ISSN 1816-0301 (print)

ЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ

LOGICAL DESIGN

УДК 519.7

Поступила 24.04.2018
Received 24.04.2018

П. Н. Бибило, Н. А. Кириенко, Ю. Ю. Ланкевич

Объединенный институт проблем информатики
Национальной академии наук Беларуси, Минск, Беларусь

ЛОГИЧЕСКАЯ ОПТИМИЗАЦИЯ МНОГОУРОВНЕВЫХ ПРЕДСТАВЛЕНИЙ СИСТЕМ БУЛЕВЫХ ФУНКЦИЙ НА ОСНОВЕ БЛОЧНОГО РАЗБИЕНИЯ И РАЗЛОЖЕНИЯ ШЕННОНА

Аннотация. Описываются результаты экспериментального исследования эффективности применения процедур оптимизации систем булевых функций, предварительно выполняемых при синтезе комбинационных схем. Процедуры используют алгоритмы разбиения систем функций на подсистемы и алгоритмы оптимизации многоуровневых представлений функций на основе разложения Шеннона. Показывается, что разбиение исходной системы булевых функций на подсистемы и оптимизация подсистем на основе разложения Шеннона с нахождением инверсных подфункций, входящих в разложения (BDDI-оптимизация), во многих случаях позволяют при синтезе уменьшать площадь комбинационных схем из библиотечных элементов. Совместная BDDI-оптимизация является более предпочтительным методом по сравнению с раздельной технологически независимой BDDI-оптимизацией, так как площадь схем, построенных по совместным BDDI, в подавляющем числе случаев меньше площади схем, построенных по раздельным BDDI.

Ключевые слова: булева функция, дизъюнктивная нормальная форма, диаграмма двоичного выбора, разбиение системы булевых функций, синтез логической схемы, КМОП-технология

Для цитирования. Бибило, П. Н. Логическая оптимизация многоуровневых представлений систем булевых функций на основе блочного разбиения и разложения Шеннона / П. Н. Бибило, Н. А. Кириенко, Ю. Ю. Ланкевич // Информатика. – 2018. – Т. 15, № 3. – С. 56–70.

P. N. Bibilo, N. A. Kirienko, Y. Y. Lankevich

The United Institute of Informatics Problems of the National Academy
of Sciences of Belarus, Minsk, Belarus

LOGICAL OPTIMIZATION THE MULTILEVEL REPRESENTATIONS OF SYSTEMS OF BOOLEAN FUNCTIONS BASED ON PARTITIONING INTO BLOCKSAND SHANNON DECOMPOSITION

Abstract. The results of experimental study of the effectiveness of optimization procedures for systems of Boolean functions which are used in the synthesis of combinational circuits are described. The procedures use algorithms for partitioning systems of functions into subsystems and algorithms for optimizing multilevel representations of functions based on Shannon decomposition. The Shannon decomposition uses the procedure for finding inverse subfunctions, contained in decomposition result (BDDI-optimization). It is shown that these procedures can reduce the area of combinational circuits from library gates in many cases in the process of synthesis. Joint BDDI optimization is more preferable method in comparison to separate technologically independent BDDI optimization, since the area of circuits built on joint BDDI is smaller than the area of circuits built on separate BDDI in most cases.

Keywords: Boolean function, disjunctive normal form, binary decision diagram (BDD), partitioning systems of functions, synthesis of logical circuit, CMOS-technology

For citation. Bibilo P. N., Kirienko N. A., Lankevich Y. Y. Logical optimization the multilevel representations of systems of Boolean functions based on partitioning into blocks and Shannon decomposition. *Informatics*, 2018, vol. 15, no. 3, pp. 56–70 (in Russian).

Введение. Логическая оптимизация различных представлений систем полностью определенных булевых функций обычно применяется непосредственно перед этапом технологического отображения (синтеза логических схем), на котором оптимизированные представления булевых функций покрываются описаниями базисных логических элементов [1]. В настоящее время для оптимизации описаниями булевых функций исходными являются многоуровневые представления, получаемые после этапа высокоуровневого синтеза – замены алгоритмических конструкций на языках VHDL, Verilog [2] логическими уравнениями и функциональными описаниями элементов памяти. Данные представления цифровых устройств применяются при автоматизированном проектировании, заменяя используемые ранее в качестве исходных данных таблицы истинности и системы дизъюнктивных нормальных форм (ДНФ) [1, 3].

В зависимости от целевой библиотеки синтеза и вида (формы) исходного описания систем булевых функций применяются различные методы предварительной оптимизации. Например, использование программируемых логических матриц (ПЛМ) потребовало разработки методов оптимизации сетей из ПЛМ по критериям площади и методов декомпозиции «больших» ПЛМ на сети ПЛМ с ограниченными параметрами [4]. Появление FPGA вызвало новый интерес к методам декомпозиции систем булевых функций [5, 6] и методам, основанным на разложении Шеннона и других подобных разложениях [7]. Использование предварительной логической оптимизации и последующее выполнение синтеза в промышленных синтезаторах являются перспективными направлениями для улучшения такого важного параметра синтезируемых логических схем, как площадь кристалла.

Основные методы технологически независимой оптимизации базируются на разложении Шеннона и позволяют получать оптимизированные BDD (BinaryDecision Diagram – диаграммы двоичных решений) [8–10], называемые также бинарными диаграммами, двоичными решающими диаграммами и диаграммами двоичного выбора [7]. BDD и их модификации [11] широко используются при решении самых разнообразных задач, возникающих в различных областях, в том числе синтезе, построении тестов и верификации цифровых схем [12]. Результаты исследования [13] показали, что разложения Шеннона для систем булевых функций, в которых используются как коэффициенты разложений, так и инверсии коэффициентов, оказались эффективным способом предварительной логической оптимизации при синтезе КМОП-схем из библиотечных элементов (вентилей). Однако исследования эффективности применения классических BDD выявили тот факт, что наряду с совместными классическими BDD могут успешно применяться и BDD, построенные для подсистем функций, либо раздельные BDD, построенные отдельно для каждой из функций системы.

Настоящая работа посвящена экспериментальному исследованию эффективности применения при синтезе логических схем из КМОП-элементов трех оптимизационных процедур: разбиения исходного функционального описания систем полностью определенных булевых функций на блоки, ограниченные по числу входных и выходных переменных; построения матричных форм систем ДНФ для каждого из блоков; BDD-оптимизации представления каждого из блоков с минимизацией числа коэффициентов разложений Шеннона с точностью до инверсий. Эксперименты проводились для различных форм представления исходных данных (функциональных описаний) и одной и той же библиотеки логических элементов с различными вариантами разбиения функциональных описаний на блоки.

Формы представления систем булевых функций. Под векторной булевой функцией $\boldsymbol{f}(\boldsymbol{x})$, $\boldsymbol{x}=(x_1,\ x_2,\ ...,\ x_n)$, будем понимать упорядоченную систему булевых функций $\boldsymbol{f}(x)$ = = ($f^1(\boldsymbol{x})$, …, $f^m(\boldsymbol{x})$). Широко известной в литературе [3, 4] формой представления систем булевых функций является пара матриц $<T^x,\ B^f>$: троичная матрица T^x задания элементарных конъюнкций в виде троичных векторов и булева матрица B^f вхождений конъюнкций в ДНФ компонентных функций системы. В табл. 1 дан пример задания системы ДНФ векторной полностью определенной функции $\boldsymbol{f}(\boldsymbol{x})$=($f^1(\boldsymbol{x})$, $f^2(\boldsymbol{x})$, $f^3(\boldsymbol{x})$):

$$f^1 = x_1 x_2 x_3 \bar{x}_4 \vee \bar{x}_2 \bar{x}_3 \bar{x}_4 \vee \bar{x}_1 x_2 x_4,$$

$$f^2 = \bar{x}_1 \bar{x}_2 \bar{x}_3 x_4 \vee x_1 \bar{x}_2 \bar{x}_3 \bar{x}_4 \vee x_1 x_2 \bar{x}_3 x_4 \vee \bar{x}_1 x_2 x_4 \vee x_3 x_4 \vee x_2 x_3, \quad (1)$$

$$f^3 = \bar{x}_1 \bar{x}_2 \bar{x}_3 x_4 \vee x_1 \bar{x}_2 \bar{x}_3 \bar{x}_4 \vee x_1 x_2 \bar{x}_3 x_4 \vee x_1 x_2 x_3 \bar{x}_4 \vee \bar{x}_1 x_2 \bar{x}_4.$$

Таблица 1

Пример системы ДНФ трех булевых функций

T^x	B^f
$x_1\ x_2\ x_3\ x_4$	$f^1\ f^2\ f^3$
0 0 0 1	0 1 1
1 0 0 0	0 1 1
1 1 0 1	0 1 1
1 1 1 0	1 0 1
– 0 0 0	1 0 0
0 1 – 0	0 0 1
0 1 – 1	1 1 0
– – 1 1	0 1 0
– 1 1 –	0 1 0

Систему булевых функций (табл. 1) можно представить также в виде взаимосвязанных логических уравнений, которые являются формулами, полученными командой unmap в синтезаторе LeonardoSpectrum [14] после выполнения синтеза схемы по VHDL-описанию этой системы. Особенностью таких формул, называемых еще *RTL-описаниями*, является то, что каждая из них содержит только одну логическую операцию над парой булевых переменных либо операцию инверсирования одной булевой переменной. В языке SF многоуровневое представление в виде взаимосвязанных логических уравнений обозначается как формат LOG [15].

Оптимизация многоуровневых представлений

Многоуровневое разложение Шеннона с нахождением инверсных подфункций. Еще одной используемой в данной работе формой задания систем булевых функций являются алгебраические многоуровневые представления на базе разложения Шеннона с применением инверсных подфункций (коэффициентов разложения).

Разложением Шеннона полностью определенной булевой функции $f = f(\boldsymbol{x})$ по переменной x_i называется представление

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

Функции $f_0 = f(x_1,...,x_{i-1},0,x_{i+1},...,x_n)$, $f_1 = f(x_1,...,x_{i-1},1,x_{i+1},...,x_n)$ в правой части (2) называются коэффициентами разложения по переменной x_i, остаточными подфункциями либо просто *подфункциями*. Они получаются из функции $f = f(x_1,...,x_n)$ путем подстановки вместо переменной $\boldsymbol{x}_i$ констант 0 и 1 соответственно. Каждая из подфункций $f_0 = f(x_1,...,x_{i-1},0,x_{i+1},...,x_n)$ и $f_1 = f(x_1,...,x_{i-1},1,x_{i+1},...,x_n)$ может быть разложена по одной из переменных из множества $\{x_1,...,x_{i-1},x_{i+1},...,x_n\}$. Процесс разложения подфункций заканчивается, когда все n переменных будут использованы для разложения либо когда все подфункции выродятся до констант 0, 1. На каждом шаге разложения выполняется сравнение на равенство полученных подфункций и оставляется одна из нескольких попарно равных (с точностью до инверсии) подфункций. Если же подфункции разложения по переменной x_i равны

$$f(x_1,...,x_{i-1},0,x_{i+1},...,x_n) = f(x_1,...,x_{i-1},1,x_{i+1},...,x_n), \quad (3)$$

то переменная x_i называется *несущественной* (фиктивной) и $f(x_1,...,x_{i-1},x_i,x_{i+1},...,x_n) = f(x_1,...,x_{i-1},x_{i+1},...,x_n)$.

Под BDDI (Binary Decision Diagram with Inverse cofactors) в настоящей работе понимается ориентированный ациклический граф, задающий последовательные разложения Шеннона булевой функции $f(x_1,...,x_n)$ по всем ее переменным $x_1,...,x_n$ при заданном порядке (перестановке) переменных, по которым проводятся разложения. Граф BDDI одной полностью определенной булевой функции содержит три вида вершин: функциональные вершины, соответствующие разлагаемым функциям и подфункциям (и их инверсиям); вершины-переменные; листовые вершины, соответствующие константам 0, 1. Ориентация дуг обычно не показывается, так как при изображении BDDI все дуги ориентируются сверху вниз. Функциональная вершина BDDI реализует одну функцию φ (подфункцию) либо две функции (подфункцию φ и ее инверсию $\bar{\varphi}$) [13]. Смысл выражения «с точностью до инверсии» заключается в том, что любую из пары взаимно инверсных подфункций разложения можно считать неинверсной подфункцией, тогда другая подфункция такой пары будет инверсной к выбранной.

Далее для $\boldsymbol{f}(\boldsymbol{x}) = (f^1(\boldsymbol{x}),\ldots, f^m(\boldsymbol{x}))$ будут рассматриваться BDDI двух видов – совместные и раздельные. *Совместными* BDDI, представляющими $\boldsymbol{f}(\boldsymbol{x})$, будем называть такие BDDI, которые построены по общей для всех компонентных функций $f^i(\boldsymbol{x})$ перестановке переменных. Если же BDDI для каждой из компонентных функций системы строятся независимо, т. е. так, что каждая из функций разлагается по своей перестановке переменных, то такие BDDI для системы функций будем называть *раздельными*.

Для системы функций совместная BDDI (рис. 1) реализует следующее многоуровневое представление:

$$f^1 = \bar{x}_2 s_0 \vee x_2 s_1,\ f^2 = \bar{x}_2 s_2 \vee x_2 \bar{s}_0,\ f^3 = \bar{x}_2 s_3 \vee x_2 \bar{s}_2,\ s_0 = \bar{x}_3 \bar{s}_6,\ s_1 = \bar{x}_3 s_4 \vee x_3 s_5,$$
$$s_2 = \bar{x}_3 s_5 \vee x_3 s_6,\ s_3 = \bar{x}_3 s_5,\ s_4 = \bar{x}_1 s_6,\ s_5 = \bar{x}_1 s_6 \vee x_1 \bar{s}_6,\ s_6 = x_4. \quad (4)$$

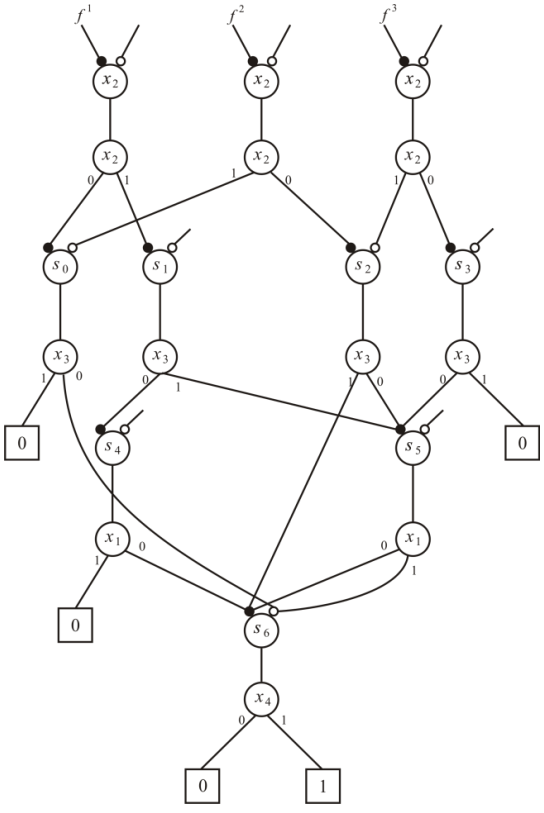


Рис. 1. Графическое представление уравнений (4)

Раздельные BDDI показаны на рис. 2. Функция f^1 (рис. 2, *а*) в форме BDDI имеет следующее многоуровневое представление:

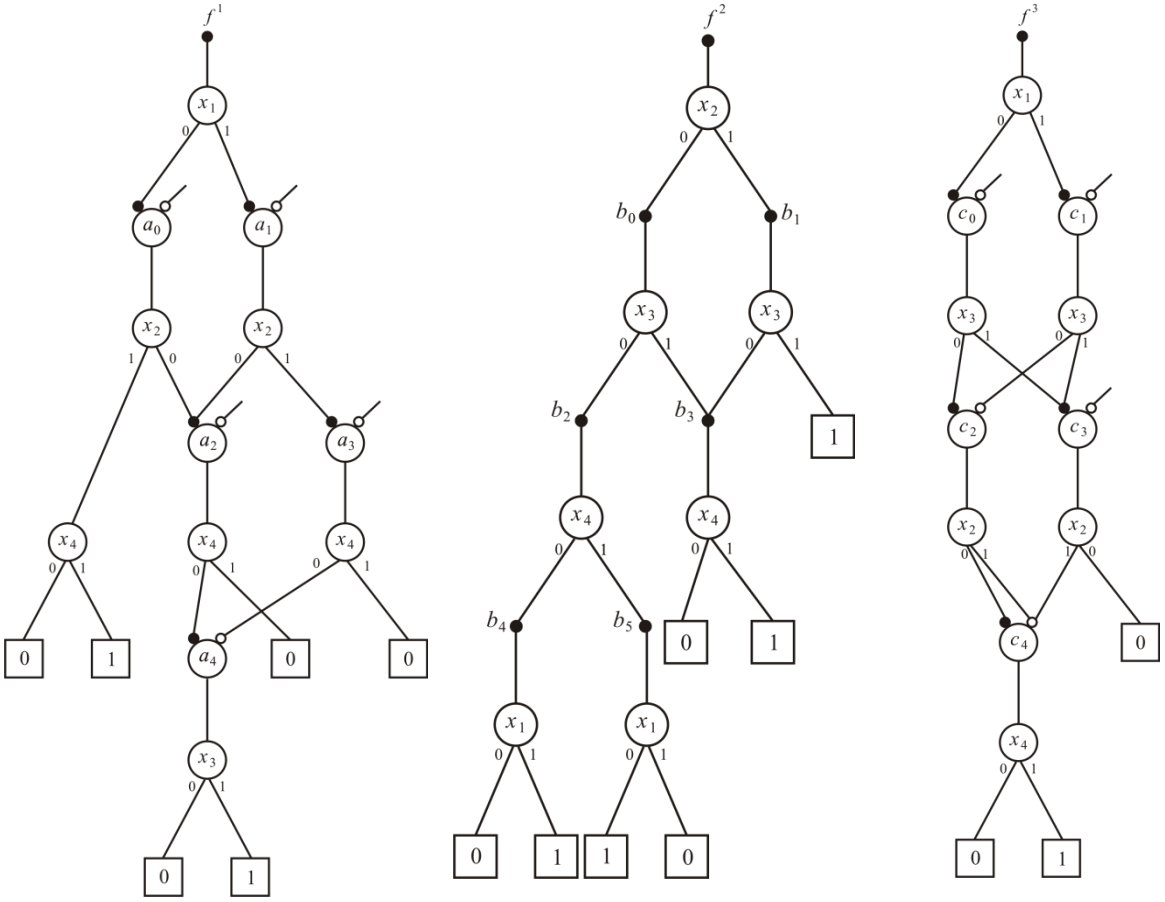
$$f^1 = \bar{x}_1 a_0 \vee x_1 a_1,\ a_0 = \bar{x}_2 a_2 \vee x_2 a_4,\ a_1 = \bar{x}_2 a_2 \vee x_2 a_3,\ a_2 = \bar{x}_4 \bar{a}_4,\ a_3 = \bar{x}_4 a_4,\ a_4 = x_3. \quad (5)$$

Функция f^2 (рис. 2, *б*) представлена в форме классической BDD [7] и реализует многоуровневое представление

$$f^2 = \bar{x}_2 b_0 \vee x_1 b_1,\ b_0 = \bar{x}_3 b_2 \vee x_3 b_3,\ b_1 = \bar{x}_3 b_1 \vee x_3,\ b_2 = \bar{x}_4 b_4 \vee x_4 b_5,\ b_3 = x_4,\ b_4 = x_1,\ b_5 = \bar{x}_1. \quad (6)$$

Функция f^3 (рис. 2, *в*) в форме BDDI имеет многоуровневое представление

$$f^3 = \bar{x}_1 c_0 \vee x_1 c_1,\ c_0 = \bar{x}_3 c_2 \vee x_3 c_3,\ c_1 = \bar{x}_3 \bar{c}_2 \vee x_3 c_3,\ c_2 = \bar{x}_2 c_4 \vee x_2 \bar{c}_4,\ c_3 = x_2 \bar{c}_4,$$
$$c_4 = x_4, b_5 = \bar{x}_1. \quad (7)$$



а) *б)* *в)*

Рис. 2. Раздельные BDDI

Переход от многоуровневого представления к системе ДНФ булевых функций. Преобразование формата LOG в матричное представление системы функций основывается на устранении промежуточных переменных и получении зависимостей компонентных функций от входных переменных (аргументов) x_1, x_2, x_3, x_4. Устранив промежуточные переменные в формулах (5)–(7), можно получить систему функций, эквивалентную заданной в табл. 1. Вместе с тем после осуществления перехода эквивалентная система ДНФ компонентных функций может быть представлена в другом виде (ДНФ могут иметь другую запись).

Разбиение многоуровневых представлений систем булевых функций на подсистемы. Одной из исследуемых оптимизационных процедур является процедура разбиения многоуровневых представлений систем булевых функций на подсистемы ограниченных размеров (блоки). Подобная задача встречается в процессе преобразования логических структур [16]. Известен подход, использующий выделение более крупных логических блоков, которые характеризуются сильной функциональной связью и строятся с использованием различных метрик кластеризации [17, 18].

Процедуру разбиения рассмотрим на графовой модели представления логической сети. Ограничениями на сложность каждого из блоков являются число r входных и число q выходных переменных блока. Текущим представлением логической сети является взвешенный ориентированный граф $G=(V, E)$. Вершине $v_i \in V$ соответствует логическое уравнение. Две вершины v_i, v_j графа G связывает дуга e_{ij}, принадлежащая множеству дуг E (дуга исходит из вершины v_i и заходит в вершину v_j), в том и только в том случае, когда выходная переменная уравнения, соответствующего v_i, является входной переменной уравнения, соответствующего v_j.

Задачу разбиения исходной системы логических уравнений на подсистемы можно представить как задачу разбиения множества V вершин графа G на множество непересекающихся подмножеств вершин $\{V_1,\ldots,V_p\}$ графа (блоков), таких, что $V_i \subset V$, $V_i \neq \varnothing$, $V_i \bigcap\limits_{i \neq j} V_j = \varnothing$, $\bigcup\limits_{i=1}^{p} V_i = V$ для $i, j \in \{1,\ldots,p\}$, $i \neq j$. Вершина графа может иметь числовые характеристики, которыми могут являться, например, числа входных и выходных переменных подсистемы логических уравнений, соответствующих этой вершине. В настоящей работе рассматривается задача разбиения графа, представляющего систему логических уравнений, на минимальное число p блоков V_i при выполнении заданных ограничений на числа входных (r) и выходных (q) переменных блоков V_i. Процедура разбиения реализует последовательный алгоритм формирования блоков.

Этапы построения каждого нового блока:

1. Выбор начальной вершины для блока и исключение ее из исходного множества вершин.
2. Построение множества смежных вершин для блока.
3. Просмотр множества смежных вершин и выбор очередной вершины для включения в блок. Выбранная вершина исключается из исходного множества.
4. Преобразование подсистемы логических уравнений, соответствующей данному текущему состоянию блока, с целью удаления внутренних переменных блока, т. е. устранение внутренних переменных блока.

Этапы 2–4 повторяются до тех пор, пока возможно включение уравнений в подсистему без нарушения заданных ограничений.

Перечисленные этапы можно представить как выбор вершин из исходного графа и объединение их в подграф (блок). Сформированный подграф будет характеризоваться параметрами, равными числу входных (r) и выходных (q) переменных в соответствующей подсистеме. Значения параметров каждого из блоков результирующего разбиения не должны превышать заданных ограничений.

В результате применения процедуры группирования с параметрами для блоков r = 8 (входов) и q = 10 (выходов) для RTL-описания, полученного командой unmap после синтеза схемы по табл. 1, получаем двухблочное разбиение (рис. 3).

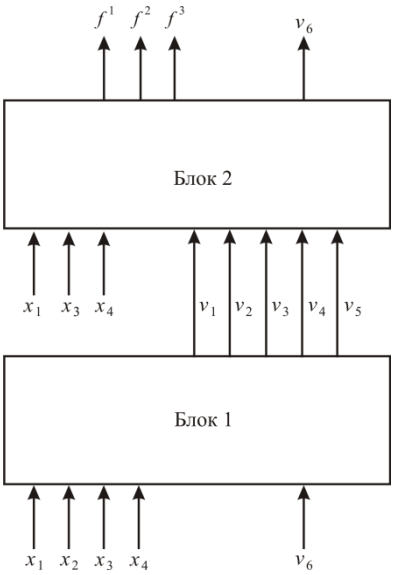


Рис. 3. Разбиение логических уравнений на два блока

Матричные функциональные описания полученных блоков в виде систем ДНФ представлены в табл. 2 и 3.

Таблица 2

Функции блока 1

$x_1\ x_2\ x_3\ x_4\ v_6$	$v_1\ v_2\ v_3\ v_4\ v_5$
0 0 – – 0	0 1 0 0 0
– 0 – 1 0	0 1 0 0 0
0 – – 0 0	0 1 0 0 0
– 1 – 0 0	0 1 0 0 0
0 1 – – 0	0 1 0 0 0
0 1 – – –	0 0 0 1 0
– 1 – 0 –	1 0 0 0 0
– 0 – 0 –	0 0 1 0 0
– – 0 – –	0 0 1 0 0
– – 1 – –	0 0 0 0 1
– 1 – – –	0 0 0 0 1
– – – 1 –	0 0 0 0 1

Таблица 3

Функции блока 2

$x_1\ x_3\ x_4\ v_1\ v_2\ v_3\ v_4\ v_5$	$f^1 f^2 f^3 v_6$
– – 0 – – – 1 –	0 0 1 0
– – – – – – – 0	1 0 0 0
1 1 – 1 – – – –	1 0 1 0
– – – – – 0 – –	0 1 0 0
– 0 – – 0 – – –	0 1 1 0
– – 1 – – – 1 –	1 1 0 0
0 – 1 – – – 0 –	0 0 0 1

Обратим внимание на то, что выходная переменная (функция) $v_6 = \bar{x}_1 x_4 \bar{v}_4$, где $v_4 = \bar{x}_1 x_2$, зависит от переменных x_1, x_2, x_4, поэтому в схеме на рис. 3 нет обратных связей. Данный пример как раз иллюстрирует то, что объединение уравнений в блоки может покрывать груп-

пируемые уравнения различным образом. От параметров группирования зависят число и функциональность получаемых блоков.

Экспериментальное исследование. Общая идея организации экспериментов состояла в том, что различные оптимизационные (технологически независимые) процедуры для разных форм описания систем булевых функций выполнялись на языке SF в системе FLC [15], затем полученные минимизированные описания конвертировались в VHDL-описания и подавались на вход синтезатора LeonardoSpectrum [14], который выполнял построение (синтез) логических схем. Все схемы строились в одной и той же библиотеке КМОП-элементов (табл. 4). Сравнивались результаты исходного и повторного синтезов, осуществляемых с использованием оптимизационных процедур (и без их использования).

Таблица 4

Библиотека 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
LAT	*D*-триггер (по уровню)	20
DFF	*D*-триггер (по переднему фронту)	32
DFFRS	*D*-триггер (по переднему фронту) с установкой и сбросом	44
IBUF	$y = A$, входной буфер	4
OBUF	$y = A$, выходной буфер	10

Программы для экспериментов. Построение совместных BDDI производится программой BDD_Builder [13], применяющей при выборе очередной переменной разложения эвристику «минимальное число различных (с точностью до инверсии) подфункций». Эта программа в качестве исходных данных использует задание системы функций в виде матричного представления системы ДНФ, в языке SF такое представление называется SDF [15]. Построение раздельных BDDI осуществляется в системе FLC с помощью стратегий «декомпозиция по функциям», «BDDI-оптимизация листьев проекта», «устранение иерархии» [15].

Если исходная система функций задана многоуровневым представлением в виде логических уравнений, то для построения BDDI требуется осуществить переход от многоуровневого представления к двухуровневому (так часто в литературе называют представления булевых

функций в виде ДНФ). Такой переход программно реализован в системе FLC и называется ликвидацией внутренних переменных.

В экспериментах для блочного разбиения многоуровневых представлений используется алгоритм «полный с редукцией», представленный в работе [19]. Частным случаем группирования является разбиение системы функций на подсистемы, содержащие только одну компонентную функцию. В системе FLC для матричных форм функций такое разбиение осуществляется с помощью программы «Декомпозиция по функциям».

Примеры для экспериментов. Все примеры для экспериментов были взяты из практики проектирования цифровых схем. В эксперименте 1 сравнение проводилось на восьми примерах (Apex6, C8, Cht, Count, Dalu, Frg2, Too_large, X3) многоуровневых описаний комбинационной логики. В эксперименте 2 были использованы схемные реализации алгоритмических описаний промышленных VHDL-проектов. В экспериментах 3 и 4 сравнение проводилось на потоке из 28 примеров систем ДНФ булевых функций из библиотеки примеров [20] и на двух примерах Verg1, Verg2 матричных форм описания систем ДНФ булевых функций, взятых из практики проектирования.

Управление синтезом в синтезаторе LeonardoSpectrum. Для удобства потоки примеров обрабатывались единообразно с помощью соответствующих скриптов. Пример скрипта, который использовался при синтезе схем в LeonardoSpectrum:

```
clean_all;
set encoding Gray;
set modgen_select Smallest;
set asic_auto_dissolve_limit 500;
set auto_dissolve_limit 500;
read addm4.vhd;
load_library s3lib.syn;
set -hierarchy flatten
set effort standard
optimize -target s3lib -macro -area -effort standard -hierarchy flatten
report_area -cell_usage
```

Если требовалось получить RTL-описание (эксперименты 2 и 3) для повторного синтеза, то в конце скрипта добавлялись две команды:

```
unmap
auto_write addm4.vhd
```

В скрипте кроме значения standard жирным шрифтом выделены также важные для синтеза значения flatten параметра hierarchy и значение 500 параметра asic_auto_dissolve_limit. Установка значения flatten параметра hierarchy ориентирует синтезатор на выполнение синтеза с учетом устранения иерархии описания (схема синтезируется в виде одного блока), в этом случае имеется больше возможностей для оптимизации. Параметр asic_auto_dissolve_limit задает число «растворяемых» элементов при оптимизации. «Растворение» понимается как преобразование структурного описания подсхемы в функциональное. По умолчанию значение этого параметра равно 30. При эксперименте задается значение 500. При таком значении результирующие схемы имеют меньшую площадь, оптимизируются подсхемы большей размерности. Задание значения Gray параметра encoding важно при синтезе схем с памятью, схемы получаются более экономичными.

Эксперимент 1. Исходными данными являлись многоуровневые функциональные описания комбинационных схем в виде взаимосвязанных логических уравнений (формат LOG) языка SF. Для каждого из данных примеров строились три варианта логических схем и оценивалась их сложность (площадь S) в числе транзисторов.

Вариант 1.1. Исходное SF-описание переводилось в VHDL-описание и реализовывалось без выполнения предварительной оптимизации. Назовем этот вариант непосредственной схемной реализацией.

Вариант 1.2. Исходное SF-описание переводилось в систему ДНФ (внутренние переменные в описании устранялись). Затем выполнялась логическая оптимизация на основе разложения Шеннона, SF-описание переводилось в VHDL-описание, после чего синтезировалась логическая схема. Назовем этот вариант глобальной схемной реализацией на основе BDDI.

Вариант 1.3. Исходное SF-описание подвергалось процедуре разбиения на блоки, каждый блок SF-описания представлялся в виде системы ДНФ. Затем выполнялась логическая оптимизация на основе разложения Шеннона, SF-описание переводилось в VHDL-описание, после чего синтезировалась логическая схема. Назовем этот вариант схемной реализацией на основе блочного разбиения.

Эксперимент 2. Повторный синтез комбинационных частей схем, реализующих алгоритмические описания. Исходными данными являлись синтезируемые алгоритмические VHDL-описания цифровых устройств.

Вариант 2.1. После синтеза схемы по алгоритмическому VHDL-описанию оценивалась сложность комбинационной части схемы, структурное описание схемы (нетлист) переводилось в SF-описание. Затем элементы памяти выделялись в отдельный блок, логические комбинационные элементы – в комбинационную часть схемы и устранялась иерархия описания. Полученные логические уравнения представляли собой многоуровневое RTL-описание функций комбинационной части схемы с памятью.

Вариант 2.2. Повторный синтез комбинационного блока без оптимизации. Назовем такой вариант повторным синтезом от RTL-описания.

Вариант 2.3. Схемная реализация комбинационной части на основе блочного разбиения (см. вариант 1.3).

Эксперимент 3. Сравнение результатов синтеза от исходных матричных описаний комбинационных схем в виде систем ДНФ с результатами повторного синтеза от RTL-описаний и с результатами схемной реализации RTL-описаний на основе блочного разбиения. Исходными данными являлись матричные описания систем ДНФ (формат SDF). Каждый раз перед процедурой синтеза осуществлялась конвертация SF-описания схемы в VHDL-описание.

Выполнялись три варианта синтеза.

Вариант 3.1. Синтез по неоптимизированным исходным матричным представлениям систем ДНФ.

Вариант 3.2. Повторный синтез от RTL-описания схемы. RTL-описания были получены командой unmap после синтеза в библиотеке POWER [15].

Вариант 3.3. Разбиение RTL-описания на блоки, которые имеют восемь входов и два выхода, раздельная BDDI-оптимизация каждого из блоков, схемная реализация полученных оптимизированных представлений. В качестве исходных для варианта 3.3 были взяты те же RTL-описания, что и для варианта 3.2.

Эксперимент 4. Сравнение эффективности использования процедур оптимизации на основе совместных и раздельных BDDI. Исходными данными являлись матричные описания систем ДНФ (формат SDF).

Вариант 4.1. Конвертация исходного (неоптимизированного) SF-описания в VHDL-описание и его схемная реализация.

Вариант 4.2. Глобальная схемная реализация на основе оптимизированных совместных BDDI: оптимизация на основе совместной BDDI, конвертация BDDI в VHDL-описание и схемная реализация.

Вариант 4.3. Глобальная схемная реализация на основе оптимизированных раздельных BDDI: оптимизация на основе раздельных BDDI, конвертация раздельных BDDI в VHDL-описания и схемная реализация.

Рассмотрим результаты некоторых экспериментов (табл. 5) для примера системы функций из табл. 1. Синтезированная по варианту 4.2 логическая схема оказалась лучшей по площади, поскольку логические элементы, из которых она состоит, содержат наименьшее суммарное число транзисторов.

Таблица 5

Результаты экспериментов для системы ДНФ трех булевых функций

Варианты	Исходные данные	Площадь схемы по числу транзисторов, S	Число элементов схемы, L
4.1 и 3.1	Система ДНФ функций (см. табл. 1)	86	19
4.2	Совместная BDDI (см. (5))	74	17
4.3	Раздельные BDDI (см. (6)–(8))	86	18
3.3	BDDI, построенные по подсистемам функций (см. табл. 2 и 3)	78	15

Результаты экспериментов. Результаты проведенных четырех экспериментов представлены в табл. 6–9, где n – число аргументов булевых функций; m – число функций; k – число элементарных конъюнкций в системе ДНФ булевых функций; S – площадь логической схемы, задаваемая в числе транзисторов, из которых состоят элементы, лучшие решения выделены жирным шрифтом; L – число элементов логической схемы. Два последних столбца в табл. 6 и 7 относятся к варианту 1.3 (табл. 6) и варианту 2.2 (табл. 7) соответственно.

Таблица 6

Результаты эксперимента 1

Пример	n	m	Площадь схемы, S			Число блоков, p	Ограничения на параметры r, q блоков
			Вариант 1.1	Вариант 1.2	Вариант 1.3		
Apex6	135	94	**1830**	2110	1874	13	25, 100
C8	28	18	**312**	324	318	9	10, 100
Cht	47	36	680	670	**652**	7	10, 100
Count	35	16	256	256	256	8	8, 100
Dalu	75	16	1834	1396	**1286**	23	30, 100
Frg2	143	139	4224	–	**3800**	102	10, 100
Too_large	38	3	10902	**9038**	–	–	–
X3	135	99	3462	3788	**3154**	87	8, 100
Число лучших решений по площади			2	1	**4**		

Таблица 7

Результаты эксперимента 2

Пример	n	m	Площадь схемы, S			Число блоков, p	Ограничения на параметры r, q блоков
			Вариант 2.1	Вариант 2.2	Вариант 2.3		
Alu_avr	205	13	**1472**	1504	1700	40	15, 100
Alu_1	38	32	**10934**	11110	11340	323	10, 100
Timer	84	42	766	**424**	**424**	24	10, 100
VGA	150	106	2891	**2604**	3076	40	15, 100
simmetria	53	20	15390	**15006**	15988	361	10, 100
Число лучших решений по площади			2	**3**	0		

Таблица 8

Результаты эксперимента 3, параметры блоков: восемь входов, два выхода

Пример	*n*	*m*	*k*	Площадь схемы, *S*		
				Вариант 3.1	Вариант 3.2	Вариант 3.3
Add6	12	7	1092	5750	**4414**	4830
Addm4	9	8	512	**4104**	4108	4134
B12	15	9	431	258	262	**236**
B2	15	17	110	2506	**2418**	2430
B9	16	5	123	404	402	**392**
Dc2	8	7	58	452	372	**354**
Dk48	15	17	148	394	394	**390**
In0	15	11	138	2134	**1926**	1934
In2	19	10	137	2034	1822	**1810**
Intb	15	7	664	**7340**	7502	7946
M2	8	16	96	976	**896**	938
M3	8	16	128	1344	**1194**	1236
Newcpla1	9	16	38	428	382	**376**
Newtpla	15	5	23	212	**210**	214
P82	5	14	24	348	**346**	354
Root	8	5	256	958	**900**	**900**
Shift	19	16	100	**806**	1216	1186
Sqn	7	3	96	502	**372**	446
Sym10	10	1	837	1374	996	**760**
Tial	14	8	640	**5704**	5740	6024
Z5xp1	7	10	128	1644	**1552**	1568
Z9sym	9	1	420	1022	820	**658**
Число лучших решений по площади				4	**10**	9

Согласно результатам исследования эффективной процедурой технологически независимой оптимизации является глобальная оптимизация на основе совместных BDDI в тех случаях, когда от многоуровневого RTL-описания в виде логических уравнений удается перейти к матричному описанию и воспользоваться программой оптимизации совместных BDDI. Если же не удается перейти к такому описанию, то другой маршрут проектирования, а именно предварительное разбиение системы функций на подсистемы и последующее построение совместных BDDI (эксперимент 1), позволяет уменьшить площадь схемы. При этом уменьшение параметров блоков, на которые разбиваются логические уравнения, позволяет добиваться лучших результатов по площади.

В экспериментах 2 и 3 был использован метод повторного синтеза. Результаты эксперимента 3 показали, что повторный синтез комбинационных частей схем для проектов, содержащих элементы памяти, либо дает небольшой выигрыш, либо такой выигрыш (уменьшение площади) получить не удается, при этом процедура разбиения на блоки только ухудшает решение. Смена библиотеки POWER на новую библиотеку (см. табл. 4) не изменяет общую ситуацию при синтезе: если оптимизация проведена хорошо, то для различных библиотек площадь схемы уменьшается по сравнению со схемой, получаемой по некачественному результату технологически независимой оптимизации.

Таблица 9

Результаты эксперимента 4

Пример	*n*	*m*	*k*	Вариант 4.1		Вариант 4.2		Вариант 4.3	
				S	*L*	*S*	*L*	*S*	*L*
addm4	9	8	512	4104	758	**1332**	241	1488	263
b12	15	9	431	**258**	48	262	50	306	59
b2	16	17	110	**2506**	516	5902	1046	3524	655
b9	16	5	123	404	79	440	90	**402**	82
in2	19	10	137	2034	392	1674	303	**1260**	238
intb	15	7	664	7340	1209	**4588**	781	4866	862
m181	15	9	430	**266**	49	276	51	300	57
mp2d	11	14	123	558	114	268	57	**262**	59
root	8	5	256	958	185	474	84	**406**	84
soar	83	94	529	2462	488	2204	432	**1862**	387
t3	12	8	152	292	57	252	53	**250**	52
tial	14	8	640	5704	952	8010	1384	**4704**	805
Verg1	17	61	2003	10284	1633	17282	2928	**6996**	1274
Verg2	18	63	2129	–	–	**9600**	1541	10648	1711
vtx1	27	6	110	**282**	60	574	117	408	87
x9dn	27	7	120	**286**	61	554	120	408	87
Число лучших решений по площади				5		3		**8**	

Согласно результатам эксперимента 4 применение совместных BDDI для исходных матричных форм задания функций является более предпочтительным приемом технологически независимой минимизации при синтезе логических схем из библиотечных элементов, так как площадь схем, построенных по совместным BDDI, в подавляющем числе случаев меньше площади схем, построенных по раздельным BDDI.

Заключение. Предварительная оптимизация исходных функциональных описаний систем булевых функций дает уменьшение площади при синтезе КМОП-схем, при этом конкурирующими методами являются совместные и раздельные BDDI, что аналогично ситуации с совместной и раздельной минимизацией систем булевых функций в классе ДНФ.

Повторные оптимизация (с использованием имеющихся программ) и синтез по RTL-описаниям, полученным по уже синтезированным схемам, практически не имеют смысла, так как могут лишь незначительно уменьшить площадь схемы либо даже увеличить ее. Такой вывод справедлив как для комбинационных схем (без элементов памяти), так и для схем с памятью и обусловлен тем, что в RTL-описаниях большой размерности, составленных из «мелких» уравнений, трудно выделить «связные» функциональные блоки и получить другое многоуровневое представление, по которому повторный синтез будет проведен более эффективно, чем синтез по исходному многоуровневому описанию, полученному в результате замены алгоритмических конструкций языка VHDL логическими уравнениями.

Список использованных источников

1. 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.

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

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

4. Закревский, А. Д. Логический синтез каскадных схем / А. Д. Закревский. – М. : Наука, 1981. – 416 с.

5. Sasao, T. FPGA design by generalized functional decomposition / T. Sasao // Representations of Discrete Functions ; ed. by T. Sasao, M. Fujita. – Kluwer Academic Publishers, 1996. – P. 233–258.

6. Scholl, C. Functional Decomposition with Applications to FPGA Synthesis / C. Scholl. – Kluwer Academic Publishers, 2001. – 288 p.

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

8. 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.

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

10. Meinel, C. Algorithms and Data Structures in VLSI Design: OBDD – Foundations and Applications / C. Meinel, T. Theobald. – Berlin, Heidelberg : Springer-Verlag, 1998. – 267 p.

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

12. Валидация на системном уровне. Высокоуровневое моделирование и управление тестированием / М. Чэнь [и др.]. – М. : Техносфера, 2014. – 296 с.

13. 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.

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

15. 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.

16. Григорьян, С. Г. Конструирование электронных устройств систем автоматизации и вычислительной техники / С. Г. Григорьян. – Ростов н/Д : Феникс, 2007. – 303 с.

17. Кузовлев, В. И. Выявление высокоуровневых иерархических структур сверхбольших интегральных схем через сильно связанные логические группы / В. И. Кузовлев, Н. А. Иванова // Вестник МГТУ им. Н. Э. Баумана. Сер. Приборостроение. – 2016. – № 4. – С. 4–18.

18. Netlist and system partitioning / A. B. Kahng [et al.] // VLSI Physical Design: From Graph Partitioning to Timing Closure. – Springer, 2011. – Chap. 2. – P. 31–54.

19. Бибило, П. Н. Оптимизационные преобразования логической схемы на основе блочного разбиения / П. Н. Бибило, Н. А. Кириенко // Информатика. – 2009. – № 3(23). – С. 5–15.

20. Jeong, C. Computer-Aided Design of Digital Systems / C. Jeong // Department of Computer Science [Electronic resource]. – Mode of access: http://www1.cs.columbia.edu/~cs6861/sis/espresso-examples/ex. – Date of access: 20.03.2018.

References

1. Brayton R. K., Hachtel G. D., Sangiovanni-Vincentelli A. L. Sintez mnogourovnevyh kombinacionnyh logicheskih skhem [Multilevel Logic Synthesis]. TIIJeR [*TIIER*], 1990, vol. 78, no. 2, pp. 38–83 (in Russian).

2. Polyakov A. K. Yazyki VHDL i VERILOG v proektirovanii tsifrovoi apparatury. *The VHDL and VERILOG Languages in Designing of Digital Hardware*. Moscow, SOLON-Press Publ., 2003, 320 p. (in Russian).

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

4. Zakrevskij A. D. Logicheskij sintez kaskadnyh skhem. *Logical Synthesis of Cascade Circuits*. Moscow, Nauka Publ., 1981, 416 p. (in Russian).

5. Sasao T. FPGA design by generalized functional decomposition. *Representations of Discrete Functions.* Kluwer Academic Publishers, 1996, pp. 233–258.

6. Scholl C. *Functional Decomposition with Applications to FPGA Synthesis*. Kluwer Academic Publishers, 2001, 288 p.

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

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

9. Bryant R. E., Meinel C. Ordered binary decision diagrams. *Logic Synthesis and Verification.* Kluwer Academic Publishers, 2002, pp. 285–307.

10. Meinel C., Theobald T. *Algorithms and Data Structures in VLSI Design: OBDD – Foundations and Applications*. Berlin, Heidelberg, Springer-Verlag Publ., 1998, 267 p.

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

12. Chen M., Qin K., Ku H.-M., Mishra P. Validaciya na sistemnom urovne. Vysokourovnevoe modelirovanie i upravlenie testirovaniem. *Validation at the System Level. High-Level Simulation and Testing Management*. Moscow, Tekhnosfera Publ., 2014, 296 p. (in Russian).

13. 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).

14. 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 Publ., 2005, 384 p. (in Russian).

15. 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, Belarus. Navuka Publ., 2011, 279 p. (in Russian).

16. Grigor'yan S. G. Konstruirovanie jelektronnyh ustrojstv sistem avtomatizacii i vychislitel'noj tekhniki. *Design of Electronic Devices of Automation Systems and Computers*. Rostov n/D, Feniks Publ., 2007, 303 p. (in Russian).

17. Kuzovlev V. I., Ivanova N. A. Vyyavlenie vysokourovnevyh ierarhicheskih struktur sverhbol'shih integral'nyh skhem cherez sil'no svyazannye logicheskie gruppy [Circuit detection through tangled logic structures]. Vestnik MGTU im. N. E. Baumana, ser. Priborostroenie [*Herald of the Bauman Moscow State Tech. Univ., Instrum. Eng*.], 2016, no. 4, pp. 4–18 (in Russian).

18. Kahng A. B., Liening J., Markov I. L., Hu J. Netlist and system partitioning. *VLSI Physical Design: From Graph Partitioning to Timing Closure*. Springer, 2011, ch. 2, pp. 31–54.

19. Bibilo P. N., Kirienko N. A. Optimizacionnye preobrazovaniya logicheskoj skhemy na osnove blochnogo razbieniya [Optimizing conversions of a logic circuit by partitioning into blocks]. Informatika [*Informatics*], 2009, no. 3, pp. 5–15 (in Russian).

20. 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).

Информация об авторах

Бибило Петр Николаевич – доктор технических наук, профессор, Объединенный институт проблем информатики Национальной академии наук Беларуси (ул. Сурганова, 6, 220012, Минск, Республика Беларусь). E-mail: bibilo@newman.bas-net.by

Кириенко Наталья Алексеевна – кандидат технических наук, доцент, Объединенный институт проблем информатики Национальной академии наук Беларуси (ул. Сурганова, 6, 220012, Минск, Республика Беларусь). E-mail: kir@newman.bas-net.by

Ланкевич Юрий Юрьевич – младший научный сотрудник, Объединенный институт проблем информатики Национальной академии наук Беларуси (ул. Сурганова, 6, 220012, Минск, Республика Беларусь). E-mail: yurafreedom18@gmail.com

Information about the authors

Petr N. Bibilo – D. Sc. (Engineering), Professor, The United Institute of Informatics Problems of the National Academy of Sciences of Belarus (6, Surganova Str., 220012, Minsk, Republic of Belarus).

E-mail: bibilo@newman.bas-net.by

Natalia A. Kirienko – Ph. D. (Engineering), The United Institute of Informatics Problems of the National Academy of Sciences of Belarus (6, Surganova Str., 220012, Minsk, Republic of Belarus). E-mail: kir@newman.bas-net.by

Yury Y. Lankevich – Researcher, The United Institute of Informatics Problems of the National Academy of Sciences of Belarus (6, Surganova Str., 220012, Minsk, Republic of Belarus). E-mail: yurafreedom18@gmail.com