ISSN 1816-0301 (Print)
ISSN 2617-6963 (Online)

ЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ

LOGICAL DESIGN

УДК 681.32

Поступила в редакцию 06.11.2018
Received 06.11.2018

Принята к публикации 26.11.2018
Accepted 26.11.2018

Верификация систем с параллелизмом поведения на основе графа достижимых состояний

Ю. В. Поттосин✉, В. И. Романов, Л. Д. Черемисинова

Объединенный институт проблем информатики
Национальной академии наук Беларуси, Минск, Беларусь
✉*E-mail: pott@newman.bas-net.by*

Аннотация. Рассматривается задача верификации систем управления на основе моделей их поведения, которая состоит в проверке соответствия поведения системы требованиям, предъявляемым спецификацией на ее проектирование. Тестирование предполагает выполнение экспериментов, заключающихся в моделировании исследуемой системы, в ходе которого она проверяется на вход-выходное соответствие модели. Тестовая последовательность генерируется на основе модели, описывающей желаемое поведение системы. Предлагается метод построения тестовой последовательности для верификации схемной (или программной) реализации системы управления с параллелизмом поведения, который основан на обходе графа состояний, достижимых при функционировании системы. Описывается метод построения множества достижимых полных состояний для параллельного алгоритма описания поведения системы управления и получения тестовых наборов. Полагается, что описание функционирования системы, заданное спецификацией на проектирование, корректно; проверке подлежит схемная (или программная) реализация, которая должна соответствовать этой спецификации.

Ключевые слова: параллельный алгоритм, верификация, параллельный автомат, граф достижимых состояний, спецификация на проектирование

Благодарности. Работа выполнена при финансовой поддержке БРФФИ (проект Ф17АРМ-008).

Для цитирования. Поттосин, Ю. В. Верификация систем с параллелизмом поведения на основе графа достижимых состояний / Ю. В. Поттосин, В. И. Романов, Л. Д. Черемисинова // Информатика. – 2019. – Т. 16, № 2. – С. 62–72.

Verification of systems with behavior parallelism on the basis of the graph of reachable states

Yuri V. Pottosin✉, Vladimir I. Romanov, Ljudmila D. Cheremisinova

The United Institute of Informatics Problems of the National Academy
of Sciences of Belarus, Minsk, Belarus
✉*E-mail: pott@newman.bas-net.by*

Abstract. Considered problem of model based verification of control systems is the checking whether the system behavior satisfies the requirements fixed in the design specification The testing includes the experiments consisting in simulation of investigated system to see input-output correspondence to the model.

The test sequence is generated on the basis of the model that describes the desired behavior of the system. The method to construct a test sequence for verification of hardware (or software) implementation of a control system with behavior parallelism is suggested that is based on traversal of the graph of the states that are reachable in system functioning. A method for constructing the set of reachable global states for a parallel algorithm of the control system behavior and a method to obtain the test sets are described. The description of the system functioning, which is given by the design specification, is assumed to be correct. The hardware (or software) implementation that must conform to this specification is to be verified.

Key words: parallel algorithm, verification, parallel automaton, graph of reachable states, specification for design

Acknowledgements. This work was supported by the BRFFR (project F17APM-008).

For citation. Pottosin Yu. V., Romanov V. I., Cheremisinova L. D. Verification of systems with behavior parallelism on the basis of the graph of reachable states. *Informatics*, 2019, vol. 16, no 2, pp. 62–72.

Введение. Развитие микроэлектроники и средств автоматизации проектирования обеспечило возможность проектирования микроэлектронных управляющих систем значительной сложности. При проектировании и реализации таких систем невозможно избежать ошибок, число которых растет вместе с расходами на исправление и ликвидацию их последствий [1, 2]. Снижение надежности проектирования и реализации современных микроэлектронных систем обусловило тот факт, что неотъемлемой частью процесса проектирования стало тестирование, в частности проверка соответствия поведения устройств требованиям, предъявляемым спецификацией на их проектирование. В русскоязычной литературе эта задача называется верификацией устройства. В англоязычной литературе верификация конкретизируется как соответствие реализации объекта проектирования условиям ее спецификации, или тестирование на основе модели. Тестирование на основе моделей предполагает проверку свойств реального управляющего устройства, связанных с его функциональностью, путем выполнения ряда экспериментов. В ходе экспериментов, заключающихся в моделировании исследуемой системы, проверяется ее функциональность, т. е. правильно ли она реагирует на подаваемые стимулы. Тестовая последовательность генерируется на основе модели, описывающей желаемое поведение системы. Успешное прохождение тестов, сгенерированных надлежащим образом на основе модели, служит достаточной гарантией правильности реализации этой системы.

Ключевым моментом тестирования является заключение о том, корректна ли (в некотором смысле) реализация относительно данной спецификации. Тестирование является одним из наиболее важных и широко используемых методов проверки схемных реализаций и программного обеспечения, на него приходится до половины общих затрат на их разработку. Это мотивирует всевозрастающий интерес к данной задаче [2].

В настоящей работе рассматривается задача верификации систем управления на основе моделей их поведения, или задача проверки системы на вход-выходное соответствие модели. Наиболее разработанным направлением в этой области является верификация на основе моделей, представленных конечными автоматами [3, 4].

Наряду с традиционно организованными системами, которые реализуют чисто последовательное поведение, задаваемое на языках описания конечных автоматов, существует ряд систем, в которых выразительных средств аппарата конечных автоматов оказывается недостаточно. Наиболее важным свойством таких систем управления является присущий им параллелизм происходящих в них процессов. В качестве моделей этих цифровых систем на алгоритмическом уровне используются их представления на языках параллельных алгоритмов управления (в основе которых лежит аппарат сетей Петри) [5–7]. Данные языки позволяют задавать и исследовать параллелизм процессов при функционировании цифровых систем.

Рассматривается также задача построения тестовой последовательности для верификации схемной (или программной) реализации управляющего устройства с параллелизмом поведения. Тестовая последовательность формируется на основе графа состояний, достижимых при функционировании системы с параллелизмом поведения. Предлагается метод построения множества достижимых полных состояний согласно описанию спецификации на проектирование устройства и получения тестовых наборов. Предполагается, что описание функционирования устройства, заданного спецификацией на проектирование, корректно (не содержит ошибок). Проверке

подлежит схемная (или программная) реализация, которая должна соответствовать спецификации на области задания последней.

Язык задания спецификации на проектирование систем с параллелизмом поведения. Одной из важнейших проблем автоматизации производственных процессов в различных отраслях промышленности является проблема проектирования систем управления. При решении задач реализации систем управления приходится иметь дело с параллелизмом, присутствующим в объектах управления. Управление такими объектами заключается в обеспечении согласованной работы взаимодействующих компонентов, работающих параллельно и асинхронно. Параллелизм, присутствующий в объектах управления, отражается в функциональной модели цифровых систем, управляющих данными объектами. Для цифровых систем рассматриваемого класса устройств характерно также и то, что управляющие воздействия и сигналы о состоянии объектов управления описываются булевыми переменными, лишь небольшой процент всей информации является числовым. В настоящее время в качестве языка задания спецификации на проектирование управляющих систем используются сети взаимодействующих конечных автоматов и языки, базирующиеся на формальной модели сети Петри.

Для задания спецификации на проектирование систем с параллелизмом поведения предлагается использовать параллельные алгоритмы логического управления, которые широко применяются при проектировании и тестировании цифровых систем. Одним из языков спецификации систем является язык ПРАЛУ [8] описания простых алгоритмов логического управления. Верификация алгоритма управления на языке ПРАЛУ значительно упрощается при приведении его к стандартному виду, представляемому моделью, названной параллельным автоматом [8]. Алгоритмы в таком виде являются подклассом цветных сетей Петри [5, 6] – расширенных сетей свободного выбора [9]. На языке параллельных автоматов алгоритм представляет собой совокупность стандартных цепочек вида

$$\mu_i : - k_i^1 \to k_i^2 \to \nu_i, \tag{1}$$

где μ_i и ν_i – множества частичных состояний, в которых автомат находится перед и после срабатывания i-го перехода; k_i^1 и k_i^2 – элементарные конъюнкции булевых переменных. В отличие от классического конечного последовательного автомата параллельный автомат может одновременно находиться в нескольких состояниях, называемых частичными состояниями. Все множество частичных состояний, в которых рассматриваемый параллельный автомат находится в некоторый момент времени, составляет полное внутреннее состояние. Смысл приведенной цепочки заключается в следующем. Если автомат находится одновременно в состояниях, составляющих множество μ_i, и булевы переменные приняли значения, обращающие конъюнкцию k_i^1 в единицу, то конъюнкция k_i^2 приобретает значение единицы и автомат переходит из частичных состояний, составляющих множество μ_i, в частичные состояния, составляющие множество ν_i. Любая из операций $- k_i^1$ или $\to k_i^2$ (ожидания или действия) может отсутствовать. Отсутствие операции $- k_i^1$ означает тождественное равенство единице конъюнкции k_i^1. Отсутствие $\to k_i^2$ означает в зависимости от интерпретации данной модели либо то, что все выходные переменные обращаются в нуль, либо то, что значения сигналов на выходе не меняются. Любое описание алгоритма на языке ПРАЛУ легко преобразуется в параллельный автомат [8].

Параллельный автомат задается:

– множеством $S = \{1, 2,\ldots\}$частичных внутренних состояний, входящих в μ_i и ν_i;

– входным и выходным алфавитами X и Y, которые состоят из входных и выходных булевых переменных, входящих соответственно в конъюнкции k_i^1 и k_i^2;

– переходами $\tau_i = (\mu_i \to \nu_i) / (k_i^1 \to k_i^2)$, соответствующими цепочкам (1).

Переход τi автомата срабатывает, когда текущая маркировка N_t включает все состояния из μ_i и переменные принимают значения, обращающие k_i^1 в единицу. После срабатывания перехода переменным из k_i^2 присваиваются значения, обращающие k_i^2 в единицу, а маркировка N_t заменяется на $(N_t \setminus \mu_i) \cup \nu_i$.

В качестве примера приведем описание параллельного автомата, заданного следующим множеством обобщенных переходов:

$\tau_1 = (1 \rightarrow 10) / (x_1\, x_2 \rightarrow y_1\ \bar{y}_2)$,
$\tau_2 = (10 \rightarrow 2.3.4) /(\ \bar{x}_2)$,
$\tau_3 = (2 \rightarrow 5.6) /(\rightarrow \bar{y}_1)$,
$\tau_4 = (3.5 \rightarrow 8) /(x_2)$,
$\tau_5 = (4 \rightarrow 7) /(x_1 \rightarrow \bar{y}_1)$,
$\tau_6 = (4 \rightarrow 9) /(x_1 \rightarrow y_1)$,
$\tau_7 = (7 \rightarrow 9) /(x_2)$,
$\tau_8 = (6.8.9 \rightarrow 11) /(\rightarrow \bar{y}_2)$,
$\tau_9 = (11 \rightarrow 1) /(x_1)$.

Постановка задачи проверки соответствия и метод ее решения. Задача проверки соответствия между спецификацией и ее схемной реализацией существенно отличается от проблем проверки эквивалентности пары схемных реализаций путем их функционального тестирования, верификации и валидации путем установления корректности проектируемой системы (соответствия поведения объектов взаимодействия ряду свойств). Отличие заключается в том, что проверяется не корректность спецификации (предполагается, что описание, отраженное в спецификации, корректно), а соответствие функциональности схемной реализации (или обнаружение неисправностей) функциональности, заданной спецификацией на проектирование. Основным средством тестирования схемной реализации на соответствие спецификации является моделирование, для проведения которого предварительно строится тестовая (проверяющая) последовательность, или просто тест. Под тестом понимается упорядоченная последовательность наборов значений входных сигналов и соответствующая последовательность изменений значений выходных сигналов, которые должны происходить после подачи этих наборов. Тестирование осуществляется на уровне вход-выходных последовательностей путем выполнения экспериментов над реальным устройством или его моделью. При моделировании тестовые наборы подаются на входные полюсы моделируемой системы, определяются изменения значений сигналов, происходящие на ее выходах, которые сравниваются с эталонными значениями.

На рис. 1 показаны форматы приведенного описания параллельного автомата в программном комплексе LOCON (LogicalControl) [10].

```
TITLE pott1
FORMAT PRL
AUTHOR Bibilo
DATE 28.04.2017
PROJECT LOCON_VHDL
DCL_PIN
EXT
INP
x1 x2
OUT
y1 y2
INTER
END_PIN
BLOCK pott1main
1: -^x1*x2 > y1*^y2 > 10;
10: -^x2 > 2.3.4;
2: > ^y1 > 5.6;
3.5: -x2 > 8;
4: -^x1 > ^y1 > 7;
   -x1 > y2 > 9;
7: -^x2 > 9;
6.8.9: >^y2 > 11;
11: -x1 > 1;
END_BLOCK pott1main
END_pott1
```

а)

```
TITLE pott1
FORMAT  PA
AUTHOR Bibilo
DATE 08.11.2017
PROJECT LOCON_VHDL
DCL_PIN
EXT
INP
x1 x2
OUT
y1 y2
INTER
END_PIN
BLOCK pott1main
1>10: ^x1*x2>y1*^y2;
10>2.3.4: ^x2>;
2>5.6: >^y1;
3.5>8: x2>;
4>7: ^x1>^y1;
4>9: x1>y2;
7>9: ^x2>;
6.8.9>11: >^y2;
11>1: x1>;
END_BLOCK pott1main
END_pott1
```

б)

Рис. 1. Форматы представления алгоритмов логического управления:
а) ПРАЛУ-алгоритм; *б)* параллельный автомат

Рассматриваемая задача проверки соответствия (рис. 2) включает этапы:
– генерации проверяющей последовательности на основе заданной спецификации;
– подачи тестовых наборов на входы устройства;
– наблюдения поведения устройства;
– определения, соответствует ли реализация исходной спецификации.

Рис. 2. Проверка соответствия между спецификацией и ее схемной реализацией

По способу проверки соответствия между реализацией и моделью, заданной спецификацией, можно выделить следующие варианты тестирования схемы:

1. *Проверку соответствия внутренних состояний схемы и модели*. В этом случае предполагается, что состояния, в которых находится схема, наблюдаемы. Проверяющая последовательность строится по модели и представляет собой последовательность троек (S_i^b, X_i, S_i^e), задающих S_i^b и S_i^e – начальное и конечное множества состояний, X_i – входной стимул.

2. *Проверку соответствия выходных откликов схемы и модели*. Проверяющая последовательность строится в виде пар (X_i, Y_i), где Y_i – изменение значений переменных при подаче на вход схемы входного набора X_i. Предполагается, что схема не допускает наблюдаемость состояний, но возможен перевод ее в начальное состояние, начиная с которого проводится эксперимент, путем сброса значений триггеров. В противном случае, если сброс невозможен, выполняется процедура инициализации схемы предварительной подачей на ее входы установочной последовательности.

3. *Проверку соответствия состояний и выходных откликов схемы и модели*. Проверяющая последовательность строится в виде четверок (S_i^b, X_i, S_i^e,Y_i), что обеспечивает наиболее полное тестирование схемы на соответствие спецификации.

Задача синтеза проверяющего теста заключается в построении конечного множества воздействий на систему, по реакции на которые можно определить правильность ее функционирования. Можно выделить два основных способа генерации входных воздействий в проверяющей последовательности:

– генерацию псевдослучайных наборов значений входных переменных. Минусы такого способа заключаются в том, что число возможных входных воздействий в каждом состоянии схемы (и ее модели) экспоненциально зависит от числа переменных и, главное, не все такие наборы попадают в область определения модели (а значит, и реализации);

– генерацию наборов значений входных переменных и проверяющей последовательности в целом исходя из описания модели. Этот случай не только обеспечивает наиболее полное тестирование схемы на области, определяемой спецификацией на ее проектирование, но и позволяет сократить длину проверяющей последовательности.

При тестировании параллельных управляющих систем будет рассматриваться задача генерации проверяющей последовательности в виде последовательности четверок (S_i^b, X_i, S_i^e, Y_i). Каждая четверка может порождаться переходами исходного алгоритма.

Основным инструментом, лежащим в основе методов анализа поведенческих свойств управляющей системы с параллелизмом поведения, служит граф достижимых состояний параллельного алгоритма управления, являющегося спецификацией на проектирование этой системы. Вершинам ориентированного графа достижимых состояний соответствуют полные состояния параллельного автомата, задаваемые разметками N_t, а дугам – переходы между полными состояниями. Дуга помечается символом τ_i перехода автомата и соединяет разметки N_t, такие, что сеть переходит от первой разметки ко второй при срабатывании перехода τ_i.

Граф достижимых состояний представляет собой ориентированный мультиграф, в котором могут быть петли и кратные дуги, различающиеся присвоенными им метками. Кроме того, очевидно, что каждый переход алгоритма управления может повторяться много раз в качестве метки дуг графа достижимости. Граф достижимости рассматриваемого выше параллельного алгоритма имеет 12 вершин (рис. 3).

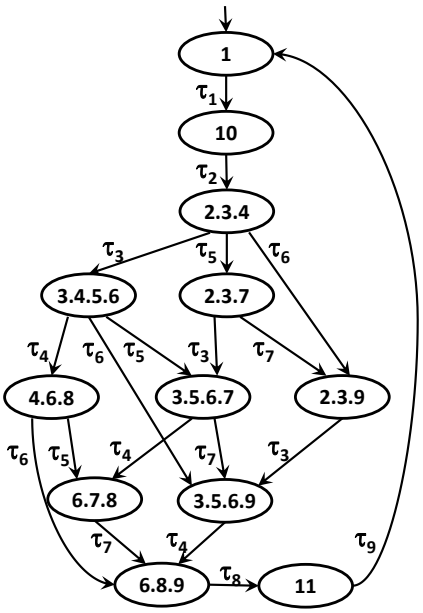


Рис. 3. Граф достижимых состояний

Построение множества полных состояний параллельного автомата. Для построения графа достижимости состояний предварительно получается множество всех полных состояний автомата. Рассматривается «скелет» параллельного автомата, подобный α-сети [8], который задается множеством частичных состояний $S = \{s_1, s_2, \ldots, s_n\}$ и множеством переходов $T = \{\tau_1, \tau_2, \ldots, \tau_k\}$, где $\tau_i = (S_i, F_i)$ и $S_i, F_i \subseteq S$, $i = 1, 2, \ldots, k$. Задается также начальное полное состояние $P_1 \subseteq S$. Переход τ_i происходит следующим образом: если $S_i \subseteq P_g$, где P_g – текущее полное состояние автомата, то полным состоянием в следующий момент времени будет $P_h = (P_g \setminus S_i) \cup F_i$.

Перед запуском алгоритма построения графа достижимости выполняется последовательное получение полных состояний $P = \{P_1, P_2, \ldots, P_l\}$, начиная с заданного начального состояния P_1. Далее происходит просмотр всех заданных переходов $\tau_i = (S_i, F_i)$ и, если $S_i \subseteq P_1$, по формуле $P_j = (P_1 \setminus S_i) \cup F_i$ формируется новое множество P_j, которое объявляется достижимым полным состоянием. Этот процесс повторяется для каждого из вновь внесенных в P множеств P_j. Процесс заканчивается, когда не получается новых множеств такого вида, отличных от уже полученных.

Алгоритм получения множества P полных состояний автомата:

1) $|P| := i := j := 0$, $P := P \cup \{S_1\}$;
2) $i := i + 1$; если $i \le |P|$, $j := 0$, перейти к п. 3,
   иначе перейти к п. 5;
3) $j := j + 1$, если $j \le k$, перейти к п. 4,
   иначе перейти к п. 2;
4) если $S_j \subseteq P_i$, $P := P \cup \{(P_i \setminus S_j) \cup F_j\}$, перейти к п. 3,
   иначе перейти к п. 3;
5) конец.

Результатом работы алгоритма для параллельного автомата *pott1* (см. рис. 1) является следующее множество, состоящее из 12 полных состояний: {{1}, {10}, {2, 3, 4}, {3, 4, 5, 6}, {2, 3, 7}, {2, 3, 9}, {4, 6, 8}, {3, 5, 6, 7}, {6, 8, 7}, {3, 5, 6, 9}, {6, 8, 9}, {11}}.

Построение графа достижимых состояний параллельного автомата. Вершинами ориентированного графа достижимых состояний являются полные состояния в виде множеств $P_1, P_2, \ldots, P_t$, которые получаются в результате выполнения описанного выше алгоритма. Из вершины P_g исходит дуга, заходящая в вершину P_h, если в исходном задании имеется переход $\tau_i = (S_i, F_i)$, такой, что $P_h = (P_g \setminus S_i) \cup F_i$). Метод, лежащий в основе данного алгоритма,

заключается в поиске пар вида (P_g, P_h), которым соответствует указанный переход $\tau_i = (S_i, F_i)$. Каждой такой паре соответствует дуга искомого графа. Представлением получаемого графа является перечень дуг и приписанных им меток. Это представление удобно для описания алгоритма. Оно является также довольно компактным, поскольку графы данного вида обладают сравнительно небольшим числом дуг.

Исходными данными для алгоритма служат следующие объекты:

– множество частичных состояний $S = \{s_1, s_2, \ldots, s_n\}$;

– совокупность троек (S_1, F_1, n_1), (S_2, F_2, n_2), …, (S_s, F_s, n_s), задающих переходы $\tau_i = (S_i, F_i)$, где $n_i = i$ – номер перехода автомата;

– множество полных состояний $P = \{P_1, P_2, \ldots, P_t\}$.

В результате строится ориентированный граф, заданный перечислением дуг: $X = \{x_1, x_2, \ldots, x_r\}$ – номера начал дуг, $Y = \{y_1, y_2, \ldots, y_r\}$ – номера концов дуг, $N = \{n_1, n_2, \ldots, n_r\}$ – номера переходов, являющихся метками дуг.

В работе алгоритма используются множества U, V и W для представления промежуточных результатов.

Алгоритм построения графа достижимых состояний:

1) $X := Y := \varnothing$, $i := 0$;

2) $i := i + 1$; если $i \le t$, $j := 0$, перейти к п. 3,
иначе перейти к п. 7;

3) $j := j + 1$; если $j = i$, перейти к п. 3,
иначе, если $j \le t$, перейти к п. 4,
иначе перейти к п. 2;

4) $W := P_i \cap P_j$, $U := P_i \setminus W$, $V := P_j \setminus W$, $l := 0$, перейти к п. 5;

5) $l := l + 1$; если $l \le s$, перейти к п. 6,
иначе перейти к п. 3;

6) если $U = S_l$ и $V = F_l$, $X := X \cup \{i\}$, $Y := Y \cup \{j\}$, $N := N \cup \{l\}$, перейти к п. 3,
иначе перейти к п. 3;

7) конец.

При применении алгоритма к автомату *pott1* (см. рис. 1) получается граф достижимости (см. рис. 3), имеющий следующее множество дуг: ({1}, {10}, τ_1), ({10}, {2,3,4}, τ_2), ({2,3,4}, {3,4,5,6}, τ_3), ({2,3,4}, {2,3,7}, τ_5), ({2,3,4}, {2,3,9}, τ_6), ({3,4,5,6}, {4,6,8}, τ_4), ({3,4,5,6}, {3,5,6,7}, τ_5), ({3,4,5,6}, {3,5,6,9}, τ_6), ({2,3,7}, {3,5,6,7}, τ_3), ({2,3,7}, {2,3,9}, τ_7), ({4,6,8}, {6,7,8}, τ_5), ({4,6,8}, {6,8,9}, τ_6), ({3,5,6,7}, {6,7,8}, τ_4), ({3,5,6,7}, {3,5,6,9}, τ_7), ({2,3,9}, {3,5,6,9}, τ_3), ({6,7,8}, {6,8,9}, τ_7), ({3,5,6,9}, {6,8,9}, τ_4), ({6,8,9}, {11}, τ_8), ({11}, {1}, τ_9).

В общем случае граф достижимости является ориентированным мультиграфом, так как он может содержать кратные дуги, различающиеся присвоенными им метками.

Предложенные алгоритмы построения графа достижимых состояний параллельного автомата были программно реализованы на языке C++ в среде кроссплатформенного программирования Qt на основе использования разработанных инструментальных средств логического проектирования [11], включающих классы булевых матриц и векторов для представления множеств, а также класс представления параллельных автоматов, которые были модернизированы после переноса в среду Qt.

Построение тестовой последовательности для функциональной верификации. Задача синтеза проверяющего теста заключается в построении конечного множества воздействий на систему, по реакциям на которые можно определить правильность ее функционирования. Разработка алгоритмов синтеза тестов требует использования математических моделей, позволяющих отобразить поведение системы. В связи с практической значимостью и теоретическим интересом наиболее изученной областью верификации на основе моделей является тестирование конечных автоматов. Первые работы в этой области появились около 50 лет назад. Тестовая последовательность формируется путем объединения нескольких подпоследовательностей, и, как правило, подпоследовательности имеют перекрытия. В литературе встречается достаточное число работ, в которых предлагаются эвристики для сокращения числа перекрытий с целью

уменьшения общей длины тестовой последовательности. Тестовая последовательность строится следующим образом:

– заранее, до начала процесса тестирования, в виде единого маршрута по графу, проходящего через все дуги графа;

– заранее в виде множества маршрутов, начинающихся в начальной вершине (состоянии) и покрывающих в совокупности все дуги графа;

– в процессе моделирования системы, в этом случае тестовые наборы генерируются динамически при обходе графа в зависимости от откликов схемы на поданные наборы.

Очевидно, что построение тестовой последовательности в виде единого маршрута возможно, если граф достижимых состояний является сильно связным, т. е. если из каждой его вершины достижима любая другая вершина. Если граф достижимости не является сильно связным, то возможно построение тестовой последовательности в виде множества маршрутов из начальной вершины графа. Если граф не является сильно связным, но все его вершины достижимы из начальной, то при моделировании управляющей системы требуется ее рестарт (сброс триггеров блока памяти) при переходе от одного теста к другому. После рестарта производится выбор следующей тестовой последовательности.

Задача построения тестовой последовательности на основе графа достижимых состояний заключается в построении такого кратчайшего ориентированного маршрута (чередующейся последовательности вершин и дуг) на орграфе $G = (V, E)$, который проходит через каждую дугу графа по крайней мере один раз (в общем случае не один раз). В этой постановке задача построения кратчайшего ориентированного маршрута аналогична задаче китайского почтальона [12], в которой ищется кратчайший путь, проходящий через все дуги заданного орграфа. В силу трудоемкости решения такой задачи нахождение точного решения (с минимумом числа возможных повторных прохождений дуг графа достижимости) в общем случае не представляется возможным.

Для решения задачи обхода графа достижимых состояний можно использовать один из известных методов обхода дуг ориентированного графа, разработанных для случая детерминированных автоматных моделей [3, 4, 13–16]. По последовательности переходов алгоритма управления, соответствующих меткам найденной последовательности дуг ориентированного графа, можно определить входные стимулы тестовой последовательности для подачи на входы тестируемой реализации системы управления. Следует заметить, что таким образом для каждого достижимого состояния системы определяются только те стимулы, которые заданы в реализуемой спецификации, т. е. тестирование проводится только на заданной спецификацией области определения. Если в реализации допустимы какие-то другие стимулы, то это никак не влияет на процесс тестирования. Фактически это означает, что для заданной реализации исходного модельного алгоритма управления тестируется некоторая часть ее функциональности, заданная переходами по входным стимулам и состояниям модельного алгоритма, достижимым из начального состояния.

Обход дуг графа достижимости, изображенного на рис. 3, задается следующим циклическим маршрутом, начинающимся и заканчивающимся в начальном состоянии 1:

1, τ_1, 10, τ_2, 2.3.4, τ_3, 3.4.5.6, τ_4, 4.6.8, τ_6, 6.8.9, τ_8, 11, τ_9,
1, τ_1, 10, τ_2, 2.3.4, τ_3, 3.4.5.6, τ_4, 4.6.8, τ_5, 6.7.8, τ_7, 6.8.9, τ_8, 11, τ_9,
1, τ_1, 10, τ_2, 2.3.4, τ_3, 3.4.5.6, τ_6, 3.5.6.9, τ_4, 6.8.9, τ_8, 11, τ_9,
1, τ_1, 10, τ_2, 2.3.4, τ_3, 3.4.5.6, τ_5, 3.5.6.7, τ_4,6.7.8, τ_7, 6.8.9, τ_8, 11, τ_9,
1, τ_1, 10, τ_2, 2.3.4, τ_3, 3.4.5.6, τ_5, 3.5.6.7, τ_7, 3.5.6.9, τ_4, 6.8.9, τ_8, 11, τ_9,
1, τ_1, 10, τ_2, 2.3.4, τ_5, 2.3.7, τ_3, 3.5.6.7, τ_4, 6.7.8, τ_7, 6.8.9, τ_8, 11, τ_9,
1, τ_1, 10, τ_2, 2.3.4, τ_5, 2.3.7, τ_3, 3.5.6.7, τ_7, 3.5.6.9, τ_4, 6.8.9, τ_8, 11, τ_9,
1, τ_1, 10, τ_2, 2.3.4, τ_5, 2.3.7, τ_7, 2.3.9, τ_3, 3.5.6.9, τ_4, 6.8.9, τ_8, 11, τ_9,
1, τ_1, 10, τ_2, 2.3.4, τ_6, 2.3.9, τ_3, 3.5.6.9, τ_4, 6.8.9, τ_8, 11, τ_9, 1.

При проверке соответствия между состояниями и выходными откликами схемной реализации и ее моделью, задаваемой параллельным алгоритмом управления, тестовая последовательность строится в виде четверок (S_i^b, X_i, S_i^e, Y_i), где S_i^b и S_i^e – начальное и конечное множества состояний, X_i – входной стимул, Y_i – изменение значений переменных при подаче на вход схемы

входного набора X_i. Это обеспечивает наиболее полное тестирование схемы на соответствие спецификации. При отображении тестовой последовательности начальное состояние S_i^b (кроме S_1^b) будем опускать, считая его равным S^e_{i-1}:

$(1, x_1 x_2, 10, y_1 \bar{y}_2)$, $(\bar{x}_2, 2.3.4, –)$, $(–, 3.4.5.6, \bar{y}_1)$, $(x_2, 4.6.8, –)$, $(x_1, 6.8.9, y_1)$, $(–,11, \bar{y}_2)$, $(x_1, 1, –)$,

$(x_1 x_2, 10, y_1 \bar{y}_2)$, $(\bar{x}_2, 2.3.4, –)$, $(–, 3.4.5.6, \bar{y}_1)$, $(x_2, 4.6.8, –)$, $(x_1, 6.7.8, \bar{y}_1)$, $(x_2, 6.8.9, y_1)$, $(–, 11, \bar{y}_2)$, $(x_1, 1, –)$,

$(x_1 x_2, 10, y_1 \bar{y}_2)$, $(\bar{x}_2, 2.3.4, –)$, $(–, 3.4.5.6, \bar{y}_1)$, $(x_1, 3.5.6.9, y_1)$, $(x_2, 6.8.9, –)$, $(–, 11, \bar{y}_2)$, $(x_1, 1, –)$,

$(x_1 x_2, 10, y_1 \bar{y}_2)$, $(\bar{x}_2, 2.3.4, –)$, $(–, 3.4.5.6, \bar{y}_1)$, $(x_1, 3.5.6.7, \bar{y}_1)$, $(x_2, 6.7.8, –)$, $(x_2, 6.8.9, y_1)$, $(–, 11, \bar{y}_2)$, $(x_1, 1, –)$,

$(x_1 x_2, 10, y_1 \bar{y}_2)$, $(\bar{x}_2, 2.3.4, –)$, $(–, 3.4.5.6, \bar{y}_1)$, $(x_1, 3.5.6.7, \bar{y}_1)$, $(x_2, 3.5.6.9, –)$, $(x_2, 6.8.9, –)$, $(–, 11, \bar{y}_2)$, $(x_1, 1, –)$,

$(x_1 x_2, 10, y_1 \bar{y}_2)$, $(\bar{x}_2, 2.3.4, –)$, $(x_1, 2.3.7, \bar{y}_1)$, $(–, 3.5.6.7, \bar{y}_1)$, $(x_2, 6.7.8, –)$, $(x_2, 6.8.9, y_1)$, $(–, 11, \bar{y}_2)$, $(x_1, 1, –)$,

$(x_1 x_2, 10, y_1 \bar{y}_2)$, $(\bar{x}_2, 2.3.4, –)$, $(x_1, 2.3.7, \bar{y}_1)$, $(–, 3.5.6.7, \bar{y}_1)$, $(x_2, 3.5.6.9, –)$, $(x_2, 6.8.9, –)$, $(–, 11, \bar{y}_2)$, $(x_1, 1, –)$,

$(x_1 x_2, 10, y_1 \bar{y}_2)$, $(\bar{x}_2, 2.3.4, –)$, $(x_1, 2.3.7, \bar{y}_1)$, $(x_2, 2.3.9, –)$, $(–, 3.5.6.9, \bar{y}_1–)$, $(x_2, 6.8.9, –)$, $(–, 11, \bar{y}_2)$, $(x_1, 1, –)$,

$(x_1 x_2, 10, y_1 \bar{y}_2)$, $(\bar{x}_2, 2.3.4, –)$, $(x_1, 2.3.9, y_1)$, $(–, 3.5.6.9, \bar{y}_1)$, $(x_2, 6.8.9, –)$, $(–, 11, \bar{y}_2)$, $(x_1, 1, –)$.

Если граф достижимости не является сильно связным, но каждое его состояние достижимо из начального и тестируемая схемная реализация допускает рестарт из начального состояния, то можно строить сразу не граф, а усеченное дерево достижимости (рис. 4). Его построение из графа достижимости производится путем обрывания путей из начальной вершины при обнаружении достигнутых ранее разметок. В таком случае искомая тестовая последовательность ищется в виде множества цепей, начинающихся в начальной вершине (состоянии) и имеющих концы в листовых вершинах дерева достижимости. При этом цепи в совокупности должны покрывать все дуги дерева.

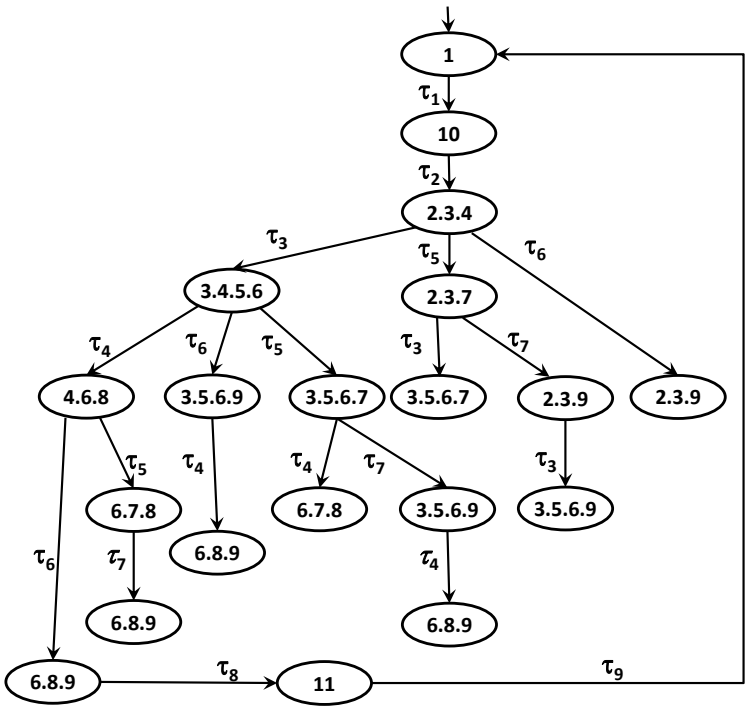


Рис. 4. Дерево достижимых состояний

Получается восемь тестовых последовательностей:

(1, $x_1 x_2$, 10, y_1 $\bar{y}_2$), ($\bar{x}_2$, 2.3.4, –), (–, 3.4.5.6, $\bar{y}_1$), (x_2, 4.6.8, –), (x_1, 6.8.9, y_1), (–, 11, $\bar{y}_2$), (x_1, 1, –);

(1, $x_1 x_2$, 10, y_1 y_2), ($\bar{x}_2$, 2.3.4, –), (–, 3.4.5.6, $\bar{y}_1$), (x_2, 4.6.8, –), (x_1 , 6.7.8, $\bar{y}_1$), (x_2, 6.8.9, y_1);

(1, $x_1 x_2$, 10, y_1 $\bar{y}_2$), ($\bar{x}_2$, 2.3.4, –), (–, 3.4.5.6, $\bar{y}_1$), (x_1, 3.5.6.9,y_1), (x_2, 6.8.9, –);

(1, $x_1 x_2$, 10, y_1 $\bar{y}_2$), ($\bar{x}_2$, 2.3.4, –) ,(–, 3.4.5.6, $\bar{y}_1$), (x_1, 3.5.6.7, $\bar{y}_1$), (x_2, 6.7.8,–);

(1, $x_1 x_2$, 10, y_1 y_2), ($\bar{x}_2$, 2.3.4, –), (–, 3.4.5.6, $\bar{y}_1$), (x_1, 3.5.6.7, $\bar{y}_1$), (x_2, 3.5.6.9,–), (x_2, 6.8.9, –);

(1, $x_1 x_2$, 10, y_1 $\bar{y}_2$), ($\bar{x}_2$, 2.3.4, –), (x_1, 2.3.7, $\bar{y}_1$), (–, 3.5.6.7, $\bar{y}_1$);

(1, $x_1 x_2$, 10, y_1 $\bar{y}_2$), ($\bar{x}_2$, 2.3.4, –), (x_1, 2.3.7, $\bar{y}_1$), (x_2, 2.3.9,–), (–, 3.5.6.9, $\bar{y}_1$–);

(1, $x_1 x_2$, 10, y_1 $\bar{y}_2$), ($\bar{x}_2$, 2.3.4, –),(x_1, 2.3.9, y_1).

Заключение. В работе рассматривается задача анализа системы управления на входвыходное соответствие модели для наиболее сложного и недостаточно изученного случая, когда исходная спецификация описывает систему с параллелизмом поведения. Предложенный метод основан на построении графа достижимости параллельного алгоритма управления и поиска обхода дуг этого графа.

Список использованных источников

1. Валидация на системном уровне. Высокоуровневое моделирование и управление тестированием : пер. с англ. Е. Б. Махияновой / М. Чэнь [и др.]. – М. : Техносфера, 2014. – 296 с.

2. Tretmans, J. Model based testing with labelled transition systems / J. Tretmans // Formal Methods and Testing: Lecture Notes in Computer Science. – Springer, 2008. – Vol. 4949. – P. 1–38.

3. Lee, D. Principles and methods of testing finite state machine – a survey / D. Lee, M. Yannakakis // Proceedings of the IEEE. – 1996. – Vol. 84, no. 8. – P. 1090–1123.

4. Верификация автоматных программ / С. Э. Вельдер [и др.]. – СПб. : Наука, 2011. – 244 с.

5. Питерсон, Дж. Теория сетей Петри и моделирование систем : пер. с англ. М. В. Горбатовой, В. Л. Торхова, В. Н. Четверикова / Дж. Питерсон. – М. : Мир, 1984. – 264 с.

6. Котов, В. Е. Сети Петри / В. Е. Котов. – М. : Наука, 1984. – 160 с.

7. Karatkevich, A. Dynamic Analysis of Petri Net-based Discrete Systems / A. Karatkevich. – Berlin : Springer-Verlag, 2007. – Vol. 358. – 166 p.

8. Закревский, А. Д. Параллельные алгоритмы логического управления / А. Д. Закревский. – Минск : Ин-т техн. кибернетики НАН Беларуси, 1999. – 202 с.

9. Hack, M. Analysis of production schemata by Petri nets / M. Hack // Project MAK-94. – Cambridge, 1972. – 119 p.

10. Experimental system of automated design of logical control devices / A. D. Zakrevskij [et al.] // Proc. of the Intern. Workshop "Discrete Optimization Methods in Scheduling and Computer-Aided Design". – Минск : Ин-т техн. кибернетики НАН Беларуси, 2000. – С. 216–221.

11. Романов, В. И. Разработка инструментальных средств логического проектирования / В. И. Романов // Логическое проектирование. – Минск : Ин-т техн. кибернетики НАН Беларуси, 2001. – Вып. 6. – С.151–170.

12. Thimbleby, H. The directed Chinese Postman Problem / H. Thimbleby // Software Practice and Experience. – 2003. – Vol. 33, no. 11. – P. 1081–1096.

13. Бурдонов, И. Б. Неизбыточные алгоритмы обхода ориентированных графов. Детерминированный случай / И. Б. Бурдонов, А. С. Косачев, В. В. Кулямин // Программирование. – 2003. – № 5. – С. 11–30.

14. Черемисинова, Л. Д. Построение тестов полного перебора для оценки энергопотребления последовательностных схем / Л. Д. Черемисинова // Информатика. – 2017. – № 4. – С. 104–110.

15. Kanso, B. Compositional testing for FSM-based models / B. Kanso, O. Chebaro // Intern. J. of Software Engineering & Applications (IJSEA). – 2014. – Vol. 5, no. 3. – P. 1–20.

16. Витязь, К. А. Алгоритмы построения функциональных тестов для цифровой схемы на основе автоматной модели ее поведения / К. А. Витязь, В. И. Романов // Танаевские чтения : доклады Восьмой Междунар. науч. конф., Минск, 27–30 марта 2018 г. – Минск : ОИПИ НАН Беларуси, 2018. – С. 52–56.

References

1. Chen M., Qin X., Koo H.-M., Mishra P. *System-Level Validation High-Level Modeling and Directed Test Generation Techniques*. New York, Springer-Verlag, 2013, 250 p.

2. Tretmans J. Model based testing with labelled transition systems. *Formal Methods and Testing: Lecture Notes in Computer Science*. Springer, 2008, vol. 4949, pp. 1–38.

3. Lee D., Yannakakis M. Principles and methods of testing finite state machine – a survey. *Proceedings of the IEEE*, 1996, vol. 84, no. 8, pp. 1090–1123.

4. Vel'der S. E., Lukin M. A., Shalyto A. A., Jaminov B. R. Verifikacija avtomatnyh program. *Verification of Automation Programs*. Saint Petersburg, Nauka, 2011, 244 p. (in Russian).

5. Peterson J. *Petri Net Theory and the Modeling of Systems*. New York, Prentice Hall, 1981, 290 p.

6. Kotov V. E. Seti Petri. *Petri Nets*. Moscow, Nauka, 1984, 160 p. (in Russian).

7. Karatkevich A. *Dynamic Analysis of Petri Net-based Discrete Systems*. Berlin, Springer-Verlag, 2007, vol. 358, 166 p.

8. Zakrevskij A. D. Parallel'nye algoritmy logicheskogo upravlenija. *Parallel Algorithms of Logical Control*. Minsk, Institut tehnicheskoj kibernetiki Nacional'noj akademii nauk Belarusi, 1999, 202 p. (in Russian).

9. Hack M. Analysis of production schemata by Petri nets. *Project MAK-94*, Cambridge, 1972,119 p.

10. Zakrevskij A. D., Pottosin Yu. V., Vasilkova I. V., Romanov V. I. Experimental system of automated design of logical control devices. *Proceedings of the International Workshop "Discrete Optimization Methods in Scheduling and Computer-Aided Design"*. Minsk, Institut tehnicheskoj kibernetiki Nacional'noj akademii nauk Belarusi, 2000, pp. 216–221.

11. Romanov V. I. Razrabotka instrumental'nyh sredstv logicheskogo proektirovayija [Development of Instruments for Logical Design]. Logicheskoe proektirovanie.Vyp. 6 [*Logical Design*. *Issue 6*]. Minsk, Institut tehnicheskoj kibernetiki Nacional'noj akademii nauk Belarusi, 2001, pp. 151–170 (in Russian).

12. Thimbleby H. The directed Chinese Postman Problem. *Software Practice and Experience*, 2003, vol. 33, no. 11, pp. 1081–1096.

13. Burdonov I. B, Kosachev A. S., Kuljamin V. V. Neizbytochnye algoritmy obhoda orientirovannyh grafov. Determinirovannyj sluchaj [Irredandant algorithms for traversal of directed graphs. The determinate case]. Programmirovanie [*Programming*], 2003, no. 5, pp. 11–30 (in Russian).

14. Cheremisinova L. D. Postroenie testov polnogo perebora dlja ocenki energopotreblenija posledovatel'nostnyh shem [Constructing tests of exhaustive search for estimation of power consumption of sequential circuits]. Informatika [*Informatics*], 2017, no. 4, pp. 104–110 (in Russian).

15. Kanso B., Chebaro O. Compositional testing for FSM-based models. *International Journal of Software Engineering & Applications (IJSEA)*, 2014, vol. 5, no 3, pp. 1–20.

16. Vitjaz' K. A., Romanov V. I. Algoritmy postroenija funkcional'nyh testov dlja cifrovoj shemy na osnove avtomatnoj modeli ejo povedenija [Algorithms for constructing functional tests for a digital circuit on the base of automaton model of its behavior]. Tanaevskie chtenija: doklady Vos'moj Mezhdunarodnoj nauchnoj konferencii [*Tanaev Lecturings: Proceedings of the Eighth International Scientific Conference*, *Minsk, 27–30 March 2018*], Minsk, Ob"edinennyj institut problem informatiki Nacional'noj akademii nauk Belarusi, 2018, pp. 52–56 (in Russian).

Информация об авторах

Поттосин Юрий Васильевич, кандидат физико-математических наук, ведущий научный сотрудник, Объединенный институт проблем информатики Национальной академии наук Беларуси, Минск, Беларусь.
E-mail: pott@newman.bas-net.by

Романов Владимир Ильич, кандидат технических наук, ведущий научный сотрудник, Объединенный институт проблем информатики Национальной академии наук Беларуси, Минск, Беларусь.
E-mail: rom@newman.bas-net.by

Черемисинова Людмила Дмитриевна, доктор технических наук, главный научный сотрудник, Объединенный институт проблем информатики Национальной академии наук Беларуси, Минск, Беларусь.
E-mail: cld@newman.bas-net.by

Information about the authors

Yuri V. Pottosin, Cand. Sci. (Phys.-Math.), Leading Researcher, The United Institute of Informatics Problems of the National Academy of Sciences of Belarus, Minsk, Belarus.
E-mail: pott@newman.bas-net.by

Vladimir I. Romanov, Cand. Sci. (Eng.), Leading Researcher, The United Institute of Informatics Problems of the National Academy of Sciences of Belarus, Minsk, Belarus.
E-mail: rom@newman.bas-net.by

Ljudmila D. Cheremisinova, Dr. Sci. (Eng.), Chief Researcher, The United Institute ofInformatics Problems of the National Academy of Sciences of Belarus, Minsk, Belarus.
E-mail:cld@newman.bas-net.by

ISSN 1816-0301 (Print)
ISSN 2617-6963 (Online)
УДК 519.714; 519.7

Поступила в редакцию 27.11.2018
Received 27.11.2018

Принята к публикации 08.01.2019
Accepted 08.01.2019

Логическая минимизация булевых сетей с использованием разложения Шеннона

П. Н. Бибило✉, Ю. Ю. Ланкевич

Объединенный институт проблем информатики
Национальной академии наук Беларуси, Минск, Беларусь
✉*E-mail: bibilo@newman.bas-net.by*

Аннотация. Синтез логических схем, реализующих комбинационные блоки сверхбольших интегральных схем, – одна из важнейших задач компьютерного проектирования, так как размерность задач проектирования увеличивается, возрастает также время выполнения этапов синтеза логических схем. Особенно трудоемкой является глобальная технологическая независимая оптимизация – первый этап синтеза логической схемы. Суть второго этапа заключается в технологическом отображении оптимизированных логических представлений функций на логические элементы технологической библиотеки. Основные характеристики логической схемы, такие как площадь, быстродействие, энергопотребление, зависят от эффективности первого этапа. Эволюция методов глобальной логической оптимизации показала эффективность разложения Шеннона при оптимизации многоуровневых представлений систем полностью определенных булевых функций. Разработано множество методов и программ, использующих графические представления разложений Шеннона – BDD-представления. Большинство разработанных методов оптимизации BDD-представлений используют задания исходных систем булевых функций в виде дизъюнктивных нормальных форм (ДНФ).

Предлагается алгоритм минимизации числа вершин булевой сети, являющейся многоуровневым представлением системы полностью определенных булевых функций. Минимизация осуществляется на основе разложения Шеннона и поиска вершин сети, реализующих одинаковые и взаимно инверсные функции. Предложенный алгоритм логической оптимизации реализован в виде программы. Эксперименты показали, что данный алгоритм и полученную программу целесообразно использовать в случае, когда исходное многоуровневое представление функций невозможно представить (за приемлемое время работы компьютерной программы) в виде системы ДНФ либо когда система ДНФ, полученная из многоуровневого представления, содержит большое число (десятки и сотни тысяч) элементарных конъюнкций.

Ключевые слова: булева функция, булева сеть, разложение Шеннона, дизъюнктивная нормальная форма, синтез логических схем, BDD

Благодарности. Исследование выполнено при финансовой поддержке БРФФИ в рамках проекта № Ф19-023.

Для цитирования. Бибило, П. Н. Логическая минимизация булевых сетей с использованием разложения Шеннона / П. Н. Бибило, Ю. Ю. Ланкевич // Информатика. – 2019. – Т. 16, № 2. – С. 73–89.

Logical optimization of Boolean nets using Shannon expansion

Petr N. Bibilo✉, Yury Y. Lankevich

The United Institute of Informatics Problems of the National Academy
of Sciences of Belarus, Minsk, Belarus
✉*E-mail: bibilo@newman.bas-net.by*

Abstract. A synthesis of logical circuits, comprising functional combination blocks of very large scale integration circuits, is one of the most important tasks of computer-aided design. As the data size of design tasks increases, the execution time of synthesis of logic circuits also increases. The global technological independent

optimization as the first stage of synthesis of logical circuit is especially labor-consuming. The second stage is technological mapping of optimized logical representations of functions to the logical elements of technological library. The main features of logical circuit, such as area, performance, power consumption, depend on the efficiency of the first stage – global logical optimization. The evolution of methods of global logical optimization has revealed the efficiency of Shannon expansion in case of optimization of multi-level representations of the systems of fully defined Boolean function. A number of methods and programs were developed using graphical representations of Shannon expansions – BDD representations. Most of the developed methods of optimization of BDD-representations use the initial representations of functions systems in the form of disjunctive normal form (DNF).

In the article an algorithm of minimization of nodes number of Boolean net, which is a multi-level representation of the system of fully defined Boolean function, is proposed. Minimization is based on Shannon expansion and a search of equal (with accuracy up to inversion) nodes in Boolean net. Such algorithm of logical optimization was implemented as application. The experiments have shown that this algorithm and the application is reasonable to use in cases when the initial multi-level representation of functions is impossible to define as DNF system, or when DNF system contains a large number of elementary conjunctions.

Keywords: Boolean function, Boolean net, Shannon expansion, disjunctive normal form, synthesis of logical circuits, BDD

Acknowledgements. The study was carried out with the financial support of the BRFFR in the framework of project No. F19-023.

For citation. Bibilo P. N., Lankevich Y. Y. Logical optimization of Boolean nets using Shannon expansion. *Informatics*, 2019, vol. 16, no. 2, pp. 73–89 (in Russian).

Введение. Синтез логических схем, реализующих функциональные комбинационные блоки цифровых заказных сверхбольших интегральных схем (СБИС), по-прежнему остается одной из важных задач автоматизированного проектирования, так как возрастает размерность задач проектирования и, соответственно, растет время выполнения этапов синтеза. Особенно трудоемкой является глобальная технологически независимая оптимизация, являющаяся первым этапом синтеза схемы [1, 2]. Суть второго этапа синтеза – технологического отображения (technology mapping) – заключается в «покрытии» оптимизированных логических представлений функций библиотечными логическими элементами. Основные характеристики логической схемы, такие как площадь (часто выражаемая в числе транзисторов), быстродействие и энергопотребление, зависят во многом от эффективности выполнения первого этапа – глобальной логической оптимизации. На втором этапе при покрытии оптимизированных логических уравнений функциональными описаниями библиотечных логических элементов выполняется локальная оптимизация с учетом особенностей логических элементов соответствующей библиотеки. Элементы библиотеки могут реализовать симметричные либо несимметричные булевы функции, покрываемые фрагменты могут иметь несущественные (фиктивные) переменные и т. д.

Как указано в фундаментальном обзоре [3], в первых системах автоматизированного проектирования основными методами технологически независимой оптимизации были методы минимизации (совместной или раздельной с учетом инверсирования) в классе ДНФ [4], после чего осуществлялась алгебраическая факторизация [5, 6].

Алгебраическая факторизация – выделение общих частей алгебраических представлений функций – осуществлялась на уровне представлений функций в виде булевых сетей. Булевы сети (графы) используют «мелкозернистые» функциональные описания вершин (узлов), узлы при покрытии объединяются в кластеры, а каждый кластер реализуется при этом библиотечным логическим элементом. «Мелкие» логические выражения удобны для формирования кластеров, поэтому они и нашли применение в программах синтеза [7, 8].

Развитие методов глобальной логической оптимизации показало эффективность разложения Шеннона при оптимизации многоуровневых представлений. Было разработано много методов [9–12] и программ, использующих графические представления разложений Шеннона булевых функций – так называемые BDD-представления (Binary Decision Diagram – диаграмма двоичного выбора). В русскоязычной литературе BDD называют также диаграммами двоичных решений, бинарными диаграммами решений, двоичными решающими диаграммами и т. д. Наибольшее развитие получили методы и программы оптимизации с помощью BDD для исходных систем функций, заданных в виде ДНФ и представленных парой матриц (троичная мат-

рица задает элементарные конъюнкции, булева матрица – вхождения элементарных конъюнкций в ДНФ функций системы). Однако получение систем ДНФ в современных синтезаторах не предусматривается. Получение исходных для логической оптимизации представлений в современных синтезаторах логических схем осуществляется после выполнения высокоуровневого синтеза – локальной замены алгоритмических конструкций языков VHDL и Verilog логическими выражениями (формулами), задающими многоуровневые представления систем булевых функций. Например, получение булевых сетей в синтезаторе LeonardoSpectrum [13] осуществляется командой unmap, в результате выполнения которой синтезированная комбинационная схема представляется, по сути, в виде булевой сети. Получаемые многоуровневые представления близки к булевым сетям: кроме операций инверсии (отрицания) и логических двухвходовых операций дизъюнкции и конъюнкции в таких формулах имеется еще операция «исключающее ИЛИ», часто называемая «суммой по модулю два». Такое представление функций может быть оптимизировано с помощью методов глобальной оптимизации, что позволяет достаточно часто при повторном синтезе улучшать результаты начального синтеза логической схемы [12].

В настоящей работе предлагается алгоритм минимизации числа вершин в булевой сети, являющейся многоуровневым представлением системы полностью определенных булевых функций, которая получается после этапа высокоуровневого синтеза либо при повторном синтезе схемы. Минимизация осуществляется на основе разложения Шеннона и поиска вершин сети, реализующих одинаковые булевы выражения (функции), а также поиска вершин, реализующих взаимно инверсные булевы выражения. Такой алгоритм глобальной технологически независимой логической оптимизации программно реализован и экспериментально исследован. Эксперименты показали, что его целесообразно применять в тех случаях, когда исходное многоуровневое представление функций невозможно представить (за приемлемое время (часы) работы компьютерной программы) в виде системы ДНФ либо когда система ДНФ, полученная из многоуровневого представления, содержит многие десятки и сотни тысяч элементарных конъюнкций.

Представления систем булевых функций в виде булевых сетей. Булева сеть – это ориентированный ациклический граф [3]. Вершины, обладающие нулевой полустепенью исхода, помечаются как выходы сети, а вершины, обладающие нулевой полустепенью захода, – как входы. Каждой вершине графа соответствует некоторая булева переменная. Каждую вершину, не являющуюся входом сети, представляет булева функция. Вершина i называется входной вершиной для вершины j, если в сети есть дуга, ведущая из i в j. Переменные, соответствующие выходам сети, называются выходными; переменные, относящиеся к входам сети, – входными, а переменные, которые соответствуют остальным вершинам сети, – промежуточными. В статье рассматриваются булевы сети, представляющими функциями вершин которых могут быть двухоперандные логические операции И (*, конъюнкция), ИЛИ (+, $\vee$, дизъюнкция), а также однооперандная операция НЕ (^, инверсия). Чтобы построить булеву сеть по выражению, задающему булеву функцию, необходимо для каждого знака логической операции этого выражения построить вершину сети и поставить ей в соответствие данную операцию и некоторую промежуточную переменную. Далее необходимо добавить входные вершины и правильным образом соединить вершины дугами. На рис. 1 изображена булева сеть, построенная по формуле $f = \psi_0 \vee \psi_1 = \bar{x}_i \varphi_0 \vee x_i \varphi_1$.

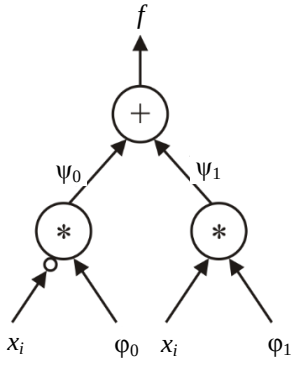


Рис. 1. Булева сеть из трех узлов

Сложность многоуровневого представления системы функций в булевом базисе И, ИЛИ, НЕ будем оценивать суммарным числом двухвходовых логических операторов в соответствующей булевой сети. Инверсии переменных при подсчете сложности представления не принимаются во внимание. Такая оценка сложности хорошо согласуется с известной в литературе [3] оценкой сложности алгебраических представлений булевых функций по общему числу литералов булевых переменных.

Рассмотрим пример булевой сети (рис. 2), имеющей четыре входные переменные *x*0, *x*1, *x*2, *x*3 и две выходные переменные *y*1, *y*2, функции семи узлов которой задаются формулами

$$\begin{gathered} y1 = x0 + \mathrm{tmp\_0}, \\ \mathrm{tmp\_0} = x1 + \mathrm{tmp\_2}, \\ \mathrm{tmp\_2} = \wedge x2 * x3, \\ y2 = x0 + \mathrm{tmp\_3}, \\ \mathrm{tmp\_3} = \mathrm{tmp\_4} + \mathrm{tmp\_5}, \\ \mathrm{tmp\_4} = \wedge x1 * x2, \\ \mathrm{tmp\_5} = \wedge x1 * \wedge x3. \end{gathered} \tag{1}$$

Сложность булевой сети на рис. 2 равна семи: три узла реализуют операцию * конъюнкции, четыре узла – операцию + дизъюнкции.

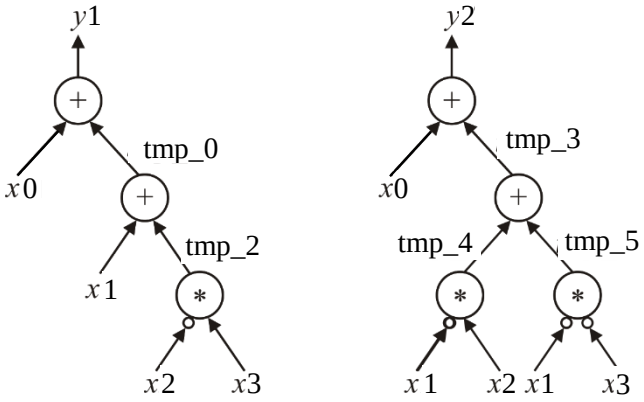


Рис. 2. Булева сеть из семи узлов для формул (1)

Функциональные разложения, используемые при оптимизации многоуровневых представлений систем булевых функций, чаще всего базируются на разложении Шеннона. Формула разложения Шеннона для одной булевой функции $f(x) = f(x_1, \ldots, x_{i-1}, x_i, x_{i+1}, \ldots, x_n)$ имеет вид

$$f(x) = \bar{x}_i f(x_1, ..., x_{i-1}, 0, x_{i+1}, ..., x_n) \vee x_i f(x_1, ..., x_{i-1}, 1, x_{i+1}, ..., x_n). \tag{2}$$

Разложение Давио использует коэффициенты (подфункции) $f(x_1, ..., x_{i-1}, 0, x_{i+1}, ..., x_n)$, $f(x_1, ..., x_{i-1}, 1, x_{i+1}, ..., x_n)$, на его базе строятся функциональные диаграммы решений (Functional Decision Diagram, FDD).

Положительное разложение Давио имеет вид

$$\begin{gathered} f(x) = f(x_1, ..., x_{i-1}, 0, x_{i+1}, ..., x_n) \oplus x_i (f(x_1, ..., x_{i-1}, 0, x_{i+1}, ..., x_n) \oplus \\ \oplus f(x_1, ..., x_{i-1}, 1, x_{i+1}, ..., x_n)), \end{gathered} \tag{3}$$

отрицательное разложение Давио –

$$\begin{gathered} f(x) = f(x_1, ..., x_{i-1}, 1, x_{i+1}, ..., x_n) \oplus \bar{x}_i (f(x_1, ..., x_{i-1}, 0, x_{i+1}, ..., x_n) \oplus \\ \oplus f(x_1, ..., x_{i-1}, 1, x_{i+1}, ..., x_n)). \end{gathered} \tag{4}$$

Кронекеровские функциональные диаграммы решений (Kronecker FDD, KFDD) используют как разложение Шеннона, так и разложение Давио, однако на каждом шаге разложения, т. е. при разложении по очередной переменной, используется только один из видов разложений (2)–(4).

Краткий обзор функциональных разложений приведен в работе [14], обобщения для неполностью определенных (частичных) булевых функций предложены в работе [15].

Одним из новых функциональных разложений по двум переменным является разложение вида

$$f(x) = f(x_1, ..., x_i, ..., x_j, ..., x_n) = \\ = (x_i \oplus x_j) f(x_1, ..., \bar{x}_j, ..., x_j, ..., x_n) \vee (x_i \sim x_j) f(x_1, ..., x_j, ..., x_j, ..., x_n), \quad (5)$$

подробно изученное в работе [14] и названное biconditional expansion. В выражении (5) через $\oplus$ обозначена логическая операция «исключающее ИЛИ», через $\sim$ – логическая операция «эквивалентность».

Постановка задачи. Задано многоуровневое представление системы $f(x) = (f^1(\boldsymbol{x}), \ldots, f^m(\boldsymbol{x}))$, $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$, полностью определенных булевых функций в виде взаимосвязанных логических формул. Каждая из формул имеет вид ДНФ. Требуется минимизировать сложность булевой сети, представляющей систему функций $f(x) = (f^1(\boldsymbol{x}), \ldots, f^m(\boldsymbol{x}))$.

В настоящей работе предлагается проводить минимизацию булевых сетей на основе разложения Шеннона с поиском одинаковых (и взаимно инверсных) подвыражений, реализуемых узлами булевой сети.

Запишем разложение Шеннона (2) в виде $f = \bar{x}_i\varphi_0 \vee x_i\varphi_1 = \psi_0 \vee \psi_1$. Формула (2) представляется логической сетью с тремя узлами (вершинами) (см. рис. 1), выходная вершина $f = \psi_0 \vee \psi_1$ реализует дизъюнкцию, две входные ψ_0, ψ_1 вершины сети – конъюнкции: $\psi_0 = \bar{x}_i\varphi_0$, $\psi_1 = x_i\varphi_1$, т. е. каждая формула разложения Шеннона заменяется своей «малой» булевой сетью, состоящей из трех вершин. В классических алгоритмах минимизации многоуровневых представлений на основе разложения Шеннона (BDD-оптимизации) находятся одинаковые коэффициенты (подфункции) φ_0, φ_1 в разложениях различных функций системы. В работе [16] введено понятие BDDI и находятся одинаковые и взаимно инверсные коэффициенты. Под BDDI (Binary Decision Diagram with Inverse cofactors – диаграмма двоичных решений с инверсными коэффициентами) понимается ориентированный ациклический граф, задающий последовательные разложения Шеннона булевой функции $f(x_1, \ldots, x_n)$ по всем ее переменным $x_1, \ldots, x_n$ при заданном порядке (последовательности, перестановке) переменных, по которым проводятся разложения. Граф BDDI одной полностью определенной булевой функции содержит функциональные вершины, соответствующие разлагаемым функциям и подфункциям (и их инверсиям), вершины-переменные и листовые вершины, соответствующие константам 0, 1. Функциональная вершина BDDI реализует одну функцию φ (подфункцию) либо две функции (подфункцию φ и ее инверсию $\bar{\varphi}$). Эксперименты [16] показали, что нахождение инверсных коэффициентов позволяет получать более компактные формулы и менее сложные схемы при последующем синтезе схем из библиотечных КМОП-элементов, образующих библиотеку проектирования отечественных заказных СБИС.

В отличие от минимизации BDD- и BDDI-представлений, основанных на поиске одинаковых (и взаимно инверсных) коэффициентов φ_0, φ_1, в булевых сетях после выполнения очередного разложения Шеннона находятся одинаковые и взаимно инверсные булевы выражения (функции узлов булевой сети). Такие выражения определяются на основе законов булевой алгебры [17], примеры взаимно инверсных булевых выражений приведены в табл. 1.

Таблица 1
Взаимно инверсные булевы выражения – функции узлов булевой сети

Выражение (функция узла)	Инверсное выражение
$\psi_0 + \psi_1$	$\overline{\psi}_0 * \overline{\psi}_1$
$\overline{x}_i * \varphi_0$	$x_i + \overline{\varphi}_0$
$x_i * \varphi_1$	$\overline{x}_i + \overline{\varphi}_1$

Алгоритм минимизации булевой сети. Алгоритм включает три этапа:

Этап 1. Строится булева сеть по исходному заданию системы булевых функций, зависящих от n переменных. На данном этапе инверсные подвыражения не находятся. Замена многооперандных операций дизъюнкции и конъюнкции осуществляется каскадными формулами (бинарными деревьями) из соответствующих двухвходовых операций. Например, формула многооперандной дизъюнкции $D = k1 + k2 + k3 + k4 + k5 + k6 + k7$ заменяется совокупностью формул:

$$D = t1 + t2 \text{ (уровень 3)};$$
$$t1 = k1 + t3,\ t2 = t4 + t5 \text{ (уровень 2)};$$
$$t3 = k2 + k3,\ t4 = k4 + k5,\ t5 = k6 + k7 \text{ (уровень 1)}.$$

Аналогично можно записать формулы булевой подсети для многооперандной конъюнкции.

Этап 2. Находится первая перестановка входных переменных булевой сети, по которой строится первое оптимизированное представление системы функций в виде булевой сети.

Алгоритм второго этапа является локально-оптимальным и состоит из следующих шагов:

Шаг 1 (итеративный). Выбирается первая переменная, по которой будет проведено разложение Шеннона.

Для каждой из входных переменных x_i множества $X = \{x_1, x_2, \ldots, x_n\}$ булевой сети проводится построение разложения Шеннона по переменной x_i, полученные булевы подсети редуцируются по законам булевой алгебры. Первая остаточная подсеть (остаточные уравнения) получаются при подстановке $x_i = 0$, вторая подсеть – при подстановке $x_i = 1$. Редукция каждой из остаточных подсетей производится с учетом нахождения одинаковых и взаимно инверсных подвыражений, являющихся функциями вершин остаточных подсетей. После этого подсети объединяются – добавляются вершины, соответствующие формулам разложения Шеннона для каждой из выходных переменных, затем булева сеть повторно упрощается за счет нахождения одинаковых и взаимно инверсных узлов. Для полученной булевой сети проводится еще одно дополнительное сокращение путем нахождения вырожденных формул разложения Шеннона. Так, формула разложения Шеннона $f = \overline{x_i}\varphi_0 \vee x_i\varphi_1$, когда $\varphi_1 = 1$, заменяется более простой формулой $f = \varphi_0 \vee x_i$, так как $\overline{x}_i\varphi_0 \vee x_i = \varphi_0 \vee x_i$. Аналогично, если формула разложения Шеннона для какой-то из функций имеет вид $\varphi_0 = 1$, то она заменяется формулой $f = \overline{x}_i \vee \varphi_1$. После этого дополнительного сокращения производится оценка сложности булевой сети, построенной по переменной x_i разложения Шеннона.

После оценки разложений Шеннона по всем переменным множества $X = \{x_1, x_2, \ldots, x_n\}$ выбирается переменная x_{j_1}, по которой редуцированная булева сеть будет иметь наименьшую сложность. Эта переменная будет первой переменной в искомой перестановке $< x_{j_1}, x_{j_2}, \ldots, x_{j_n} >$ переменных для построения многоуровневого представления.

При программной реализации были испытаны три эвристики оценки сложности булевой сети.

Шаг 2 (итеративный). Выбираются остальные $n - 1$ переменные в искомой перестановке переменных для разложения Шеннона.

По выбранной на шаге 1 переменной x_{j_1} строится разложение Шеннона для булевой сети, полученная сеть будет исходной для выбора следующей переменной x_{j_2} разложения, которая выбирается аналогичным образом из множества оставшихся переменных, т. е. тех переменных, по которым еще не проведено разложение Шеннона функций, реализуемых булевой сетью.

Этап 3 (итеративный). Перебираются перестановки $< x_{j_1}, x_{j_2}, ..., x_{j_n} >$ для уменьшения сложности булевой сети.

Перебор является итеративным и осуществляется с помощью алгоритма второго этапа. Исходным представлением будет булева сеть, полученная в результате выполнения второго этапа. После нахождения очередной перестановки она запоминается, как и сложность соответствующей булевой сети. Итерации (перебор перестановок) прекращаются, если сгенерирована такая перестановка, которая была уже рассмотрена ранее, либо время вычислений превысило заданное.

Пример работы алгоритма на сети, заданной уравнениями (1).

Этап 1. Каждое из уравнений может быть представлено одной вершиной булевой сети (см. рис. 2).

Этап 2. *Шаг 1.* Нахождение первой перестановки переменных, по которым строятся разложения Шеннона.

Итерация 1. Оценка сложности остаточной сети, полученной при разложении Шеннона по переменной *x*0.

Для большей ясности построения разложений Шеннона перепишем функции выходных переменных в виде

$$y1 = x0 + tmp\_0,$$
$$y2 = x0 + tmp\_3$$

и получим

$$y1 = \hat{}x0 * tn\_0 + x0 * tn\_2,$$
$$y2 = \hat{}x0 * tn\_1 + x0 * tn\_3.$$

Подставим в формулы (1) значение *x*0 = 0, при этом для каждой вершины сети (см. рис. 2) сформируем инверсное выражение. Получим остаточные уравнения, приведенные в табл. 2.

Сравним остаточные уравнения (в том числе инверсные) на равенство. Одинаковых уравнений нет.

Таблица 2

Результаты подстановки *x*0 = 0 в формулы (1)

Остаточные уравнения	Инверсии остаточных уравнений
tn_0 = tmp_0	^tn_0 = ^tmp_0
tmp_0 = *x*1 + tmp_2	^tmp_0 = ^*x*1 * ^tmp_2
tmp_2 = ^*x*2 * *x*3	^tmp_2 = *x*2 + ^*x*3
tn_1 = tmp_3	^tn_1 = ^tmp_3
tmp_3 = tmp_4 * tmp_5	^tmp_3 = ^tmp_4 + ^tmp_5
tmp_4 = ^*x*1 * *x*2	^tmp_4 = *x*1 + ^*x*2
tmp_5 = ^*x*1 * ^*x*3	^tmp_5 = *x*1 + *x*3

Подставим в формулы (1) значение *x*0 = 1, при этом для каждой неконстантной вершины сети сформируем инверсное выражение.

Остаточными уравнениями при *x*0 = 1 будут уравнения tn_2 = 1 и tn_3 = 1. Сравним остаточные уравнения (в том числе инверсные) на равенство. Одинаковых уравнений нет.

Объединим подсети, полученные на шагах 2, 4, и получим остаточную булеву сеть для оценки по переменной *x*0:

$$\begin{gathered}
\text{tmp_0} = x1 + \text{tmp_2}, \\
\text{tmp_2} = \hat{}x2 * x3, \\
\text{tmp_3} = \text{tmp_4}*\text{tmp_5}, \\
\text{tmp_4} = \hat{}x1 * x2, \\
\text{tmp_5} = \hat{}x1 * \hat{}x, \\
y1 = \hat{}x0 * \text{tmp_0} + x0, \\
y2 = \hat{}x0 * \text{tmp_3} + x0.
\end{gathered} \tag{6}$$

При записи уравнений в $y1 = \hat{}x0 * \text{tn\_0} + x0 * \text{tn\_2}$ переменную tn_0 заменили на tmp_0, tn_2 = 1, в уравнении $y2 = \hat{}x0 * \text{tn\_1} + x0 * \text{tn\_3}$ переменную tn_1 заменили на tmp_3, tn_3 = 1.

Сложность остаточной сети, полученной разложением Шеннона по переменной $x0$, равна 5.

Итерация 2. Оценка сложности остаточной сети, полученной при разложении Шеннона по переменной $x1$.

Подставим в формулы (1) значение $x1 = 0$, при этом для каждой вершины сети (см. рис. 2) сформируем инверсное выражение (табл. 3).

Таблица 3

Результаты подстановки $x1 = 0$ в формулы (1)

Остаточные уравнения	Инверсии остаточных уравнений
$\text{tn\_6} = x0 + \text{tmp\_2}$	$\hat{}\text{tn\_6} = \hat{}x0 * \hat{}\text{tmp\_2}$
$\text{tmp\_2} = \hat{}x2 * x3$	$\hat{}\text{tmp\_2} = x2 + \hat{}x3$
$\text{tn\_7} = x0 + \text{tn\_5}$	$\hat{}\text{tn\_7} = \hat{}x0 * \hat{}\text{tn\_5}$
$\text{tn\_5} = x2 + \hat{}x3$	$\hat{}\text{tn\_5} = \hat{}x2 * x3$

Сравним остаточные уравнения (в том числе инверсные) на равенство. Найдено одно совпадение:

$$\text{tn\_5} = \hat{}\text{tmp\_2}.$$

Заменим tn_5 на ^tmp_2 и получим следующие уравнения:

$$\begin{gathered}
\text{tn_6} = x0 + \text{tmp_2}\ (\hat{}\text{tn_6} = \hat{}x0 * \hat{}\text{tmp_2}), \\
\text{tn_7} = x0 + \hat{}\text{tmp_2}\ (\hat{}\text{tn_7} = \hat{}x0 * \text{tmp_2}), \\
\text{tmp_2} = \hat{}x2 * x3\ (\hat{}\text{tmp_2} = x2 + \hat{}x3).
\end{gathered}$$

Подставив в формулы (6) значение $x1 = 1$, получим $\text{tn\_8} = 1$ и $\text{tn\_9} = x0$. Сравним остаточные уравнения (в том числе инверсные) на равенство. Одинаковых уравнений нет. Объединим подсети, тогда остаточная булева сеть для оценки по переменной $x1$ имеет вид

$$\begin{gathered}
\text{tn_6} = x0 + \text{tmp_2}, \\
\text{tn_7} = x0 + \hat{}\text{tmp_2}, \\
\text{tmp_2} = \hat{}x2 * x3, \\
\text{tn_9} = x0.
\end{gathered} \tag{7}$$

В результате получим выражения

$$\begin{gathered}
y1 = \text{tn_6} + x1, \\
y2 = \hat{}x1 * \text{tn_7} + x1 * x0.
\end{gathered}$$

В уравнении $y1 = \hat{}x1 * \text{tn\_6} + x1 * \text{tn\_8}$ переменная $\text{tn\_8} = 1$, в уравнении $y1 = \hat{}x1 * \text{tn\_7} + x1 * \text{tn\_9}$ промежуточная переменная $\text{tn\_9} = x0$. Остаточная сеть по переменной $x1$ имеет сложность 3. Итерация 3 (для переменной $x2$) и итерация 4 (для переменной $x3$) первого этапа выполняются аналогично.

Шаг 2 (итеративный). Поиск первой перестановки переменных разложения Шеннона.

Остаточная сеть по переменной разложения $x1$ имеет наименьшую сложность 3, поэтому переменная $x1$ является первой переменной в искомой перестановке $< x_{j_1}, x_{j_2}, ..., x_{j_n} >$, а формулы (7) будут исходными для выбора следующей переменной для проведения следующего разложения Шеннона. Оценив сложности остаточных уравнений для переменных $x0$, $x2$, $x3$, получим, что следующей переменной разложения является переменная $x0$ (один узел в остаточной сети). Для оставшихся двух переменных $x2$, $x3$ остаточная сеть имеет вид tmp_2 = ^$x2$ * $x3$ со сложностью 1.

Результатом выполнения шага 2 (и этапа 2) является перестановка <$x1$, $x0$, $x2$, $x3$> переменных, по которым проведены разложения Шеннона. Полученной перестановке соответствует совокупность формул

$$\begin{gathered} y1 = \text{tn\_6} + x1, \\ y2 = \text{\^{}}x1 * \text{tn\_7} + x1 * x0, \\ \text{tn\_6} = \text{tmp\_2} + x0, \\ \text{tn\_7} = \text{\^{}tmp\_2} + x0, \\ \text{tmp\_2} = \text{\^{}}x2 * x3. \end{gathered} \tag{8}$$

Сложность соответствующей булевой сети равна 7.

Этап 3. Перебор перестановок для уменьшения сложности булевой сети.

Исходной является булева сеть, реализующая формулы (8). Применив к ней шаги этапа 2 алгоритма, получим новую перестановку <$x0$, $x1$, $x2$, $x3$> и соответствующую булеву сеть, реализующую формулы (6):

$$\begin{gathered} y1 = \text{\^{}tn\_57} + x0, \\ y2 = \text{tn\_57} + x0, \\ \text{tn\_57} = \text{\^{}tn\_88} * \text{\^{}}x1, \\ \text{tn\_88} = \text{\^{}}x2 * x3. \end{gathered} \tag{9}$$

Применив к формулам (9) шаги этапа 2 алгоритма, получим перестановку <$x0$, $x1$, $x2$, $x3$>, которая была найдена ранее. Поэтому формулы (9) являются итогом работы алгоритма минимизации для исходной сети, заданной уравнениями (1). В результате получено многоуровневое представление (рис. 3) с двумя конъюнкциями и двумя дизъюнкциями (сложность 4), при этом изначально сеть состояла из трех конъюнкций и четырех дизъюнкций (сложность 7).

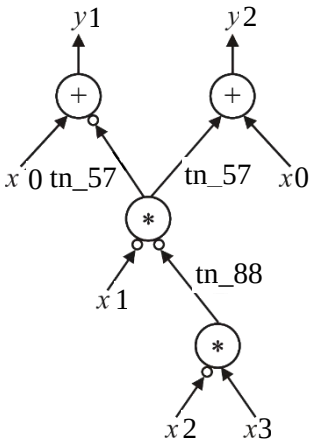


Рис. 3. Булева сеть из четырех узлов для формул (9)

Заметим, что представление булевых функций *y*1, *y*2 в виде BDD

$$\begin{gathered} y1 = \hat{}x0 * sf1 + x0, \\ y2 = \hat{}x0 * sf3 + x0, \\ sf1 = \hat{}x1 * sf5 + x1, \\ sf3 = \hat{}x1 * sf6, \\ sf5 = \hat{}x2 * x3, \\ sf6 = \hat{}x2 * \hat{}x3 + x2 \end{gathered} \tag{10}$$

имеет сложность 10, а представление в виде BDDI

$$\begin{gathered} y2 = \hat{}x0 * \hat{}sf0 + x0, \\ y1 = \hat{}x0 * sf0 + x0, \\ sf0 = \hat{}x1 * sf2 + x1, \\ sf2 = \hat{}x2 * x3 \end{gathered} \tag{11}$$

имеет сложность 7.

Экспериментальные исследования. Предложенный в настоящей работе алгоритм был программно реализован. В разработанной программе BDD_Builder2 исследованы три эвристики:

1. Для проведения разложения Шеннона выбирается переменная x_i, для которой сложность остаточной сети является наименьшей. Сложность подсети, задающей формулы разложений Шеннона, не учитывается.

2. Для проведения разложения Шеннона выбирается переменная x_i, для которой суммарная сложность остаточной сети и подсети, реализующей формулы разложения Шеннона, является наименьшей.

3. Для проведения разложения Шеннона выбирается переменная x_i, для которой суммарная сложность остаточной сети и подсети, реализующей формулы разложения Шеннона, является наименьшей, при этом учитывается только одна десятая сложности подсети для формул разложения Шеннона. Данная эвристика выражает компромисс между эвристиками 1 и 2, при котором сложность остаточной сети имеет бóльший приоритет.

Цель экспериментов состояла в том, чтобы сравнить результаты синтеза по неоптимизированным и оптимизированным функциональным описаниям многовыходных комбинационных схем. Исходные описания представлялись на языке SF в системе FLC логической оптимизации [18], затем выполнялась программа BDD_Builder2. Полученные оптимизированные функциональные описания конвертировались в VHDL-описания и подавались на вход синтезатора LeonardoSpectrum [13], который выполнял построение логических схем. Все схемы строились в одной и той же библиотеке *s3lib* КМОП-элементов (табл. 4). Сравнивались результаты синтеза по площади схем и их быстродействию. Каждое выполнение программы BDD_Builder2 сопровождалось формальной верификацией исходного и полученного оптимизированного описаний.

Первый поток из девяти примеров для проведения экспериментов (add6, b2, Intb, root, sist_4, tial, tms, z5xp1, z9sym) включал функциональные описания в виде взаимосвязанных логических уравнений, которые являются формулами, полученными командой unmap в синтезаторе LeonardoSpectrum [13] после выполнения синтеза. Особенностью таких формул является то, что каждая из них содержит только одну логическую операцию над парой булевых переменных либо операцию инверсирования одной булевой переменной. Заметим, что исходный синтез для данных примеров был проведен в расширенной библиотеке, приведенной в работе [18], а повторный синтез осуществлялся в библиотеке *s3lib* элементов (табл. 4). Исходные описания примеров систем ДНФ находятся в библиотеке примеров схем [19].

Таблица 4

Библиотека *s3lib* логических КМОП-элементов проектирования заказных СБИС

Элемент	Функция	Число транзисторов
GND	$y = 0$	1
VCC	$y = 1$	1
N	$y = \overline{A}$, инвертор	2
NX2	$y = \overline{A}$, двухкратный инвертор	4
NX4	$y = \overline{A}$, четырехкратный инвертор	8
NA	$y = \overline{AB}$	4
NO	$y = \overline{A \vee B}$	4
NAO	$y = \overline{(A \vee B)C}$	6
NOA	$y = \overline{(AB) \vee C}$	6
NA3O	$y = \overline{(A \vee B)CD}$	8
NO3A	$y = \overline{(AB) \vee C \vee D}$	8
NA3	$y = \overline{ABC}$	6
NO3	$y = \overline{A \vee B \vee C}$	6

Второй поток – это восемь широко известных примеров (Apex6, C8, Cht, Count, Dalu, Frg2, Too_large, X3) многоуровневых описаний комбинационной логики. Для этих описаний осуществлялся синтез схем без выполнения программ предварительной оптимизации и с помощью предварительной оптимизации булевых сетей, предложенной в настоящей работе.

Эксперимент 1. Сравнивались эффективности применения трех эвристик оптимизации булевых сетей без использования упрощений

$$f = \overline{x}_i\varphi_0 \vee x_i = \varphi_0 \vee x_i, \quad f = \overline{x}_i \vee x_i\varphi_1 = \overline{x}_i \vee \varphi_1. \tag{12}$$

Эксперимент 2. Сравнивались эффективности применения трех эвристик оптимизации булевых сетей с использованием упрощений (12).

Эксперимент 3. Сравнивались предложенный алгоритм и его программная реализация (три эвристики) с известными в литературе алгоритмами (программами) оптимизации многоуровневых представлений, которые предназначаются для использования в качестве предварительного этапа при синтезе логических схем. Целью таких алгоритмов оптимизации является сокращение сложности функциональных описаний, что благоприятно сказывается на дальнейшей минимизации площади схем при последующем их синтезе.

Для сравнения были выбраны:

– алгоритм, основанный на оптимизации BDDI [16], использующий три эвристики и применимый для исходного матричного задания систем булевых функций в виде ДНФ;

– программа минимизации числа вершин графа BBDD (Biconditional Binary Decision Diagrams – диаграмма двоичного выбора с двумя условиями), задающего последовательные разложения (5) (URL: http://lsi.epfl.ch/BBDD).

Для программы минимизации BBDD исходными данными являются булевы сети, функциями вершин которых могут быть двухоперандные логические операции: конъюнкция, дизъюнкция, исключающее ИЛИ, а также однооперандная операция – инверсия.

В качестве библиотеки синтеза в эксперименте 3 использовалась та же библиотека (power) КМОП-элементов, что и в работе [16].

Исходные матричные описания систем ДНФ булевых функций, используемые в работе [16] и взятые из набора [19], были переведены в булевы сети, задаваемые в виде структурных описаний на языке Verilog [20]. Программа минимизации BBDD, реализующая алгоритмы из работы [14], минимизировала число вершин в графе, задающем BBDD-представление системы функций. Минимизированные BBDD-представления задавались в виде функциональных опи-

саний на языке Verilog, которые были исходными для синтеза логических схем в библиотеке power с помощью синтезатора LeonardoSpectrum.

Для примера в левой части табл. 5 приведены логические уравнения (1) на языке Verilog, являющиеся исходными данными для программы минимизации BBDD, а в правой части – функциональное описание графа BBDD (рис. 4), полученное по описанию из левой части табл. 5 с помощью этой программы.

Таблица 5

Verilog-описания: исходные данные и результат работы программы минимизации BBDD для булевой сети, заданной уравнениями (1)

Исходные данные (структурное описание булевой сети)	Результат (функциональное описание BBDD)
module example1 (x0, x1, x2, x3, y1, y2); *input x0, x1, x2, x3;* *output y1, y2;* *wire tmp0, tmp2, tmp3, tmp4, tmp5, tmp6, tmp7, tmp8;* *orx ix3 (.a (x0), .b (tmp0), .O (y1));* *orx ix6 (.a (x0), .b (tmp3), .O (y2));* *andx ix5 (.a (tmp7), .b (x3), .O (tmp2));* *invx ix2 (.a (x2), .O (tmp7));* *andx ix8 (.a (tmp6), .b (x2), .O (tmp4));* *andx ix9 (.a (tmp6), .b (tmp8), .O (tmp5));* *invx ix10 (.a (x3), .O (tmp8));* *invx ix1 (.a (x1), .O (tmp6));* *orx ix4 (.a (x1), .b (tmp2), .O (tmp0));* *orx ix7 (.a (tmp4), .b (tmp5), .O (tmp3));* *endmodule*	*module example1 (x0, x1, x2, x3, y1, y2);* *input x0, x1, x2, x3;* *output y1, y2;* *wire one, node6, node3, node1, node5, node4, node2;* *assign node1 = (x1 ^ x0) ? one : x0;* *assign node3 = (x2 ^ x1) ? node1 : one;* *assign node6 = (x3 ^ x2) ? node3 : node1;* *assign node2 = (x1 ^ x0) ? x0 : one;* *assign node4 = (x2 ^ x1) ? node2 : x0;* *assign node5 = (x3 ^ x2) ? node4 : node2;* *assign one = 1;* *assign y1 = node6;* *assign y2 = node5;* *endmodule*

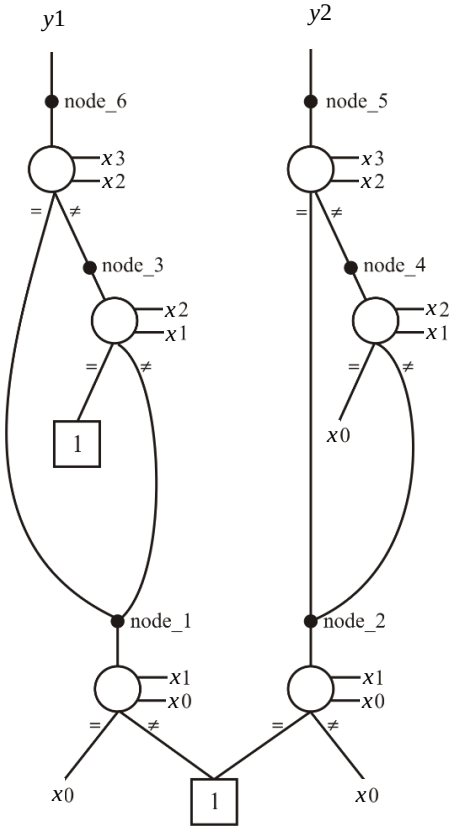


Рис. 4. Граф BBDD для примера example1

В структурном описании используются следующие логические элементы: *orx* – логическое ИЛИ (дизъюнктор), *andx* – логическое И (конъюнктор), *invx* – логическое отрицание (инвертор); в описаниях данных элементов: *a, b* – имена входов, *O* – имя выхода. В функциональном описании BBDD используется оператор *assign* присвоения значения сигналу, в правой части находятся логические операторы ^ ($\oplus$ – исключающее ИЛИ), а также тернарные (условные) операторы «?». Например, оператор

assign node6 = (x3 ^ x2) ? node3 : node1;

понимается следующим образом: если выражение $x3 \oplus x2$ истинно, то промежуточный сигнал *node6* получает значение сигнала *node3*, если же выражение $x3 \oplus x2$ ложно, то сигнал *node6* получает значение сигнала *node1*. Более подробно об операторах языка Verilog можно прочесть в работе [20], там же дано соответствие операторов языков VHDL и Verilog. Функциональное описание графа BBDD на языке VHDL для примера, приведенного в табл. 5, дано ниже:

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;
entity example1 is
  port (
    x0, x1, x2, x3 : IN std_logic ;
    y1, y2 : OUT std_logic) ;
end;
architecture beh of example1 is
signal node1, node2, node3,node4, node5, node6 : std_logic ;
begin
node1 <= (( x1 xor x0) and '1')       or  ((x1 xnor x0) and x0);
node2 <= (( x1 xor x0) and x0)        or  ((x1 xnor x0) and '1');
node3 <= (( x2 xor x1) and  node1) or  ((x2 xnor x1) and '1');
node4 <= (( x2 xor x1) and  node2) or  ((x2 xnor x1) and x0);
node5 <= (( x3 xor x2) and node4)  or  ((x3 xnor x2) and node2);
node6 <= (( x3 xor x2) and node3)  or  ((x3 xnor x2) and node1);
y1  <= node6;
y2  <= node5;
end;
```

Каждая функциональная вершина графа BBDD на рис. 4, т. е. вершина, в которую слева заходят две булевы переменные, соответствует разложению (5). Проиллюстрируем построение графа на примере функции узла node3:

$$z = {\wedge}x0 * x1 * x2 + x0 * x1 * x2 + {\wedge}x0 * {\wedge}x1 * {\wedge}x2 + x0 * {\wedge}x1 * {\wedge}x2 + \\ + {\wedge}x0 * x1 * {\wedge}x2 + x0 * {\wedge}x1 * x2 + x0 * x1 * {\wedge}x2. \qquad (13)$$

Построим разложение вида (5) функции z, полагая $x_i = x2$, $x_j = x1$:

$$z = (x2 \oplus x1) * z(x0, {\wedge}x1, x1) + (x2 \sim x1) * z(x0, x1, x). \qquad (14)$$

Для вычисления подфункции $z(x0, {\wedge}x1, x1)$ заменим в формуле (13) переменную $x2$ на $^x1$, получим функцию node1 = $^x0 * x1 + x0 * ^x1 + x0 * x1$. Для вычисления $z(x0, x1, x1)$ в (14) заменим в (13) переменную $x2$ на $x1$, получим $z(x0, x1, x1) = {\wedge}x0 * x1 + x0 * x1 + {\wedge}x0 * {\wedge}x1 + x0 * {\wedge}x1 = 1$. Аналогично поступим и для остальных функциональных вершин графа BBDD.

Булевы сети, являющиеся исходными для программы BDD_Builder [16], после замены операции $\oplus$ исключающего ИЛИ формулой ($x_i \oplus x_j = \bar{x}_i x_j \vee x_i \bar{x}_j$) переводились в VHDL-описания и подавались на вход синтезатора LeonardoSpectrum.

Результаты экспериментов 1 и 2 представлены в табл. 6, где n – число переменных, m – число функций, S_{ASIC} – площадь логической схемы (суммарное число транзисторов), τ – задержка схемы (нс). Лучшие решения выделены жирным шрифтом.

Таблица 6

Результаты экспериментов 1 и 2, синтез схем в библиотеке *s3lib*

Вид исходного задания	Схема	n	m	Синтез по исходным описаниям		Синтез по оптимизированным описаниям, программа BDD Builder2					
						Эвристика 1		Эвристика 2		Эвристика 3	
				S_{ASIC}	τ	S_{ASIC}	τ	S_{ASIC}	τ	S_{ASIC}	τ
Результаты эксперимента 1											
Unmap-описания	add6	12	7	2812	6,61	208	3,51	**206**	**3,01**	208	3,42
	b2	16	7	**2210**	6,03	2860	**4,43**	3888	5,16	2834	4,48
	intb	15	7	6848	8,01	**4982**	5,40	6008	5,93	5416	**4,44**
	root	8	5	814	3,50	**496**	**2,59**	**496**	**2,59**	**496**	**2,59**
	sist_4	17	12	**828**	3,91	2030	**3,91**	1942	4,13	2170	4,42
	tial	14	8	5184	6,93	5144	**3,82**	5564	5,10	**4904**	4,29
	tms	8	16	**636**	5,09	684	2,74	678	2,80	714	**2,46**
	z5xp1	7	10	1380	5,04	502	2,37	**460**	**1,75**	516	2,11
	z9sym	9	1	426	4,81	**226**	2,55	**226**	2,55	**226**	2,55
Исходные описания	FRG2	143	139	4224		9458	6,52			13144	7,39
	APEX6	135	94	**1832**	3,34	1992	2,95	2222	3,07	1900	**2,54**
	C8	28	18	310	1,30	312	**1,30**	322	1,40	**308**	**1,30**
	CHT	47	36	680	1,51	**658**	1,34	696	1,32	662	**1,02**
	COUNT	35	16	**256**	4,07	**256**	**4,07**	**256**	**4,07**	**256**	**4,07**
	DALU	75	16	1834	5,24	**932**	**2,61**			1038	3,07
Результаты эксперимента 2											
Unmap-описания	add6	12	7	2812	6,61	**204**	**2,96**	206	**2,96**	**204**	**2,96**
	b2	16	7	2210	6,03	2762	4,42	3842	5,20	3116	4,77
	intb	15	7	6848	8,01	6658	5,84	6270	5,70	5230	5,08
	root	8	5	814	3,50	**416**	**2,31**	**416**	**2,31**	**416**	**2,31**
	sist_4	17	12	828	3,91	2118	4,16	1700	3,87	1818	3,54
	tial	14	8	5184	6,93	5664	4,68	5002	5,11	4984	3,96
	tms	8	16	636	5,09	–	–	688	**2,43**	700	2,66
	z5xp1	7	10	1380	5,04	516	2,10	550	1,89	530	2,10
	z9sym	9	1	426	4,81	**222**	2,60	**222**	2,60	**222**	2,60
Исходные описания	FRG2	143	139	4224	–	9424	5,79	–	–	–	–
	APEX6	135	94	1832	3,34	2256	3,26	2444	3,69	1936	2,97
	C8	28	18	310	1,30	312	**1,09**	326	1,40	318	1,30
	CHT	47	36	680	1,51	660	1,07	696	1,36	662	1,04
	COUNT	35	16	256	4,07	264	4,27	256	4,07	264	4,27
	DALU	75	16	1834	5,24	**872**	2,57	1274	4,05	994	3,44

Анализируя данные табл. 6, можно сделать вывод о том, что дополнительные проверки и упрощения по формулам (12) лишь незначительно улучшают решения (либо могут даже незначительно ухудшить их, см. результаты для intb, tial, C8, CHT). В целом можно отметить, что предложенный способ многоуровневой оптимизации практически всегда позволяет при последующем синтезе уменьшать задержки схем.

Результаты эксперимента 3 представлены в табл. 7, где k – число общих элементарных конъюнкций в матричном задании системы ДНФ булевых функций, S_{ASIC} – суммарная площадь логических элементов схемы в условных единицах площади. Лучшие решения по трем эвристикам выделены жирным шрифтом.

Эксперимент 3 показал, что в одной трети случаев программа BDD_Builder2 позволяет получать лучшие решения по площади в сравнении с программой BDD_Builder, исходными данными для которой являются системы ДНФ, заданные в матричном виде. Однако переход к таким формам задания не всегда возможен для систем булевых функций, зависящих от 35 и более переменных. Разработанная программа BDD_Builder2 ориентируется на более общий способ задания минимизируемых систем булевых функций – булевы сети, и в этом заключается ее преимущество.

Кроме того, программа минимизации BBDD оказалась неконкурентоспособной для применения в качестве оптимизационной процедуры при синтезе схем из КМОП-элементов.

Таблица 7

Результаты эксперимента 3, синтез схем в библиотеке power

Схема	*n*	*m*	*k*	Минимизация BDDI, программа BDD_Builder [16], S_{ASIC}	Минимизация булевых сетей, программа BDD_Builder2, S_{ASIC}	Минимизация BBDD, программа Amaru, S_{ASIC}
add6	12	7	1092	**12 806**	15 719	41 967
b12	15	9	431	**16 137**	16 221	41 152
b2	16	17	110	164 526	**161 457**	349 777
b9	16	5	123	26 081	**22 019**	112 169
br1	12	8	34	**23 843**	25 043	42 425
br2	12	8	35	19 653	**19 234**	36 170
dist	8	5	256	**60 085**	66 887	77 752
in0	15	11	138	**91 116**	91 367	177 729
in2	19	10	137	**69 811**	84 185	170 106
intb	15	7	664	246 764	**229 717**	397 514
life	9	1	512	14 391	**14 346**	21 159
log8mod	8	5	47	23 687	**22 817**	31 192
m181	15	9	430	16 439	**15 808**	40 795
mlp4	8	8	256	**68 439**	71 441	86 127
newtpla	15	5	23	**11 316**	12 577	28 118
newtpla1	10	2	4	**3421**	3962	9977
newtpla2	10	4	9	**6640**	**6640**	140 90
p82	5	14	24	**19 988**	20 032	25 713
radd	8	5	120	**8074**	8119	11 539
rd53	5	3	32	7321	**6830**	8928
rd73	7	3	147	**15 925**	16 656	16 383
root	8	5	256	26 075	**23 732**	39 590
sex	9	14	23	**11 891**	12 354	31 672
tial	14	8	640	**261 278**	285 250	482 341

Заключение. Разработанная программа BDD_Builder2 глобальной оптимизации булевых сетей на основе разложения Шеннона позволяет обрабатывать примеры систем с десятками и сотнями переменных и функций, а также уменьшать площадь схем при повторном синтезе (на потоке unmap-описаний) в шести случаях из девяти. При этом для примеров add6, z5xp1, DALU были получены значительные выигрыши по площади. Предварительная глобальная оптимизация многоуровневой логики также часто целесообразна, однако в некоторых примерах (например, FRG2) реализация исходного многоуровневого описания приводит к схеме меньшей площади. Применение разработанной программы BDD_Builder2 оптимизации булевых сетей позволяет в двух третях случаев (примеров) улучшать результаты (площадь и быстродействие) логического синтеза в экспериментах 1 и 2. В эксперименте 3 программа BDD_Builder2 в 10 случаях из 24 позволила улучшить результаты синтеза по сравнению с программой, оптимизирующей матричные представления систем ДНФ булевых функций. Это свидетельствует о том, что переход к представлениям функций в виде булевых сетей может быть целесообразен и в тех случаях, когда имеются матричные представления систем ДНФ функций.

Программа BDD_Builder2 успешно прошла промышленную проверку и включена в систему FLC логической оптимизации.

Список использованных источников

1. Advanced Techniques in Logic Synthesis, Optimizations and Applications / ed.: S. P. Khatri, K. Gulati. – Springer, 2010. – 423 p.

2. Advanced Logic Synthesis / ed.: A. I. Reis, R. Drechsler. – Springer, 2017. – 232 p.

3. Брейтон, Р. К. Синтез многоуровневых комбинационных логических схем / Р. К. Брейтон, Г. Д. Хэчтел, А. Л. Санджованни-Винчентелли // ТИИЭР. – 1990. – Т. 78, № 2. – С. 38–83.

4. Logic Minimization Algorithm for VLSI Synthesis / K. R. Brayton [et al.]. – Boston : Kluwer Academic Publishers, 1984. – 193 p.

5. Brayton, K. R. Factoring logic functions / K. R. Brayton // IBM J. of Research & Development. – 1987. – Vol. 31, no. 2. – P. 187–198.

6. Синтез асинхронных автоматов на ЭВМ / под ред. А. Д. Закревского. – Минск : Наука и техника, 1975. – 184 с.

7. MIS: A multiple-level logic optimization systems / K. R. Brayton [et al.] // IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems. – 1987. – Vol. 6, no. 6. – P. 1062–1081.

8. Mailhot, F. Algorithms for technology mapping based on binary decision diagrams and on Boolean operations / F. Mailhot, G. Micheli // IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems. – 1993. – Vol. 12, no. 5. – P. 599–620.

9. Bryant, R. E. Graph-based algorithms for Boolean function manipulation / R. E. Bryant // IEEE Transactions on Computers. – 1986. – Vol. 35, no. 8. – P. 677–691.

10. Bryant, R. E. Ordered binary decision diagrams / R. E. Bryant, C. Meinel // Logic Synthesis and Verification / ed.: S. Hassoun, T. Sasao, R. K. Brayton. – Kluwer Academic Publishers, 2002. – P. 285–307.

11. Yang, S. BDS: a BDD-based logic optimization system / S. Yang, M. Ciesielski // IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems. – 2002. – Vol. 21, no. 7. – P. 866–876.

12. Бибило, П. Н. Применение диаграмм двоичного выбора при синтезе логических схем / П. Н. Бибило. – Минск : Беларус. навука, 2014. – 231 с.

13. Бибило, П. Н. Системы проектирования интегральных схем на основе языка VHDL. StateCAD, ModelSim, LeonardoSpectrum / П. Н. Бибило. – М. : СОЛОН-Пресс, 2005. – 384 с.

14. Amaru, L. G. New Data Structures and Algorithms for Logic Synthesis and Verification / L. G. Amaru. – Springer, 2017. – 156 p.

15. Прихожий, А. А. Частично определенные логические системы и алгоритмы / А. А. Прихожий. – Минск : БНТУ, 2013. – 343 с.

16. Бибило, П. Н. Использование полиномов Жегалкина при минимизации многоуровневых представлений систем булевых функций на основе разложения Шеннона / П. Н. Бибило, Ю. Ю. Ланкевич // Программная инженерия. – 2017. – № 3. – С. 369–384.

17. Закревский, А. Д. Логические основы проектирования дискретных устройств / А. Д. Закревский, Ю. В. Поттосин, Л. Д. Черемисинова. – М. : Физматлит, 2007. – 592 с.

18. Бибило, П. Н. Логическое проектирование дискретных устройств с использованием продукционно-фреймовой модели представления знаний / П. Н. Бибило, В. И. Романов. – Минск : Беларус. навука, 2011. – 279 с.

19. Jeong, C. Computer-aided design of digital systems / C. Jeong [Electronic resource] // Department of Computer Scienc. – Mode of access: http://www1.cs.columbia.edu/~cs6861/sis/espresso-examples/ex. – Date of access: 20.03.2018.

20. Поляков, А. К. Языки VHDL и VERILOG в проектировании цифровой аппаратуры / А. К. Поляков. – М. : СОЛОН-Пресс, 2003. – 320 с.

References

1. Khatri S. P., Gulati K. (eds.). *Advanced Techniques in Logic Synthesis, Optimizations and Applications*. Springer, 2010, 423 p.

2. Reis A. I., Drechsler R. (eds.). *Advanced Logic Synthesis*. Springer, 2017, 232 p.

3. Brayton R. K., Hachtel G. D., Sangiovanni-Vincentelli A. L. Sintez mnogourovnevyh kombinacionnyh logicheskih skhem [Multilevel logic synthesis]. Trudy Instituta inzhenerov po jelektronike i radiotehnike [*Proceedings of the Institute of Electronics and Radio Engineering*], 1990, vol. 78, no. 2, pp. 38–83 (in Russian).

4. Brayton K. R., Hachtel G. D., McMullen C., Sangiovanni-Vincentelli A. L. *Logic Minimization Algorithm for VLSI Synthesis*. Boston, Kluwer Academic Publishers, 1984, 193 p.

5. Brayton K. R. Factoring logic functions. *IBM Journal of Research & Development*, 1987, vol. 31, no. 2, pp. 187–198.

6. Zakrevskij A. D. Sintez asinhronnyh avtomatov na JeVM. *Synthesis of Asynchronous Machines on a Computer*. Minsk, Nauka i tekhnika, 1975, 184 p.

7. Brayton K. R., Rudell R., Sangiovanni-Vincentelli A. L., Wang A. R. MIS: A multiple-level logic optimization systems. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 1987, vol. 6, no. 6, pp. 1062–1081.

8. Mailhot F., Micheli G. Algorithms for technology mapping based on binary decision diagrams and on Boolean operations. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 1993, vol. 12, no. 5, pp. 599–620.

9. Bryant R. E. Graph-based algorithms for Boolean function manipulation. *IEEE Transactions on Computers*, 1986, vol. 35, no. 8, pp. 677–691.

10. Bryant R. E., Meinel C. Ordered binary decision diagrams. *Logic Synthesis and Verification*. In Hassoun S., Sasao T., Brayton R. K. (eds.). Kluwer Academic Publishers, 2002, pp. 285–307.

11. Yang S., Ciesielski M. BDS: a BDD-based logic optimization system. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2002, vol. 21, no. 7, pp. 866–876.

12. Bibilo P. N. Primenenie diagramm dvoichnogo vybora pri sinteze logicheskih shem. *Application of Binary Decision Diagrams at Synthesis of Logical Circuits.* Minsk, Belaruskaja navuka, 2014, 231 p. (in Russian).

13. Bibilo P. N. Cistemy proektirovaniya integral'nyh skhem na osnove yazyka VHDL. StateCAD, ModelSim, LeonardoSpectrum. *Integrated Circuit Design Systems Based on the VHDL Language. StateCAD, ModelSim, LeonardoSpectrum*. Moscow, SOLON-Press, 2005, 384 p. (in Russian).

14. Amaru L. G. *New Data Structures and Algorithms for Logic Synthesis and Verification*. Springer, 2017, 156 p.

15. Prihozhij A. A. Chastichno opredelennye logicheskie sistemy i algoritmy. *Partially Defined the Logical Systems and Algorithms*. Minsk, Belorusskij nacional'nyj tehnicheskij universitet, 2013, 343 p.

16. Bibilo P. N., Lankevich Yu. Yu. Ispol'zovanie polinomov Zhegalkina pri minimizacii mnogourovnevyh predstavlenij sistem bulevyh funkcij na osnove razlozheniya Shennona [The use of Zhegalkin polynomials with minimization of multilevel representations of systems of Boolean functions on the basis of the Shannon decomposition]. Programmnaya inzheneriya [*Software Engineering*], 2017, no. 3, pp. 369–384 (in Russian).

17. Zakrevskij A. D., Pottosin Ju. V., Cheremisinova L. D. Logicheskie osnovy proektirovanija diskretnyh ustrojstv. *Logical Bases of Design of Discrete Devices*. Moscow, Fizmatlit, 2007, 592 p. (in Russian).

18. Bibilo P. N., Romanov V. I. Logicheskoe proektirovanie diskretnyh ustrojstv s ispol'zovaniem produkcionno-frejmovoj modeli predstavlenija znanij. *Logical Design of Discrete Devices with Use of Productional and Frame Model of Representation of Knowledge*. Minsk, Belaruskaja navuka, 2011, 279 p. (in Russian).

19. Jeong C. Computer-aided design of digital systems. *Department of Computer Science*. Available at : http://www1.cs.columbia.edu/~cs6861/sis/espresso-examples/ex (accessed 20.03.2018).

20. Poljakov A. K. Jazyki VHDL i VERILOG v proektirovanii cifrovoj apparatury. *VHDL and VERILOG in the design of digital equipment*. Moscow, SOLON-Press, 2003, 320 p.

Информация об авторах

Бибило Петр Николаевич, доктор технических наук, профессор, Объединенный институт проблем информатики Национальной академии наук Беларуси, Минск, Беларусь.
E-mail: bibilo@newman.bas-net.by

Ланкевич Юрий Юрьевич, младший научный сотрудник, Объединенный институт проблем информатики Национальной академии наук Беларуси, Минск, Беларусь.
E-mail: yurafreedom18@gmail.com

Information about the authors

Petr N. Bibilo, Dr. Sci. (Eng.), Prof., The United Institute of Informatics Problems of the National Academy of Sciences of Belarus, Minsk, Belarus.
E-mail: bibilo@newman.bas-net.by

Yury Y. Lankevich, Junior Researcher, The United Institute of Informatics Problems of the National Academy of Sciences of Belarus, Minsk, Belarus.
E-mail: yurafreedom18@gmail.com

ISSN 1816-0301 (Print)
ISSN 2617-6963 (Online)

ЗАЩИТА ИНФОРМАЦИИ И НАДЕЖНОСТЬ СИСТЕМ

INFORMATION PROTECTION AND SYSTEM RELIABILITY

УДК 621.383

Поступила в редакцию 14.01.2019
Received 14.01.2019

Принята к публикации 15.02.2019
Accepted 15.02.2019

Достоверность принятой информации при ее регистрации в однофотонном канале связи при помощи счетчика фотонов

А. М. Тимофеев

Белорусский государственный университет информатики и радиоэлектроники, Минск, Беларусь
E-mail: tamvks@mail.ru

Аннотация. При измерении маломощных оптических сигналов приемные модули систем должны обеспечивать достаточно высокую достоверность принятых данных. В этой связи целесообразно использовать высокочувствительные счетчики фотонов, которым, однако, присущи ошибки регистрации данных. Поэтому целью работы является оценка влияния средней скорости счета импульсов на выходе счетчика фотонов на достоверность принятой информации с учетом мертвого времени счетчика фотонов.

Получено выражение для расчета достоверности данных, принятых по асинхронному двоичному несимметричному однородному каналу связи без памяти и со стиранием, в котором в качестве приемного модуля используется счетчик фотонов с мертвым временем продлевающегося типа.

По результатам математического моделирования установлено, что с ростом средней скорости счета сигнальных импульсов на выходе счетчика фотонов при передаче символов 1 (n_{s1}) достоверность принятых данных D растет, достигая насыщения. При прочих равных параметрах с увеличением средней длительности мертвого времени продлевающегося типа (τ_d) насыщение зависимости $D(n_{s1})$ наблюдается при больших значениях средней скорости счета сигнальных импульсов: $n_{s1} \geq 35{,}0\times10^4$ с$^{-1}$ для $\tau_d = 0$; $n_{s1} \geq 38{,}9\times10^4$ с$^{-1}$ для $\tau_d = 5$ мкс; $n_{s1} \geq 43{,}7\times10^4$ с$^{-1}$ для $\tau_d = 10$ мкс; $n_{s1} \geq 50{,}0\times10^4$ с$^{-1}$ для $\tau_d = 15$ мкс.

Ключевые слова: канал связи, однофотонная передача информации, достоверность принятой информации, счетчик фотонов, мертвое время

Для цитирования. Тимофеев, А. М. Достоверность принятой информации при ее регистрации в однофотонном канале связи при помощи счетчика фотонов / А. М. Тимофеев // Информатика. – 2019. – Т. 16, № 2. – С. 90–98.

The reliability of the received information if it is registered in the single photon communication channel using the photon counter

Alexander M. Timofeev

Belarusian State University of Informatics and Radioelectronics, Minsk, Belarus
E-mail: tamvks@mail.ru

Abstract. When measuring low-power optical signals, the receiving modules of systems should ensure a sufficiently high accuracy of the received data. In this regard, it is advisable to use photon counters. They are highly sensitive, but are characterized by data recording errors. The aim of this work is to determine the

influence of average pulse count rate of photons as the output of the counter on the fidelity of received information taking into account also the dead time of the counter.
The formula is obtained for calculating the reliability of data received over an asynchronous binary asymmetric homogeneous communication channel without memory with erasure, when photon counter with a dead time of prolonging type was used as a receiving module.
According to the results of mathematical modeling, it was established that with an increase in the average count rate of signal pulses at the output of the photon counter in symbols 1 (n_{s1}), the reliability of the received data D grows up to saturation. Moreover, when other parameters being equal, with an increase of the average duration of the dead time of a prolonged type (τ_d), the saturation of the $D(n_{s1})$ dependence is fixed for large values of the average counting rate of signal pulses. For example, with $n_{s1} \geq 35{,}0\times10^4$ s^{-1} for $\tau_d = 0$; with $n_{s1} \geq 38{,}9\times10^4$ s^{-1} for $\tau_d = 5$ μs; with $n_{s1} \geq 43{,}7\times10^4$ s^{-1} for $\tau_d = 10$ μs; with $n_{s1} \geq 50{,}0\times10^4$ s^{-1} for $\tau_d = 15$ μs.

Keywords: communication channel, single photon information transfer, reliability of the received information, photon counter, dead time

For citation. Timofeev A. M. The reliability of the received information if it is registered in the single photon communication channel using the photon counter. *Informatics*, 2019, vol. 16, no. 2, pp. 90–98 (in Russian).

Введение. Одной из основных задач, решаемых при построении волоконно-оптических каналов связи, в которых информационная безопасность достигается за счет использования маломощных оптических импульсов, является обеспечение достаточно высокой достоверности данных при их регистрации [1–3]. Под достоверностью будем понимать вероятность того, что принятые данные соответствуют переданным [3]. При использовании маломощных оптических импульсов, содержащих не более десяти фотонов в расчете на каждый бит (символ), что имеет место для квантово-криптографических каналов связи, решение указанной задачи возможно благодаря применению высокочувствительных приемных модулей – счетчиков фотонов, построенных на базе лавинных фотоприемников [1–4]. Известные методы оценки показателей надежности, учитывающие ошибки при передаче информации, не применимы для однофотонных каналов связи. Так, например, методы, представленные в работах [5–8], не учитывают мертвое время счетчика фотонов. В течение мертвого времени счетчик фотонов не чувствителен к падающему на него оптическому излучению [1, 2, 9, 10]. В результате возникают так называемые просчеты, которые влияют на количество ошибок при регистрации данных, а следовательно, и на достоверность принятых данных. В работе [4] применительно к однофотонным каналам связи получены выражения для оценки вероятностей ошибочной регистрации двоичных символов 0 и 1, которые учитывают длительность мертвого времени счетчика фотонов. Однако данные, как правило, представляют собой не одноименные символы, а последовательности, содержащие как символы 0, так и символы 1.

Целью настоящего исследования являлось установление влияния скорости счета импульсов на выходе счетчика фотонов на достоверность принятой информации при передаче данных по однофотонному каналу связи с учетом мертвого времени счетчика фотонов.

Объектом исследования был асинхронный двоичный несимметричный однородный однофотонный канал связи без памяти и со стиранием, содержащий в качестве приемного модуля счетчик фотонов с мертвым временем продлевающегося типа. Выбор в качестве объекта исследования такого канала связи обусловлен тем, что его использование не требует наличия дополнительных линий связи для передачи и приема синхроимпульсов [2–4, 11]. Мертвым временем продлевающегося типа характеризуются счетчики фотонов на базе лавинных фотоприемников, включенных по схеме пассивного гашения лавины [2].

Выражение для оценки достоверности принятых данных. Дальнейшие рассуждения будут основаны на том, что передача информации осуществляется по однофотонному каналу связи двоичными символами (0 и 1) в течение времени τ_b. При этом при передаче символов 0 и 1 используются оптические сигналы мощностью W_1 и W_2 соответственно ($W_1 < W_2$), которые содержат от одного до нескольких десятков фотонов и транслируются в линию связи в течение времени однофотонной передачи $\Delta t = \tau_b /2$, а прием ведется с помощью счетчика фотонов, выполненного на базе лавинного фотоприемника, включенного по схеме пассивного гашения лавины [2]. Следовательно, в течение времени $t_з = \tau_b /2$ данные в канал связи не передаются, т. е. между каждой парой символов находится так называемый «защитный» временной интер-

вал. Поскольку символы 0 и 1 передаются импульсами различной мощности, то на выходе счетчика фотонов за время Δt формируется различное количество электрических импульсов, которое будет прямо пропорционально мощности оптического излучения. Всеми потерями информации, за исключением потерь в счетчике фотонов, пренебрегаем.

Обозначим вероятности появления символов 0 и 1 на выходе канала связи как $P'_s(0)$ и $P'_s(1)$ соответственно. Поскольку в общем случае принимаемые двоичные данные содержат как символы 0, так и символы 1, для оценки достоверности принятых данных воспользуемся формулой

$$D = P'_s(0)D_0 + P'_s(1)D_1, \tag{1}$$

где D_0 и D_1 – достоверности принятых символов 0 и 1 соответственно.

Достоверности D_0 и D_1 можно определить на основании соответствующих вероятностей ошибочной регистрации символов 0 и 1 ($P_{ош0}$ и $P_{ош1}$), полученных в работе [4]. При этом необходимо учитывать следующее. Если число зарегистрированных на выходе счетчика фотонов импульсов находится в диапазоне от нижнего порогового уровня регистрации N_1 до верхнего порогового уровня регистрации N_2, делается вывод, что передан символ 0. При превышении зарегистрированных импульсов числа N_2 делается вывод, что передан символ 1. В случае регистрации импульсов в количестве, меньшем N_1, принимается решение, что символ отсутствует.

Таким образом, алфавит кодовых слов на входе канала связи не совпадает с алфавитом кодовых слов на его выходе. Вероятность отсутствия символа 0 или 1 на выходе канала связи, как и вероятность его приема, не зависит ни от того, какой символ был на входе канала, ни от ранее принятых символов. При передаче символа 0 или 1 на выходе канала связи может быть не зарегистрировано ни символа 0, ни символа 1. Следовательно, рассматриваемый канал связи является однофотонным асинхронным двоичным несимметричным однородным без памяти и со стиранием [12].

Учитывая приведенные выше рассуждения, можно записать:

$$D_0 = \frac{P(0/0)}{P(0/0) + P(0/1)}, \tag{2}$$

$$D_1 = \frac{P(1/1)}{P(1/1) + P(1/0)}, \tag{3}$$

где $P(0/0)$ и $P(1/1)$ – вероятности регистрации на выходе канала связи символов 0 и 1 соответственно при их наличии на входе канала связи; $P(0/1)$ и $P(1/0)$ – вероятности регистрации на выходе канала связи символов 0 и 1 при наличии на входе канала связи символов 1 и 0 соответственно.

Переходные вероятности $P(0/0)$, $P(0/1)$ и $P(1/1)$, $P(1/0)$, входящие в соответствующие выражения (2) и (3), могут быть рассчитаны на основании статистических распределений числа импульсов на выходе счетчика фотонов [13]:

$$P(0/0) = \sum_{N=N_1}^{N_2} \frac{\left[(n_t + n_{s0})(\Delta t - \tau_d)\right]^N \exp\left[-(n_t + n_{s0})(\Delta t - \tau_d)\right]}{N!}, \tag{4}$$

$$P(0/1) = \sum_{N=N_1}^{N_2} \frac{\left[(n_t + n_{s1})(\Delta t - \tau_d)\right]^N \exp\left[-(n_t + n_{s1})(\Delta t - \tau_d)\right]}{N!}, \tag{5}$$

$$P(1/1) = 1 - \sum_{N=0}^{N_2} \frac{\left[(n_t + n_{s1})(\Delta t - \tau_d)\right]^N \exp\left[-(n_t + n_{s1})(\Delta t - \tau_d)\right]}{N!}, \tag{6}$$

$$P(1/0) = 1 - \sum_{N=0}^{N_2} \frac{\left[(n_t + n_{s0})(\Delta t - \tau_d)\right]^N \exp\left[-(n_t + n_{s0})(\Delta t - \tau_d)\right]}{N!}, \tag{7}$$

где n_t – средняя скорость счета темновых импульсов на выходе счетчика фотонов; n_{s0} и n_{s1} – средние скорости счета сигнальных импульсов на выходе счетчика фотонов при передаче символов 0 и 1 соответственно; τ_d – средняя длительность мертвого времени продлевающегося типа.

Отметим, что для оценки мертвого времени продлевающегося типа используют среднее значение, так как его длительность зависит от интенсивности оптического излучения [2].

Темновые и сигнальные – это импульсы, которые появляются на выходе счетчика фотонов соответственно при отсутствии оптического сигнала и в результате воздействия фотонов регистрируемого излучения [1, 2].

Предположим, что на выходе рассматриваемого канала связи зарегистрирована случайная последовательность двоичных данных, для которой $P'_s(0) = P'_s(1) = 0{,}5$. В этом случае достоверность принятых данных определяется подстановкой формул (2) и (3) в (1) с учетом выражений (4)–(7) и окончательно примет следующий вид:

$$D = 0{,}5 \times \left\{ \frac{\sum\limits_{N=N_1}^{N_2} \frac{\left[(n_t + n_{s0})(\Delta t - \tau_d)\right]^N \exp\left[-(n_t + n_{s0})(\Delta t - \tau_d)\right]}{N!}}{\sum\limits_{N=N_1}^{N_2} \frac{\left[(n_t + n_{s0})(\Delta t - \tau_d)\right]^N \exp\left[-(n_t + n_{s0})(\Delta t - \tau_d)\right]}{N!} + \sum\limits_{N=N_1}^{N_2} \frac{\left[(n_t + n_{s1})(\Delta t - \tau_d)\right]^N \exp\left[-(n_t + n_{s1})(\Delta t - \tau_d)\right]}{N!}} + \right.$$
$$\left. + \frac{1 - \sum\limits_{N=0}^{N_2} \frac{\left[(n_t + n_{s1})(\Delta t - \tau_d)\right]^N \exp\left[-(n_t + n_{s1})(\Delta t - \tau_d)\right]}{N!}}{2 - \sum\limits_{N=0}^{N_2} \frac{\left[(n_t + n_{s1})(\Delta t - \tau_d)\right]^N \exp\left[-(n_t + n_{s1})(\Delta t - \tau_d)\right]}{N!} - \sum\limits_{N=0}^{N_2} \frac{\left[(n_t + n_{s0})(\Delta t - \tau_d)\right]^N \exp\left[-(n_t + n_{s0})(\Delta t - \tau_d)\right]}{N!}} \right\}. \quad (8)$$

Результаты математического моделирования и их обсуждение. Вычисление достоверности принятых данных выполнялось для каналов связи, содержащих в качестве приемного модуля счетчик фотонов с мертвым временем продлевающегося типа при различных значениях τ_d и n_{s1}.

На рис. 1 представлена зависимость достоверности принятых данных от средней скорости счета сигнальных импульсов n_{s1} для различных средних длительностей мертвого времени продлевающегося типа.

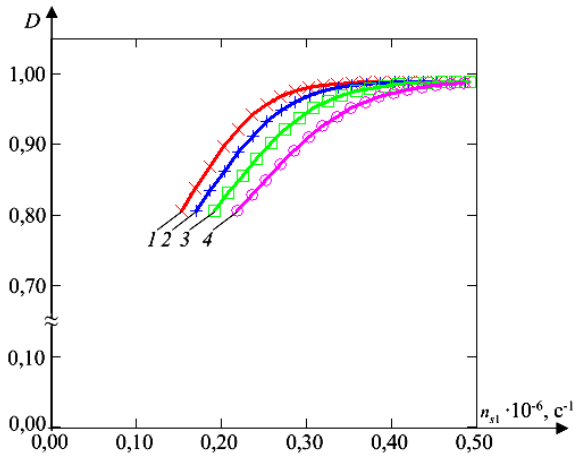


Рис. 1. Зависимость достоверности принятых данных от средней скорости счета сигнальных импульсов n_{s1} при $N_1 = 1$, $N_2 = 7$, $n_t = 10^3$ с$^{-1}$, $\tau_b = 100$ мкс

Средняя длительность мертвого времени:
1 – ×τ_d = 0 мкс, 2 – + τ_d = 5 мкс, 3 – □ τ_d = 10 мкс, 4 – ○ τ_d = 15 мкс

Зависимости $D(n_{s1})$ построены в диапазонах средних скоростей счета сигнальных импульсов, на которых переходные вероятности $P(1/1)$ больше либо равны 0,5 при заданных средних длительностях мертвого времени продлевающегося типа. Это обусловлено тем, что при $P(1/1) < 0{,}5$ использование счетчиков фотонов для регистрации данных в рассматриваемом канале связи становится нецелесообразным. Для сравнения полученных зависимостей $D(n_{s1})$ величины средних скоростей счета сигнальных импульсов n_{s0} фиксировались постоянными и выбирались по методике, описанной в работе [3]. Вначале определялись диапазоны средних скоростей счета сигнальных импульсов n_{s0}, на которых переходные вероятности $P(0/0)$ больше либо равны 0,5 при заданных средних длительностях мертвого времени продлевающегося типа, по аналогии с выбором диапазона значений n_{s1}. Затем из каждого полученного диапазона выбиралось оптимальное значение n_{s0}. При этом критерием оптимальности являлось наименьшее значение n_{s0}, при котором переходная вероятность $P(1/0)$ минимальна. Такой выбор скорости счета сигнальных импульсов n_{s0} позволяет обеспечить наибольшее значение достоверности принятых данных. Расчет проводился для одинаковых значений нижнего и верхнего пороговых уровней регистрации $N_1 = 1$ и $N_2 = 7$, средней скорости счета темновых импульсов $n_t = 10^3$ с$^{-1}$ и среднего времени передачи одного бита (символа) $\tau_b = 100$ мкс. Отметим, что пороговые уровни регистрации N_1 и N_2 можно выбирать и другими, отличными от 1 и 7, но при сравнении зависимостей $D(n_{s1})$ для различных средних длительностей мертвого времени N_1 и N_2 следует фиксировать постоянными, как и среднее значение скорости счета темновых импульсов n_t, и среднее время передачи одного бита (символа) τ_b. При этом важно учитывать, что для рассматриваемого канала связи τ_d не может превышать Δt, которое, в свою очередь, должно быть меньше средней длительности передачи одного бита (символа) τ_b на величину защитного временного интервала. В противном случае использование счетчиков фотонов для регистрации данных становится нецелесообразным [14]. Отметим, что при других значениях N_1, N_2 и отношениях $\tau_d/\Delta t$, n_t/n_{s0} и n_t/n_{s1} проявление эффекта мертвого времени продлевающегося типа для рассматриваемого канала связи аналогично представленному на рис. 1.

Из полученных результатов видно, что с увеличением средних скоростей счета сигнальных импульсов n_{s1} зависимости $D(n_{s1})$ растут, достигая насыщения, что имеет место для всех исследуемых значений τ_d (см. рис. 1). С увеличением средних длительностей мертвого времени продлевающегося типа τ_d это насыщение происходит при больших значениях n_{s1}: $n_{s1} \geq 35{,}0\times10^4$ с$^{-1}$ для $\tau_d = 0$; $n_{s1} \geq 38{,}9\times10^4$ с$^{-1}$ для $\tau_d = 5$ мкс; $n_{s1} \geq 43{,}7\times10^4$ с$^{-1}$ для $\tau_d = 10$ мкс; $n_{s1} \geq 50{,}0\times10^4$ с$^{-1}$ для $\tau_d = 15$ мкс.

Указанные особенности поведения зависимостей $D(n_{s1})$ объясняются характером изменения переходных вероятностей $P(1/1)$ и $P(0/1)$ с увеличением средних скоростей счета сигнальных импульсов n_{s1} (рис. 2).

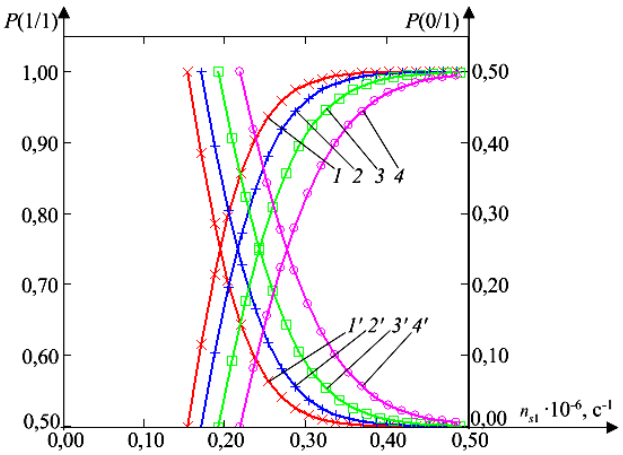


Рис. 2. Зависимости переходных вероятностей $P(1/1)$, кривые 1–4, и $P(0/1)$, кривые 1'–4', от средней скорости счета сигнальных импульсов n_{s1} при $N_1 = 1$, $N_2 = 7$, $n_t = 10^3$ с$^{-1}$, $\tau_b = 100$ мкс

Средняя длительность мертвого времени: 1 и 1' – × $\tau_d = 0$ мкс, 2 и 2' – + $\tau_d = 5$ мкс, 3 и 3' – □ $\tau_d = 10$ мкс, 4 и 4' – ○ $\tau_d = 15$ мкс

На рис. 2 видно, что с увеличением средней скорости счета сигнальных импульсов n_{s1} вероятность $P(1/1)$ растет вплоть до насыщения, а вероятность $P(0/1)$ уменьшается, также переходя в насыщение. Это наблюдается как при наличии мертвого времени продлевающегося типа (см. рис. 2, кривые 2–4, 2'–4'), так и при его отсутствии (см. рис. 2, кривые 1 и 1'). Насыщение зависимостей $P(1/1)$ и $P(0/1)$ от n_{s1} происходит при одних и тех же средних скоростях счета сигнальных импульсов n_{s1} для соответствующих средних длительностей мертвого времени продлевающегося типа: $n_{s1} = 35{,}0\times10^4$ с$^{-1}$ для $\tau_d = 0$; $n_{s1} = 38{,}9\times10^4$ с$^{-1}$ для $\tau_d = 5$ мкс; $n_{s1} = 43{,}7\times10^4$ с$^{-1}$ для $\tau_d = 10$ мкс; $n_{s1} = 50{,}0\times10^4$ с$^{-1}$ для $\tau_d = 15$ мкс. Такое поведение зависимостей $P(1/1)$ и $P(0/1)$ с ростом n_{s1} объясняется тем, что статистические распределения смеси числа темновых и сигнальных импульсов на выходе счетчика фотонов при регистрации символов 1 $P_{st1}(N)$ имеют явно выраженный максимум, свойственный распределению Пуассона [1–4]. Для всех исследуемых диапазонов n_{s1} при наименьших значениях средних скоростей счета сигнальных импульсов n_{s1} этот максимум находится между нижним N_1 и верхним N_2 пороговыми уровнями регистрации. В данном случае достаточно велика вероятность того, что на выходе канала связи будет зарегистрирован символ 0 в то время, когда на вход канала связи подается символ 1, поэтому переходная вероятность $P(0/1)$ максимальна, что, в свою очередь, не позволяет достичь наибольшего значения переходной вероятности $P(1/1)$. С увеличением n_{s1} происходит сдвиг максимумов статистических распределений $P_{st1}(N)$ в сторону больших значений N [4]. Следовательно, повышается вероятность регистрации на выходе счетчика фотонов импульсов в количестве, превышающем верхний пороговый уровень регистрации (N_2), поэтому переходная вероятность $P(0/1)$ уменьшается вплоть до наименьшего значения, а переходная вероятность $P(1/1)$ растет, достигая наибольшего значения (см. рис. 2).

В результате в диапазоне n_{s1}, на котором с увеличением n_{s1} переходная вероятность $P(1/1)$ растет, а переходная вероятность $P(0/1)$ уменьшается, рост зависимости $D(n_{s1})$ объясняется снижением отношения $P(0/1) / P(1/1)$ с увеличением n_{s1} (рис. 3).

В диапазоне n_{s1}, на котором $P(1/1) \approx 1$ и $P(0/1) \approx 0$, зависимость $D(n_{s1})$ практически неизменна и близка к единице за счет того, что отношение $P(0/1) / P(1/1) \approx 0$ (см. рис. 1–3).

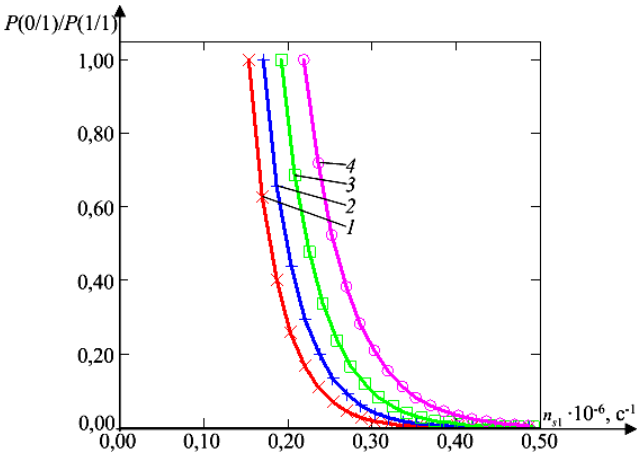


Рис. 3. Зависимость отношения $P(0/1) / P(1/1)$ от средней скорости счета сигнальных импульсов n_{s1} при $N_1 = 1$, $N_2 = 7$, $n_t = 10^3$ с$^{-1}$, $\tau_b = 100$ мкс

Средняя длительность мертвого времени:
1 – × $\tau_d = 0$ мкс, 2 – + $\tau_d = 5$ мкс, 3 – □ $\tau_d = 10$ мкс, 4 – ○ $\tau_d = 15$ мкс

Как показано на рис. 2, в диапазонах средних скоростей счета сигнальных импульсов n_{s1}, на которых зависимости $P(1/1)$ от n_{s1} растут, а $P(0/1)$ от n_{s1} уменьшаются, увеличение средней длительности мертвого времени продлевающегося типа при прочих равных параметрах приводит к уменьшению переходных вероятностей $P(1/1)$ и к росту переходных вероятностей $P(0/1)$. Это обусловлено тем, что при увеличении τ_d максимумы статистических распределений $P_{st1}(N)$ сдвигаются в сторону меньших значений N [4]. В результате такого смещения повышается ве-

роятность регистрации на выходе счетчика фотонов импульсов в количестве, меньшем N_2, поэтому $P(1/1)$ уменьшается, а $P(0/1)$ растет. Так, например, при $n_{s1} = 33{,}0\times10^4$ с$^{-1}$ переходные вероятности $P(1/1)$ и $P(0/1)$ равны соответственно $99{,}28\times10^{-2}$ и $0{,}72\times10^{-2}$ для $\tau_d = 0$; $98{,}09\times10^{-2}$ и $1{,}91\times10^{-2}$ для $\tau_d = 5$ мкс; $95{,}24\times10^{-2}$ и $4{,}76\times10^{-2}$ для $\tau_d = 10$ мкс; $89{,}07\times10^{-2}$ и $10{,}93\times10^{-2}$ для $\tau_d = 15$ мкс. Это приводит к тому, что на всех исследуемых диапазонах средних скоростей счета сигнальных импульсов n_{s1} при прочих равных параметрах с увеличением τ_d достоверность принятых данных уменьшается за счет роста отношения $P(0/1)$ / $P(1/1)$. В результате, например, при $n_{s1} = 28{,}0\times10^4$ с$^{-1}$ достоверность принятых данных D и отношение $P(0/1)$ / $P(1/1)$ равны соответственно $97{,}29\times10^{-2}$ и $3{,}17\times10^{-2}$ для $\tau_d = 0$; $95{,}61\times10^{-2}$ и $6{,}94\times10^{-2}$ для $\tau_d = 5$ мкс; $92{,}76\times10^{-2}$ и $14{,}72\times10^{-2}$ для $\tau_d = 10$ мкс; $88{,}59\times10^{-2}$ и $30{,}79\times10^{-2}$ для $\tau_d = 15$ мкс.

Выполненная оценка показала, что достоверность принятых данных, равная $98{,}70\times10^{-2}$, достигается при наибольших значениях $P(0/0)$ и $P(1/1)$, которые с увеличением τ_d, в свою очередь, обеспечиваются при более высоких значениях n_{s0} и n_{s1} соответственно: $n_{s0} = 66{,}6\times10^3$ с$^{-1}$ и $n_{s1} = 35{,}0\times10^4$ с$^{-1}$ для $\tau_d = 0$; $n_{s0} = 74{,}1\times10^3$ с$^{-1}$ и $n_{s1} = 38{,}9\times10^4$ с$^{-1}$ для $\tau_d = 5$ мкс; $n_{s0} = 83{,}5\times10^3$ с$^{-1}$ и $n_{s1} = 43{,}7\times10^4$ с$^{-1}$ для $\tau_d = 10$ мкс; $n_{s0} = 95{,}6\times10^3$ с$^{-1}$ и $n_{s1} = 50{,}0\times10^4$ с$^{-1}$ для $\tau_d = 15$ мкс.

Заключение. Применительно к асинхронному двоичному несимметричному однородному однофотонному каналу связи без памяти и со стиранием, в котором в качестве приемного модуля используется счетчик фотонов с мертвым временем продлевающегося типа, получено выражение для расчета достоверности принятых данных. На основании этого выражения выполнено математическое моделирование канала связи, по результатам которого установлено, что с ростом средней скорости счета сигнальных импульсов на выходе счетчика фотонов при передаче символов 1 (n_{s1}) достоверность принятых данных D растет, достигая насыщения. Установлено, что при прочих равных параметрах с увеличением средней длительности мертвого времени продлевающегося типа насыщение зависимости $D(n_{s1})$ наблюдается при больших значениях средней скорости счета сигнальных импульсов n_{s1}.

Выполненные исследования также показали, что при прочих равных параметрах увеличение средней длительности мертвого времени продлевающегося типа τ_d приводит к уменьшению достоверности принятых данных за счет роста отношения $P(0/1)$ / $P(1/1)$.

Автору настоящей работы представляются весьма актуальными исследования, направленные на обоснование выбора лавинного фотоприемника, используемого при построении счетчика фотонов. Такие фотоприемники могут отличаться как по структуре полупроводниковых областей, так и по площади фоточувствительной поверхности. В этой связи в ходе дальнейших комплексных исследований планируется определить, как эти параметры влияют на достоверность принятых данных для рассматриваемого канала связи.

Список используемых источников

1. Килин, С. Я. Квантовая криптография: идеи и практика / С. Я. Килин ; ред. С. Я. Килин, Д. Б. Хорошко, А. П. Низовцев. – Минск : Беларус. навука, 2007. – 391 с.

2. Гулаков, И. Р. Фотоприемники квантовых систем : монография / И. Р. Гулаков, А. О. Зеневич. – Минск : УО ВГКС, 2012. – 276 с.

3. Тимофеев, А. М. Методика повышения достоверности принятых данных счетчика фотонов на основе анализа скорости счета импульсов при передаче двоичных символов «0» / А. М. Тимофеев // Приборы и методы измерений. – 2019. – Т. 10, № 1. – С. 80–89.

4. Тимофеев, А. М. Оценка влияния продлевающегося мертвого времени счетчика фотонов на вероятность ошибочной регистрации данных квантово-криптографических каналов связи / А. М. Тимофеев // Вестник связи. – 2018. – № 1(147). – С. 56–62.

5. Дмитриев, С. А. Волоконно-оптическая техника: современное состояние и перспективы / С. А. Дмитриев, Н. Н. Слепов. – 2-е изд., перераб. и доп. – М. : ООО «Волоконно-оптическая техника», 2005. – 576 с.

6. Тузлуков, В. П. Вероятность ошибок при приеме сигналов по уплотненным каналам связи с ортогональным частотным разделением / В. П. Тузлуков // Доклады БГУИР. – 2017. – № 5(107). – С. 77–84.

7. Щеглов, А. Ю. Анализ и проектирование защиты информационных систем. Контроль доступа к компьютерным ресурсам: методы, модели, технические решения / А. Ю. Щеглов. – СПб. : Профессиональная литература, 2017. – 416 с.

8. Олифер, В. Г. Компьютерные сети. Принципы, технологии, протоколы / В. Г. Олифер, Н. А. Олифер. – СПб. : Питер, 2012. – 944 с.

9. Reduced deadtime and higher rate photon-counting detection using a multiplexed detector array / S. A. Castelletto [et al.] // J. of Modern Optics. – 2007. – Vol. 54. – P. 337–352.

10. Single-photon detectors combining high efficiency, high detection rates, and ultra-high timing resolution / I. E. Zadeh [et al.] // APL Photonics. – 2017. – Vol. 2. – P. 111301-1–111301-7.

11. Тимофеев, А. М. Устройство для передачи и приема двоичных данных по волоконно-оптическому каналу связи / А. М. Тимофеев // Приборы и методы измерений. – 2018. – Т. 9, № 1. – С. 17–27.

12. Клюев, Л. Л. Теория электрической связи / Л. Л. Клюев. – Минск : Технoперспектива, 2008. – 423 с.

13. Тимофеев, А. М. Энтропия потерь однофотонного асинхронного волоконно-оптического канала связи с приемником на основе счетчика фотонов с продлевающимся мертвым временем / А. М. Тимофеев // Актуальные проблемы науки XXI века. – 2018. – Вып. 7. – С. 5–10.

14. Тимофеев, А. М. Влияние времени однофотонной передачи информации на вероятность ошибочной регистрации данных асинхронных квантово-криптографических каналов связи / А. М. Тимофеев // Вестник ТГТУ. – 2019. – Т. 25, № 1. – С. 36–46.

References

1. Kilin S. Ya. Kvantovaya kriptografiya: idei i praktika. *Quantum Cryptography: Ideas and Practices*. In Kilin S. Ja., Horoshko D. B., Nizovcev A. P. (eds.). Minsk, Belaruskaja navuka, 2007, 391 p. (in Russian).

2. Gulakov I. R., Zenevich A. O. Fotopriemniki kvantovyih system. *Photodetectors of Quantum Systems*. Minsk, Vysshij gosudarstvennyj kolledzh svjazi, 2012, 276 p. (in Russian).

3. Timofeev A. M. Metodika povyisheniya dostovernosti prinyatyih dannyih schetchika fotonov na osnove analiza skorosti scheta impulsov pri peredache dvoichnyih simvolov «0» [Methods of increasing the reliability of the received data of the photon counter based on the analysis of the pulse counting rate during the transmission of binary symbols «0»]. Pribory i metodyi izmereniy [*Devices and Methods of Measurements*], 2019, vol. 10, no. 1, pp. 80–89 (in Russian).

4. Timofeev A. M. Otsenka vliyaniya prodlevayuschegosya mertvogo vremeni schetchika fotonov na veroyatnost oshibochnoy registratsii dannyih kvantovo-kriptograficheskih kanalov svyazi [Estimation of the photons counter lasting dead time influence on the probability of erroneous data registration of quantum-cryptographic communication channels]. Vestnik svyazi [*Communication Bulletin*], 2018, no. 1(147), pp. 56–62 (in Russian).

5. Dmitriev S. A., Slepov N. N. Volokonno-opticheskaya tehnika: sovremennoe sostoyanie i perspektivyi. *Fiber-Optic Technology: Current Status and Prospects*. Moscow, Volokonno-opticheskaja tehnika, 2005, 576 p. (in Russian).

6. Tuzlukov V. P. Veroyatnost oshibok pri prieme signalov po uplotnennyim kanalam svyazi s ortogonalnyim chastotnyim razdeleniem [Probability of errors when receiving signals on sealed communication channels with orthogonal frequency separation]. Doklady Belorusskogo gosudarstvennogo universiteta informatiki i radiojelektroniki [*Doklady Belarusian State University of Informatics and Radioelectronics*], 2017, no. 5(107), pp. 77–84 (in Russian).

7. Scheglov A. Yu. Analiz i proektirovanie zaschityi informatsionnyih sistem. Kontrol dostupa k kompyuternyim resursam: metodyi, modeli, tehnicheskie resheniya. *Analysis and Design of Information Systems Protection. Control of Access to Computer Resources: Methods, Models, Technical Solutions.* Saint Petersburg, Professional'naja literatura, 2017, 416 p. (in Russian).

8. Olifer V. G., Olifer N. A. Kompyuternyie seti. Printsipyi, tehnologii, protokoly. *Computer Networks. Principles, Technologies, Protocols*. Saint Petersburg, Piter, 2012, 944 p. (in Russian).

9. Castelletto S. A., Degiovanni I. P., Schettini V., Migdall A. L. Reduced deadtime and higher rate photon-counting detection using a multiplexed detector array. *Journal of Modern Optics*, 2007, vol. 54, pp. 337–352.

10. Zadeh I. E., Los J. W., Gourgues R. B., Steinmetz V., Bulgarini G., …, Dorenbos S. N. Single-photon detectors combining high efficiency, high detection rates, and ultra-high timing resolution. *APL Photonics*, 2017, vol. 2, pp. 111301-1–111301-7.

11. Timofeev A. M. Ustroystvo dlya peredachi i priema dvoichnyih dannyih po volokonno-opticheskomu kanalu svyazi [Device for binary data transmitting and receiving over a fiber-optic communication channel]. Pribory i metodyi izmereniy [*Devices and Methods of Measurements*], 2018, vol. 9, no. 1, pp. 17–27 (in Russian).

12. Klyuev L. L. Teoriya elektricheskoy svyazi. *The Theory of Electrical Communication*. Minsk, Tehnoperspektiva, 2008, 423 p. (in Russian).

13. Timofeev A. M. Entropiya poter' odnofotonnogo asinhronnogo volokonno-opticheskogo kanala svyazi s priemnikom na osnove schetchika fotonov s prodlevayuschimsya mertvyim vremenem [Entropy of losses of a single-photon asynchronous fiber-optic communication channel with a receiver based on a photon counter with prolonged dead time]. Aktualniye problemy nauki XXI veka [*Current Issues of Science in the 21st Century*], 2018, vol. 7, pp. 5–10 (in Russian).

14. Timofeev A. M. Vliyanie vremeni odnofotonnoy peredachi informatsii na veroyatnost oshibochnoy registratsii dannyih asinhronnyih kvantovo-kriptograficheskih kanalov svyazi [The effect of single photon transmission time on the probability of erroneous registration of asynchronous data of quantum cryptographic communication channels]. Vestnik Tambovskogo gosudarstvennogo tehnicheskogo universiteta [*Transactions of the Tambov State Technical University*], 2019, vol. 25, no. 1, pp. 36–46 (in Russian).

Информация об авторе

Тимофеев Александр Михайлович, кандидат технических наук, доцент, доцент кафедры защиты информации, Белорусский государственный университет информатики и радиоэлектроники, Минск, Беларусь.
E-mail: tamvks@mail.ru

Information about the author

Alexander M. Timofeev, Cand. Sci. (Eng.), Assoc. Prof., Assoc. Prof. of the Department of Information Security, Belarusian State University of Informatics and Radioelectronics, Minsk, Belarus.
E-mail: tamvks@mail.ru

ISSN 1816-0301 (Print)
ISSN 2617-6963 (Online)
УДК 004.312

Поступила в редакцию 14.01.2019
Received 14.01.2019

Принята к публикации 28.02.2019
Accepted 28.02.2019

Синтез симметричных путей физически неклонируемой функции типа арбитр на FPGA

А. А. Иванюк

Белорусский государственный университет
информатики и радиоэлектроники, Минск, Беларусь
E-mail: ivaniuk@bsuir.by

Аннотация. Физическая криптография является одним из актуальных направлений среди существующих методов защиты цифровых устройств от нелегального доступа. Схемотехнические решения, лежащие в основе физической криптографии, получили название цифровых физически неклонируемых функций (ФНФ), реализация которых обеспечивает уникальность, невоспроизводимость (неклонируемость) защищаемого цифрового устройства. Кроме того, ФНФ эффективны с точки зрения аппаратных ресурсов при их реализации. Существующие ФНФ типа арбитр основаны на синтезе конфигурируемых симметричных путей, каждое звено которых представляет собой пару двухвходовых мультиплексоров, обеспечивающих две конфигурации трансляции тестовых сигналов: прямую и перекрестную. Для построения на программируемой логической интегральной схеме (ПЛИС) типа FPGA одного звена необходимо применение двух встроенных LUT-блоков, обеспечивающих реализацию двух мультиплексоров, при этом ресурсы LUT-блоков используются не полностью. В статье предлагается новая архитектура звеньев симметричных путей ФНФ типа арбитр, позволяющая эффективно применять ресурсы LUT-блоков различных кристаллов FPGA.

Ключевые слова: физически неклонируемая функция, арбитр, симметричные пути, FPGA, LUT-блок

Для цитирования. Иванюк, А. А. Синтез симметричных путей физически неклонируемой функции типа арбитр на FPGA / А. А. Иванюк // Информатика. – 2019. – Т. 16, № 2. – С. 99–108.

Synthesis of symmetric paths of arbiter physically unclonable function on FPGA

Alexander A. Ivaniuk

Belarusian State University of Informatics and Radioelectronics, Minsk, Belarus
E-mail: ivaniuk@bsuir.by

Abstract. Physical cryptography is one of the current trends among the existing methods of protecting digital devices from illegal access. Circuit design solutions in physical cryptography are called digital physically unclonable functions (PUFs), which to be implemented ensure the uniqueness, non-reproducibility (non-cloning) of the protected digital device. In addition, PUFs should be efficient as hardware resources. The existing implementations of the arbiter PUF are based on the synthesis of configurable symmetric paths, when each link is a pair of two-input multiplexers providing two configurations of test signal translation: direct and cross. In order to build a single link on FPGA, it is necessary to use two built-in LUT-blocks, providing the implementation of two multiplexers, meanwhile the hardware resources of LUT-blocks are not fully utilized. The article presents a new architecture of symmetric paths of the arbiter PUF, allowing efficient use of hardware resources of LUT-blocks for various FPGA families.

Keywords: physically unclonable function, arbiter, symmetrical paths, FPGA, LUT-block

For citation. Ivaniuk A. A. Synthesis of symmetric paths of arbiter physically unclonable function on FPGA. *Informatics*, 2019, vol. 16, no. 2, pp. 99–108 (in Russian).

Введение. Обеспечение защиты цифровых устройств от несанкционированного использования, копирования и модификаций достигается различными методами, алгоритмами и техническими средствами. Среди них можно выделить относительно новое направление под общим названием физическая криптография, основу которого составляют так называемые физически неклонируемые функции [1]. Суть ФНФ заключается в извлечении уникальных физических характеристик из изготовленного цифрового устройства. Вариации технологических процессов изготовления интегральных схем вносят в их физическую структуру случайные, непредсказуемые изменения, делающие каждый экземпляр цифрового устройства уникальным, неповторимым и невоспроизводимым. Для извлечения уникальных характеристик устройство проектируют с добавлением специализированных цифровых схем, позволяющих по определенным запросам вырабатывать уникальные цифровые ответы, свойственные только данному экземпляру. В общем случае схемотехническая реализация ФНФ представляет собой схему с n цифровыми входами, на которые подается n-битный запрос C (Challenge) из 2^n возможных, и одним выходом ответа R (Response). Поведение подобной ФНФ-схемы можно описать случайной булевой функцией, осуществляющей отображение $\{0,1\}^n \rightarrow \{0,1\}$. Случайность такой функции обусловлена тем, что данное отображение множества запросов на множество ответов неизвестно до момента изготовления конкретного экземпляра устройства и зависит от случайных неконтролируемых вариаций всех составляющих технологического процесса.

Таким образом, множество всех возможных пар запрос-ответ ФНФ $CR_\alpha = \{c_0 r_0, c_1 r_1, ..., c_{2^n-1} r_{2^n-1}\}$ определяет уникальность конкретного экземпляра α цифрового устройства, а $r_i = PUF_\alpha(c_i)$, $i = \{0,1,...,2^n - 1\}$, определяет уникальную зависимость ответа r_i от запроса c_i.

Для возможности применения в физической криптографии ФНФ должна удовлетворять ряду критериев [2, 3]:

1. Аппаратурные затраты на реализацию ФНФ не должны превышать затраты на реализацию защищаемого устройства.

2. Сбор, хранение и анализ множества CR_α должны быть физически недостижимыми на современном оборудовании и за приемлемое время. ФНФ, обладающую таким свойством, называют сильной ФНФ, для которой параметр n является достаточно большим. Например, для n = 64 только для хранения одного множества R_α необходимо запоминающее устройство с информационной емкостью 16 эксабит, а время сбора всех значений R_α будет составлять более 580 лет с учетом времени отклика устройства, равного 1 нс.

3. Обладая информацией о паре запрос-ответ $c_i r_i$ для конкретного экземпляра устройства α, невозможно рассчитать, смоделировать либо иным математическим способом предсказать значение пары $c_j r_j$, $i \neq j$, или значения множества других пар. Если ФНФ удовлетворяет этому условию, то она считается случайной и непредсказуемой.

4. Для конкретного экземпляра устройства α множество ответов R_α^*, $|R_\alpha^*| < |R_\alpha|$, может быть неоднократно извлечено с высокой степенью надежности путем подачи соответствующего множества различных запросов C_α^*, $|C_\alpha^*| < |C_\alpha|$. Обладающая данным свойством ФНФ считается стабильной.

5. Для множества $\mathrm{A} = \{\alpha_0, \alpha_1, ..., \alpha_{m-1}\}$, $|\mathrm{A}| = m$, различных экземпляров цифрового устройства со встроенной схемой ФНФ, выполненных по единым технологическим нормам, должно соблюдаться следующее условие: $CR_{\alpha_0} \neq CR_{\alpha_1} \neq ... \neq CR_{\alpha_{m-1}}$. Данное условие может быть усилено

следующим образом: $R^*_{\alpha_0} \neq R^*_{\alpha_1} \neq ... \neq R^*_{\alpha_{m-1}}$ для соответствующего множества различных запросов $C^*_{\alpha_0} = C^*_{\alpha_1} = ... = C^*_{\alpha_{m-1}}$, $\left|C^*_{\alpha_0}\right| = \left|C^*_{\alpha_1}\right| = ... = \left|C^*_{\alpha_{m-1}}\right| = \lceil \log_2 m \rceil$. Если описанные условия выполняются, ФНФ считается уникальной.

При удовлетворении описанным критериям ФНФ может быть эффективно использована в качестве криптографического примитива для решения следующих задач [1]:

– неклонируемой идентификации цифровых устройств;
– надежной аутентификации цифровых устройств;
– генерирования случайных невоспроизводимых числовых последовательностей;
– реализации аппаратных хеш-функций;
– реализации аппаратных водяных знаков и отпечатков пальцев;
– защиты цифровых устройств от клонирования и модификаций.

Существует множество схемотехнических реализаций ФНФ для цифровых устройств [1, 2]: ФНФ типа арбитр, ФНФ кольцевых генераторов, ФНФ типа бабочка, ФНФ на основе запоминающих устройств и др. Практически все они основаны на измерении задержек распространения сигналов по путям, сформированным множеством последовательно подключенных цифровых элементов.

Одним из наиболее известных методов реализации ФНФ для цифровых устройств, основанных на измерении задержки распространения сигналов, является ФНФ типа арбитр (А-ФНФ) [4–8]. В отличие от других типов данная ФНФ является сильной и обладает приемлемыми аппаратурными затратами. Однако ее практическая реализация имеет ряд недостатков, которые пытаются устранить разработчики и исследователи в области физической криптографии. К основным проблемам А-ФНФ можно отнести слабую стабильность подмножества пар запрос-ответ и возможность построения ее точной модели по известному множеству пар CR^*_α, $\left|CR^*_\alpha\right| \ll \left|CR_\alpha\right|$ ввиду линейности ее схемотехнической структуры.

Рассмотрим особенности схемотехнического синтеза А-ФНФ для программируемых логических устройств типа FPGA.

Синтез классической схемы А-ФНФ на FPGA. Платы быстрого прототипирования цифровых устройств на основе кристаллов FPGA являются основной платформой для исследования, тестирования и верификации различных схемотехнических решений ФНФ [9, 10]. Рассмотрим обобщенную структуру А-ФНФ и результаты ее синтеза для FPGA.

Структура А-ФНФ состоит из трех основных блоков, последовательно соединенных между собой (рис. 1): генератора тестового сигнала (ГТС), блока симметричных путей (БСП) и блока арбитра (АРБ).

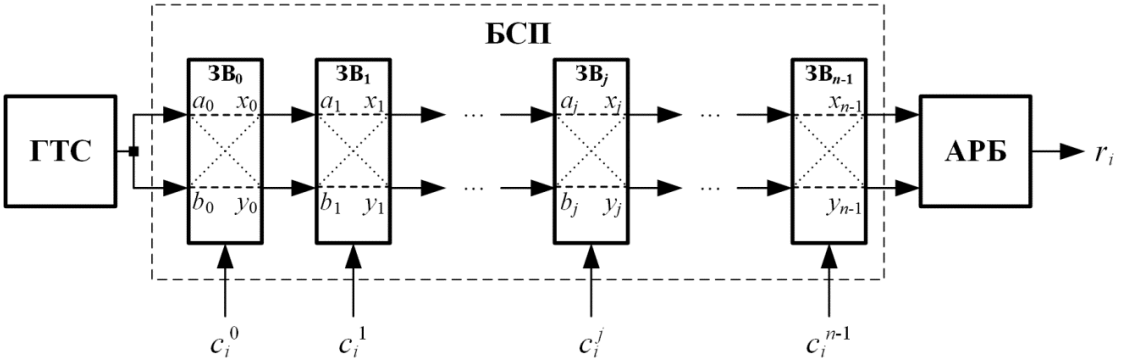


Рис. 1. Обобщенная структура А-ФНФ

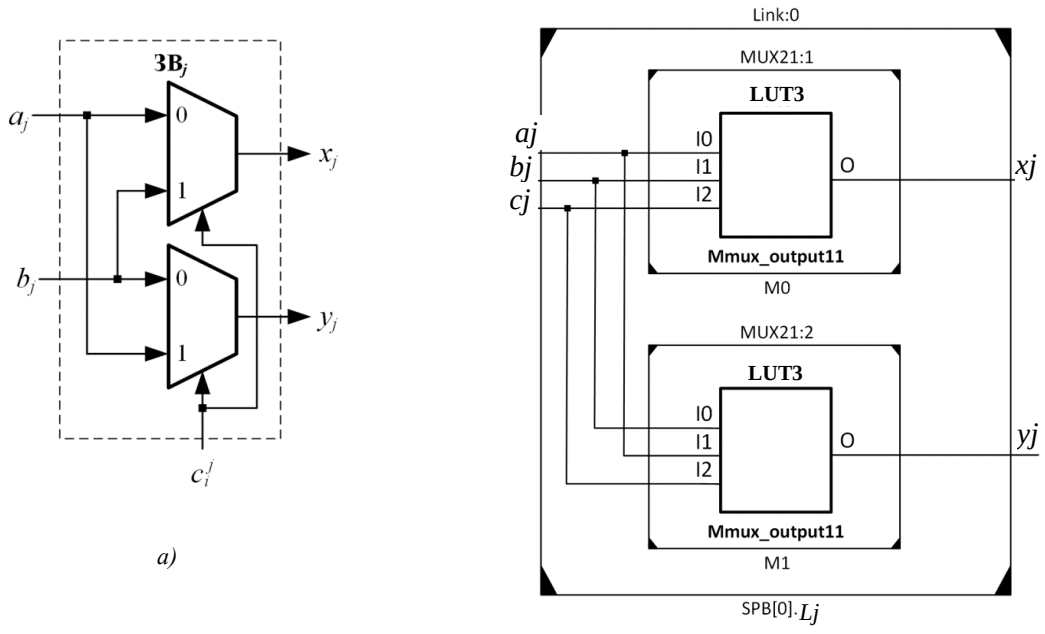
В свою очередь, БСП состоит из n звеньев (ЗВ_j), управляемых внешними сигналами $c_i^j \in \{0,1\}$, $j = \{0,1,2,...,n-1\}$, значения которых равны значениям соответствующих разрядов

подаваемого запроса c_i. Каждое звено $ЗВ_j$ имеет два входа a_j, b_j и два выхода x_j, y_j. В случае $c_j^i = 0$ происходит передача сигнала со входа a_j на выход x_j и со входа b_j на выход y_j. В противном случае, когда $c_j^i = 1$, передача сигнала осуществляется от входа a_j на выход y_j и от входа b_j на выход x_j. Соответственно, каждое звено имеет две конфигурации: прямой передачи сигналов и перекрестной передачи сигналов. Общее число последовательно соединенных звеньев *n* обеспечивает 2^n различных конфигураций путей прохождения двух копий тестового сигнала от блока ГТС до схемы АРБ. Назначение арбитра заключается в определении, какая из копий сигнала пришла раньше. Например, если сигнал, поступивший с выхода x_{n-1} на вход арбитра, оказался раньше сигнала y_{n-1}, то арбитр выработает на своем выходе значение $r_i = 1/0$. В противном случае арбитр установит на своем выходе значение $r_i = 0/1$.

В большинстве схемотехнических реализаций ГТС вырабатывает тестовый сигнал, в котором определяющим для работы всей схемы является его передний фронт [1]. В этом случае схема АРБ синтезируется с применением синхронного триггера D-типа, для которого на вход синхронизации поступает сигнал с выхода x_{n-1}, а на вход данных – с выхода y_{n-1} последнего звена. Кроме этого, некоторые разработчики применяют в качестве арбитра триггерные схемы с асинхронным сбросом и переустановкой, а также мультитриггерные схемы [5]. Основной проблемой применения перечисленных схемотехнических решений для синтеза схем АРБ является эффект метастабильности, негативно влияющий на стабильность всей схемы А-ФНФ. В работе [5] была предложена схема АРБ, позволяющая улучшить стабильность арбитра за счет обнаружения состояния метастабильности с дальнейшим его кодированием стабильным двухбитным двоичным значением.

При синтезе схемы сильной А-ФНФ с параметром $n \geq 8$ основные аппаратные затраты приходятся на БСП. Рассмотрим подробнее реализацию звеньев БСП.

Базовым подходом для реализации одного звена является схема, состоящая из двух мультиплексоров (рис. 2, *а*). Реализация данной схемы на ПЛИС типа FPGA будет выполнена с применением двух блоков LUT3, реализующих комбинационные схемы мультиплексоров с тремя входами.



а) *б)*

Рис. 2. Схемная реализация одного звена БСП: *а)* RTL-схема; *б)* технологическая схема

В свою очередь, блок LUT3 является моделью технологического элемента, позволяющей оценить реализацию комбинационной схемы на блоках LUT реального кристалла FPGA. Серийно выпускаемые кристаллы FPGA такого производителя, как Xilinx, имеют аппаратную структуру, способную реализовывать переключательные функции максимум от четырех либо шести аргументов в зависимости от архитектуры ПЛИС [11]. Структурно LUT-блоки состоят из памяти конфигурации и набора мультиплексоров, обеспечивающих трансляцию выбранного значения из этой памяти на свой единственный выход. Значение адреса памяти формируется из значений используемых управляемых входов мультиплексоров. Неиспользуемые входы при реализации малого числа аргументов принимают, как правило, константное значение 0.

На рис. 3 изображена структурная схема блока LUT FPGA фирмы Xilinx серии Spartan-3E [12], сконфигурированного для реализации одного мультиплексора с выходом x_j из схемы ЗВ_j (см. рис. 2, *а*).

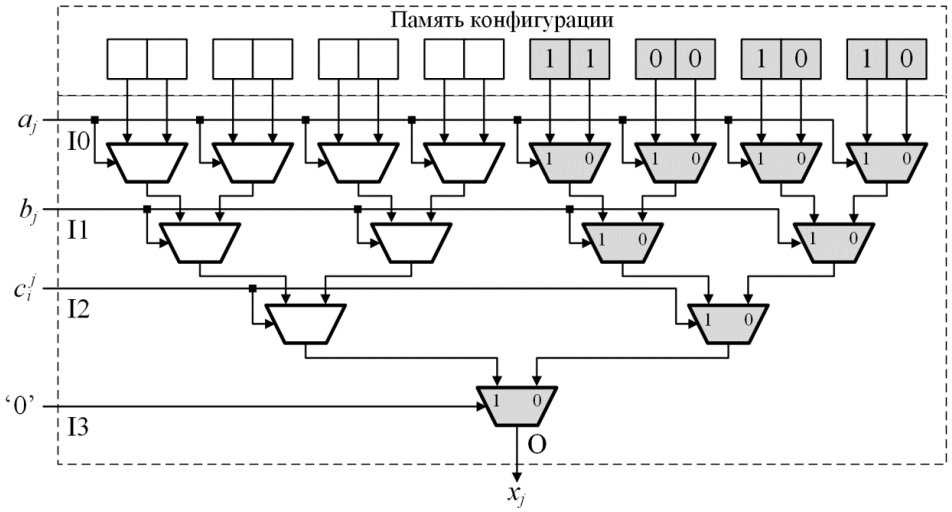


Рис. 3. Схемная реализация мультиплексора одного звена БСП ресурсами LUT-блока

Видно, что ресурс LUT-блока использован лишь наполовину. При больших значениях параметра *n* это может привести к существенным затратам на реализацию схемы А-ФНФ. Например, для $n=128$ реализация БСП для FPGA Spartan-3E будет использовать 256 LUT-блоков, которые составляют более 26 % от всех доступных ресурсов такого кристалла, как XC3S100E [12].

Предлагаемая архитектура симметричных путей. При реализации классической схемы БСП на FPGA каждый LUT-блок обладает уникальными параметрами, в том числе и временем срабатывания внутренних мультиплексоров, которые обеспечивают трансляцию выбранного сигнала на единственный выход.

Обозначим время распространения фронта тестового сигнала от входа a_j до выхода x_j как $\delta(x_j, a_j)$ для $c_i^j = 0$, а время распространения фронта тестового сигнала от входа b_j до выхода x_j как $\delta(x_j, b_j)$ для $c_i^j = 1$. Соответствующим образом введем обозначения для второго мультиплексора звена ЗВ_j: $\delta(y_j, a_j)$ для $c_i^j = 1$, $\delta(y_j, b_j)$ для $c_i^j = 0$. Для оценки перечисленных параметров воспользуемся параметрической моделью звена ЗВ_j, восстановленной после технологического синтеза для FPGA XC3S100E. Так, для звена ЗВ_0 параметры имеют следующие значения: $\delta(x_0, a_0) = 1{,}488$ нс, $\delta(x_0, b_0) = 1{,}445$ нс, $\delta(y_0, a_0) = 1{,}397$ нс, $\delta(y_0, b_0) = 1{,}465$ нс. Для смежного звена ЗВ_1 данные параметры будут принимать уже другие значения: $\delta(x_0, a_0) = 0{,}963$ нс, $\delta(x_0, b_0) = 1{,}016$ нс, $\delta(y_0, a_0) = 0{,}986$ нс, $\delta(y_0, b_0) = 0{,}993$ нс. Связано это в первую очередь с уникальностью самих LUT-блоков и с асимметрией конфигурируемых связей, их

соединяющих. Уникальные значения приведенных параметров позволяют реализовать вторую копию звена на свободных ресурсах одного LUT-блока. При этом ранее не использованный вход (I3 на рис. 3) будет применен для выбора первой либо второй копии звена. На рис. 4 показана новая структура звена БСП и схемотехническая реализация его верхней части на двух LUT-блоках с четырьмя входами. Представленная схема обеспечивает четыре конфигурации соединения входов a_j, b_j с выходами x_j, y_j: два прямых соединения при $c_i^j = 0$, $c_i^{j+1} = 0$ и $c_i^j = 1$, $c_i^{j+1} = 1$ и два перекрестных соединения при $c_i^j = 1$, $c_i^{j+1} = 0$ и $c_i^j = 0$, $c_i^{j+1} = 1$.

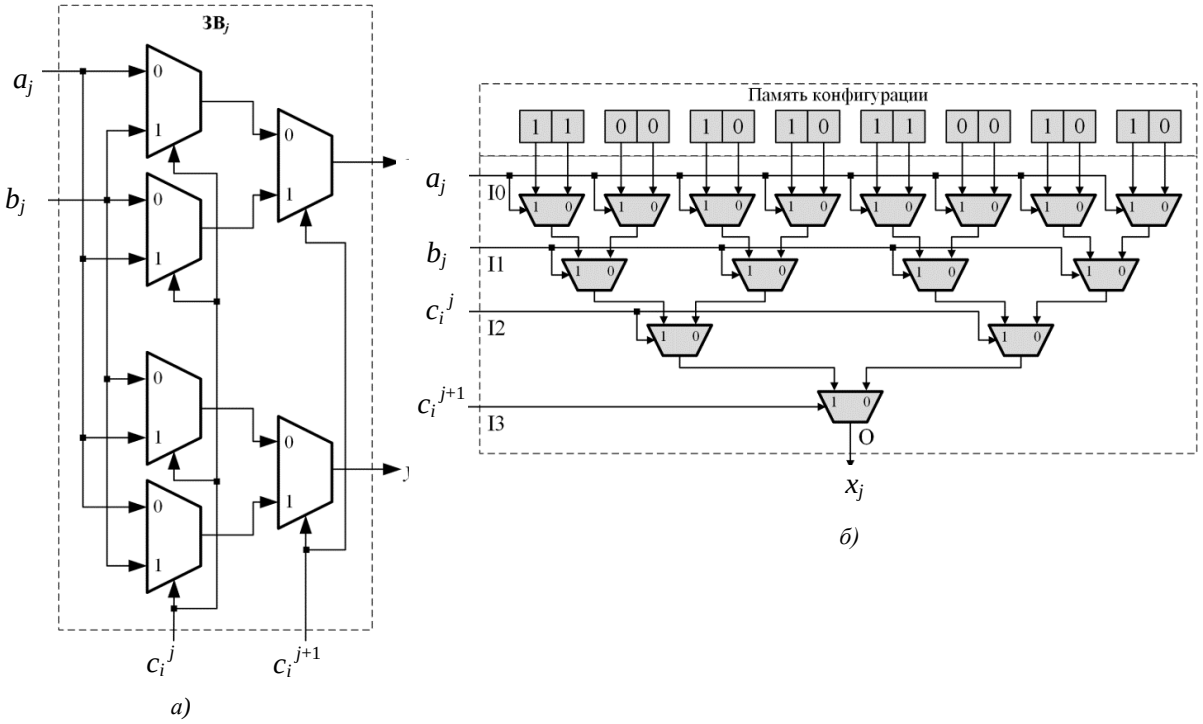


Рис. 4. Предлагаемая структура одного звена БСП (*а*) и схемная реализация его части на блоке LUT4 (*б*)

В табл. 1 представлены значения времени распространения фронта тестового сигнала от входов до выходов предложенной схемы в зависимости от значений c_i^j и c_i^{j+1} для двух различных звеньев одного БСП, полученные путем параметрического моделирования А-ФНФ с параметром $n = 16$ для FPGA SPARTAN-3E XC3S100E.

Таблица 1

Значения задержки распространения сигнала для двух различных звеньев БСП

Биты запроса, c_i^j c_i^{j+1}	Тип задержки	Значение задержки для звена ЗВ$_j$, нс	
		ЗВ$_2$	ЗВ$_1$
00	$\delta(x_j, a_j)$	0,950	1,394
	$\delta(y_j, b_j)$	0,861	0,992
01	$\delta(x_j, b_j)$	0,992	0,861
	$\delta(y_j, a_j)$	0,859	1,326
10	$\delta'(x_j, b_j)$	2,002	1,218
	$\delta'(y_j, a_j)$	0,191	1,168
11	$\delta'(x_j, a_j)$	0,060	1,037
	$\delta'(y_j, b_j)$	1,911	1,150

Представленные результаты свидетельствуют о потенциальной возможности использования предложенной архитектуры в качестве А-ФНФ.

Анализ аппаратных затрат. Как было показано ранее, предложенная структура звена БСП полностью вписывается в архитектуру блоков LUT4, что дает существенную экономию ресурсов FPGA-кристаллов. С учетом того что структурные блоки ГТС и АРБ (см. рис. 1) имеют незначительные затраты на реализацию (ГТС использует три LUT-блока и три триггера, АРБ использует один триггер в реализации, описанной в работе [5]), основная доля аппаратных затрат приходится на БСП. Предложенная архитектура звеньев БСП позволяет в два раза сократить аппаратные затраты в сравнении с классической архитектурой. В табл. 2 приведено сравнение аппаратных затрат при реализации двух схем А-ФНФ для FPGA XC3S100E.

Таблица 2

Аппаратные затраты на реализацию А-ФНФ ($n = 128$)

Название ресурса FPGA	Число использованных блоков		Число имеющихся блоков	Доля затрат, %	
	Классическая А-ФНФ	Предлагаемая А-ФНФ		Классическая А-ФНФ	Предлагаемая А-ФНФ
Slices	131	69	960	13,64	7,18
Flip-Flops	4	4	1920	0,141	0,141
4-input LUTs	259	131	1920	13,48	6,82

Из табл. 2 видно, что предлагаемая архитектура звеньев А-ФНФ позволяет почти в два раза сократить затраты на реализацию всей схемы и может быть реализована на FPGA с LUT-блоками, у которых число входов больше четырех [11].

Анализ основных характеристик ФНФ типа арбитр. Было проведено сравнение двух подходов к реализации А-ФНФ для FPGA XC3S100E с применением САПР Xilinx ISE 14.7 [13]. Для этого использовались две параметрические модели реализованных схем А-ФНФ с параметром $n=16$. В качестве схемы АРБ применялась схема синхронного D-триггера (технологический элемент FDC). Сами схемы А-ФНФ и тестовые модули к ним были описаны на языке Verilog. Тестовые модули, представляющие собой testbench-компоненты, осуществляли подачу всех возможных 2^n запросов на входы, генерирование тестового сигнала и анализ ответов схемы А-ФНФ. Кроме этого, тестовые модули осуществляли анализ временной разницы $\Delta(x_{n-1}, y_{n-1})$ между фронтами двух копий тестового сигнала, приходящих от выходов x_{n-1} и y_{n-1} на входы схемы АРБ.

На рис. 5 изображены три графика отсортированных по возрастанию значений временных различий, наблюдаемых на входах схемы АРБ, для трех различных реализаций: А-ФНФ-16 – классической схемы с числом звеньев $n=16$; А-ФНФ-16-Н и А-ФНФ-32-Н – схем с предложенной архитектурой звеньев $n=16$ и $n=32$ соответственно. При этом на оси абсцисс представлены не сами значения запросов, а их порядковые индексы C_{index}. Ввиду сложности генерирования всех возможных запросов для схемы А-ФНФ-32-Н, как и для остальных схем, были сгенерированы 2^{16} запросов. С целью обеспечения равномерности распределения этих запросов на множестве всех 2^{32} возможных значений использовался метод генерирования псевдослучайных М-последовательностей на основе 32-разрядной схемы LFSR. Отрицательные значения $\Delta(x_{n-1}, y_{n-1})$ на графиках означают, что фронт тестового сигнала с выхода y_{n-1} пришел позже, чем с выхода x_{n-1}.

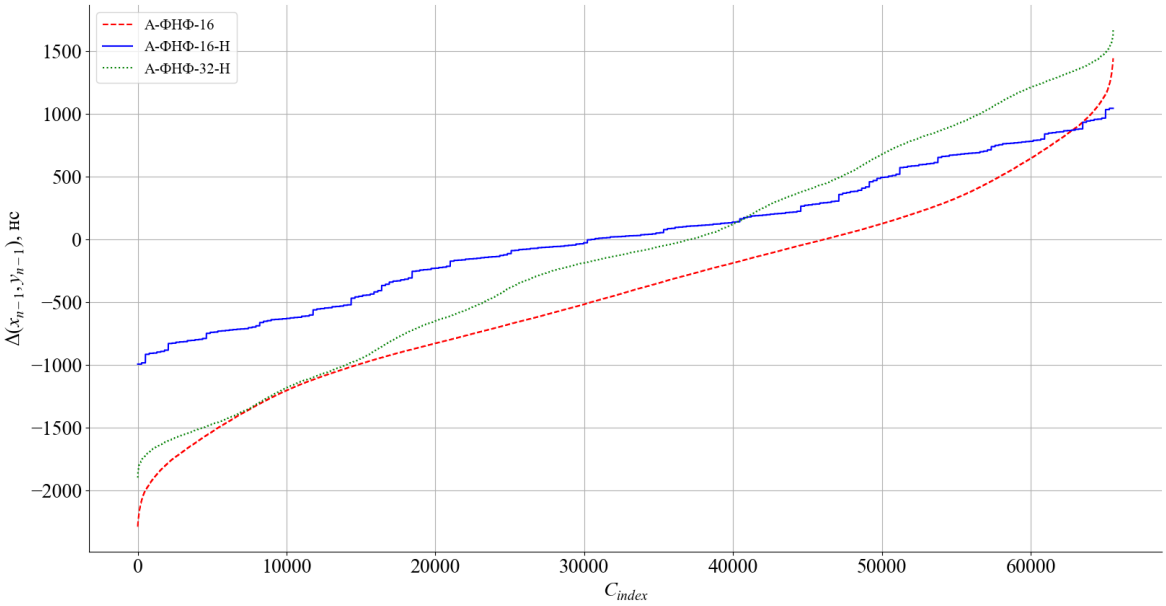
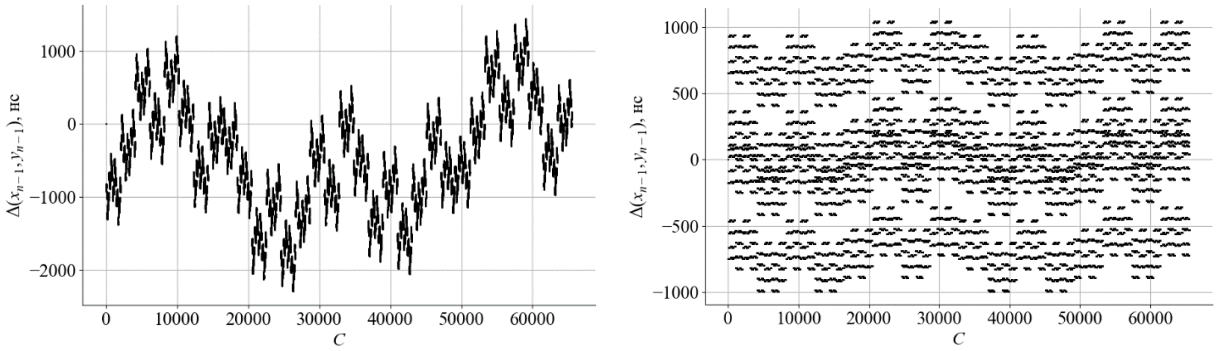


Рис. 5. Графики значений $\Delta(x_{n-1}, y_{n-1})$ для трех различных реализаций А-ФНФ

На рис. 5 видно, что графики значений $\Delta(x_{n-1}, y_{n-1})$ для А-ФНФ-16-Н и А-ФНФ-32-Н являются более симметричными относительно двух координатных осей, а их нелинейная форма усложняет зависимость между парами запрос-ответ в сравнении с классической схемой А-ФНФ-16. Асимметричность графика А-ФНФ-16 говорит о несбалансированности множества всех ответов, при которой вероятность появления нулевого ответа будет гораздо больше вероятности появления единичного ответа. Кроме этого, уменьшился диапазон результирующего значения $\Delta(x_{n-1}, y_{n-1})$. Так, для классической схемы А-ФНФ-16 этот диапазон равен [–2289; 1442] нс, а для предложенной архитектуры при том же параметре n он уменьшился и составляет [–994; 1043] нс. Обусловлено это в первую очередь фактическим уменьшением длин симметричных путей (числа LUT-блоков), что подтверждается оцененным диапазоном для схемы А-ФНФ-32-Н, равным [–1898; 1668] нс. Для каждой схемы также было оценено число ответов, при которых схема АРБ переходит в метастабильное состояние. Так, для схемы А-ФНФ-16 был получен 1241 метастабильный ответ, что составляет 1,89 % от всех зарегистрированных ответов. В то же время для схем А-ФНФ-16-Н и А-ФНФ-32-Н это значение равно 1,01 и 0,39 % соответственно, что является потенциальным показателем большей стабильности. Кроме того, в ходе экспериментов был оценен показатель внутрикристальной уникальности схем А-ФНФ-16 и А-ФНФ-16-Н путем реализации восьми идентичных компонентов на одном кристалле и сравнения множеств ответов при подаче одного множества различных запросов. Под внутрикристальной уникальностью понимается численный показатель, методика вычисления которого представлена в работе [6]. Этот показатель принимает значения в диапазоне [0;1] и характеризует степень различия множества ответов от реализованного на одном кристалле множества идентичных схем А-ФНФ при подаче на них одинаковых запросов. Значение уникальности, равное 0, означает, что все ответы являются идентичными. Значение показателя, равное 1, говорит о том, что все ответы от тестируемых схем А-ФНФ являются уникальными.

В итоге для схемы А-ФНФ-16 данный показатель уникальности равен 0,489, а для схемы А-ФНФ-16-Н равен 0,473, что свидетельствует о потенциальном применении данных схем для реализации уникальных неклонируемых идентификаторов. Реальные показатели стабильности и уникальности сильно зависят от параметра n, от схемы блока АРБ и могут быть оценены только на реальной аппаратуре [10] при многократной подаче одних и тех же запросов.

На рис. 6 показаны графики функциональной зависимости значений $\Delta(x_{n-1}, y_{n-1})$ от значений запросов, линейно упорядоченных по значению C от 0 до 65 535, для схем А-ФНФ-16 и А-ФНФ-16-Н.



а) *б)*

Рис. 6. Графики функциональной зависимости значений $\Delta(x_{n-1}, y_{n-1})$ от значений запросов:
а) для схемы А-ФНФ-16; *б)* для схемы А-ФНФ-16-Н

Как видно из представленных графиков, схема А-ФНФ-16-Н обладает большей случайностью и меньшей корреляционной зависимостью значений $\Delta(x_{n-1}, y_{n-1})$ от значений запросов C, что потенциально может усложнить построение точной математической модели схемы А-ФНФ злоумышленниками [4].

Заключение. В статье предложена новая архитектура звеньев блока симметричных путей для схемотехнической реализации физически неклонируемой функции типа арбитр на программируемых логических интегральных схемах типа FPGA. Показано, что за счет конфигурации встроенных LUT-блоков можно практически в два раза снизить аппаратные затраты на реализацию и при этом значительно улучшить качественные характеристики реализуемой ФНФ. Результаты описанных в статье экспериментов были получены с применением САПР Xilinx ISE 14.7 [13] и языка проектирования цифровой аппаратуры Verilog. Полученные результаты нуждаются в верификации на реальной аппаратуре с целью установления истинных показателей межкристальной уникальности, случайности, стабильности и возможности построения математической модели А-ФНФ методами машинного обучения. Предложенная схемная реализация также может быть применена для проектирования конфигурируемых ФНФ.

Список использованных источников

1. Design and implementation of high-quality physical unclonable functions for hardware-oriented cryptography / S. S. Zalivaka [et al.] // Secure System Design and Trustable Computing. – Switzerland : Springer, 2016. – P. 39–81.

2. Ярмолик, В. Н. Физически неклонируемые функции / В. Н. Ярмолик, Ю. Г. Вашинко // Информатика. – 2011. – № 2(30). – С. 92–103.

3. Иванюк, А. А. Проектирование встраиваемых цифровых устройств и систем / А. А. Иванюк. – Минск : Бестпринт, 2012. – 337 с.

4. Zalivaka, S. S. Reliable and modeling attack resistant authentication of arbiter PUF in FPGA implementation with trinary quadruple response / S. S. Zalivaka, A. A. Ivaniuk, Ch.-H. Chang // IEEE Transactions on Information Forensics and Security. – 2018. – Vol. 4, no. 14. – P. 1109–1123.

5. Multi-valued arbiters for quality enhancement of PUF responses on FPGA implementation / S. S. Zalivaka [et al.] // Proc. IEEE/ACM Asia and South Pacific Design Automation Conf. – Macau, 2016. – P. 533–538.

6. Quantitative and statistical performance evaluation of arbiter physical unclonable functions on FPGAs / Y. Hori [et al.] // Proc. Intern. Conf. "Reconfigurable Computing and FPGAs". – Mexico, 2010. – P. 298–303.

7. Becker, G. T. On the pitfalls of using Arbiter-PUFs as building blocks / G. T. Becker // IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems. – 2015. – Vol. 34, no. 8. – P. 1295–1307.

8. A technique to build a secret key in integrated circuits for identification and authentication applications / J. W. Lee [et al.] // Proc. of the IEEE VLSI Circuits Symp. (VLSI'04). – Honolulu, 2004. – P. 176–179.

9. Morozov, S. An analysis of delay based PUF implementations on FPGA / S. Morozov, A. Maiti, P. Schaumont // Proc. Intern. Symp. "Applied Reconfigurable Computing". – Berlin, 2010. – P. 382–387.

10. Nexys 4 Artix-7 FPGA Trainer Board [Electronic resource]. – Mode of access: https:// store. digilentinc.com/nexys-4-artix-7-fpga-trainer-board-limited-time-see-nexys4-ddr. – Date of access: 20.11.2018.

11. 7 Series FPGAs Data Sheet: Overview [Electronic resource]. – Mode of access: https:// www.xilinx.com/ support/documentation/data_sheets/ds180_7Series_Overview.pdf. – Date of access: 20.11.2018.

12. Spartan-3E FPGA Family Data Sheety [Electronic resource]. – Mode of access: https:// www.xilinx.com/ support/documentation/data_sheets/ds312.pdf. – Date of access: 28.12.2018.

13. ISE Design Suite [Electronic resource]. – Mode of access: https://www.xilinx.com/products/ design-tools/ise-design-suite.html. – Date of access: 20.11.2018.

References

1. Zalivaka S. S., Zhang L., Klybik V. P., Ivaniuk A. A., Chang C.-H. Design and implementation of high-quality physical unclonable functions for hardware-oriented cryptography. *Secure System Design and Trustable Computing*. Switzerland, Springer, 2016, pp. 39–81. DOI: 10.1007/978-3-319-14971-4

2. Yarmolik V. N., Vashinko Y. G. Fizicheski nekloniruemye funkcii [Physically unclonable functions]. Informatika [*Informatics*], 2011, no. 2(30), pp. 92–103 (in Russian).

3. Ivaniuk A. A. Projektirovanie vstraivaemyh cifrovyh ustrojstv i system. *Design of Embedded Digital Devices and Systems*. Minsk, Bestprint, 2012, 337 p. (in Russian).

4. Zalivaka S. S., Ivaniuk A. A., Chang Ch.-H. Reliable and modeling attack resistant authentication of arbiter PUF in FPGA implementation with trinary quadruple response. *IEEE Transactions on Information Forensics and Security*, 2018, vol. 4, no. 14, pp. 1109–1123. DOI: 10.1109/TIFS.2018.2870835

5. Zalivaka S. S., Puchkov A. V., Klybik V. P., Ivaniuk A. A., Chang C.-H. Multi-valued arbiters for quality enhancement of PUF responses on FPGA implementation. *Proceedings IEEE/ACM Asia and South Pacific Design Automation Conference*. Macau, 2016, pp. 533–538. DOI: 10.1109/ASPDAC.2016.7428066

6. Hori Y., Yoshida T., Katashita T., Satoh A. Quantitative and statistical performance evaluation of arbiter physical unclonable functions on FPGAs. *Proceedings International Conference "Reconfigurable Computing and FPGAs"*. Mexico, 2010, pp. 298–303. DOI: 10.1109/ReConFig.2010.24

7. Becker G. T. On the pitfalls of using Arbiter-PUFs as building blocks. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2015, vol. 34, no. 8, pp. 1295–1307. DOI: 10.1109/TCAD.2015.2427259

8. Lee J. W., Gassend B., Lim D., Suh G. E. A technique to build a secret key in integrated circuits for identification and authentication applications. *Proceedings of the IEEE VLSI Circuits Symposium (VLSI'04)*. Honolulu, 2004, pp. 176–179. DOI: 10.1109/VLSIC.2004.1346548

9. Morozov S., Maiti A., Schaumont P. An analysis of delay based PUF implementations on FPGA. *Proceedings International Symposium "Applied Reconfigurable Computing"*. Berlin, 2010, pp. 382–387. DOI: 10.1007/978-3-642-12133-3_37

10. Nexys 4 Artix-7 FPGA Trainer Board. Available at: https:// store.digilentinc.com/nexys-4-artix-7-fpga-trainer-board-limited-time-see-nexys4-ddr (accessed 20.11.2018).

11. 7 Series FPGAs Data Sheet: Overview. Available at: https:// www.xilinx.com/support/documentation/ data_sheets/ds180_7Series_Overview.pdf (accessed 20.11.2018).

12. Spartan-3E FPGA Family Data Sheety. Available at: https:// www.xilinx.com/support/documentation/ data_sheets/ds312.pdf (accessed 28.12.2018).

13. ISE Design Suite. Available at: https://www.xilinx.com/products/design-tools/ise-design-suite.html (accessed 20.11.2018).

Информация об авторе

Иванюк Александр Александрович, доктор технических наук, профессор кафедры информатики, Белорусский государственный университет информатики и радиоэлектроники, Минск, Беларусь.
E-mail: ivaniuk@bsuir.by

Information about the author

Alexander A. Ivaniuk, Dr. Sci. (Eng.), Professor Computer Science Department, Belarusian State University of Informatics and Radioelectronics, Minsk, Belarus.
E-mail: ivaniuk@bsuir.by

ISSN 1816-0301 (Print)
ISSN 2617-6963 (Online)

БИОИНФОРМАТИКА

BIOINFORMATICS

УДК 57.087.1

Поступила в редакцию 30.08.2018
Received 30.08.2018

Принята к публикации 14.02.2019
Accepted 14.02.2019

Разработка алгоритмов и программных средств классификации кодирующих и некодирующих нуклеотидных последовательностей

В. Р. Закирова[1]✉, Д. А. Сырокваш[1], С. В. Гилевский[1], П. В. Назаров[2], Н. Н. Яцков[1]

[1]*Белорусский государственный университет, Минск, Беларусь*
✉*E-mail: veranika.zakirava@gmail.com*
[2]*Люксембургский институт здоровья, Штрассен, Люксембург*

Аннотация. Проведено исследование кодирующих и некодирующих нуклеотидных последовательностей референсного генома человека. Разработаны семь моделей векторизации нуклеотидных последовательностей на основе частот моно-, би- и триграммов нуклеотидов, параметров модели частот и позиций сочетаний нуклеотидов (category-position-frequency model), длин последовательностей, корреляционных факторов нуклеотидов, статистических признаков кодирующих и некодирующих участков молекул ДНК. Определены наиболее информативные признаки моделей векторизации с использованием алгоритмов автоматического выбора признаков и классификации на основе методов случайного леса и опорных векторов. Установлено различие кодирующих и некодирующих фрагментов нуклеотидных последовательностей. Ошибка классификации последовательностей с использованием метода случайного леса на наборе из 23 наиболее информативных признаков составила 2,93 %.

Ключевые слова: ДНК, экзон, интрон, классификация, метод случайного леса, метод опорных векторов, алгоритмы автоматического отбора информативных признаков, программирование на языке *R*

Для цитирования. Разработка алгоритмов и программных средств классификации кодирующих и некодирующих нуклеотидных последовательностей / В. Р. Закирова [и др.] // Информатика. – 2019. – Т. 16, № 2. – С. 109–118.

Development of algorithms and software for classification of nucleotide sequences

Veranika R. Zakirava[1]✉, Dzmitry A. Syrakvash[1], Stanislau V. Hileuski[1], Petr V. Nazarov[2], Mikalai M. Yatskou[1]

[1]*Belarusian State University, Minsk, Belarus*
✉*E-mail: veranika.zakirava@gmail.com*
[2]*Luxembourg Institute of Health, Strassen, Luxembourg*

Abstract. Coding and non-coding nucleotide sequences of the human reference genome have been investigated. Seven models of vectorization of nucleotide sequences based on mono-, bi-, trigram nucleotide frequencies, parameters of the category-position-frequency model, the lengths of sequences, nucleotide correlation factors,

statistical features of coding and non-coding regions of DNA molecules were developed. The most informative features of vectorization models were determined using feature selection and classification algorithms based on the random forests and support vector machine methods. The difference between coding and non-coding fragments of nucleotide sequences was established. An error of the coding and non-coding sequences classification using the random forests method on a set of the 23 most informative features is 2,93 %.

Keywords: DNA, exon, intron, classification, Random Forests, Support Vector Machine, feature selection, *R* programming

For citation. Zakirava V. R., Syrakvash D. A., Hileuski S. V., Nazarov P. V., Yatskou M. M. Development of algorithms and software for classification of nucleotide sequences. *Informatics*, 2019, vol. 16, no. 2, pp. 109–118 (in Russian).

Введение. Появление новых технологий секвенирования и инструментов для точечного манипулирования структурой ДНК позволяет на генетическом уровне подавлять болезни, повышать устойчивость организмов к неблагоприятным условиям среды и продлевать продолжительность их жизни [1]. В данном контексте определение предназначения генов и их кодирующих и некодирующих участков, экзонов и интронов является одной из первоочередных задач.

Важным этапом обработки нуклеотидных последовательностей является формирование вектора признаков. Существующие модели формирования вектора признаков нуклеотидных последовательностей [2–7] имеют ряд ограничений: они неуниверсальны, в основном предназначены для решения специализированных задач, разработаны для анализа выборок небольшого объема, не включают алгоритмы автоматического выбора признаков [8], что существенно снижает как вычислительную эффективность алгоритмов, так и точность классификации последовательностей вследствие наличия избыточных и неинформативных признаков. Например, исследование способов векторизации нуклеотидных последовательностей для классификации экзонов и интронов представлено в работе [3], однако проведено лишь приближенное сравнение средних значений характеристик классификации на малом объеме данных (менее 10 000 последовательностей) без учета информативности признаков нуклеотидных последовательностей. Перспективным направлением повышения эффективности и точности классификации нуклеотидных последовательностей является отбор их наиболее информативных признаков.

Цель исследования заключается в разработке статистического подхода и программного пакета для классификации кодирующих и некодирующих нуклеотидных последовательностей геномных данных с учетом отбора наиболее информативных признаков нуклеотидных последовательностей. В качестве исходных данных используются опубликованные файлы референсного генома человека [9].

Экспериментальные данные. Рассмотрены нуклеотидные последовательности генома человека [9]. Файлы *gencode.v1.annotation.gtf* и *GRCh38.genome.fa* содержат информацию о биологических последовательностях.

Файл *gencode.v1.annotation.gtf* (*GeneralTransferFormat*) имеет размер 1,13 ГБ и включает:

− имя последовательности;
− источник данных или название программы – генератора данных;
− тип последовательности;
− позицию начала последовательности в *FASTA*-файле;
− позицию конца последовательности в *FASTA*-файле;
− направление прочтения последовательности;
− позицию начала первого кодона;
− дополнительные признаки [9].

Файл *GRCh38.genome.fa* (формат *FASTA*) имеет размер 2,98 ГБ и содержит восстановленный геном человека. Данные представлены парами строк. В первой строке за символом > следует название последовательности. Во второй строке последовательность посимвольно описывается. В файле представлены пять закодированных символов: *A, T, G, C*, соответствующих нуклеотидам аденину, тимину, гуанину, цитозину, а также символ *N*, обозначающий неопределенные нуклеотиды [9].

Исходные данные разделены на 24 файла, характеризующие 22 аутосомы, *X*- и *Y*-хромосомы. Объем файлов данных – 6,13 ГБ; общее количество последовательностей – 2 127 864, из них 1 162 077 экзонов и 965 787 интронов. Ввиду ограничений вычислительных ресурсов анализ данных для каждой из хромосом проводился отдельно.

Рассмотрен набор данных хромосом 1, 4, 7 и 10, включающий более 456 324 последовательностей. Часть данных использовалась в качестве эталонной выборки объемом 1 000 или 100 000 последовательностей, часть являлась тестируемой выборкой объемом 1 000, 10 000 или более последовательностей.

Разработка алгоритмов и программных средств. *Статистический подход для классификации кодирующих и некодирующих нуклеотидных последовательностей*. Разработанный статистический подход для классификации кодирующих и некодирующих последовательностей реализуется в три этапа: предварительная обработка данных, выбор оптимальной модели векторизации, отбор значимых признаков и формирование на их основе модели классификации (рис. 1).

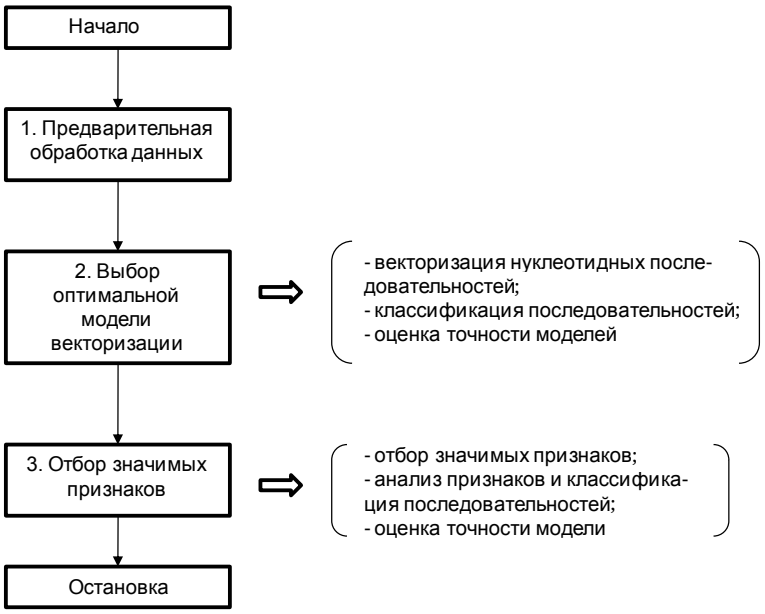


Рис. 1. Общая схема статистического подхода

На этапе предварительной обработки данных осуществляется разбиение исходной последовательности на неперекрывающиеся интронные и экзонные участки, результатом работы является *.csv*-файл со списком обнаруженных последовательностей и указанием их типов.

В ходе второго этапа производится векторизация нуклеотидных последовательностей интронов и экзонов с целью последующего применения алгоритмов классификации. Векторизация подразумевает переход от символьных последовательностей к набору векторов или признаков, характеризующих нуклеотидные последовательности. Рассмотрены следующие модели формирования вектора признаков:

модель 1 – частотная модель, включающая в качестве признаков частоты моно- и биграммов нуклеотидов (20 признаков);

модель 2 – частотная модель с использованием частот триграммов нуклеотидов (64 признака);

модель 3 – модель на основе частот и позиций сочетаний нуклеотидов *CPF* (*category-position-frequency*) [2] (12 признаков);

модель 4 – модификация модели *CPF* с уменьшенным числом компонентов (8 признаков);

модель 5 – модель на основе общих статистических признаков, таких как длина последовательности, частоты нуклеотидов и корреляционные факторы нуклеотидов [10] (13 признаков);

модели 6 и 7 – две модели на основе статистических признаков экзонов и интронов (9 и 12 признаков). Модель 7 включает набор признаков модели 6, а также длину последовательности и флаги начала позиций с биграммов *CT* и *GT*.

Полный набор признаков семи моделей векторизации последовательностей – 110.

Модели 1 и 2 рассматриваются в качестве базовых, наиболее цитируемых в литературе. Модели 3 и 4 являются перспективными, так как используются для решения аналогичных задач в работе [2]. В данных моделях устранен ключевой недостаток моделей 1 и 2, а именно отсутствие в векторе признаков информации о положении и порядке символов в последовательности.

Модели 5–7 на основе статистических признаков четко ориентированы на решение конкретных задач, они популярны ввиду высокой точности получаемых классификаторов.

В качестве общих признаков моделей выбраны [10]:

– длина последовательности;

– частоты нуклеотидов *A*, *T*, *G*, *C*;

– частоты биграммов *AT* и *GC*;

– корреляционные факторы нуклеотидов.

В качестве специальных признаков для решения проблемы классификации экзонов и интронов выбраны [11]:

– логарифм длины последовательности (логарифмическое преобразование позволяет устранить эффект чрезмерного влияния признака с большой вариацией);

– частоты биграммов *TA* и *CG*;

– частоты триграммов *AAA* и *TTT*;

– флаги начала последовательности с триграммов *CTA*, *CTG*, *GTA*, *GTG*.

На втором этапе осуществляется выбор наиболее оптимальных моделей векторизации и классификатора. В качестве алгоритмов классификации рассмотрены наиболее популярные: метод случайного леса [12–14] и метод опорных векторов с радиальной базисной функцией в качестве ядра [15, 16]. Осуществляется грубая классификация нуклеотидных последовательностей с использованием моделей 1–7.

На третьем этапе статистического подхода производится отбор наиболее информативных признаков моделей 1–7 для точной классификации экзонных и интронных последовательностей. В качестве алгоритмов автоматического выбора наиболее информативных признаков экзонов и интронов используются фильтрующие, оберточные и встроенные алгоритмы.

Оценка качества классификации нуклеотидных последовательностей. Оценкой качества классификатора служит уровень допущенных ошибок *ER* (*error rate*):

$$ER = \frac{N_{12} + N_{21}}{N_{11} + N_{12} + N_{21} + N_{22}} \times 100\,\%,$$

где N_{ij} – число последовательностей типа i (кодирующих/некодирующих), распознанных как последовательность типа j (некодирующих/кодирующих). Ошибка классификации оценивалась по тестируемой выборке. Данную оценку целесообразно использовать в ходе анализа выборок, содержащих сопоставимое количество объектов каждого из классов (экзонов и интронов).

Программная реализация алгоритмов анализа данных. В качестве платформы для расчета признаков нуклеотидных последовательностей выбран язык *R* [17], который является языком высокого уровня с открытым исходным кодом для решения статистических задач. Важнейшие достоинства языка *R* – открытость и простота изучения, к недостаткам можно отнести низкую производительность сложных алгоритмов в силу особенностей языка [17]. Для ускорения программных кодов на отдельных этапах анализа данных используется язык программирования *C++*. Ключевым пакетом языка *R* для обработки больших массивов данных является пакет *data.table*, который включает методы быстрой загрузки, формирования выборок и фильтрации данных. Язык *R* обладает довольно ограниченными возможностями при работе с символьными

строками. На языке *C++* реализован алгоритм подсчета количества вхождений подстроки в строку. Для интеграции программных кодов *C++* в язык *R* использовался пакет *Rcpp*. С целью повышения производительности вычислений модели 2 и 3 реализованы на языке *C++*, что дает почти 100-кратное увеличение скорости вычислений в сравнении с *R*-реализациями.

Разработана программа для загрузки и преобразования данных в удобный для дальнейшей работы формат. В качестве параметров принимаются пути к файлам *.gtf* и *.fasta*, название требуемой хромосомы и имя выходного файла. Результатом работы программы является *csv*-файл со списком обнаруженных последовательностей и указанием их типов (экзон или интрон).

Для реализации классификатора на основе метода случайного леса выбран *R*-пакет *randomForest*, содержащий оригинальный алгоритм автора метода Лео Бреймана (Leo Breiman). Особенностью данной реализации является возможность использования алгоритма для решения задач классификации и регрессии, а также гибкой подстройки внутренних параметров алгоритма и оценки значимости входных параметров с помощью индекса Джини (Gini).

Для реализации классификатора на основе нелинейного метода опорных векторов с радиальной базисной функцией в качестве ядра использовался *R*-пакет *e1071*.

Фильтрующие методы. Фильтрующие методы обрабатывают статистические признаки исследуемого набора данных и анализируют каждый признак независимо от остального набора. Основными достоинствами методов являются быстродействие и невысокие требования к производительности вычислительных ресурсов [18].

В качестве программной реализации фильтрующего метода отбора признаков рассмотрен метод одномерных классификаторов *SBF* (*selection by filtering*) *R*-пакета *caret*. Метод строит одномерные линейные классификаторы для каждого из признаков и вычисляет *p*-значение критерия Фишера для оценки значимости признака. Результатом работы метода *SBF* является набор признаков, ранжированный в соответствии со средними *p*-значениями критериев Фишера, полученными в результате *V*-кратной перекрестной проверки. Метод *SBF* возвращает минимальное значение количества признаков, при котором достигается определенное *p*-значение, например, соответствующее заданной точности классификации.

Оберточные методы. Оберточные методы помимо набора тестовых данных требуют информацию об алгоритме классификации. Идея методов заключается в итеративном построении классификаторов на различных подмножествах признаков с использованием результатов классификации в качестве оценки информативности наборов признаков. Методы обладают высокой точностью и избирательностью, позволяют предсказывать оптимальный набор признаков с учетом особенностей классификатора, однако требуют существенных временных затрат. Время их работы нелинейно возрастает в зависимости от количества признаков, что фактически затрудняет применение оберточных методов для анализа данных, характеризующихся большим количеством признаков [18].

В качестве оберточного метода выбран метод рекурсивного удаления признаков *RFE* (*recursive feature elimination*) пакета *caret* [19]. Метод *RFE* строит диаграмму зависимости точности классификатора от количества признаков, позволяя пользователю выбрать необходимый набор признаков, соответствующий заданной точности классификации.

Набор алгоритмов автоматического отбора признаков используется на этапе 3 разработанного статистического подхода. В качестве примера алгоритма из группы встроенных методов рассмотрен метод случайного леса.

Результаты анализа нуклеотидных последовательностей. Вычислительный эксперимент реализуется в три этапа:

1. Предварительная обработка данных. В результате предварительного этапа анализа сформирован *.csv*-файл, список полей которого включает название гена, тип последовательности (экзон или интрон) и символы последовательности.

2. Выбор оптимальной модели векторизации. Вычислены ошибки классификаторов методов случайного леса и опорных векторов при использовании разработанных моделей векторизации (табл. 1). Объемы обучающей и тестируемой выборок составляют 1 000 и 10 000 последовательностей соответственно.

Таблица 1

Уровень ошибки классификации

Модель	Метод случайного леса	Метод опорных векторов
1	18,94	16,32
2	14,60	11,35
3	17,03	15,24
4	17,11	15,29
5	15,55	18,56
6	10,89	9,70
7	8,10	8,38

Оптимизация параметров алгоритмов классификации с помощью пакета *caret* не привела к существенному улучшению результатов.

Можно сделать ряд важных заключений:

1. Наилучшего значения точности (*ER* 8–11 %) удалось достичь при использовании моделей 6 и 7 на основе статистических признаков нуклеотидных последовательностей, в то время как точность моделей 1–5 значительно ниже (*ER* более 11 %).

2. Модель *CPF* (модель 3) действительно содержит избыточные признаки, так как ее точность сопоставима с моделью 4.

3. При использовании модели 5, содержащей длину последовательности в качестве признака, значительно увеличился процент ошибки классификации с применением машины опорных векторов, что обусловлено известной неустойчивостью алгоритма к классификации нестандартизированных данных в условиях высокого шума.

Уровни точности классификации двух методов практически сопоставимы. На третьем этапе анализа данных используется метод случайного леса.

3. Отбор значимых признаков. Исследование значимости отдельных признаков в рамках моделей 1–7 проведено на выборке из 100 000 последовательностей с помощью коэффициента расщепления Джини для классификаторов, построенных с использованием метода случайного леса, показателя *AUC* (*area under curve*, площадь под кривой рабочей характеристики приемника *ROC*) для метода опорных векторов и *p*-значений критериев Фишера для метода *SBF*. В результате сравнительного анализа алгоритмов автоматического отбора признаков исходный набор из 110 признаков моделей 1–7 сокращен до 27 (табл. 2), оценки информативности которых существенно выше, чем у остального набора признаков. Модели 3 и 4 в таблице не представлены, так как они не содержат значимых признаков.

Таблица 2

Наиболее значимые признаки

Мод. 1	Мод. 2	Мод. 5	Мод. 6	Мод. 7
F_{TA}	F_{AAA}	θ_{AT}	*Log*(*Length*)	*Log*(*Length*)
F_{TG}	F_{TAA}	θ_{AG}	*isCTG*	*isCT*
F_{TC}	F_{TAG}	θ_{AC}	*isCTA*	*isGT*
F_{CA}	F_{TTT}	θ_{TG}	*isGTG*	*isCTG*
F_{CG}	F_{TCG}	θ_{TC}	*isGTA*	*isCTA*
–	F_{GTA}	θ_{GC}	–	*isGTG*
–	F_{CGT}	*Length*	–	*isGTA*
–	F_{CCC}	–	–	–

Дальнейшее исследование наборов из 4, 6, 8, 10, 12, 14, 17, 20, 23 и 27 наиболее значимых признаков выполнено на примере обучающей и тестируемой выборок размеров 100 000 и 1 000 соответственно с использованием метода случайного леса и пятикратной перекрестной проверки (рис. 2). Ошибка классификации уменьшается с 6 до 1 % в зависимости от количества при-

знаков. Наименьший уровень ошибки достигается для набора из 23 признаков. Список 23 наиболее информативных признаков представлен в табл. 3.

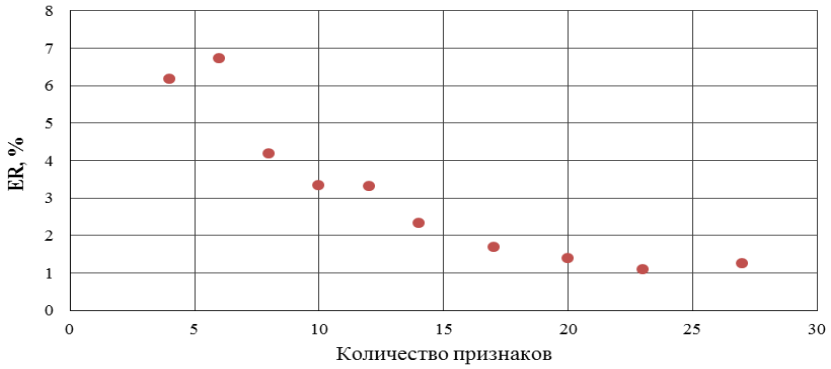


Рис. 2. Зависимость уровня ошибки от количества информативных признаков в результате классификации с использованием метода случайного леса

Признаки длины последовательности *Length* и *Log*(*Length*) наиболее информативные, что очевидно, так как длины интронных и экзонных участков молекул ДНК существенно различаются. В числе наиболее значимых выделены признаки начала последовательности с определенного биграмма (признаки, названия которых начинаются с *is*), корреляционные факторы нуклеотидов, частоты биграммов и триграммов.

Таблица 3

Ранжированный по значениями индекса Джини список признаков

Признак	Ранг *RFE*	Индекс Джини	Признак	Ранг *RFE*	Индекс Джини
Length	1	8574,5	*isGTG*	21	1151,2
Log(*Length*)	2	8274,2	θ_{TG}	5	1141,3
isGT	3	4317,6	F_{CGT}	12	1052,4
isCT	15	3658,2	F_{TG}	14	1048,5
F_{TCG}	10	2082,9	F_{TA}	16	1043,6
F_{TAG}	7	2073,8	F_{TC}	18	1040
isCTG	13	2073	F_{GTA}	9	962,9
θ_{GC}	4	1802,7	θ_{AG}	19	794,9
isGTA	6	1782	F_{CA}	17	794,4
F_{CCC}	8	1761,1	θ_{TC}	23	729,1
F_{CG}	11	1422,6	θ_{AT}	20	652,6
θ_{AC}	22	1340,5	–	–	–

Дополнительно исследованы два набора из 4 и 23 наиболее информативных признаков. Для обучения сформирована выборка размером 100 000 последовательностей, для тестирования сформирован полный набор последовательностей размером 456 324. Ошибки классификации составляют 8,14 и 2,93 % для наборов из 4 и 23 признаков соответственно.

Заключение. В работе предложен статистический подход для классификации кодирующих и некодирующих нуклеотидных последовательностей геномных данных с учетом отбора наиболее информативных признаков нуклеотидных последовательностей. Разработаны и реализованы семь моделей векторизации нуклеотидной последовательности, алгоритмы автоматического выбора признаков, алгоритмы классификации на основе методов случайного леса и опорных векторов.

Проведен анализ экзонных и интронных последовательностей референсного генома человека с использованием разработанных программных средств, по результатам которого выделены 27 информативных признаков моделей векторизации последовательностей.

Выполнено исследование моделей векторизации, алгоритмов классификации и автоматического отбора признаков с целью разделения экзонных и интронных последовательностей на примере анализа аннотированных последовательностей референсного генома человека. Выделены наиболее информативные признаки моделей векторизации последовательностей. Уровень ошибки классификации для наилучшей модели векторизации на основе 23 наиболее значимых признаков составил 2,93 %.

В результате проведенного исследования установлено различие кодирующих и некодирующих фрагментов нуклеотидных последовательностей в референсном геноме человека.

Список использованных источников

1. Edwards, D. J. Beginner's guide to comparative bacterial genome analysis using next-generation sequence data / D. J. Edwards, K. E. Holt // Microbial Informatics and Experimentation. – 2013 – Vol. 3:2. – P. 1–9.

2. Bao, J. An improved alignment-free model for DNA sequence similarity metric / J. Bao, R. Yuan, Z. Bao // BMC Bioinformatics. – 2014. – Vol. 15:321. – P. 1–15.

3. Li, C. Relative entropy of DNA and its application / C. Li, J. Wang // Physica A. – 2005. – Vol. 347. – P. 465–471.

4. Numerical characteristics of word frequencies and their application to dissimilarity measure for sequence comparison / Q. Dai [et al.] // J. of Theoretical Biology. – 2011. – Vol. 276. – P. 174–180.

5. Liu, L. Clustering DNA sequences by feature vectors / L. Liu, Y. K. Ho, S. Yau // Mol Phylogenet Evol. – 2006. – Vol. 41. – P. 64–69.

6. Wang, J. Wse, a new sequence distance measure based on word frequencies / J. Wang, X. Zheng // Mathematical Biosciences. – 2008. – Vol. 215. – P. 78–83.

7. Zhao, B. A new distribution vector and its application in genome clustering / B. Zhao, R. L. He, S. T. Yau // Mol Phylogenet Evol. – 2011. – Vol. 59. – P. 438–443.

8. Application of high-dimensional feature selection: evaluation for genomic prediction in man / M. L. Bermingham [et al.] // Scientific Reports. – 2015. – Vol. 5:10312. – P. 1–12.

9. GFF/GTF File Format – Definition and Supported Options [Electronic resource]. – 2014. – Mode of access: www.ensembl.org/info/website/upload/gff.html. – Date of access: 16.10.2014.

10. Comparative analyses between retained introns and constitutively spliced introns in arabidopsos thaliana using random forest and support vector machine / R. Mao [et al.] // PLoS One. – 2014. – Vol. 9, no. 8. – P. 1–12.

11. Разработка алгоритмов и автоматизированных программных средств для классификации кодирующих и некодирующих нуклеотидных последовательностей / Д. А. Сырокваш [и др.] // Междунар. конгресс по информатике: информационные системы и технологии : материалы конгресса, Минск, 24–27 окт. 2016 г. ; редкол.: С. В. Абламейко [и др.]. – Минск : БГУ, 2016. – С. 189–193.

12. Do we need hundreds of classifiers to solve real world classification problems? / M. Fernández-Delgado [et al.] // J. of Machine Learning Research. – 2014. – Vol. 15. – P. 3133–3181.

13. Liaw, A. Breiman and Custler's Random Forests for Classification and Regression [Electronic resource] / A. Liaw, M. Wiener. – 2016. – Mode of access: http://www.stat.berkley.edu/~breiman/RandomForest/cc_home.htm#workings. – Date of access: 11.02.2016.

14. Breiman, L. Random forest / L. Breiman // Machine Learning. – 2001. – Vol. 45(1). – P. 5–32.

15. Вапник, В. Н. Восстановление зависимостей по эмпирическим данным / В. Н. Вапник. – М. : Наука, 1979. – 448 с.

16. Вьюгин, В. В. Математические основы машинного обучения и прогнозирования / В. В. Вьюгин. – М. : МЦНМО, 2014. – 304 с.

17. Мастицкий, С. Э. Статистический анализ и визуализация данных с помощью *R* [Электронный ресурс] / С. Э. Мастицкий, В. К. Шитиков. – 2014. – Режим доступа: http://r-analytics.blogspot/.com. – Дата доступа: 13.03.2015.

18. Advancing Feature Selection Research – ASU Feature Selection Repository [Electronic resource] / Z. Zhao [et al.]. – 2010. – Mode of access: https://www.researchgate.net/publication/305083748_Advancing_feature_selection_research. – Date of access: 10.04.2019.

19. Kuhn, M. The Caret Package [Electronic resource] / M. Kuhn. – 2017. – Mode of access: https://topepo.github.io/caret. – Date of access: 11.04.2017.

References

1. Edwards D. J., Holt K. E. Beginner's guide to comparative bacterial genome analysis using next-generation sequence data. *Microbial Informatics and Experimentation*, 2013, vol. 3:2, pp. 1–9.
2. Bao J., Yuan R., Bao Z. An improved alignment-free model for DNA sequence similarity metric. *BMC Bioinformatics*, 2014, vol. 15:312, pp. 1–15.
3. Li C., Wang J. Relative entropy of DNA and its application. *Physica A*, 2005, vol. 347, pp. 465–471.
4. Dai Q., Liu X., Yao Y., Zhao F. Numerical characteristics of word frequencies and their application to dissimilarity measure for sequence comparison. *Journal of Theoretical Biology*, 2011, vol. 276, pp. 174–180.
5. Liu L., Ho Y. K., Yau S. Clustering DNA sequences by feature vectors. *Mol Phylogenet Evol*, 2006, vol. 41, pp. 64–69.
6. Wang J., Zheng X. Wse, a new sequence distance measure based on word frequencies. *Mathematical Biosciences*, 2008, vol. 215, pp. 78–83.
7. Zhao B., He R. L., Yau S. T. A new distribution vector and its application in genome clustering. *Mol Phylogenet Evol*, 2011, vol. 59, pp. 438–443.
8. Bermingham M. L., Pong-Wong R., Spiliopoulou A., Hayward C., Rudan I., …, Haley C. S. Application of high-dimensional feature selection: evaluation for genomic prediction in man. *Scientific Reports*, 2015, vol. 5:10312, pp. 1–12.
9. *GFF/GTF File Format – Definition and Supported Options*, 2014. Available at: www.ensembl.org/info/website/upload/gff.html (accessed 16.10.2014).
10. Mao R., Kumar P. K. R., Guo C., Zhang Y., Liang C. Comparative analyses between retained introns and constitutively spliced introns in arabidopsos thaliana using random forest and support vector machine. *PLoS One*, 2014, vol. 9, no. 8, pp. 1–12.
11. Syrakvash D. A., Jackov N. N., Nazarov P. V., Skakun V. V. Razrabotka algoritmov i avtomatizirovannyh programmnyh sredstv dlya klassifikacii kodirujushchih i nekodiruyushchih nukleotidnyh posledovatel'nostey [Development of algorithms and automated software for the classification of coding and non-coding nucleotide sequences]. Mejdunarodnyi congress po informatike: informacionnye sistemy i tehnologii [*International Congress on Informatics: Information Systems and Technologies*]. Minsk, Belorusskij gosudarstvennyj universitet, 2016, pp. 189–193 (in Russian).
12. Fernández-Delgado M., Cernadas E., Barro S., Amorim D. Do we need hundreds of classifiers to solve real world classification problems? *Journal of Machine Learning Research*, 2014, vol. 15, pp. 3133–3181.
13. Liaw A., Wiener M. *Breiman and Custler's Random Forests for Classification and Regression*, 2016. Available at: http://www.stat.berkley.edu/~breiman/RandomForest/cc_home.htm#workings (accessed 11.02.2016).
14. Breiman L. Random forest. *Machine Learning*, 2001, vol. 45(1), pp. 5–32.
15. Vapnik V. N. Vosstanovlenie zavisimostey po empiricheskim dannym. *Recovering Dependencies from Empirical Data*. Moscow, Nauka, 1979, 448 p. (in Russian).
16. V'ugin V. V. Matematicheskie osnovy mashinnogo obucheniya i prognozirovaniya. *Mathematical Foundations of Machine Learning and Prediction*. Moscow, Moskovskij centr nepreryvnogo matematicheskogo obrazovanija, 2014, 304 p. (in Russian).
17. Mastickiy C. E., Shitikov V. K. Statisticheskiy analiz i vizualizaciya dannyh s pomoshchju *R*. *Statistical Analysis and Data Visualization with R*, 2014. Available at: http://r-analytics.blogspot.com (accessed 13.03.2015) (in Russian).
18. Zhao Z., Sharma S., Morstatter F., Alelyani S. *Advancing Feature Selection Research – ASU Feature Selection Repository*, 2010. Available at: https://www.researchgate.net/publication/305083748_Advancing_feature_selection_research (accessed 10.04.2019).
19. Kuhn M. *The Caret Package*, 2017. Available at: https://topepo.github.io/caret (accessed 11.04.2017).

Информация об авторах

Закирова Вероника Рашидовна, магистрант, кафедра системного анализа и компьютерного моделирования, факультет радиофизики и компьютерных технологий, Белорусский государственный университет, Минск, Беларусь.
E-mail: veranika.zakirava@gmail.com

Information about the authors

Veranika R. Zakirava, Master Student, Department of Systems Analysis and Computer Modelling, Faculty of Radiophysics and Computer Technologies, Belarusian State University, Minsk, Belarus.
E-mail: veranika.zakirava@gmail.com

Сырокваш Дмитрий Алексеевич, магистр, кафедра системного анализа и компьютерного моделирования, факультет радиофизики и компьютерных технологий, Белорусский государственный университет, Минск, Беларусь.
E-mail: dzmitry.syrakvash@gmail.com

Гилевский Станислав Викентьевич, доцент, кандидат технических наук, кафедра системного анализа и компьютерного моделирования, факультет радиофизики и компьютерных технологий, Белорусский государственный университет, Минск, Беларусь.
E-mail: Hileuski@bsu.by

Назаров Петр Владимирович, кандидат физико-математических наук, отдел исследования протеома и генома, Люксембургский институт здоровья, отделение онкологии, Штрассен, Люксембург.
E-mail: petr.nazarov@lih.lu

Яцков Николай Николаевич, доцент, кандидат физико-математических наук, кафедра системного анализа и компьютерного моделирования, факультет радиофизики и компьютерных технологий, Белорусский государственный университет, Минск, Беларусь.
E-mail: yatskou@bsu.by

Dzmitry A. Syrakvash, Master, Department of Systems Analysis and Computer Modelling, Faculty of Radiophysics and Computer Technologies, Belarusian State University, Minsk, Belarus.
E-mail: dzmitry.syrakvash@gmail.com

Stanislau V. Hileuski, Associate Professor, Cand. Sci. (Eng.), Department of Systems Analysis and Computer Modelling, Faculty of Radiophysics and Computer Technologies, Belarusian State University, Minsk, Belarus.
E-mail: Hileuski@bsu.by

Petr V. Nazarov, Cand. Sci. (Phys.-Math.), Proteome and Genome Research Unit, Luxembourg Institute of Health, Department of Oncology, Strassen, Luxembourg.
E-mail: petr.nazarov@lih.lu

Mikalai M. Yatskou, Associate Professor, Cand. Sci. (Phys.-Math.), Department of Systems Analysis and Computer Modelling, Faculty of Radiophysics and Computer Technologies, Belarusian State University, Minsk, Belarus.
E-mail: yatskou@bsu.by

ISSN 1816-0301 (Print)
ISSN 2617-6963 (Online)

НАУЧНОЕ НАСЛЕДИЕ

SCIENTIFIC HARITAGE

УДК 683.735.33

Поступила в редакцию 22.04.2019
Received 22.04.2019

Принята к публикации 24.04.2019
Accepted 24.04.2019

Научная школа профессора А. А. Петровского

М. И. Вашкевич, И. С. Азаров✉, В. А. Вишняков

Белорусский государственный университет информатики и радиоэлектроники, Минск, Беларусь
E-mail: azarov@bsuir.by

Аннотация. Представлены два периода научной деятельности профессора Александра Александровича Петровского, который на протяжении 15 лет (2004–2019) являлся членом редакционной коллегии журнала «Информатика». Показаны основные научные результаты, его вклад в области разработки теории и аппаратно-программных средств проблемно-ориентированных систем реального времени и обработки звуковой, речевой, графической информации, приведен перечень наиболее значимых трудов ученого.

Ключевые слова: электронно-вычислительные средства, цифровая обработка сигналов, дискретное косинусное преобразование, алгебра кватернионов

Для цитирования. Вашкевич, М. И. Научная школа профессора А. А. Петровского / М. И. Вашкевич, И. С. Азаров, В. А. Вишняков // Информатика. – 2019. – Т. 16, № 2. – С. 119–124.

Scientific school of professor A. A. Petrovsky

Maxim I. Vashkevich, Elias S. Azarov✉, Uladzimir A. Vishniakou

Belarusian State University of Informatics and Radioelectronics, Minsk, Belarus
E-mail: azarov@bsuir.by

Abstract. Two periods of scientific activity of Professor Alexander Alexandrovich Petrovsky, who was a member of the editorial board of the journal "Informatics" for 15 years (2004–2019), are presented. The main scientific results, his contribution to the development of the theory and to the hardware and software of the problem-oriented real-time systems and the processing of audio, speech and graphic information are shown, a list of the most significant works of the scientist is given.

Keywords: electronic computing tools, digital signal processing, discrete cosine transform, quaternion algebra

For citation. Vashkevich M. I., Azarov E. S., Vishniakou U. A. Scientific school of professor A. A. Petrovsky. *Informatics*, 2019, vol. 16, no. 2, pp. 119–124 (in Russian).

Научное наследие А. А. Петровского впечатляет своей широтой. Профессора всегда интересовали новые научные направления, которые он обогащал своими идеями и многочисленными результатами. Выбирая ученикам темы научных исследований, он всегда предоставлял возможность им самим решать поставленные задачи и поощрял личную инициативу, при этом умел воодушевить, поддержать, подсказать и поделиться своим богатым опытом. Благодаря этому Александр Александрович сформировал вокруг себя молодой и инициативный коллектив, что и позволило общими усилиями внести существенный вклад в развитие многих направлений цифровой обработки сигналов. Деятельность ученого охватывает два периода: 1975–1995 гг. (теория и средства микропроцессорных проблемно-ориентированных систем реального времени) и 1995–2019 гг. (теория и микропроцессорные средства обработки речи и изображений).

Первый период научной деятельности (1975–1995 гг.). А. А. Петровский начинает работу в 1975 г. младшим научным сотрудником, затем ассистентом кафедры ЭВМ в тогда еще Минском радиотехническом институте. В 1977 г. поступает учиться в дневную аспирантуру, работает над аппаратно-программной поддержкой стендовых испытаний и в срок представляет и защищает кандидатскую диссертацию «Разработка и исследование вычислительных устройств управления стендовыми испытаниями на пространственно-многомерную случайную вибрацию». Цифровая система управления стендовыми испытаниями на пространственно-временные случайные вибрации, созданная при его участии, в 1981 г. получила серебряную медаль ВДНХ СССР.

В 1981 г. Александр Александрович переводится на должность доцента кафедры (получает это научное звание в 1983 г.), подготавливает и читает курс лекций по ЭВМ. Заочно оканчивает двухгодичные курсы английского языка при Институте иностранных языков и проходит конкурс на годичную научную стажировку в Лондонском университете. С 1985 г. – научный руководитель группы, а затем лаборатории микропроцессорных систем реального времени, которая выполняла хоздоговорные работы оборонного характера. Многоканальный анализатор спектра, созданный его группой, в 1986 г. получил золотую медаль ВДНХ СССР. По результатам этих работ А. А. Петровский опубликовал монографию и учебное пособие [1, 2], а в 1989 г. в Киеве защитил докторскую диссертацию «Теория и практика построения алгоритмических и аппаратно-программных средств микропроцессорных распределенных проблемно-ориентированных систем реального времени», связанную с оборонной тематикой.

В 1990 г. в жизни ученого происходит много событий. С грифом Министерства образования БССР в соавторстве с ним выходит второе учебное пособие [3], посвященное микроЭВМ. Под руководством А. А. Петровского защищают кандидатские диссертации в области разработки компонентов специализированных микроЭВМ первые ученики: Ю. Ганнушкин, А. Цирульников и М. Качинский. Выдающимся результатом его коллектива является разработка комплекса учебных персональных ЭВМ «Немига» (с лучшими показателями и параметрами того времени), которые успешно использовались в школах и вузах республики [4]. В 1990 г. был получен аттестат профессора СССР. В этом же году профессор А. А. Петровский в возрасте 37 лет назначен на должность заведующего кафедрой конструирования и производства электронно-вычислительной аппаратуры, для которой он разработал новый учебный план по специальности «электронно-вычислительные средства» (ЭВС) и которой бессменно руководил до ноября 2017 г.

Новый план подготовки специалистов был направлен на проектирование проблемно-ориентированных вычислительных средств различного назначения с использованием САПР и постоянно изменяющейся элементной базой, по основным дисциплинам выходит учебное пособие [5]. На кафедре создаются новые учебные лаборатории: по микропроцессорным устройствам, цифровой обработке сигналов, САПР ЭВС с изучением процессоров Texas Instrument, Motorola и ПО поддержки логического проектирования SNAP. В рамках специальности начинается подготовка по трем новым специализациям: «Проектирование проблемно-ориентированных ЭВМ», «Проектирование и технология персональных компьютеров и периферийных устройств», «ЭВС для мультимедийных устройств». Профессор читает курсы «Введение в специальность», «Теория и применение цифровой обработки сигналов», «Проектирование электронных вычислительных средств с динамически реконфигурируемой архитек-

турой» и «Алгоритмические основы компьютерной графики». Ученики Александра Александровича В. Клюс, В. Сидоренко и А. Давыдов защищают кандидатские диссертации в области проектирования аппаратного и программного обеспечения специализированных микроЭВМ.

В 1994 г. А. А. Петровский работал в Ахенском университете на кафедре профессора П. Вари, где занимался обработкой речи, и после его возвращения в научной лаборатории микропроцессорных систем реального времени стали заниматься обработкой звуковой и речевой информации.

Второй период научной деятельности (1995–2019 гг.).

Банки цифровых фильтров. А. А. Петровский активно развивал современную теорию банков фильтров и специализированных методов частотной декомпозиции дискретных сигналов для различных практических задач, а также метод Фурье-анализа сигнала с неравномерным частотным разрешением [6]. Большой вклад ученый внес в теорию построения эффективных аппроксимаций дискретного косинусного преобразования (ДКП) [7]. Совместно с М. Перфенюком и М. Вашкевичем Александр Александрович разработал теорию проектирования неравнополосных косинусно-модулированных банков фильтров [8, 9]. На протяжении почти двух десятков лет им также развивалась теория параунитарных банков фильтров на основе алгебры кватернионов [10–12] и с ее помощью решались практические задачи.

Методы обработка речи и звука. Речевая коммуникация является одной из важнейших отличительных способностей человека. Поэтому профессора привлекала область обработки речевых сигналов, которой посвящено более сотни работ (например, [13]).

Кодирование речи и звука. Научная школа А. А. Петровского внесла существенный вклад в развитие методов кодирования речевых и звуковых сигналов. В этом направлении выполнены следующие кандидатские работы его учеников: «Кодирования речевого сигнала на основе антропоморфической обработки и синусоидальных моделей» Д. С. Лихачева [14] и «Перцептуальный широкополосный CELP-кодер речи» М. З. Лившица [15]. Отдельно следует отметить кандидатскую и докторскую диссертации старшего сына профессора Алексея Александровича Петровского, посвященные методам и средствам перцептуального субполосного кодирования аудиоданных [16].

Шумоподавление. Много работ в научной школе профессора А. А. Петровского выполнено в области шумоочистки речевых сигналов. Совместно с М. Парфенюком и А. Боровичем Александр Александрович описал систему перцептуального шумоподавления на основе Фурье-анализа с неравномерным частотным разрешением [6]. Вместе с Я. Башуном разработал метод фильтрации речевого сигнала в модуляционной области, который затем был развит и в работах с И. С. Азаровым [17]. Особо в этом направлении стоит упомянуть метод обработки сигнала в подпространствах, разработанный с А. Боровичем [18].

Конверсия голоса, частота основного тона. В рамках кандидатских исследований своего вьетнамского ученика Тхая Киена профессор обращается к научной проблематике конверсии голоса [19]. Данное направление впоследствии была развито в работах двух других его учеников И. С. Азарова [20] и В. А. Захарьева [21]. Одной из фундаментальных задач в области обработки речевых сигналов является определение частоты основного тона. На протяжении двух десятков лет А. А. Петровский со своими учениками делает ряд удачных исследований в направлении развития подходов к решению этой сложной задачи. Различные варианты методов оценивания основного тона были получены и представлены на конференциях мирового уровня совместно с В. Серковым [22], П. Зубрицким [23], А. Павловцом [24] и И. Азаровым [25].

Обработка сигнала для слуховых аппаратов и кохлеарных имплантов. Александр Александрович стремился служить науке и обществу. В рамках своей деятельности он развивал методы и средства цифровой обработки сигналов для протезирования слуха. Под его руководством польский аспирант Я. Башун защитил кандидатскую диссертацию, связанную с разработкой кохлеарных имплантов [26]. Через 10 лет другой его ученик М. И. Вашкевич защитил диссертацию, посвященную совершенствованию обработки сигналов в слуховых аппаратах [27]. Впоследствии А. А. Петровский вместе с И. С. Азаровым и М. И. Вашкевичем разработал мобильное приложение, позволяющее смартфону выполнять функции слухового аппарата [28].

Методы обработки изображений. В последние пять лет в научной школе профессора А. А. Петровского появился новый вектор развития, связанный с методами сжатия изоб-

ражений без потерь. В этом направлении написаны кандидатская диссертация младшего сына профессора Николая Александровича Петровского «Обработка изображений при помощи параунитарных банков фильтров на кватернионах» и диссертационная работа В. В. Ключени «Обработка изображений на основе ДКП» [29].

Аппаратная реализация методов цифровой обработки сигнала. Особенностью работ, выполняемых под руководством профессора, всегда была их направленность на практическую аппаратную реализацию. Для большинства предложенных методов обработки сигналов разработаны специализированные процессоры, представлены способы их эффективной аппаратной и программной реализации. Во многих его трудах описаны реализация дискретного косинусного преобразования, параунитарные банки фильтров в алгебре кватернионов, аппаратные платформы кодеров речи и звука [30]. Несколько последних работ, написанных совместно с врачем Ю. Н. Рушкевич и докторантом М. И. Вашкевичем РНПЦ Неврологии и нейрохирургии, были посвящены проблеме детектирования признаков бокового амиотрофического склероза по голосу [31, 32].

Александр Александрович за свою научно-педагогическую жизнь создал всемирно известную научную школу по цифровой обработке сигналов и мультимедиа, включающую кафедру электронно-вычислительных средств, 30 кандидатов и двух докторов наук, большое количество широко используемых в республике и мире аппаратных и программных разработок, а также опубликовал более 600 научных трудов.

14 марта 2019 г. Александр Александрович ушел из жизни. Деятельность выдающегося ученого – достойный пример служения науке и обществу.

Список использованных источников

1. Петровский, А. А. Методы и микропроцессорные средства обработки широкополосных и быстро меняющихся процессов в реальном времени / А. А. Петровский. – Минск : Наука и техника, 1988. – 271 с.

2. Применение управляющих вычислительных машин / А. А. Петровский [и др.]. – Минск : Вышэйшая школа, 1989. – 238 с.

3. Вишняков, В. А. Системное обеспечение микроЭВМ / В. А. Вишняков, А. А. Петровский. – Минск : Вышэйшая школа, 1990. – 304 с.

4. Бытовые и персональные ЭВМ. Энциклопедический справочник / А. А. Петровский [и др.]. – Минск : Белорусская энциклопедия, 1995. – С. 741–821.

5. Петровский, А. А. Проектирование проблемно-ориентированных ЭВС на цифровых процессорах сигналов : метод. пособие / А. А. Петровский, М. В. Качинский, В. Б. Клюс. – Минск : БГУИР, 1996. – 76 с.

6. Borowicz, A. An application of the warped discrete Fourier transform in the perceptual speech enhancement / A. Borowicz, M. Parfieniuk, A. Petrovsky // Speech Communication. – 2006. – Vol. 48. – P. 1024–1036.

7. Parfieniuk, M. Near-perfect reconstruction oversampled nonuniform cosine-modulated filter banks based on frequency warping and subband merging / M. Parfieniuk, A. Petrovsky // Intern. J. of Electronics and Telecommunications. – 2012. – Vol. 58, no. 2. – P. 177–192.

8. Вашкевич, М. И. Косинусно-модулированные банки фильтров с фазовым преобразованием: реализация и применение в слуховых аппаратах / М. И. Вашкевич, И. С. Азаров, А. А. Петровский. – М. : Горячая линия – Телеком, 2014. – 210 с.

9. Parfieniuk, M. Quaternionic lattice structures for four-channel paraunitary filter banks / M. Parfieniuk, A. Petrovsky // EURASIP J. on Advances in Signal Processing. – 2007. – 12 p.

10. Parfieniuk, M. Inherently lossless structures for eight- and six-channel linear-phase paraunitary filter banks based on quaternion multipliers / M. Parfieniuk, A. Petrovsky // Signal Processing. – 2010. – Vol. 90, no. 6. – P. 1755–1767.

11. Petrovsky, N. A. Embedded distributed arithmetic based quaternions multiplier of paraunitary filter bank for lossless-to-lossy image coding / N. A. Petrovsky, E. V. Rybenkov, A. A. Petrovsky // Microprocessors and Microsystems. – 2017. – Vol. 52. – P. 510–522.

12. Parfieniuk, M. Structurally orthogonal finite precision implementation of the eight point DCT / M. Parfieniuk, A. Petrovsky // IEEE Intern. Conf. on Acoustics Speech and Signal Processing Proceedings. – Toulouse, 2006. – P. 161–164.

13. Анализаторы речевых и звуковых сигналов: методы, алгоритмы и практика (с MATLAB примерами) : монография / А. А. Петровский [и др.] ; под ред. А. А. Петровского. – Минск : БГУИР, 2009. – 460 с.

14. Лихачев, Д. С. Анализ и синтез устройств кодирования речевого сигнала на основе антропоморфической обработки и синусоидальных моделей / Д. С. Лихачев, А. А. Петровский // Доклады БГУИР. – 2006. – № 3(51). – С. 35–43.

15. Лившиц, М. З. Широкополосный CELP-кодер с мультиполосным возбуждением и многоуровневым векторным квантованием по кодовой книге с реконфигурируемой структурой / М. З. Лившиц, М. Парфенюк, А. А. Петровский // Цифровая обработка сигналов. – 2005. – № 2. – С. 20–35.

16. Petrovsky, Al. Hybrid signal decomposition based on instantaneous harmonic parameters and perceptually motivated wavelet packets for scalable audio coding / Al. Petrovsky, E. Azarov, A. Petrovsky // Signal Processing. Special issue "Fourier Related Transforms for Non-Stationary Signals". – 2011. – Vol. 91, iss. 6. – P. 1489–1504.

17. Азаров, И. С. Алгоритм очистки речевого сигнала от сложных помех путем фильтрации в модуляционной области / И. С. Азаров, М. И. Вашкевич, А. А. Петровский // Цифровая обработка сигналов. – 2013. – № 4. – С. 25–31.

18. Borowich, A. Signal subspace approach for psychoacoustically motivated speech enhancement / A. Borowich, A. Petrovsky // Speech Communication. Elsiver. – 2011. – Vol. 53. – P. 210–219.

19. Тхай, Ч. Киен. Реализация и выбор параметров при использовании алгоритма выравнивания временных масштабов для систем конверсии голоса / Ч. Киен Тхай // Доклады БГУИР. – 2008. – № 3(33). – С. 96–102.

20. Real-time voice conversion using artificial neural networks with rectified linear units / E. Azarov [et al.] // Proc. INTERSPEECH. – Lyon, France, 2013. – P. 1032–1036.

21. Захарьев, В. А. Система синтеза речи по тексту с возможностью настройки на голос целевого диктора / В. А. Захарьев, А. А. Петровский, Б. М. Лобанов // Тр. СПИИРАН. – 2014. – № 32. – С. 82–98.

22. Sercov, V. V. The method of pitch frequency detection on the base of tuning to its harmonics / V. V. Sercov, A. A. Petrovsky // 9th European Signal Processing Conf. (EUSIPCO 1998). – Island of Rhodes, 1998. – P. 1–4.

23. Zubrycki, P. Analysis/synthesis speech model based on the pitch-tracking periodic-aperiodic decomposition / P. Zubrycki, A. A. Petrovsky // Information Processing and Security Systems. – 2005. – P. 33–42.

24. Pavlovets, A. Robust HNR-based closed-loop pitch and harmonic parameters estimation / A. Pavlovets, A. A. Petrovsky // INTERSPEECH. – Florence, 2011. – P. 1981–1984.

25. Azarov, E. Instantaneous pitch estimation based on RAPT framework / E. Azarov, M. Vashkevich, A. Petrovsky // 20th European Signal Processing Conf. (EUSIPCO 2012). – Bucharest, Romania, 2012. – P. 2787–2791.

26. Baszun, Ja. Flexible cochlear system based on digital model of cochlea: structure, algorithms and testing / Ja. Baszun, A. A. Petrovsky // 10th European Signal Processing Conf. (EUSIPCO 2000). – Tampere, 2000. – P. 1–4.

27. Вашкевич, М. И. Косинусно-модулированные банки фильтров с фазовым преобразованием: реализация и применение в слуховых аппаратах / М. И. Вашкевич, И. С. Азаров, А. А. Петровский. – М. : Горячая линия – Телеком, 2014. – 212 с.

28. Система коррекции слуха на мобильной вычислительной платформе / И. С. Азаров [и др.] // Информатика. – 2014. – № 2(42). – С. 5–24.

29. Ключеня, В. В. Быстрое прототипирование встраиваемых программируемых систем на ПЛИС для мультимедийных приложений / В. В. Ключеня, А. А. Петровский // Информатика. – 2015. – № 3(47). – С. 13–28.

30. Петровский, Ал. А. Быстрое проектирование систем мультимедиа от прототипа / Ал. А. Петровский, А. В. Станкевич, А. А. Петровский. – Минск : Бестпринт, 2011 – 410 с.

31. Features extraction for the automatic detection of ALS disease from acoustic speech signals / M. Vashkevich [et al.] // 2018 Signal Processing: Algorithms, Architectures, Arrangements, and Applications (SPA). – Poznan, 2018. – P. 321–326.

32. Акустический анализ голоса для выявления речевых нарушений при боковом амиотрофическом склерозе / М. И. Вашкевич [и др.] // Доклады БГУИР. – 2018. – № 7(117). – С. 64–68.

Информация об авторах

Вашкевич Максим Иосифович, кандидат технических наук, доцент, доцент кафедры ЭВС, Белорусский государственный университет информатики и радиоэлектроники, Минск, Беларусь.
E-mail: vashkevich@bsuir.by

Азаров Илья Сергеевич, доктор технических наук, доцент, заведующий кафедрой ЭВС, Белорусский государственный университет информатики и радиоэлектроники, Минск, Беларусь.
E-mail: azarov@bsuir.by

Вишняков Владимир Анатольевич, доктор технических наук, профессор, профессор кафедры инфокоммуникационных технологий, Белорусский государственный университет информатики и радиоэлектроники, Минск, Беларусь.
E-mail: vush2002@list.ru

Information about the authors

Maxim I. Vashkevich, Cand. Sci. (Eng.), Assoc. Prof., Assoc. Prof. of the Department of EMU, Belarusian State University of Informatics and Radioelectronics, Minsk, Belarus.
E-mail: vashkevich@bsuir.by

Elias S. Azarov, Dr. Sci. (Eng.), Assoc. Prof., Head of the Department of EMU, Belarusian State University of Informatics and Radioelectronics, Minsk, Belarus.
E-mail: azarov@bsuir.by

Uladzimir A. Vishniakou, Dr. Sci. (Eng.), Prof., Professor of the Department of Information and Communication Technologies, Belarusian State University of Informatics and Radioelectronics, Minsk, Belarus.
E-mail: vush2002@list.ru

Правила для авторов

Редакция журнала «Информатика» просит авторов руководствоваться приведенными ниже правилами:

1. Статьи принимаются в редакцию через электронную систему подачи по адресу http://inf.grid.by в формате файлов текстовых редакторов Microsoft Word. Основной текст статьи не должен превышать 17 стр., включая рисунки, таблицы и достаточное количество наиболее актуальных ссылок; обзорной статьи – 10 стр., включая все основные ссылки. Текст набирается с переносами, шрифт Times New Roman 11 пт, интервал между строками одинарный, абзацный отступ 0,5 см, поля по 2,5 см со всех сторон.

Изложенный в статье материал должен быть четко структурированным: введение, цели и задачи, методы, результаты, заключение (выводы).

2. Статьи о результатах работ, проведенных в научных учреждениях, должны иметь разрешение на публикацию (сопроводительное письмо за подписью руководителя или выписку из заседания ученого совета, отдела или кафедры, акт экспертизы).

3. Статья в обязательном порядке должна иметь следующую структуру: индекс по универсальной десятичной классификации (УДК); инициалы и фамилии всех авторов, название статьи, полное название учреждений, где работают авторы, с указанием города, страны, аннотацию (150–250 слов), подрисуночные надписи, названия таблиц и ключевые слова (7–10) на русском и английском языках, адрес электронной почты контактного лица.

4. Аннотация (авторское резюме) должна кратко представлять результаты работы и быть информативной, содержательной. Приветствуется структура аннотации, повторяющая структуру статьи и включающая введение, цели и задачи, методы, результаты, заключение.

5. Формулы, рисунки, таблицы в статье нумеруются в соответствии с порядком их упоминания в тексте. Ссылки на рисунки и таблицы в тексте обязательны. Рисунки должны быть выполнены с хорошим разрешением в масштабе, позволяющем четко различать надписи и обозначения. Подрисуночные подписи с расшифровкой всех позиций, представленных на рисунке, набираются шрифтом гарнитуры основного текста размером 9 пт. Цветные иллюстрации печатаются только в том случае, когда это необходимо для понимания излагаемого материала.

6. Набор формул выполняется в формульном редакторе Microsoft Equation или Math Type. Прямым шрифтом набираются: греческие и русские буквы; математические символы (sin, lg, ∞); символы химических элементов (C, Cl, $CHCl_3$); цифры (римские и арабские); векторы; индексы (верхние и нижние), являющиеся сокращениями слов. Курсивом набираются латинские буквы, символы физических величин (в том числе и в индексе).

7. Сокращения в тексте статьи (за исключением единиц измерения) могут быть использованы только после упоминания полного термина. Единицы измерения физических величин следует приводить в Международной системе единиц (СИ).

8. Цитируемые в статье фамилии авторов теорем, теорий, законов и т. д. следует приводить в скобках на языке оригинала после русского написания. Например, теорема Эйлера (Euler).

9. Список использованной литературы оформляется в соответствии с требованиями Высшей аттестационной комиссии Республики Беларусь (ГОСТ 7.5–2008). Номер литературной ссылки в тексте дается порядковым номером в квадратных скобках. Ссылаться на неопубликованные работы не допускается.

10. Отдельно приводится список цитированных источников в *романском* (латинском) алфавите со следующей структурой: авторы (транслитерация), название статьи в транслитерированном варианте [перевод названия статьи на английский язык в квадратных скобках], название русскоязычного источника (транслитерация) [перевод названия источника на английский язык – парафраз (для журналов можно не делать)], выходные данные с обозначениями на английском языке.

11. Поступившие в редакцию статьи направляются на рецензирование специалистам. Основным критерием целесообразности публикации является новизна и информативность статьи. Если по рекомендациям рецензента статья возвращается автору на доработку, а переработанная рукопись вновь рассматривается редколлегией, датой поступления считается день получения редакцией ее окончательного варианта. Статьи не по профилю журнала возвращаются авторам после заключения редколлегии.

12. Статьи, направляемые на доработку, должны быть возвращены в исправленном виде с ответами на все замечания.

13. Редакция журнала предоставляет возможность первоочередного опубликования статей, представленных лицами, которые осуществляют послевузовское обучение (аспирантура, докторантура, соискательство) в год завершения обучения.

14. Авторы несут ответственность за направление в редакцию статей, уже опубликованных ранее или принятых к публикации другими изданиями.

15. Редакция оставляет за собой право на редакционные изменения, не искажающие основное содержание статьи. Окончательное решение о публикации принимается редакционной коллегией.

Индексы

00827

для индивидуальных подписчиков

008272

для предприятий и организаций

ИНФОРМАТИКА Т. 16, № 2 апрель–июнь 2019