УДК 681.518.5:004.052.32

ПОСТРОЕНИЕ САМОПРОВЕРЯЕМЫХ СТРУКТУР СИСТЕМ ФУНКЦИОНАЛЬНОГО КОНТРОЛЯ НА ОСНОВЕ РАВНОВЕСНОГО КОДА «2 ИЗ 4»

В.В. Сапожников, Вл.В. Сапожников, Д.В. Ефанов

Рассмотрен метод логического дополнения для организации систем функционального контроля на основе равновесного кода «2 из 4» (2/4-кода). Формализованы правила вычисления функций логического дополнения для преобразования любого вектора значений рабочих функций в системе функционального контроля до вектора 2/4-кода. Сформулированы необходимые и достаточные условия, предъявляемые к контролируемому логическому устройству для обеспечения свойства полной самопроверяемости структуры системы функционального контроля.

Ключевые слова: система функционального контроля, логическое дополнение, код «2 из 4», полностью самопроверяемая структура, тестирование.

ВВЕДЕНИЕ

При проектировании надежных систем автоматики и вычислительной техники часто применяются методы функционального, или рабочего, контроля [1—3]. Один из распространенных подходов заключается в применении систем функционального контроля, позволяющих косвенно по результатам вычислений значений реализуемых объектом диагностирования функций определять его техническое состояние. При этом объект диагностирования $F(x)$ снабжается специализированной схемой контроля, позволяющей в процессе функционирования определять его техническое состояние без отключения выходов от управляемых объектов [4].

На рис. 1 изображены структурные схемы систем функционального контроля логических устройств автоматики и вычислительной техники. Первая из них реализована согласно классической структуре с использованием разделимых (m, k)-кодов (m и k — длины информационных и контрольных векторов соответственно) [5—8]. Вектор значений рабочих функций отождествлен с информационным вектором (m, k)-кода. Схема контроля содержит в своей структуре блок контрольной логики $G(x)$, формирующий векторы значений контрольных функций, соответствующие контрольным векторам заранее выбранного (m, k)-кода, а также схему тестера *TSC*. Тестер предназначен для установления факта соответствия информационных и контрольных векторов друг другу. Если соответствие нарушено, то тестер формирует контрольный непарафазный сигнал <00> или <11> [9].

Характеристики системы функционального контроля определяются топологией логического устройства $F(x)$, выбранным на этапе проектирования (m, k)-кодом, а также способом реализации блоков схемы контроля. Для установленного (m, k)-кода система функционального контроля по структурной схеме, изображенной на рис. 1, *а*, может быть построена единственно возможным способом, «жестко» определяющим характеристики обнаруживаемых на выходах блока $F(x)$ ошибок. Этого недостатка лишена система функционального контроля, реализующая принцип логического дополнения функций (рис. 1, *б*) [10—14].

В отличие от классической схемы системы функционального контроля, приведенной на рис. 1, *а*, данная схема позволяет с использованием выбранного кода строить некоторое множество систем функционального контроля, различных по своим характеристикам, в том числе и по обнаружению неисправностей. При этом возможна оптимизация системы по критерию минимума структурной избыточности. Возможность получения различных структур систем функционального контроля по методу логического дополнения реализуется при помощи блока логического дополнения, содержащего каскад сумматоров по модулю два (*XOR*). Каждый сумматор позволяет преобразовать значение разряда функции f_i в требуемое значение конт-

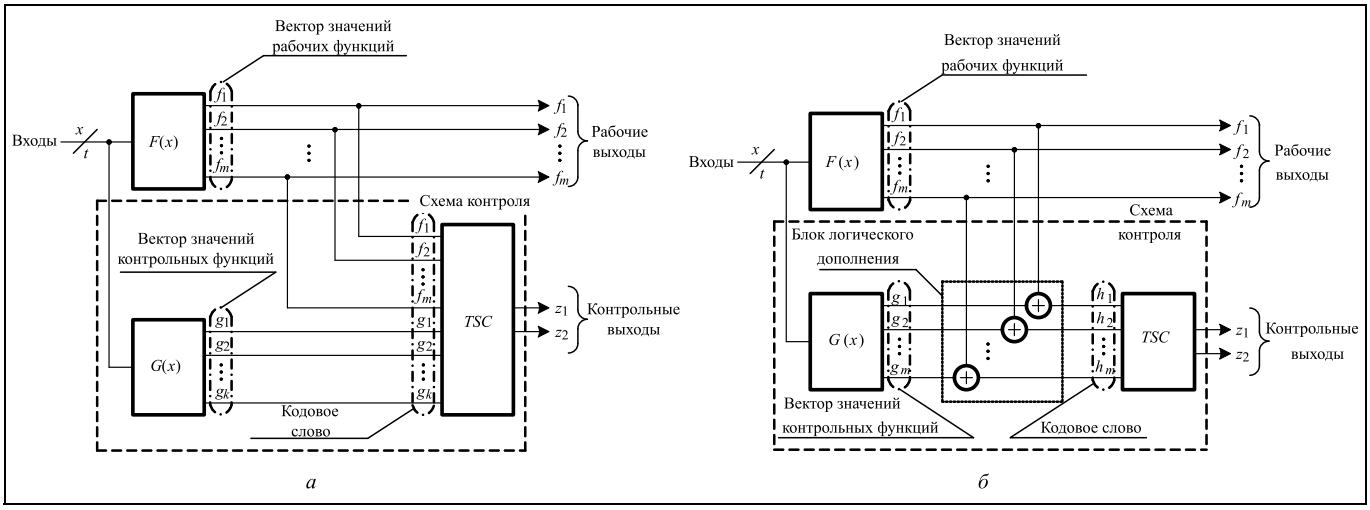

Рис. 1. Структурная схема системы функционального контроля на основе: *а* — разделимого кода; *б* — логического дополнения

рольной функции h_i: $h_i = f_i \oplus g_i$, $i = \overline{1, m}$. Тестер в описываемой системе выполняет ту же роль устройства фиксации неисправностей, что и в системе, приведенной на рис. 1, *а*.

При организации системы функционального контроля важно обеспечить свойство полной самопроверяемости ее структуры. Для решения этой задачи необходимо выполнить два условия. Первое заключается в том, что устройство $F(x)$ должно быть проверяемым, другими словами, любая неисправность из заданного класса (это, как правило, одиночные константные неисправности выходов внутренних логических элементов [4]) обнаруживалась в момент ее первого проявления на выходах устройства. Второе условие связано с обеспечением свойства полной самопроверяемости структуры схемы контроля, что в свою очередь требует, чтобы блок контрольной логики $G(x)$ был проверяемым, как и блок $F(x)$, а тестер *TSC*, как «последний сторож» в системе, — полностью самопроверяемым. Вообще, свойство полной самопроверяемости некоторого устройства подразумевает, что любая неисправность из заданного класса на выходах элементов его внутренней структуры должна проявиться на выходах устройства в виде некоторой защитной комбинации хотя бы на одном входном воздействии. Для тестера *TSC* это требует появления на его входах множества тестовых комбинаций, необходимых для его полной проверки [9].

Для обеспечения свойства полной самопроверяемости структуры, приведенной на рис. 1, *б*, помимо обозначенных выше условий, требуется также обеспечить подачу полного множества тестовых комбинаций для каждого элемента *XOR* в блоке логического дополнения. В общем случае, вне зависимости от привязки к технологии реализации данного элемента, например, при его канонической реализации, это набор комбинаций {00; 01; 10; 11} [15]. При рассмотрении функциональных же особенностей элементов *XOR* комбинация <11> может быть исключена из множества тестовых наборов.

В данной работе предлагается способ организации системы функционального контроля на основе метода логического дополнения с полностью самопроверяемой структурой на основе равновесного 2/4-кода.

1. ОСОБЕННОСТИ ПРИМЕНЕНИЯ РАВНОВЕСНЫХ КОДОВ ПРИ ОРГАНИЗАЦИИ СИСТЕМ ФУНКЦИОНАЛЬНОГО КОНТРОЛЯ

Равновесные коды при организации систем функционального контроля находят широкое применение по двум причинам. Прежде всего, для обеспечения полной самопроверяемости тестеров равновесных кодов с малой длиной кодового слова требуется небольшое число комбинаций. Например, для тестера равновесного 1/4-кода (рис. 2, *а*) множество тестовых комбинаций содержит четыре элемента — {0001; 0010; 0100; 1000} [9]. Аналогичную мощность множества тестовых комбинаций имеют и наиболее простые тестеры 2/4-кодов (рис. 2, *б*) — {0011; 1100; 1001; 0110} [16]. Кроме того, равновесные коды обладают свойством идентификации любых монотонных искажений в кодовых словах. При этом, однако, равновесные коды не обнаруживают некоторую долю симметричных ошибок (ошибок, содержащих группы искажений $\{0 \to 1; 1 \to 0\}$ [17, 18]). Их число определяется выражением

$$N_{r,m} = \sum_{d=2}^{m,(m-1)} C_m^r C_r^{d/2} C_{m-r}^{d/2}, \qquad (1)$$

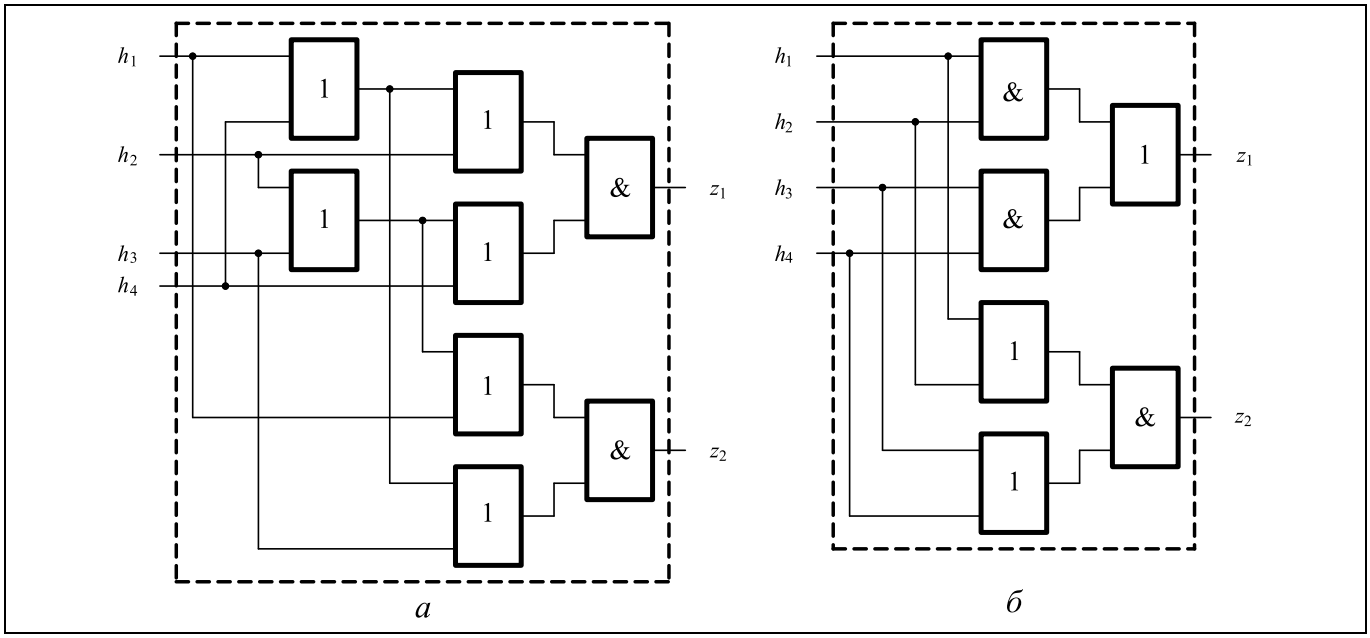

Рис. 2. Схемы тестеров: *а* — 1/4-кода; *б* — 2/4-кода

где r — вес кодовых слов кода «r из m», d — кратность необнаруживаемых ошибок (d — четное).

Сомножитель C_m^r в формуле (1) определяет общее число кодовых слов кода «r из m», сомножитель $C_r^{d/2}$ — число искажаемых единичных разрядов, а сомножитель $C_{m-r}^{d/2}$ — число искажаемых нулевых разрядов в кодовых словах равновесного кода.

Например, применяя формулу (1) для 2/4-кода, получаем

$$N_{2,4} = \sum_{d=2}^{4} C_4^2 C_2^{d/2} C_2^{d/2} = C_4^2 C_2^1 C_2^1 + C_4^2 C_2^2 C_2^2 = 6 \cdot 2 \cdot 2 + 6 \cdot 1 \cdot 1 = 24 + 6 = 30$$

необнаруживаемых ошибок в его кодовых словах (что составляет 12,5 % от общего числа ошибок в кодовых векторах длины $m = 4$). Среди 30-ти необнаруживаемых ошибок у 2/4-кода имеется 24 двукратных и 6 четырехкратных ошибок.

Несмотря на то, что в классе необнаруживаемых ошибок у равновесных кодов присутствует некоторая доля симметричных ошибок, они могут эффективно применяться при синтезе систем функционального контроля со 100 %-м обнаружением любых одиночных неисправностей. Существуют алгоритмы модификации топологии логических устройств автоматики и вычислительной техники в структуры, допускающие возникновение на их выходах только монотонных искажений [19, 20].

В работах [10, 12, 21, 22] описаны особенности применения 1/4-кодов в задачах организации систем функционального контроля. В работе [16] предложен способ организации системы функционального контроля на основе 2/4-кода, обладающий рядом преимуществ перед способом организации системы диагностирования на основе 1/4-кода. Для обеспечения свойства полной самопроверяемости структуры, однако, указанные способы требуют детального анализа формируемых векторов значений рабочих функций блока $F(x)$, а определение значений функций дополнения связано с подбором тестовых комбинаций для тестера *TSC* и элементов *XOR*. В § 2 покажем, что можно формализовать процесс определения значений функций логического дополнения для 2/4-кода и избежать процедуры подбора значений функций дополнения.

2. ФУНКЦИИ ЛОГИЧЕСКОГО ДОПОЛНЕНИЯ

При организации системы функционального контроля необходимо значения разрядов вектора рабочих функций $<f_1\ f_2\ f_3\ f_4>$ преобразовать в значения разрядов векторов 2/4-кода $<h_1\ h_2\ h_3\ h_4>$ (рис. 3). Для преобразования любого вектора $<f_1\ f_2\ f_3\ f_4>$ в вектор $<h_1\ h_2\ h_3\ h_4>$ достаточно использовать только две функции дополнения, например, функции g_3 и g_4. Таким образом, в структуре системы функционального контроля минимизируется сложность блоков контрольной логики и логического дополнения (см. рис. 2). Сами функции логического дополнения должны быть такими, чтобы на входах каждого элемента сложения по модулю два XOR_3 и XOR_4 хотя бы по разу фор-

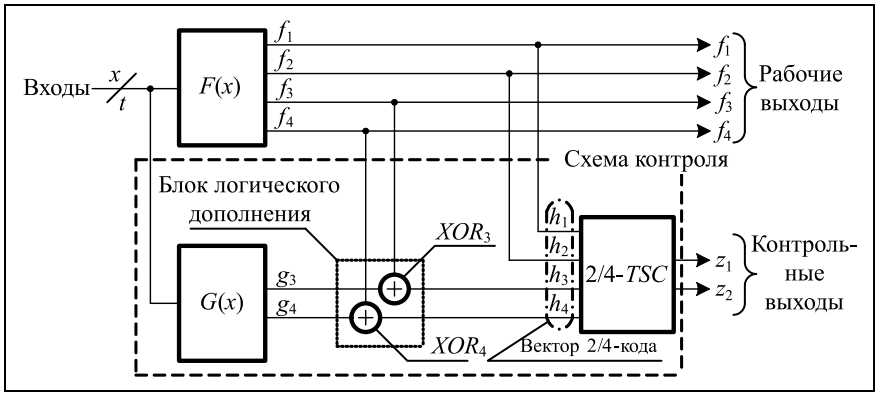

Рис. 3. Структурная схема системы функционального контроля на основе 2/4-кода

мировались все четыре тестовые комбинации, а также хотя бы по разу появлялись все тестовые комбинации 2/4-*TSC* {0011; 1100; 1001; 0110}. Поскольку две рабочие функции f_1 и f_2 не преобразуются в схеме контроля, то их значения соответствуют значениям разрядов h_1 и h_2 вектора 2/4-кода. В табл. 1 представлены все возможные векторы $<f_1\ f_2\ f_3\ f_4>$, функции логического дополнения и векторы $<h_1\ h_2\ h_3\ h_4>$. Заполнены все клетки, значения в которых соответствуют структуре, представленной на рис. 2, и являются однозначными. Часть клеток остаются пустыми.

В строках с номерами 0—3 формируется вектор 2/4-кода 0011, а в строках с номерами 12—15 — вектор 1100. Для обеспечения свойства полной самопроверяемости тестера необходимо на оставшихся восьми наборах $<f_1\ f_2\ f_3\ f_4>$ сформировать комбинации 0110 и 1001. Учитывая тот факт, что в строках с номерами 4—7 разряды $h_1 = 0$ и $h_2 = 1$, целесообразно доопределить значения оставшихся разрядов вектора $<h_1\ h_2\ h_3\ h_4>$ как $h_3 = 1$ и $h_4 = 0$. Аналогично в строках с номерами 8 — 11 доопределяются разряды $h_1 = 1$ и $h_2 = 0$, а оставшиеся разряды вектора $<h_1\ h_2\ h_3\ h_4>$ заполняются значениями $h_3 = 0$ и $h_4 = 1$. В табл. 2 приведена заполненная таблица функций логического дополнения.

Анализируя табл. 2, нетрудно установить, что помимо необходимых тестовых наборов для 2/4-*TSC* для каждого элемента XOR_3 и XOR_4 хотя бы по одному разу формируются все необходимые тестовые комбинации.

Из табл. 2 следует, что функции логического дополнения определяются по формулам:

$$\begin{cases} g_1 = 0; \\ g_2 = 0; \\ g_3 = f_1 f_3 \vee \overline{f}_1 \overline{f}_3 = \overline{f_1 \oplus f_3} = f_1 \sim f_3; \\ g_4 = f_2 f_4 \vee \overline{f}_2 \overline{f}_4 = \overline{f_2 \oplus f_4} = f_2 \sim f_4. \end{cases} \quad (2)$$

Таблица 1

Базовая таблица функций логического дополнения

№	Вектор рабочих функций				Функции логического дополнения				Вектор 2/4-кода			
	f_1	f_2	f_3	f_4	g_1	g_2	g_3	g_4	h_1	h_2	h_3	h_4
0	0	0	0	0	0	0	1	1	0	0	1	1
1	0	0	0	1	0	0	1	0	0	0	1	1
2	0	0	1	0	0	0	0	1	0	0	1	1
3	0	0	1	1	0	0	0	0	0	0	1	1
4	0	1	0	0	0	0			0	1		
5	0	1	0	1	0	0			0	1		
6	0	1	1	0	0	0			0	1		
7	0	1	1	1	0	0			0	1		
8	1	0	0	0	0	0			1	0		
9	1	0	0	1	0	0			1	0		
10	1	0	1	0	0	0			1	0		
11	1	0	1	1	0	0			1	0		
12	1	1	0	0	0	0	0	0	1	1	0	0
13	1	1	0	1	0	0	0	1	1	1	0	0
14	1	1	1	0	0	0	1	0	1	1	0	0
15	1	1	1	1	0	0	1	1	1	1	0	0

Таблица 2

Значения функций логического дополнения

№	Вектор рабочих функций				Функции логического дополнения				Вектор 2/4-кода			
	f_1	f_2	f_3	f_4	g_1	g_2	g_3	g_4	h_1	h_2	h_3	h_4
0	0	0	0	0	0	0	1	1	0	0	1	1
1	0	0	0	1	0	0	1	0	0	0	1	1
2	0	0	1	0	0	0	0	1	0	0	1	1
3	0	0	1	1	0	0	0	0	0	0	1	1
4	0	1	0	0	0	0	1	0	0	1	1	0
5	0	1	0	1	0	0	1	1	0	1	1	0
6	0	1	1	0	0	0	0	0	0	1	1	0
7	0	1	1	1	0	0	0	1	0	1	1	0
8	1	0	0	0	0	0	0	1	1	0	0	1
9	1	0	0	1	0	0	0	0	1	0	0	1
10	1	0	1	0	0	0	1	1	1	0	0	1
11	1	0	1	1	0	0	1	0	1	0	0	1
12	1	1	0	0	0	0	0	0	1	1	0	0
13	1	1	0	1	0	0	0	1	1	1	0	0
14	1	1	1	0	0	0	1	0	1	1	0	0
15	1	1	1	1	0	0	1	1	1	1	0	0

3. УСЛОВИЯ ОБЕСПЕЧЕНИЯ ПОЛНОЙ САМОПРОВЕРЯЕМОСТИ СТРУКТУРЫ

Из табл. 2 следует, что каждый вектор 2/4-кода, необходимый для проверки 2/4-*TSC*, может формироваться на одном из четырех векторов $\langle f_1 f_2 f_3 f_4\rangle$. Например, вектор $\langle h_1\ h_2\ h_3\ h_4\rangle = 0011$ формируется на каждом векторе из множества $A_1 = \{0000; 0001; 0010; 0011\}$. В табл. 3 приведены множества A_i, $i \in \{1, 2, 3, 4\}$, для всех четырех наборов проверяющего теста 2/4-*TSC*. Для каждого вектора $\langle f_1 f_2 f_3 f_4\rangle$ в скобках указан его десятичный эквивалент.

Для проверки элемента XOR_3 необходимо подать на его входы (см. рис. 3) четыре входных набора $\langle f_3 g_3\rangle \in \{00; 01; 10; 11\}$. Из табл. 2 также следует, что формирование каждого такого набора возможно на одном из четырех векторов $\langle f_1\ f_2\ f_3\ f_4\rangle$. Например, набор $\langle f_3\ g_3\rangle = \langle 00\rangle$ образуется на каждом векторе из множества $B_1 = \{1000; 1001; 1100; 1101\}$. В табл. 4 приведены соответствующие множества B_i, $i \in \{1, 2, 3, 4\}$. В табл. 5 представлены множества D_i, $i \in \{1, 2, 3, 4\}$, содержащие векторы $\langle f_1 f_2 f_3 f_4\rangle$, на которых формируются проверяющие наборы $\langle f_4\ g_4\rangle$, необходимые для проверки элемента XOR_4. Отметим, что в табл. 4 и 5 приведены наборы для формирования всех четырех комбинаций, подаваемых для тестирования элементов блока логического дополнения. Столбцы B_4 и D_4 из данных таблиц могли бы быть исключены с учетом привязки только к функциональным особенностям элементов *XOR*. Тем не менее, на конечный результат это не повлияет, так как способ доопределения функций логического дополнения в табл. 2 позволяет сгенерировать все четыре комбинации для каждого элемента *XOR* «автоматически», исходя из необходимости обеспечения полного множества тестовых комбинаций для 2/4-*TSC*.

Анализ таблиц 2—5 позволяет сформулировать два следующих положения.

Теорема. Структура системы функционального контроля на основе 2/4-кода, построенная в соответствии с рис. 4 на элементах любого базиса, является полностью самопроверяемой тогда и только тогда, когда на выходах контролируемого устройства $F(x)$ формируется множество двоичных векторов W такое, что

$$W \cap A_i \neq \varnothing, \quad W \cap B_i \neq \varnothing, \quad W \cap D_i \neq \varnothing,$$
$$i \in \{1, 2, 3, 4\}. \ \blacklozenge$$

Так как минимальное число комбинаций, подача которых на входы 2/4-*TSC* обеспечивает его полную проверку, равно четырем, то справедливо

Утверждение 1. *Минимально возможное число векторов $\langle f_1 f_2 f_3 f_4\rangle$, при которых в структуре, построенной в соответствии с рис. 3, обеспечивается полная проверка 2/4-TSC и элементов XOR, равно четырем.* ♦

Рассмотрим табл. 3. Определим число множеств $W(A)$ с мощностью $\mu = 4$ векторов $\langle f_1 f_2 f_3 f_4\rangle$, формирование которых на выходах блока $F(x)$ обеспе-

Таблица 3

Распределение векторов $\langle f_1\ f_2\ f_3\ f_4\rangle$ на группы тестовых комбинаций, необходимых для полной проверки 2/4-*TSC*

Тестовые комбинации 2/4-*TSC*			
0011	0110	1001	1100
Множества векторов $\langle f_1 f_2 f_3 f_4\rangle$, на которых формируются проверяющие комбинации			
A_1	A_2	A_3	A_4
0000 (0) 0001 (1) 0010 (2) 0011 (3)	0100 (4) 0101 (5) 0110 (6) 0111 (7)	1000 (8) 1001 (9) 1010 (10) 1011 (11)	1100 (12) 1101 (13) 1110 (14) 1111 (15)

Таблица 4

Распределение векторов $\langle f_1\ f_2\ f_3\ f_4\rangle$ на группы тестовых комбинаций, необходимых для полной проверки элемента XOR_3

Тестовые комбинации XOR_3			
00	01	10	11
Множества векторов $\langle f_1 f_2 f_3 f_4\rangle$, на которых формируются проверяющие комбинации			
B_1	B_2	B_3	B_4
1000 (8) 1001 (9) 1100 (12) 1101 (13)	0000 (0) 0001 (1) 0100 (4) 0101 (5)	0010 (2) 0011 (3) 0110 (6) 0111 (7)	1110 (14) 1111 (15) 1010 (10) 1011 (11)

Таблица 5

Распределение векторов $\langle f_1\ f_2\ f_3\ f_4\rangle$ на группы тестовых комбинаций, необходимых для полной проверки элемента XOR_4

Тестовые комбинации XOR_4			
00	01	10	11
Множества векторов $\langle f_1 f_2 f_3 f_4\rangle$, на которых формируются проверяющие комбинации			
D_1	D_2	D_3	D_4
0100 (4) 0110 (6) 1100 (12) 1110 (14)	0000 (0) 0010 (2) 1000 (8) 1010 (10)	0001 (1) 0011 (3) 1001 (9) 1011 (11)	0101 (5) 0111 (7) 1101 (13) 1111 (15)

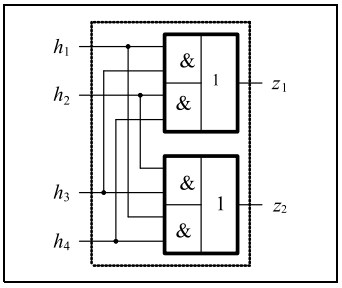

Рис. 4. Структурная схема модуля *TRC*

чивает полную проверку 2/4-*TSC*. В табл. 6 представлены 16 множеств $W(A)$, содержащих векторы, которым соответствуют десятичные эквиваленты 0 и 4. Такое же число множеств $W(A)$ содержат векторы 0 и 5, а также векторы 0 и 6 и векторы 0 и 7. Общее число множеств $W(A)$, содержащих вектор <0000>, равно 64. Очевидно, что каждый из векторов 0001, 0010 и 0011 также содержится в 64 множествах $W(A)$. Таким образом, общее число множеств $W(A) = 256$.

Так как структуры табл. 4 и 5 совпадают со структурой табл. 3, то число множеств $W(B)$ и $W(D)$ векторов $<f_1 f_2 f_3 f_4>$, формирование которых на выходах блока $F(x)$ обеспечивает полную проверку соответственно элементов XOR_3 и XOR_4, также равняется 256. Сравнение множеств $W(A)$, $W(B)$ и $W(D)$ между собой позволило определить, что существует 16 множеств W^4, содержащих четыре вектора $<f_1 f_2 f_3 f_4>$, формирование которых на выходе контролируемого блока обеспечивает полную проверку как 2/4-*TSC*, так и обоих элементов *XOR*. Эти множества приведены в табл. 7.

Утверждение 2. *Структура системы функционального контроля на основе 2/4-кода, построенная в соответствии с рис. 4, является полностью самопроверяемой в том случае, если на выходах контролируемого устройства $F(x)$ формируется множество двоичных векторов W такое, что существует хотя бы одно множество $W^4 \in W$.* ♦

Таблица 6

Множества $W(A)$

{0;4;8;12}	{0;4;9;12}	{0;4;10;12}	{0;4;11;12}
{0;4;8;13}	{0;4;9;13}	{0;4;10;13}	{0;4;11;13}
{0;4;8;14}	{0;4;9;14}	{0;4;10;14}	{0;4;11;14}
{0;4;8;15}	{0;4;9;15}	{0;4;10;15}	{0;4;11;15}

Таблица 7

Множества W^4

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

4. АЛГОРИТМ ПОСТРОЕНИЯ СИСТЕМЫ ФУНКЦИОНАЛЬНОГО КОНТРОЛЯ ДЛЯ МНОГОВЫХОДНЫХ СХЕМ

Прежде всего отметим, что для получения полностью самопроверяемой структуры комбинационного устройства необходимо с помощью представленных в работах [19, 20] методов преобразовать схему блока $F(x)$ в схему, в которой возможно возникновение на выходах только монотонных искажений.

Если у комбинационного логического устройства большое число выходов, то применение классической схемы функционального контроля (см. рис. 1) связано с решением сложной задачи обеспечения полной проверки самопроверяемого тестера. Для этого строится система с раздельным контролем по группам выходов [23]. В этом случае самопроверяемые тестеры имеют небольшое число входов и, следовательно, не требуют значительного числа входных тестовых воздействий.

Данный подход эффективно реализуется с помощью рассмотренной схемы контроля на основе 2/4-кода. Рассмотрим метод построения системы функционального контроля для устройства, реализующего функции из множества $M = \{f_1, f_2, ..., f_m\}$.

Алгоритм. Правила построения полностью самопроверяемой структуры системы функционального контроля на основе 2/4-кода.

Шаг 1. На основе случайного или направленного перебора определяются подмножества выходов $\{f_{i_1}, f_{i_2}, f_{i_3}, f_{i_4}\}$, $i_1, i_2, i_3, i_4 \in \{1, 2, ..., m\}$, отвечающие условию утверждения 2.

Шаг 2. Определяется подмножество выходов M^1 путем исключения из множества M всех выходов, которые вошли в подмножества выходов, полученных на шаге 1.

Шаг 3. На основе случайного или направленного перебора определяются подмножества выходов $\{f_{i_1}, f_{i_2}, f_{i_3}, f_{i_4}\}$, $i_1, i_2, i_3, i_4 \in M^1$, отвечающие условию теоремы.

Шаг 4. Определяется подмножество выходов M^2 путем исключения из множества M^1 всех выходов, которые вошли в подмножества выходов, полученных на шаге 3.

Шаг 5. Далее процесс получения подмножеств $\{f_{i_1}, f_{i_2}, f_{i_3}, f_{i_4}\}$ продолжается для выходов, входящих в множество M^2, с учетом возможности многократного использования любых выходов. Если в результате формируется некоторое подмножество выходов, которые не могут быть включены в под-

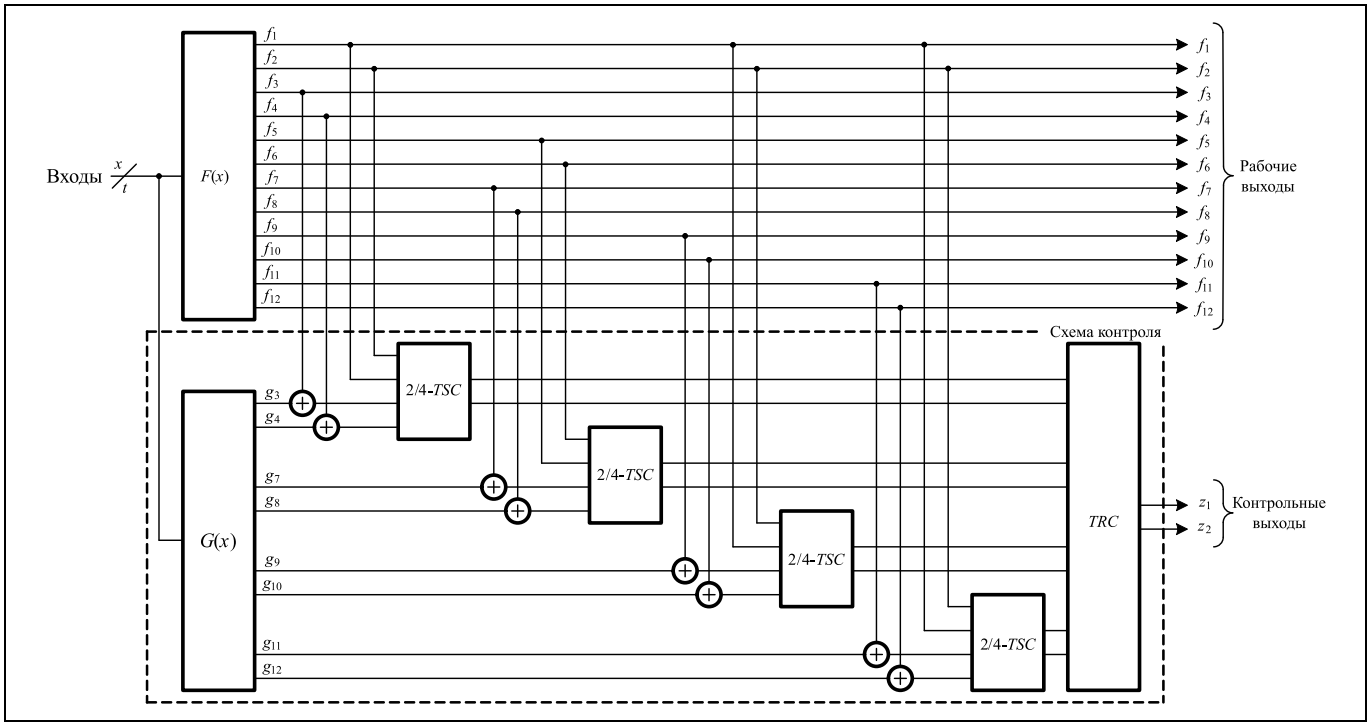

Рис. 5. Полностью самопроверяемая структура для рассматриваемого примера

множества $\{f_{i_1}, f_{i_2}, f_{i_3}, f_{i_4}\}$, отвечающие условиям теоремы, контроль этих выходов осуществляется методами, описанными в работе [16].

Шаг 6. Для контроля технического состояния каждого полученного подмножества «четверок» выходов используются 2/4-*TSC*, парафазные выходы которых объединяются на входе компаратора *TRC*, построенного на основе каскадного соединения модулей сжатия парафазных сигналов (рис. 4) [9].

Рассмотрим, например, комбинационное устройство, реализующее 12 функций от трех переменных, представленных в табл. 8.

Анализ позволяет выявить подмножество выходов $<f_1 f_2 f_3 f_4>$, которое отвечает условию утверждения 2. В самом деле, множество $W\{f_1; f_2; f_3; f_4\} = \{0; 5; 6; 9; 15\}$ полностью включает в себя подмножество $W^4 = \{0; 6; 9; 15\}$, представленное в табл. 7.

Исключение из полного множества выходов устройства выходов f_1, f_2, f_3, f_4 образует сокращенное множество выходов $\{f_5, f_6, ..., f_{12}\}$. Анализ позволяет выявить подмножество выходов $\{f_5, f_6, f_7, f_8\}$, которое не отвечает условию утверждения 2, но отвечает условию теоремы.

Подмножество выходов $W\{f_5; f_6; f_7; f_8\} = \{0; 5; 7; 9; 12; 15\}$ не включает в себя полностью ни одно из множеств W^4 из табл. 7, но содержит хотя бы по одному двоичному вектору из множеств A_i, B_i и D_i, $i \in \{1, 2, 3, 4\}$, а именно $0(A_1)$, $7(A_2)$, $9(A_3)$, $15(A_4)$, $9(B_1)$, $0(B_2)$, $7(B_3)$, $15(B_4)$, $12(D_1)$, $0(D_2)$, $9(D_3)$, $15(D_4)$.

Выходы $f_9, f_{10}, f_{11}, f_{12}$, которые не вошли в указанные подмножества, образуют подмножество, для которого множество $W\{f_9; f_{10}; f_{11}; f_{12}\} = \{1; 2; 9; 13; 15\}$ не отвечает условию теоремы.

Анализ возможности повторного использования выходов $f_1 \div f_8$ позволяет выявить подмножества $\{f_1, f_2, f_9, f_{10}\}$ ($W\{f_1; f_2; f_9; f_{10}\} = \{3; 4; 8; 12; 14; 15\}$) и $\{f_1, f_2, f_{11}, f_{12}\}$ ($W\{f_1; f_2; f_{11}; f_{12}\} = \{1; 6; 9; 10; 13; 14; 15\}$), которые отвечают условию утверждения 2.

На основании полученных подмножеств $\{f_{i_1}, f_{i_2}, f_{i_3}, f_{i_4}\}$ строится полностью самопроверяемая схема, представленная на рис. 5.

Таблица 8

Таблица истинности устройства *F(x)*

$x_1x_2x_3$	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}
000	0	0	0	0	1	1	1	1	1	1	0	1
001	0	1	0	1	1	0	0	1	0	0	1	0
010	1	0	0	1	0	1	0	1	0	0	1	0
011	0	1	1	0	0	1	1	1	0	0	1	0
100	1	1	1	1	1	1	1	1	0	0	1	0
101	1	0	0	1	0	0	0	0	0	0	0	1
110	1	1	1	1	1	0	0	1	1	1	1	1
111	1	1	1	1	1	1	0	0	1	0	0	1

ЗАКЛЮЧЕНИЕ

Предложен способ вычисления логических функций дополнения в системе функционального контроля, позволяющий организовать полностью самопроверяемую систему функционального контроля логического дополнения на основе 2/4-кода. Его преимущество, например, перед методом, описанным в работе [16], заключается в отсутствии необходимости подбора значений функций логического дополнения на каждом векторе значений рабочих функций $<f_1 f_2 f_3 f_4>$. Единственное требование, предъявляемое к контролируемому объекту, состоит в необходимости формирования на его выходах минимального множества векторов $<f_1 f_2 f_3 f_4>$, необходимых для обеспечения свойства полной самопроверяемости тестера и элементов сложения по модулю два, входящих в структуру системы функционального контроля. Однако это требование не является жестким.

Отметим, что структура системы функционального контроля, представленная на рис. 2, может быть реализована путем дополнения любых двух функций f_i, f_j, $i, j \in \{1, 2, 3, 4\}$. Кроме того, возможно построение структур (как это отмечено в работе [16]), в которых устанавливаются дополнительные инверторы на некоторых входах $h_1 \div h_4$ тестера 2/4-*TSC*.

ЛИТЕРАТУРА

1. *Согомонян Е.С., Слабаков Е.В.* Самопроверяемые устройства и отказоустойчивые системы. — М.: Радио и связь, 1989, 208 с.
2. *Goessel M., Graf S.* Error Detection Circuits. — London: McGraw-Hill, 1994, 261 p.
3. *Fujiwara E.* Code Design for Dependable Systems: Theory and Practical Applications. — New Jersey: John Wiley & Sons, 2006, 720 p.
4. *Пархоменко П.П., Согомонян Е.С.* Основы технической диагностики (оптимизация алгоритмов диагностирования, аппаратурные средства). — М.: Энергоатомиздат, 1981, 320 с.
5. *Smith J.E., Dussault J.* Fault Secure Multiple-Valued Logic Networks // Proceeding of the 8th International Symposium on Multiple-Valued Logic (MVL '78), Los Alamitos, CA, USA, 1978, pp. 287—297.
6. *McCluskey E.J.* Logic Design Principles: With Emphasis on Testable Semicustom Circuits. — New Jersey: Prentice Hall PTR, 1986. — 549 p.
7. *Nicolaidis M., Zorian Y.* On-Line Testing for VLSI — A Compendium of Approaches // Journal of Electronic Testing: Theory and Applications. — 1998. — Issue 12. — P. 7—20.
8. *Ubar R., Raik J., Vierhaus H.-T.* Design and Test Technology for Dependable Systems-on-Chip (Premier Reference Source). — Information Science Reference, Hershey — New York, IGI Global, 2011. — 578 p.
9. *Сапожников В.В., Сапожников Вл.В.* Самопроверяемые дискретные устройства. — СПб: Энергоатомиздат, 1992. — 224 с.
10. *Сапожников В.В., Сапожников Вл.В., Дмитриев А.В.* и др. (Всех 5-х авторов см. в оригинале ст.) Организация функционального контроля комбинационных схем методом логического дополнения // Электронное моделирование. — 2002. — Т. 24. — № 6. — С. 51—66.
11. *Гессель М., Морозов А.В., Сапожников В.В., Сапожников Вл.В.* Логическое дополнение — новый метод контроля комбинационных схем // Автоматика и телемеханика. — 2003. — № 1. — С. 167—176.
12. *Гессель М., Морозов А.В., Сапожников В.В., Сапожников Вл.В.* Контроль комбинационных схем методом логического дополнения // Автоматика и телемеханика. — 2005. — № 8. — С. 161—172.
13. *Göessel M., Ocheretny V., Sogomonyan E., Marienfeld D.* New Methods of Concurrent Checking: Edition 1. — Dordrecht: Springer Science + Business Media B.V., 2008. — 184 p.
14. *Das D.K., Roy S.S., Dmitriev A., Morozov A., Gössel M.* Constraint Don't Cares for Optimizing Designs for Concurrent Checking by 1-out-of-3 Codes // Proceedings of the 10th International Workshops on Boolean Problems, Freiberg, Germany, September, 2012. — P. 33—40.
15. *Аксенова Г.П.* Необходимые и достаточные условия построения полностью проверяемых схем свертки по модулю два // Автоматика и телемеханика. — 1979. — № 9. — С. 126—135.
16. *Сапожников В.В., Сапожников Вл.В., Ефанов Д.В.* Метод функционального контроля комбинационных логических устройств на основе кода «2 из 4» // Известия вузов. Приборостроение. — 2016. — Т. 59, № 7. — С. 524—533.
17. *Сапожников В.В., Сапожников Вл.В., Ефанов Д.В.* Классификация ошибок в информационных векторах систематических кодов // Известия вузов. Приборостроение. — 2015. — Т. 58. — № 5. — С. 333—343.
18. *Jha N.K., Vora M.B.* A t-unidirectional Error-Detecting Systematic Code // Computers & Mathematics with Application. — 1988. — Vol. 16, iss. 9. — P. 705—714.
19. *Busaba F.Y., Lala P.K.* Self-Checking Combinational Circuit Design for Single and Unidirectional Multibit Errors // Journal of Electronic Testing: Theory and Applications. — 1994. — Vol. 5, iss. 5. — P. 19—28.
20. *Saposhnikov V.V., Morosov A., Saposhnikov Vl.V., Göessel M.* A New Design Method for Self-Checking Unidirectional Combinational Circuits // Journal of Electronic Testing: Theory and Applications. — 1998. — Vol. 12. — Issue 1—2. — P. 41—53.
21. *Goessel M., Saposhnikov Vl., Saposhnikov V., Dmitriev A.* A New Method for Concurrent Checking by Use of a 1-out-of-4 Code // Proceedings of the 6th IEEE International On-line Testing Workshop, 3—5 July 2000, Palma de Mallorca, Spain, P. 147—152.
22. *Saposhnikov V.V., Saposhnikov Vl.V., Morozov A., Osadtchi G., Gossel M.* Design of Totally Self-Checking Combinational Circuits by Use of Complementary Circuits // Proceedings of East-West Design & Test Workshop, Yalta, Ukraine, 2004. — P. 83—87.
23. *Гессель М., Морозов А.А., Сапожников В.В., Сапожников Вл.В.* Исследование комбинационных самопроверяемых устройств с независимыми и монотонно независимыми выходами // Автоматика и телемеханика. — 1997. — № 2. — С. 180—193.

Статья представлена к публикации членом редсовета чл.-корр. РАН П.П. Пархоменко.

Сапожников Валерий Владимирович — д-р техн. наук, проф.,
✉ port.at.pgups1@gmail.com,

Сапожников Владимир Владимирович — д-р техн. наук, проф.,
✉ at.pgups@gmail.com,

Ефанов Дмитрий Викторович — канд. техн. наук,
✉ TrES-4b@yandex.ru,

Федеральное государственное бюджетное образовательное учреждение высшего образования
«Петербургский государственный университет путей сообщения Императора Александра I».