Результаты исследования реализуемости подфункций φ_{11}, φ_{12}, φ_{21} и φ_{22} приведены в табл. 3.

Анализ табл. 3 показывает, что вероятность одновременной реализации подфункций φ_{11}, φ_{12}, φ_{21} и φ_{22} для двухкаскадного мультиплексорного ветвления значительно выше, чем для одновременной реализации подфункций φ_1 и φ_2 для однокаскадной версии. Для функций более сложных, чем приведенная в примере, двухкаскадной реализации может оказаться недостаточно. В этом случае алгоритм можно применять рекурсивно для подфункций каждого каскада до получения удовлетворяющих результатов.

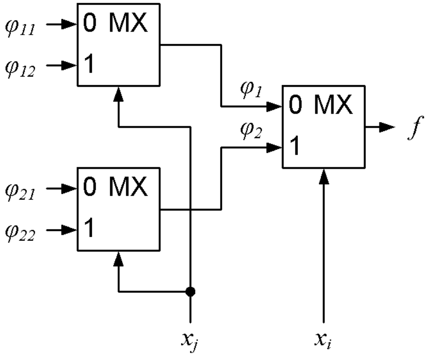

Рис. 4. Двухкаскадное мультиплексорное разбиение функции f

Таблица 3 – Реализуемость подфункций φ_{11}, φ_{12}, φ_{21} и φ_{22} в зависимости от выбранных управляющих переменных x_i и x_j

	x_1				x_2				x_3				x_4				x_5			
	x_2	x_3	x_4	x_5	x_1	x_3	x_4	x_5	x_1	x_2	x_4	x_5	x_1	x_2	x_3	x_5	x_1	x_2	x_3	x_4
φ_{11}	V	V	V	V	V	V	V	V	V	V	V	V	V	V	V	V	V	V	V	V
φ_{12}	V	V	V	V	×	×	V	×	V	V	V	V	V	V	V	V	V	V	V	V
φ_{21}	×	V	V	V	V	V	V	V	V	×	V	V	V	×	V	V	V	×	V	V
φ_{22}	V	V	V	V	V	V	V	V	V	V	V	V	V	V	V	V	V	V	V	V

С учетом всего вышесказанного алгоритм синтеза синдромно тестируемых древовидных схем с организацией многокаскадного ветвления на основе мультиплексоров "2→1" выглядит следующим образом:

1) использовать алгоритм построения ОС к заданной функции $f(X)$, $X = (x_1,\ldots,x_n)$;

2) если функция реализуема, то поставленная цель достигнута – реализация в виде ОС наиболее оптимальна, иначе выбрать одну из переменных x_i в качестве управляющей мультиплексором и разбить ММ функции $f(X)$ относительно выбранной управляющей переменной на ММ двух подфункций φ_1 и φ_2, выполнить реализацию каждой из подфункций в виде ОС;

3) если обе подфункции реализуемы, то поставленная цель достигнута, иначе выбрать в качестве управляющей другую переменную x_i и повторить шаг 2;

4) если ни при одной из переменных $x_1,\ldots,x_n$ подфункции φ_1 и φ_2 одновременно не реализуемы, то в качестве управляющих выбираются 2 управляющие переменные x_i и x_j, а ММ функции $f(X)$ разбивается на 4 ММ подфункций φ_{11}, φ_{12}, φ_{21} и φ_{22}, каждая из которых реализуется в виде одномерной сети;

И т.д.

Последний шаг: если ни при одном наборе из n-1 управляющих переменных x_i, x_j,…,x_k подфункции функции $\Phi = (\varphi_{ij\dots k},\dots)$ одновременно не реализуемы, то реализуется последний каскад управляющих мультиплексоров "2→1", на информационные входы которых подаются фиксированные значения 0 и 1, а на управляющие входы – значения переменных x_n.

Для произвольной функции $f(X)$, $X = (x_1,\dots,x_n)$ возможна n-1 итерация этого алгоритма, при условии, если хотя бы одна из подфункций φ_i, φ_{ij}, $\varphi_{ijk},\dots$ $\varphi_{ij\dots k}$ не реализуется в виде одномерной сети. В таком случае ветвление рассматриваемой функции будет полностью выполнено на мультиплексорах "2→1", что является допустимым, но не оптимальным решением.

Список литературы: **1.** *Шалыто А.А.* Логическое управление. Методы аппаратной и программной реализации. – СПб.: Наука, 2000 г. – 780 с. **2.** *Татаренко Д.А.* Програмовані логічні контролери з вбудованими засобами тестового та функціонального діагностування: автореф. дис. канд. техн. наук: 05.13.05 / НТУ "ХПІ". – Х., 2007. – 20 с. – укр. **3.** *Бережная М.А., Ковзель Н.О., Татаренко Д.А.* Синтез синдромно тестируемых программируемых логических контролеров. Часть 2. Примеры синтеза легко тестируемых схем // Информационно – управляющие системы на железнодорожном транспорте. – 2005. – № 3(53). – с. 49 – 52. **4.** *Уильямс Т.У., Паркер К.П.* Проектирование контролепригодных устройств//ТИИЭР. – 1983. – т.71. – №1. – С.122–139. **5.** *Сапожников В.В., Сапожников В.В.* Методы синтеза надежных автоматов //Л.: Энергия. – 1980. – 96 с.

Поступила в редколлегию 4.08.09