Министерство образования Республики Беларусь

Учреждение образования
«Белорусский государственный университет
информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра электронных вычислительных машин

Дисциплина: Арифметические и логические основы
вычислительной техники

К ЗАЩИТЕ ДОПУСТИТЬ

___________ Ю. А. Луцик

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
к курсовой работе
на тему

ПРОЕКТИРОВАНИЕ И ЛОГИЧЕСКИЙ СИНТЕЗ СУММАТОРА-УМНОЖИТЕЛЯ ДВОИЧНО-ЧЕТВЕРИЧНЫХ ЧИСЕЛ

БГУИР КР 1-40 02 01 512 ПЗ

Студент Н. А. Кивачук

Руководитель И. В. Лукьянова

МИНСК 2022

Министерство образования Республики Беларусь

Учреждение образования
«Белорусский государственный университет
информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра электронных вычислительных машин

Дисциплина: Арифметические и логические основы
вычислительной техники

УТВЕРЖДАЮ
Заведующий кафедрой ЭВМ
_________ Б. В. Никульшин
«____» _________2022г.

*

ЗАДАНИЕ
по курсовой работе студента
Кивачука Никиты Андреевича

1 Тема работы: «Проектирование и логический синтез сумматора-умножителя двоично-четверичных чисел»
2 Срок сдачи студентом законченной работы: до 20 мая;
3 Исходные данные к работе:
3.1Исходные сомножители: Мн=78,69; Мт=11,59;
3.2 Алгоритм умножения: А;
3.3 Метод умножения: умножение закодированного двоично-четверичного множимого на два разряда двоичного множителя одновременно в дополнительных кодах;
3.4 Коды четверичных цифр множимого для перехода к двоично-четверичной системе кодирования: 0_4 – 01, 1_4 – 11, 2_4 – 10, 3_4 – 00;
3.5 Тип синтезируемого умножителя: 2;
3.6 Логический базис для синтеза ОЧС: ИЛИ-НЕ; метод минимизации – карты Карно-Вейча;
3.7 Логический базис для синтеза ОЧУС: ИЛИ, СУММА ПО МОДУЛЮ 2, КОНСТАНТА 1; метод минимизации – алгоритм Рота;

4 Содержание пояснительной записки (перечень подлежащих разработке вопросов):

Введение. 1. Разработка алгоритма умножения. 2. Разработка структурной схемы сумматора-умножителя. 3. Разработка функциональных схем основных узлов сумматора-умножителя. 4. Синтез комбинационных схем устройств на основе мультиплексоров. 5. Оценка результатов разработки. Заключение. Список литературы.

5 Перечень графического материала:

5.1 Сумматор-умножитель второго типа. Схема электрическая структурная.

5.2 Одноразрядный четверичный сумматор. Схема электрическая функциональная.

5.3 Одноразрядный четверичный умножитель-сумматор. Схема электрическая функциональная.

5.4 Преобразователь множителя. Реализация на мультиплексорах. Схема электрическая функциональная.

5.5 Одноразрядный четверичный сумматор. Реализация на мультиплексорах. Схема электрическая функциональная.

КАЛЕНДАРНЫЙ ПЛАН

Наименование этапов курсовой работы	Объем этапа,%	Срок выполнения этапа	Примечания
Разработка алгоритма умножения	10	10.02-20.02	
Разработка структурной схемы сумматора-умножителя	10	21.02-09.03	С выполнением чертежа
Разработка функциональных схем основных узлов	50	10.03-30.04	С выполнением чертежа
Синтез комбинационных схем устройств на основе мультиплексоров	10	1.05-15.05	С выполнением чертежа
Завершение оформления пояснительной записки	20	15.05-20.05	

Дата выдачи задания: 10 февраля 2022 г.

Руководитель И. В. Лукьянова

ЗАДАНИЕ ПРИНЯЛ К ИСПОЛНЕНИЮ _________

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Данная курсовая работа посвящена разработке алгоритмов выполнения операций умножения и сложения. На основе полученных алгоритмов требуется разработать и синтезировать следующие устройства: одноразрядный четвертичный сумматор (ОЧС), одноразрядный четвертичный умножитель-сумматор (ОЧУС), а также переключательные функции ОЧС на мультиплексорах. Минимизация перечисленных устройств осуществляется с помощью карт Карно-Вейча и алгоритма извлечения Рота. На основе полученных данных требуется построить схемы этих устройств и проанализировать результаты (эффективность минимизации и время выполнения операций).

1 Разработка алгоритма умножения

Исходные данные:
- исходные сомножители: Мн = 78,69; Мт = 11,59;
- алгоритм умножения: А;
- метод умножения: умножение закодированного двоично-четверичного множимого на два разряда двоичного множителя одновременно в прямых кодах;
- кодирование четверичных цифр множимого для перехода к двоично-четверичной системе кодирования: 0_4 – 01, 1_4 – 11, 2_4 – 10, 3_4 – 00;
- тип синтезируемого умножителя: 2-й;

Перевод сомножителей из десятичной системы счисления в четверичную.

Множимое

_78|_4	0.69	Мн_4 = 1032,23
4 | 19|_4	4	в соответствии с заданной кодировкой
38 16| 4|4	2.76	множимого
36 3 4|1	4	$\text{Мн}_{2/4}$ = 11010010, 1000
2 0	3.40	

Множитель

_78|_4	0.69	Мн_4 = 1032,23
4 | 19|_4	4	в соответствии с заданной кодировкой
38 16| 4|4	2.76	множимого
36 3 4|1	4	$\text{Мн}_{2/4}$ = 11010010, 1000
2 0	3.40	

Запишем сомножители в форме с плавающей запятой в прямом коде:
Мн = 0,110100101000 $P_{\text{Мн}}$ = 0.1101 $+04_{10}$ — закодировано по заданию
Мт = 0,101110010111 $P_{\text{Мт}}$ = 0.0010 $+02_{10}$ — закодировано традиционно

Порядок произведения будет следующим:
$P_{\text{Мн}}$ = 0.1101 10_4
$P_{\text{Мт}}$ = 0.0010 02_4
$P_{\text{Мн}\cdot\text{Мт}}$ = 0.1110 12_4

Результат закодирован в соответствии с заданием на кодировку множимого.

Знак произведения определяется суммой по модулю два знаков сомножителей, т. е.:

зн Мн $\oplus$ зн Мт = 0 $\oplus$ 0 = 0.

Преобразование множителя и перемножение мантисс.

Мт$^{п}_{4}$ = *11´1´2121´*
Мт$^{п}_{2}$ = 10´1´0´1´1001100´1´

Таблица 1.4.1 – Перемножение мантисс

Четверичная с/с			**Двоично-четверичная с/с**			**Комментарии**
1			**2**			**3**
0 3	0000000 2301110		0 1	01010101010101 10000111111101		$\sum_0$ $П_1$= -Мн
3 3 0	2301110 3230111 2131120		1 1 0	10000111111101 00100001111111 10110011111001		$\sum_1$ $\sum_1 * 4^{-1}$ $П_2$=2Мн
0 0 0	2021231 0202123 1032230	 1 	0 0 0	10011011100011 01100110111000 11010010100001	 11 	$\sum_2$ $\sum_2 * 4^{-1}$ $П_3$ = Мн
0 0 0	1301013 0130101 2131120	1 31 	0 0 0	11000111011100 01110001110111 10110001011001	11 0011 	$\sum_3$ $\sum_3 * 4^{-1}$ $П_4$ = 2Мн
0 0 3	2321221 0232122 2301110	31 131 	0 0 1	10001011101011 01100010111010 10000111111101	0011 110011 	$\sum_4$ $\sum_4 * 4^{-1}$ $П_5$ = -Мн
3 3 3	3133232 3313323 2301110	131 2131 	1 1 1	00010000100010 00000100001000 10000111111101	110011 10110011 	$\sum_5$ $\sum_5 * 4^{-1}$ $П_6$ = -Мн
3 0 0	2221033 3222103 1032230	2131 32131 	1 0 0	10101011010000 00101010110100 11010010100001	10110011 0010110011 	$\sum_6$ $\sum_6 * 4^{-1}$ $П_7$ = Мн
0 0	0320333 0032033	32131 332131	0 0	01001001000000 01010010010000	0010110011 000010110011	$\sum_7$ $\sum_7 * 4^{-1}$

После окончания умножения необходимо оценить погрешность вычислений. Для этого полученное произведение приводится к нулевому порядку, а затем переводится в десятичную систему счисления:

$(Мн \cdot Мт)_4 = 0{,}0032033332131$
$(Мн \cdot Мт)_{10} = 911{,}9758$

Результат прямого перемножения операндов даёт следующее значение:
$Мн_{10} * Мт_{10} = 912{,}0171$

Абсолютная погрешность:
$\Delta = 912{,}0171 - 911{,}9758 = 0{,}0413$.

Относительная погрешность:

$\delta = \frac{\Delta}{Мн \cdot Мт} = \frac{0{,}0413}{912{,}0171} = 0{,}000045$ ($\delta = 0{,}0045\%$)

Эта погрешность получена за счёт приближённого перевода из десятичной системы счисления в четверичную обоих сомножителей, а также за счёт округления полученного результата произведения.

2. РАЗРАБОТКА СТРУКТУРНОЙ СХЕМЫ СУММАТОРА-УМНОЖИТЕЛЯ

Структурная схема сумматора-умножителя второго типа для алгоритма умножения «А» представлена на в приложении А.

Структурная схема второго типа строится на базе заданных узлов ОЧУС, ОЧС, формирователя дополнительного кода и регистра результата. Управление режимами работы схемы осуществляется внешним сигналом *Mul/sum,* который определяет вид текущей арифметической операции (умножение или суммирование).

Если устройство работает как сумматор, то оба слагаемых последовательно (за два такта) заносятся в регистр множимого, а на управляющий вход формирователя дополнительного кода F2 поступает «1».

Если устройство работает как умножитель, то множимое и множитель помещаются в соответствующие регистры, а на управляющий вход ФДК F2 поступает «0».

Таблица 2.1 - Режимы работы формирователя дополнительного кода

Сигналы на входах ФДК		Результат на выходах ФДК
F_1	F_2	
0	0	Дополнительный код множимого
0	1	Дополнительный код слагаемого
1	0	Меняется знак Мн
1	1	Меняется знак слагаемого

3. РАЗРАБОТКА ФУНКЦИОНАЛЬНЫХ СХЕМ ОСНОВНЫХ УЗЛОВ СУММАТОРА-УМНОЖИТЕЛЯ

3.1. Логический синтез одноразрядного четверичного сумматора

Одноразрядный четверичный сумматор – это комбинационное устройство, имеющее 5 двоичных входов (2 разряда одного слагаемого, 2 разряда второго слагаемого и вход переноса) и 3 двоичных выхода.

Функциональная схема реализации ОЧС приведена в приложении Б.

Принцип работы ОЧС представлен с помощью таблицы истинности (таблица 3.1.1).

Разряды обоих слагаемых закодированы: 0 – 01; 1 – 11; 2 – 10; 3 – 00.

Таблица 3.1.1 – Таблица истинности ОЧС

a_1	a_2	b_1	b_2	p	П	S_1	S_2	Пример операции в четверичной с/с
1	2	3	4	5	6	7	8	9
0	0	0	0	0	1	1	0	3+3+0=12
0	0	0	0	1	1	0	0	3+3+1=13
0	0	0	1	0	0	0	0	3+0+0=03
0	0	0	1	1	1	0	1	3+0+1=10
0	0	1	0	0	1	1	1	3+2+0=11
0	0	1	0	1	1	1	0	3+2+1=12
0	0	1	1	0	1	0	1	3+1+0=10
0	0	1	1	1	1	1	1	3+1+1=11
0	1	0	0	0	0	0	0	0+3+0=03
0	1	0	0	1	1	0	1	0+3+1=10
0	1	0	1	0	0	0	1	0+0+0=00
0	1	0	1	1	0	1	1	0+0+1=01
0	1	1	0	0	0	1	0	0+2+0=02
0	1	1	0	1	0	0	0	0+2+1=03
0	1	1	1	0	0	1	1	0+1+0=01
0	1	1	1	1	0	1	0	0+1+1=02

1	0	0	0	0	1	1	1	2+3+0=11
1	0	0	0	1	1	1	0	2+3+1=12
1	0	0	1	0	0	1	0	2+0+0=02
1	0	0	1	1	0	1	1	2+0+1=03
1	0	1	0	0	1	0	1	2+2+0=10
1	0	1	0	1	1	1	1	2+2+1=11
1	0	1	1	0	0	0	1	2+1+0=03
1	0	1	1	1	1	0	0	2+1+1=10
1	1	0	0	0	1	0	1	1+3+0=10
1	1	0	0	1	1	1	1	1+3+1=11
1	1	0	1	0	0	1	1	1+0+0=01
1	1	0	1	1	0	1	0	1+0+1=02
1	1	1	0	0	0	0	0	1+2+0=03
1	1	1	0	1	1	0	1	1+2+1=10
1	1	1	1	0	0	1	0	1+1+0=02
1	1	1	1	1	0	0	1	1+1+1=03

Минимизацию функции ***S₁*** проведем картой Вейча:

a_2

a_1		1				1	1	1	
a_1	1	1		1			1	1	b_2
		1	1	1		1			b_2
				1	1	1		1	

b_1

p p

$f_{min\,(\text{ДНФ})} = \overline{\overline{a_1 + a_1} + a_2 + b_1} + \overline{\overline{a_1 + a_1} + b_1 + b_2} + \overline{a_2 + b_1 + b_2 + P} +$
$\overline{a_2 + \overline{b_1 + b_1} + b_2 + \overline{p + p}} + \overline{a_1 + \overline{b_1 + b_1} + \overline{b_2 + b_2} + \overline{p + p}} + \overline{a_1 + \overline{b_1 + b_1} + b_2 + \overline{p + p}} +$
$\overline{\overline{a_2 + a_2} + \overline{b_1 + b_1} + \overline{b_2 + b_2} + \overline{p + p}} + \overline{\overline{a_1 + a_1} + \overline{a_2 + a_2} + b_1 + \overline{p + p}} +$
$\overline{\overline{a_2 + a_2} + b_1 + \overline{b_2 + b_2} + \overline{p + p}}$

Минимизацию функции P проведем алгоритмом Рота:
Определим множество единичных кубов:

L = {00000,00001,00011,00100,00101,00110,00111,01001,10000,10001,10100, 10101, 10111, 11000, 11001, 11101};
И множество безразличных кубов:
N = {∅};
Минимизацию безразличных кубов проводить не требуется.
Сформируем множество $C_0 = L \cup N$:
C_0=L={00000,00001,00011,00100,00101,00110,00111,01001,10000,10001,10100 , 10101, 10111, 11000, 11001, 11101};

Первым этапом алгоритма Рота является нахождение множества простых импликант.

Для реализации этого этапа будем использовать операцию умножения (*) над множествами C_0, C_1 и т. д., пока в результате операции будут образовываться новые кубы большей размерности.

Первый шаг умножения (C_0*C_0) приведён в таблице 3.1.2.

Таблица 3.1.2 – Поиск простых импликант(C_0*C_0)

C0*C0	00000	00001	00011	00100	00101	00110	00111	01001	10000	10001	10100	10101	10111	11000	11001	11101
00000	-															
00001	0000y	-														
00011	000yy	000y1	-													
00100	00y00	00y0y	00yyy	-												
00101	00y0y	00y01	00yy1	0010y	-											
00110	00yy0	00yyy	00y1y	001y0	001yy	-										
00111	00yyy	00yy1	00y11	001yy	001y1	0011y	-									
01001	0y00y	0y001	0y0y1	0yy0y	0yy01	0yyyy	0yyy1	-								
10000	y0000	y000y	y00yy	y0y00	y0y0y	y0yy0	y0yyy	yy00y	-							
10001	y000y	y0001	y00y1	y0y0y	y0y01	y0yyy	y0yy1	yy001	1000y	-						
10100	y0y00	y0y0y	y0yyy	y0100	y010y	y01y0	y01yy	yyy0y	10y00	10y0y	-					
10101	y0y0y	y0y01	y0yy1	y010y	y0101	y01yy	y01y1	yyy01	10y0y	10y01	1010y	-				
10111	y0yyy	y0yy1	y0y11	y01yy	y01y1	y011y	y0111	yyyy1	10yyy	10yy1	101yy	101y1	-			
11000	yy000	yy00y	yy0yy	yyy00	yyy0y	yyyy0	yyyyy	y100y	1y000	1y00y	1yy00	1yy0y	1yyyy	-		
11001	yy00y	yy001	yy0y1	yyy0y	yyy01	yyyyy	yyyy1	y1001	1y00y	1y001	1yy0y	1yy01	1yyy1	1100y	-	
11101	yyy0y	yyy01	yyyy1	yy10y	yy101	yy1yy	yy1y1	y1y01	1yy0y	1yy01	1y10y	1y101	1y1y1	11y0y	11y01	-
A1	0000x 00x00 x0000	000x1 00x01 0x001 x0001	00x11	0010x 001x0 x0100	001x1 x0101	0011x	x0111	x1001	1000x 10x00 1x000	10x01 1x001	1010x	101x1 1x101	Ø	1100x	11x01	Ø

В результате сформируем новое множество кубов:

A1 = { 0000x; 00x00; x0000; 000x1; 00x01; 0x001; x0001; 00x11; 0010x; 001x0; x0100; 001x1; x0101; 0011x; x0111; x1001; 1000x; 10x00; 1x000; 10x01; 1x001; 1010x; 101x1; 1x101; 1100x; 11x01 }

Множество Z_0 кубов, не участвовавших в образовании новых кубов, пустое. Также формируется множество $B_1 = C_0 — Z_0$

Для следующего шага получения множества Z формируется множество $C_1 = A_1 \cup B_1$.

C1 = { 0000x; 00x00; x0000; 000x1; 00x01; 0x001; x0001; 00x11; 0010x; 001x0; x0100; 001x1; x0101; 0011x; x0111; x1001; 1000x; 10x00; 1x000; 10x01; 1x001; 1010x; 101x1; 1x101; 1100x; 11x01 }

В таблице 3.1.3 приведён следующий шаг поиска простых импликант с помощью операции $C_1 * C_1$.

Таблица 3.1.3– Поиск простых импликант(C_1*C_1)

C1*C1	0000x	00x00	x0000	000x1	00x01	0x001	x0001	00x11	0010x	001x0	x0100	001x1	x0101	0011x	x0111	x1001	1000x	10x00	1x000	10x01	1x001	1010x	101x1	1x101	1100x	11x01
0000x	-																									
00x00	00000	-																								
x0000	00000	00000	-																							
000x1	00001	0000y	0000y	-																						
00x01	00001	00x0y	0000y	00001	-																					
0x001	00001	0000y	0000y	00001	00001	-																				
x0001	00001	0000y	x000y	00001	00001	00001	-																			
00x11	000y1	00xyy	000yy	00011	00xy1	000y1	000y1	-																		
0010x	00y0x	00100	00y00	00y01	00101	00y01	00y01	001y1	-																	
001x0	00y00	00100	00y00	00yxy	0010y	00y0y	00y0y	0011y	00100	-																
x0100	00y00	00100	x0y00	00y0y	0010y	00y0y	x0y0y	001yy	00100	00100	-															
001x1	00y01	0010y	00y0y	00yx1	00101	00y01	00y01	00111	00101	001xy	0010y	-														
x0101	00y01	0010y	x0y0y	00y01	00101	00y01	x0y01	001y1	00101	0010y	x010y	00101	-													
0011x	00yyx	001y0	00yy0	00y11	001y1	00yy1	00yy1	00111	001yx	00110	001y0	00111	001y1	-												
x0111	00yy1	001yy	x0yyy	00y11	001y1	00yy1	x0yy1	00111	001y1	0011y	x01yy	00111	x01y1	00111	-											
x1001	0y001	0y00y	xy00y	0y001	0y001	01001	xy001	0y0y1	0yy01	0yy0y	xyy0y	0yy01	xyy01	0yyy1	xyyy1	-										
1000x	y000x	y0000	10000	y0001	y0001	y0001	10001	y00y1	y0y0x	y0y00	10y00	y0y01	10y01	y0yyx	10yy1	1y001	-									
10x00	y0000	y0x00	10000	y000y	y0x0y	y000y	1000y	y0xyy	y0100	y0100	10100	y010y	1010y	y01y0	101yy	1y00y	10000	-								
1x000	y0000	y0000	10000	y000y	y000y	yx00y	1000y	y00yy	y0y00	y0y00	10y00	y0y0y	10y0y	y0yy0	10yyy	1100y	10000	10000	-							
10x01	y0001	y0x0y	1000y	y0001	y0x01	y0001	10001	y0xy1	y0101	y010y	1010y	y0101	10101	y01y1	101y1	1y001	10001	10x0y	1000y	-						
1x001	y0001	y000y	1000y	y0001	y0001	yx001	10001	y00y1	y0y01	y0y0y	10y0y	y0y01	10y01	y0yy1	10yy1	11001	10001	1000y	1x00y	10001	-					
1010x	y0y0x	y0100	10y00	y0y01	y0101	y0y01	10y01	y01y1	y010x	y0100	10100	y0101	10101	y01yx	101y1	1yy01	10y0x	10100	10y00	10101	10y01	-				
101x1	y0y01	y010y	10y0y	y0yx1	y0101	y0y01	10y01	y0111	y0101	y01xy	1010y	y01x1	10101	y0111	10111	1yy01	10y01	1010y	10y0y	10101	10y01	10101	-			
1x101	y0y01	y010y	10y0y	y0y01	y0101	yxy01	10y01	y01y1	y0101	y010y	1010y	y0101	10101	y01y1	101y1	11y01	10y01	1010y	1xy0y	10101	1xy01	10101	10101	-		
1100x	yy00x	yy000	1y000	yy001	yy001	y1001	1y001	yy0y1	yyy0x	yyy00	1yy00	yyy01	1yy01	yyyyx	1yyy1	11001	1y00x	1y000	11000	1y001	11001	1yy0x	1yy01	11y01	-	
11x01	yy001	yyx0y	1y00y	yy001	yyx01	y1001	1y001	yyxy1	yy101	yy10y	1y10y	yy101	1y101	yy1y1	1y1y1	11001	1y001	1yx0y	1100y	1yx01	11001	1y101	1y101	11101	11001	-
A2	00x0x x000x	00x0x x0x00	x000x x0x00	00xx1	00xx1 x0x01	xx001	x0x01 xx001	Ø	001xx x010x	001xx	x010x	x01x1	x01x1	Ø	Ø	Ø	10x0x 1x00x	10x0x	1x00x	1xx01	1xx01	Ø	Ø	Ø	Ø	Ø

В результате этой операции образуется множество кубов:

A2 = { 00x0x; x000x; x0x00; 00xx1; x0x01; xx001; 001xx; x010x; x01x1; 10x0x; 1x00x; 1xx01 }

Множество Z_1 кубов, не участвовавших в образовании новых кубов, пустое. Также формируется множество B_2 = C_1 — Z_1.

Для следующего шага получения множества Z формируется множество $C_2 = A_2 \cup B_2$.

C2 = { 00x0x; x000x; x0x00; 00xx1; x0x01; xx001; 001xx; x010x; x01x1; 10x0x; 1x00x; 1xx01 }

В таблице 3.1.4 приведён следующий шаг поиска простых импликант – операция C_2*C_2.

Таблица 3.1.4 – Поиск простых импликант (C_2*C_2)

C2*C2	00x0x	x000x	x0x00	00xx1	x0x01	xx001	001xx	x010x	x01x1	10x0x	1x00x	1xx01
00x0x	-											
x000x	0000x	-										
x0x00	00x00	x0000	-									
00xx1	00x01	00001	00x0y	-								
x0x01	00x01	x0001	x0x0y	00x01	-							
xx001	00001	x0001	x000y	00001	x0001	-						
001xx	0010x	00y0x	00100	001x1	00101	00y01	-					
x010x	0010x	x0y0x	x0100	00101	x0101	x0y01	0010x	-				
x01x1	00101	x0y01	x010y	001x1	x0101	x0y01	001x1	x0101	-			
10x0x	y0x0x	1000x	10x00	y0x01	10x01	10001	y010x	1010x	10101	-		
1x00x	y000x	1000x	10000	y0001	10001	1x001	y0y0x	10y0x	10y01	1000x	-	
1xx01	y0x01	10001	10x0y	y0x01	10x01	1x001	y0101	10101	10101	10x01	1x001	-
A3	x0x0x	x0x0x	x0x0x	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø

В результате этой операции образуется множество кубов:

A3 = { x0x0x }

Множество Z_2 кубов, не участвовавших в образовании новых кубов имеет вид:

Z2 = { 00xx1; xx001; 001xx; x01x1; 1x00x; 1xx01 }

Также формируется множество $B_3 = C_2 — Z_2$.

Для следующего шага получения множества Z формируется множество $C_3 = A_3 \cup B_3$.

C3 = { x0x0x }

В таблице 3.1.5 приведён следующий шаг поиска простых импликант – операция C_3*C_3.

Таблица 3.1.5

C3*C3	x0x0x
x0x0x	-
A4	Ø

Новых кубов (четвертой размерности) не образовалось.
Поскольку $|C_4| \leq 1$, поиск простых импликант заканчивается. Множество простых импликант:

Z = { 00xx1; xx001; 001xx; x01x1; 1x00x; 1xx01; x0x0x }

Следующий этап – поиск L-экстремалей на множестве простых импликант (таблица 3.1.6). Для этого используется операция # (решётчатое вычитание).

Таблица 3.1.6-Поиск L-экстремалей

z#(Z-z)	00xx1	xx001	001xx	x01x1	1x00x	1xx01	x0x0x
00xx1	-	1x001 x1001	001x0	101x1	1x00x	1xx01	10x0x x0x00
xx001	001x1 00x11	-	001x0	101x1	1x000	1x101	1010x 10x00 x0x00
001xx	00011	1x001 x1001	-	101x1	1x000	1x101	1010x 10x00 10x00 x0000
x01x1	00011	1x001 x1001	001x0	-	1x000	11101	10100 10x00 10x00 x0000
1x00x	00011	01001	001x0	101x1	-	11101	10100 10100 10100 00000
1xx01	00011	01001	001x0	10111	1x000	-	10100 10100 10100 00000
x0x0x	00011	01001	00110	10111	11000	11101	-
Остаток	00011	01001	00110	10111	11000	11101	10100 10100 10100 00000

В таблице 3.1.6 из каждой простой импликанты поочередно вычитаются все остальные простые импликанты Z#(Z\z), результат операции (последняя строка таблицы) указывает на то, что L-экстремалями стали следующие простые импликанты:

$\boldsymbol{E_0}$= { 00xx1; xx001; 001xx; x01x1; 1x00x; 1xx01; x0x0x }

Проверяем в таблице 3.1.7, нет ли среди полученных L-экстремалей таких, которые стали L-экстремалями за счёт безразличных кубов.

Таблица 3.1.7- Проверка на L-экстремальность

z#(Z-z) n L	00000	00001	00011	00100	00101	00110	00111	01001	10000	10001	10100	10101	10111	11000	11001	11101
00011	Ø	Ø	00011	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø
01001	Ø	Ø	Ø	Ø	Ø	Ø	Ø	01001	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø
00110	Ø	Ø	Ø	Ø	Ø	00110	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø
10111	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	10111	Ø	Ø	Ø
11000	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	11000	Ø	Ø
11101	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	11101
10100	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	10100	Ø	Ø	Ø	Ø	Ø
10100	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	10100	Ø	Ø	Ø	Ø	Ø
10100	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	10100	Ø	Ø	Ø	Ø	Ø
00000	00000	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø

Z′ = Z - E = { Ø }

Минимальное покрытие - множество L-экстремалей E = { 00xx1; xx001; 001xx; x01x1; 1x00x; 1xx01; x0x0x }

$f_{min\ (\text{ДНФ})} = \overline{a_1}\,\overline{a_2}b_1 + \bar{a}_1\overline{a_2}p + \overline{a_2}\,\overline{b_2} + \overline{a_2}b_1p + a_1\overline{a_2}\,\overline{b_2}p + \overline{b_1}\,\overline{b_2}p + a_1\overline{b_1}\,\overline{b_2}$

Минимизацию S_2 проведем картами Вейча

a_2

a_1									b_2
a_1	1	1	1		1	1		1	
a_1	1		1		1		1		b_2
	1	1		1	1	1			b_2
		1			1				

b_1

p p

$$f_{min\,(ДНФ)} = \overline{\overline{b_1 + b_1} + a_2 + p} + \overline{\overline{a_1 + a_1} + a_2 + b_1 + \overline{b_2 + b_2 + \overline{p + p}}} +$$
$$\overline{\overline{a_1 + a_1} + b_1 + b_2 + P} + \overline{\overline{a_1 + a_1} + a_2 + \overline{b_1 + b_1} + b_2} + \overline{a_1 + a_2 + \overline{b_1 + b_1} + \overline{b_2 + b_2}} +$$
$$\overline{\overline{a_1 + a_1} + \overline{a_2 + a_2} + b_2 + \overline{p + p}} + \overline{\overline{a_1 + a_1} + \overline{a_2 + a_2} + \overline{b_1 + b_1} + \overline{p + p}} +$$
$$\overline{a_1 + \overline{a_2 + a_2} + b_1 + \overline{p + p}} + \overline{\overline{a_2 + a_2} + b_1 + \overline{b_2 + b_2} + p}$$

3.2. Логический синтез одноразрядного четверичного умножителя-сумматора

ОЧУС – это комбинационное устройство, имеющее шесть входов (два разряда из регистра множимого, два разряда из регистра множителя, вход переноса и управляющий вход h) и три выхода.

Функциональная схема реализации ОЧУС приведена в приложении В.

Принцип работы ОЧУС представлен с помощью таблицы истинности(таблица 3.2.1).

Разряды множителя закодированы: 0 – 00, 1 – 01, 2 – 10, 3 – 11.
Разряды множимого закодированы: 0 – 01, 1 – 11, 2 – 10, 3 – 00.

Таблица истинности ОЧУС(3.2.1)

Пер.	Мн		Мт		Упр.	Перенос	Результат		Результат операции в четверичной с/с
***P*1**	***x*1**	***x*2**	***y*1**	***y*2**	***h***	***P***	***Q*1**	***Q*2**	
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
0	0	0	0	0	0	x	x	x	3·3+0=15
0	0	0	0	0	1	x	x	x	Выход – код «03»
0	0	0	0	1	0	x	x	x	3·0+0=00
0	0	0	0	1	1	x	x	x	Выход – код «03»

0	0	0	1	0	0	1	1	0	3·2+0=12
0	0	0	1	0	1	0	0	0	Выход – код «03»
0	0	0	1	1	0	0	0	0	3·1+0=03
0	0	0	1	1	1	0	0	0	Выход – код «03»
0	0	1	0	0	0	x	x	x	0·3+0=00
0	0	1	0	0	1	x	x	x	Выход – код «00»
0	0	1	0	1	0	x	x	x	0·0+0=00
0	0	1	0	1	1	x	x	x	Выход – код «00»
0	0	1	1	0	0	0	0	1	0·2+0=00
0	0	1	1	0	1	0	0	1	Выход – код «00»
0	0	1	1	1	0	x	x	x	0·1+0=00
0	0	1	1	1	1	x	x	x	Выход – код «00»
0	1	0	0	0	0	x	x	x	2·3+0=12
0	1	0	0	0	1	x	x	x	Выход – код «02»
0	1	0	0	1	0	x	x	x	2·0+0=02
0	1	0	0	1	1	x	x	x	Выход – код «02»
0	1	0	1	0	0	1	0	0	2·2+0=10
0	1	0	1	0	1	0	1	0	Выход – код «02»
0	1	0	1	1	0	x	x	x	2·1+0=02
0	1	0	1	1	1	x	x	x	Выход – код «02»
0	1	1	0	0	0	x	x	x	1·3+0=03
0	1	1	0	0	1	x	x	x	Выход – код «01»
0	1	1	0	1	0	0	0	0	1·0+0=00
0	1	1	0	1	1	0	1	1	Выход – код «01»
0	1	1	1	0	0	0	1	0	1·2+0=02
0	1	1	1	0	1	0	1	1	Выход – код «01»
0	1	1	1	1	0	0	1	1	1·1+0=01
0	1	1	1	1	1	0	1	1	Выход – код «01»
1	0	0	0	0	0	x	x	x	3·3+1=22
1	0	0	0	0	1	x	x	x	Выход – код «03»
1	0	0	0	1	0	x	x	x	3·0+1=01
1	0	0	0	1	1	x	x	x	Выход – код «03»
1	0	0	1	0	0	1	0	0	3·2+1=13
1	0	0	1	0	1	x	x	x	Выход – код «03»
1	0	0	1	1	0	x	x	x	3·1+1=10
1	0	0	1	1	1	x	x	x	Выход – код «03»

1	0	1	0	0	0	x	x	x	0·3+1=01
1	0	1	0	0	1	x	x	x	Выход – код «00»
1	0	1	0	1	0	x	x	x	0·0+1=01
1	0	1	0	1	1	x	x	x	Выход – код «00»
1	0	1	1	0	0	1	1	1	0·2+1=01
1	0	1	1	0	1	x	x	x	Выход – код «00»
1	0	1	1	1	0	x	x	x	0·1+0=00
1	0	1	1	1	1	x	x	x	Выход – код «00»
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1	1	0	0	0	0	x	x	x	2·3+1=13
1	1	0	0	0	1	x	x	x	Выход – код «02»
1	1	0	0	1	0	x	x	x	2·0+1=01
1	1	0	0	1	1	x	x	x	Выход – код «02»
1	1	0	1	0	0	1	1	1	2·2+1=11
1	1	0	1	0	1	x	x	x	Выход – код «02»
1	1	0	1	1	0	x	x	x	2·1+1=03
1	1	0	1	1	1	x	x	x	Выход – код «02»
1	1	1	0	0	0	x	x	x	1·3+1=10
1	1	1	0	0	1	x	x	x	Выход – код «01»
1	1	1	0	1	0	x	x	x	1·0+1=02
1	1	1	0	1	1	x	x	x	Выход – код «01»
1	1	1	1	0	0	0	0	1	1·2+1=03
1	1	1	1	0	1	x	x	x	Выход – код «01»
1	1	1	1	1	0	x	x	x	1·1+1=02
1	1	1	1	1	1	x	x	x	Выход – код «01»

Минимизация выходов ОЧУС:

Минимизация функции P картой Карно:

y_1y_2h / $p_1x_1x_2$	000	001	011	010	110	111	101	100
000	*	*	*	*	*	*	*	*
001	*	*	*	*	*	*	*	*
011	*	*	0	*	*	*	*	*
010	*	*	0	*	*	*	*	*
110	0	*	0	*	*	*	*	*
111	0	*	0	*	*	*	*	*
101	0	0	0	0	*	*	*	*
100	1	0	0	1	1	0	1	1

$f_{min(\text{ДНФ})} = \big((p_1 \oplus 1 + x_1) \oplus 1\big) + \big((x_2 + y_2 + h) \oplus 1\big).$

Минимизация функции Q1 картой Карно:

y_1y_2h / $p_1x_1x_2$	000	001	011	010	110	111	101	100
000	*	*	*	*	*	*	*	*
001	*	*	*	*	*	*	*	*
011	*	*	1	*	*	*	*	*
010	*	*	0	*	*	*	*	*
110	0	*	1	*	*	*	*	*
111	0	*	1	*	*	*	*	*
101	0	0	1	1	*	*	*	*
100	1	0	1	0	1	0	1	0

$f_{min(\text{ДНФ})} = \big((x_1 \oplus 1 + h \oplus 1) \oplus 1\big) + \Big(\big((P \oplus 1) + (x \oplus 1) + x_2\big) \oplus 1\Big) + \big((P \oplus 1) + x_1 + (x_2 \oplus 1) \oplus 1 + (P + (x \oplus 1) + (x_2 \oplus 1) + y_1 \oplus 1)\big) \oplus 1 + \big(P + x_1 + x_2 + y2 \quad + h\big) \oplus 1$

Минимизация функции Q2 картой Карно:

y_1y_2h / $p_1x_1x_2$	000	001	011	010	110	111	101	100
000	*	*	*	*	*	*	*	*
001	*	*	*	*	*	*	*	*
011	*	*	1	*	*	*	*	*
010	*	*	0	*	*	*	*	*
110	0	*	1	*	*	*	*	*
111	0	*	1	*	*	*	*	*
101	0	1	1	0	*	*	*	*
100	0	1	0	0	1	1	1	0

$f_{min(\text{ДНФ})} = (x_2 \oplus 1 + h \oplus 1) \oplus 1 + (x_1 \oplus 1 + y_1 \oplus 1 + y_2 \oplus 1) \oplus 1 + x_1 \oplus 1 +$
$(p \oplus 1 + x_1 \oplus 1) \oplus 1 + P \oplus 1 + x_2 \oplus 1$

4. ЛОГИЧЕСКИЙ СИНТЕЗ ОДНОРАЗРЯДНОГО ЧЕТВЕРИЧНОГО СУММАТОРА НА ОСНОВЕ МУЛЬТИПЛЕКСОРА

Мультиплексор – это логическая схема, имеющая n информационных входов, m управляющих входов и один выход.

Принцип работы мультиплексора состоит в следующем. На выход мультиплексора может быть пропущен без изменений любой (один) логический сигнал, поступающий на один из информационных входов. Порядковый номер информационного входа, значение которого в данный момент должно быть передано на выход, определяется двоичным кодом, поданным на управляющие входы.

Функции ОЧС зависят от пяти переменных. Удобно взять мультиплексор с тремя адресными входами, это позволит упростить одну нашу большую функцию от пяти аргументов до восьми функций от двух переменных. Функции от двух переменных достаточно просты для того, чтобы самостоятельно заметить их минимальную форму.

Синтез дополнительных логических схем для ПФ ОЧС приведен в таблице 4.1.

Таблица 4.1. – Таблица истинности для ОЧС на мультиплексорах

a_1	a_2	b_1	b_2	**p**	**П**	**ВЫХОД П**	S_1	**ВЫХОД S1**	S_2	**ВЫХОД S2**
0	0	0	0	0	1	b_2p+b_2'	1	$b_2'p'$	0	b_2p
0	0	0	0	1	1		0		0	
0	0	0	1	0	0		0		0	
0	0	0	1	1	1		0		1	
0	0	1	0	0	1	“1”	1	b_2p+b_2'	1	$b_2+b_2'p'$
0	0	1	0	1	1		1		0	
0	0	1	1	0	1		0		1	
0	0	1	1	1	1		1		1	
0	1	0	0	0	0	$b_2'p$	0	b_2p	0	$b_2+b_2'p$
0	1	0	0	1	1		0		1	
0	1	0	1	0	0		0		1	
0	1	0	1	1	0		1		1	
0	1	1	0	0	0	“0”	1	$b_2'p'+b_2$	0	b_2p'
0	1	1	0	1	0		0		0	
0	1	1	1	0	0		1		1	
0	1	1	1	1	0		1		0	

Продолжение схемы 4.1

1	0	0	0	0	1	b_2'	1	$b_2'+b_2p'$	1	$b_2'p'$
1	0	0	0	1	1		1		0	
1	0	0	1	0	0		1		0	
1	0	0	1	1	0		0		0	
1	0	1	0	0	1	b_2p+b_2'	0	$b_2'p$	1	$b_2'+\ b_2p$
1	0	1	0	1	1		1		1	
1	0	1	1	0	0		0		0	
1	0	1	1	1	1		0		1	
1	1	0	0	0	1	b_2'	0	$b_2+b_2'p$	1	$b_2'+\ b_2p'$
1	1	0	0	1	1		1		1	
1	1	0	1	0	0		1		1	
1	1	0	1	1	0		1		0	
1	1	1	0	0	0	$b_2'p$	0	b_2p'	0	$b_2'p$
1	1	1	0	1	1		0		1	
1	1	1	1	0	0		1		0	
1	1	1	1	1	0		0		0	

5 ЛОГИЧЕСКИЙ СИНТЕЗ ПРЕОБРАЗОВАТЕЛЯ МНОЖИТЕЛЯ (ПМ)

Преобразователь множителя (ПМ) служит для исключения из множителя диад 11, заменяя их на триады $10\overline{1}$.

Таблица 5.1 - Таблица истинности ПМ.

Вх. диада		Мл. бит	Зн.	Вых. диада	
Q_n	Q_{n-1}	Q_{n-2}	P	S_1	S_2
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	0	1
0	1	1	0	1	0
1	0	0	1	1	0
1	0	1	1	0	1
1	1	0	1	0	1
1	1	1	1	0	0

Минимизируем выходные функции P, S_1иS_2картами Карно.

Таблица 5.2 – Минимизация функции P

$Q_{n-1}Q_{n-2}$

Q_n	00	01	11	10
0				
1	1	1	1	1

$P_{\text{МДНФ}} = Q_n$

Таблица 5.3 – Минимизация функции S_1

$Q_{n-1}Q_{n-2}$

Q_n	00	01	11	10
0			1	
1	1			

$S_{1\text{МДНФ}} = Q_n\overline{Q}_{n-1}\overline{Q}_{n-2} + \overline{Q}_nQ_{n-1}Q_{n-2}$

Таблица 5.4 – Минимизация функции S_2

$Q_{n-1}Q_{n-2}$

Q_n	00	01	11	10
0		1		1
1		1		1

$$S_{2\text{МДНФ}} = \overline{Q}_{n-1}Q_{n-2} + Q_{n-1}\overline{Q}_{n-2} = Q_{n-1}\oplus Q_{n-2}$$

Функциональная схема ПМ приведена в приложении Д.

6 ВРЕМЕННЫЕ ЗАТРАТЫ НА УМНОЖЕНИЕ И ЭФФЕКТИВНОСТЬ МИНИМИЗАЦИИ

Формула расчёта временных затрат на умножение:

$T = n*(T_{ПМ} + T_{ФДК} + T_{СДВ} + T_{ОЧУС} + m*T_{ОЧС})$, где

$T_{ПМ}$ – время преобразования множителя;

$T_{ФДК}$ – время формирования дополнительного кода множимого;

$T_{ОЧУС}$ – время умножения на ОЧУС;

$T_{ОЧС}$ – время формирования единицы переноса в ОЧС;

$T_{СДВ}$ – время сдвига частичного произведения;

n – количество разрядов множителя;

m – количество разрядов множимого.

Для проведения оценки эффективности минимизации переключательных функций необходимо посчитать цену схемы до минимизации и цену схемы после минимизации. Эффективность минимизации k определяется как:

$$k = \frac{C_{до_{мин}}}{C_{после_{мин}}}$$

Таблица 6.1 – Эффективность минимизации ОЧУС

Вых. схемы	Рассчитанная цена схемы		Эфф. мин. k
	До минимизации	После минимизации	
P	c=	c=	
Q_1	c=	c=	
Q_2	c=	c=	

Таблица 6.2 – Эффективность минимизации ОЧС

Вых. схемы	Рассчитанная цена схемы		Эфф. мин. k
	До минимизации	После минимизации	
П	c=	c=	
S_1	c=	c=	
S_2	c=	c=	

ЗАКЛЮЧЕНИЕ

В процессе выполнения курсовой работы была разработана структурная схема сумматора-умножителя второго типа, а также функциональные схемы основных узлов данного устройства. Для уменьшения стоимости логических схем были выполнены минимизации переключательных функций различными способами. Такой подход позволил выявить достоинства и недостатки этих алгоритмов.

В качестве главного достоинства минимизации картами Карно-Вейча можно выделить простоту и минимальные затраты времени. Однако применение данного способа для функций многих переменных будет затруднительно. Функциональные схемы были построены в различных логических базисах. Это позволило закрепить теоретические знания основных законов булевой алгебры, например, правило де Моргана. Также можно отметить, что необходимо сократить количество уровней в логической схеме для уменьшения времени работы данного устройства.

Реализация переключательных функций на основе мультиплексоров позволила облегчить процесс минимизации этих функций.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Луцик Ю.А., Лукьянова И.В. – Учебное пособие по курсу "Арифметические и логические основы вычислительной техники". – Минск: БГУИР, 2014 г.
2. Луцик Ю.А., Лукьянова И.В. – Методические указания к курсовому проекту по курсу “Арифметические и логические основы вычислительной техники”. – Мн.: БГУИР, 2004 г.
3. Искра, Н. А. Арифметические и логические основы вычислительной техники: пособие / Н. А. Искра, И. В. Лукьянова, Ю. А. Луцик. – Минск: БГУИР, 2016. – 75 с.

ПРИЛОЖЕНИЕ А
(обязательное)

Сумматор-умножитель второго типа. Схема электрическая структурная

ПРИЛОЖЕНИЕ Б

(обязательное)

Одноразрядный четверичный сумматор.
Схема электрическая функциональная

ПРИЛОЖЕНИЕ В
(обязательное)

Одноразрядный четверичный умножитель-сумматор. Схема электрическая функциональная

ПРИЛОЖЕНИЕ Г
(обязательное)

Одноразрядный четверичный сумматор.
Реализация на мультиплексорах.
Схема электрическая функциональная

ПРИЛОЖЕНИЕ Д
(обязательное)

Преобразователь множителя.
Схема электрическая функциональная

ПРИЛОЖЕНИЕ Е
(обязательное)

Ведомость документов