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
цифровых устройств

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 202ПЗ

Студент Н. Г. Альхимович

Руководитель Ю. А. Луцик

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
цифровых устройств

УТВЕРЖДАЮ
Заведующий кафедрой ЭВМ
_____ Б. В. Никульшин
«____» _____ 20__ г.

ЗАДАНИЕ
по курсовой работе студента
Альхимович Нины Геннадьевны

1. Тема работы: «Проектирование и логический синтез сумматора-умножителя двоично-четверичных чисел»

2. Срок сдачи студентом законченной работы: до 20 мая 2022 г.

3. Исходные данные к работе:
   3.1. исходные сомножители: Мн = 15,79; Мт = 48,33;
   3.2. алгоритм умножения: Б;
   3.3. метод умножения: умножение закодированного двоично-четверичного множимого на два разряда двоичного множителя одновременно в прямых кодах;
   3.4. коды четверичных цифр множимого для перехода к двоично-четверичной системе кодирования: 0_4 – 10, 1_4 – 11, 2_4 – 00, 3_4 – 01;
   3.5. тип синтезируемого умножителя: 2;
   3.6. логический базис для реализации ОЧС: ИЛИ, исключающее ИЛИ, генератор «1»; метод минимизации – карты Карно – Вейча;
   3.7. логический базис для реализации ОЧУ: И, ИЛИ, НЕ; 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 первого типа. Схема электрическая структурная.
   5.2. Одноразрядный четверичный сумматор. Схема электрическая функциональная.
   5.3. Одноразрядный четверичный умножитель. Схема электрическая функциональная.
   5.4. Регистр-аккумулятор. 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 сумматора-умножителя	50	10.03–30.04	С выполнением чертежей
Синтез комбинационных схем устройств на основе мультиплексоров	10	01.05–15.05	С выполнением чертежа
Завершение оформления пояснительной записки	20	15.05–20.05	

Дата выдачи задания: 10 февраля 2022 г.

Руководитель _____ Ю. А. Луцик

ЗАДАНИЕ ПРИНЯЛ К ИСПОЛНЕНИЮ _____

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Данная курсовая работа по дисциплине «Арифметические и логические основы цифровых устройств» предусматривает проектирование и синтез цифровых схем арифметического устройства, выполняющего операции сложения и умножения над числами, представленными в форме с плавающей запятой в двоичной и двоично-четверичной системах счисления (с/с).

По исходным данным необходимо разработать: алгоритм выполнения операции умножения, алгоритм выполнения операции сложения, структурную схему вычислительного устройства, выполняющего сложение и умножение, функциональные схемы основных узлов проектируемого сумматора-умножителя в заданном логическом базисе: комбинационного одноразрядного четверичного сумматора (ОЧС), в том числе и на мультиплексорах, одноразрядного комбинационного четверичного умножителя-сумматора (ОЧУС), комбинационной схемы преобразователя множителя (ПМ). Минимизации переключательных функций по каждому выходу схемы (выполняется с применением алгоритма Рота и карт Карно – Вейча). По результатам разработки необходимо построить схемы перечисленных устройств и оценить эффективность минимизации и время выполнения операций.

1. РАЗРАБОТКА АЛГОРИТМА УМНОЖЕНИЯ

1.1. Перевод сомножителей из десятичной системы счисления в четверичную.

Множимое

```
  15 | 4          0,79
- 12   3        *   4
   3             3,16
               *   4
                 0,64
               *   4
                 2,56
               *   4
                 2,24
```

$Мн_4 = 33,3022$.

В соответствии с заданной кодировкой множимого:
$Мн_{2/4} = 0101,01100000$.

Множитель

```
  48 | 4          0,33
- 48   12 | 4     *   4
   0  - 12   3      1,32
          0       *   4
                    1,28
                  *   4
                    1,12
```

$Мт_4 = 300,111$.

В соответствии с обычной весомозначной кодировкой множителя (для всех вариантов):

$Мт_{2/4} = 110000,010101$.

1.2. Запишем сомножители в форме с плавающей запятой в прямом коде:

Мн = 0,10101100000 $P_{Мн} = 0.1000 + 02_{10}$ – закодировано по заданию,
Мт = 0,110000010101 $P_{Мт} = 0.0011 + 03_{10}$ – закодировано традиционно.

1.3. Умножение двух чисел с плавающей запятой на два разряда множителя одновременно в прямых кодах. Это сводится к сложению порядков, формированию знака произведения, преобразованию

разрядов множителя согласно алгоритму и перемножению мантисс сомножителей.

Порядок произведения будет следующим:

$P_{Мн}$ = 0.1000 02_4
$P_{Мт}$ = 0.0011 03_4
$P_{Мн \cdot Мт}$ =0.1111 11_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.

Для умножения мантисс необходимо предварительно преобразовать множитель. При умножении чисел в прямых кодах диада $11(3_4)$ заменяется на триаду $10\bar{1}$. Преобразованный множитель имеет вид: $Мт_4^{п} = 1\bar{1}00111$ или $Мт_2^{п} = 010\bar{1}0000010101$. Перемножение мантисс по алгоритму «Б» приведено в таблице 1.1.

Таблица 1.1 – Перемножение мантисс

Четверичная с/с		**Двоично-четверичная с/с**		**Комментарии**
1		**2**		**3**
0.	000000000000	0.	10 10 10 10 10 10 10 10 10 10 10 10	$\sum_0^{ч} = 0$
0.	000000333022	0.	10 10 10 10 10 10 01 01 01 10 00 00	$П_1^{ч} = Мн \cdot 2^0$
0.	000000333022	0.	10 10 10 10 10 10 01 01 01 10 00 00	$\sum_1^{ч}$
0.	000003330220	0.	10 10 10 10 10 01 01 01 10 00 00 10	$П_2^{ч} = Мн \cdot 2^1$
0.	000010323302	0.	10 10 10 10 11 10 01 00 01 01 10 00	$\sum_2^{ч}$
0.	000033302200	0.	10 10 10 10 01 01 01 10 00 00 10 10	$П_3^{ч} = Мн \cdot 2^2$
0.	000110232102	0.	10 10 10 11 11 10 00 01 00 11 10 00	$\sum_3^{ч} = \sum_4^{ч} = \sum_5^{ч}$
3.	300031200000	1.	01 10 10 10 01 11 00 10 10 10 10 10	$П_6^{ч} = Мн \cdot (-1) \cdot 2^5$
3.	300202032102	1.	01 10 10 00 10 00 10 01 00 11 10 00	$\sum_6^{ч}$
0.	333022000000	0.	01 01 01 10 00 00 10 10 10 10 10 10	$П_7^{ч} = Мн \cdot 2^6$
0.	233230032102	0.	00 01 01 00 01 10 10 01 00 11 10 00	$\sum_7^{ч}$

После окончания умножения необходимо оценить погрешность вычислений. Для этого полученное произведение ($Мн \cdot Мт_4$= 0,233230032102, $P_{Мн \cdot Мт}$ = 5) приводится к нулевому порядку, а затем переводится в десятичную систему счисления:

$Мн \cdot Мт_4 = 23323{,}0032102 \quad P_{Мн \cdot Мт} = 0;$
$Мн \cdot Мт_{10} = 763{,}0558.$

Результат прямого перемножения операндов даёт следующее значение:

$Мн_{10} \cdot Мт_{10} = 15{,}79 \cdot 48{,}33 = 763{,}1307.$

Абсолютная погрешность:

$\Delta = 763{,}1307 - 763{,}0558 = 0{,}0749.$

Относительная погрешность:

$$\delta = \frac{\Delta}{Мн \cdot Мт} = \frac{0{,}0749}{763{,}1307} = 0{,}00009815 \ (\delta = 0{,}00981483\ \%).$$

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 «Б» приведена на рисунке приложения А.

Если устройство работает как сумматор, то оба слагаемых последовательно (за два такта) заносятся в регистр множимого, а на управляющий вход формирователя дополнительного кода F_2 поступает «1».

Если устройство работает как умножитель, то множимое и множитель помещаются в соответствующие регистры, а на управляющий вход ФДК F_2 поступает «0».

3. РАЗРАБОТКА ФУНКЦИОНАЛЬНЫХ СХЕМ ОСНОВНЫХ УЗЛОВ СУММАТОРА-УМНОЖИТЕЛЯ

3.1. 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.

Принцип работы ОЧУС представлен с помощью таблицы истинности (таблица 3.1).

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

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

Управляющие вход h определяет тип операции:

«0» – умножение закодированных цифр, поступивших на информационные входы;

«1» – вывод на входы без изменения значения разрядов, поступивших из регистра множимого.

Таблица 3.1.1 – Таблица истинности ОЧУС

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	0	0	0	2·0+0=00
0	0	0	0	0	1	0	0	0	Выход – код «00»
0	0	0	0	1	0	0	0	0	2·1+0=02
0	0	0	0	1	1	0	0	0	Выход – код «00»
0	0	0	1	0	0	1	1	0	2·2+0=10
0	0	0	1	0	1	0	0	0	Выход – код «00»
0	0	0	1	1	0	x	x	x	2·3+0=12
0	0	0	1	1	1	x	x	x	Выход – код «00»
0	0	1	0	0	0	0	1	0	3·0+0=00
0	0	1	0	0	1	0	0	1	Выход – код «01»
0	0	1	0	1	0	0	0	1	3·1+0=03
0	0	1	0	1	1	0	0	1	Выход – код «01»
0	0	1	1	0	0	1	0	0	3·2+0=12
0	0	1	1	0	1	0	0	1	Выход – код «01»
0	0	1	1	1	0	x	x	x	3·3+0=21
0	0	1	1	1	1	x	x	x	Выход – код «01»
0	1	0	0	0	0	0	1	0	0·0+0=00
0	1	0	0	0	1	0	1	0	Выход – код «10»
0	1	0	0	1	0	0	1	0	0·1+0=00

0	1	0	0	1	1	0	1	0	Выход – код «10»
0	1	0	1	0	0	0	1	0	0·2+0=00
0	1	0	1	0	1	0	1	0	Выход – код «10»
0	1	0	1	1	0	x	x	x	0·3+0=00
0	1	0	1	1	1	x	x	x	Выход – код «10»
0	1	1	0	0	0	0	1	0	1·0+0=00
0	1	1	0	0	1	0	1	1	Выход – код «11»
0	1	1	0	1	0	0	1	1	1·1+0=01
0	1	1	0	1	1	0	1	1	Выход – код «11»
0	1	1	1	0	0	0	0	0	1·2+0=02
0	1	1	1	0	1	0	1	1	Выход – код «11»
0	1	1	1	1	0	x	x	x	1·3+0=03
0	1	1	1	1	1	x	x	x	Выход – код «11»
1	0	0	0	0	0	x	x	x	2·0+1=01
1	0	0	0	0	1	x	x	x	Выход – код «00»
1	0	0	0	1	0	x	x	x	2·1+1=03
1	0	0	0	1	1	x	x	x	Выход – код «00»
1	0	0	1	0	0	1	1	1	2·2+1=11
1	0	0	1	0	1	x	x	x	Выход – код «00»
1	0	0	1	1	0	x	x	x	2·3+1=13
1	0	0	1	1	1	x	x	x	Выход – код «00»
1	0	1	0	0	0	x	x	x	3·0+1=01
1	0	1	0	0	1	x	x	x	Выход – код «01»
1	0	1	0	1	0	x	x	x	3·1+1=10
1	0	1	0	1	1	x	x	x	Выход – код «01»
1	0	1	1	0	0	1	0	1	3·2+1=13
1	0	1	1	0	1	x	x	x	Выход – код «01»
1	0	1	1	1	0	x	x	x	3·3+1=22
1	0	1	1	1	1	x	x	x	Выход – код «01»
1	1	0	0	0	0	x	x	x	0·0+1=01
1	1	0	0	0	1	x	x	x	Выход – код «10»
1	1	0	0	1	0	x	x	x	0·1+1=01
1	1	0	0	1	1	x	x	x	Выход – код «10»
1	1	0	1	0	0	0	1	1	0·2+1=01
1	1	0	1	0	1	x	x	x	Выход – код «10»
1	1	0	1	1	0	x	x	x	0·3+1=01
1	1	0	1	1	1	x	x	x	Выход – код «10»
1	1	1	0	0	0	x	x	x	1·0+1=01
1	1	1	0	0	1	x	x	x	Выход – код «11»
1	1	1	0	1	0	x	x	x	1·1+1=02
1	1	1	0	1	1	x	x	x	Выход – код «11»
1	1	1	1	0	0	0	0	1	1·2+1=03

1	1	1	1	0	1	x	x	x	Выход – код «11»
1	1	1	1	1	0	x	x	x	1·3+1=10
1	1	1	1	1	1	x	x	x	Выход – код «11»

В таблице 3.1 выделено 36 безразличных наборов, т. к. на входы ОЧУС из разрядов множителя не может поступить код «11», при работе ОЧУС как сумматора на вход переноса не может поступить единица, а при умножении на ноль или единицу на вход переноса также не может поступить единица.

Минимизацию переключательных функций проведём с помощью карт Вейча, функцию Q_2 также минимизируем с помощью алгоритма Рота. На рисунках 3.1.1 – 3.1.3 символом “x” отмечены наборы, на которых функция может принимать произвольное значение (безразличные наборы).

Для функции P:

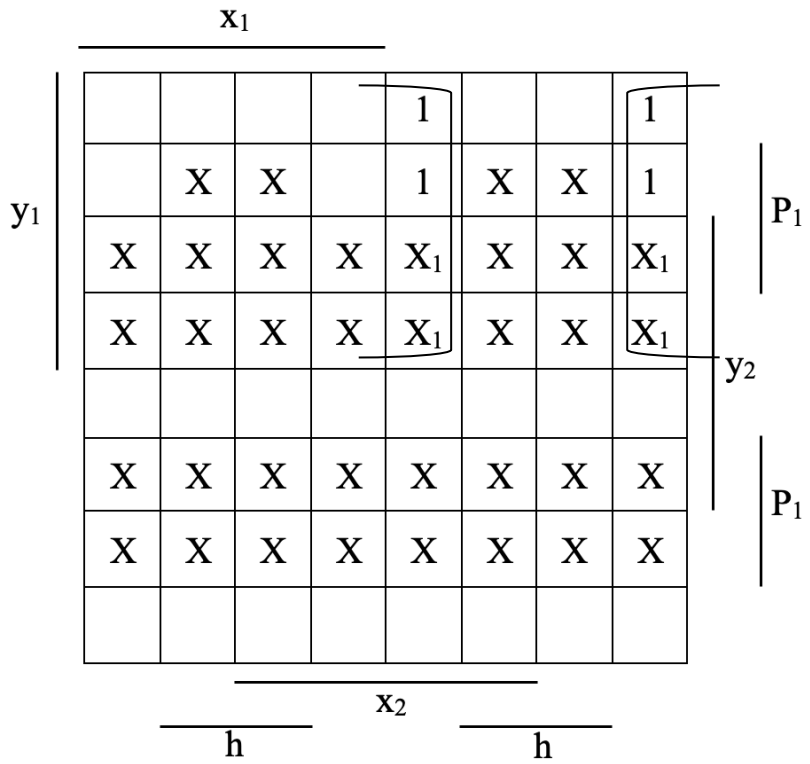

Рисунок 3.1.1 – Минимизация функции P при помощи карты Вейча

$P_{МДНФ} = \overline{x_1} y_1 \bar{h}$

Функция для реализации в заданном базисе (А1) будет иметь вид:

$P_{МДНФ} = \overline{x_1} y_1 \bar{h}$

Для функции Q_1:

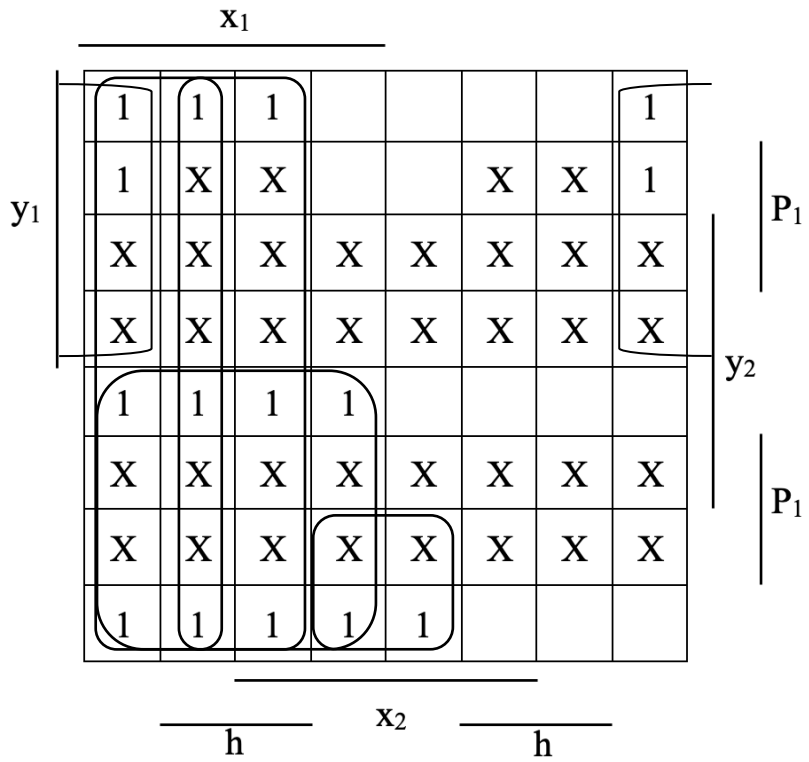

Рисунок 3.1.2 – Минимизация функции Q_1 при помощи карты Вейча

$Q_{1МДНФ} = x_1 h + x_1\overline{x_2} + x_1\overline{y_1} + \overline{x_2}y_1\bar{h} + x_2\overline{y_1 y_2}\bar{h}$

Функция для реализации в заданном базисе (А1) будет иметь вид:

$Q_{1МДНФ} = x_1 h + x_1\overline{x_2} + x_1\overline{y_1} + \overline{x_2}y_1\bar{h} + x_2\overline{y_1 y_2}\bar{h}$

Для функции Q_2:

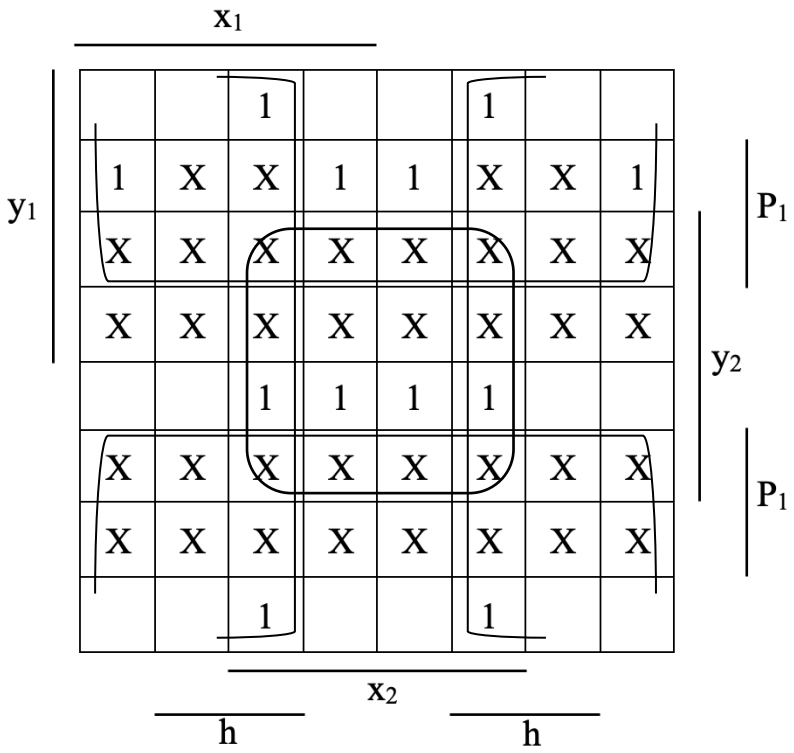

Рисунок 3.1.3 – Минимизация функции Q_2 при помощи карты Вейча

$Q_{2МДНФ} = p_1 + x_2y_2 + x_2h$

Минимизацию переключательной функции Q_2 проведём также с помощью алгоритма Рота.

Определим множество единичных кубов:

L = {001001, 001010, 001011, 001101, 011001, 011010, 011011, 011101, 100100, 101100, 110100, 111100}

Далее определим множество безразличных кубов:

N = {000110, 000111, 001110, 001111, 010110, 010111, 011110, 011111, 100000, 100001, 100010, 100011, 100101, 100110, 100111, 101000, 101001, 101010, 101011, 101101, 101110, 101111, 110000, 110001, 110010, 110011, 110101, 110110, 110111, 111000, 111001, 111010, 111011, 111101, 111110, 111111}

Склеим всевозможные кубы во множествах L и N:

L = {0x1010, 0x1101, 0x10x1, 1xx100}

N = {1xx110, 0xx11x, 1xx1x1, 1xx0xx}

Сформируем множество $C_0 = L \cup N$:

C_0 = {0x1010, 0x1101, 0x10x1, 1xx100, 1xx110, 0xx11x, 1xx1x1, 1xx0xx}

Первым этапом алгоритма Рота является нахождение множества простых импликант.

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

Таблица 3.1.2 – Поиск простых импликант (C_0*C_0)

C_0*C_0	0x1010	0x1101	0x10x1	1xx100	1xx110	0xx11x	1xx1x1	1xx0xx
0x1010	—							
0x1101		—						
0x10x1	0x101y	0x1y01	—					
1xx100				—				
1xx110				1xx1y0	—			
0xx11x	0x1y10	0x11y1	0x1y11		yxx110	—		
1xx1x1		yx1101		1xx10y	1xx11y	yxx111	—	
1xx0xx	yx1010		yx10x1	1xxy00	1xxy10		1xxyx1	—
A_1	0x101x 0x1x10 xx1010	0x1x01 0x11x1 xx1101	0x1x11 xx10x1	1xx1x0 1xx10x 1xxx00	xxx110 1xx11x 1xxx10	xxx111	1xxxx1	∅

В результате этой операции сформируется новое множество кубов:

A_1 = {0x101x, 0x1x10, xx1010, 0x1x01, 0x11x1, xx1101, 0x1x11, xx10x1, 1xx1x0, 1xx10x, 1xxx00, xxx110, 1xx11x, 1xxx10, xxx111, 1xxxx1}

Множество Z_0 кубов, не участвовавших в образовании новых кубов, пустое:

Z_0 = {∅}

B_1 = {0x1010, 0x1101, 0x10x1, 1xx100, 1xx110, 0xx11x, 1xx1x1, 1xx0xx}

Далее формируется множество $C_1 = A_1 \cup B_1$. Для уменьшения мощности множества кубов C_1 выполним операцию поглощения кубов. Ее результат:

C_1 = {0x101x, 0x1x10, xx1010, 0x1x01, 0x11x1, xx1101, 0x1x11, xx10x1, 1xx1x0, 1xx10x, 1xxx00, xxx110, 1xx11x, 1xxx10, xxx111, 1xxxx1, 0xx11x, 1xx0xx}

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

A2 = {0x1x1x, xx101x, xx1x10, 0x1xx1, xx1x01, xx11x1, xx1x11, 1xx1xx, 1xxxx0, 1xxx0x, xxx11x, 1xxx1x}

Z_1 = {Ø}

B_2 = {0x101x, 0x1x10, xx1010, 0x1x01, 0x11x1, xx1101, 0x1x11, xx10x1, 1xx1x0, 1xx10x, 1xxx00, xxx110, 1xx11x, 1xxx10, xxx111, 1xxxx1, 0xx11x, 1xx0xx}

C_2 = {0x1x1x, xx101x, xx1x10, 0x1xx1, xx1x01, xx11x1, xx1x11, 1xx1xx, 1xxxx0, 1xxx0x, xxx11x, 1xxx1x, xx10x1, 1xxxx1, 1xx0xx}

В таблице 3.1.4 приведён следующий шаг поиска простых импликант с помощью операции C_2*C_2.

Таблица 3.1.3 – Поиск простых импликант (C^1*C^1)

C_1*C_1	0x101x	0x1x10	xx1010	0x1x01	0x11x1	xx1101	0x1x11	xx10x1	1xx1x0	1xx10x	1xxx00	xxx110	1xx11x	1xxx10	xxx111	1xxxx1
0x101x	—															
0x1x10		—														
xx1010			—													
0x1x01				—												
0x11x1					—											
xx1101						—										
0x1x11		0x1x1y		0x1xy1			—									
xx10x1			xx101y		0x1yx1	xx1y01		—								
1xx1x0									—							
1xx10x										—						
1xxx00											—					
xxx110			xx1y10							1xx1y0	1xx1y0	—				
1xx11x										1xx1yx			—			
1xxx10		yx1x10								1xx1y0	1xxxy0			—		
xxx111						xx11y1		xx1y11	1xx11y	1xx1y1		xxx11y		1xx11y	—	
1xxxx1				yx1x01	yx11x1		yx1x11		1xx1xy		1xxx0y	1xx11y		1xxx1y		—
0xx11x	0x1y1x								yxx110				yxx11x	yxx110		yxx111
1xx0xx	yx101x								1xxyx0	1xxy0x		1xxy10	1xxy1x		1xxy11	
A_2	0x1x1x xx101x	xx1x10	xx101x	0x1xx1 xx1x01	xx11x1	∅	xx1x11	∅	1xx1xx xxx110 1xxxx0	1xx1x0 1xxx0x	∅	xxx11x 1xx11x 1xxx10	1xxx1x	∅	1xxx11	xxx111

Таблица 3.1.4 – Поиск простых импликант (C_2*C_2)

C_2*C_2	0x1x1x	xx101x	xx1x10	0x1xx1	xx1x01	xx11x1	xx1x11	1xx1xx	1xxxx0	1xxx0x	xxx11x	1xxx1x
0x1x1x	—											
xx101x		—										
xx1x10			—									
0x1xx1				—								
xx1x01					—							
xx11x1						—						
xx1x11			xx1x1y		xx1xy1		—					
1xx1xx								—				
1xxxx0									—			
1xxx0x										—		
xxx11x		xx1y1x								1xx1yx	—	
1xxx1x	yx1x1x									1xxxyx		—
xx10x1						xx1yx1						
1xxxx1				yx1xx1					1xxxxy			
1xx0xx								1xxyxx			1xxy1x	
A_3	xx1x1x	Ø	Ø	xx1xx1	Ø	Ø	Ø	1xxxxx	Ø	Ø	Ø	Ø

A_3 = {xx1x1x, xx1xx1, 1xxxxx}

Z_2 = {Ø}

B_3 = {0x1x1x, xx101x, xx1x10, 0x1xx1, xx1x01, xx11x1, xx1x11, 1xx1xx, 1xxxx0, 1xxx0x, xxx11x, 1xxx1x, xx10x1, 1xxxx1, 1xx0xx}

C_3 = {xx1x1x, xx1xx1, 1xxxxx, xxx11x}

В таблице 3.1.5 приведён следующий шаг поиска простых импликант с помощью операции C_3*C_3.

Таблица 3.1.5 – Поиск простых импликант (C_3*C_3)

C_3*C_3	xx1x1x	xx1xx1	1xxxxx
xx1x1x	—		
xx1xx1		—	
1xxxxx			—
xxx11x			
A_4	Ø	Ø	Ø

Из таблицы следует, что A_4 = Ø. Таким образом, новых кубов при выполнении операции C_3*C_3 не было получено.

$B_4 = C_3 \setminus Z_3$ = Ø

$C_4 = A_4 \cup B_4$ = Ø

На этом процесс выявления простых импликант окончен. Таким образом сформировано множество простых импликант:

Z = {xx1x1x, xx1xx1, 1xxxxx, xxx11x}

Следующий этап – поиск *L*-экстремалей на множестве простых импликант (таблица 3.1.6). Для этого из каждой простой импликанты поочередно вычитаются все остальные простые импликанты Z#(Z\z).

Таблица 3.1.6 – Поиск *L*-экстремалей

z#(Z-z)	xx1x1x	xx1xx1	1xxxxx	xxx11x
xx1x1x	—	zzzz0z xx1x01	zz0z0z 1x0xxx 1xxx0x	zz0zzz xx011x

xx1xx1	zzzzz0 xx1x10	—	zzyzz0 1x0xxx zz0zz0 1x0x0x 1xxx00	zzyzz0 xx011x
1xxxxx	0zzzzz 0x1x10	0zzzzz 0x1x01	—	0zzzzz 0x011x
xxx11x	zzz0zz 0x1010	zzz0yz 0x1x01	zzz00z 1x00xx 1x0x0x zzz0yz 1x0x0x zzz0yz 1xxx00	—
Остаток	0x1010	0x1x01	1x00xx 1x0x0x 1xxx00	0x011x

Результат операции (последняя строка таблицы) указывает на то, что *L*-экстремалями стали следующие простые импликанты:

E = {xx1x1x, xx1xx1, 1xxxxx, xxx11x}

Необходимо проверить, нет ли среди полученных *L*-экстремалей таких, которые стали *L*-экстремалями за счёт безразличных кубов. Для этого в таблице 3.1.7 из кубов множества *L* вычитаются остатки простых импликант, полученные в таблице 3.1.6.

Таблица 3.1.7 – Проверка L-экстремалей

z#(Z-z) ∩ L	001 001	001 010	001 011	001 101	011 001	011 010	011 011	011 101	100 100	101 100	110 100	111 100
0x1 010	Ø	001 010	Ø	Ø	Ø	011 010	Ø	Ø	Ø	Ø	Ø	Ø
0x1 x01	001 001	Ø	Ø	001 101	011 001	Ø	Ø	011 101	Ø	Ø	Ø	Ø
1x0 0xx	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø
1x0 x0x	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	100 100	Ø	110 100	Ø
1xx x00	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	100 100	101 100	110 100	111 100
0x0 11x	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø

По результатам таблицы 3.1.7 L-экстремалью, не связанной с единичными наборами, стал куб xxx11x. Этот куб не будет входить в минимальное покрытие.

E = {xx1x1x, xx1xx1, 1xxxxx}

Необходимо проанализировать, какие из вершин комплекса L не покрываются L-экстремалями. Для этого из каждого куба комплекса L вычитаем элементы множества E (таблица 3.1.8). В результате вычитания получим $L_1 = L\#E$.

Таблица 3.1.8 – Поиск непокрытых исходных наборов

L#E	001001	001010	001011	001101	011001	011010	011011	011101	100100	101100	110100	111100
xx1x1x	zzzzyz 001001	zzzzzz Ø	zzzzzz Ø	zzzzyz 001101	zzzzyz 011001	zzzzzz Ø	zzzzzz Ø	zzzzyz 011101	zzyzyz 100100	zzzzyz 101100	zzyzyz 110100	zzzzyz 111100
xx1xx1	zzzzzz Ø	Ø	Ø	zzzzzz Ø	Ø	Ø	Ø	zzzzzz Ø	zzyzzy 100100	zzzzzy 101100	zzyzzy 110100	zzzzzy 111100
1xxxxx	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	zzzzzz Ø	zzzzzz Ø	zzzzzz Ø	zzzzzz Ø
Остаток	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø

Из таблицы 3.1.8 видно, что все единичные кубы покрыты.

Следовательно, существует одна тупиковая (минимальная) форма:

F_{min} = {xx1x1x, xx1xx1, 1xxxxx}

$Q_{2МДНФ} = p_1 + x_2y_2 + x_2h$

Функция для реализации в заданном базисе (А1) будет иметь вид:

$Q_{2МДНФ} = p_1 + x_2y_2 + x_2h$

Эффективность минимизации можно оценить отношением числа входов схем, реализующих переключательную функцию до и после минимизации:

$К_Р = \frac{4*6+4+6}{3+2} = 6{,}8$

$К_{Q1} = \frac{15*6+15+6}{3*2+3+4+5+4} = 5{,}1$

$К_{Q2} = \frac{12*6+12+6}{2*2+3} = 12{,}9$

Функциональная схема ОЧУС в заданном базисе представлена в приложении Б.

3.2. 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.

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

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

Поскольку ОЧС синтезируется для схемы второго типа, то безразличные наборы в таблице истинности отсутствуют.

Таблица 3.2.1 – Таблица истинности ОЧС

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

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

Минимизацию переключательных функций проведём с помощью карт Карно. Заполненные карты приведены на рисунках 3.2.1 – 3.2.3.

Для функции П:

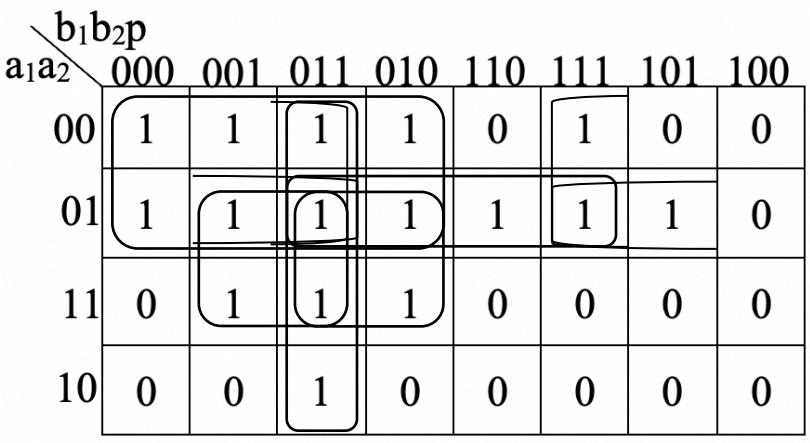

Рисунок 3.2.1 – Минимизация функции П при помощи карты Карно

$П_{МДНФ} = \overline{a_1}\overline{b_1} + \overline{b_1}b_2(p + a_2) + \overline{a_1}a_2(b_2 + p) + \overline{a_1}b_2p + a_2\overline{b_1}p$

Функция для реализации в заданном базисе (А3) будет иметь вид:

$П_{МДНФ} = (a_1 + b_1) \oplus 1 + \big(b_1 + (b_2 \oplus 1) + (p \oplus 1)\big) \oplus 1 + \big(b_1 + (b_2 \oplus 1) + (a_2 \oplus 1)\big) \oplus 1 + \big(a_1 + (a_2 \oplus 1) + (b_2 \oplus 1)\big) \oplus 1 + \big(a_1 + (a_2 \oplus 1) + (p \oplus 1)\big) \oplus 1 + \big(a_1 + (b_2 \oplus 1) + (p \oplus 1)\big) \oplus 1 + ((a_2 \oplus 1) + b_1 + (p \oplus 1)) \oplus 1$

Для функции S1:

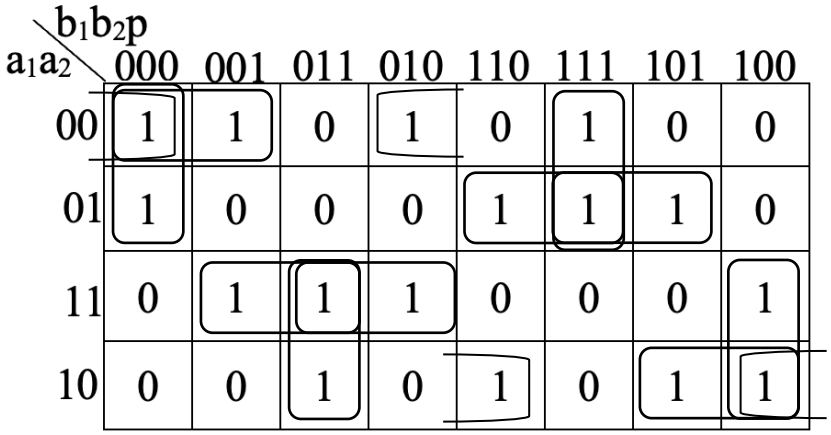

Рисунок 3.2.2 – Минимизация функции S_1 при помощи карты Карно

$S_{1МДНФ} = a_1 b_1 \overline{b_2} \bar{p} + a_1 \overline{a_2} b_1 \overline{b_2} + \overline{a_1} a_2 b_1 p + \overline{a_1} b_1 b_2 p + a_1 \overline{a_2} b_1 \bar{p} + \overline{a_1} a_2 b_1 b_2 + a_1 \overline{b_1} b_2 p + a_1 a_2 \overline{b_1} b_2 + a_1 a_2 \overline{b_1} p + \overline{a_1} \overline{b_1} \overline{b_2} \bar{p} + \overline{a_1 a_2} \overline{b_1} \bar{p} + \overline{a_1 a_2} \overline{b_1} \overline{b_2}$

Функция для реализации в заданном базисе (А3) будет иметь вид:

$S_{1МДНФ} = \big((a_1 \oplus 1) + (b_1 \oplus 1) + b_2 + p\big) \oplus 1 + \big((a_1 \oplus 1) + a_2 + (b_1 \oplus 1) + b_2\big) \oplus 1 + \big(a_1 + (a_2 \oplus 1) + (b_1 \oplus 1) + (p \oplus 1)\big) \oplus 1 + \big(a_1 + (b_1 \oplus 1) + (b_2 \oplus 1) + (p \oplus 1)\big) \oplus 1 + \big((a_1 \oplus 1) + a_2 + (b_1 \oplus 1) + p\big) \oplus 1 + \big(a_1 + (a_2 \oplus 1) + (b_1 \oplus 1) + (b_2 \oplus 1)\big) \oplus 1 + \big((a_1 \oplus 1) + b_1 + (b_2 \oplus 1) + (p \oplus 1)\big) \oplus 1 + \big((a_1 \oplus 1) + (a_2 \oplus 1) + b_1 + (b_2 \oplus 1)\big) \oplus 1 + \big((a_1 \oplus 1) + (a_2 \oplus 1) + b_1 + (p \oplus 1)\big) \oplus 1 + (a_1 + b_1 + b_2 + p) \oplus 1 + (a_1 + a_2 + b_1 + p) \oplus 1 + (a_1 + a_2 + b_1 + b_2) \oplus 1$

Для функции S2:

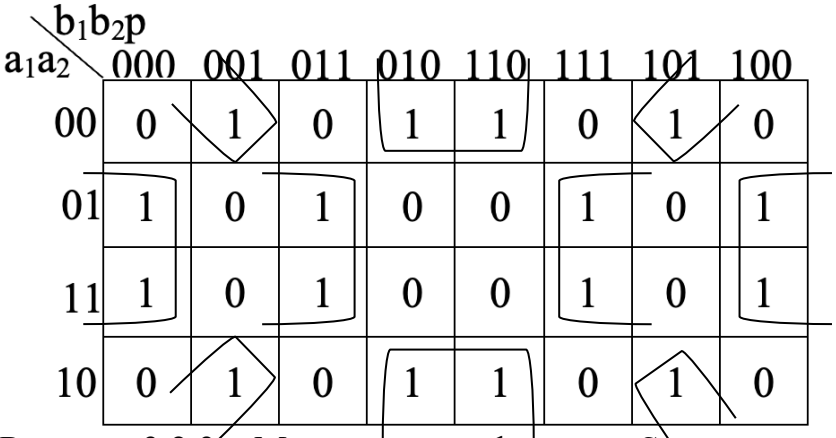

Рисунок 3.2.3 – Минимизация функции S_2 при помощи карты Карно

$S_{2МДНФ} = \overline{a_2} b_2 \bar{p} + a_2 b_2 p + \overline{a_2} \overline{b_2} p + a_2 \overline{b_2} \bar{p}$

Функция для реализации в заданном базисе (А3) будет иметь вид:

$S_{2МДНФ} = (a_2 + (b_2 \oplus 1) + p) \oplus 1 + \big((a_2 \oplus 1) + (b_2 \oplus 1) + (p \oplus 1)\big) \oplus 1 + \big(a_2 + b_2 + (p \oplus 1)\big) \oplus 1 + ((a_2 \oplus 1) + b_2 + p) \oplus 1$

Эффективность минимизации можно оценить отношением числа входов схем, реализующих переключательную функцию до и после минимизации:

$К_П = \frac{16*5+16+5}{2+4*3+2*2+5+2} = 4{,}04$

$К_{S1} = \frac{16*5+16+5}{12*4+12+5} = 1{,}6$

$К_{S2} = \frac{16*5+16+5}{4*3+4+3} = 5{,}3$

Функциональная схема ОЧС в заданном базисе представлена в приложении В.

4. ЛОГИЧЕСКИЙ СИНТЕЗ ОДНОРАЗРЯДНОГО ЧЕТВЕРИЧНОГО СУММАТОРА НА ОСНОВЕ МУЛЬТИПЛЕКСОРА

Мультиплексор – это логическая схема, имеющая n информационных входов, m управляющих входов и один выход. При этом будет выполняться условие n = 2m.

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 большую функцию от пяти аргументов до восьми функций от двух переменных.

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

Таблица 4.1. – Таблица истинности для ОЧС на мультиплексорах

a_1	a_2	b_1	b_2	p	П	П	S_1	S_1	S_2	S_2
1	**2**	**3**	**4**	**5**	**6**	7	**8**	**9**	**10**	**11**
0	0	0	0	0	1	1	1	$\overline{b_2} + \bar{p}$	0	$b_2 \oplus p$
0	0	0	0	1	1		1		1	
0	0	0	1	0	1		1		1	
0	0	0	1	1	1		0		0	
0	0	1	0	0	0	$b_2 p$	0	$b_2 p$	0	$b_2 \oplus p$
0	0	1	0	1	0		0		1	
0	0	1	1	0	0		0		1	
0	0	1	1	1	1		1		0	
0	1	0	0	0	1	1	1	$\overline{b_2}\bar{p}$	1	$\overline{b_2}\bar{p} + b_2 p$
0	1	0	0	1	1		0		0	
0	1	0	1	0	1		0		0	
0	1	0	1	1	1		0		1	
0	1	1	0	0	0	$p + b_2$	0	$p + b_2$	1	$\overline{b_2}\bar{p} + b_2 p$
0	1	1	0	1	1		1		0	
0	1	1	1	0	1		1		0	
0	1	1	1	1	1		1		1	
1	0	0	0	0	0	$b_2 p$	0	$b_2 p$	0	$b_2 \oplus p$
1	0	0	0	1	0		0		1	

1	0	0	1	0	0		0		1	
1	0	0	1	1	1		1		0	
1	0	1	0	0	0	0	1	$\overline{b_2} + \bar{p}$	0	$b_2 \oplus p$
1	0	1	0	1	0		1		1	
1	0	1	1	0	0		1		1	
1	0	1	1	1	0		0		0	
1	1	0	0	0	0	$p + b_2$	0	$p + b_2$	1	$\overline{b_2}\bar{p} + b_2 p$
1	1	0	0	1	1		1		0	
1	1	0	1	0	1		1		0	
1	1	0	1	1	1		1		1	
1	1	1	0	0	0	0	1	$\bar{p} + b_2$	1	$\bar{p} + b_2$
1	1	1	0	1	0		0		0	
1	1	1	1	0	0		1		1	
1	1	1	1	1	0		1		1	

Функциональная схема реализации ОЧС на мультиплексорах приведена в приложении Г.

5. ЛОГИЧЕСКИЙ СИНТЕЗ ПРЕОБРАЗОВАТЕЛЯ МНОЖИТЕЛЯ

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

Таблица 5.1 – Таблица истинности ПМ

Вх. диада		***Мл. бит***	***Пер.***	***Знак***	***Вых. диада***	
Q_n	Q_{n-1}	Q_{n-2}	***P***	***S***	S_1	S_2
0	0	0	0	0	0	0
0	0	1	0	0	0	1
0	1	0	0	0	0	1
0	1	1	0	0	1	0
1	0	0	0	0	1	0
1	0	1	1	1	0	1
1	1	0	1	1	0	1
1	1	1	1	0	0	0

Минимизацию переключательных функций проведём с помощью карт Карно. Заполненные карты приведены на рисунках 5.1.1 – 5.1.4.

Для функции P:

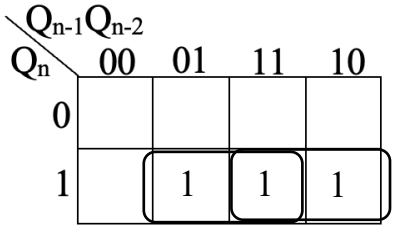

Рисунок 5.1.1 – Минимизация функции P при помощи карты Карно

$P_{МДНФ} = Q_n(Q_{n-2} + Q_{n-1})$

Для функции S:

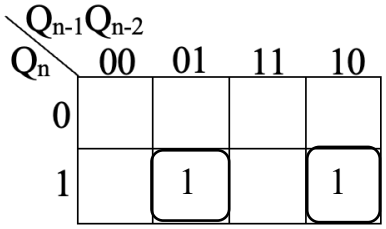

Рисунок 5.1.2 – Минимизация функции S при помощи карты Карно

$S_{МДНФ} = Q_n(\overline{Q_{n-1}}Q_{n-2} + Q_{n-1}\overline{Q_{n-2}})$

Для функции S_1:

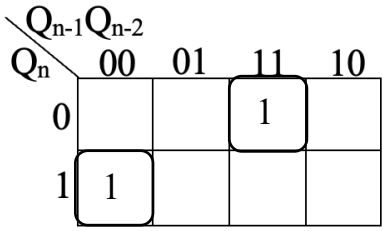

Рисунок 5.1.3 – Минимизация функции S_1 при помощи карты Карно

$S_{1МДНФ} = Q_n\overline{Q_{n-1}}\,\overline{Q_{n-2}} + \overline{Q_n}Q_{n-1}Q_{n-2}$

Для функции S_2:

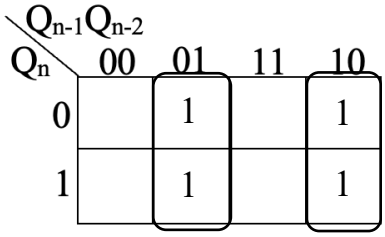

Рисунок 5.1.4 – Минимизация функции S_2 при помощи карты Карно

$S_{2МДНФ} = \overline{Q_{n-1}}Q_{n-2} + Q_{n-1}\overline{Q_{n-2}}$

Эффективность минимизации можно оценить отношением числа входов схем, реализующих переключательную функцию до и после минимизации:

$K_P = \frac{3*3+3+3}{2+2} = 3{,}4$

$K_S = \frac{2*3+2+3}{2+2+2} = 1{,}2$

$K_{S1} = \frac{2*3+2+3}{2*3+2+3} = 1$

$K_{S2} = \frac{4*3+4+3}{2*2+2+2} = 2{,}4$

Функциональная схема ПМ приведена в приложении Д.

6. ВРЕМЕННЫЕ ЗАТРАТЫ НА УМНОЖЕНИЕ

Формула расчёта временных затрат на умножение:
$T_{\text{УМН}} = n * (t_{\text{ПМ}} + t_{\text{ФДК}} + t_{\text{ОЧУС}} + (m + 1) * t_{\text{ОЧС}} + t_{\text{сдвига}})$, где
$t_{\text{ПМ}}$ – время преобразования множителя;
$t_{\text{ФДК}}$ – время формирования дополнительного кода множимого;
$t_{\text{ОЧУС}}$ – время умножения на ОЧУС;
$t_{\text{ОЧС}}$ – время формирования единицы переноса в ОЧС;
$t_{\text{сдвига}}$ – время сдвига частичной суммы;
n – количество разрядов множителя;
m – количество разрядов множимого.

ЗАКЛЮЧЕНИЕ

В результате выполнения курсовой работы была разработана структурная схема сумматора-умножителя, функциональные схемы основных узлов сумматора-умножителя в заданном логическом базисе, что помогло сформировать навыки практической реализации устройств посредством логического синтеза. В целях уменьшения стоимости логических схем была выполнена минимизация переключательных функций по каждому выходу схем при помощи алгоритма извлечения Рота, а также карт Карно – Вейча.

Главным достоинством алгоритма Рота является полная формализация действий на всех этапах минимизации функции. Однако необходимо отметить, что в случае, если функция зависит от большого числа переменных, процесс минимизации может занять длительный промежуток времени.

Применение карт Карно – Вейча является крайне эффективным при небольшом количестве переменных, предоставляя простой, эффективный и быстрый способ решения поставленной задачи минимизации. Тем не менее, как и в случае с алгоритмом извлечения, необходимость работы с большим количеством переменных фактически лишает метод указанных достоинств.

Построение функциональных схем основных узлов спроектированного устройства в различных базисах позволило закрепить знания основных правил и законов булевой алгебры.

Синтез комбинационной схемы устройстве на основе мультиплексоров потребовало для каждой переключательной функции отдельного мультиплексора, однако значительно помогло упростить функциональную схему.

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) Искра, Н. А. Арифметические и логические основы вычислительной техники: пособие / Н. А. Искра, И. В. Лукьянова, Ю. А. Луцик. – Минск: БГУИР, 2016. – 75 с.
3) Лысиков, Б. Г. Цифровая вычислительная техника / Б. Г. Лысиков. – Минск : Выш. шк., 2003. – 242 с.
4) Единая система конструкторской документации (ЕСКД) : справ. пособие / С. С. Борушек [и др.]. – М. : Изд-во стандартов, 1989. – 352 с.
5) Основные требования к текстовым документам (ГОСТ 2.105–95) [Электронный ресурс]. – 2014 – Режим доступа : http://graph.power.nstu.ru/ wolchin/umm/eskd/eskd/GOST/2_105.htm.

ПРИЛОЖЕНИЕ А

(обязательное)

Сумматор-умножитель второго типа. Схема электрическая структурная

ПРИЛОЖЕНИЕ Б

(обязательное)

Одноразрядный четверичный умножитель-сумматор. Схема электрическая функциональная

ПРИЛОЖЕНИЕ В

(обязательное)

Одноразрядный четверичный сумматор. Схема электрическая функциональная

ПРИЛОЖЕНИЕ Г

(обязательное)

Одноразрядный четверичный сумматор. Реализация на мультиплексорах. Схема электрическая функциональная

ПРИЛОЖЕНИЕ Д

(обязательное)

Преобразователь множителя. Схема электрическая функциональная