УДК 519.7

П.Н. Бибило, П.В. Леончик

Алгоритм построения диаграммы двоичного выбора для системы полностью определенных булевых функций

Предложен алгоритм оптимизации многоуровневых представлений систем ДНФ полностью определенных булевых функций на основе построения диаграмм двоичного выбора. Приведены результаты экспериментального исследования этого алгоритма, используемого в качестве предварительного оптимизационного этапа синтеза комбинационных схем в библиотеках проектирования базовых матричных кристаллов и логических схем, реализуемых в составе *FPGA*.

The algorithm of optimization of multilevel representations of DNF systems of the completely defined Boolean functions based on the construction of binary decision diagrams is suggested. The results of the experimental research of this algorithm which is used as a preliminary optimization stage of the synthesis of combinational circuits in the design library of Gate Arrays and logical circuits implemented in the FPGA, are presented.

Запропоновано алгоритм оптимізації багаторівневих представлень систем ДНФ повністю визначених бульових функцій на основі побудови діаграм двійкового вибору. Наведено результати експериментального дослідження цього алгоритму, який використано як попередній оптимізаційний етап синтезу комбінаційних схем у бібліотеках проектування базових матричних кристалів та логічних схем, які реалізуються в складі *FPGA*.

Введение. Синтез комбинационных логических схем традиционно разбивается на два больших этапа: глобальную технологически независимую оптимизацию и технологическое отображение (*technology mapping*). В качестве технологически независимой оптимизации чаще всего применяется совместная либо раздельная минимизация системы булевых функций в классе дизъюнктивных нормальных форм (ДНФ), позволяющая получать оптимизированные двухуровневые (и/или) представления функций. Оптимизированные многоуровневые представления в виде скобочных алгебраических форм функций получаются на основе факторизационных методов оптимизации, а также многократного применения разложения Шеннона – такие представления получили названия бинарных программ [1] или диаграмм двоичного выбора (*BDD – Binary Decision Diagram*) [2]. *BDD* оказались эффективной формой представления функций и получили широкое применение [3–6]. Основная проблема при построении *BDD* – выбор последовательности переменных, по которой ведется разложение Шеннона.

В данной статье предлагается алгоритм построения *BDD* для системы полностью определенных булевых функций. Функции системы задаются в виде ДНФ, особенности алгоритма – «блоковый» подход к проведению разложений Шеннона при выбранной последовательности переменных разложения и сочетание различных эвристик при выборе такой последовательности. Экспериментально исследуется эффективность предложенного алгоритма для синтеза комбинационных схем в базисе элементов отечественных базовых матричных кристаллов (БМК) и структур программируемых логических интегральных схем типа *FPGA* (*Field-Programmable Gate Arrays*).

Диаграмма двоичного выбора (*BDD*)

Разложением Шеннона булевой функции $f(x_1,...,x_n)$ по переменной x_i называется представление $f(x_1,...,x_n)$ в виде

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

Функции $f(x_1,...,x_{i-1},1,x_{i+1},...x_n)$, $f(x_1,...,x_{i-1},0,x_{i+1},...x_n)$ в (1) называются *коэффициентами* разложения. Они получаются из функции $f(x_1,...,x_n)$ подстановкой вместо переменной x_i константы 1 и 0 соответственно. Очевидно, что если коэффициенты равны, то $f(x_1,...,x_n) = f(x_1,...,x_{i-1},x_{i+1},...x_n)$. Переменную x_i называют в этом случае несущественной или *фиктивной* переменной функции $f(x_1,...,x_n)$. Каждый из *коэффициентов* $f(x_1,...,x_{i-1},1,x_{i+1},...x_n)$, $f(x_1,...,x_{i-1},0,x_{i+1},...x_n)$ может быть разложен по одной из переменных из множества $\{x_1,...,$

$x_{i-1}, x_{i+1}, ...x_n\}$. Процесс разложения коэффициентов заканчивается, когда все n переменных будут использованы для разложения. На последнем шаге разложения коэффициенты вырождаются до констант 0, 1.

Под *диаграммой двоичного выбора*, т.е. под *BDD*, понимается граф, задающий разложение Шеннона булевой функции $f(x_1,...,x_n)$ по всем ее переменным $x_1,...,x_n$ при заданном порядке (перестановке) переменных, по которым проводится разложение. В статье рассматриваются *BDD* для системы функций, при этом перестановки переменных, по которым ведутся разложения, одинаковы для всех функций системы. Пример *BDD* для системы, состоящей из трех функций, приведены на рис. 1. *BDD* содержит три вида вершин:

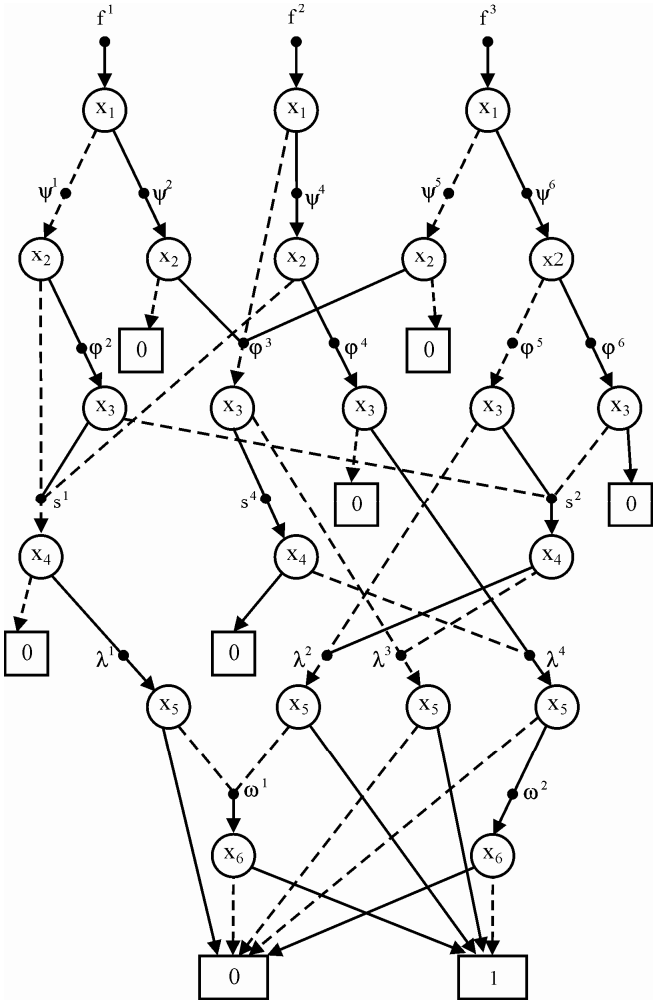

Рис. 1

• *функциональные*, соответствующие разлагаемым функциям (из них три корневые вершины-функции f^1, f^2, f^3);

• *вершины–переменные*, т.е. вершины, соответствующие переменным разложения;

• *листовые*, соответствующие константным (0, 1) значениям функций.

По диаграмме двоичного выбора легко записывается многоуровневое представление булевой функции, так как каждой паре <функциональная вершина, вершина-переменная> соответствует разложение (1) Шеннона некоторой функции, при этом различные разлагаемые функции могут иметь одинаковые коэффициенты разложения.

Примечание. При изображении *BDD* функциональные вершины обычно не показываются, так как предполагается, что всем дугам, ведущим в вершину-переменную, соответствует одна и та же булева функция, разлагаемая по этой переменной.

Под *сложностью BDD* понимается число функциональных вершин. Вершины-переменные и листовые вершины не учитываются при подсчете сложности *BDD*.

Алгоритм построения *BDD* для системы функций

Постановка задачи

Задана система $F(x_1,...,x_n) = (f^1(x_1,...,x_n),\ldots, f^m(x_1,...,x_n))$ ДНФ полностью определенных булевых функций. Требуется построить *BDD* минимальной сложности для системы F функций.

Решение задачи традиционно разбивается на два этапа.

Этап 1. Выбор последовательности (перестановки) переменных $x_1,...,x_n$, по которой ведется разложение Шеннона.

Этап 2. Построение *BDD* по заданной перестановке переменных разложения.

Опишем сначала алгоритм этапа 2 – построение *BDD* системы $F(x_1,...,x_n) = (f^1(x_1,...,x_n),\ldots, f^m(x_1,...,x_n))$ при заданной перестановке множества $X=\{x_1,...,x_n\}$ переменных. Основная

идея алгоритма заключается в том, что переменные разложения группируются в блоки, разложение ведется по всем переменным очередного блока без сравнения коэффициентов, сравнение коэффициентов на равенство осуществляется тогда, когда проведено разложение по последней переменной блока. При этом сравнение коэффициентов сначала проводится «грубо» (два коэффициента считаются равными, если они заданы на одном и том же множестве элементарных конъюнкций). Окончательное сравнение коэффициентов на равенство осуществляется при «обратном проходе».

Алгоритм этапа 2 состоит из следующих шагов:

Ш а г 1. Разбиение множества X на попарно непересекающиеся блоки (подмножества) Y^1, $Y^2,...,Y^q$.

1.1. Если $n > 20$, то разбиваем X на q блоков $Y^1,Y^2,...,Y^q$, таких, что $\left|Y^i\right| = 3$, $i = 2, 3, \ldots, q$, где $\left|Y^i\right|$ – мощность блока Y^i, а $\left|Y^i\right| \le 5$.

1.2. Если же $n \le 20$, то разбиение переменных на блоки осуществляется следующим образом. Пусть k – число различных элементарных конъюнкций, входящих в ДНФ всех функций системы F. Если $3k < 2^n$, то разбиваем X на два блока Y^1,Y^2, причем $\left|Y^1\right| = n-3$, $\left|Y^2\right| = 3$. Если $3k > 2^n$, то разбиение на блоки осуществляется так же, как и в п.1.1.

Ш а г 2 (итерационный). Построение коэффициентов разложения Шеннона для функций системы F по переменным блока Y^i $i = 1, 2, \ldots, q-1$ и сравнение их на равенство.

Сравниваемые коэффициенты разложения функций системы F задаются в виде ДНФ на «остатках» исходных элементарных конъюнкций. Из ДНФ, представляющих коэффициенты, удаляются поглощаемые конъюнкции. Затем строится общий список всех конъюнкций (троичных векторов), на которых заданы все коэффициенты, и каждая ДНФ задается множеством номеров из общего списка. Если имеется троичный вектор, состоящий только из неопределенных элементов «–», то все ДНФ, в которые он попадает, полагаются равными единице. Это приближенная проверка на равенство единице рассматриваемой ДНФ, так как очевидно, что покрыть все булево пространство может не только один вектор (–,…–), такое покрытие может осуществить несколько троичных векторов, состоящих как из неопределенных, так и определенных (0,1) компонент. После того, как все ДНФ заданы на общем списке конъюнкций, осуществляется проверка ДНФ на равенство – именно эта процедура позволяет сократить число функциональных вершин BDD. Если множества номеров конъюнкций двух ДНФ имеют неравную мощность, то такие ДНФ не сравниваются. Сравнение ДНФ, заданных на равномощных множествах номеров конъюнкций, сводится к проверке равенства множеств номеров: ДНФ считаются равными, если они заданы на одном и том же множестве номеров. Естественно, такой способ сравнения ДНФ приближенный, так как некоторые ДНФ могут быть равными, но состоять из различных подмножеств элементарных конъюнкций. Как уже говорилось, точная проверка коэффициентов на равенство будет сделана позднее.

В результате выполнения шага 2 коэффициенты разложения будут представлять собой функции $f(Y^q)$, зависящие от трех переменных, входящих в блок Y^q.

Ш а г 3. Построение коэффициентов разложения Шеннона по переменным блока Y^q и точное сравнение их на равенство. На этом шаге осуществляется представление функций $f(Y^q)$ таблицами истинности, после чего восьмикомпонентный вектор-столбец значений функции интерпретируется как число, заданное в двоичной системе счисления. Например, вектор 00001001 интерпретируется как число девять. Среди полученных коэффициентов находятся равные (одинаковые) коэффициенты. Так как каждой из $2^{2^3} = 256$ булевых функций, зависящих от трех аргументов, соответствует единственное десятичное число, то сравнение коэффициентов $f(Y^q)$ сводится к проверке равенства чисел, представляющих эти коэффициенты (функции), и осуществляется быстро.

В результате выполнения шага 3 строится *BDD*, однако в ней могут быть вершины, соответствующие одинаковым коэффициентам.

Ш а г 4 (итерационный). Сокращение числа вершин *BDD*, соответствующих переменным блоков Y^1, Y^2,…, Y^{q-1}, начиная с переменных блока Y^{q-1}. Переменные блоков Y^1, Y^2,…, Y^{q-1} рассматриваются в порядке, обратном порядку разложения. Проверка на равенство коэффициентов $s^1(x_1,...,x_j)$, $s^2(x_1,...,x_j)$, зависящих от j переменных, сводится к проверке на равенство упорядоченных пар $<s_0^1(x_1,...,x_{j-1})$, $s_1^1(x_1,...,x_{j-1})>$, $<s_0^2(x_1,...,x_{j-1})$, $s_1^2(x_1,...,x_{j-1})>$ коэффициентов, зависящих от $j-1$ переменной: если в парах равны компоненты с одинаковыми номерами, т.е. если $s_0^1(x_1,...,x_{j-1}) = s_0^2(x_1,...,x_{j-1})$ и $s_1^1(x_1,...,x_{j-1}) = s_1^2(x_1,...,x_{j-1})$, то и коэффициенты $s^1(x_1,...,x_j)$, $s^2(x_1,...,x_j)$ равны.

Пример. Проиллюстрируем алгоритм на примере системы F, состоящей из трех функций f^1, f^2, f^3 (табл. 1). Пусть последовательность разложения является $<x_1,x_2,x_3,x_4,x_5,x_6>$.

Таблица 1

T^x						B^f		
x_1	x_2	x_3	x_4	x_5	x_6	f^1	f^2	f^3
1	1	–	0	1	0	1	0	0
0	–	–	1	0	1	1	0	0
0	–	–	0	1	0	0	1	0
0	–	0	–	1	–	0	1	0
1	1	1	–	1	0	0	1	0
1	0	–	1	0	1	0	1	0
1	0	0	–	–	1	0	0	1
1	0	–	1	–	1	0	0	1
1	–	0	1	–	1	0	0	1
0	1	–	0	1	0	0	0	1
1	0	–	–	1	–	0	0	1
–	1	0	–	1	–	1	0	1

Ш а г и 1 – 2. Последовательность переменных разобьется на два блока $Y^1 = <x_1,x_2,x_3>$, $Y^2 = <x_4,x_5,x_6>$. Проведем разложение Шеннона по переменным блока Y^1, ненулевые коэффициенты разложения, полученные непосредственной подстановкой наборов констант вместо переменных разложения, заданы в табл. 2. Коэффициенты разложения зададим на множестве $p_1, \ldots, p_6$ различных элементарных конъюнкций, всего получается семь коэффициентов $s^1, \ldots, s^7$ (табл. 3), не считая коэффициента s^0, равного нулю. После задания коэффициентов (функций) s^i, зависящих от трех переменных x_4, x_5 x_6, на наборах булева пространства, выясняется, что $s^2 = s^7$, поэтому функция s^7 удаляется из рассмотрения. Начальный вид *BDD*, полученной после разложения по переменным блока Y^1 и сравнения коэффициентов, показан на рис. 2.

Таблица 2

p_i	$x_4\,x_5\,x_6$	Коэффициент		Функция
		f_j^i	s^i	
p_1	101	f_{000}^1	s^1	
p_1	101	f_{001}^1	s^1	f^1
p_1, p_3	101 –1–	f_{010}^1	s^2	
p_1	101	f_{011}^1	s^1	
p_2	010	f_{110}^1	s^4	
p_2	010	f_{111}^1	s^4	
p_3	–1–	f_{000}^2	s^3	f^2
p_2	010	f_{001}^2	s^4	
p_3	–1–	f_{010}^2	s^3	
p_2	010	f_{011}^2	s^4	
p_1	101	f_{100}^2	s^1	
p_1	101	f_{101}^2	s^1	
p_6	–10	f_{111}^2	s^5	
p_3	–1–	f_{010}^3	s^3	f^3
p_2	010	f_{011}^3	s^4	
p_3, p_4, p_5	–1– ––1 1–1	f_{100}^3	s^6	
p_3, p_5	–1– 1–1	f_{101}^3	s^7	
p_3, p_5	–1– 1–1	f_{110}^3	s^7	

Ш а г 3. Разложения функций $s^1,\ldots, s^7$ по переменным блока $Y^2 = <x_4,x_5,x_6>$. Результаты разложения проиллюстрированы на рис. 3, а соответствующее многоуровневое представление имеет вид:

$$s^1 = x_4\lambda^1;\ s^2 = \overline{x_4}\lambda^3 \vee x_4\lambda^2;\ s^3 = \overline{x_4}\lambda^3 \vee x_4\lambda^3;$$
$$s^4 = \overline{x_4}\lambda^4;\ s^5 = \overline{x_4}\lambda^4 \vee x_4\lambda^4;\ s^6 = \overline{x_4}\lambda^2 \vee x_4\lambda^2;$$
$$\lambda^1 = \overline{x_5}\omega^1;\ \lambda^2 = \overline{x_5}\omega^1 \vee x_5;\ \lambda^3 = x_5;\ \lambda^4 = x_5\omega^2;$$
$$\omega^1 = x_6;\ \omega^2 = \overline{x_6}.$$

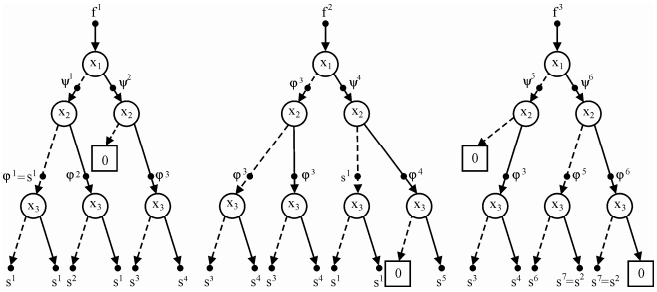

Рис. 2

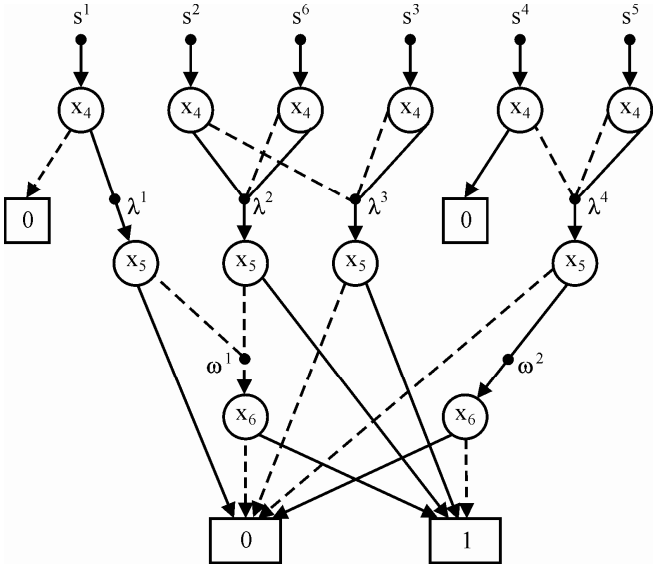

Рис. 3

Таблица 3

$x_4\,x_5\,x_6$	s^0	s^1	s^2	s^3	s^4	s^5	s^6	$s^7=s^2$
000	0	0	0	0	0	0	0	0
001	0	0	0	0	0	0	1	0
010	0	0	1	1	1	1	1	1
011	0	0	1	1	0	0	1	1
100	0	0	0	0	0	0	0	0
101	0	1	1	0	0	0	1	1
110	0	0	1	1	0	1	1	1
111	0	0	1	1	0	0	1	1
p_i		p_1	p_1, p_3	p_3	p_2	p_6	$p_3,\ p_4,\ p_5$	p_3, p_5

Ш а г 4. Сокращение числа вершин *BDD*. Процесс рассмотрения вершин осуществляется от листовых вершин вверх по дереву – к корневым вершинам, помеченным исходными функциями f^1, f^2, f^3. На рис. 3 показано, что разложение коэффициента (функции) s^6 по переменной x_4 дает два одинаковых коэффициента λ^2, следовательно, переменная x_4 несущественна для s^6 и потому $s^6=\lambda^2$. Переменная s^6 исключается из многоуровневого представления функций. Аналогично: $s^3=\lambda^3$, $s^5=\lambda^4$, переменные s^3, s^5 исключаются из многоуровневого представления функций. Каждая различная упорядоченная пара коэффициентов, исходящих из вершины-переменной, задает некоторый коэффициент. Например, пара $<s^2, s^1>$ задает функцию φ^2, пара $<s^3, s^4>$ – функцию φ^3 и т.д. На рис. 2 показано, что разложение коэффициента (функции) φ^1 по переменной x_3 дает два одинаковых коэффициента s^1, следовательно, переменная x_3 является несущественной для φ^1, т.е. $\varphi^1=s^1$, и переменная φ^1 также исключается из многоуровневого представления функций системы *F*. Пройдя вверх до корневых вершин и найдя одинаковые коэффициенты, получаем результирующую *BDD* (рис. 1), которой соответствует результирующее многоуровневое представление функций системы:

$$f^1=\overline{x_1}\psi^1\vee x_1\psi^2;\ \ f^2=\overline{x_1}\varphi^3\vee x_1\psi^4;$$
$$f^3=\overline{x_1}\psi^5\vee x_1\psi^6;$$
$$\psi^1=\overline{x_2}\varphi^1\vee x_2\varphi^2;\ \ \psi^2=x_2\varphi^3;\ \ \psi^4=\overline{x_2}s^1\vee x_2\varphi^4;$$
$$\psi^5=x_2\varphi^3;\ \ \psi^6=\overline{x_2}\varphi^5\vee x_2\varphi^6;$$
$$\varphi^2=\overline{x_3}s^2\vee x_3s^1;\ \ \varphi^3=\overline{x_3}s^3\vee x_3s^4;\ \ \varphi^4=x_3s^5;$$
$$\varphi^5=\overline{x_3}s^6\vee x_3s^2;\ \ \varphi^6=\overline{x_3}s^2;$$
$$s^1=x_4\lambda^1;\ \ s^2=\overline{x_4}\lambda^3\vee x_4\lambda^2;\ \ s^4=\overline{x_4}\lambda^4;$$
$$\lambda^1=\overline{x_5}\omega^1;\ \ \lambda^2=\overline{x_5}\omega^1\vee x_5;\ \ \lambda^3=x_5;\ \ \lambda^4=x_5\omega^2;$$
$$\omega^1=x_6;\ \ \omega^2=\overline{x}_6.$$

Алгоритм этапа 1 выбора перестановки аргументов

Данный алгоритм состоит в последовательном применении следующих трех эвристик. Каждая эвристика применяется итерационно до тех пор, пока очередная итерация не даст уменьшения сложности *BDD*. Для каждой из рассматриваемых промежуточных перестановок подсчитывается сложность *BDD*.

Эвристика 1 – «перестановка одного аргумента x_i с правым соседом». Исходной являет-

ся начальная перестановка $<x_1,...,x_n>$. Аргументы $x_1,...,x_n$ рассматриваются поочередно. Сначала аргумент x_1 меняется позицией с правым аргументом x_2 (осуществляется транспозиция элементов перестановки), затем x_1 меняется позицией с правым соседом x_3 и так далее, пока x_1 не окажется на последнем месте. Среди пройденных выбирается лучшая перестановка (лучшей перестановке соответствует *BDD* меньшей сложности), она является начальной для последующего движения аргумента x_2, при этом x_2 помещается на первую позицию и двигается путем перестановки с правым соседом на последнюю позицию, опять выбирается лучшая перестановка, являющаяся начальной для движения аргумента x_3, и так далее, пока не выполнит аналогичное движение аргумент x_n. Поочередное движение всех аргументов составляет одну итерацию применения эвристики 1. Если итерация применения эвристики 1 уменьшает сложность *BDD*, то выполняется следующая итерация. Если же итерация не приводит к уменьшению сложности *BDD*, то осуществляется переход к применению эвристики 2.

Эвристика 2 – «попарная перестановка». Берется лучшая перестановка $X_{best.}$, полученная в результате применения эвристики 1. Очередная рассматриваемая перестановка отличается от предыдущей переменой мест только двух аргументов, пока не будет найдена новая лучшая перестановка X_{best1}. Если она найдена, то процесс попарной перестановки повторяется уже для новой лучшей перестановки X_{best1}. Эвристика 2 прекращает работу, если после рассмотрения всех попарных перестановок нет уменьшения сложности *BDD*.

Эвристика 3 – «оконная перестановка». Исходной является лучшая перестановка X_{best2}, полученная в результате применения эвристики 2. «Окно» представляет собой четыре последовательно расположенных аргумента $\langle x_i, x_{i+1}, x_{i+2}, x_{i+3}\rangle$, внутри которого проводятся все перестановки четырех аргументов. Затем окно перемещается на один аргумент вправо – получается новое окно $\langle x_{i+1}, x_{i+2}, x_{i+3}, x_{i+4}\rangle$, внутри которого опять осуществляются все перестановки. Сдвиг окна до последней позиции и перебор внутри него всех перестановок свидетельствует об окончании одной итерации применения эвристики 3. Лучшая перестановка – начало для следующей итерации применения эвристики 3.

Если итерация не приводит к уменьшению сложности *BDD*, то полученная в результате применения эвристики 3 перестановка результирующая для алгоритма этапа 1.

Экспериментальные исследования

Алгоритм построения *BDD* программно реализован: программа *OPT_BDD*, реализующая алгоритм, подвергнута обширному экспериментальному исследованию.

Эксперимент 1. Предлагаемый на этапе 1 алгоритм поиска лучшей перестановки сравнивался с алгоритмом случайного перебора перестановок на потоке примеров систем *F* ДНФ – схемы программируемых логических матриц (ПЛМ) из библиотеки *Berkeley PLA Test Set* и четыре схемы (*lal*, *ttt*2, *too_large*, *x*1) многоуровневой логики, преобразованные в ПЛМ. Результаты эксперимента 1 приведены в табл. 4, где (и далее в других таблицах) используются следующие обозначения: *n* – число переменных; *m* – число функций; *k* – число различных элементарных конъюнкций, входящих в ДНФ всех функций системы *F;* S_{BDD} – сложность *BDD*; α – число перебранных перестановок, знак + свидетельствует о полном переборе всех перестановок; *t* – время в секундах построения *BDD* (процессор *Pentium* 4 с тактовой частотой 2,8 ГГц). Лучшие решения в табл. 4 (и других таблицах) выделены жирным шрифтом.

Эксперимент 2. Сравнение предложенного алгоритма с алгоритмом из [6]. Результаты эксперимента 2 приведены в табл. 5. Экспериментальные данные (сложность *BDD* и время), характеризующие программу из [6] и приведенные в табл. 5, взяты из статьи [6].

Таблица 4

				Случайный перебор перестановок			*OPT_BDD*		
Имя схемы	*n*	*m*	*k*	S_{BDD}	α	*t*, с	S_{BDD}	α	*t*, с
*add*6	12	7	1092	59	5000	276,12	**45**	907	52,09
*b*2	16	17	110	620	50000	24746,9	**538**	1227	177,07
*b*9	16	5	123	90	10000	1720,2	**69**	719	37,77
*br*1	12	8	34	83	15000	156,13	**76**	436	4,13
*br*2	12	8	35	73	5000	39,02	**71**	9763	44,33
*dc*2	8	7	58	**59**	40320+	335,78	**59**	65	0,52
dist	8	5	255	**144**	40320+	340,57	**144**	23	0,17
*in*0	15	11	135	378	5000	466,12	**301**	467	9,85
*in*2	19	10	137	299	5000	2761,23	**231**	4749	663,64
intb	15	7	664	809	1000	1269,74	**629**	757	240,29
*t*3	12	8	148	61	10000	142,21	**54**	187	1,95
tial	14	8	640	728	35000	36115,6	**682**	3912	1460,55
xparc	41	73	551	2762*	1000	8670,31	**1927**	2676	600,42
*vtx*1	27	6	110	230*	1000	1203,73	**196**	1256	118,11
*x*6*dn*	39	5	121	264*	1000	1468,71	**238**	1431	337,99
*x*9*dn*	27	7	120	260*	1000	1319,32	**212**	642	83,55
signet	39	8	124	2938*	–	–	**1500**	17945	3197,2
shift	19	16	100	57*	10000	1122,01	**50**	1067	12,84
so*ar*	83	94	529	956*	1000	50737,84	**560**	9318	1700,2

* Улучшить начальную перестановку с помощью случайной перестановки аргументов не удалось.

Таблица 5

				Программа [6]		Программа *OPT_BDD*	
Имя схемы	*n*	*m*	*k*	S_{BDD}	*t*, с	S_{BDD}	*t*, с
*alu*4	14	8	1028	747	21,65	**735**	20,06
*apex*2	39	3	1035	739	38,53	**333**	843,80
*apex*3	54	50	280	1129	87,23	**997**	83,37
*e*64	65	65	65	865	183,80	**128**	399,19
*misex*3	14	14	1848	**523**	22,88	581	142,62
*table*3	14	14	175	798	40,21	**747**	11,01
*table*5	17	15	158	**689**	28,15	708	8,98

Целью экспериментов 3–5 было сравнение эффективности применения оптимизации *BDD*, выступающей в качестве предварительного этапа синтеза комбинационных схем, реализуемых в составе БМК.

В каждом из этих экспериментов синтезировалась схема в базисе БМК. В качестве системы синтеза использовалась система *Leonardo* [7]. Целевой библиотекой синтеза была выбрана библиотека БМК [7, с. 159], состоящая из 35 элементов. Сложность схемы $S_{БМК}$ в библиотеке проектирования БМК (далее просто схемы БМК) подсчитывается как сумма площадей, входящих в данную схему элементов. В экспериментах 3–8 исходные примеры ПЛМ и оптимизированные представления *BDD* задавались на языке *VHDL*.

Эксперимент 3. Синтез схем БМК по исходным *VHDL*-описаниям схем ПЛМ.

Эксперимент 4. Синтез схем БМК по оптимизированным (с помощью случайного перебора перестановок) представлениям *BDD*.

Эксперимент 5. Синтез схем БМК по оптимизированным с помощью программы *OPT_BDD* представлениям *BDD*.

Результаты экспериментов 3–5 приведены в табл. 6.

Таблица 6

Имя схемы	*n*	*m*	*k*	Эксперимент **3.** Синтез схем БМК по **исходным** ПЛМ	Эксперимент **4.** Синтез схем БМК по *BDD*, случайный перебор перестановок	Эксперимент **5.** Синтез схем БМК по *BDD*, *OPT_BDD*
				$S_{БПЛ}$	$S_{БПЛ}$	$S_{БПЛ}$
*add*6	12	7	1092	4935	167	**137**
*addm*4	9	8	480	3985	1379	**846**
*b*12	15	9	431	192	194	**160**
*b*2	16	17	110	2474	2825	**1682**
*in*0	15	11	135	2067	1411	**1036**
*in*2	19	10	137	1845	1120	**711**
*m*181	15	9	430	192	219	**177**
*m*3	8	16	128	1385	975	**562**
*mlp*4	8	8	225	2564	1069	**674**
tial	14	8	640	4505	4030	**2389**
*z*9*sym*	9	1	420	920	212	**161**
intb	15	7	664	5810	5810	**2269**
alu1	12	8	19	**66**	**66**	**66**
*in*1	16	17	110	2862	2468	**1711**
*vtx*1	27	6	110	**175**	314	349
*x*9*dn*	27	7	120	**198**	330	370
soar	83	94	529	**1705**	2228	2043
gary	15	11	442	1119	1287	**1046**
*max*1024	10	6	1024	4048	**1255**	1307
lal	26	19	117	**280**	448	298
*ttt*2	24	21	222	466	663	**432**
too_large	38	3	1027	5868	–	**4415**
*x*1	51	35	274	**801**	–	1245

Цель экспериментов 6–8 – сравнение эффективности применения оптимизации *BDD*, выступающей в качестве предварительного этапа синтеза комбинационных схем, реализуемых в составе *FPGA*.

В качестве целевой микросхемы *FPGA* выбрана микросхема *XC*2*s*100 семейства *SPARTAN*2. Сложность (площадь) полученных логических схем *FPGA* подсчитывалась в числе

программируемых элементов *LUT* (*Look Up Table*), имеющих 4 входа.

Эксперимент 6. Синтез схем *FPGA* по исходным *VHDL*-описаниям схем ПЛМ.

Эксперимент 7. Синтез схем *FPGA* по оптимизированным (с помощью случайного перебора перестановок) представлениям *BDD*.

Эксперимент 8. Синтез схем *FPGA* по оптимизированным с помощью программы *OPT_BDD* представлениям *BDD*.

Таблица 7

Имя схемы	*n*	*m*	*k*	Эксперимент **6**. Синтез схем БМК по **исходным** ПЛМ	Эксперимент **7**. Синтез схем БМК по *BDD*, случайный перебор перестановок	Эксперимент **8**. Синтез схем БМК по *BDD*, *OPT_BDD*
				LUT	*LUT*	*LUT*
*add*6	12	7	1092	66	19	**14**
*addm*4	9	8	480	**95**	114	151
*b*12	15	9	431	25	24	**23**
*b*2	16	17	110	476	381	**318**
*in*0	15	11	135	**141**	**141**	195
*in*2	19	10	137	186	191	**125**
*m*181	15	9	430	**25**	**25**	26
*m*3	8	16	128	**69**	**69**	111
mlp4	8	8	225	**68**	76	123
tial	14	8	640	652	**452**	478
z9sym	9	1	420	76	**18**	19
intb	15	7	664	423	423	**391**
*alu*1	12	8	19	**8**	**8**	**8**
*in*1	16	17	110	392	468	**297**
*vtx*1	27	6	110	**28**	53	99
x9dn	27	7	120	**31**	53	90
soar	83	94	529	**312**	384	347
gary	15	11	442	197	263	**191**
*max*1024	10	6	1024	676	255	**254**
*la*l	26	19	117	**53**	81	55
*ttt*2	24	21	222	78	103	**72**
too_large	38	3	1027	1152	–	**847**
*x*1	51	35	274	**146**	–	213

Обсуждение результатов экспериментов Экспериментальные исследования показали, что предложенный алгоритм эффективен, конкурентоспособен с известным алгоритмом и пригоден для практического использования. Применение оптимизации многоуровневых представлений систем полностью определенных булевых функций с помощью *BDD* полезно (приводит к сокращению сложностей схем) при синтезе в базисе БМК. Однако применение *BDD* в качестве предварительного этапа оптимизации схем *FPGA* оказалось полезным только в половине примеров схем. Случайный поиск перестановок аргументов оказывается в общем случае неэффективным (в сравнении с предложенным алгоритмом поиска порядка переменных разложения) как для оптимизации схем БМК, так и для оптимизации схем *FPGA*.

Заключение. В статье описан алгоритм оптимизации многоуровневых представлений систем полностью определенных булевых функций и результаты экспериментального исследования. Эксперименты показали достаточно высокую эффективность предложенного алгоритма. Программа *OPT_BDD*, реализующая алгоритм построения *BDD*, включена в систему *Micel* [8] схемной реализации параллельных алгоритмов логического управления.

1. *Кузнецов О.П.* О программной реализации логических функций и автоматов //Автоматика и телемеханика. – 1977. – № 7. – С. 63 – 74; № 9. – С. 138 – 149.
2. *Akers S.B.* Binary Decision Diagrams // IEEE Trans. on Computers. – 1978. – **C–27**, № 6. – P. 509–516.
3. *Bryant R.E.* Graph-based algorithms for boolean functions manipulation. // Ibid. – 1986. – **C–35**, № 8. – P. 677–691.
4. *Bryant R.E.*, *Meinel C.* Ordered Binary Decision Diagrams // Logic synthesis and verification (Ed. by S. Hassoun, T. Sasao, R.K. Brayton). Kluwer Acad. Publ., 2002. – P. 285–307.
5. *Meinel C.*, *Theobald T.* Algorithms and Data Structures in VLSI Design: OBDD – Found. and Appl. – Berlin, Heidelberg: Springer-Verlag, 1998. – 267 p.
6. *Dynamic* variable reordering for BDD minimization / E. Felt, G. York, R. Brayton et al. // Proc. EURO–DAC, 1993, 20–24 Sep. – P. 130–135.
7. *Бибило П.Н.* Синтез логических схем с использованием языка *VHDL* – М.: Солон–Р, 2002. – 384 с.
8. *Программный* комплекс *Micel* высокоуровневого и логического синтеза параллельных алгоритмов логического управления / П.Н. Бибило, С.Н. Кардаш, Н.А. Кириенко и др. // УСиМ. – 2009. – № 5. – С. 81–88.

Поступила 30.04.2009

Тел. для справок: + 375 (17) 284-2084 (Минск)
E-mail: *bibilo@newman.bas-net.by*