УДК 681.518

А. В. Горелик, д-р техн. наук
И. А. Журавлев, канд. техн. наук
А. В. Орлов, канд. техн. наук
Д. В. Сперанский, д-р. техн. наук

*Кафедра «Системы управления транспортной инфраструктурой»,
Российский университет транспорта (МИИТ), Москва*

СИНТЕЗ ЦИФРОВЫХ СХЕМ ДЛЯ ГЕНЕРАЦИИ ЗАДАННЫХ ВЫХОДНЫХ ПОСЛЕДОВАТЕЛЬНОСТЕЙ

Рассматривается задача построения дискретных устройств, предназначенных для синтеза заданной выходной последовательности. Такая задача возникает при решении проблемы повышения надежности сложных дискретных систем. Облегчить решение этой проблемы возможно, например, путем применения схем встроенного контроля. В их роли могут быть использованы схемы, генерирующие на своих выходах последовательности, которые являются тестами для контроля или диагностирования особо критичных модулей сложных систем. Генерацию теста можно реализовать как комбинационными схемами, так и схемами с памятью. Предложены методы разработки таких генераторов тестов. Эти методы базируются на использовании их математических моделей в виде систем булевых функций и конечных автоматов. Показано, что задача синтеза заданного теста всегда может быть решена с использованием комбинационной схемы. Для ее упрощения предложен способ поиска несущественных переменных в ее математической модели с последующим их удалением. Для случая генератора теста в виде устройства с памятью используется модель конечного автомата, представляемого в виде таблицы соответствия между входными и выходными его словами. Такая таблица называется автоматным оператором, и он обладает специальными свойствами. Описан метод синтеза реализующего оператор автомата, который называется автоматом многократного действия. Метод проиллюстрирован на примере. Рассмотрены этапы абстрактного и структурного синтеза генераторов тестов, обсуждаются вопросы их схемной реализации.

Техническая диагностика, схемы встроенного контроля, генераторы тестов, математические модели генераторов, системы булевых функций, конечные автоматы, абстрактный и структурный синтез генераторов

DOI: 10.20295/2412-9186-2021-7-1-85-100

Введение

С ростом сложности проектируемых цифровых устройств (ЦУ) и ответственности возлагаемых на них функций растут и требования к их надежности. Для эффективного решения этой проблемы необходимо уже на этапе проектирования ЦУ заложить возможности проведения технического диагностирования. Это достигается различными путями — выбором элементной базы, подходящим

размещением функциональных узлов и блоков, организацией тестовой и функциональной диагностики. Как правило, обеспечение достоверности результатов, минимального времени обнаружения и локализации отказов реализуется схемами контроля. Опыт показывает, что различные методы диагностирования требуют применения разных подходов к выбору типа используемых средств и построению диагностической инфраструктуры. Для функционального диагностирования обычно применяется тестопригодное проектирование. При тестовом диагностировании создается отдельная инфраструктура, которая включает в себя средства генерации тестовых векторов и анализа выходных реакций диагностируемого модуля. Поддержка работоспособности и исправности функционирования всех программно-аппаратных средств ЦУ обеспечивается встроенными средствами тестового, функционального диагностирования и восстановления работоспособности, которые включаются в диагностическую инфраструктуру.

Обзор литературы по обеспечению надежности ЦУ лежит вне рамок нашей статьи, однако мы укажем несколько монографий [1–5], посвященных этой проблеме. Надежность и отказоустойчивость функционирования ЦС и их подсистем относится к стратегическому уровню защиты безопасности систем автоматического и автоматизированного управления. Обладающие отмеченными качествами ЦУ широко используются в различных областях. Например, проблемам проектирования таких устройств в области криптографии посвящены публикации [6–11], в области железнодорожного транспорта – [3, 12]. Такие же устройства востребованы в аэрокосмической отрасли, в медицине и т. д.

Один из подходов к синтезу ЦУ с обнаружением отказов состоит во внесении избыточности во внутреннюю структуру ЦУ, облегчающую решения задачи обнаружения неисправности. Она обычно решается специальной схемой контроля (тестером), в качестве которой можно использовать заранее заданную схему. Придание упомянутых свойств ЦУ возможно при его проектировании с применением либо методов помехоустойчивого кодирования, либо тестирования особо критичных модулей ЦУ. В последнем случае возникает задача синтеза довольно простой схемы (генератора теста), включаемой в состав ЦУ в качестве составляющей диагностической инфраструктуры, которая генерирует на ее выходах заданную тестовую последовательность для конкретного модуля ЦУ.

1. Постановка задачи и математические модели генератора теста

Содержательно сформулированная в конце предыдущего раздела задача в некотором смысле аналогична той, что исследовалась в [13] для линейной стационарной динамической системы, связанной с построением управлений (входных воздействий), обеспечивающих заданный выход. Математической моделью в ней являлась система дифференциальных уравнений, а искомая функция задавалась в пространстве R^n . Здесь в роли генератора теста будут рассматриваться дискретные устройства, описываемые системами булевых функ-

ций, либо системами рекуррентных уравнений переходов и выходов. Естественным требованием к интересующему нас генератору теста, рассматриваемого как встроенная в ЦУ схема, должна быть достаточная простота. Такой генератор может быть реализован либо в виде комбинационной схемы, либо в виде схемы с памятью. Далее в качестве математической модели комбинационной схемы будет использоваться система булевых функций [14]. Что касается схем с памятью, то в качестве математической модели для них применяется конечный детерминированный автомат [15]. Техническая реализация таких автоматов произвольного вида может быть выполнена с помощью триггеров. Если же автомат линейный, то наиболее приемлемы для него, как правило, модели линейных-последовательностных машин (ЛПМ) [16], реализуемых на элементах задержки.

Прежде чем перейти к постановке задачи, введем следующие обозначения:

$$\bar{u}(t) = (u_1(t), \dots, u_v(t)), \bar{y}(t) = (y_1(t), \dots, y_w(t)), \bar{s}(t) = (s_1(t), \dots, s_n(t)),$$

где $\bar{u}(t), \bar{y}(t), \bar{s}(t)$ — двоичные входной, выходной векторы и вектор состояния (в случае схемы с памятью). Число компонентов входных и выходных векторов могут быть равны.

Используя введенные обозначения, приведем формальную постановку исследуемой задачи. Задана выходная последовательность $\bar{y}_1, \bar{y}_2, \dots, \bar{y}_k$, где $\bar{y}_i$ есть w — разрядный кортеж, общее количество k которых соответствует количеству различных тестовых наборов. Требуется построить устройство (генератор теста) в виде комбинационной схемы или схемы с памятью, определить входную последовательность $\bar{u}_1, \dots, \bar{u}_k$, где $\bar{u}_i$ — v -разрядный кортеж, найти начальное состояние $\bar{s}(0)$ для случая схемы с памятью, обеспечивающее при подаче последовательности $\bar{u}_1, \dots, \bar{u}_k$ получение на выходе синтезируемого устройства (генератора теста) заданной выше выходной последовательности $\bar{y}_1, \bar{y}_2, \dots, \bar{y}_k$.

2. Синтез встроенного генератора теста в виде комбинационной схемы

ЦУ — это устройства, использующие только сигналы 0 или 1, поэтому далее рассмотрим двоичные ЦУ, описываемые моделями булевых функций. Условимся считать, что длина каждого выходного кортежа (набора) генератора теста $\bar{y}_i$ содержит w разрядов, а длина каждого входного двоичного кортежа (набора) $\bar{u}_i$ содержит v двоичных разрядов.

Начнем с выяснения вопроса, всегда ли можно сгенерировать заданный тест $\bar{y}_1, \dots, \bar{y}_k$ с помощью комбинационной схемы.

Положим величину $v = \lceil \log_2 k \rceil$, где v есть такое наименьшее целое число, что $2^v \geq k$ (округление v до ближайшего целого в большую сторону). Построим таблицу формата F , содержащую два столбца. Первый состоит из v — разрядных двоичных наборов $0...00, 0...01, \dots, 1...11$, второй — из расположенных последовательно двоичных разрядов некоторого кортежа $\bar{y}_i$. Очевидно, что таблицу F

можно рассматривать как таблицу истинности некоторой булевой функции $f_i(x_1, x_2, \dots, x_v)$ от v переменных. Действительно, в таблице F все входные наборы при таком выборе значения v попарно различны, а во втором столбце каждому из них соответствует некоторое значение функции $f_i(x_1, x_2, \dots, x_v)$. Таким образом, при принятых обозначениях имеет место следующее утверждение.

Лемма. Если заданы двоичные w -разрядные последовательности $\bar{y}_1, \dots, \bar{y}_k$, то всегда можно построить w таблиц формата F , содержащих не более k входных наборов, которые являются таблицами истинности некоторых булевых функции $f_i(x_1, x_2, \dots, x_v), i = 1, 2, \dots, w$ от v переменных, где $v = \lceil \log_2 k \rceil$, что их выходы образуют кортежи $\bar{y}_1, \dots, \bar{y}_k$.

Из леммы следует ответ на сформулированный выше вопрос — искомый генератор теста всегда может быть реализован в виде комбинационной схемы. Однако при большом числе выходов у этой схемы и большой длине заданного теста сложность ее реализации, например определяемая числом используемых элементов, также будет значительной. Таким образом, принципиальная возможность реализации генератора теста в виде комбинационной схемы еще не гарантирует, что она будет более простой, чем схема с памятью. В некоторых случаях при выборе может потребоваться анализ и сравнение по сложности обоих вариантов реализации генератора теста.

Вернемся теперь к задаче синтеза тестов генератором комбинационного типа. В этом случае необходимо предложить метод построения w булевых функций $f_i(x_1, x_2, \dots, x_v)$, при подстановке в которые подходящих входных наборов появится выходная последовательность $\bar{y}_1, \dots, \bar{y}_k$. Если такие функции существуют, то будем задавать их таблицами истинности.

Рассмотрим представленную ниже таблицу 1, построенную на основе таблиц истинности булевых функций $f_i(x_1, x_2, \dots, x_v)$. В первом слева столбце указываются порядковые номера входных наборов, а сами наборы (как их получить, будет сказано ниже) выписываются последовательно во втором столбце слева в формате $u_1(i), u_2(i), \dots, u_v(i)$, где запись $u_j(i)$ — это значение i -го двоичного разряда в j -м входном наборе u_j .

Значениями функций $f_i(x_1, x_2, \dots, x_v)$ являются i -е двоичные разряды кортежей $\bar{y}_1, \bar{y}_2, \dots, \bar{y}_k$, последовательно выписанные в виде столбцов с применением аналога только что введенных обозначений.

Таблица 1

№ П\П	$x_1 \quad x_2 \dots x_v$	$f_1(x_1, \dots, x_v)$	$f_2(x_1, \dots, x_v)$	...	$f_w(x_1, \dots, x_v)$
1	$u_1(1) u_2(1) \dots u_v(1)$	$y_1(1)$	$y_2(1)$	...	$y_w(1)$
2	$u_1(2) u_2(2) \dots u_v(2)$	$y_1(2)$	$y_2(2)$	...	$y_w(2)$
...	...	...	...	...	...
k	$u_1(k) u_2(k) \dots u_v(k)$	$y_1(k)$	$y_2(k)$	...	$y_w(k)$

Построенная описанным способом совокупность булевых функций в качестве значений при подстановке в них последовательно входных наборов второго столбца будет давать в соответствующих строках двоичные кортежи $\bar{y}_1, \bar{y}_2, \dots, \bar{y}_k$, образующие тест.

Для упрощения схемной реализации системы функций $f_i(x_1, x_2, \dots, x_v)$ необходимо найти минимальное число v существенных переменных, от которых эти функции зависят.

Проверку возможности представления булевой функции путем исключения из нее несущественных переменных можно реализовать способом, который проиллюстрируем примером. Пусть таблица 2 задает функцию от трех переменных. Исключим первый столбец для проверки переменной x_1 , после чего блоки из строк 1–4 и 5–8 таблицы 2 становятся идентичными и совпадают с таблицей 3. Это означает, что $f(x_1, x_2, x_3) = f(x_2, x_3)$, т. е. переменная x_1 несущественна. Теперь проверим переменную x_2 , удаляя в таблице 2 второй столбец вместо первого. После этого видно, что значения функции $f(x_1, x_3)$ на наборах в 1-й и 3-й строках различны, т. е. x_2 — существенная переменная. Удаляя 3-й столбец таблицы 2, приходим к выводу, что x_3 также существенная переменная. Итак, таблица истинности функции $f(x_1, x_2, x_3)$ в действительности совпадает с таблицей 3.

Таблица 2

№	$x_1 x_2 x_3$	$f(x_1, x_2, x_3)$
1	0 0 0	0
2	0 0 1	1
3	0 1 0	1
4	0 1 1	0
5	1 0 0	0
6	1 0 1	1
7	1 1 0	1
8	1 1 1	0

Таблица 3

№	$x_2 x_3$	$f(x_2, x_3)$
1	0 0	0
2	0 1	1
3	1 0	1
4	1 1	0

Заметим, что удаление из таблицы истинности функции одной несущественной переменной уменьшает число строк ее таблицы истинности в два раза.

Рассмотренный пример приводит к выводу: если в столбце значений булевой функции имеется некоторый блок расположенных подряд периодически повторяющихся значений, то это признак наличия у функции несущественной переменной. Так, в нашем примере столбец значений таблицы 3 два раза подряд повторяется в столбце значений таблицы 2.

Упомянутый признак, однако, только необходимое условие для минимизации функции, но не достаточное. Можно привести примеры таких функций, у которых после удаления нескольких несущественных переменных в разных повторяющихся блоках в оставшейся части таблицы истинности появляются одинаковые входные наборы, которым соответствуют разные значения функции. Это означает, что реализация в виде комбинационной схемы в данном случае невозможна.

Если проверка покажет, что среди функций $f_1, f_2, \dots, f_w$ имеется хотя бы одна такая функция, то реализация искомого генератора теста в целом в виде комбинационной схемы невозможна. При положительном результате проверки для каждой из функций мы располагаем соответствующей таблицей истинности, построенной описанным выше методом.

Теперь на основе полученного множества таблиц истинности можно синтезировать схему для совместной реализации системы w булевых функций $f_1, f_2, \dots, f_w$ любым из известных в теории дискретных устройств методом. Далее эта схема может быть минимизирована, для чего также известен целый ряд методов [14, 17]. Булевы функции $f_i(x_1, x_2, \dots, x_v)$ в рассматриваемом случае могут оказаться не полностью определенными, что дает дополнительные возможности для минимизации. Из изложенного вытекает метод решения задачи в случае принципиальной возможности «абстрактного» синтеза комбинационного генератора теста, и потому нет необходимости описывать его более детально.

Построенная система булевых функций, реализующих комбинационный генератор теста в виде комбинационной схемы, представляет собой его функциональную модель. Завершающим шагом конструирования генератора теста будет логическое и техническое проектирование. При логическом проектировании разрабатывается логическая структура генератора в виде схемы из элементов выбранного базиса. В результате определяется ее состав и перечень элементов. Выполнение этого этапа зависит от выбранной базовой технологии производства (МОП, КМОП, биполярной), вида схемы и требований к ней. Одна из основных задач этого этапа состоит в создании структурной модели, т. е. схемной реализации алгоритма функционирования, лучшей по некоторым важным параметрам. К их числу относятся площадь кристалла, быстродействие, тестопригодность и т. п.

Решение задачи структурного синтеза не является единственным. Поэтому не существует универсального метода синтеза, пригодного для любого базиса

и при любых требованиях. В настоящее время наиболее эффективны методы синтеза, развитые для конкретных базисов, описанных в [14, 17].

3. Синтез встроенного генератора теста в виде схемы с памятью

Остановимся теперь на синтезе генератора теста с использованием модели конечного детерминированного автомата [15]. Генератор теста должен преобразовывать входные слова $\bar{u}_1, \dots, \bar{u}_k$ в выходные слова $\bar{y}_1, \bar{y}_2, \dots, \bar{y}_k$. В исследуемой задаче множество выходных слов (тест) задано заранее. Множество входных слов может быть также задано либо его потребуется определить. Для получения генератора нужно синтезировать такой автомат, который упомянутое преобразование реализует.

Решим задачу каноническим методом синтеза автоматов [17]. Он состоит из двух этапов. Первый осуществляет абстрактный, второй — структурный синтез автомата. После выполнения первого этапа получим абстрактный автомат, представленный в виде таблицы переходов и выходов (или в виде графа). Этот автомат в общем случае не является минимальным и потому может потребовать минимизации числа состояний.

Абстрактный синтез автомата осуществим на основе таблицы соответствия между входными и выходными словами. Такая таблица называется алфавитным оператором. Напомним [17], что такой оператор называют автоматным, если он обладает следующими свойствами:

- однозначно отображает входные слова в выходные;
- удовлетворяет условиям полноты (любой начальный отрезок допустимого слова автомата также допустим);
- сохраняет длину преобразуемого слова;
- осуществляет преобразование совпадающих начальных отрезков входных слов в совпадающие начальные отрезки выходных слов.

Известно [17], если алфавитный оператор автоматный, то всегда можно синтезировать реализующий его автомат.

Если исходный оператор не является автоматным, он может быть преобразован в таковой с использованием двух операций — выравнивания и пополнения. Выравнивание используется, если нарушено третье свойство автоматности. Эта операция состоит в приписывании пустого символа «е» (он добавляется к входному и выходному алфавитам автомата) справа к входному или слева к выходному слову. Пополнение используется, если нарушено четвертое свойство автоматности, и состоит в приписывании пустого символа «е» справа к входному слову и одновременно слева к выходному слову.

В таблице 4 приведен пример применения операции выравнивания к алфавитному оператору, преобразующему двухбуквенные слова в однобуквенные.

В таблице 5 дан пример применения операции пополнения к алфавитному оператору, преобразующему двухбуквенные слова в двухбуквенные.

Таблица 4

№	$q_0 q_1$	$r_0 r_1$
1	00	e 0
2	01	e 1
3	10	e 0
4	11	e 1

Таблица 5

№	$q_0 q_1 q_2$	$r_0 r_1 r_2$
1	00 e	e 0 1
2	01 e	e 1 0
3	10 e	e 1 1
4	11 e	e 0 1

В таблице 5 первоначально одинаковые начальные отрезки из одной буквы $q_0 = 0$ в первом и втором входном словах преобразуются в $r_1 = 0$ в первом и в $r_1 = 1$ во втором выходном слове, т. е. нарушено четвертое условие автоматности. Поэтому к этим входным словам справа и к соответствующим выходным словам слева дописывается пустой символ «e». После этого одинаковые начальные отрезки первого и второго входного слова преобразуются в одинаковые начальные отрезки выходных слов из буквы «e». То же самое необходимо проделать с третьим и четвертым словом.

Для алфавитного оператора используется еще один способ представления — в виде ориентированного графа, являющегося деревом [14]. Путь (последовательность дуг) от начальной вершины графа (корня) к конечной (листу) соответствует одному входному слову. Таким образом, число уровней дерева равно длине входного слова. Из каждой вершины графа (она изображается в виде кружка между концом одной дуги и началом следующей) исходит число дуг, равное мощности входного алфавита автомата. Каждая последняя висячая вершина любого пути (лист) называется конечной.

При помощи дерева входных последовательностей осуществляется построение графа искомого конечного автомата. В дереве входных слов вершины отмечаются символами состояний автомата s_i . Начальную вершину (корень) пометим символом s_0 , а каждую конечную (лист) одним и тем же символом s_k . Для получения графа автомата Мили каждую дугу графа нагружаем парой символов q/r , где q — символ входного алфавита, r — символ выходного, а все вершины s_k совмещаем в одну.

Искомый автомат после преобразования каждого входного слова должен быть готов к преобразованию очередного входного слова. Это означает, что он

должен находиться в состоянии s_0 . Чтобы выполнить такое условие, из совмещенного в одну вершину состояния s_k необходимо провести дуги в состояние s_0 , нагруженные парами символов входного и выходного алфавита. Такой автомат называется автоматом многократного действия. В результате получаем автомат, который будет осуществлять генерацию заданного теста. В этом автомате предварительно следует выполнить минимизацию числа состояний. Проиллюстрируем изложенное на простом примере (заимствован из [17]) построения абстрактного автомата Мили по алфавитному оператору. Рассмотрим алфавитный оператор, заданный таблицей 6, который преобразует трехразрядные двоичные входные слова (первый столбец слева) в соответствующие выходные слова (второй столбец). Легко проверить, что этот оператор является алфавитным, т. е. обладает всеми четырьмя свойствами, приведенными выше.

Таблица 6

Входное слово	Выходное слово
$q_0 \ q_1 \ q_2$	$r_0 \ r_1 \ r_2$
000	001
001	000
010	001
011	000
100	001
101	000
110	001
111	000

Дерево входных последовательностей с дугами, размеченными входными и выходными словами, соответствующее таблице 6, и обозначениями состояний автомата, приведены на рисунке 1.

После объединения всех вершин s_k в одну получим граф автомата, изображенный на рисунке 2.

На втором этапе абстрактного синтеза выполним минимизацию числа состояний полученного автомата. Из рисунка 2 видно, что автомат имеет две пары эквивалентных состояний — (s_3, s_4) и (s_5, s_6) . Напомним, что два состояния автомата называются эквивалентными, если при переходе из них по одинаковым входным символам в одно и то же состояние происходит выдача одинаковых выходных символов. После их объединения (состояния обозначим как s_{34} и s_{56}) получается граф автомата с шестью состояниями. В этом полученном графе состояния s_{34} и s_{56} тоже эквивалентны и их можно объединить в одно состояние s_{3456} . Вновь полученный граф с пятью состояниями также имеет два

эквивалентных состояния s_1 и s_2 . Объединив их и обозначив новое состояние через s_{12} , в итоге получим граф с четырьмя состояниями, представленный на рисунке 3.

Завершающий шаг построения графа автомата Мили состоит в получении автомата многократного действия. С этой целью в последнем графе совместим состояние s_0 с состоянием s_k , и после переобозначений символов состояний ($S_{12} \rightarrow S_1, S_{3456} \rightarrow S_2$) получим граф на рисунке 4.

Последнему графу соответствует таблица 7 – переходы и выходы автомата.

Рис. 1. Дерево входных последовательностей

Рис. 2. Граф автомата Мили

Рис. 3. Граф автомата Мили после минимизации

Рис. 4. Граф автомата Мили многократного действия

Таблица 7

$s \backslash q$	0	1
s_0	$s_1 / 0$	$s_1 / 0$
s_1	$s_2 / 0$	$s_2 / 0$
s_2	$s_0 / 1$	$s_0 / 0$

Таблица 8

t	$t_0 \ t_1 \ t_2$
s	$s_0 \ s_1 \ s_2$
q	1 1 0
r	0 0 1

Для подтверждения правильности требуемой в задаче генерации выходных слов полученным автоматом рассмотрим, например, результат его работы при преобразовании входного слова 1 1 0 в выходное слово. Этот результат (выходное слово 0 0 1), представленный в таблице 8, подтверждает корректность функционирования автомата.

В рассмотренном примере минимизация автомата реализована с использованием его графа.

Для автоматов с большим числом состояний этот метод очень трудоемок. Очевидно, что в этом случае предпочтительнее применять известные формализованные методы минимизации числа состояний, описанные, например, в [14, 18].

После завершения абстрактного синтеза автомата выполняется очередной этап проектирования — его структурный синтез. Он имеет много общего со структурным синтезом комбинационных устройств.

Исходными данными для него, кроме таблицы переходов-выходов конечного автомата, будут состав и перечень выбранных логических элементов, типы элементов памяти и т. п. На этом этапе формулируются требования к схеме авто-

мата по сложности, быстродействию и пр. Результатом станет функциональная схема автомата из заданных логических элементов. Как и в случае с комбинационными устройствами, решение задачи структурного синтеза для конечных автоматов не является единственным.

Соответствующие методы структурного проектирования устройств с памятью в настоящее время довольно хорошо разработаны, поэтому мы на них не останавливаемся. Отметим только, что с некоторыми из этих методов можно познакомиться в работах [4, 5, 14, 17].

Заключение

В статье рассмотрена задача синтеза заданных выходных последовательностей цифровых схем, возникающая при решении проблемы повышения надежности особо ответственных сложных технических систем. К последним, несомненно, относятся системы железнодорожной автоматики, контролирующие выполнение условий безопасности движения поездов — ответственного технологического процесса [3, 12].

Упрощению решения проблемы может способствовать использование встроенных схем контроля в составе подобных систем. В качестве таких схем могут применяться генераторы тестов для контроля или диагностирования особо критичных модулей сложных систем.

Эти генераторы могут быть реализованы в виде комбинационных схем или схем с памятью.

В статье доказано, что генераторы всегда могут быть реализованы в виде комбинационных схем. Изложен способ их синтеза, базирующийся на широко известных методах синтеза комбинационных устройств. В качестве математической модели генераторов теста предполагается использование совокупности булевых функций, заданных в виде таблиц истинности. Понятно, что при большом числе выходов генератора и большой длине теста получить достаточно компактную реализацию генератора непросто.

Альтернативой комбинационному генератору теста является генератор с памятью. В статье изложен способ синтеза таких устройств, базирующийся на использовании известного канонического метода синтеза автоматов. В этом случае в качестве математической модели генератора теста используется конечный автомат многократного действия. Предполагается, что он задается с помощью алфавитного автоматного оператора, на базе которого и синтезируется искомый генератор.

При синтезе встроенного генератора теста важным критерием будет достаточная простота соответствующей схемы. Поэтому в некоторых случаях при выборе вида генератора теста может потребоваться анализ и сравнение по сложности синтезируемых генераторов как в виде комбинационной схемы, так и в виде схемы с памятью.

Библиографический список

1. *Согомонян Е. С., Слабаков Е. В.* Самопроверяемые устройства и отказоустойчивые системы. М.: Радио и связь, 1989. – 208 с.
2. *Сапожников В. В., Сапожников Вл. В.* Самопроверяемые дискретные устройства. СПб: Энергоатомиздат, Санкт-Петербург. отд-ние, 1992. – 224 с.
3. *Ефанов Д. В.* Функциональный контроль и мониторинг устройств железнодорожной автоматики и телемеханики. СПб: ФГОУ ВО ПГУПС, 2016. – 171 с.
4. *Fujiwara E.* Code Design for Dependable Systems: Theory and Practical Applications. – NY: John Wiley & Sons, 2006. – 720 p.
5. *Ubar R., Raik J., Vierhaus H.-T.* Design and Test Technology for Dependable Systems-on-Chip (Premier Reference Source) – Information Science Reference, Her-shey. – N. Y., IGI Global. – 2011. – 578 p.
6. *Bertoni G. et al.* Error analysis and detection procedures for a hardware implementation of the advanced encryption standard//IEEE Transactions on Computers. – 2003. – Т. 52. – № 4. – Pp. 492–505.
7. *Di Natale G., Flottes M. L., Rouzeyre B.* On-Line Self-Test of AES Hardware Implementations//The 37-th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, 2007. – Pp. 37–42.
8. *Blömer J., Krummel V.* Analysis of countermeasures against access driven cache attacks on AES//International Workshop on Selected Areas in Cryptography. – Springer Berlin. Heidelberg, 2007. – Pp. 96–109.
9. *Opritoiu F., Bozesan A., Vladutiu M.* Pseudo random self-test architecture or Advanced Encryption Standard//Design and Technology in Electronic Packaging (SIITME).- IEEE 19th International Symposium. – 2013. – Pp. 271–276.
10. *Mathew J. et al.* On the synthesis of attack tolerant cryptographic hardware//18th IEEE/IFIP International Conference on VLSI and System-on-Chip. – 2010. – Pp. 286–291.
11. *Дербунович Л. В., Караман Д. Г.* Методы функционального диагностирования ошибок шифрования в симметричных криптографических системах//Вестник НТУ «ХПИ». – № 57. – С. 81–86.
12. *Дербунович Л. В., Малиновский М. Л., Караман Д. Г., Мирошник М. А., Осипенко А. Н.* Метод проектирования строго безопасных автоматов локомотивной сигнализации// Інформаційно-керуючі системи на залізничному транспорті. – 2012. – № 5. – С. 25–42.
13. *Зубова С. П., Раецкая Е. В.* Построение управлений, обеспечивающих заданный выход для дискретной линейной стационарной динамической системы//Автоматика и телемеханика. – 2018. – № 5. – С. 3–23.
14. *Закревский А. Д., Поттосин Ю. В., Черемисинова Л. Д.* Логические основы проектирования дискретных устройств. – М.: ФИЗМАТЛИТ. – 2007. – 592 с.
15. *Хопкрофт Дж., Мотвани Р., Ульман Дж.* Введение в теорию автоматов, языков и вычислений. М.: Издательский дом «Вильямс». – 2002. – 528 с.
16. *Гилл А.* Линейные последовательностные машины. Анализ, синтез и применение. М.: Наука, Главная редакция физ.-мат. литературы. – 1974. – 288 с.
17. *Пестриков В. М., Дудкин В. С., Петров Г. А.* Дискретная математика: учеб. пособие. – СПб ГТУРП. – 2013. – 136 с.
18. *Гилл А.* Введение в теорию конечных автоматов. М.: Наука, Главная редакция физ.-мат. литературы. – 1966. – 272 с.

Aleksandr V. Gorelik, D.Sc.
Ilya A. Zhuravlev, Ph.D.
Aleksandr V. Orlov, Ph.D.
Dmitriy V. Speranskiy, D.Sc.

*Department of Systems of Transport Infrastructure Control
Russian University of Transport (MIIT), Moscow*

SYNTHESIS OF DIGITAL CIRCUITS FOR GENERATING GENERATED OUTPUT SEQUENCES

The problem of constructing discrete devices intended for the synthesis of a given output sequence is considered. This problem arises when solving the problem of increasing the reliability of complex discrete systems. It is possible to facilitate the solution of this problem, for example, by using built-in control schemes. In their role, circuits can be used that generate sequences at their outputs, which are tests for monitoring or diagnosing especially critical modules of complex systems.

Test generation can be implemented both by combinational circuits and circuits with memory. Methods for the development of such test generators are proposed. These methods are based on the use of their mathematical models in the form of systems of Boolean functions and finite automata. It is shown that the problem of synthesizing a given test can always be solved using a combinational circuit. To simplify it, a method is proposed to search for insignificant variables in its mathematical model with their subsequent removal.

For the case of a test generator in the form of a device with memory, the finite state machine model is used, which is represented as a table of correspondence between its input and output words. Such a table is called an automaton operator, and it has special properties. A synthesis method is described for an automaton that implements the operator, which is called an automaton of repeated action. The method is illustrated with an example. The stages of abstract and structural synthesis of test generators are considered, issues of their circuit implementation are discussed.

Technical diagnostics; built-in control circuits; test generators; mathematical models of generators; systems of boolean functions; finite state machines; abstract and structural synthesis of generators

DOI: 10.20295/2412-9186-2021-7-1-85-100

References

1. Soghomonyan E. S., Slabakov E. V. (1989) *Samoproveryayemye ustroystva i otkazoustoychevye sistemy* [Self-testing devices and fail-safe systems]. Moscow, Radio and Communication Publ., 208 p. (In Russian)
2. Sapozhnikov V. V., Sapozhnikov V. I. (1992) *Samoproveryayemye diskretnye ustroystva* [Self-test discrete devices]. Saint Petersburg, Energoatomizdat Publ., 224 p. (In Russian)
3. Efanov D. V. (2016) *Funktsionalny kontrol i monitoring ustroystv gheleznodoroghnay avtomatiki i telemekhaniki* [Functional control and monitoring of the railway automatics and telemechanics devices]. Saint Petersburg: FGOU VO PGUPS Publ., 171 p. (In Russian)
4. Fujiwara E. (2006) *Code Design for Dependable Systems: Theory and Practical Applications*. New York, John Wiley & Sons Publ., 720 p.
5. Ubar R., Raik J., Vierhaus H.-T. (2011) *Design and Test Technology for Dependable Systems-on-Chip (Premier Reference Source)*. Information Science Reference, Hershey. New York, IGI Global Publ., 578 p.

6. Bertoni G. et al. (2010) Error analysis and detection procedures for a hardware implementation of the advanced encryption standard. *IEEE Transactions on Computers*, vol. 52, no. 4, pp. 492–505.
7. Di Natale, G., Flottes, M. L., Rouzeyre B. (2007) On-Line Self-Test of AES Hardware Implementations. *The 37-th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, pp. 37–42.
8. Blömer J., Krummel V. (2007) Analysis of countermeasures against access driven cache attacks Springer on AES. *International Workshop on Selected Areas in Cryptography*. Berlin Heidelberg, Springer Publ., pp. 96–109.
9. Opritoiu F., Bozesan A., Vladutiu M. (2013) Pseudo random self-test architecture of Advanced Encryption Standard. *Design and Technology in Electronic Packaging (SIITME)*. IEEE 19th International Symposium, pp. 271–276.
10. Mathew J. et al. (2010) On the synthesis of attack tolerant cryptographic hardware. *18th IEEE/IFIP International Conference on VLSI and System-on-Chip*, pp. 286–291.
11. Derbunovich L. V., Karaman, D. G. (2015) Metody funktsionalnoy diagnostiki kriptograficheskikh oshibok v simmetrichnykh kriptograficheskikh sistemakh [Methods of the functional diagnostics of the encryption errors in the symmetric cryptographic systems]. *Vestnik NTU “KHPi” [National Technical University Kharkiv Polytechnic Institute]*, no. 57, pp. 81–86. (In Russian)
12. Derbunovich L. V., Malinovsky M. L., Karaman D. G., Miroshnik M. A., Osipenko A. N. (2012) Metody proektirovaniya strogo bezopasnykh avtomatov lokomotivnoy signalizatsii [Design method of the strictly safe automata of the locomotive signaling]. *Informatsionnye i upravlyaushie sistemy na geleznodorozhnom transporte [Information and control systems on the railway transport]*, no. 5, pp. 25–42. (In Russian)
13. Zubova S. P., Raetskaya E. V. (2018) Construction of controls providing a given output for a linear stationary dynamic system. *Automation and Remote Control*, vol. 79, no. 5, pp. 775–792.
14. Zakrevskiy A. D., Pottosin Yu. V., Cheremisinova L. D. (2007) *Logicheskie osnovy proektirovaniya diskretnykh ustroystv [Logical fundamentals of the discrete device design]*. Moscow, Fizmatlit Publ., 592 p. (In Russian)
15. Hopcroft J., Motwani R., Ullman J. (2001) *Introduction to automata theory, languages and computation*. Boston, New York, Addison-Wesley Publishing Company, Inc., 528 p.
16. Gill A. (1966) *Linear Sequential Circuits: Analysis, Synthesis, and Applications*. New York, McGraw-Hill Publ., 216 p.
17. Pestrikov V. M., Dudkin V. S., Petrov G. A. (2013) *Diskretnaya matematika: uchebnoe posobie [Textbook on discrete mathematics]*. Saint Petersburg, SPb GTURP [Petersburg State University of Industrial Technologies and Design] Publ., 136 p. (In Russian)
18. Gill A. (1962) *Introduction to the theory of finite-state machines*. New York, McGraw-Hill Publ., 212 p.

Статья представлена к публикации членом редколлегии

В. И. Хахановым

Поступила в редакцию 22.06.2020, принята к публикации 07.08.2020

ГОРЕЛИК Александр Владимирович — доктор технических наук, профессор, заведующий кафедрой «Системы управления транспортной инфраструктурой» Российского университета транспорта, Москва
e-mail: agorelik@yandex.ru

ЖУРАВЛЕВ Илья Александрович — кандидат технических наук, доцент кафедры «Системы управления транспортной инфраструктурой» Российского университета транспорта, Москва
e-mail: zhuravlev_ia@mail.ru

ОРЛОВ Александр Валерьевич — кандидат технических наук, доцент кафедры «Системы управления транспортной инфраструктурой» Российского университета транспорта, Москва
e-mail: summerman1978@gmail.com

СПЕРАНСКИЙ Дмитрий Васильевич — доктор технических наук, профессор кафедры «Системы управления транспортной инфраструктурой» Российского университета транспорта, Москва
e-mail: Speranskiy.dv@gmail.com

© Горелик А. В., Журавлев И. А., Орлов А. В., Сперанский Д. В., 2021