УДК 004.052.32+681.518.5 DOI: http://doi.org/10.25728/pu.2024.4.3

СИНТЕЗ САМОПРОВЕРЯЕМЫХ ЦИФРОВЫХ УСТРОЙСТВ НА ОСНОВЕ ЛОГИЧЕСКОЙ КОРРЕКЦИИ СИГНАЛОВ С ПРИМЕНЕНИЕМ ВЗВЕШЕННЫХ КОДОВ БОУЗА – ЛИНА

Д. В. Ефанов*,**, Е. И. Елина*

*Санкт-Петербургский политехнический университет Петра Великого, г. Санкт-Петербург
**Российский университет транспорта (МИИТ), г. Москва

*✉ TrES-4b@yandex.ru, **✉ eseniya-elina@mail.ru

Аннотация. Предложен метод синтеза самопроверяемых цифровых устройств, основанный на использовании логической коррекции сигналов и взвешенных кодов Боуза – Лина. В отличие от предыдущих исследований, разработанный метод подразумевает логическую коррекцию сигналов в схеме встроенного контроля для тех функций, описывающих выходы исходных устройств, которые участвуют в формировании информационных символов взвешенных кодов Боуза – Лина. Так как одному и тому же контрольному вектору у таких кодов, как и у абсолютного большинства равномерных разделимых кодов, соответствует большое количество информационных векторов, это дает возможность выбора способа доопределения функций логической коррекции сигналов. Описан один из алгоритмов, позволяющий доопределить значения этих функций на каждом входном наборе с учетом обеспечения полной проверки тестера и элементов преобразования в схеме встроенного контроля. Предложенный метод основан на использовании так называемой «базовой» структуры для контроля многовыходных устройств по группам выходов. Он позволяет проектировщику самопроверяемого устройства иметь большую вариативность в выборе способа его построения, а значит, и влиять на такие важные показатели, как структурная избыточность, контролепригодность, энергопотребление и др. Эксперимент с тестовыми комбинационными схемами из набора MCNC Benchmarks показал высокую эффективность метода по показателям структурной избыточности в сравнении с широко применяемым на практике методом дублирования. Предложенный метод синтеза самопроверяемых устройств может оказаться эффективным при решении задач синтеза реальных устройств с обнаружением неисправностей, используемых во всех областях техники, в том числе в системах критического применения в промышленности и на транспорте.

Ключевые слова: самопроверяемое устройство, схема встроенного контроля, логическая коррекция сигналов, взвешенный код с суммированием, взвешенный код Боуза – Лина.

ВВЕДЕНИЕ

При решении задачи синтеза высоконадежных и безопасных дискретных блоков и узлов систем критического применения важно обеспечивать своевременное обнаружение возникающих в ходе их эксплуатации неисправностей и ошибок в вычислениях [1–4]. Данное свойство достигается путем использования средств тестового и рабочего (функционального) диагностирования [5–7].

Одним из подходов к построению дискретных систем с обнаружением неисправностей является разработка устройств с контролепригодными и самопроверяемыми структурами [8–12]. Это требует внесения избыточности по определенным принципам в само исходное устройство (назовем его объектом диагностирования) либо использования внешних средств технического диагностирования, в том числе снабжения объектов диагностирования самопроверяемыми схемами встроенного контроля (СВК) [1, 13].

Схемы встроенного контроля синтезируются таким образом, чтобы можно было по значениям вычисляемых объектом диагностирования функций или функций, формируемых в структуре объекта диагностирования в специальных контрольных точках, судить о корректности работы объекта диагностирования. Физические дефекты приводят к возникновению неисправностей на выходах элементов внутренней структуры объекта диагностирования и, как следствие, к появлению искажений в значениях вычисляемых функций, что и фиксируется СВК. В качестве диагностических признаков могут использоваться, например, принадлежность формируемых в СВК двоичных векторов множеству кодовых слов каких-либо заранее установленных двоичных избыточных кодов [1] либо принадлежность формируемых в СВК функций заранее оговоренному особому классу булевых функций (например, линейных, монотонных или самодвойственных) [14, 15]. Могут использоваться совместно и сразу же несколько диагностических признаков [16, 17].

Существуют два основных подхода к организации СВК для дискретных устройств. Первый (классический) состоит в том, что объект диагностирования снабжается СВК, в которой происходит дополнение информационного вектора, формируемого на выходах объекта диагностирования, контрольным вектором, вычисляемым с помощью блока контрольной логики [1]. Для проверки соответствия информационного и контрольного векторов друг другу устанавливается тестер. Второй подход подразумевает коррекцию в СВК функций, вычисляемых объектом диагностирования, таким образом, чтобы формировалось кодовое слово заранее выбранного избыточного кода либо чтобы функции оказывались принадлежащими особым классам булевых функций [11]. Этот подход основан на использовании логической коррекции сигналов (ЛКС) в СВК. Он менее исследован, но позволяет синтезировать гораздо большее количество вариантов СВК для одного и того же объекта диагностирования, чем при следовании первому подходу [18]. Это упрощает задачу построения самопроверяемого устройства, а также позволяет влиять на показатели структурной избыточности, контролепригодности, энергопотребления и др. Использование ЛКС предоставляет возможность синтеза самопроверяемых устройств, даже в тех случаях, когда это становится невозможно осуществить другими методами, например, широко применяемым методом дублирования [19].

Настоящая статья посвящена изложению новых результатов в области исследований особенностей применения ЛКС совместно с двоичными избыточными кодами. Предложен метод синтеза самопроверяемых устройств комбинационного типа, основанный на ЛКС и применении взвешенных кодов с суммированием в кольце вычетов по заранее установленному модулю (взвешенных кодов Боуза – Лина).

1. СТРУКТУРА СХЕМЫ ВСТРОЕННОГО КОНТРОЛЯ НА ОСНОВЕ ЛОГИЧЕСКОЙ КОРРЕКЦИИ СИГНАЛОВ

Структура организации СВК на основе ЛКС изображена на рис. 1. В ней объектом диагностирования является блок $F(X)$, снабженный t входами и n выходами. На входы данного блока в процессе его эксплуатации поступают булевы векторы $\langle X\rangle = \langle x_t\ x_{t-1}\ \dots\ x_2\ x_1\rangle$, на которых рассчитываются значения реализуемых блоком $F(X)$ функций и формируется булев вектор $\langle f_n(X)\ f_{n-1}(X)\ \dots\ f_2(X)\ f_1(X)\rangle$. Неисправности, возникающие в процессе эксплуатации блока $F(X)$, приводят к искажению значений сформированного на его выходах вектора $\langle f_n(X)\ f_{n-1}(X)\ \dots\ f_2(X)\ f_1(X)\rangle$. Контролируя возникновение данных искажений, можно косвенно определять наличие неисправностей в объекте диагностирования [1].

Для контроля вычислений на выходах объекта диагностирования устанавливается СВК. В отличие от классической структуры, приведенной, например, в работе [1], в СВК представленной на рис. 1 структуры осуществляется не дополнение вектора $\langle f_n(X)\ f_{n-1}(X)\ \dots\ f_2(X)\ f_1(X)\rangle$, а логическая коррекция сигналов с помощью элементов сложения по модулю $M = 2$ (элементов XOR). Вектор $\langle f_n(X)\ f_{n-1}(X)\ \dots\ f_2(X)\ f_1(X)\rangle$ преобразуется в вектор $\langle h_n(X)\ h_{n-1}(X)\ \dots\ h_2(X)\ h_1(X)\rangle$, который может иметь особые диагностические свойства – например, он может принадлежать множеству кодовых слов заранее выбранного двоичного равномерного кода.

Для коррекции каждого значения $f_i(X), i=\overline{1,n},$ используются двухвходовые элементы XOR, на первые входы которых подаются сами значения $f_i(X)$, а на вторые – значения одноименных функций коррекции $g_i(X)$. Они вычисляются на тех же наборах входных переменных, что и значения $f_i(X)$, блоком вычисления функций коррекции $G(X)$. Таким образом, преобразования в СВК осуществляются по правилу

$$h_i(X) = f_i(X) \oplus g_i(X), i=\overline{1,n}.$$

Каскад элементов XOR образует блок логической коррекции сигналов (БКС). Для контроля

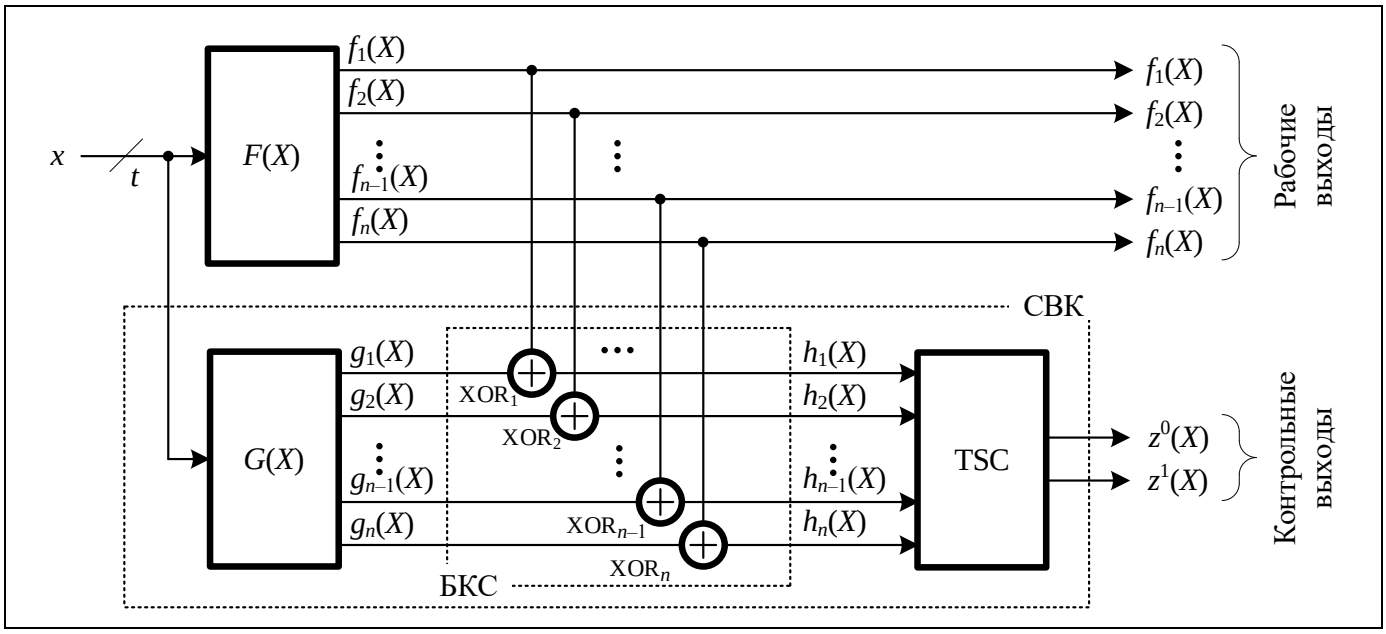

Рис. 1. Структура организации СВК на основе логической коррекции сигналов

принадлежности сформированного на выходах БКС вектора $<h_n(X)\ h_{n-1}(X)\ \ldots\ h_2(X)\ h_1(X)>$ множеству кодовых слов выбранного для контроля вычислений кода устанавливается самопроверяемый тестер TSC (*totally self-checking checker*). Данное устройство имеет два выхода $z^0(X)$ и $z^1(X)$, которые являются контрольными выходами СВК: наличие на выходах парафазного сигнала <01> или <10> свидетельствует об отсутствии ошибок в вычислениях на выходах объекта диагностирования и в блоках СВК; нарушение парафазности указывает на наличие ошибок.

Особенностью структуры, приведенной на рис. 1, является то, что для одного выбранного двоичного равномерного кода можно построить большое количество структур СВК. Это позволяет решить самую сложную задачу при организации СВК – обеспечить самопроверяемость ее структуры. При этом можно влиять на показатели структурной избыточности синтезируемого самопроверяемого устройства. Самопроверяемое же устройство, синтезируемое с СВК на основе классической структуры из работы [1], для выбранного кода при имеющейся структуре TSC и заданном способе реализации блока $G(X)$ реализуется единственным способом, что затрудняет обеспечение полной самопроверяемости СВК, а в ряде случаев делает это невозможным. В работе [18] приводится пример, демонстрирующий эту особенность классической структуры СВК для случая использования кода Бергера и кода с повторением, лежащего в основе системы дублирования. Таким образом, даже столь популярный стандартный метод, как дублирование со сравнением результатов вычислений, не всегда дает возможность построения самопроверяемой СВК из-за трудностей обеспечения самопроверяемости компаратора. Структура СВК на основе ЛКС предоставляет проектировщику возможность синтезировать самопроверяемые устройства даже в тех случаях, когда это невозможно осуществить иными методами.

Ключевым при организации СВК на основе ЛКС является выбор диагностического признака. Это может быть, как отмечалось выше, принадлежность формируемого на выходах БКС вектора $<h_n(X)\ h_{n-1}(X)\ \ldots\ h_2(X)\ h_1(X)>$ заданному равномерному двоичному коду. Тогда свойства обнаружения ошибок в кодовых словах данного кода будут определять возможности обнаружения искажений на выходах объекта диагностирования. Естественно, различные двоичные равномерные коды обладают различными свойствами обнаружения ошибок по их видам (по числу сочетаний искажений нулевых и единичных значений) и кратностям [20, 21]. Выбор кода для организации контроля вычислений становится определяющим как с позиции покрытия возникающих на выходах объекта диагностирования ошибок, так и с позиции обеспечения самопроверяемости СВК.

Для того, чтобы СВК была самопроверяемой, необходимо выполнение следующих условий. В процессе эксплуатации самопроверяемого устройства необходимо при подаче заданного множества входных воздействий обеспечить:

• Проверяемость блока $G(X)$, подразумевающую возможность проявления любой неисправно-

сти из заданной модели в виде искажений на его выходах хотя бы на одном наборе $<x_t\ x_{t-1}\ \dots\ x_2\ x_1>$ [5].

• Формирование проверяющих тестов для элементов преобразования в БКС. Полный тест элемента XOR в канонической его реализации содержит все четыре комбинации {00, 01, 10, 11} [22].

• Формирование проверяющего теста для TSC, что связано с выбранным на этапе проектирования избыточным кодом и особенностями реализации структуры самого TSC [23].

В качестве кодов, которые могут являться основой для синтеза СВК с использованием структуры, приведенной на рис. 1, могут быть выбраны любые двоичные равномерные блоковые коды (как неразделимые, так и разделимые). У неразделимых кодов в кодовых словах нельзя выделить информационные и проверочные символы, а у разделимых кодов информационные символы могут быть объединены в информационный вектор, а проверочные – в контрольный вектор. При выборе кода целесообразно отталкиваться от некоторой границы, которая определяется в случае использования неразделимых кодов мощностью множества кодовых слов заданного кода, а для разделимых кодов – числом проверочных символов кода и не должна превышать величины $k = n$ (именно такое число проверочных символов имеют коды с повторением, лежащие в основе широко используемой структуры дублирования). Дублирование дает покрытие любых сочетаний ошибок на выходах объекта диагностирования, но при этом приводит к существенной структурной избыточности – показатели сложности технической реализации устройства с СВК могут в 3-4 (и даже больше) раза превышать показатели сложности реализации самого объекта диагностирования. Поэтому часто при синтезе СВК рассматриваются коды с малой избыточностью. К таким кодам относятся неразделимые равновесные коды, а также разделимые коды с суммированием [1, 2, 11, 20, 21, 23]. Особое внимание уделяется кодам, имеющим избыточность, близкую к минимально возможной для решения задачи обнаружения ошибок на выходах объектов диагностирования. Такую нижнюю границу $k = 1$ дает использование кода с контролем четности/нечетности (кода паритета). Именно поэтому возникает интерес к исследованию возможностей применения для контроля вычислений таких избыточных равномерных кодов, которые имеют $k = 2$ и близкие к ним значения числа проверочных символов.

Для построения кода с числом проверочных символов $k = 2$ можно воспользоваться принципом подсчета числа значащих сигналов в информационном векторе в кольце вычетов по модулю $M = 4$. Тогда будут построены так называемые остаточные, или модульные коды с суммированием [1]. Они в зарубежной литературе часто называются кодами Боуза – Лина [23]. Например, в работе [24] показаны преимущества применения таких кодов при синтезе СВК на основе классической структуры с дополнением информационного вектора, формируемого на выходах объекта диагностирования, контрольным вектором. При этом продемонстрирован эффект по сравнению и с дублированием, и с применением кода Бергера для контроля вычислений. Коды Боуза – Лина можно применять и при синтезе СВК на основе ЛКС. В работе [25] приведен метод синтеза СВК на основе ЛКС с помощью данных кодов, позволяющий использовать преобразование только части значений вектора $<f_n(X)\ f_{n-1}(X)\ \dots\ f_2(X)\ f_1(X)>$, отвечающей за формирование проверочных символов кодов Боуза – Лина.

Коды Боуза – Лина, несмотря на преимущества в простоте построения, не являются оптимальными с позиции обнаружения ошибок в информационных векторах при заданных значениях m и k количества информационных и проверочных символов соответственно. Такими являются коды, у которых все 2^m информационных векторов могут быть равномерно распределены на группы, соответствующие всем 2^k контрольным векторам [26]. Существует способ построения некоторого модифицированного кода Боуза – Лина с модулем $M = 4$, обладающего таким свойством.

Задача покрытия ошибок на выходах объекта диагностирования с использованием равномерных двоичных избыточных кодов решалась во многих исследованиях. Ее решение для различных кодов приведено, например, в работах [1, 20, 21]. Сфокусируем внимание читателя на решении наиболее сложной задачи обеспечения самопроверяемости СВК.

2. «БАЗОВЫЙ» КОД ДЛЯ ОРГАНИЗАЦИИ СТРУКТУРЫ СХЕМЫ ВСТРОЕННОГО КОНТРОЛЯ

Построим модификацию кода Боуза – Лина следующим способом. Зафиксируем значение m – число информационных символов. Образуем для него информационный вектор $<y_m\ y_{m-1}\ \dots\ y_2\ y_1>$. Припишем данному информационному вектору массив весовых коэффициентов $[w_m, w_{m-1}, \dots, w_2, w_1]$, $w_i \in \mathbb{N}$. Далее будем определять не число значащих разрядов (вес информационного вектора) в

кольце вычетов по модулю $M = 4$, а значение вычета суммарного веса значащих разрядов по данному модулю:

$$W_4 = \sum_{i=1}^{m} y_i w_i \,(\mathrm{mod}\, 4).$$

Получая таким образом для каждого информационного вектора значение числа W_4, его двоичный аналог будем записывать в разряды контрольного вектора.

Введем обозначение модифицированного кода Боуза – Лина – *WS*(*m, k, M*), где $k = 2$ и $M = 4$, т. е. *WS*(*m*, 2, 4). Рассмотрим такой код при $m = 4$.

Различное сочетание значений весовых коэффициентов дает возможность построения большого количества *WS*(4, 2, 4)-кодов. При этом каждый весовой коэффициент может быть взвешен только числами из множества {1, 2, 3}, поскольку в контрольный вектор записывается в двоичном виде значение наименьшего неотрицательного вычета. При установлении значений весовых коэффициентов $w_i = 4j(\mathrm{mod}\ 4) = 0$, $j \in \mathbb{N}_0$, будет строиться непомехозащищенный код (разряды, весовой коэффициент которых кратен значению модуля, контролироваться не будут).

Для *WS*(4, 2, 4)-кодов существует 15 способов построения именно помехозащищенного кода, определяемых способами взвешивания информационных символов [w_4, w_3, w_2, w_1]: [1, 1, 1, 1], [1, 1, 1, 2], [1, 1, 1, 3], [1, 1, 2, 2], [1, 1, 2, 3], [1, 1, 3, 3], [1, 2, 2, 2], [1, 2, 2, 3], [1, 2, 3, 3], [1, 3, 3, 3], [2, 2, 2, 2], [2, 2, 2, 3], [2, 2, 3, 3], [2, 3, 3, 3], [3, 3, 3, 3]. Следует отметить, что здесь не учтены возможные перестановки весов в массивах, которые не влияют на общие свойства обнаружения ошибок в кодовых словах.

Все *WS*(4, 2, 4)-коды со значениями весовых коэффициентов, равными нечетным числам, не обнаруживают 54 ошибки в информационных векторах; *WS*(4, 2, 4)-коды со всеми четными значениями весовых коэффициентов ([2, 2, 2, 2]) не обнаруживают 112 ошибок в информационных векторах; остальные *WS*(4, 2, 4)-коды не обнаруживают минимально возможное при заданных значениях *M* и *m* число ошибок в информационных векторах – 48. Во всех символах кодовых слов все *WS*(4, 2, 4)-коды не обнаруживают 240 ошибок.

Важным и с позиции обнаружения ошибок в информационных векторах, и с позиции обеспечения самопроверяемости контрольного оборудования для выбранного разделимого кода является то, какое количество информационных векторов соответствует одному и тому же контрольному вектору. Все информационные векторы могут быть классифицированы на группы, соответствующие одному и тому же контрольному вектору. Максимальное количество контрольных векторов, которое может быть сформировано для кода с *k* проверочными символами, равно 2^k. Если все 2^m информационных векторов кода равномерно распределены между 2^k контрольными векторами, то такой код будет обнаруживать максимальное количество ошибок в информационных векторах. Кроме того, гораздо проще будет обеспечиваться свойство самопроверяемости контрольного оборудования для него. Например, тестеры разделимых кодов наиболее просто строятся в виде двухкаскадных структур, включающих в себя кодер и компаратор [27]. Для полной их проверки потребуется сформировать на входах компаратора хотя бы единожды каждый контрольный вектор. Это невозможно сделать в том случае, если число контрольных векторов у кода не максимально и не равно 2^k. К примеру, широко известные классические коды Бергера имеют максимальное количество контрольных векторов для *k* проверочных символов только при $m = 2^k - 1$. При $m \neq 2^k - 1$ для кодов Бергера не формируется полное множество контрольных векторов [28]. Задача построения полностью самопроверяемой СВК становится трудно реализуемой. Для рассматриваемых же в статье модифицированных кодов Боуза – Лина формируются все возможные контрольные векторы для *k* проверочных символов.

Рассмотрим далее для примера использование *WS*(4, 2, 4)-кода со значениями весовых коэффициентов [2, 2, 2, 3]. В табл. 1 дается классификация информационных векторов на группы, соответствующие одинаковым контрольным векторам, для выбранного *WS*(4, 2, 4)-кода. Из таблицы следует, что данный код является кодом с минимально возможным числом необнаруживаемых ошибок в информационных векторах, а также то, что каждому контрольному вектору соответствуют по четыре информационных вектора, что упрощает задачу обеспечения тестируемости кодера в СВК для реальных устройств, формирование информационных векторов на выходах которых, как правило, неравномерно.

При построении *WS*(4, 2, 4)-кода могут быть выбраны и другие весовые коэффициенты, что позволит получить другую классификационную таблицу информационных векторов по одинаковым контрольным векторам. Такая задача в настоящей статье не рассматривается. При этом подчеркнем, что кодер именно рассматриваемого *WS*(4, 2, 4)-кода, пригодного для решения задач синтеза СВК на основе ЛКС, имеющий большее количество четных коэффициентов, будет иметь одну из наиболее простых структур среди кодеров

Таблица 1

Классификация информационных векторов по одинаковым контрольным векторам

W_4			
0	1	2	3
Контрольные векторы			
00	01	10	11
Информационные векторы			
0000	0011	0010	0001
0110	0101	0100	0111
1010	1001	1000	1011
1100	1111	1110	1101

кодов с другими весовыми коэффициентами, а сам код не будет обнаруживать минимально возможное общее количество ошибок в информационных векторах.

3. «БАЗОВАЯ» СТРУКТУРА ДЛЯ СИНТЕЗА СХЕМЫ ВСТРОЕННОГО КОНТРОЛЯ

Покажем, где именно в структуре рис. 1 и как используется код Боуза – Лина.

На рис. 2 изображена «базовая» структура для синтеза СВК для группы из шести выходов объекта диагностирования на основе *WS*(4, 2, 4)-кода, в которой в СВК корректируются значения только двух функций, реализуемых объектом диагностирования, для формирования проверочных символов кода. Ранее она исследовалась, например, в работе [29], однако рассматривался случай использования весовых коэффициентов [4, 3, 2, 1], позволяющий синтезировать самопроверяемые СВК только в некоторых частных случаях (поскольку старший разряд не контролируется проверочными символами, следует исключать формирование некоторых информационных векторов).

В структуре, изображенной на рис. 2, для организации контроля использован *WS*(4, 2, 4)-код. В СВК кодовые слова данного кода $<h_6(X)\ h_5(X)\ h_4(X)\ h_3(X)\ h_2(X)\ h_1(X)>$ формируются на выходах БКС и, соответственно, на входах TSC. Младшие два разряда соответствуют проверочным символам, а старшие четыре – информационным символам. Информационные символы формируются напрямую на выходах объекта диагностирования $f_3(X)$, $f_4(X)$, $f_5(X)$, $f_6(X)$. Проверочные символы вычисляются по формулам $h_1(X) = f_1(X) \oplus g_1(X)$ и $h_2(X) = f_2(X) \oplus g_2(X)$. Функции g_1 и g_2 представляют собой функции коррекции, формируемые блоком $G(X)$. Таким образом, в СВК любой вектор $<f_6(X)\ f_5(X)\ f_4(X)\ f_3(X)\ f_2(X)\ f_1(X)>$ преобразуется в кодовое слово $<h_6(X)\ h_5(X)\ h_4(X)\ h_3(X)\ h_2(X)\ h_1(X)>$, принадлежащее *WS*(4, 2, 4)-коду, при подаче на входы наборов $<x_t\ x_{t-1}\ \ldots\ x_2\ x_1>$. Структура тестера данного кода типовая и содержит в себе кодер и один модуль сжатия парафазных сигналов TRC (*two-rail checker*) [30]. Отметим, что поскольку модуль TRC функционирует в парафазной логике, требуется предварительное инвертирование сигналов с выхода кодера *WS*(4, 2, 4)-кода либо инвертирование сигналов $h_1(X)$ и $h_2(X)$.

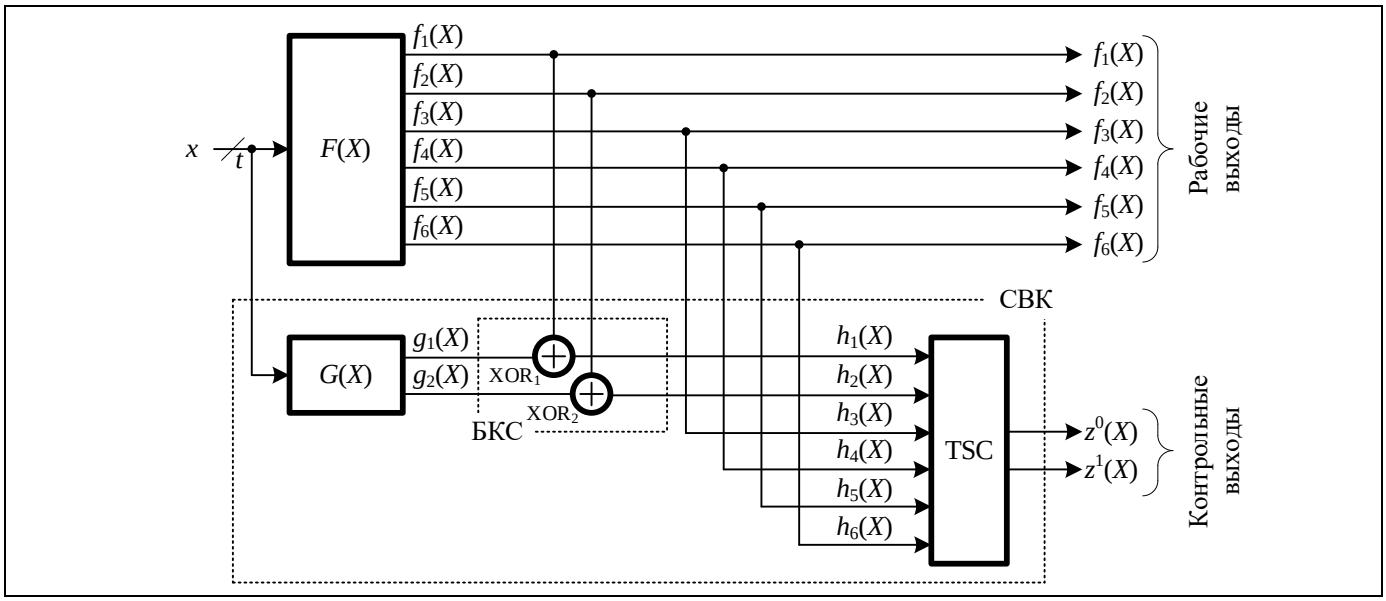

Рис. 2. Структура организации СВК с преобразованием значений части функций, реализуемых на выходах объекта диагностирования, в функции, формирующие проверочные символы *WS*(4, 2, 4)-кода

Несколько модифицируем структуру, приведенную на рис. 2. Будем корректировать значения тех функций, реализуемых объектом диагностирования, которые формируют информационные символы *WS*(4, 2, 4)-кода (рис. 3). В ней напрямую формируются только проверочные символы кодовых слов, а информационные символы вычисляются с применением блока $G(X)$. В остальном данная структура схожа с изображенной на рис. 2. Однако при ее построении можно обеспечить самопроверяемость конечного устройства в тех случаях, в которых этого не позволяет делать структура, приведенная на рис. 2, путем выбора информационных векторов, соответствующих контрольным векторам (см. табл. 1).

Особенностью модифицированной структуры является то, что контрольные векторы *WS*(4, 2, 4)-кода в СВК будут формироваться однозначно, а вот информационные векторы, которые будут им соответствовать, можно выбрать из четырех вариантов для каждого контрольного вектора (см. табл. 1). Это определяет высокую вариативность в построении «базовой» структуры и позволяет строить не один вариант блока $G(X)$, а гораздо большее их количество, что, в свою очередь, позволяет решать задачу формирования полного множества тестовых комбинаций для элементов XOR и влиять на показатели структурной избыточности СВК.

Определим число способов построения СВК по структуре, изображенной на рис. 3. Число вариантов выбора преобразуемых выходов определяется величиной $C_{4+2}^{4} = 15$. Число вариантов размещения информационных символов в информационном векторе равно $P_4 = 4! = 24$. Число вариантов размещения проверочных символов в контрольном векторе равно $P_2 = 2! = 2$. Произведение указанных величин характеризует число способов выбора последовательностей информационных и проверочных символов. Кроме того, следует учесть, что на каждой входной комбинации существует ровно по четыре способа доопределения четырех компонентов одного информационного вектора в СВК: так как входных комбинаций всего 2^t, существует $4 \cdot 2^t$ способов доопределения. Число способов построения СВК по «базовой» структуре, приведенной на рис. 3, определяется по формуле

$$N_{WS(4,\,2,\,4),\,t} = 4 \cdot 2^t \cdot 15 \cdot 24 \cdot 2 = 2\,880 \cdot 2^t. \qquad (1)$$

К примеру, если число входов равно $t = 4$, имеем $N_{WS(4,\,2,\,4),\,4} = 2\,880 \cdot 2^4 = 46\,080$.

Применение того же кода *WS*(4, 2, 4) в структуре, изображенной на рис. 2, для случая $t = 4$ дает следующее количество способов организации СВК (нельзя варьировать значения информационных символов): $N_{WS(4,\,2,\,4),\,4} = 15 \cdot 24 \cdot 2 = 720$.

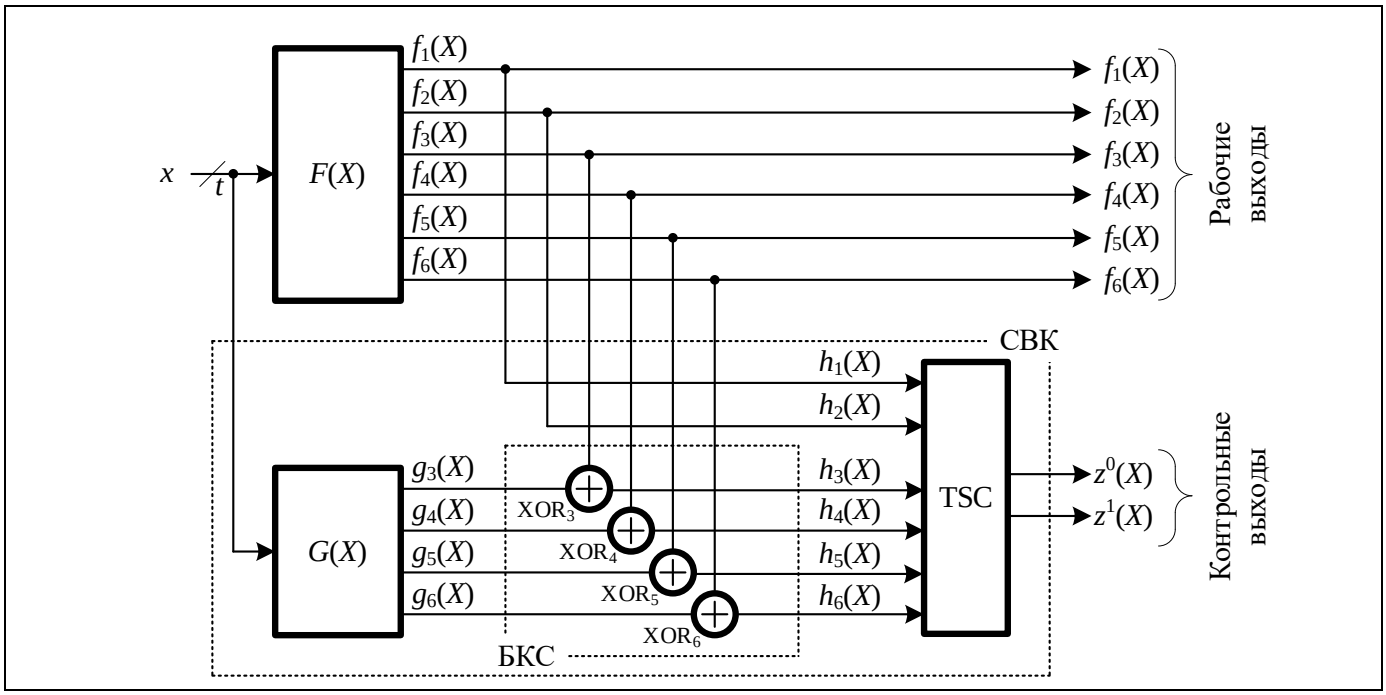

Рис. 3. Структура организации СВК с преобразованием формирующих информационные символы *WS*(4, 2, 4)-кода значений части функций, реализуемых на выходах объекта диагностирования

Полученное число в 64 раза меньше, чем для структуры, приведенной на рис. 3. Отсюда следует еще одно преимущество предложенной в настоящей работе структуры организации СВК на основе ЛКС с применением $WS(4, 2, 4)$-кода перед структурой, изображенной на рис. 2, – гораздо более высокая вариативность в построении.

«Базовая» структура организации СВК реализуется для шестивыходного объекта диагностирования. Если объект диагностирования имеет большее количество выходов, то осуществляется их разбиение на группы по шесть выходов в каждой, для которых строится отдельная СВК по «базовой» структуре с последующим объединением контрольных выходов на входах самопроверяемого компаратора.

4. АЛГОРИТМ СИНТЕЗА СХЕМЫ ВСТРОЕННОГО КОНТРОЛЯ

При построении СВК на этапе проектирования требуется обеспечить условия ее самопроверяемости. Для этого необходимо обеспечить проверяемость блока $G(X)$, что связано только с методами его синтеза и требует получения для него такой контролепригодной структуры относительно выбранной модели неисправностей, которая даст возможность проявления неисправностей в виде ошибочных значений на его выходах при подаче на входы хотя бы одного набора $<x_t\ x_{t-1}\ \dots\ x_2\ x_1>$. Также требуется подача проверяющих тестов на каждый элемент БКС и TSC. Для каждого элемента БКС требуется хотя бы единожды в процессе эксплуатации самопроверяемого устройства сформировать каждую из комбинаций из множества {00, 01, 10, 11} [22]. Для проверки тестера потребуется хотя бы единожды сформировать каждый контрольный вектор $WS(4, 2, 4)$-кода [20, 21]. Реализуем алгоритм синтеза СВК таким образом, чтобы эти условия обеспечивались в процессе функционирования самопроверяемого устройства.

Идея алгоритма состоит в следующем. При синтезе СВК требуется однозначно определить, какие именно кодовые слова $<h_6(X)\ h_5(X)\ h_4(X)\ h_3(X)\ h_2(X)\ h_1(X)>$ $WS(4, 2, 4)$-кода будут формироваться на выходах БКС при поступлении на входы каждого набора $<x_t\ x_{t-1}\ \dots\ x_2\ x_1>$. Изначально в структуре, изображенной на рис. 3, два символа однозначно определены значениями $h_1(X) = f_1(X)$, $h_2(X) = f_2(X)$, а значения символов $h_3(X)$, $h_4(X)$, $h_5(X)$ и $h_6(X)$ считаются неопределенными. Как следует из табл. 1, каждому контрольному вектору будут соответствовать по четыре информационных вектора. Таким образом, на этапе проектирования СВК потребуется на каждом входном наборе $<x_t\ x_{t-1}\ \dots\ x_2\ x_1>$ зафиксировать одно из четырех кодовых слов $WS(4, 2, 4)$-кода. Это можно сделать произвольным образом. Основной задачей в процессе синтеза СВК будет является именно доопределение значений $h_3(X)$, $h_4(X)$, $h_5(X)$ и $h_6(X)$ на выходах БКС, а затем получение значений функций коррекции, вычисляемых блоком $G(X)$. Значения функций коррекции получаются на каждом входном наборе $<x_t\ x_{t-1}\ \dots\ x_2\ x_1>$ однозначно, поскольку

$$h_i(X) = f_i(X) \oplus g_i(X) \Rightarrow$$

$$\Rightarrow g_i(X) = f_i(X) \oplus h_i(X).$$

Задача алгоритма – получение способа формирования значений информационных символов $h_3(X)$, $h_4(X)$, $h_5(X)$ и $h_6(X)$ с учетом условий обеспечения формирования проверяющих тестов для элементов БКС и TSC.

Из вышесказанного становится ясно, что существует большое разнообразие алгоритмов доопределения значений информационных символов $h_3(X)$, $h_4(X)$, $h_5(X)$ и $h_6(X)$ на выходах БКС в структуре, изображенной на рис. 3, на каждом входном наборе $<x_t\ x_{t-1}\ \dots\ x_2\ x_1>$. Рассмотрим один из них.

Алгоритм синтеза СВК на основе ЛКС с использованием $WS(4, 2, 4)$-кода для шестивыходного устройства:

Шаг 1. Для каждой функции, реализуемой на выходах устройства $F(X)$, проверяется, принимает ли она значение 0 (и 1) на не менее чем двух наборах значений аргументов. Это необходимо для того, чтобы в процессе доопределения значений информационных символов $h_3(X)$, $h_4(X)$, $h_5(X)$ и $h_6(X)$ на выходах БКС можно было сформировать проверяющие тесты для элементов XOR.

Шаг 2. Выбираются те выходы устройства $F(X)$, которые напрямую подключаются к TSC и соответствуют проверочным символам $WS(4, 2, 4)$-кода – например, выходы $f_1(X)$ и $f_2(X)$. Для них проверяется формирование хотя бы на одном входном наборе каждого сочетания значений {00, 01, 10, 11}. Это необходимо, так как требуется обеспечить условие формирования проверяющего теста для TSC. Если данное условие не выполняется, то следует выбрать другие не преобразуемые выходы. Если же такого сочетания из двух выходов нет, то построить самопроверяемую СВК рассматриваемым методом не удастся.

Шаг 3. Формируется таблица значений на выходах блоков $F(X)$, $G(X)$ и БКС, представляющая собой таблицу истинности. Данную таблицу нужно заполнить на этапе проектирования СВК. В ней

однозначно определены значения $f_i(X)$ и $h_1(X) = f_1(X)$, $h_2(X) = f_2(X)$ на каждом входном наборе $<x_t\, x_{t-1} \ldots x_2\, x_1>$. Значения функций $h_3(X)$, $h_4(X)$, $h_5(X)$, $h_6(X)$ и $g_3(X)$, $g_4(X)$, $g_5(X)$, $g_6(X)$ сначала считаются не определенными; их нужно получить в процессе доопределения.

Шаг 4. Осуществляется построчное заполнение столбцов $h_3(X)$, $h_4(X)$, $h_5(X)$ и $h_6(X)$, начиная со столбца, соответствующего старшему разряду информационного вектора. Для этого можно пользоваться табл. 1. Для функций f_5 и f_6 проверяется, принимает ли каждая из них значение 0 (и 1) на не менее чем одном наборе значений аргументов из первой их половины (№ 0 … $2^{t-1} - 1$), а также на входных наборах из второй их половины (№ $2^{t-1} \ldots 2^t - 1$). Такая проверка позволит исключить последующую проверку формирования всех комбинаций из проверяющего теста для элементов преобразования значений функций f_5 и f_6 в БКС.

Шаг 5. На первой половине наборов значений аргументов функциям $h_6(X)$ и $h_5(X)$ присваиваются значения 0, а на второй половине наборов значений аргументов – значения 1. Это позволит на первой половине входных наборов сформировать хотя бы единожды тестовые комбинации <00> и <11>, а на второй – тестовые комбинаций <01> и <10> для элементов XOR, осуществляющих преобразования $h_6(X) = f_6(X) \oplus g_6(X)$ и $h_5(X) = f_5(X) \oplus g_5(X)$.

Шаг 6. Выполняется однозначное заполнение столбцов $h_3(X)$ и $h_4(X)$, исходя из данных табл. 1. Заполнение однозначно, поскольку определены значения старших разрядов информационных векторов.

Шаг 7. Проверяется выполнение условия формирования проверяющих тестов для элементов, осуществляющих преобразования $h_3(X) = f_3(X) \oplus g_3(X)$ и $h_4(X) = f_4(X) \oplus g_4(X)$. Если условия выполняются, то переходим к шагу 9, иначе требуется изменить способ заполнения столбцов $h_5(X)$ и $h_6(X)$, путем перестановки выходов объекта диагностирования.

Шаг 8. Определяются значения функций $g_i(X) = f_i(X) \oplus h_i(X), i \in \{3, 4, 5, 6\}$.

Шаг 9. Выполняется оптимизация функций любым известным методом [31].

Шаг 10. Синтезируется блок $G(X)$ в выбранном элементном базисе.

Продемонстрируем работу алгоритма на следующем примере.

Пример. Пусть задано комбинационное устройство, описываемое в первых семи графах табл. 2. Здесь наборы входных аргументов пронумерованы десятичными числами, соответствующими двоичным числам, образуемым каждым набором входных аргументов.

На *шаге 1* выполняем проверку того, что каждая функция, реализуемая на выходах устройства $F(X)$, принимает значение 0 (и 1) на не менее чем двух наборах значений аргументов. В рассматриваемом примере это условие выполняется, что следует из анализа табл. 2. Далее на *шаге 2* определяем те выходы объекта диагностирования, которые напрямую подключаются к TSC. Положим, это выходы $f_1(X)$ и $f_2(X)$.

Шаг 3 состоит в формировании таблицы значений на выходах блоков $F(X)$, $G(X)$ и БКС (табл. 2). Также на данном шаге однозначно заполняем на каждом входном наборе $<x_t\, x_{t-1} \ldots x_2\, x_1>$ значения функций $f_i(X)$ и $h_1(X) = f_1(X)$, $h_2(X) = f_2(X)$. Значения функций $h_4(X)$ и $h_3(X)$ на данном этапе не определены. На *шаге 4* убеждаемся, что для функций f_5 и f_6 выполняется условие формирования значений 0 (и 1) на не менее чем одном из первой и второй половин наборов значений аргументов. Заполняем столбцы $h_6(X)$ и $h_5(X)$ на *шаге 5* (табл. 2). На *шаге 6* выполняем однозначное заполнение столбцов $h_3(X)$ и $h_4(X)$. Заполненная таблица представлена ниже (табл. 3).

Далее на *шаге 7* проверяем условие формирования проверяющих тестов для элементов преобразования значений функций f_4 и f_3. Проверяющие тесты формируются (см. табл. 3). Затем на *шаге 8* определяем значения функций $g_i(X) = f_i(X) \oplus h_i(X), i \in \{3, 4, 5, 6\}$.

Оптимизируем функции g_3 – g_6 (*шаг 9*). Выпишем только номера разрешенных наборов для функций $g_3 - g_6$ (или же можно выписать все конъюнкции для каждой из этих функций, на которых они принимают единичные значения): $g_6(X) = \{0, 5, 6, 7, 9, 11, 12, 13\}$, $g_5(X) = \{1, 3, 4, 5, 6, 9, 10, 13, 15\}$, $g_4(X) = \{1, 2, 4, 6, 7, 10, 13, 14\}$, $g_3(X) = \{0, 4, 7, 9, 10, 11, 13, 15\}$. Далее шаги по оптимизации тривиальны. Можно оптимизировать функции каждую в отдельности либо как систему булевых функций [31]. На *шаге 10* выбирается элементный базис и синтезируется СВК. Действия на данном шаге также не нуждаются в детальном описании. ♦

Далее определим показатель L сложности технической реализации СВК в заранее выбранной метрике (например, число входов внутренних элементов или число транзисторов, используемых при реализации устройства на конкретной элементной базе). Данный показатель может сравниваться с показателем сложности реализации СВК по методу дублирования L_D. Если $L < L_D$, то представленный метод эффективнее по сравнению с дублированием при обеспечении самопроверяемости обеих структур. Если нет, то выбирается другой способ разбиения, и шаги алгоритма повторяются.

Таблица 2

Сигналы, получаемые при формировании значений функций h_6 и h_5

№	$f_6(X)$	$f_5(X)$	$f_4(X)$	$f_3(X)$	$f_2(X)$	$f_1(X)$	$h_6(X)$	$h_5(X)$	$h_4(X)$	$h_3(X)$	$h_2(X)$	$h_1(X)$	$g_6(X)$	$g_5(X)$	$g_4(X)$	$g_3(X)$	Комбинации на входах элементов XOR			
																	XOR_6	XOR_5	XOR_4	XOR_3
0	1	0	1	1	1	0	0	0	–	–	1	0	–	–	–	–	11	00	–	–
1	0	1	1	0	0	0	0	0	–	–	0	0	–	–	–	–	00	11	–	–
2	0	0	0	1	0	1	0	0	–	–	0	1	–	–	–	–	00	00	–	–
3	0	1	1	0	1	0	0	0	–	–	1	0	–	–	–	–	00	11	–	–
4	0	1	0	0	0	1	0	0	–	–	0	1	–	–	–	–	00	11	–	–
5	1	1	1	0	1	0	0	0	–	–	1	0	–	–	–	–	11	11	–	–
6	1	1	0	1	0	1	0	0	–	–	0	1	–	–	–	–	11	11	–	–
7	1	0	1	0	1	1	0	0	–	–	1	1	–	–	–	–	11	00	–	–
8	1	1	1	0	1	0	1	1	–	–	1	0	–	–	–	–	10	10	–	–
9	0	0	1	0	0	1	1	1	–	–	0	1	–	–	–	–	01	01	–	–
10	1	0	0	1	1	0	1	1	–	–	1	0	–	–	–	–	10	01	–	–
11	0	1	0	1	0	0	1	1	–	–	0	0	–	–	–	–	01	10	–	–
12	0	1	0	0	0	0	1	1	–	–	0	0	–	–	–	–	01	10	–	–
13	0	0	1	1	0	0	1	1	–	–	0	0	–	–	–	–	01	01	–	–
14	1	1	0	1	0	1	1	1	–	–	0	1	–	–	–	–	10	10	–	–
15	1	0	1	1	1	0	1	1	–	–	1	0	–	–	–	–	10	01	–	–

Примечание. Знаком «–» заполнены столбцы таблицы, значения в строках которых до шага 7 не определены.

Таблица 3

Значения сигналов в СВК

№	$f_6(X)$	$f_5(X)$	$f_4(X)$	$f_3(X)$	$f_2(X)$	$f_1(X)$	$h_6(X)$	$h_5(X)$	$h_4(X)$	$h_3(X)$	$h_2(X)$	$h_1(X)$	$g_6(X)$	$g_5(X)$	$g_4(X)$	$g_3(X)$	Комбинации на входах элементов XOR			
																	XOR_6	XOR_5	XOR_4	XOR_3
0	1	0	1	1	1	0	0	0	1	0	1	0	1	0	0	1	11	00	10	11
1	0	1	1	0	0	0	0	0	0	0	0	0	0	1	1	0	00	11	11	00
2	0	0	0	1	0	1	0	0	1	1	0	1	0	0	1	0	00	00	01	10
3	0	1	1	0	1	0	0	0	1	0	1	0	0	1	0	0	00	11	10	00
4	0	1	0	0	0	1	0	0	1	1	0	1	0	1	1	1	00	11	01	01
5	1	1	1	0	1	0	0	0	1	0	1	0	1	1	0	0	11	11	10	00
6	1	1	0	1	0	1	0	0	1	1	0	1	1	1	1	0	11	11	01	10
7	1	0	1	0	1	1	0	0	0	1	1	1	1	0	1	1	11	00	11	01
8	1	1	1	0	1	0	1	1	1	0	1	0	0	0	0	0	10	10	10	00
9	0	0	1	0	0	1	1	1	1	1	0	1	1	1	0	1	01	01	10	01
10	1	0	0	1	1	0	1	1	1	0	1	0	0	1	1	1	10	01	01	11
11	0	1	0	1	0	0	1	1	0	0	0	0	1	0	0	1	01	10	00	11
12	0	1	0	0	0	0	1	1	0	0	0	0	1	0	0	0	01	10	00	00
13	0	0	1	1	0	0	1	1	0	0	0	0	1	1	1	1	01	01	11	11
14	1	1	0	1	0	1	1	1	1	1	0	1	0	0	1	0	10	10	01	10
15	1	0	1	1	1	0	1	1	1	0	1	0	0	1	0	1	10	01	10	11

Временна́я сложность предложенного алгоритма, как это следует из формулы (1), асимптотически оценивается величиной $2^{O(t)}$, т. е. задача решается за экспоненциальное время с линейной экспонентой.

Представленный алгоритм основан на учете того свойства *WS*(4, 2, 4)-кода со значениями весовых коэффициентов [2, 2, 2, 3], что для каждого контрольного вектора формируются по одному разу четыре информационных вектора, для которых значения старших разрядов равны 00, 01, 10 и 11 соответственно. Отметим, что таким же свойством обладают *WS*(4, 2, 4)-коды со значениями весовых коэффициентов [1, 1, 1, 2], [1, 1, 2, 3], [1, 2, 2, 3]. Данные коды также можно использовать совместно с предложенным в данном параграфе алгоритмом синтеза «базовой» структуры. Для остальных вариантов взвешивания информационных символов может быть разработан аналогичный алгоритм, например, с учетом неповторяемости других двух информационных символов в информационных векторах для каждого контрольного вектора. К примеру, *WS*(4, 2, 4)-коды со значениями весовых коэффициентов [1, 1, 1, 2], [1, 1, 2, 2], [1, 1, 2, 3], [1, 2, 2, 2], [2, 2, 2, 3], [2, 2, 3, 3], [2, 3, 3, 3] характеризуются тем, что для каждого контрольного вектора формируются четыре информационных вектора, где по одному разу второй и третий по старшинству разряды принимают значения 00, 01, 10 и 11 соответственно.

Следует также отметить важный факт: в ходе работы алгоритма не решается задача покрытия любых комбинаций ошибок на выходах объекта диагностирования. Однако при синтезе самопроверяемого устройства необходимо предварительно определиться с элементной базой, на которой реализуется устройство, с рассматриваемой моделью неисправностей и выбрать соответствующие методы синтеза [1, 11, 20, 21]. Задача покрытия ошибок, вызванных неисправностями из заданной модели, решается различными методами, – например, преобразованием структур в контролепригодные по заданному коду или выделением контролепригодных групп выходов [32, 33].

5. СИНТЕЗ СХЕМЫ ВСТРОЕННОГО КОНТРОЛЯ ДЛЯ УСТРОЙСТВ С ЧИСЛОМ ВЫХОДОВ БОЛЕЕ ШЕСТИ

Для многовыходных устройств с числом выходов $n>6$ СВК синтезируется по структуре, приведенной на рис. 4. При этом множество выходов $W = \{f_1, f_2, \ldots, f_{n-1}, f_n\}$ исходного устройства разбивается на подмножества $W_1, W_2, \ldots, W_{q-1}, W_q$ мощностью 6, $q = \left\lceil \frac{n}{6} \right\rceil$. Причем если $n(\mathrm{mod}6) = 0$, то все q групп содержат по шесть неповторяющихся выходов; если же $n(\mathrm{mod}6) \neq 0$, то q–1 группа содержит по шесть неповторяющихся выходов, а последняя группа образуется из выходов $W_q = \{f_{n-5}, f_{n-4}, f_{n-3}, f_{n-2}, f_{n-1}, f_n\}$. Для каждой группы из шести выходов объекта диагностирования синтезируется структура, изображенная на рис. 3, по приведенному выше алгоритму. Выходы каждого $\mathrm{TSC}_1 \ldots \mathrm{TSC}_q$ объединяются на входах самопроверяемого компаратора qTRC1, состоящего из q–1 модуля TRC.

Далее может быть выполнено сравнение полученного самопроверяемого устройства по показателям сложности реализации, например, с устройством, реализованным на основе дублирования. Для этого определяется показатель сложности технической реализации самопроверяемого устройства по представленному методу в заданной метрике: $L = \sum_{i=1}^{q} L_i$. Затем дается оценка эффективности по сравнению с дублированием.

Отметим, что существует довольно много способов выделения групп выходов в подмножества мощности, равной 6. Выходы для множества W_1 могут быть выбраны C_n^6 числом способов. Выходы для множества W_2 могут быть выбраны C_{n-6}^6 числом способов и т. д. Таким образом, для произвольного $j \in \{1, q-1\}$ (множество W_q выбирается единственным образом) имеем число способов, равное $C_{n-6(j-1)}^6$. Общее же число способов построения СВК определяется величиной $\prod_{j=1}^{q-1} C_{n-6(j-1)}^6$. Ее можно переписать в виде

$$\begin{aligned}\prod_{j=1}^{q-1} C_{n-6(j-1)}^6 &= C_n^6 C_{n-6}^6 C_{n-12}^6 \cdot \ldots \cdot C_{n-6(q-3)}^6 C_{n-6(q-2)}^6 = \\ &= \frac{n!}{6!(n-6)!} \cdot \frac{(n-6)!}{6!(n-12)!} \cdot \frac{(n-12)!}{6!(n-18)!} \cdot \ldots \times \\ &\times \frac{(n-6(q-3))!}{6!(n-6(q-2))!} \cdot \frac{(n-6(q-2))!}{6!(n-6(q-1))!} = \\ &= \frac{n!}{(6!)^{q-1}(n-6q+6)!}.\end{aligned} \quad (2)$$

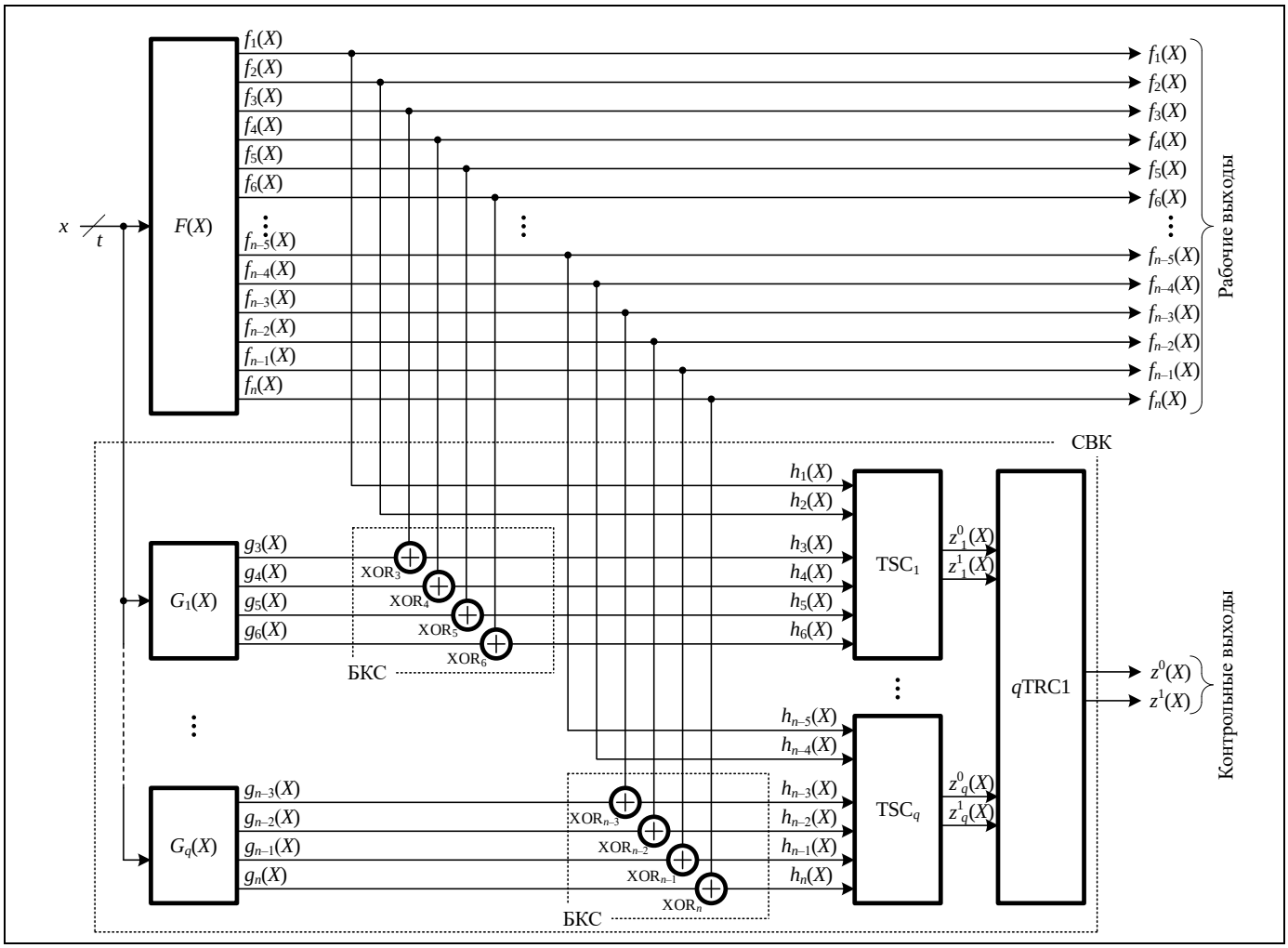

Рис. 4. Структура организации СВК для устройств с числом выходов $n>6$ на основе ЛКС с преобразованием формирующих информационные символы *WS*(4, 2, 4)-кода значений части функций, реализуемых на выходах объекта диагностирования

Учитывая, что $q=\left\lceil \frac{n}{6} \right\rceil$, выражение (2) можно представить в виде, где фигурирует только число n:

$$\prod_{j=1}^{\left\lceil \frac{n}{6} \right\rceil -1} C_{n-6(j-1)}^{6}=\frac{n!}{(6!)^{\left\lceil \frac{n}{6} \right\rceil -1}\left(n-6\left\lceil \frac{n}{6} \right\rceil +6\right)!}. \qquad (3)$$

К примеру, при числе выходов устройства $n=20$ с использованием формулы (3) получаем следующее количество способов выбора подмножеств:

$$\prod_{j=1}^{\left\lceil \frac{20}{6} \right\rceil -1=3} C_{20-6(j-1)}^{6}=\frac{20!}{(6!)^{\left\lceil \frac{20}{6} \right\rceil -1}\left(20-6\left\lceil \frac{20}{6} \right\rceil +6\right)!}=$$

$$=\frac{20!}{(6!)^{3}\,2!}=3\,259\,095\,840.$$

Необходимо отметить, что приведенные формулы дают представление об общем числе способов выбора подмножеств выходов с учетом минимально необходимого количества групп для полного их покрытия и без выбора одинаковых выходов для контроля в разных группах (за исключением, возможно, последней группы). Внесение одинаковых выходов в различные группы для контроля может потребоваться для обеспечения обнаружения требуемых сочетаний искажений на выходах. Так или иначе, отмеченное обстоятельство свидетельствует о том, что число способов организации СВК может быть увеличено по сравнению с тем, которое дают рассмотренный алгоритм и формула (3).

В заключение данного раздела обратим внимание читателя на то, что из выражения (3) следует, что задача анализа всех групп решается за факториальное время (временная сложность оценивается величиной $O(n!)$).

6. РЕЗУЛЬТАТЫ ЭКСПЕРИМЕНТОВ С ТЕСТОВЫМИ КОМБИНАЦИОННЫМИ СХЕМАМИ

Для проверки эффективности предлагаемого метода при синтезе СВК на основе ЛКС с применением *WS*(4, 2, 4)-кода были проведены эксперименты с тестовыми комбинационными схемами из набора MCNC Benchmarks [34]. В эксперименте для каждой тестовой схемы оценивались показатели сложности технической реализации устройств с СВК, синтезированных по структуре, изображенной на рис. 4. С помощью интерпретатора SIS [35, 36] осуществлялась оптимизация функций логической коррекции и синтезировался блок $G(X)$. Далее определялись показатели сложности технической реализации в условных единицах библиотеки stdcell2_2.genlib, характеризующих площадь, занимаемую устройством на кристалле. Некоторые результаты экспериментов сведены в табл. 4 и дополнительно приведены на рис. 5. В таблице для каждой схемы приведены значения ее параметров (t и n – число входов и выходов), число выделенных групп (q), а также значения показателей сложности технической реализации ($L_{F(X)}$, $L_{G(X)}$, L – условный показатель сложности реализации блоков $F(X)$, $G(X)$ и устройства с СВК). Для сравнения в графе L_D приведено значение показателя сложности реализации устройства по методу дублирования, а в последней графе таблицы – значение относительной величины δ, показывающей, какую долю составляет показатель сложности реализации устройства с СВК по представленному методу от показателя сложности реализации устройства с СВК по методу дублирования:

$$\delta = \frac{L}{L_D} \cdot 100\%.$$

При проведении эксперимента перестановки выходов тестовых схем не осуществлялись. Были выделены группы выходов в том порядке, в котором они следуют в описании тестовой схемы. Для каждой группы выходов строилась СВК, а затем выполнялись процедуры совместной реализации блоков $G_j(X)$, $j = \overline{1, q}$, и оценка показателей структурной избыточности.

Для 20-ти представленных в табл. 4 и на рис. 5 тестовых комбинационных схем получены следующие результаты. В 18-ти случаях показатель структурной избыточности снизился по сравнению с дублированием. В среднем для 20-ти рассмотренных схем получено значение δ = 81,729 %, что говорит о высокой эффективности представленного метода при синтезе устройств с СВК по показателям структурной избыточности.

Произвольное выделение групп выходов без каких-либо перестановок выходов в группах не для всех тестовых комбинационных схем дало возможность реализации самопроверяемой СВК, где формируются все тестовые комбинации для элементов преобразования и тестера. Однако простые процедуры перестановки выходов внутри групп дают эффективное решение. Приведем здесь пример работы со схемой dc1.

Изначально выходы тестовых схем не переставлялись. Для схемы dc1 это привело к формированию распределения тестовых комбинаций для элементов СВК, представленному в табл. 5. Читатель может обратить внимание на неравномерность распределения числа тестовых комбинаций, что, безусловно, связано с особенностями схемы dc1 (помимо разнообразного формирования на выходах значений логического нуля и логической единицы, у нее для входных комбинаций <1010> … <1111> для всех выходов формируются значения логического нуля). Также видно, что для элемента XOR_4 не формируется тестовая комбинация <01>. Простая перестановка выходов f_5 и f_4 без изменения способа построения СВК позволила получить перераспределение тестовых комбинаций и обеспечить формирование комбинации <01> для элемента XOR_4 в СВК для первой группы выходов. Число тестовых комбинаций для TSC_1 и TSC_2 в обоих случаях не поменялось, так как переставлялись только два выхода до этапа доопределения кодовых слов *WS*(4, 2, 4)-кода.

Перестановка выходов повлияла и на показатель структурной избыточности самопроверяемого устройства. Если для первоначального случая был получен показатель сложности совместной реализации блоков $G_j(X)$, равный 696 усл. ед. библиотеки stdcell2_2.genlib, то во втором случае этот показатель стал равным 672 усл. ед. библиотеки stdcell2_2.genlib. Это привело к уменьшению значения показателя L с 2872 до 2848 и значения показателя δ с 89,303 % до 88,557 % для данной схемы с СВК. Изменение не столь существенное.

Наличие большого числа способов перестановки выходов и выделения контролируемых групп позволяет добиваться формирования проверяющих тестов для всех элементов СВК, а также влиять на показатели структурной избыточности самопроверяемого устройства.

Таблица 4

Результаты экспериментов

№	Схема	t	n	q	$L_{F(X)}$, усл. ед.	$L_{G(X)}$, усл. ед.	L, усл. ед.	L_D, усл. ед.	δ, %
1	dc1	4	7	2	976	696	2872	3216	89,303
2	dekoder	4	7	2	736	752	2688	2736	98,246
3	wim	4	7	2	712	656	2568	2688	95,536
4	newbyte	5	8	2	592	680	2472	2656	93,072
5	p82	5	14	3	2368	1712	5976	7456	80,15
6	m1	6	12	2	3064	880	5144	8432	61,006
7	newapla2	6	7	2	600	592	2392	2464	97,078
8	sqr6	6	12	2	2648	2184	6032	7600	79,368
9	inc	7	9	2	2376	1792	5368	6432	83,458
10	newcpla2	7	10	2	1896	1440	4536	5680	79,859
11	max128	7	24	4	20192	2520	25304	45184	56,002
12	m2	8	16	3	10096	3024	15016	23328	64,369
13	m3	8	16	3	13464	3744	19104	30064	63,544
14	m4	8	16	3	18704	7152	27752	40544	68,449
15	mlp4	8	8	2	7224	9224	17648	15920	110,854
16	tms	8	16	3	6784	3032	11712	16704	70,115
17	dk27	9	9	2	528	1168	2896	2736	105,848
18	max512	9	6	1	9632	5624	15760	20320	77,559
19	newcpla1	9	16	3	2520	2528	6944	8176	84,932
20	newxcpla1	9	23	4	3760	2832	9184	12112	75,826
Среднее значение									81,729

Таблица 5

Число тестовых комбинаций для элементов СВК для схемы «dc1»

Номер группы	Элемент	I вариант				II вариант			
		00	01	10	11	00	01	10	11
1	XOR_6	5	7	1	3	5	7	1	3
	XOR_5	3	6	2	5	3	6	2	5
	XOR_4	9	0	3	4	8	1	2	5
	XOR_3	6	4	5	1	6	4	5	1
	TSC_1	6	2	1	7	6	2	1	7
2	XOR_7	4	6	2	4	4	6	2	4
	XOR_6	5	7	1	3	5	7	1	3
	XOR_5	6	3	3	4	7	2	4	3
	XOR_4	6	3	5	2	7	2	6	1
	TSC_2	6	4	2	4	6	4	2	4

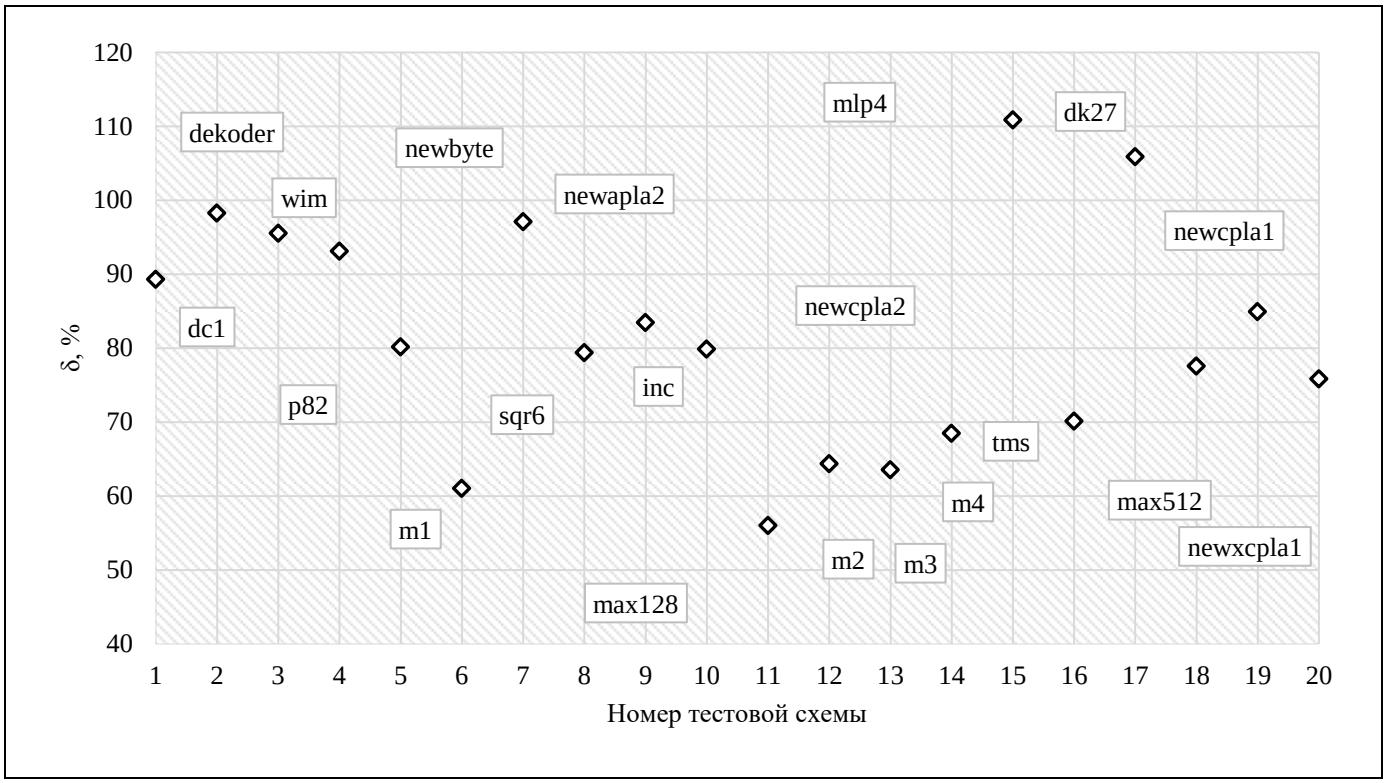

Рис. 5. Графическое представление полученных результатов

Дальнейшие исследования и эксперименты могут быть направлены на изучение возможностей минимизации значения показателя L, на обеспечение тестируемости всех компонентов СВК и поиск решений, связанных с получением равномерного распределения тестовых комбинаций для кодеров и элементов преобразования в структурах СВК. В рамках настоящего исследования эти задачи не решались.

ЗАКЛЮЧЕНИЕ

Предложенный в статье метод синтеза СВК на основе ЛКС с применением $WS(4, 2, 4)$-кода и формированием «базовой» структуры самопроверяемого устройства позволяет проектировщику самопроверяемого устройства иметь бóльшую вариативность, чем известный ранее метод, при котором в СВК корректируются только те функции, которые участвуют в формировании проверочных символов кодов. Это, в свою очередь, дает возможность оптимизации показателей структурной избыточности, контролепригодности, энергопотребления и др. для синтезируемых самопроверяемых устройств.

Как показал эксперимент, во многих случаях удается снизить показатели структурной избыточности по сравнению со стандартным методом дублирования, а примерно для половины схем удается достигать показателей $\delta = 50–70$ %. Эти результаты говорят о возможностях применения метода для практической реализации самопроверяемых цифровых устройств.

В качестве достоинств метода можно отметить следующее: если существует возможность построения самопроверяемого устройства данным методом, то он будет давать результат. Такое положение следует из необходимости для каждой функции, реализуемой на выходах устройства $F(X)$, проверять, принимает ли она значение 0 (и 1) не менее чем на двух наборах значений аргументов. Если это условие не выполняется, то задачу построения самопроверяемого устройства даже в случае применения стандартных методов, например, дублирования, решить не удастся (не будет обеспечено формирование проверяющего теста для компаратора). Кроме того, метод подразумевает безусловное формирование тестовых комбинаций для первых двух элементов преобразования из четырех. В качестве недостатка следует отметить, что для двух оставшихся элементов преобразования могут не сформироваться все комбинации, входящие в проверяющий тест (что и показал для ряда тестовых схем эксперимент). В этом случае

требуется перестановка выходов и, возможно, их перегруппировка.

Необходимо также отметить, что предложенная «базовая» структура позволяет получить большое количество вариантов формирования значений функций коррекции, а представленный алгоритм – только один из немногих. Это говорит о перспективах применения метода и возможностях формирования иных алгоритмов синтеза самопроверяемых устройств с его использованием.

Интересным развитием представленного метода является обобщенный метод построения СВК на основе ЛКС с применением взвешенных кодов Боуза – Лина с произвольным значением m, а также с различными значениями модулей $M \in \left\{2^2, 2^3, ..., 2^{\lceil \log_2(m+1) \rceil}\right\}$. Здесь существуют как минимум два направления исследований. Первое состоит в изучении особенностей применения последовательности весовых коэффициентов, образующей натуральный ряд чисел, при построении кода, поскольку наличие весовых коэффициентов, кратных значению модуля, приводит к появлению и с ростом m – к увеличению числа однократных необнаруживаемых ошибок [21]. Второе состоит в исследовании особенностей применения последовательностей весовых коэффициентов, образующих ряды с произвольными натуральными числами.

В заключение отметим, что использование ЛКС при синтезе самопроверяемых СВК – это до конца не исследованный подход к построению устройств и систем с обнаружением неисправностей, который может дать существенное улучшение показателей их эффективности по сравнению с известными методами.

ЛИТЕРАТУРА

1. *Согомонян Е.С., Слабаков Е.В.* Самопроверяемые устройства и отказоустойчивые системы. – М.: Радио и связь, 1989. – 208 с. [*Sogomonyan, E.S., Slabakov, E.V.* Samoproveryaemye ustroistva i otkazoustoichivye sistemy. – M.: Radio i svyaz', 1989. – 208 s. (In Russian)].
2. *Lala, P.K.* Self-Checking and Fault-Tolerant Digital Design. – San Francisco: Morgan Kaufmann Publishers, 2001. – 216 p.
3. *Gharibi W., Hahanov V., Chumachenko S.*, et al. Vector-Logic Computing for Faults-As-Address Deductive Simulation // IAES International Journal of Robotics and Automation. – 2023. – Vol. 12, no. 3. – P. 274–288. – DOI: 10.11591/ijra.v12i3.pp274-288.
4. *Ubar R., Raik J., Jenihhin M., Jutman A.* Structural Decision Diagrams in Digital Test: Theory and Applications. – Cham: Springer Nature Switzerland AG, 2024. – 595 p. – DOI: 10.1007/978-3-031-44734-1.
5. *Пархоменко П.П., Согомонян Е.С.* Основы технической диагностики (оптимизация алгоритмов диагностирования, аппаратурные средства). – М.: Энергоатомиздат, 1981. – 320 с. [*Parkhomenko, P.P., Sogomonyan, E.S.* Osnovy tekhnicheskoi diagnostiki (optimizatsiya algoritmov diagnostirovaniya, apparaturnye sredstva). – M.: Ehnergoatomizdat, 1981. – 320 s. (In Russian)].
6. *Дрозд А.В., Харченко В.С., Антощук С.Г.* Рабочее диагностирование безопасных информационно-управляющих систем / Под ред. А.В. Дрозда и В.С. Харченко. – Харьков: Национальный аэрокосмический университет им. Н.Е. Жуковского «ХАИ», 2012. – 614 с. [*Drozd, A.V., Kharchenko, V.S., Antoshchuk, S.G.* Rabochee diagnostirovanie bezopasnykh informatsionno-upravlyayushchikh sistem / Pod red. A.V. Drozda i V.S. Kharchenko. – Khar'kov: Natsional'nyi aehrokosmicheskii universitet im. N.E. Zhukovskogo «KHAI», 2012. – 614 s. (In Russian)].
7. *Mikoni, S.* Top Level Diagnostic Models of Complex Objects // Lecture Notes in Networks and Systems. – 2022. – Vol. 442. – P. 238–249. – DOI: 10.1007/978-3-030-98832-6_21.
8. *Bennetts, R.G.* Design of Testable Logic Circuits. – Boston: Addison-Wesley Publishing Company, 1984. – 164 p.
9. *McCluskey, E.J.* Logic Design Principles (with Emphasis on Testable Semicustom Circuits). – New Jersey: Prentice-Hall, 1986. – 549 p.
10. *Abramovici, M., Breuer, M.A., Friedman, A.D.* Digital System Testing and Testable Design. – New Jersey: IEEE Press, 1998. – 652 p.
11. *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.
12. *Sahana A.R., Chiraag V., Suresh G.*, et al. Application of Error Detection and Correction Techniques to Self-Checking VLSI Systems: An Overview // Proceedings of 2023 IEEE Guwahati Subsection Conference (GCON). – Guwahati, 2023. – DOI: 10.1109/GCON58516.2023.10183449.
13. *Mitra, S., McCluskey, E.J.* Which Concurrent Error Detection Scheme to Choose? // Proceedings of International Test Conference. – Atlantic City, 2000. – P. 985–994. – DOI: 10.1109/TEST.2000.894311.
14. *Сагалович Ю.Л., Соломенников В.Ю.* Обнаружение неисправностей в схемной реализации системы монотонных булевых функций // Проблемы передачи информации. – 1997. – Т. 33, № 2. – С. 81–93. [*Sagalovich, Yu.L., Solomennikov, V.Yu.* Obnaruzhenie neispravnostei v skhemnoi realizatsii sistemy monotonnykh bulevykh funktsii // Problemy peredachi informatsii. – 1997. – Vol. 33, no. 2. – P. 81–93. (In Russian)].
15. *Гессель М., Дмитриев А.В., Сапожников В.В., Сапожников Вл.В.* Самотестируемая структура для функционального обнаружения отказов в комбинационных схемах // Автоматика и телемеханика. – 1999. – № 11. – С. 162–174. [*Gessel', M., Dmitriev, A.V., Sapozhnikov, V.V., Sapozhnikov, Vl.V.* A Functional Fault-Detection Self-test for Combinational Circuits // Automation and Remote Control. – 1999. – Vol. 60, no. 11. – P. 1653–1663.]
16. *Ефанов Д.В., Погодина Т.С.* Исследование свойств самодвойственных комбинационных устройств с контролем вы-

числений на основе кодов Хэмминга // Информатика и автоматизация. – 2023. – Т. 22, № 2. – С. 349–392. – DOI: 10.15622/ia.22.2.5. [*Efanov, D.V., Pogodina, T.S.* Issledovanie svoistv samodvoistvennykh kombinatsionnykh ustroistv s kontrolem vychislenii na osnove kodov Khehminga // Informatika i avtomatizatsiya. – 2023. – Vol. 22, no. 2. – P. 349–392. – DOI: 10.15622/ia.22.2.5. (In Russian)].

17. *Ефанов Д.В.* Особенности реализации самопроверяемых структур на основе метода инвертирования данных и линейных кодов // Вестник Томского государственного университета. Управление, вычислительная техника и информатика. – 2023. – № 65. – С. 126–138. – DOI: 10.17223/19988605/65/13. [*Efanov, D.V.* Osobennosti realizatsii samoproveryaemykh struktur na osnove metoda invertirovaniya dannykh i lineinykh kodov // Vestnik Tomskogo gosudarstvennogo universiteta. Upravlenie, vychislitel'naya tekhnika i informatika. – 2023. – No. 65. – P. 126–138. – DOI: 10.17223/19988605/65/13. (In Russian)].

18. *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. [*Goessel' M., Morozov A.V., Sapozhnikov V.V., Sapozhnikov Vl.V.* Checking Combinational Circuits by the Method of Logic Complement // Automation and Remote Control. – 2005. – Vol. 66, no. 8. – P. 1336–1346.]

19. *Goessel, M., Graf, S.* Error Detection Circuits. – London: McGraw-Hill, 1994. – 261 p.

20. *Сапожников В.В., Сапожников Вл.В., Ефанов Д.В.* Коды с суммированием для систем технического диагностирования. Т. 1: Классические коды Бергера и их модификации. – М.: Наука, 2020. – 383 s. [*Sapozhnikov, V.V., Sapozhnikov, Vl.V., Efanov, D.V.* Kody s summirovaniem dlya sistem tekhnicheskogo diagnostirovaniya. Vol. 1: Klassicheskie kody Bergera i ikh modifikatsii. – M.: Nauka, 2020. – 383 s. (In Russian)].

21. *Сапожников В.В., Сапожников Вл.В., Ефанов Д.В.* Коды с суммированием для систем технического диагностирования. Т. 2: Взвешенные коды с суммированием. – М.: Наука, 2021. – 455 с. [*Sapozhnikov, V.V., Sapozhnikov, Vl.V., Efanov, D.V.* Kody s summirovaniem dlya sistem tekhnicheskogo diagnostirovaniya. Vol. 2: Vzveshennye kody s summirovaniem. – M.: Nauka, 2021. – 455 s. (In Russian)].

22. *Аксёнова Г.П.* Необходимые и достаточные условия построения полностью проверяемых схем свертки по модулю 2 // Автоматика и телемеханика. – 1979. – № 9. – С. 126–135. [*Aksenova, G.P.* Necessary and Sufficient Conditions for Design of Completely Checkable Modulo 2 Convolution Circuits // Automation and Remote Control. – 1979. – Vol. 40, no. 9. – P. 1362–1369.]

23. *Piestrak, S.J.* Design of Self-Testing Checkers for Unidirectional Error Detecting Codes. – Wrocław: Oficyna Wydawnicza Politechniki Wrocławskiej, 1995. – 111 p.

24. *Das, D., Touba, N.A.* Synthesis of Circuits with Low-Cost Concurrent Error Detection Based on Bose-Lin Codes // Journal of Electronic Testing: Theory and Applications. – 1999. – Vol. 15, iss. 1-2. – P. 145–155. – DOI: 10.1023/A:1008344603814.

25. *Efanov, D.V., Sapozhnikov, V.V., Sapozhnikov, Vl.V.* The Self-Checking Concurrent Error-Detection Systems Synthesis Based on the Boolean Complement to the Bose-Lin Codes with the Modulo Value M = 4 // Electronic Modeling. – 2021. – Vol. 43, iss. 1. – P. 28–45. – DOI: 10.15407/emodel.43.01.028.

26. *Ефанов Д.В., Сапожников В.В., Сапожников Вл.В.* Способ построения семейства кодов с суммированием с наименьшим общим количеством необнаруживаемых ошибок в информационных векторах // Информатика. – 2019. – Т. 16, № 3. – С. 101–118. [*Efanov, D.V., Sapozhnikov, V.V., Sapozhnikov, Vl.V.* Sposob postroeniya semejstva kodov s summirovaniem s naimen'shim obshchim kolichestvom neobnaruzhivaemyh oshibok v informacionnyh vektorah // Informatika. – 2019. – Vol. 16, no. 3. – P. 101–118. (In Russian)].

27. *Сапожников В.В., Сапожников Вл.В.* Самопроверяемые дискретные устройства. – СПб: Энергоатомиздат, 1992. – 224 с. [*Sapozhnikov, V.V., Sapozhnikov, Vl.V.* Samoproveryaemye diskretnye ustrojstva. – SPb: Energoatomizdat, 1992. – 224 s. (In Russian)].

28. *Ефанов Д.В., Сапожников В.В., Сапожников Вл.В.* О свойствах кода с суммированием в схемах функционального контроля // Автоматика и телемеханика. – 2010. – № 6. – С. 155–162. [*Efanov, D.V., Sapozhnikov, V.V., Sapozhnikov, Vl.V.* On Summation Code Properties in Functional Control Circuits // Automation and Remote Control. – 2010. – Vol. 71, no. 6. – P. 1117–1123.]

29. *Пашуков А.В.* Применение взвешенных кодов с суммированием при синтезе схем встроенного контроля по методу логического дополнения // Автоматика на транспорте. – 2022. – Т. 8, № 1. – С. 101–114. – DOI: 10.20295/2412–9186-2022-8-01-101–114. [*Pashukov, A.V.* Primenenie vzveshennykh kodov s summirovaniem pri sinteze skhem vstroennogo kontrolya po metodu logicheskogo dopolneniya // Avtomatika na transporte. – 2022. – Vol. 8, No. 1. – P. 101–114. – DOI: 10.20295/2412–9186-2022-8-01-101–114. (In Russian)].

30. *Patent* US3559167A. Self-Checking Error Checker for Two-Rail Coded Data: ser. no. 747533: filed July 25, 1968: patented Jan. 26, 1971 / Carter, W.C., Duke, K.A., Schneider, P.R.

31. *Закревский А.Д., Поттосин Ю.В., Черемисинова Л.Д.* Логические основы проектирования дискретных устройств. – М.: Физматлит, 2007. – 592 с. [*Zakrevskii, A.D., Pottosin, Yu.V., Cheremisinova, L.D.* Logicheskie osnovy proektirovaniya diskretnykh ustroistv. – M.: Fizmatlit, 2007. – 592 s. (In Russian)].

32. *Morosow, A., Sapozhnikov, V.V., Sapozhnikov, Vl.V., Goessel, M.* Self-Checking Combinational Circuits with Unidirectionally Independent Outputs // VLSI Design. – 1998. – Vol. 5, iss. 4. – P. 333–345. – DOI: 10.1155/1998/20389.

33. *Efanov, D.V., Sapozhnikov, V.V., Sapozhnikov, Vl.V.* Organization of a Fully Self-Checking Structure of a Combinational Device Based on Searching for Groups of Symmetrically Independent Outputs // Automatic Control and Computer Sciences. – 2020. – Vol. 54, iss. 4. – P. 279–290. – DOI: 10.3103/S0146411620040045.

34. *Collection* of Digital Design Benchmarks. – URL: https://ddd.fit.cvut.cz/www/prj/Benchmarks/ (дата обращения: 24.02.24). [Accessed February 24, 2024.]

35. *Sentovich, E.M., Singh, K.J., Moon, C.* Sequential Circuit Design Using Synthesis and Optimization // Proceedings of IEEE International Conference on Computer Design: VLSI in Computers & Processors. Cambridge, 1992. – P. 328–333. – DOI: 10.1109/ICCD.1992.276282.

36. *Sentovich, E.M., Singh K.J., Lavagno, L.* SIS: A System for Sequential Circuit Synthesis. – Berkeley: Electronics Research Laboratory, Department of Electrical Engineering and Computer Science, University of California, 1992. – 45 p.

Статья представлена к публикации членом редколлегии В. Г. Лебедевым.

Поступила в редакцию 15.03.2024,
после доработки 24.08.2024.
Принята к публикации 29.08.2024.

Ефанов Дмитрий Викторович – д-р техн. наук, Санкт-Петербургский политехнический университет Петра Великого, г. Санкт-Петербург; Российский университет транспорта (МИИТ), г. Москва,
✉ TrES-4b@yandex.ru
ORCID iD: https://orcid.org/0000-0002-4563-6411

Елина Есения Игоревна – аспирант, Санкт-Петербургский политехнический университет Петра Великого, г. Санкт-Петербург,
✉ eseniya-elina@mail.ru
ORCID iD: https://orcid.org/0009-0004-4167-3591

DESIGN OF SELF-CHECKING DIGITAL DEVICES WITH BOOLEAN SIGNALS CORRECTION USING WEIGHT-BASED BOSE–LIN CODES

D. V. Efanov*,** and Y. I. Yelina*

*Peter the Great Saint Petersburg Polytechnic University, St. Petersburg, Russia
**Russian University of Transport, Moscow, Russia

*✉ TrES-4b@yandex.ru, **✉ eseniya-elina@mail.ru

Abstract. This paper proposes a method for designing self-checking digital devices with Boolean signals correction and weight-based Bose–Lin codes. Unlike previous studies, the method involves Boolean signals correction (BSC) in the concurrent error-detection (CED) circuit for those functions describing the outputs of source devices that participate in the formation of data symbols of weight-based Bose–Lin codes. In such codes, as in the absolute majority of uniform separable codes, large number of data vectors correspond to the same check vector; therefore, it is possible to choose a method for determining BSC functions. We describe an algorithm for determining their values for each input combination, considering the testability of the checker and transformation elements in the CED circuit. The method involves the so-called "base" structure for monitoring multi-output devices by output groups. With this method, the designer of a self-checking device has high variability in choosing the design method and can regulate important indicators (structure redundancy, controllability, energy consumption, and others). Experiments with combinational benchmarks from MCNC Benchmarks were carried out. According to the experimental data, the method has high efficiency in terms of structure redundancy compared to the duplication method widespread in practice. The method can be effective when designing real devices with fault detection used in all areas of technology, including critical application systems in industry and transport.

Keywords: self-checking device, concurrent error-detection circuit, Boolean signals correction, weight-based sum code, weight-based Bose–Lin code.