УДК 519.714
DOI: 10.17587/prin.13.363-382

П. Н. Бибило, д-р техн. наук, проф., зав. лаб., bibilo@newman.bas-net.by,
Объединенный институт проблем информатики Национальной академии наук Беларуси, Минск

Аппаратная реализация преобразователей кодов, предназначенных для сокращения длины двоичных кодируемых слов

Рассмотрено аппаратное решение задачи проектирования кодовых преобразователей, предназначенных для сокращения длины слов из заданного набора кодируемых двоичных слов. Кодирование предполагает, что различные двоичные слова (наборы бит) будут закодированы различными двоичными кодами меньшей длины (разрядности). Предлагаемые способы решения данной задачи основаны на составлении и логической минимизации таких форм систем не полностью определенных булевых функций, как дизъюнктивные нормальные формы и бинарные диаграммы решений, называемые BDD-представлениями (BDD — Binary Decision Diagram). Минимизация составляемых функциональных описаний ориентирована на уменьшение аппаратной сложности комбинационных схем в базисе библиотечных элементов заказных СБИС либо программируемых элементов FPGA, реализующих преобразователи кода рассматриваемого класса.

Ключевые слова: *преобразователь кода, система булевых функций, дизъюнктивная нормальная форма (ДНФ), Binary Decision Diagram (BDD), разложение Шеннона, синтез логической схемы, VHDL, СБИС*

Введение

В цифровых вычислительных и управляющих системах широкое распространение получили преобразователи кодов — устройства, предназначенные для преобразования двоичного кода из одной формы в другую. Для представления информации используют разнообразные двоичные и двоично-десятичные коды чисел: прямой; обратный; дополнительный и их модификации [1]; унитарные коды; двоичные коды чисел по модулю [2] и т. д. Проектирование таких преобразователей кодов опирается на известные формы задания входной и выходной информации, т. е. кодируемых двоичных слов и кодов (кодирующих слов). Однако при проектировании цифровых систем могут возникать сложности передачи по шинам данных длинных двоичных слов, т. е. таких, разрядность которых превышает разрядность шины данных. Например, по 16-разрядной шине данных требуется передавать 18-разрядные или 17-разрядные слова. Конечно, каждое такое слово можно передать за два такта функционирования системы, однако такой подход снижает общее быстродействие всей системы. Одним из подходов для разрешения таких сложностей является разработка комбинационных схем, которые осуществляют преобразование длинных двоичных кодируемых слов в более короткие. При получении информации, переданной по шине данных, может потребоваться обратный преобразователь, осуществляющий преобразование переданного слова в исходную форму.

В настоящей работе рассмотрено решение задачи логического проектирования кодовых преобразователей, предназначенных для сокращения длины каждого слова из заданного набора кодируемых двоичных слов. Кодирование предполагает, что различные двоичные слова (наборы бит) одинаковой длины будут закодированы различными двоичными кодами меньшей длины (разрядности). Естественно, коды также должны иметь одинаковую длину. Предлагаемые способы решения задачи основаны на составлении и логической минимизации различных форм функционального описания проектируемых преобразователей кода, в качестве таких форм выступают дизъюнктивные нормальные формы (ДНФ) [3] и бинарные диаграммы решений, называемые BDD-представлениями [4]. Минимизация функциональных описаний ориентирована на уменьшение аппаратной сложности

преобразователей кода и существенным образом использует такие модели функционирования проектируемых цифровых устройств, как не полностью определенные двоичные (булевы) функции и многозначные функции, зависящие от булевых переменных. Уменьшение сложности однотактных комбинационных схем, реализующих функции преобразователей кодов, и является основной целью логического проектирования преобразователей кодов рассматриваемого класса.

1. Определения и постановка задачи

Булевыми называются двоичные (0, 1) функции $f(\boldsymbol{x}) = f(x_1, x_2, ..., x_n)$ двоичных (булевых) переменных $x_1, x_2, ..., x_n$. Пусть V^x — булево пространство, построенное над переменными булева вектора $\boldsymbol{x} = (x_1, ..., x_n)$. Элементами этого пространства являются n-компонентные наборы (векторы) $\boldsymbol{x}^*$ нулей и единиц. Булева функция, значения 0, 1 которой определены на всех элементах $\boldsymbol{x}^* \in V^x$, называется *полностью определенной*. Если на некоторых элементах булева пространства V^x значения функции не определены, то такая функция называется не полностью определенной, или *частичной*. Частичная булева функция принимает единичное значение на элементах $\boldsymbol{x}^*$ подмножества M_f^1 булева пространства V^x и нулевое значение на элементах подмножества M_f^0. На всех остальных элементах пространства V^x, образующих подмножество M_f^- пространства V^x, значения частичной функции не определены и имеют значение "–". Частичная функция f_1 *реализуется* частичной (либо полностью определенной) функцией f_2 (обозначается $f_1 \prec f_2$, $f_2 \succ f_1$), если $M_{f_1}^1 \subseteq M_{f_2}^1$, $M_{f_1}^0 \subseteq M_{f_2}^0$. Если функция f_2 *реализует* f_1 $(f_1 \prec f_2)$, то функцию f_2 называют *доопределением* функции f_1. Для пары f_1, f_2 полностью определенных булевых функций отношение реализации является отношением *равенства*. Под *векторной* булевой функцией $\boldsymbol{f}(\boldsymbol{x})$ будем понимать упорядоченную систему частичных булевых функций $\boldsymbol{f}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), ..., f_m(\boldsymbol{x}))$, значениями векторных функций на элементах $\boldsymbol{x}^*$ булева пространства являются m-компонентные троичные векторы $\boldsymbol{f}(\boldsymbol{x}^*)$.

Задача 1. Пусть задана булева (двоичная) матрица B, имеющая k различных строк b^i и n столбцов, при этом $k \leqslant 2^{n-1}$. Требуется закодировать строки b^i матрицы B различными минимальными по длине m булевыми векторами (строками) c^i, чтобы комбинационная логическая схема, осуществляющая преобразование строк b^i в строки c^i, имела возможно меньшую сложность.

Понятие сложности схем зависит от используемого технологического базиса и будет уточнено далее.

Пусть кодирующие комбинации c^i, соответствующие строкам матрицы B, образуют булеву матрицу C. Другими словами, для булевой матрицы B требуется получить булеву матрицу C, содержащую k различных строк и m столбцов, где $m = \lceil \log_2 k \rceil$, через $\lceil a \rceil$ обозначается ближайшее целое число, большее либо равное a.

Назовем задачу 1 задачей синтеза схемы кодового преобразователя (либо преобразователя кодов).

Обозначим столбцы булевой матрицы B через $x_1, x_2, ..., x_n$, столбцы матрицы C — через $c_1, c_2, ..., c_m$. Пара матриц (B, C) задает *частичную* векторную булеву функцию $\boldsymbol{c}(\boldsymbol{x}) = (c_1(\boldsymbol{x}), ..., c_m(\boldsymbol{x}))$, $\boldsymbol{x} = (x_1, x_2, ..., x_n)$, значениями данной векторной функции на двоичных наборах из матрицы B являются m-компонентные *двоичные* наборы (строки) из матрицы C. На наборах, принадлежащих множеству $V^x \backslash B$, значения векторной функции $\boldsymbol{c}(\boldsymbol{x})$ не определены — соответствующие *троичные векторы* значений компонентных функций состоят только из неопределенных значений "–".

Перейдем к обсуждению постановки задачи 1.

Во-первых, для решения задачи синтеза любой логической схемы (не только схемы кодового преобразователя) требуется указать базис синтеза (набор логических элементов), часто называемый также технологической библиотекой синтеза [5].

Во-вторых, логические схемы могут быть однотактными (срабатывать в течение одного такта функционирования дискретной системы) либо иметь конвейерную организацию и осуществлять требуемое преобразование набора значений входных сигналов в значения выходных сигналов за несколько тактов смены значений синхросигнала.

В-третьих, синтез схем осуществляется традиционно в два этапа: на первом этапе выполняется технологически независимая оптимизация, на втором — технологическое отображение (*technology mapping*), когда оптимизированные логические описания булевых функций покрываются функциональными описаниями библиотечных элементов [4, 5].

В-четвертых, будем считать, что на вход синтезированной логической схемы при ее функционировании могут поступать только входные наборы из матрицы B (т. е. наборы из множества $V^x \backslash B$ никогда не должны поступать на вход схемы), что позволит доопределять исходную частичную векторную функцию до полностью определенной произвольным образом.

Примем следующие *соглашения* (ограничения). Пусть библиотекой является библиотека логических элементов заказных КМОП СБИС, описанная в работе [6], системой синтеза логических схем — синтезатор LeonardoSpectrum [5], а программами технологически независимой оптимизации —

программы системы FLC2 логической оптимизации функциональных описаний систем булевых функций [7]. Синтез будет выполняться в целях получения однотактных (не конвейерных) реализаций схем. Синтезатор LeonardoSpectrum после синтеза схемы подсчитывает сложность (площадь) схем из библиотечных элементов как сумму площадей всех логических элементов схемы и выдает значение данного параметра под названием *Area* (площадь). Заметим, что задержка синтезированной схемы также вычисляется в виде значения параметра *Delay*.

Решение задачи 1 сводится к последовательному решению описанных далее взаимосвязанных задач 2—4. Критерием оптимизации при решении задач 2 и 3 является сложность функционального описания системы ДНФ либо сложность BDD-представления, задающих функции кодового преобразователя. Критерием оптимизации при решении задачи 4 является сложность комбинационной схемы кодового преобразователя. Для промышленных синтезаторов логических схем уменьшение сложности (либо задержки) результирующей логической схемы может достигаться установкой соответствующих управляющих опций синтезатора, в нашем случае LeonardoSpectrum.

Задача 2. Найти кодирующую матрицу C и, возможно, доопределение частичной векторной функции $\boldsymbol{c}(\boldsymbol{x}) = (c_1(\boldsymbol{x}), ..., c_m(\boldsymbol{x}))$ на наборах из множества $V^x \backslash B$, получить неоптимизированное представление функций $c_1(\boldsymbol{x}), ..., c_m(\boldsymbol{x})$ кодового преобразователя в виде ДНФ либо СДНФ (совершенной ДНФ).

Доопределение предполагает замену троичных векторов — значений функции $\boldsymbol{c}(\boldsymbol{x})$, состоящих только из неопределенных значений "–" полностью определенными (0, 1) двоичными векторами. В этом случае векторная функция $\boldsymbol{c}(\boldsymbol{x}) = (c_1(\boldsymbol{x}), ..., c_m(\boldsymbol{x}))$ является полностью определенной.

Задача 3. Выполнить логическую минимизацию частичной либо полностью определенной векторной функции $\boldsymbol{c}(\boldsymbol{x}) = (c_1(\boldsymbol{x}), ..., c_m(\boldsymbol{x}))$ и получить ее оптимизированное (минимизированное) алгебраическое представление.

Булевы векторы $c(b^i)$ значений функции $\boldsymbol{c}(\boldsymbol{x}) = (c_1(\boldsymbol{x}), ..., c_m(\boldsymbol{x}))$ на наборах b^i являются *кодами строк b^i матрицы B*. Для обратного преобразования в роли исходной матрицы B выступает матрица C. Пара матриц (C, B) задает систему булевых функций, которую надо реализовать комбинационной схемой обратного преобразования булевых строк матрицы C в булевы строки матрицы B. Частичные векторные булевы функции $\boldsymbol{c}(\boldsymbol{x}) = (c_1(\boldsymbol{x}), ..., c_m(\boldsymbol{x}))$ являются промежуточными математическими моделями при нахождении кодов строк матрицы B.

Решение задачи 3 будем выполнять описанными далее двумя программами системы FLC-2 [7].

Первая из них — программа Minim раздельной минимизации системы частичных либо полностью определенных функций в классе ДНФ, т. е. каждая из компонентных функций $c_1(\boldsymbol{x}), ..., c_m(\boldsymbol{x})$ будет минимизироваться независимо от других. Результат работы программы Minim — минимизированная система ДНФ, задающая полностью определенные функции кодового преобразователя.

Вторая программа — программа BDD_Builder совместной минимизации многоуровневых BDDI-представлений систем полностью определенных булевых функций [8]. Результат работы программы BDD_Builder — взаимосвязанные логические уравнения (формулы разложений Шеннона), задающие полностью определенные функции кодового преобразователя.

Результатом решения задачи 3 является минимизированное представление полностью определенной векторной функция $\boldsymbol{c}(\boldsymbol{x})$, являющееся исходными данными для решения задачи 4.

Задача 4. Синтезировать логическую схему в заданном базисе логических элементов по минимизированному представлению векторной функции $\boldsymbol{c}(\boldsymbol{x})$.

2. Способы решения задач 2, 3

Способ 1 (тривиальный). Закодируем строки матрицы B двоичными наборами, задающими их номера (нумерация начинается с нуля), выполним тривиальное (без комбинаторного поиска) доопределение системы частичных функций —заменим неопределенные значения системы функций $\boldsymbol{c}(\boldsymbol{x})$ на наборах из множества $V^x \backslash B$, нулевыми строками. Заметим, что номера строк (коды) можно задавать не по порядку, а ***случайным*** образом размещая их в кодирующей матрице C, однако это не меняет суть способа 1.

Пример 1. Пусть булева матрица B имеет 16 строк и 5 столбцов:

$$B = \begin{array}{c} x_1x_2x_3x_4x_5 \\ \begin{matrix} 0\,0\,0\,0\,0 \\ 0\,0\,1\,0\,1 \\ 0\,1\,0\,1\,0 \\ 0\,1\,1\,0\,0 \\ 1\,0\,0\,0\,1 \\ 1\,0\,1\,1\,0 \\ 1\,1\,0\,0\,0 \\ 0\,0\,0\,0\,1 \\ 0\,0\,1\,1\,0 \\ 0\,1\,0\,0\,0 \\ 0\,1\,1\,0\,1 \\ 1\,0\,0\,1\,0 \\ 1\,0\,1\,0\,0 \\ 1\,1\,0\,0\,1 \\ 0\,0\,0\,1\,0 \\ 0\,0\,1\,0\,0 \end{matrix} \end{array}.$$

Таблица 1

Результаты логической оптимизации в способе 1 (пример 1)

Доопределение "нулевое", коды — номера строк (оптимизация не проведена)		Раздельно минимизированные ДНФ		BDDI-представление
$x_1x_2x_3x_4x_5$	$c_1c_2c_3c_4$	$x_1x_2x_3x_4x_5$	$c_1c_2c_3c_4$	
0 0 0 0 0	0 0 0 0	0 0 − 0 1	0 0 0 1	c1=^x4*s0+x4*s1;
0 0 1 0 1	0 0 0 1	0 1 − 0 0	0 0 0 1	c2=^x4*s2+x4*s3;
0 1 0 1 0	0 0 1 0	1 0 − 1 0	0 0 0 1	c3=^x4*s4+x4*s5;
0 1 1 0 0	0 0 1 1	0 0 1 0 −	0 0 0 1	c4=^x4*s6+x4*s7;
1 0 0 0 1	0 1 0 0	1 1 0 0 0	0 0 1 0	s2=^x3*^s9+x3*s11;
1 0 1 1 0	0 1 0 1	0 0 0 0 1	0 0 1 0	s1=^x3*s11+x3*s13;
1 1 0 0 0	0 1 1 0	0 1 1 0 −	0 0 1 0	s3=^x3*s13+x3*s7;
0 0 0 0 1	0 1 1 1	0 − 0 1 0	0 0 1 0	s5=^x3*s14;
0 0 1 1 0	1 0 0 0	0 0 0 1 0	0 1 0 0	s0=^x3*s15+x3*s16;
0 1 0 0 0	1 0 0 1	− 0 0 0 1	0 1 0 0	s6=^x3*s19+x3*s20;
0 1 1 0 1	1 0 1 0	1 0 1 − 0	0 1 0 0	s4=^x3*s8+x3*s9;
1 0 0 1 0	1 0 1 1	1 1 0 0 −	0 1 0 0	s20=^x2*^x1+x2*s22;
1 0 1 0 0	1 1 1 0	0 1 0 0 0	1 0 0 0	s11=^x2*^x5;
1 1 0 0 1	1 1 0 1	0 1 1 0 1	1 0 0 0	s9=^x2*^x5+x2*^x1;
0 0 0 1 0	1 1 1 0	1 1 0 0 1	1 0 0 1	s14=^x2*^x5+x2*s22;
0 0 1 0 0	1 1 1 1	0 0 − 1 0	1 0 0 0	s16=^x2*^x5+x2*s26;
0 1 0 0 1	**0 0 0 0**	− 0 1 0 0	1 1 1 0	s13=^x2*s22;
0 1 1 1 0	**0 0 0 0**	− 0 0 1 0	1 0 1 0	s19=^x2*s26+x2*s30;
1 0 0 0 0	**0 0 0 0**			s8=^x2*s26+x2*s31;
1 0 1 0 1	**0 0 0 0**			s7=^x2*s31;
1 1 0 1 0	**0 0 0 0**			s15=x2*s30;
0 0 0 1 1	**0 0 0 0**			s22=^x1*^x5;
0 0 1 1 1	**0 0 0 0**			s30=^x1*^x5+x1*x5;
0 1 0 1 1	**0 0 0 0**			s26=^x1*x5;
0 1 1 1 1	**0 0 0 0**			s31=x1*^x5;
1 0 0 1 1	**0 0 0 0**			
1 0 1 1 1	**0 0 0 0**			
1 1 0 1 1	**0 0 0 0**			
1 1 1 0 0	**0 0 0 0**			
1 1 1 0 1	**0 0 0 0**			
1 1 1 1 0	**0 0 0 0**			
1 1 1 1 1	**0 0 0 0**			
Схема 1		**Схема 2**		**Схема 3**

В данном примере $m = \lceil \log_2 16 \rceil = 4$, доопределенные значения на наборах из множества $V^x \backslash B$ выделены в табл. 1 полужирным шрифтом. В табл. 1 в левой части приводится таблица истинности системы полностью определенных булевых функций, данной таблице соответствует система совершенных ДНФ булевых функций, каждой строке матрицы B соответствуют своя полная элементарная конъюнкция. Например, второй строке (0 0 1 0 1) соответствует полная элементарная конъюнкция $\bar{x}_1\bar{x}_2x_3\bar{x}_4x_5$, данная конъюнкция входит в СДНФ только одной компонентной функции c^4. В средней части табл. 1 находится функциональное описание, полученное в результате логической минимизации системы функций из левой части программой Minim. Например, матричная форма минимизированной ДНФ функции c_4 в алгебраической форме записывается в виде $c_4 = \bar{x}_1\bar{x}_2\bar{x}_4x_5 \vee \bar{x}_1x_2\bar{x}_4\bar{x}_5 \vee x_1\bar{x}_2x_4\bar{x}_5 \vee \bar{x}_1\bar{x}_2x_3\bar{x}_4$.

В правой части табл. 1 заданы формулы многоуровневого BDDI-описания, полученные программой BDD_Builder. Для упрощения текстовой записи логических уравнений использованы следующие обозначения: символ " + " обозначает дизъюнкцию, "*" - конъюнкцию, "^" - инверсию (отрицание) булевой переменной, которая идет после этого символа. Например, уравнение `c1 = = ^x4*s0 + x4*s1`; (табл. 1) в общепринятой записи имеет вид $c_1 = \bar{x}_4s_0 \vee x_4s_1$.

В нижней строке табл. 1 приводятся номера схем, которые будут синтезированы по соответствующим функциональным описаниям.

Если не проводить нулевое доопределение, приводящее к СДНФ, а рассматривать систему частичных функций, то можно улучшить решение путем раз-

дельной минимизации в классе ДНФ системы частичных функций. Однако на практике такая возможность использования модели частичных булевых функций при логической минимизации обычно игнорируется.

Способ 2. Замена двоичных строк матрицы *B* троичными строками, кодирование строк номерами. В способе 2 каждая двоичная строка матрицы *B* (полная элементарная конъюнкция) заменяется троичной строкой — элементарной конъюнкцией. Получение такой строки сводится к раздельной минимизации специально составленной (по матрице *B*) системы частичных функций в классе ДНФ. Для рассматриваемого примера исходная и результирующая системы функций заданы в табл. 2. По сути, минимизация сводится к расширению единственного набора из области единичных значений функции k_i до интервала за счет наборов из области неопределенных значений этой функции. Например, троичный вектор $k_2 = (-\ 0\ 1\ -\ 1)$ (минимизированная полностью определенная булева функция $k_2 = \bar{x}_2 x_3 x_5$) получается при доопределении неопределенных значений частичной функции k_2 единичными значениями на наборах (0 0 1 1 1), (1 0 1 0 1), (1 0 1 1 1).

Чтобы получить доопределение исходной частичной функции, получаемое в способе 2, доста-

Таблица 2

Расширение наборов до интервалов

Система частичных булевых функций		Минимизированная система полностью определенных булевых функций	
Наборы $x_1x_2x_3x_4x_5$	$k_1\ k_2\ k_3\ k_4 \ldots k_{14}\ k_{15}\ k_{16}$	Интервалы $x_1x_2x_3x_4x_5$	$k_1\ k_2\ k_3\ k_4 \ldots k_{14}\ k_{15}\ k_{16}$
0 0 0 0 0	1000000000000000	0 0 0 0 0	1000000000000000
0 0 1 0 1	0100000000000000	– 0 1 – 1	0100000000000000
0 1 0 1 0	0010000000000000	– 1 – 1 –	0010000000000000
0 1 1 0 0	0001000000000000	– 1 1 – 0	0001000000000000
1 0 0 0 1	0000100000000000	1 0 – – 1	0000100000000000
1 0 1 1 0	0000010000000000	1 – 1 1 –	0000010000000000
1 1 0 0 0	0000001000000000	1 1 – – 0	0000001000000000
0 0 0 0 1	0000000100000000	0 – 0 – 1	0000000100000000
0 0 1 1 0	0000000010000000	0 – 1 1 –	0000000010000000
0 1 0 0 0	0000000001000000	0 1 0 0 –	0000000001000000
0 1 1 0 1	0000000000100000	0 1 – – 1	0000000000100000
1 0 0 1 0	0000000000010000	1 – 0 1 –	0000000000010000
1 0 1 0 0	0000000000001000	1 – 1 0 –	0000000000001000
1 1 0 0 1	0000000000000100	1 1 – – 1	0000000000000100
0 0 0 1 0	0000000000000010	0 0 0 1 –	0000000000000010
0 0 1 0 0	0000000000000001	0 0 1 0 0	0000000000000001
0 1 0 0 1	----------		
0 1 1 1 0	----------		
1 0 0 0 0	----------		
1 0 1 0 1	----------		
1 1 0 1 0	----------		
0 0 0 1 1	----------		
0 0 1 1 1	----------		
0 1 0 1 1	----------		
0 1 1 1 1	----------		
1 0 0 1 1	----------		
1 0 1 1 1	----------		
1 1 0 1 1	----------		
1 1 1 0 0	----------		
1 1 1 0 1	----------		
1 1 1 1 0	----------		
1 1 1 1 1	----------		

Таблица 3

Результаты логической оптимизации в способе 2 (пример 1)

Доопределение по интервалам, коды — номера строк (оптимизация не проведена)		Раздельно минимизированные ДНФ		BDDI-представление
$x_1x_2x_3x_4x_5$	$c_1c_2c_3c_4$	$x_1x_2x_3x_4x_5$	$c_1c_2c_3c_4$	
0 0 0 0 0	0 0 0 0	− 0 1 − 1	0 0 0 1	c1=^x4*s0+x4*s1;
− 0 1 − 1	0 0 0 1	− 1 1 − 0	0 0 0 1	c2=^x4*s2+x4*s3;
− 1 − 1 −	0 0 1 0	1 1 − − 1	0 0 0 1	c3=^x4*s4+x4*s5;
− 1 1 − 0	0 0 1 1	1 − − 1 −	0 0 0 1	c4=^x4*s6+x4*s7;
1 0 − − 1	0 1 0 0	0 0 1 0 −	0 0 0 1	s2=^x1*s11+x1*s12;
1 − 1 1 −	0 1 0 1	− 1 − 1 −	0 0 1 0	s4=^x1*s13+x1*s14;
1 1 − − 0	0 1 1 0	− − 0 1 −	0 0 1 0	s3=^x1*s15+x1*s12;
0 − 0 − 1	0 1 1 1	1 1 − − 0	0 0 1 0	s1=^x1*s17+x1*s18;
0 − 1 1 −	1 0 0 0	0 − 0 − 1	0 0 1 1	s6=^x1*s19+x1*s20;
0 1 0 0 −	1 0 0 1	0 − 1 0 0	0 0 1 0	s7=^x1*s8+x1;
0 1 − − 1	1 0 1 0	0 1 − − 1	0 0 1 0	s0=^x1*s9+x1*s10;
1 − 0 1 −	1 0 1 1	1 − 1 − −	0 1 0 0	s9=^x2*^s22+x2*s22;
1 − 1 0 −	1 1 0 0	− − 0 − 1	0 1 0 0	s5=^x2*^x3+x2;
1 1 − − 1	1 1 0 1	− 0 1 0 0	0 1 0 0	s18=^x2*^x3+x2*s22;
0 0 0 1 −	1 1 1 0	1 1 − − −	0 1 0 0	s15=^x2*^x3+x2*s37;
0 0 1 0 0	1 1 1 1	0 0 0 1 −	0 1 0 0	s13=^x2*s25+x2*s28;
		0 − 1 1 −	1 0 0 0	s11=^x2*s25+x2*s37;
		0 1 0 0 −	1 0 0 1	s12=^x2*s28+x2;
		1 − 0 1 −	1 0 0 0	s19=^x2*s28+x2*^s31;
		1 − 1 0 −	1 0 0 0	s20=^x2*s31+x2*s28;
		− 1 − − 1	1 0 0 0	s10=^x2*x3+x2*s28;
		0 0 − 1 −	1 0 0 0	s8=^x2*x5+x2*s25;
		0 0 1 − 0	1 0 0 0	s17=^x2+x2*s28;
				s14=x2*^x5;
				s37=^x3*x5;
				s28=^x3*x5+x3;
				s25=^x3*x5+x3*^x5;
				s22=^x3+x3*x5;
				s31=x3*x5;
Схема 4		**Схема 5**		**Схема 6**

точно построить таблицу истинности функций по системе ДНФ, приведенной в левой (либо средней) части табл. 3. Минимизация в классе ДНФ изменяет форму задания векторной функции, но не ее таблицу истинности.

Способ 3. Замена двоичных строк матрицы *B* троичными строками, кодирование строк матрицы *B* согласно эвристики 1. В способе 3 каждая двоичная строка матрицы *B* (полная элементарная конъюнкция) заменяется троичной строкой — элементарной конъюнкцией так же, как и в способе 2. Однако кодирование полученных троичных векторов выполняется с помощью ***эвристики 1****: троичные векторы, содержащие много определенных (0, 1) элементов, кодируются булевыми векторами, содержащими возможно меньшее число единичных элементов.*

Данная эвристика ориентирована на сокращение общего числа литералов в системе ДНФ, задающей векторную полностью определенную функцию. Составим множество строк (конъюнкций), содержащих пять определенных элементов (литерала):

$$M^5 = \{(0\ 0\ 0\ 0\ 0),\ (0\ 0\ 1\ 0\ 0)\}.$$

Аналогично:

$$M^4 = \{(0\ 1\ 0\ 0\ -),\ (0\ 0\ 0\ 1\ -)\},$$

$$M^3 = \{(-\ 0\ 1\ -\ 1),\ (-\ 1\ -\ 1\ -),\ (-\ 1\ 1\ -\ 0),\ (1\ 0\ -\ -\ 1),\ (1\ -\ 1\ 1\ -),\ (1\ 1\ -\ -\ 0),\ (0\ -\ 0\ -1),\ (0\ -\ 1\ 1\ -),\ (0\ 1\ -\ -\ 1),\ (1\ -\ 0\ 1\ -),\ (1\ -\ 1\ 0\ -),\ (1\ 1\ -\ -\ 1)\}.$$

$$M^2 = \{(-\ \ 1\ \ -\ \ 1-)\}.$$

В примере булева строка (0 0 0 0 0) кодируется комбинацией (0 0 0 0) (табл. 4), поэтому полная элементарная конъюнкция $\overline{x}_1\overline{x}_2\overline{x}_3\overline{x}_4\overline{x}_5$ будет отсутствовать в ДНФ каждой из четырех компонентных функций c_1, c_2, c_3, c_4. Строка (0 0 1 0 0) кодируется вектором (0 0 1 0) с одной единичной компонентой, поэтому полная элементарная конъюнкция,

Таблица 4

Результаты логической оптимизации в способе 3 (пример 1)

Доопределение по интервалам, кодирование по эвристике 1		Раздельно минимизированные ДНФ		BDDI-представление
$x_1x_2x_3x_4x_5$	$c_1c_2c_3c_4$	$x_1x_2x_3x_4x_5$	$c_1c_2c_3c_4$	
0 0 0 0 0	0 0 0 0	1 – 1 – –	0 0 0 1	c1=^x5*s0+x5*s1;
– 0 1 – 1	0 0 1 1	1 – – 1 –	0 0 0 1	c2=^x5*s2+x5*s3;
– 1 – 1 –	1 1 1 1	0 – – – 1	0 0 0 1	c3=^x5*s4+x5*s5;
– 1 1 – 0	0 1 1 0	– – 1 1 –	0 0 0 1	c4=^x5*s6+x5*s7;
1 0 – – 1	1 1 1 0	0 0 1 – –	0 0 1 0	s4=^x2*^s9+x2*s23;
1 – 1 1 –	1 1 0 1	– 1 1 – 0	0 1 1 0	s0=^x2*s10+x2*s8;
1 1 – – 0	1 1 0 0	1 0 – – 1	0 1 0 0	s3=^x2*s14+x2*x4;
0 – 0 – 1	1 0 1 1	1 1 – – 0	0 1 0 0	s6=^x2*s16+x2*s17;
0 – 1 1 –	0 1 1 1	– – – 1 –	0 1 0 0	s1=^x2*s9+x2;
0 1 0 0 –	1 0 0 0	– 1 – 1 –	1 0 1 1	s2=^x2*x4+x2*s19;
0 1 – – 1	1 0 0 1	1 – 1 1 –	1 0 0 0	s5=^x2+x2*s8;
1 – 0 1 –	0 1 0 1	1 – – – 1	1 0 1 0	s9=^x1*^x3+x1;
1 – 1 0 –	0 0 0 1	1 1 – – –	1 0 0 0	s16=^x1*s20+x1*s23;
1 1 – – 1	1 0 1 0	– – 0 – 1	1 0 1 0	s19=^x1*s23+x1;
0 0 0 1 –	0 1 0 0	– 1 – – 1	1 0 0 0	s8=^x1*s27+x1;
0 0 1 0 0	0 0 1 0	– 1 0 – –	1 0 0 0	s14=^x1*x4+x1;
				s17=^x1*x4+x1*s23;
				s7=^x1+x1*s23;
				s10=x1*s20;
				s23=^x3*x4+x3;
				s27=^x3+x3*x4;
				s20=x3*x4;
Схема 7		**Схема 8**		**Схема 9**

содержащая пять литералов, войдет в ДНФ только одной функции c_3. Строки из множества M^4 кодируются векторами (1 0 0 0), (0 1 0 0) соответственно. Затем кодируются строки из множества M^3, используя оставшийся вектор (0 0 0 1) с одной единичной компонентой и векторы с двумя и тремя единичными компонентами. Для единственного троичного вектора (– 1 – 1 –) из множества M^2 присваивается кодовая комбинация (1 1 1 1) с четырьмя единичными компонентами. Формульное задание системы ДНФ, полученной в способе 3, содержит 96 литералов и имеет вид

$$c_1 = x_2x_4 \vee x_1\overline{x}_2x_5 \vee x_1x_3x_4 \vee x_1x_2\overline{x}_5 \vee \vee\ \overline{x}_1\overline{x}_3x_5 \vee \overline{x}_1x_2\overline{x}_3\overline{x}_4 \vee \overline{x}_1x_2x_5 \vee x_1x_2x_5;$$

$$c_2 = x_2x_4 \vee x_2x_3\overline{x}_5 \vee x_1\overline{x}_2x_5 \vee x_1x_3x_4 \vee \vee\ x_1x_2\overline{x}_5 \vee \overline{x}_1x_3x_4 \vee x_1\overline{x}_3x_4 \vee \overline{x}_1\overline{x}_2\overline{x}_3x_4;$$

$$c_3 = \overline{x}_2x_3x_5 \vee x_2x_4 \vee x_2x_3\overline{x}_5 \vee x_1\overline{x}_2x_5 \vee \vee\ \overline{x}_1\overline{x}_3x_5 \vee \overline{x}_1x_3x_4 \vee x_1x_2x_5 \vee \overline{x}_1\overline{x}_2x_3\overline{x}_4\overline{x}_5;$$

$$c_4 = \overline{x}_2x_3x_5 \vee x_2x_4 \vee x_1x_3x_4 \vee \overline{x}_1\overline{x}_3x_5 \vee \vee\ \overline{x}_1x_3x_4 \vee \overline{x}_1x_2x_5 \vee x_1\overline{x}_3x_4 \vee x_1x_3\overline{x}_4.$$

После того как коды присвоены, можно выполнить минимизацию в классе ДНФ полученной системы полностью определенных булевых функций — решать задачу 3. В результате такой минимизации число литералов в системе ДНФ сокращается с 96 до 47. Матричная форма минимизированных ДНФ приведена в средней части табл. 4. При синтезе логических схем из библиотечных элементов уменьшение числа литералов в представлениях систем полностью определенных булевых функций приводит к менее сложным схемам [9]. Поэтому число литералов - это основной критерий минимизации алгебраических представлений функций на этапе технологически независимой оптимизации — первом этапе синтеза логических схем.

Способы 2 и 3 ориентируются на уменьшение сложности функционального описания кодового преобразователя в классе ДНФ на основе минимизации суммарного числа литералов в представлении функций в виде ДНФ. Кроме программы Minim может быть использована программа раздельной минимизации из системы ABC [10], для логической минимизации вместо программ раздельной минимизации могут быть применены мощные программы совместной минимизации, такие как ESPRESSO [3]. Описываемые далее способы 4 и 5 ориентируются на другой вид представления систем булевых функций — бинарные диаграммы решений (BDD).

Способ 4. Минимизация в классе BDD частичной многозначной функции, зависящей от булевых переменных. Доопределение BDD "вертикальное", кодирование строк матрицы *B* согласно эвристики 2. Данный способ состоит из описанных далее пяти этапов.

Этап 1. Составление по матрице B функционального описания не полностью определенной многозначной (k-значной) функции: каждой строке b^i матрицы B ставится в соответствие значение p^i многозначной переменной p, являющейся функцией $p(x_1, ..., x_n)$, зависящей от булевых переменных $x_1, ..., x_n$.

Этап 2. Построение такого графа BDD многозначной функции $p(x_1, ..., x_n)$, который имеет возможно меньшее число функциональных вершин. Листовые вершины минимизированного графа BDD соответствуют значениям k-значной функции, так как в результате минимизации числа вершин BDD каждое неопределенное значение "–" доопределяется некоторым значением p^i функции $p(x_1, ..., x_n)$. Соответствующее доопределение BDD, названное "вертикальным", будет рассмотрено на примере.

Заметим, что BDD, реализующие k-значные функции булевых аргументов, являются частным случаем k-значных функций, зависящих от k_i-значных аргументов x_i и представляющих эти функции диаграмм решений, рассмотренных в работе [11].

Этап 3. Кодирование значений k-значной функции булевыми кодами по ***эвристике 2****: соседние листовые вершины графа BDD кодируются соседними (различающимися значениями одной компоненты) m-компонентными булевыми векторами. Для оставшихся листовых вершин коды присваиваются произвольным образом.*

Этап 4. Получение системы ДНФ полностью определенных булевых функций по минимизированному графу BDD и кодированию листовых вершин.

Этап 5. Для полученной системы ДНФ выполняются программы логической минимизации.

Рассмотрим применение способа 4 для матрицы B из примера 1. В табл. 5 дана 16-значная функция $p = p(x_1, ..., x_5)$ — результат выполнения этапа 1. На этапе 2 рассмотрим только одну перестановку $<x_1, x_2, x_3, x_4, x_5>$ переменных, по которой построим BDD. Заметим, что для n переменных ($n = 5$) имеется $n!$ (факториал числа n) перестановок и задача поиска перестановки, обеспечивающей минимальное число функциональных вершин BDD, является сложной комбинаторной задачей, которой посвящено большое число работ для случая 2-знач-

Таблица 5

Частичная многозначная функция и кодирование ее значений (способ 4)

$x_1x_2x_3x_4x_5$	Многозначная функция p	Кодирующие переменные p_j^i	Значения кодирующих переменных, кодирование по эвристике 2
0 0 0 0 0	p^1	$p_1^1\ p_2^1\ p_3^1\ p_4^1$	1 1 0 1
0 0 1 0 1	p^2	$p_1^2\ p_2^2\ p_3^2\ p_4^2$	1 0 0 1
0 1 0 1 0	p^3	$p_1^3\ p_2^3\ p_3^3\ p_4^3$	0 1 1 0
0 1 1 0 0	p^4	$p_1^4\ p_2^4\ p_3^4\ p_4^4$	0 1 0 0
1 0 0 0 1	p^5	$p_1^5\ p_2^5\ p_3^5\ p_4^5$	0 0 1 0
1 0 1 1 0	p^6	$p_1^6\ p_2^6\ p_3^6\ p_4^6$	1 0 1 0
1 1 0 0 0	p^7	$p_1^7\ p_2^7\ p_3^7\ p_4^7$	0 0 0 0
0 0 0 0 1	p^8	$p_1^8\ p_2^8\ p_3^8\ p_4^8$	1 1 0 0
0 0 1 1 0	p^9	$p_1^9\ p_2^9\ p_3^9\ p_4^9$	1 1 1 0
0 1 0 0 0	p^{10}	$p_1^{10}\ p_2^{10}\ p_3^{10}\ p_4^{10}$	0 1 1 1
0 1 1 0 1	p^{11}	$p_1^{11}\ p_2^{11}\ p_3^{11}\ p_4^{11}$	0 1 0 1
1 0 0 1 0	p^{12}	$p_1^{12}\ p_2^{12}\ p_3^{12}\ p_4^{12}$	0 0 1 1
1 0 1 0 0	p^{13}	$p_1^{13}\ p_2^{13}\ p_3^{13}\ p_4^{13}$	1 1 1 1
1 1 0 0 1	p^{14}	$p_1^{14}\ p_2^{14}\ p_3^{14}\ p_4^{14}$	0 0 0 1
0 0 0 1 0	p^{15}	$p_1^{15}\ p_2^{15}\ p_3^{15}\ p_4^{15}$	1 0 1 1
0 0 1 0 0	p^{16}	$p_1^{16}\ p_2^{16}\ p_3^{16}\ p_4^{16}$	1 0 0 0
0 1 0 0 1	–	– – – –	
0 1 1 1 0	–	– – – –	
1 0 0 0 0	–	– – – –	
1 0 1 0 1	–	– – – –	
1 1 0 1 0	–	– – – –	
0 0 0 1 1	–	– – – –	
0 0 1 1 1	–	– – – –	
0 1 0 1 1	–	– – – –	
0 1 1 1 1	–	– – – –	
1 0 0 1 1	–	– – – –	
1 0 1 1 1	–	– – – –	
1 1 0 1 1	–	– – – –	
1 1 1 0 0	–	– – – –	
1 1 1 0 1	–	– – – –	
1 1 1 1 0	–	– – – –	
1 1 1 1 1	–	– – – –	

ных (булевых) функций. Краткий обзор таких работ представлен в [4, с. 39].

На этапе 2 требуется найти граф BDD функции с минимальным числом функциональных вершин. Граф BDD задает последовательные дизъюнктивные разложения функции $p(x_1, ..., x_n)$, в примере $n = 5$. Для булевых функций такие разложения называются разложениями Шеннона. По аналогии с булевыми функциями определим разложение Шеннона для k-значной функции. *Разложением Шеннона* полностью либо не полностью определенной k-значной функции $p = p(\boldsymbol{x}) = p(x_1, x_2, ..., x_n)$, зависящей от булевых переменных $x_1, x_2, ..., x_n$, по переменной x_i назовем представление

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

Многозначные функции $p_0 = p(x_1, ..., x_{i-1}, 0, x_{i+1}, ..., x_n)$, $p_1 = p(x_1, ..., x_{i-1}, 1, x_{i+1}, ..., x_n)$ в правой части (1) называются *остаточными функциями* либо *кофакторами* (*cofactors*, англ.) разложения по переменной x_i. Они получаются из функции p подстановкой вместо переменной x_i константы 0 и 1 соответственно. Каждый из кофакторов p_0 и p_1 может быть разложен по одной из переменных из множества $\{x_1, ..., x_{i-1}, x_{i+1}, ..., x_n\}$. Процесс разложения кофакторов заканчивается, когда все n переменных будут использованы для разложения, либо когда все кофакторы выродятся до констант $p^1, ..., p^k$. На каждом шаге разложения выполняется сравнение на равенство полученных кофакторов.

Кофакторы последовательных разложений исходной 16-значной функции $p(x_1, x_2, x_3, x_4, x_5)$ (табл. 5) по переменным x_1, x_2, x_3, x_4 приведены в табл. 6—9.

Таблица 6

Остаточные функции разложения по переменной x_1

$x_2x_3x_4x_5$	$x_1 = 0$	$x_1 = 1$
	p	
0 0 0 0	p^1	–
0 0 0 1	p^8	p^5
0 0 1 0	p^{15}	p^{12}
0 0 1 1	–	–
0 1 0 0	p^{16}	p^{13}
0 1 0 1	p^2	–
0 1 1 0	p^9	p^6
0 1 1 1	–	–
1 0 0 0	p^{10}	p^7
1 0 0 1	–	p^{14}
1 0 1 0	p^3	–
1 0 1 1	–	–
1 1 0 0	p^4	–
1 1 0 1	p^{11}	–
1 1 1 0	–	–
1 1 1 1	–	–

Таблица 7

Остаточные функции разложения по переменным x_1, x_2

$x_3\ x_4\ x_5$	$x_1 = 0$, $x_2 = 0$	$x_1 = 0$, $x_2 = 1$	$x_1 = 1$, $x_2 = 0$	$x_1 = 1$, $x_2 = 1$
	p			
0 0 0	p^1	p^{10}	–	p^7
0 0 1	p^8	–	p^5	p^{14}
0 1 0	p^{15}	p^3	p^{12}	–
0 1 1	–	–	–	–
1 0 0	p^{16}	p^4	p^{13}	–
1 0 1	p^2	p^{11}	–	–
1 1 0	p^9	–	p^6	–
1 1 1	–	–	–	–

Таблица 8

Остаточные функции разложения по переменным x_1, x_2, x_3

x_4x_5	$x_1 = 0$ $x_2 = 0$ $x_3 = 0$	$x_1 = 0$ $x_2 = 0$ $x_3 = 1$	$x_1 = 0$ $x_2 = 1$ $x_3 = 0$	$x_1 = 0$ $x_2 = 1$ $x_3 = 1$	$x_1 = 1$ $x_2 = 0$ $x_3 = 0$	$x_1 = 1$ $x_2 = 0$ $x_3 = 1$	$x_1 = 1$ $x_2 = 1$ $x_3 = 0$	$x_1 = 1$ $x_2 = 1$ $x_3 = 1$
	p							
0 0	p^1	p^{16}	p^{10}	p^4	–	p^{13}	p^7	–
0 1	p^8	p^2	–	p^{11}	p^5	–	p^{14}	–
1 0	p^{15}	p^9	p^3	–	p^{12}	p^6	–	–
1 1	–	–	–	–	–	–	–	–

Построим граф BDD функции $p(x_1, x_2, x_3, x_4, x_5)$ из табл. 5 для последовательности (перестановки) переменных $<x_1, x_2, x_3, x_4, x_5>$ (рис. 1). Вершины-кофакторы будем называть функциональными вершинами в BDD в отличие от вершин-переменных, по которым проводится разложение. Листовые вершины в графе BDD будут задавать определенные либо неопределенные "–" значения функции. Граф BDD, в котором не проведено сокращение функциональных вершин, показан на рис. 1.

Общий рисунок BDD является большим, поэтому разделен на три части. Одинаковыми функциональными вершинами (кофакторами) являются G^{22}, G^{28}, G^{29}, G^{30}. Эти вершины не представлены одной вершиной, чтобы показать, что одинаковые неопределенные кофакторы могут быть доопределены до различных полностью определенных кофакторов. Алгоритм "вертикального" доопределения BDD, реализующей частично определенную многозначную функцию, состоит в замене неопределенных значений листовых вершин определенными значениями соседних определенных листовых вершин. Соседними являются вершины пары кофакторов, получающихся в результате разложения Шеннона какого-либо кофактора. В примере кофактор G^{16} имеет две листовые вершины: p^{15} и "–".

Таблица 9

Остаточные функции разложения по переменным x_1, x_2, x_3, x_4

x_5	$x_1=0$ $x_2=0$ $x_3=0$ $x_4=0$	$x_1=0$ $x_2=0$ $x_3=0$ $x_4=1$	$x_1=0$ $x_2=0$ $x_3=1$ $x_4=0$	$x_1=0$ $x_2=0$ $x_3=1$ $x_4=1$	$x_1=0$ $x_2=1$ $x_3=0$ $x_4=0$	$x_1=0$ $x_2=1$ $x_3=0$ $x_4=1$	$x_1=0$ $x_2=1$ $x_3=1$ $x_4=0$	$x_1=0$ $x_2=1$ $x_3=1$ $x_4=1$	$x_1=1$ $x_2=0$ $x_3=0$ $x_4=0$	$x_1=1$ $x_2=0$ $x_3=0$ $x_4=1$	$x_1=1$ $x_2=0$ $x_3=1$ $x_4=0$	$x_1=1$ $x_2=0$ $x_3=1$ $x_4=1$	$x_1=1$ $x_2=1$ $x_3=0$ $x_4=0$	$x_1=1$ $x_2=0$ $x_3=1$ $x_4=1$	$x_1=1$ $x_2=1$ $x_3=1$ $x_4=0$	$x_1=1$ $x_2=1$ $x_3=1$ $x_4=1$
	p															
0 1	p^1 p^8	p^{15} –	p^{16} p^2	p^9 –	p^{10} –	p^3 –	p^4 p^{11}	– –	– p^5	p^{12} –	p^{13} –	p^6 –	p^7 p^{14}	– –	– –	– –

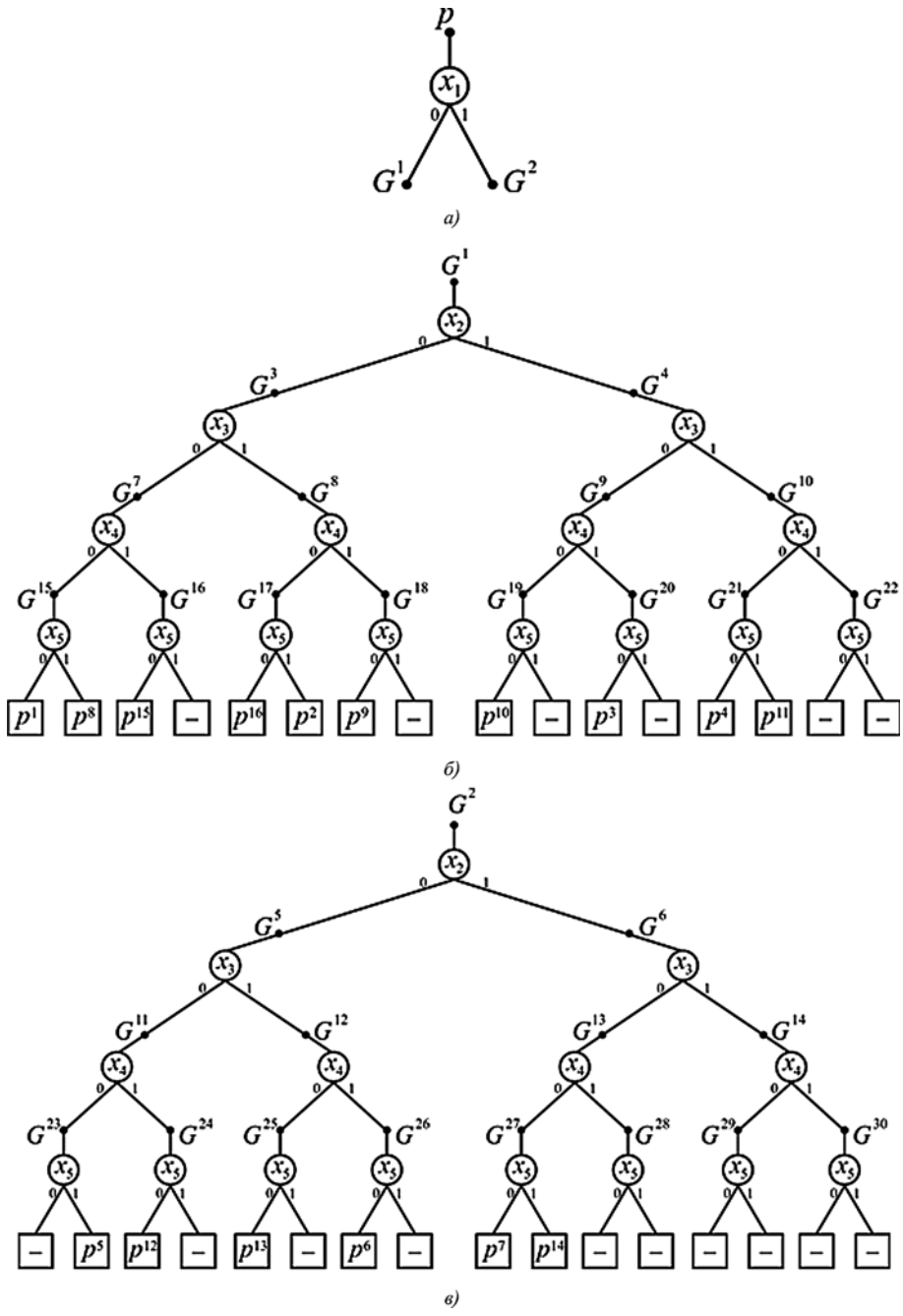

Рис. 1. Бинарная диаграмма решений частичной многозначной функции *p*:
а — разложение функции *p* по переменной x_1; *б* — разложение кофактора G^1; *в* — разложение кофактора G^2

Поэтому неопределенное значение заменяется значением p^{15} соседней листовой вершины. Для кофактора G^{19} соседнее неопределенное значение заменяется значением p^9. Аналогично для других кофакторов, которые имеют соседние листовые вершины, одна из которых является определенной p^i, другая — неопределенной "–". Для кофактора G^{10} соседними являются кофакторы G^{21}, G^{22}, поэтому кофактор G^{22} доопределяется до кофактора G^{21}, т. е. неопределенные листовые вершины кофактора G^{22} заменяются на p^4, p^{11}. Для кофактора G^6 в результате подобных пошаговых доопределений листовыми вершинами становятся p^7, p^{14}, т. е. неопределенные кофакторы G^{28}, G^{29}, G^{30} доопределяются до полностью определенного кофактора G^{28}.

Уравнения, описывающие доопределенный граф BDD (рис. 2), имеют следующий вид:

$$
\begin{gathered}
p = \overline{x}_1 G^1 \vee x_1 G^2; \\
G^1 = \overline{x}_2 G^3 \vee x_2 G^4;\ G^2 = \overline{x}_2 G^5 \vee x_2 G^6; \\
G^3 = \overline{x}_3 G^7 \vee x_3 G^8; \\
G^4 = \overline{x}_3 G^9 \vee x_3 G^{10};\ G^5 = \overline{x}_3 G^{11} \vee x_3 G^{12}; \\
G^6 = \overline{x}_5 p^7 \vee x_3 p^{14}; \\
G^7 = \overline{x}_4 G^{15} \vee x_4 p^{15};\ G^8 = \overline{x}_4 G^{17} \vee x_4 p^9; \\
G^9 = \overline{x}_4 p^{10} \vee x_4 p^3; \\
G^{10} = \overline{x}_5 p^4 \vee x_5 p^{11};\ G^{11} = \overline{x}_5 p^5 \vee x_5 p^{12}; \\
G^{12} = \overline{x}_4 p^{13} \vee x_4 p^6; \\
G^{15} = \overline{x}_5 p^1 \vee x_5 p^8;\ G^{17} = \overline{x}_5 p^{16} \vee x_5 p^2.
\end{gathered}
\qquad (2)
$$

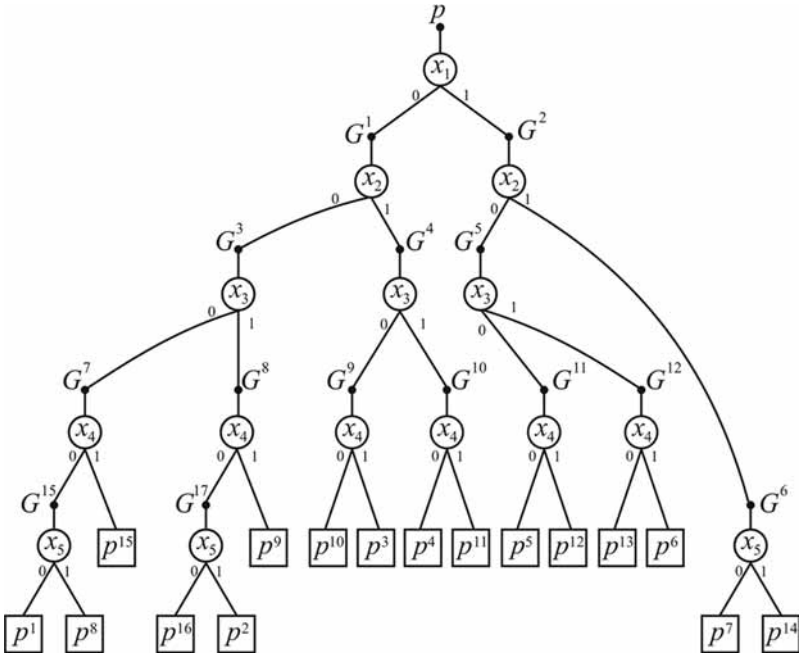

Рис. 2. Доопределенный граф BDD многозначной функции *p*

На этапе 3 осуществляется кодирование соседних листовых вершин соседними кодами. В примере соседним вершинам p^1, p^8 присваиваются соседние коды (1 1 0 1), (1 1 0 0), соответственно. *Соседние коды* (булевы векторы) различаются значениями в одном разряде, в данном случае в четвертом. Для других соседних пар: для пары p^{16}, p^2 кодами являются соседние векторы (1 0 0 0), (1 0 0 1); для пары p^4, p^{11} — соседние векторы (0 1 0 0), (0 1 0 1); для пары p^7, p^{14} — соседние векторы (0 0 0 0), (0 0 0 1). Булевы коды всех значений многозначной переменной p даны в правой части табл. 5.

На этапе 4 осуществляется замена каждого многозначного кофактора своей подсистемой булевых функций (табл. 10), зависящих от соответствующих множеств булевых переменных.

Заменим в графе BDD (см. рис. 2) многозначные значения булевыми значениями согласно табл. 10. Получим граф BDD (рис. 3), представляющий векторную булеву функцию $\boldsymbol{c}(\boldsymbol{x}) = (c_1(x), c_2(\boldsymbol{x}), c_3(\boldsymbol{x}), c_4(\boldsymbol{x}))$, $\boldsymbol{x} = (x_1, x_2, x_3, x_4)$.

Таблица 10

Представление многозначных кофакторов в виде подсистем булевых функций

Многозначная функция (кофактор)	Подсистема булевых функций
p	$c_1\ c_2\ c_3\ c_4$
G^1	$g_1\ g_2\ g_3\ g_4$
G^2	$g_5\ g_6\ g_7\ g_8$
G^3	$g_9\ g_{10}\ g_{11}\ g_{12}$
G^4	$g_{13}\ g_{14}\ g_{15}\ g_{16}$
G^5	$g_{17}\ g_{18}\ g_{19}\ g_{20}$
G^6	$g_{21}\ g_{22}\ g_{23}\ g_{24}$
G^7	$g_{25}\ g_{26}\ g_{27}\ g_{28}$
G^8	$g_{29}\ g_{30}\ g_{31}\ g_{32}$
G^9	$g_{33}\ g_{34}\ g_{35}\ g_{36}$
G^{10}	$g_{37}\ g_{38}\ g_{39}\ g_{40}$
G^{11}	$g_{41}\ g_{42}\ g_{43}\ g_{44}$
G^{12}	$g_{45}\ g_{46}\ g_{47}\ g_{48}$
G^{15}	$g_{49}\ g_{50}\ g_{51}\ g_{52}$
G^{17}	$g_{53}\ g_{54}\ g_{55}\ g_{56}$

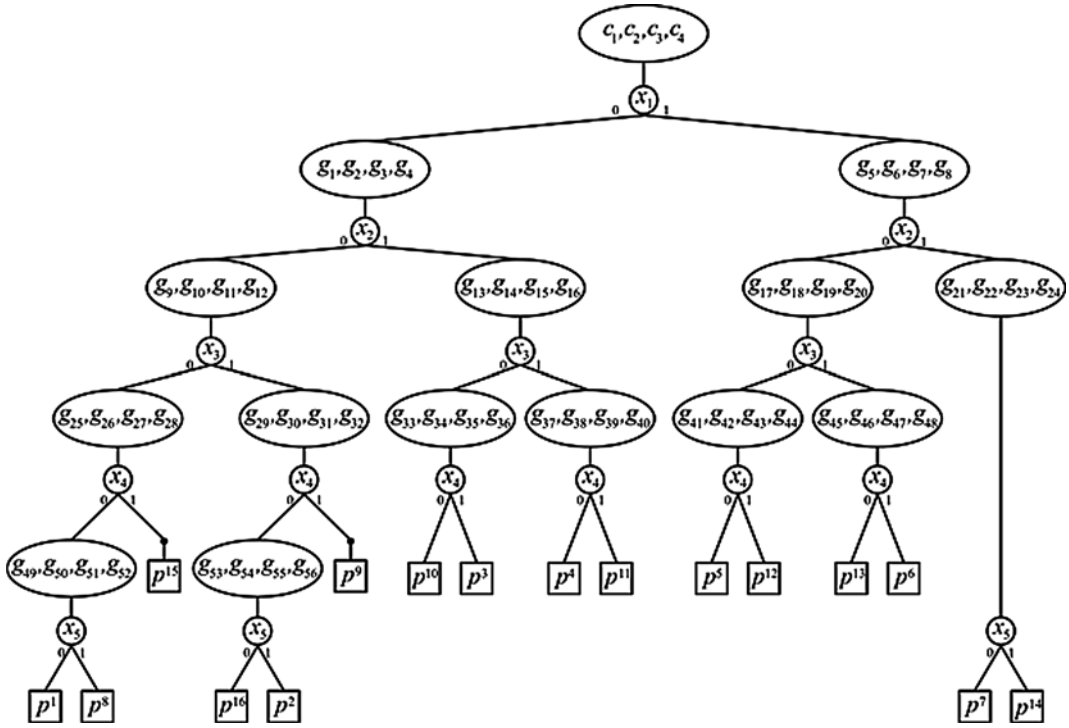

Рис. 3. BDD многозначной функции после замены многозначных кофакторов булевыми переменными

Уравнения (2) разложений Шеннона многозначной функции и ее многозначных кофакторов заменяются следующими уравнениями с булевыми переменными:

$$c_1 = \overline{x}_1 g_1 \vee x_1 g_5;\ c_2 = \overline{x}_1 g_2 \vee x_1 g_6;$$
$$c_3 = \overline{x}_1 g_3 \vee x_1 g_7;\ c_4 = \overline{x}_1 g_4 \vee x_1 g_8;$$
$$g_1 = \overline{x}_2 g_9 \vee x_2 g_{13};\ g_2 = \overline{x}_2 g_{10} \vee x_2 g_{14};$$
$$g_3 = \overline{x}_2 g_{11} \vee x_2 g_{15};\ g_4 = \overline{x}_2 g_{12} \vee x_2 g_{16};$$
$$g_5 = \overline{x}_2 g_{17} \vee x_2 g_{21};\ g_6 = \overline{x}_2 g_{18} \vee x_2 g_{22};$$
$$g_7 = \overline{x}_2 g_{19} \vee x_2 g_{23};\ g_8 = \overline{x}_2 g_{20} \vee x_2 g_{24};$$
$$g_9 = \overline{x}_3 g_{25} \vee x_3 g_{29};\ g_{10} = \overline{x}_3 g_{26} \vee x_3 g_{30};$$
$$g_{11} = \overline{x}_3 g_{27} \vee x_3 g_{31};\ g_{12} = \overline{x}_3 g_{28} \vee x_3 g_{32};$$
$$g_{13} = \overline{x}_3 g_{33} \vee x_3 g_{37};\ g_{14} = \overline{x}_3 g_{34} \vee x_3 g_{38};$$
$$g_{15} = \overline{x}_3 g_{35} \vee x_3 g_{39};\ g_{16} = \overline{x}_3 g_{36} \vee x_3 g_{40};$$
$$g_{17} = \overline{x}_3 g_{41} \vee x_3 g_{45};\ g_{18} = \overline{x}_3 g_{42} \vee x_3 g_{46};$$
$$g_{19} = \overline{x}_3 g_{43} \vee x_3 g_{47};\ g_{20} = \overline{x}_3 g_{44} \vee x_3 g_{48};$$
$$g_{21} = \overline{x}_5 p_1^7 \vee x_5 p_1^{14};\ g_{22} = \overline{x}_5 p_2^7 \vee x_5 p_2^{14};$$
$$g_{23} = \overline{x}_5 p_3^7 \vee x_5 p_3^{14};\ g_{24} = \overline{x}_5 p_5^7 \vee x_4 p_4^{14};$$
$$g_{25} = \overline{x}_4 g_{49} \vee x_4 p_1^{15};\ g_{26} = \overline{x}_4 g_{50} \vee x_4 p_2^{15};$$
$$g_{27} = \overline{x}_4 g_{51} \vee x_4 p_3^{15};\ g_{28} = \overline{x}_4 g_{52} \vee x_4 p_4^{15};$$
$$g_{29} = \overline{x}_4 g_{53} \vee x_4 p_1^{9};\ g_{30} = \overline{x}_4 g_{54} \vee x_4 p_2^{9};$$
$$g_{31} = \overline{x}_4 g_{55} \vee x_4 p_3^{9};\ g_{32} = \overline{x}_4 g_{56} \vee x_4 p_4^{9};$$
$$g_{33} = \overline{x}_4 p_1^{10} \vee x_4 p_1^{3};\ g_{34} = \overline{x}_4 p_2^{10} \vee x_4 p_2^{3};$$
$$g_{35} = \overline{x}_4 p_3^{10} \vee x_4 p_3^{3};\ g_{36} = \overline{x}_4 p_4^{10} \vee x_4 p_4^{3};$$
$$g_{37} = \overline{x}_5 p_1^{4} \vee x_5 p_1^{11};\ g_{38} = \overline{x}_5 p_2^{4} \vee x_5 p_2^{11};$$
$$g_{39} = \overline{x}_5 p_3^{4} \vee x_5 p_3^{11};\ g_{40} = \overline{x}_5 p_4^{4} \vee x_5 p_4^{11};$$
$$g_{41} = \overline{x}_4 p_1^{5} \vee x_4 p_1^{12};\ g_{42} = \overline{x}_4 p_2^{5} \vee x_4 p_2^{12};$$
$$g_{43} = \overline{x}_4 p_3^{5} \vee x_4 p_3^{12};\ g_{44} = \overline{x}_4 p_4^{5} \vee x_4 p_4^{12};$$
$$g_{45} = \overline{x}_4 p_1^{13} \vee x_4 p_1^{6};\ g_{46} = \overline{x}_4 p_2^{13} \vee x_4 p_2^{6};$$
$$g_{47} = \overline{x}_4 p_3^{13} \vee x_4 p_3^{6};\ g_{48} = \overline{x}_4 p_4^{13} \vee x_4 p_4^{6};$$
$$g_{49} = \overline{x}_5 p_1^{1} \vee x_5 p_1^{8};\ g_{50} = \overline{x}_5 p_2^{1} \vee x_5 p_2^{8};$$
$$g_{51} = \overline{x}_5 p_3^{1} \vee x_5 p_3^{8};\ g_{52} = \overline{x}_5 p_4^{1} \vee x_5 p_4^{8};$$
$$g_{53} = \overline{x}_5 p_1^{16} \vee x_5 p_1^{2};$$
$$g_{54} = \overline{x}_5 p_2^{16} \vee x_5 p_2^{2};\ g_{55} = \overline{x}_5 p_3^{16} \vee x_5 p_3^{2};$$
$$g_{56} = \overline{x}_5 p_4^{16} \vee x_5 p_4^{2}. \qquad (3)$$

Подстановка значений p_j^i из табл. 5 и элиминация (устранение) промежуточных переменных позволяют получить систему ДНФ полностью определенных функций, заданную в левой части табл. 11. Покажем, как упрощаются уравнения для кофакторов $g_{53}, g_{54}, g_{55}, g_{56}$ при подстановке значений $p^{16} = (1\ 0\ 0\ 0)$, $p^2 = (1\ 0\ 0\ 1)$ из табл. 5: $g_{53} = \overline{x}_5 1 \vee x_5 1 = 1$; $g_{54} = \overline{x}_5 0 \vee x_5 0 = 0$; $g_{55} = \overline{x}_5 0 \vee x_5 0 = 0$; $g_{56} = \overline{x}_5 0 \vee x_5 1 = x_5$. Присвоение соседних кодов ориентируется на возможно большее сокращение графа BDD и, соответственно,

Таблица 11

Результаты логической оптимизации в способе 4 (пример 1)

Доопределение по BDD, "вертикальное", кодирование по эвристике 2		Раздельно минимизированные ДНФ		BDDI-представление
$x_1x_2x_3x_4x_5$	$c_1c_2c_3c_4$	$x_1x_2x_3x_4x_5$	$c_1c_2c_3c_4$	

0 0 0 0 0
0 0 1 0 1
0 1 0 1 –
0 1 1 – 0
1 0 0 0 –
1 0 1 1 –
1 1 0 0 0
0 0 0 0 1
0 0 1 1 –
0 1 0 0 –
0 1 1 – 1
1 0 0 1 –
1 0 1 0 –
1 1 – – 1
0 0 0 1 –
0 0 1 0 0
	1 1 0 1
	1 0 0 1
	0 1 1 0
	0 1 0 0
	0 0 1 0
	1 0 1 0
	0 0 0 0
	1 1 0 0
	1 1 1 0
	0 1 1 1
	0 1 0 1
	0 0 1 1
	1 1 1 1
	0 0 0 1
	1 0 1 1
	1 0 0 0
		– 0 0 1 –
		1 1 – – 1
		– 1 1 – 1
		– 1 – 0 1
		– – 1 0 1
		0 – 0 0 0
		– 0 – 1 –
		0 1 0 – –
		1 0 – – –
		1 0 1 0 –
		0 – 0 0 –
		0 – 1 1 –
		0 1 – – –
		0 0 – – –
		– 0 1 – –
			0 0 0 1
			0 0 0 1
			0 0 0 1
			0 0 0 1
			0 0 0 1
			0 0 0 1
			0 0 1 0
			0 0 1 0
			0 0 1 0
			0 1 0 1
			0 1 0 0
			0 1 0 0
			0 1 0 0
			1 0 0 0
			1 0 0 0
				c1=^x1*^x2+x1*s1;
				c2=^x1*s2+x1*s3;
				c3=^x1*s4+x1*^x2;
				c4=^x1*s6+x1*s7;
				s7=^x2*^s10+x2*x5;
				s2=^x2*s10+x2;
				s6=^x2*s15+x2*s16;
				s3=^x2*s9;
				s1=^x2*x3;
				s4=^x2*x4+x2*^x3;
				s10=^x3*^x4+x3*x4;
				s16=^x3*^x4+x3*x5;
				s15=^x3*s20+x3*^s20;
				s9=x3*^x4;
				s20=^x4*^x5+x4;

Схема 10		**Схема 11**		**Схема 12**

на сокращение уравнений, входящих в BDDI-представление.

Результаты логической оптимизации данной системы, полученные на этапе 5, даны в средней и правой частях табл. 11.

Способ 5. Минимизация в классе BDD частичной многозначной функции, зависящей от булевых переменных, доопределение BDD "горизонтальное", кодирование строк матрицы *B* согласно эвристики 2. В реализации способа 5 выполняются этапы 1 — 3 из способа 4. Затем по полученной системе ДНФ определяется код каждой строки b^i матрицы *B*. Для этого находится элементарная конъюнкция, которую *имплицирует* полная элементарная конъюнкция, соответствующая строке b^i. Например, строке (1 1 0 0 1) матрицы *B* соответствует полная элементарная конъюнкция $x_1x_2\bar{x}_3\bar{x}_4x_5$. Легко проверить, что выполняется единственная импликация $x_1x_2\bar{x}_3\bar{x}_4x_5 \to x_1x_2x_5$. Троичный вектор (1 1 – – 1), соответствующий $x_1x_2x_5$, имеет код (0 0 0 1). Следовательно, закодируем строку (1 1 0 0 1) кодом (0 0 0 1) (см. систему ДНФ в левой части табл. 11). Выполнив такую процедуру для каждой строки матрицы *B*, получим матричное описание частичной векторной функции в левой части табл. 12.

На этапе 4 способа 5 будем выполнять логическую оптимизацию тремя программами. Функциональное описание (схема 13) получается в результате построения BDD-описания полностью определенной функции с помощью программы из работы [4, с. 201]. Эта программа реализует алгоритм доопределения частичной BDD на основе "горизонтального" доопределения. Алгоритм доопределения сводится в работе [4, с. 64] к задаче раскраски вершин неориентированного графа в минимальное число красок (цветов) так, чтобы соседние вершины графа были раскрашены разными красками. Аналогичные алгоритмы могут быть использованы и при "горизонтальном" доопределении BDD неполностью определенной *k*-значной функции.

Результаты кодирования матрицы *B* рассмотренными способами 1—5 решения задач 2, 3 приведены в табл. 13.

3. Решение задачи 4

Функциональные описания схем 1—15 были получены программами логической оптимизации, имеющимися в системе FLC-2. Эти описания являются минимизированными ДНФ либо BDDI-описаниями на языке SF векторных полностью определенных функций. ДНФ представлялись матричными описаниями, BDDI-описания — логическими уравнениями в булевом базисе. Затем осуществлялся перевод оптимизированных SF-описаний в описания на языке VHDL [12]. Схемная реализация (синтез) полученных VHDL-описаний выполнялась с помощью синтезатора LeonardoSpectrum в библиотеке проектирования КМОП СБИС. Библиотека логических КМОП-элементов приведена в работе [6]. Для каждого описания схемы синтез осуществлялся с одни-

Таблица 12

Результаты логической оптимизации в способе 5 (пример 1)

Частичная векторная функция, кодирование по эвристике 2		BDD, реализующая частичную векторную функцию	Раздельно минимизированные ДНФ		BDDI-представление
$x_1x_2x_3x_4x_5$	$c_1c_2c_3c_4$		$x_1x_2x_3x_4x_5$	$c_1c_2c_3c_4$	
0 0 0 0 0	1 1 0 1	c1=^x2*s18;	- 1 - 0 1	0 0 0 1	c1=^x3*s0+x3*^x2;
0 0 1 0 1	1 0 0 1	c2=^x4*s11+x4*s12;	0 - 1 0 1	0 0 0 1	c2=^x3*s2+x3*s3;
0 1 0 1 0	0 1 1 0	c3=^x4*s13+x4;	1 - 1 0 0	0 0 0 1	c3=^x3*s4+x3*s5;
0 1 1 0 0	0 1 0 0	c4=^x5*s7+x5*s8;	0 - 0 0 0	0 0 0 1	c4=^x3*s6+x3*s7;
1 0 0 0 1	0 0 1 0	s18=^x1+x1*x3;	- 0 0 1 0	0 0 0 1	s3=^x4*^s0+x4*s13;
1 0 1 1 0	1 0 1 0	s11=^x2*s20+x2*s18;	- - - 1 -	0 0 1 0	s2=^x4*^x1+x4*x2;
1 1 0 0 0	0 0 0 0	s12=^x2*s22+x2;	0 1 0 - -	0 0 1 0	s4=^x4*s11+x4;
0 0 0 0 1	1 1 0 0	s13=^x2*x1+x2*s20;	1 0 - - -	0 0 1 0	s7=^x4*s14;
0 0 1 1 0	1 1 1 0	s7=^x4*s20+x4*s16;	1 - 1 - -	0 0 1 0	s6=^x4*s15+x4*s16;
0 1 0 0 0	0 1 1 1	s8=^x4*s12;	- 1 - 1 -	0 1 0 0	s5=^x4*x1+x4;
0 1 1 0 1	0 1 0 1	s20=^x1*^x3+x1*x3;	1 - 1 0 -	0 1 0 0	s0=^x1*^x2;
1 0 0 1 0	0 0 1 1	s22=^x1*x3;	0 1 - - -	0 1 0 0	s15=^x1*s21+x1*s24;
1 0 1 0 0	1 1 1 1	s16=^x2*^x3;	0 - 1 1 -	0 1 0 0	s11=^x1*x2+x1*^x2;
1 1 0 0 1	0 0 0 1		0 - 0 0 -	0 1 0 0	s14=^x1*x5+x1*s21;
0 0 0 1 0	1 0 1 1		0 0 - - -	1 0 0 0	s13=^x1+x1*x2;
0 0 1 0 0	1 0 0 0		- 0 1 - -	1 0 0 0	s16=^x2*^x5;
0 1 0 0 1	- - - -				s21=^x2*^x5+x2;
0 1 1 1 0	- - - -				s24=x2*x5;
1 0 0 0 0	- - - -				
1 0 1 0 1	- - - -				
1 1 0 1 0	- - - -				
0 0 0 1 1	- - - -				
0 0 1 1 1	- - - -				
0 1 0 1 1	- - - -				
0 1 1 1 1	- - - -				
1 0 0 1 1	- - - -				
1 0 1 1 1	- - - -				
1 1 0 1 1	- - - -				
1 1 1 0 0	- - - -				
1 1 1 0 1	- - - -				
1 1 1 1 0	- - - -				
1 1 1 1 1	- - - -				
		Схема 13	**Схема 14**		**Схема 15**

Таблица 13

Коды матрицы *B* (пример 1)

Матрица B	Способы 1, 2	Способ 3	Способы 4, 5
$x_1x_2x_3x_4x_5$	$c_1c_2c_3c_4$	$c_1c_2c_3c_4$	$c_1c_2c_3c_4$
0 0 0 0 0	0 0 0 0	0 0 0 0	1 1 0 1
0 0 1 0 1	0 0 0 1	0 0 1 1	1 0 0 1
0 1 0 1 0	0 0 1 0	1 1 1 1	0 1 1 0
0 1 1 0 0	0 0 1 1	0 1 1 0	0 1 0 0
1 0 0 0 1	0 1 0 0	1 1 1 0	0 0 1 0
1 0 1 1 0	0 1 0 1	1 1 0 1	1 0 1 0
1 1 0 0 0	0 1 1 0	1 1 0 0	0 0 0 0
0 0 0 0 1	0 1 1 1	1 0 1 1	1 1 0 0
0 0 1 1 0	1 0 0 0	0 1 1 1	1 1 1 0
0 1 0 0 0	1 0 0 1	1 0 0 0	0 1 1 1
0 1 1 0 1	1 0 1 0	1 0 0 1	0 1 0 1
1 0 0 1 0	1 0 1 1	0 1 0 1	0 0 1 1
1 0 1 0 0	1 1 1 0	0 0 0 1	1 1 1 1
1 1 0 0 1	1 1 0 1	1 0 1 0	0 0 0 1
0 0 0 1 0	1 1 1 0	0 1 0 0	1 0 1 1
0 0 1 0 0	1 1 1 1	0 0 1 0	1 0 0 0

ми и теми же опциями управления синтезом. Для каждой полученной схемы подсчитывалась площадь схемы S_{ASIC} (в условных единицах площади логических элементов) и временная задержка τ (нс). Лучшие решения отмечены символом "*" (табл. 14). Схема 14 имеет наименьшую площадь, она получена в результате применения способа 5 путем оптимизации BDD частичных функций.

Результаты вычислительного эксперимента (см. табл. 14) показывают, что при одинаковом кодировании строк матрицы *B*, но при разных доопределениях частичной векторной функции $\boldsymbol{c}(\boldsymbol{x})$, сложности схем и их задержки могут иметь значительные различия.

В рамках исследования был проведен эксперимент, когда по матрицам *B*, *C* составлялись VHDL-описания систем частичных функций, которые отправлялось на синтез в LeonardoSpectrum. В этом случае технологически независимую оптимизацию

Таблица 14

Результаты синтеза логических схем (пример 1)

Способ решения задач 2 ,3	Вид логической минимизации	Сложность функционального описания			Логическая схема		Номер схемы
		Дизъюнкций	Конъюнкций	Литералов	S_{ASIC}	τ, нс	
1	Нет	29	132	165	14 274	3,89	1
	ДНФ	18	73	95	11 874	3,30	2
	BDDI	17	42	84	12 187	2,62	3
2	Нет	28	76	108	12 644	3,82	4
	ДНФ	21	50	75	10 178	3,41	5
	BDDI	26	49	104	13 269	2,92	6
3	Нет	28	64	96	7137	2,62	7
	ДНФ	17	26	47	7449	2,83	8
	BDDI	20	33	75	7466	2,20	9
4	Нет	28	103	135	10 490	2,61	10
	ДНФ	12	30	46	7226	2,85	11
	BDDI	12	25	52	7048	2,38	12
5	BDD	*9	*19	*41	6478	*1,79	13
	ДНФ	12	28	44	*5569	2,00	14
	BDDI	14	28	60	6930	3,83	15

выполнял синтезатор. Оказалось, что во всех случаях технологически независимая оптимизация системы FLC-2 приводила к лучшим результатам синтеза. Например, для примера 1 (способ 5) было составлено следующее VHDL-описание системы частичных функций:

```
Library IEEE;
use IEEE.STD _ LOGIC _ 1164.all;
entity Sposob _ 5 is
  port (x: in std _ logic _ vector (1 to 5);
        c: out std _ logic _ vector (1 to 4));
  end;
architecture BEH of Sposob _ 5 is
begin
c <= "1101" when x = "00000" else
     "1001" when x = "00101" else
     "0110" when x = "01010" else
     "0100" when x = "01100" else
     "0010" when x = "10001" else
     "1010" when x = "10110" else
     "0000" when x = "11000" else
     "1100" when x = "00001" else
     "1110" when x = "00110" else
     "0111" when x = "01000" else
     "0101" when x = "01101" else
     "0011" when x = "10010" else
     "1111" when x = "10100" else
     "0001" when x = "11001" else
     "1011" when x = "00010" else
     "1000" when x = "00100" else
     "----";
end BEH;
```

Результатом синтеза оказалась схема со следующими параметрами: $S_{ASIC} = 8002$, $\tau = 3{,}15$ нс, данная схема проигрывает всем вариантам способа 5 (схемам 13—15) по площади и схемам 13, 14 по задержке.

4. Синтез схем обратного преобразования

В табл. 15 приведены результаты синтеза схем для обратного преобразования. Лучшие результаты показал способ 5.

В данном примере лучшим способом по суммарной площади (5 569 + 5 988) схем прямого и обратного преобразования является способ 5, оптимизация в классе ДНФ, хотя доопределение было выбрано в целях минимизации многоуровневого BDD-представления функций.

5. Практический пример

Рассмотрим пример, когда матрица B имеет параметры $n = 17$, $k = 2^{16} = 65\ 536$.

Таблица 15

Результаты синтеза логических схем обратного преобразователя кода (пример 1)

Способ решения задач 2—4	Вид логической минимизации	Сложность функционального описания			Сложность схемы		Номер схемы
		Дизъюнкций	Конъюнкций	Литералов	S_{ASIC}	τ, нс	
1, 2	Нет	24	87	116	12 633	2,58	1, 4
	ДНФ	17	55	77	9804	2,51	2, 5
	BDDI	14	33	66	9893	2,45	3, 6
3	Нет	24	87	116	8934	2,43	7
	ДНФ	11	32	48	7488	2,89	8
	BDDI	11	28	56	7399	2,15	9
4, 5	Нет	24	87	116	7940	*1,98	10, 13
	ДНФ	11	32	*48	*5988	2,94	11, 14
	BDDI	*10	*24	51	6729	2,01	12, 15

Тогда $m = \lceil \log_2 k \rceil = 16$ и требуется построить кодовый преобразователь 17-разрядных двоичных слов в 16-разрядные. Оптимизация, такая же, как в способе 5 (схема 13), однако со случайным присвоением 16-разрядных кодов позволяет получить функциональное BDD-описание, содержащее 8553 двухвходовых конъюнкций, 4119 двухвходовых дизъюнкций и синтезировать схему, содержащую 43 818 транзисторов. Синтез был проведен в библиотеке проектирования системы CMOSLD [13]. Функциональное описание схемы для обратного преобразования (реализация системы СДНФ булевых функций заданной парой булевых матриц (C, B)) содержало 5473 двухвходовых конъюнкций, 2920 двухвходовых дизъюнкций, после BDDI-оптимизации и синтеза была получена логическая схема, содержащей 22 232 транзистора.

Схемная реализация того же примера кодового преобразователя для FPGA xc7k70tfbv676-1 семейства Kintex-7 [14] осуществлялась в системе автоматизированного проектирования Vivado [15], опции синтеза — Vivado Synthesis Default. Сложность схем оценивалась в числе программируемых элементов LUT-6, имеющих шесть входных переменных (LUT — Look-Up Table — таблица, реализующая логическую функцию). Сложность схемы кодового преобразователя составила 2014 LUT-6, сложность схемы обратного преобразования строк матрицы C в строки матрицы B составила 940 LUT-6.

Таким образом, построенный кодовый преобразователь позволил сократить вдвое число тактов для передачи информации по 16-разрядной шине, однако для этого потребовались соответствующие аппаратные затраты.

6. Дополнительные способы решения задач 2, 3

Рассмотренные примеры касались построения кодовых преобразователей, когда требовалось на единицу уменьшить длину кодовых слов $m = n - 1$, а число $k = 2^{n-1}$ и надо было использовать *все* 2^{n-1} различные кодирующие комбинации булевых векторов длины $m = n - 1$.

В случаях, когда k значительно меньше 2^{n-1}, появляется возможность выбора длины m кода и возможность выбора некоторого подмножества кодовых комбинаций, так как не все кодирующие комбинации заданной разрядности надо использовать. Указанные возможности позволяют также уменьшить сложность минимизированных функциональных представлений за счет того, что кодирование k булевых векторов может осуществляться не различными m-разрядными булевыми векторами, а попарно ортогональными *троичными* векторами выбранной длины m, где $\lceil \log_2 k \rceil \leqslant m \leqslant n - 1$.

Пример 2. Пусть матрицу B^2 образуют первые девять строк из уже рассмотренного примера матрицы B:

$$B^2 = \begin{matrix} x_1x_2x_3x_4x_5 \\ 0\,0\,0\,0\,0 \\ 0\,0\,1\,0\,1 \\ 0\,1\,0\,1\,0 \\ 0\,1\,1\,0\,0 \\ 1\,0\,0\,0\,1 \\ 1\,0\,1\,1\,0 \\ 1\,1\,0\,0\,0 \\ 0\,0\,0\,0\,1 \\ 0\,0\,1\,1\,0 \end{matrix}.$$

Тогда $m = \lceil \log_2 9 \rceil = 4$ и требуется использовать только девять кодирующих комбинаций из шестнадцати. Пусть кодирующие троичные векторы образуют троичную матрицу T.

Способ 7. Кодирование строк матрицы B попарно ортогональными m-разрядными троичными векторами. В способе 7 предполагается, что логической минимизации подвергается частичная векторная функция, так как на наборах, принадлежащих множеству $V^x \backslash B$, значения векторной функции не определены. Обозначим B^- через матрицу наборов, задающих множество $V^x \backslash B$ наборов. Пример кодирования строк матрицы B^2 троичными векторами приведен в табл. 16.

Таблица 16

Исходная частичная векторная функция в способе 7 (пример 2)

B^2	T
$x_1x_2x_3x_4x_5$	$c_1c_2c_3c_4$
0 0 0 0 0 0 0 1 0 1 0 1 0 1 0 0 1 1 0 0 1 0 0 0 1 1 0 1 1 0 1 1 0 0 0 0 0 0 0 1 0 0 1 1 0	0 0 0 0 – 0 0 1 – 1 1 1 – 0 1 0 – 1 0 0 – 1 1 0 – 0 1 1 1 0 0 0 – 1 0 1
B^-	– – – –

Данная частичная векторная функция может подвергаться логической минимизации программой Minim, результат минимизации — полностью определенная векторная функция — может быть обработан программой BDD_Builder.

Способ 4М (модифицированный). Минимизация в классе BDD частичной многозначной функции, зависящей от булевых переменных. Доопределение BDD "вертикальное", кодирование строк матрицы B соседними троичными векторами. При таком подходе доопределение функций, являющихся моделями функционирования кодового преобразователя, происходит в два этапа: на этапе 1 доопределяется многозначная функция, на этапе 2 — система частичных булевых функций. Кратко проиллюстрируем данный подход на примере матрицы B^2. Доопределенный граф BDD для 9-значной функции показан на рис. 4, где h_i — кофакторы разложения Шеннона компонентных функций c_1, c_2, c_3,

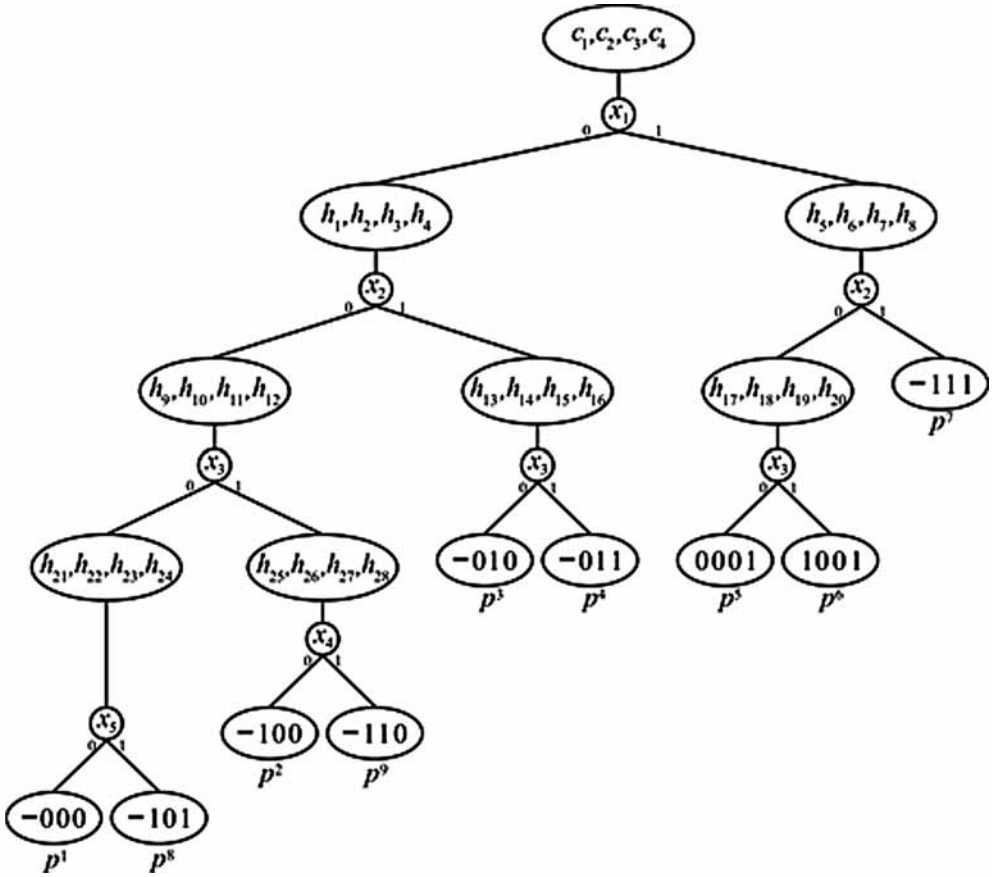

Рис. 4. Доопределенный граф BDD 9-значной функции и кодирование значений троичными векторами

Таблица 17

Частичная векторная функция в способе 4М, полученная кодированием троичными векторами (пример 2)

Компонентная функция	$x_1x_2x_3x_4x_5$	Область определения
c_1	1 0 0 – –	$M^0_{c_1}$
	1 0 1 – –	$M^1_{c_1}$
	0 0 0 – 0 0 0 0 – 1 0 0 1 1 – 0 1 0 – – 0 1 1 – – 1 1 – – –	$M^-_{c_1}$
c_2	0 0 0 – 0 0 1 0 – – 0 1 1 – – 1 0 0 – – 1 0 1 – –	$M^0_{c_2}$
	0 0 0 – 1 0 0 1 0 – 0 0 1 1 – 1 1 – – –	$M^1_{c_2}$
	Ø	$M^-_{c_2}$
c_3	0 0 0 – 0 0 0 0 – 1 0 0 1 0 – 1 0 0 – – 1 0 1 – –	$M^0_{c_3}$
	0 0 1 1 – 0 1 0 – – 0 1 1 – – 1 1 – – –	$M^1_{c_3}$
	Ø	$M^0_{c_3}$
c_4	0 0 0 – 0 0 0 1 0 – 0 0 1 1 – 0 1 0 – –	$M^0_{c_4}$
	0 0 0 – 1 0 1 1 – – 1 0 0 – – 1 0 1 – – 1 1 – – –	$M^1_{c_4}$
	Ø	$M^-_{c_4}$

c_4 (табл. 16). Практически все соседние листовые вершины закодированы соседними троичными векторами. Например, соседние листовые вершины p^2, p^9 получили соседние троичные коды (– 1 0 0), (– 1 1 0) соответственно. Теперь можно по графу BDD частичной векторной функции получить ее матричное задание (табл. 17), рассматривая пути из корневой вершины в листовые вершины каждой из компонентных функций c_1, c_2, c_3, c_4. Такой переход от BDD-представления к матричному представлению частичной функции подробно изложен в работе [4, с. 60].

Логическая оптимизация и полученное в ее результате доопределение частичной векторной функции приводят к доопределению кодов (троичных векторов) до булевых векторов (табл. 18). В табл. 18 в правой части дано минимизированное BDD-описание, содержащее 6 дизъюнкций, 12 конъюнкций и 27 литералов. Синтезированная логическая схема для кодирования строк матрицы B^2 имеет площадь S_{ASIC} = 4073 (условных единиц) и τ = 1,25 нс, что меньше, чем лучшие схемы 13, 14, предназначенные для кодирования строк матрицы B, содержащей 16 строк. Это и следовало ожидать, так как сложность схемы для перекодирования 9 векторов матрицы B^2 и должна быть меньше, чем сложность схемы для перекодирования 16 векторов, содержащихся в матрице B.

7. Методика логического проектирования кодовых преобразователей

Предложенная методика синтеза кодовых преобразователей включает:

- выбор вида описания, предназначенного для описания функций проектируемого преобразователя кодов; были рассмотрены два вида — ДНФ и BDD;
- выбор способа доопределения исходной системы частичных функций на множестве наборов

Таблица 18

Результаты логической оптимизации в способе 4М (пример 2)

Кодирование троичными векторами		Доопределение троичных векторов в результате оптимизации		BDD-представление
$x_1x_2x_3x_4x_5$	$c_1c_2c_3c_4$	$x_1x_2x_3x_4x_5$	$c_1c_2c_3c_4$	

0 0 0 0 0
0 0 1 0 1
0 1 0 1 0
0 1 1 0 0
1 0 0 0 1
1 0 1 1 0
1 1 0 0 0
0 0 0 0 1
0 0 1 1 0
	– 0 0 0
	– 1 0 0
	– 0 1 0
	– 0 1 1
	0 0 0 1
	1 0 0 1
	– 1 1 1
	– 1 0 1
	– 1 1 0
		0 0 0 0 0
		0 0 1 0 1
		0 1 0 1 0
		0 1 1 0 0
		1 0 0 0 1
		1 0 1 1 0
		1 1 0 0 0
		0 0 0 0 1
		0 0 1 1 0
			0 0 0 0
			0 1 0 0
			0 0 1 0
			0 0 1 1
			0 0 0 1
			1 0 0 1
			0 1 1 1
			0 1 0 1
			0 1 1 0
				c1=x1*s3;
				c2=^x1*s4+x1*s1;
				c3=^x1*s3+x1*s1;
				c4=^x1*s2+x1;
				s1=^x3*x2;
				s2=^x3*s4+x3*x2;
				s3=^x2*x4+x2;
				s4=^x2*s5;
				s5=^x4*x5+x4;

$V^x \backslash B$; для каждого вида минимизируемых функциональных описаний были предложены свои способы доопределения;

- способы (эвристики) кодирования наборов матрицы B;
- методы (программы) оптимизации полученной системы функций — раздельная минимизация в классе ДНФ, совместная минимизация в классе BDDI, совместная минимизация в классе BDD частичных функций.

Для различных исходных булевых матриц B и различных библиотек проектирования могут быть эффективными различные способы решения задач 2—4, однако практика проектирования показывает, что во многих случаях минимизация BDD и BDDI-представлений приводит к более простым комбинационным схемам, синтезируемым из библиотечных элементов [4, 8].

Для "обратных" кодовых преобразователей кодирование строк матрицы C уже имеется (это строки матрицы B), поэтому логическая оптимизация сводится к выбору такого доопределения частичной векторной функции, заданной парой матриц (C, B), для которого сложность функционального описания и, соответственно, логической схемы обратного преобразователя была бы по возможности меньшей.

Получение более эффективных схемных реализаций кодовых преобразователей рассматриваемого класса может включать комбинаторный перебор при решении задачи 2 доопределения BDD (перебор различных доопределений) и перебор кодирований строк матрицы B. Для решения задачи 3 могут быть использованы и другие методы оптимизации представлений систем частичных и полностью определенных функций, например, методы (и программы) совместной минимизации функций в классе ДНФ, минимизации BBDD-представлений (BBDD — *Biconditional BDD*) систем функций [16], а также методы дополнительной оптимизации полученных BDD-представлений, выполняемых с помощью замены формул разложения Шеннона более простыми формулами дизъюнктивного либо конъюнктивного разложений [17]. Аппаратная реализация может выполняться как для заказных СБИС, так и для FPGA.

Заключение

Предложена методика проектирования кодовых преобразователей, использующая различные способы составления функциональных описаний кодовых преобразователей в виде систем частичных булевых функций и различные методы логической минимизации таких систем функций. Предложенная методика проектирования кодовых преобразователей рассмотренного в статье класса может быть использована и при проектировании кодовых преобразователей других классов, если описания их функций сводятся к системам частичных булевых функций.

Список литературы

1. **Применение** интегральных микросхем в электронной вычислительной технике: Справочник / Под ред. Б. Н. Файзулаева, Б. В. Тарабрина. М.: Радио и связь, 1987. 384 с.
2. **Червяков Н. И., Сахнюк П. А., Шапошников А. В., Рядов С. А.** Модулярные параллельные вычислительные структуры нейропроцессорных систем. М.: Физматлит, 2003. 288 с.
3. **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.
4. **Бибило П. Н.** Применение диаграмм двоичного выбора при синтезе логических схем. Минск: Беларус. навука, 2014. 231 с.
5. **Бибило П. Н.** Системы проектирования интегральных схем на основе языка VHDL. StateCAD, ModelSim, LeonardoSpectrum. М.: СОЛОН-Пресс, 2005. 384 с.
6. **Авдеев Н. А., Бибило П. Н.** Эффективность логической оптимизации при синтезе комбинационных схем из библиотечных элементов // Микроэлектроника. 2015. Т. 44, № 5. С. 383—399. DOI: 10.7868/S0544126915050026.
7. **Бибило П. Н., Романов В. И.** Система логической оптимизации функционально-структурных описаний цифровых устройств на основе продукционно-фреймовой модели представления знаний // Проблемы разработки перспективных микро- и наноэлектронных систем. Сб. трудов / под общ. ред. акад. РАН А. Л. Стемпковского. М.: ИППМ РАН, 2020. № 4. С. 9—16.
8. **Бибило П. Н., Ланкевич Ю. Ю.** Использование полиномов Жегалкина при минимизации многоуровневых представлений систем булевых функций на основе разложения Шеннона // Программная инженерия. 2017. Том 8, № 8. С. 369—384. DOI: 10.17587/prin.8.369-384.
9. **Брейтон Р. К., Хэчтел Г. Д., Санджованни-Винчентелли А. Л.** Синтез многоуровневых комбинационных логических схем // ТИИЭР. 1990. Т. 78, № 2. С. 38—83.
10. **Mishchenko A.** An Introduction to Zero-Suppressed Binary Decision Diagrams. Berkeley Verification and Synthesis Research Center, Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, 2014. 15 p.
11. **Kam T., Villa T., Brayton R. K., Sangiovanni-Vincentelli A. L.** Multi-Valued Decision Diagrams for Logic Synthesis and Verification. Memorandum No. UCB/ERL M96/75, 1996. 39 p. URL: http://www2.eecs.berkeley.edu/Pubs/TechRpts/1996/ERL-96-75.pdf
12. **Ashenden P. J., Lewis J.** VHDL-2008. Just the New Stuff. Burlington, MA, USA. Morgan Kaufman Publishers, 2008. 909 p.
13. **Бибило П. Н., Авдеев Н. А., Кардаш С. Н.** и др. Система логического проектирования функциональных блоков заказных КМОП СБИС с пониженным энергопотреблением //Микроэлектроника. 2018. Т. 47, № 1. С. 72—88. DOI: 10.7868/S0544126918010076.
14. **Соловьев В. В.** Архитектуры ПЛИС фирмы Xilinx: FPGA и CPLD 7-й серии. М.: Горячая линия—Телеком, 2016. 392 с.
15. **Тарасов И. Е.** ПЛИС Xilinx. Языки описания аппаратуры VHDL и Verilog, САПР, приемы проектирования. М.: Горячая линия—Телеком, 2020. 538 с.
16. **Amaru L. G.** New Data Structures and Algorithms for Logic Synthesis and Verication. Springer, 2017. 156 p.
17. **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. P. 866—876. DOI: 10.1109/TCAD.2002.1013899.

Hardware Implementation of Code Converters Designed to Reduce the Length of Binary Encoded Words

P. N. Bibilo, bibilo@newman.bas-net.by, The United Institute of Informatics Problems of the National Academy of Sciences of Belarus, Minsk, 220012, Belarus

Corresponding author:

Petr N. Bibilo, Head of Laboratory, The United Institute of Informatics Problems of the National Academy of Sciences of Belarus, 220012, Minsk, Belarus
E-mail: petr.olibib@yandex.ru, bibilo@newman.bas-net.by

Received on June 09, 2022
Accepted on June 27, 2022

The problem of synthesis of combinational circuits of code converters designed to reduce the length of words from a given set of encoded binary words is considered. The encoding assumes that different binary words will be encoded by different binary codes of shorter length. Code converters of this type are designed to reduce the length of binary words transmitted in digital systems over data buses when the bit depth of the transmitted words exceeds the bit depth of the data bus. For example, 18-bit or 17-bit words need to be transmitted over a 16-bit data bus. Each such word can be transmitted in two cycles of operation of a digital system, however, this approach reduces the overall performance of the system. One of the approaches to solve such problems is the development of combinational circuits that convert long binary encoded words into shorter ones.

The proposed methods for solving the problem of synthesizing circuits of code converters are based on the compilation and logical minimization of such forms of systems of incompletely specified Boolean functions as disjunctive normal forms (DNF) and binary decision diagrams (BDD). Using BDD to minimize representations of k-valued functions that depend on Boolean variables is also proposed. Technologically independent logical minimization of functional descriptions of the designed code converters is proposed to be performed by programs for minimizing systems of Boolean functions in the DNF class and programs for joint minimization of BDD representations of systems of completely specified Boolean functions. Minimization of functional descriptions is aimed at reducing the hardware complexity of combinational circuits in the basis of library elements or programmable FPGA elements implementing code converters of the class in question.

Keywords: *code converter, system of Boolean functions, Disjunctive Normal Form, Binary Decision Diagram, Shannon expansion, digital logic synthesis, VHDL, VLSI*

For citation:

Bibilo P. N. Hardware Implementation of Code Converters Designed to Reduce the Length of Binary Encoded Words, *Programmnaya Ingeneria,* 2022, vol. 13, no. 8, pp. 363—382.

DOI: 10.17587/prin.13.363-382

References

1. ***Application** of integrated circuits in electronic computing*: Reference / Edited by B. N. Fayzulaev, B. V. Tarabrin. Moscow, Radio and Communications. 1987, 384 p. (in Russian).

2. **Chervyakov N. I., Sahnyuk P. A., SHaposhnikov A. V., Ryadnov S. A.** *Modular Parallel Computing Structures of Neuroprocessor Systems*, Moscow, Fizmatlit, 2003, 288 p. (in Russian).

3. **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.

4. **Bibilo P. N.** *Application of Binary Decision Diagrams in the Synthesis of Logic Circuits,* Minsk, Belaruskaja navuka, 2014, 231 p. (in Russian).

5. **Bibilo P. N.** *Integrated Circuit Design Systems Based on the VHDL Language. StateCAD, ModelSim, LeonardoSpectrum,* Moscow, SOLON-Press, 2005, 384 p. (in Russian).

6. **Avdeev N. A., Bibilo P. N.** Logical optimization efficiency in the synthesis of combinational circuits, *Mikroelektronika,* 2015, vol. 44, no. 5, pp. 383—399. DOI: 10.7868/S0544126915050026 (in Russian).

7. **Bibilo P. N., Romanov V. I.** The system of logical optimization of functional structural descriptions of digital circuits based on production-frame knowledge representation model, *Problemy razrabotki perspektivnyh mikro- i nanoelektronnyh sistem*, 2020. Sb. trudov / pod obshch. red. akad. RAN A. L. Stempkovskogo. Moscow, IPPM RAN, 2020, no. 4, pp. 9—16.

8. **Bibilo P. N., Lankevich Yu. Yu.** The use of Zhegalkin polynomials in minimizing multilevel representations of systems of Boolean functions based on the Shannon expansion, *Programmnaya ingeneria*, 2017, vol. 8, no. 8, pp. 369—384. DOI: 10.17587/prin.8.369—384 (in Russian).

9. **Brayton R. K., Hachtel G. D., Sangiovanni-Vincentelli A. L.** Synthesis of multi-level combinational logic circuits, *Trudy Institute inzhenerov po jelektronike i radiotehnike*, 1990, vol. 78, no. 2, pp. 38—83 (in Russian).

10. **Mishchenko A.** An Introduction to Zero-Suppressed Binary Decision Diagrams. Berkeley Verification and Synthesis Research Center, Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, 2014, 15 p.

11. **Kam T., Villa T., Brayton R. K., Sangiovanni-Vincentelli A. L.** Multi-Valued Decision Diagrams for Logic Synthesis and Verification. Memorandum No. UCB/ERL M96/75, 1996. 39 p., available at: http://www2.eecs.berkeley.edu/Pubs/TechRpts/1996/ERL-96-75.pdf

12. **Ashenden P. J., Lewis J.** *VHDL-2008. Just the New Stuff.* Burlington, MA, USA. Morgan Kaufman Publishers, 2008, 909 p.

13. **Bibilo P. N., Avdeev N. A., Kardash S. N.** et al. System of Logical Design of Functional Blocks of Custom CMOS VLSI with Reduced Power Consumption, *Mikroelektronika*, 2018, vol. 7, no. 1, pp. 72—88. DOI: 10.7868/S0544126918010076 (in Russian).

14. **Solovyov V. V.** *XILINX FPGA Architectures: FPGA and CPLD 7-Series*, Moscow, Goryachaya liniya Telekom, 2016, 392 p. (in Russian).

15. **Tarasov I. E.** *XILINX FPGA. Hardware Description Languages VHDL and Verilog, CAD, Design Techniques*, Moscow, Goryachaya liniya Telekom, 2020, 538 p. (in Russian).

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

17. **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. DOI: 10.1109/TCAD.2002.1013899.