УДК 004.414.2

Особенности автоматизированного проектирования генераторов комбинаторного беспорядка

С.С. Соболев[1], В.К. Зольников[2], В.П. Крюков[3]

[1]ФГБОУ ВПО «Воронежская государственная лесотехническая академия», roygbiv86@gmail.com

[2]ФГУП «НИИ Электронной техники», wkz@rambler.ru

[3]ФГУП «НИИ Электронной техники», vpk@niiet.ru

***Аннотация* — В статье рассматриваются модели, алгоритмы и методы автоматизации проектирования устройств генерации перестановок без неподвижных точек (комбинаторного беспорядка). Предложена методика проектирования таких устройств, обеспечивающая высокое качество проекта и позволяющая автоматизировать процесс проектирования на системном уровне и уровне регистровых передач (RTL-уровне). В конце статьи произведена оценка эффективности подсистемы автоматизации проектирования, разработанной на основе предложенных решений.**

***Ключевые слова* — САПР, комбинаторные перестановки, комбинаторный беспорядок, системный уровень проектирования, RTL-уровень проектирования, SystemC, HDL-описание, VHDL.**

I. Введение

Одной из распространенных задач в области информационной безопасности является формирование множества полных комбинаторных перестановок целочисленных элементов. Данная задача эффективно решается с помощью аппаратных устройств генерации комбинаторных перестановок символьной строки, которые составляют отдельных класс узкоспециализированных СБИС. Разработка и исследование таких устройств представляют интерес в следующих приложениях: системы аппаратной поддержки технологий систем автоматизированного проектирования радиоэлектронной аппаратуры [1]; системы защиты информации, применяющие различные шифры на основе перестановок [2, 3]; системы аппаратного обеспечения динамического преобразования форматов данных в ЭВМ [4].

Характерными чертами специализированных устройств генерации перестановок являются переменная разрядность символьной строки и наличие ограничений на требуемое множество генерируемых комбинаторных перестановок в некоторых приложениях. Например, в системах динамического преобразования форматов на основе транспозиционных преобразований некоторые из перестановок могут быть непригодными для формирования использующихся в этих системах дескрипторов формата (так называемые недопустимые перестановки), а значит, должны быть исключены из рассмотрения. В этом прослеживается аналогия со слабыми ключами в современных шифрах, например, DES. В рассматриваемом случае перестановка является недопустимой, если она содержит одну или несколько неподвижных точек. Если множество X является множеством упорядоченных элементов, на основе которых осуществляется генерация перестановок, то неподвижной точкой перестановки p является неподвижная точка отображения p: $X \to X$, т.е. элемент множества $N(p) = \{x \in X: p(x) = x\}$. Отсюда возникает задача генерации комбинаторного беспорядка, т.е. множества перестановок без неподвижных точек.

Применение стандартных средств проектирования при разработке рассматриваемых устройств генерации комбинаторного беспорядка связано с рядом проблем. В связи с тем, что данные стандартные средства ориентированы на широкую сферу применения, они не могут полностью учитывать обозначенную выше специфику рассматриваемых устройств. При использовании стандартной методологии проектирования, описание моделей устройства на системном уровне и уровне регистровых передач (RTL-уровне) осуществляется чаще всего вручную, путем графического или текстового ввода, что, во-первых, увеличивает трудоемкость и требует недопустимо больших затрат времени, а, во-вторых, влечет за собой разрыв между алгоритмическим описанием и RTL-описанием. Аналогичный разрыв наблюдается и при разработке тестов для верификации блоков на системном и RTL-уровнях.

Обозначенные проблемы, а также многообразие входных параметров генерации комбинаторных перестановок и увеличение сложности схемы блока генерации с ростом длины символьной строки, повышают риск субъективных ошибок и, таким образом, увеличивают влияние «человеческого фактора» на качество всего проекта. В результате использование традиционной методологии в процессе проектирования рассматриваемых устройств представляется затруднительным. В этой связи актуальной задачей представляется разработка средств автоматизации проектирования устройств генерации комбинаторного беспорядка. В настоящей статье представлены методика, модели, методы и алгоритмы для автоматизации проектирования рассматриваемых целевых устройств, позволяющие автоматизировать процесс проектирования на систем-

ном и RTL-уровнях, а также обеспечивающие высокое качество всего проекта. Кроме того, приводятся результаты использования предлагаемых решений.

II. Методика проектирования

Важнейшим принципом, на котором основывается методика проектирования современных СБИС, является использование в проекте уже готовых функционально законченных блоков (СФ-блоков, или IP-блоков) [5].

Специфика устройств генерации полных комбинаторных перестановок элементов символьной строки с отсевом недопустимых перестановок связана с их следующими особенностями:

- длина символьной строки, а также разрядность одного элемента символьной строки, для которой производится генерация комбинаторных перестановок, является переменной величиной;
- максимально допустимое количество неподвижных точек является переменной величиной.

За счет приведенных особенностей средства проектирования рассматриваемых устройств дополнены специальными средствами, которые значительно сократят время проектирования. В рамках проектирования на системном уровне предложено:

- в качества языка описания поведенческой модели использовать SystemC – язык проектирования и верификации моделей системного уровня [6];
- генерировать программные IP-блоки на SystemC с помощью программного модуля, реализованного на высокоуровневом языке C++, на основе следующих входных параметров: разрядность одного элемента символьной строки, для которой производится генерация комбинаторных перестановок; длина данной символьной строки, или мощность множества, для которого формируются перестановки; количество неподвижных точек в каждой из сгенерированных перестановок, достаточное для ее отсеивания;
- генерировать тесты для верификации поведенческой модели также с помощью программного модуля на C++ с тем же множеством входных параметров.

Полученные на системном уровне спецификации

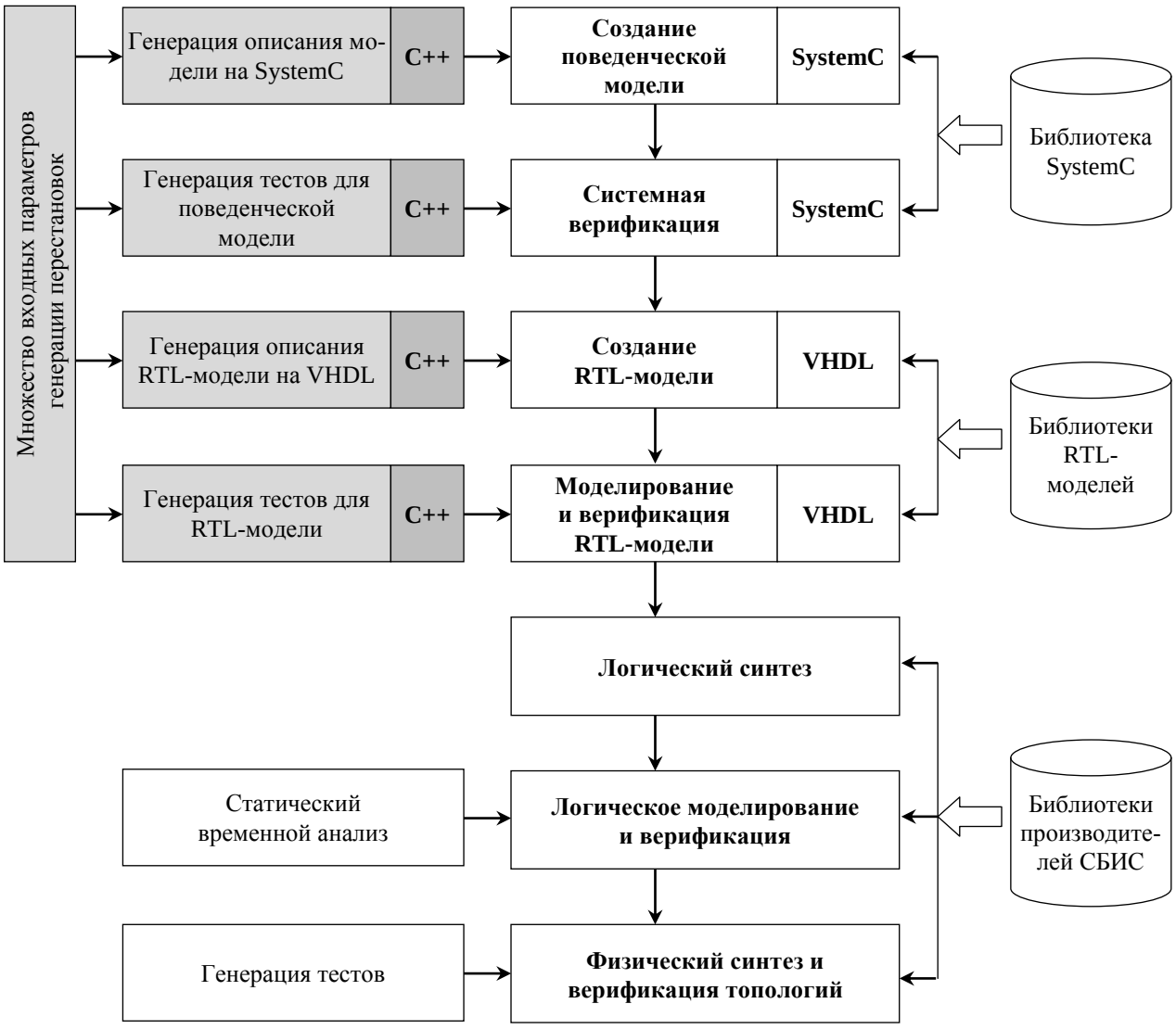

Рис. 1. Маршрут проектирования генераторов комбинаторного беспорядка

являются исходной точкой для следующего этапа: проектирование цифровых, цифро-аналоговых и аналоговых блоков. Отметим, что проектирование каждого из этих блоков может осуществляться параллельно. Остановимся более подробно на последующем маршруте проектирования цифрового блока генерации перестановок. Этот уровень часто называют логическим, или вентильным.

На этапе разработки функционального описания блока на уровне регистровых передач (RTL) создается синтезируемая RTL-модель блока на одном из языков описания аппаратуры. В рамках проектирования рассматриваемых устройств генерации перестановок, в качестве такого языка было предложено использовать язык VHDL [7, 8]. Данный этап выполняется в автоматическом режиме, предполагающем генерацию IP-блоков на VHDL с помощью программного модуля на языке C++. Учитывая специфику блока генерации полных комбинаторных перестановок элементов символьной строки, модуль принимает на вход то же множество входных параметров, что и для автоматизированного формирования SystemC модели на системном уровне.

Следующий этап включает в себя моделирование и верификацию RTL-модели. Верификация модели предполагает прохождение предварительно сформированного набора тестов, специфичных для блока генерации перестановок. Тесты для верификации на RTL-уровне генерируются также с помощью программного модуля на языке C++ с тем же множеством входных параметров, что и для модели на системной уровне. Данный метод позволяет произвести откат к системному уровню в случае неудачи верификации. Примером может послужить ситуация, когда какие-либо параметры, заложенные в блок на системном уровне, не реализуются, и тогда можно изменить саму системную модель, внеся необходимые правки в программный модуль генерации модели.

Дальнейший маршрут проектирования включает в себя логический синтез, логическое моделирование и верификацию, физический синтез и верификацию топологии. На выходе маршрута проектирования должны быть получены топология, список цепей и производственные тесты, которые далее передаются на производство.

Общий маршрут проектирования устройств генерации комбинаторного беспорядка при использовании предложенной методики проектирования изображен на рис. 1 (элементы, отличающиеся новизной, отмечены серой заливкой).

III. Метод генерации функциональных блоков

Для обеспечения гибкости системной и RTL-моделей генератора перестановок используется множество входных параметров, позволяющих по требованию модифицировать основные характеристики устройства. На основе этих параметров генерируются поведенческая модель на SystemC и RTL-модель на VHDL, отвечающие всем входным аргументам. Множество параметров, позволяющих гибко настраивать указанные модели, включает в себя:

- d – разрядность одного элемента символьной строки, для которой производится генерация комбинаторных перестановок (в битах);
- n – длина данной символьной строки, или мощность множества, для которого формируются перестановки;
- N_{max} – максимальное количество неподвижных точек в каждой из сгенерированных перестановок, при котором перестановка не отсеивается.

Для достижения большей адаптивности моделей, их генерация может происходить в двух режимах (общая схема представлена на рис. 2).

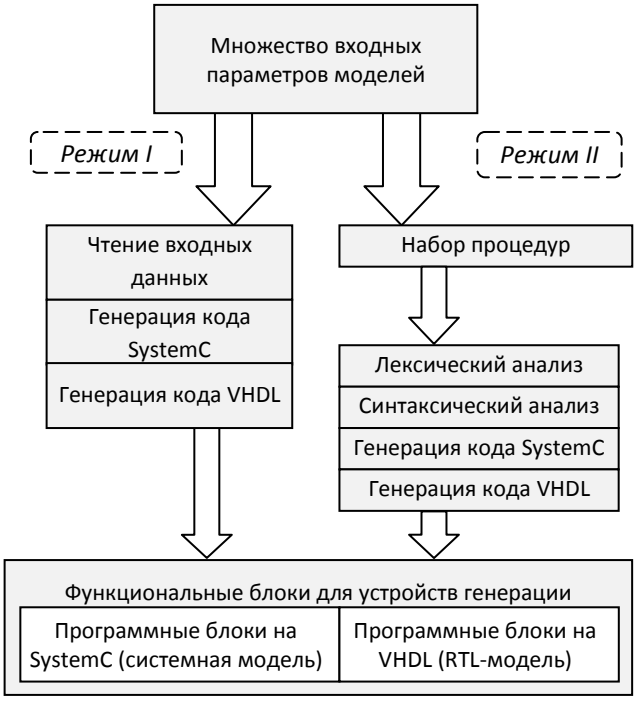

Рис. 2. Схема генерации функциональных блоков для генераторов комбинаторного беспорядка

При использовании первого режима, блоки на SystemC и VHDL генерируются напрямую с использованием входных параметров блока генерации: d, n, N_{max}.

Второй режим позволяет тонко конфигурировать модели и обеспечивает возможность трансформации системы с минимальной затратой ресурсов. Режим предусматривает процесс генерации блоков в несколько этапов. Первый этап – это формирование набора процедур, каждая из которых может принимать в качестве входных аргументов один или несколько вышеприведенных параметров, а также ряд других опциональных параметров (например, тактируемость блока, наличие порта асинхронного сброса RESET). Каждая процедура позволяет настраивать отдельные части моделей и регулировать взаимодействие между ними.

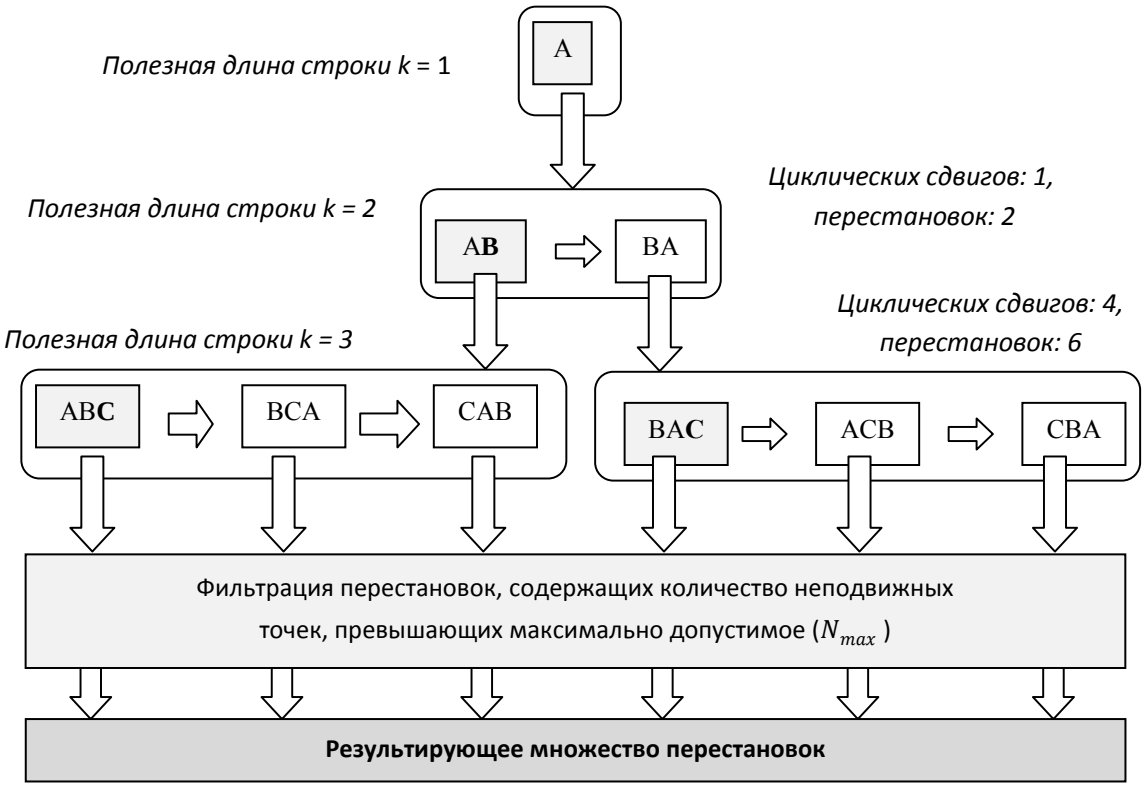

Рис. 3. Схема алгоритма генерации перестановок для $n = 3$

Второй этап заключается в синтаксическом разборе списка процедур и генерации кода конечных моделей на SystemC и VDHL. Эти действия осуществляет программный модуль на языке C++. Программный модуль включает в себя:

- лексический и синтаксический анализатор;
- шаблоны модулей, сигналов SystemC, необходимые для описания поведенческой модели генерации перестановок; шаблоны объектов, сигналов VHDL, необходимые для описания RTL-модели генерации перестановок;
- генератор кода SystemC/VHDL с использованием этих шаблонов.

Следует обратить внимание, что второй режим используется только в случае необходимости в тонкой конфигурации IP-блоков и, в отличие от первого режима, включает синтаксический и лексический анализ специального набора процедур конфигурации, заранее подготовленного проектировщиком.

Верификация системной модели и RTL-модели производится с помощью специализированного тестового окружения под управлением подсистемы верификации.

IV. Алгоритм генерации комбинаторного беспорядка

Для генерации комбинаторных перестановок элементов символьной строки длины n с возможностью отсева недопустимых перестановок предложен алгоритм на основе циклического сдвига, представляющий собой комбинацию следующих последовательно применяемых элементарных операций: добавление нового символа в строку (увеличение полезной длины строки) и циклический сдвиг строки [9]. Подобная структура делает данный алгоритм очень удобным для аппаратной реализации. Условная схема алгоритма представлена на рис. 3.

V. Моделирование генерации комбинаторного беспорядка

Общая структура подсистемы генерации полных комбинаторных перестановок представлена на рис. 4. Она включает в себя несколько блоков, каждый из которых имеет свою функциональность.

Рис. 4. Общая структура генератора комбинаторного беспорядка

Каждому блоку на рис. 4 соответствует отдельный модуль в системной модели SystemC.

Задача формирователя исходной символьной строки – сформировать символьную строку S_0 фиксированной длины n, множество элементов которой представляет собой упорядоченный набор целых чисел от 0

до $n-1$. Таким образом, элементами исходной символьной строки S_0 являются элементы множества

$$X_0 = \{ x_i \in \mathbb{Z}: x_i = i,\ i = \overline{0, n-1} \}, \quad (1)$$

Модуль формирования очередной уникальной перестановки включает в себя n-1 модулей циклического сдвига и n-1 вспомогательных модулей управления. Модули циклического сдвига соединены последовательно и имеют уровень от 2 до n в соответствии с увеличением полезной длины символьной строки согласно алгоритму на основе циклического сдвига.

Каждый модуль циклического сдвига обладает портами управления L и S. При высоком логическом уровне сигнала на входе L значения элементов символьной строки считываются с входных портов и тут же передаются на выходные порты (либо на следующий уровень m+1, либо в качестве результирующей перестановки), без циклического сдвига. При высоком логическом уровне сигнала на входе S, происходит циклический сдвиг строки, хранящейся во внутреннем массиве модуля, и её передача на выходные порты. Значения сигналов на портах L и S вычисляются согласно значениям функций $L = L(c, m, n)$ и $S = S(c, m, n)$, где m – уровень модуля, а c – внутренний счетчик вспомогательного модуля управления, увеличивающийся с каждым тактовым импульсом.

$$L(c,m,n) = \left[c \bmod \left(\prod_{i=m}^{n} i \right) = 0 \right],$$

$$S(c,m,n) = \quad (2)$$

$$= \begin{cases} \overline{L(c,m,n)}, & m = n \\ \overline{L(c,m,n)} \wedge \left[\left(c \bmod \left(\prod_{i=m}^{n} i \right) \right) \bmod \left(\prod_{i=m+1}^{n} i \right) = 0 \right], & m < n \end{cases}$$

Общая схема модуля формирования уникальной перестановки для n=4 изображена на рис. 5. На входные порты X0, X1, X2, X3 поступают элементы исходного множества, на основе которого формируются перестановки. На выходе у схемы (порты Y0, Y1, Y2, Y3) элементы очередной сгенерированной перестановки.

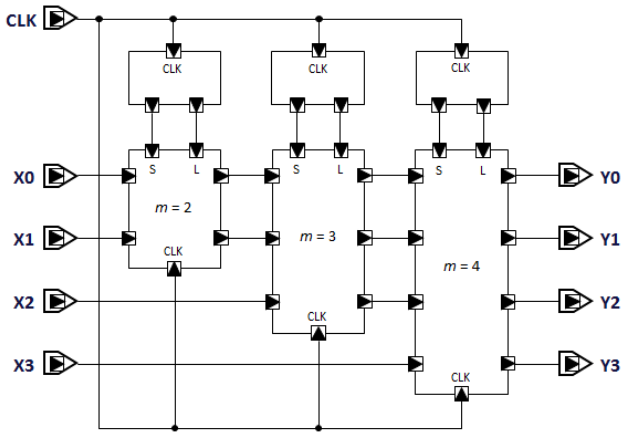

Рис. 5. Общая схема модуля формирования уникальной перестановки для n=4

Разработка RTL-описания блока генерации полных комбинаторных перестановок с отсевом недопустимых перестановок осуществлялась с применением языка описания аппаратуры VHDL [7, 8]. Базовым элементом любого проекта на VHDL является объект. Блок генерации перестановок представляет собой совокупность моделируемых объектов VHDL, каждый из которых соответствует отдельному блоку на общей схеме, изображенной на рис. 4.

Соединяющие объекты сигналы могут быть единичными или множественными (векторами). Каждый передаваемый элемент перестановки моделируется на RTL-уровне в виде d-разрядного вектора, а вся перестановка – в виде $(n \cdot d)$-разрядного вектора.

Формирователь исходной символьной строки представляет собой $(n \cdot d)$-разрядный регистр с параллельным вводом и выводом. Регистр хранит в себе элементы исходного упорядоченного множества X_0, на основе которого осуществляется генерация перестановок.

Формирователь очередной уникальной перестановки представляет собой набор регистров циклического сдвига с параллельным вводом и выводом, работа которых координируется вспомогательными объектами управления. Всего объект формирования уникальной перестановки содержит $d \cdot (n-1)$ регистров (n-1 групп по d регистров) сдвига и n-1 вспомогательных объектов управления. Ширина регистров в каждой группе ступенчато увеличивается от 2 до n бит с шагом 1 в соответствии с увеличением полезной длины символьной строки согласно алгоритму на основе циклического сдвига. Каждый вспомогательный объект управления координирует работу своей группы из d регистров с помощью сигналов управления, поступающих на входные управляющие порты регистров.

Для управления элементами перестановки, всякий регистр, помимо информационных портов, также обладает и двумя входными портами управления L и S. По переднему фронту сигнала на порте L регистр считывает соответствующие биты элементов перестановки с входных информационных портов. По переднему фронту сигнала на порте S в регистре осуществляется сдвиг на 1 бит. При низком уровне сигнала на обоих портах текущее состояние регистра остается неизменным.

Вспомогательный объект управления включает в себя два счетчика: счетчик по модулю M_1 и счетчик по модулю M_2. Счетчики устроены таким образом, что каждый из них по достижении своего максимального значения подает на выход высокий уровень сигнала, в противном случае – низкий уровень. Внутреннее состояние счетчика увеличивается при подаче очередного тактирующего импульса на входной порт тактирующего сигнала и сбрасывается по достижении максимального значения. Согласно значениям сигналов на выходе обоих счетчиков рассчитываются уровни управляющих сигналов L и S. Значения M_1 и M_2 вычисляются по формулам:

$$M_1 = \prod_{i=l}^{n} i, \quad M_2 = \begin{cases} 1, & l = n \\ \prod_{i=l+1}^{n} i, & l < n \end{cases}, \qquad (3)$$

где l – разрядность соответствующего управляемого регистра.

VI. Результаты

Для автоматизации проектирования генераторов полных комбинаторного беспорядка разработан комплекс проблемно-ориентированного программного обеспечения, внедряемый в общий маршрут проектирования данных устройств. В рамках данного программного комплекса реализованы модели, методы и алгоритмы для формирования множества комбинаторных перестановок элементов символьной строки с возможностью отсева недопустимых перестановок, содержащих неподвижные точки. Программное обеспечение включает в себя набор инструментальных средств для решения задачи автоматизации проектирования целевых устройств генерации перестановок на системном (поведенческом) уровне и уровне регистровых передач (RTL-уровне).

Результирующие IP-блоки предназначены для использования в маршруте проектирования специализированных устройств генерации комбинаторных перестановок элементов символьной строки, применяемых в устройствах аппаратной поддержки технологий систем автоматизированного проектирования радиоэлектронной аппаратуры, в системах защиты информации, а также в системах аппаратного обеспечения динамического преобразования форматов данных в ЭВМ.

Для оценки эффективности разработанных средств определена сложность логической схемы каждого из семейства аппаратных генераторов перестановок путем подсчета общего количества логических вентилей и строк VHDL-кода для соответствующих устройств. В соответствии с полученными результатами произведена оценка временных затрат проектирования с учетом средней скорости разработки 100 вентилей в день (эта оценка является стандартной и неизменной на протяжении последних пяти лет [10]). Результаты представлены в таблице 1.

Таким образом, автоматизация процессов аппаратного описания и верификации на системном и RTL-уровнях позволяет существенно сократить цикл проектирования рассматриваемых устройств за счет исключения процесса ручного описания на одном из HDL-языков. Как отражено в таблице 1, этот процесс обладает значительной трудоемкостью, возрастающей в квадратичной зависимости с увеличением длины строки *n*.

Кроме того, использование разработанных средств повышает эффективность проектирования за счет минимизации общего количества проектных ошибок и обеспечения бездефектности целевых специализированных устройств генерации комбинаторного беспорядка.

Таблица 1

Сложность логических схем устройств генерации комбинаторных перестановок и оценка временных затрат проектирования

	Длина символьной строки *n*		
	n = 16	*n* = 24	*n* = 32
Количество логических вентилей	3200	7400	13300
Количество строк VHDL-кода	1120	2180	3780
Временные затраты, в человеко-днях (станд. методика)	32	74	133
Временные затраты, в человеко-днях (предлож. методика)	1	1	1

Литература

[1] Курейчик В. М., Глушань В. М., Щербаков Л. И. Комбинаторные аппаратные модели и алгоритмы в САПР. – М.: Радио и связь. – 1990.

[2] U. S. Pat. No. 5,734,721 Bl H04L9/00 Anti-spoof without error extension (AN-SWER) / lark James Monroe. – March 31, 1998.

[3] Ritter T. Transposition Cipher with Pseudo-Random Shuffling: The Dynamic Transposition Combiner / Ritter // Cryptologia. – 1991. V. 15 (1). – P. 1-17.4.

[4] Молодченко Ж. А., Сотов Л. С., Харин В. Н. Динамическое форматирование представлений объектов реляционных СУБД на основе кластерных транспозиций // Естественные и технические науки. – М.: Изд-во компании "Спутник +". – 2007, № 6 (32). – С. 224-226.

[5] Немудров В., Мартин Г. Системы-на-кристалле. Проектирование. Проектирование и развитие. – Москва: Техносфера, 2004. – 216 с.

[6] IEEE Std 1666-2005 SystemC Language Reference Manual / The Insti-tute of Electrical and Electronics Engineers, Inc. 3 Park Avenue, New York, NY 10016-5997, USA. – 2008.

[7] IEEE Std 1076-2008 VHDL Language Reference Manual / The Institute of Electrical and Electronics Engineers, Inc. 3 Park Avenue, New York, NY 10016-5997, USA. – 2009.

[8] Бибило П.Н. Системы моделирования интегральных схем на основе языка VHDL. StateCAD, ModelSim, LeonardoSpectrum. – М.: СОЛОН Пресс, 2005. – 384 с.

[9] Соболев С. С., Харин В. Н., Сотов Л. С. Алгоритм локализации полных комбинаторных перестановок с применением циклического сдвига / Федеральное агентство по образованию, Государственное образовательное учреждение высшего профессионального образования, Воронежская государственная лесотехническая академия. – Воронеж, 2010. – 6 с. : ил. – Библиогр.: с. 6. – Деп. в ВИНИТИ РАН 02.08.2010 №480-В2010.

[10] Keating, M. Reuse Methodology Manual, Third Edition / Michael Keat-ing, Pierre Bricaud. – Kluwer Academic Publishers. – Boston/Dordrech/London. – 2002.