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

_____ Д. В. Куприянова

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
к курсовой работе
на тему

ПРОЕКТИРОВАНИЕ И ЛОГИЧЕСКИЙ СИНТЕЗ СУММАТОРА-УМНОЖИТЕЛЯ ДВОИЧНО-ЧЕТВЕРИЧНЫХ ЧИСЕЛ

БГУИР КР 1-40 02 01 324 ПЗ

Студент Полховский А. Ф.

Руководитель Куприянова Д. В.

Минск 2020

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
«___» _____2020 г.

ЗАДАНИЕ
по курсовой работе студента
Полховского Артема Федоровича

1 Тема работы: Проектирование и логический синтез сумматора-умножителя двоично-десятичных чисел

2 Срок сдачи студентом законченной работы: 20 мая 2020 г.

3 Исходные данные к работе:

3.1 Исходные сомножители: Мн = 29,71; Мт = 56,49.

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 – 01, 3_4 – 00.

3.5 Тип синтезируемого сумматора - умножителя: 1.

3.6 Логический базис для реализации ОЧС: А5 – ИЛИ НЕ; метод минимизации – алгоритм Рота.

3.7 Логический базис для реализации ОЧУ: А1 – И ИЛИ НЕ; метод минимизации – карты Карно-Вейча.

4 Содержание пояснительной записки (перечень подлежащих разработке вопросов):

Введение. 1 Разработка алгоритма умножения. 2 Разработка структурной схемы сумматора-умножителя. 3 Разработка функциональных схем основным узлов сумматора-умножителя. 4 Синтез комбинационных схем на основе мультиплексоров. 5 Логический синтез преобразователя множителя. 6 Оценка результатов разработки. Заключение. Список использованных источников.

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 сумматор. Схема электрическая функциональная.
5.4 Одноразрядный четверичный сумматор. Реализация на мультиплексорах. Схема электрическая функциональная.
5.5 Преобразователь множителя. Схема электрическая функциональная.

КАЛЕНДАРНЫЙ ПЛАН

Разделы 1, 2 к 1 марта 2020 г. – 20 %;
Раздел 3, 4 к 1 апреля 2020 г. – 50 %;
Разделы 5, 6 к 1 мая 2020 г. – 80 %;
Оформление пояснительной записки и графического материала
к 20 мая 2020 г. – 15 %
Защита курсового проекта с 25 мая 2020 г. по 10 июня 2020 г.

Дата выдачи задания: 14 февраля 2019 г.

Руководитель Д. В. Куприянова

ЗАДАНИЕ ПРИНЯЛ К ИСПОЛНЕНИЮ _____А.Ф. Полховский

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Целью данной курсовой работы является проектирование цифрового устройства двоично-четверичного сумматора-умножителя (СУ). Сумматор является одним из центральных узлов арифметико-логического устройства (АЛУ) вычислительной машины.

Для того чтобы спроектировать данное устройство, необходимо пройти несколько последовательных этапов разработки: разработка алгоритма умножения чисел, по которому работает СУ; разработка структурной схемы; разработка функциональной схемы основных узлов структурной схемы СУ (одноразрядного четверичного умножителя (ОЧУ), одноразрядного четверичного сумматора (ОЧС), а также переключательные функции ОЧС на мультиплексорах); оценка результатов проделанной работы (эффективность минимизации и время выполнения операций). Минимизация перечисленных ранее устройств осуществляется с помощью карт Карно-Вейча и алгоритма извлечения (Рота).

1. РАЗРАБОТКА АЛГОРИТМА УМНОЖЕНИЯ

1.1 Перевод сомножителей из десятичной системы счисления в четверичную.

Множимое

```
29 |__4                    0.71
28  _7|__4                *  4
 1   4   1                 2.84
     3                     *  4
                           3.36
                           *  4
                            1.44
```

$Мн_4 = 131{,}231.$
В соответствии с заданной кодировкой множимого:
$Мн_{2/4} = 110011{,}010011.$

Множитель

```
_56 |__4                   0.49
 56  _14 |__4              *  4
  0   12   3                1.96
      2                     *  4
                            3.84
                            *  4
                            3.36
```

$Мт_4 = 320{,}133.$
В соответствии с обычной весомозначной кодировкой множителя:
$Мт_{2/4} = 111000{,}011111.$

1.2 Запись сомножителей в форме с плавающей запятой в прямом коде:

Мн = 0,110011010011 $Р_{Мн} = 0.1000 + 03_{10}$ – закодировано по заданию,
Мт = 0,111000011111 $Р_{Мт} = 0.0011 + 03_{10}$ – 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 равен:

$P_{Мн}$ = 0.1000 03_4
$P_{Мт}$ = 0.0011 03_4
$P_{Мн \cdot Мт}$ = 0.1101 12_4

Результат закодирован в соответствии с заданием на кодировку множимого.

Знак произведения определяется суммой по модулю “два” знаков сомножителей:

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

Для умножения мантисс необходимо предварительно преобразовать множитель. При умножении чисел в прямых кодах диада $11(3_4)$ заменяется на триаду $1\overline{01}$. Преобразованный множитель имеет вид: $Мт^{п}_4 = 1\bar{1}20201$ или $Мт^{п}_2 = 1\overline{01}100\overline{10}00\overline{1}$. 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.	000000	000000	0.	10 10 10 10 10 10	10 10 10 10 10 10	$\sum_{0}^{ч}$
3.	333333	202103	1.	00 00 00 00 00 00	01 10 01 11 10 00	$П_1^{Ч} = [-Мн]_д$
3.	333333	202103	1.	00 00 00 00 00 00	01 10 01 11 10 00	$\sum_{1}^{ч} = \sum_{2}^{ч}$
0.	000032	312200	0.	10 10 10 10 00 01	00 11 01 01 10 10	$П_3^{Ч} = [2Мн] \cdot 2^2$
0.	000001	120303	0.	10 10 10 10 10 11	01 10 00 10 00 10	$\sum_{3}^{ч} = \sum_{4}^{ч}$
0.	003231	220000	0.	10 10 00 01 00 11	01 01 10 10 10 10	$П_5^{Ч} = [2Мн] \cdot 2^4$
0.	003233	000303	0.	10 10 00 01 00 00	10 10 10 00 10 00	$\sum_{5}^{ч}$
3.	320210	300000	1.	00 01 10 01 11 10	00 10 10 10 10 10	$П_6^{Ч} = [-Мн]_д \cdot 2^5$
3.	330103	300303	1.	00 00 10 11 10 00	00 10 10 00 10 00	$\sum_{6}^{ч}$
0.	131231	000000	0.	11 00 11 01 00 11	10 10 10 10 10 10	$П_7^{Ч} = [Мн] \cdot 2^6$
0.	121330	300303	0.	11 01 11 00 00 10	00 10 10 00 10 00	$\sum_{7}^{ч}$

После окончания умножения необходимо оценить погрешность вычислений. Для этого полученное произведение ($Мн \cdot Мт_4$ = 0, 2210323003030,

$P_{Мн \cdot Мт10}$ = 6) приводится к нулевому порядку, а затем переводится в десятичную систему счисления:

$Мн \cdot Мт_4$ = 121330.300303; $P_{Мн \cdot Мт} = 0$;
$Мн \cdot Мт_{10}$ = 1660.7624

Результат прямого перемножения операндов дает следующее значение:

$Мн_{10} \cdot Мт_{10} = 29{,}71 \cdot 56{,}49 = 1678{,}3179$

Абсолютная погрешность:

$$\Delta = 1678{,}3179 - 1660{,}7624 = 17{,}5555.$$

Относительная погрешность рассчитывается по формуле:

$$\delta = \frac{\Delta}{Мн \cdot Мт}$$

Относительная погрешность равна:

$$\delta = \frac{17{,}5555}{1678{,}3179} = 0{,}01046017563 \quad (\delta = 1{,}04602\%)$$

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

2. РАЗРАБОТКА СТРУКТУРНОЙ СХЕМЫ СУММАТОРА-УМНОЖИТЕЛЯ

Структура первого типа строится на базе заданных узлов ОЧУ, ОЧС и аккумулятора (накапливающего сумматора). Управление режимами работы схемы осуществляется внешним сигналом *mul/sum*, который определяет вид текущей арифметической операции (умножение или суммирование).

Структурная схема сумматора-умножителя первого типа для алгоритма «Б» приведена в приложении А.

Если устройство работает как сумматор (на входе *Mul/sum* – “1”), то оба слагаемых последовательно (за два такта) заносятся в регистр множимого, а на управляющий вход формирователя дополнительного кода (ФДК) F_2 поступает “1”.

Следует учесть, что числа представлены в форме с плавающей запятой, поэтому, прежде чем складывать мантиссы, необходимо выровнять порядки.

В блоке порядков необходимо обеспечить сравнение порядков, используя сумматор порядков, и в зависимости от знака результата сдвигать первое или второе слагаемое.

Реализация сдвига мантиссы числа с меньшим порядком будет производится в регистре множимого.

На выходах ФДК формируется дополнительный код одного из слагаемых с учётом знака. Это слагаемое может быть записано в регистр результата, при этом управляющие сигналы, поступающие на входы *h* всех ОЧУ, дают возможность переписать на выходы ОЧУ разряды слагаемого без изменений (рисунок 2.1).

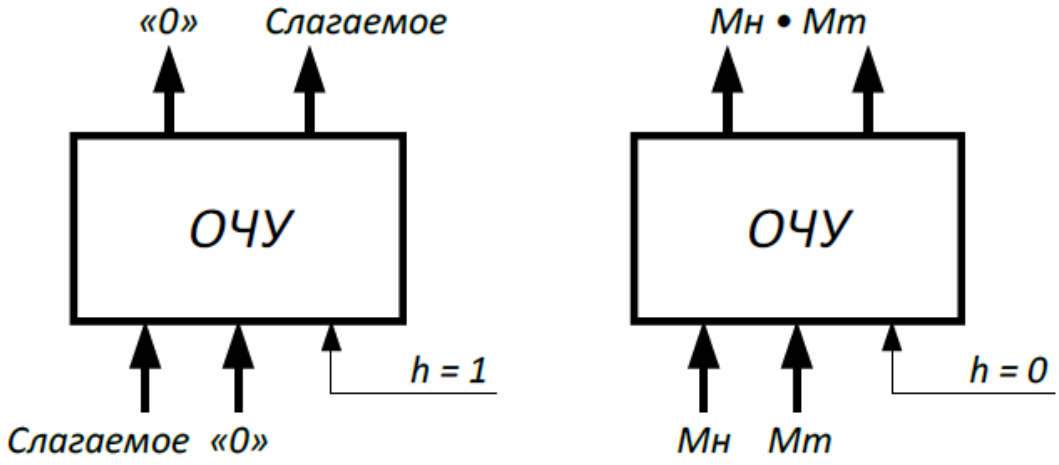

Рисунок 2.1 – Режимы работы ОЧУ

При необходимости выравнивания порядков в регистре-аккумуляторе может выполняться сдвиг мантиссы первого слагаемого. Если на вход поступает “0”, то ОЧУ перемножает разряды Мн и Мт.

Одноразрядный четверичный сумматор предназначен для сложения двух двоично-четверичных цифр, подаваемых на его входы. (рисунок 2.2).

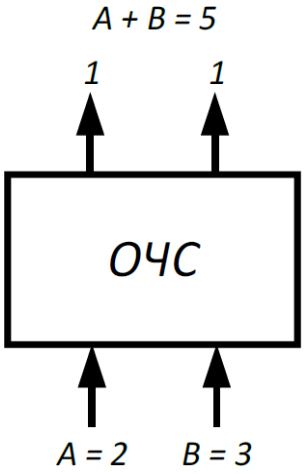

Рисунок 2.2 – Одноразрядный четверичный сумматор

В ОЧС первое слагаемое складывается с нулём, т.к. на старших выходах ОЧУ будут формироваться только коды нуля. Затем первое слагаемое попадает в регистр-аккумулятор, который изначально обнулён.

На втором такте второе слагаемое из регистра множимого через цепочку ОЧУ и ОЧС попадает аккумулятор (накапливающий сумматор) складывает операнды и хранит результат.

Если устройство работает как умножитель (на входе *Mul/sum* - “0”), то множимое и множитель помещаются в соответствующие регистры, а на управляющий вход ФДК F_2 поступает “0”.

Диада множителя поступает на входы преобразователя множителя (ПМ). Задачей ПМ является преобразование диады множителя в соответствии с алгоритмом преобразования. При этом в случае образования единицы переноса в старшую диаду множителя она должна быть учтена при преобразовании следующей старшей диады (выход 1 ПМ), т.е. сохраняться до следующего такта на триггер.

В регистре множителя в конце каждого такта умножения содержимое сдвигается на два двоичных разряда и в последнем такте умножения регистр обнуляется.

Выход 2 ПМ переходит в единичное состояние, если текущая диада содержит отрицание ($\overline{01}$). В этом случае инициализируется управляющий вход *F1* формирователя дополнительного кода (ФДК) и на выходах ФДК формируется дополнительный код множимого с обратным знаком (умножение на “- 1”).

Принцип работы ФДК, в зависимости от управляющих сигналов, приведён в таблице 2.1.

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

На выходах 3 и 4 ПМ формируются диады преобразованного множителя, которые поступают на входы ОЧУ вместе с диадами множимого.

ОЧУ предназначен лишь для умножения двух четверичных цифр. Если в процессе умножения возникает перенос в следующий разряд, необходимо предусмотреть возможность его прибавления.

Для суммирования результата умножения текущей диады Мн·Мт с переносом из предыдущей диады предназначены ОЧС. Следовательно, чтобы полностью сформировать частичное произведение четверичных сомножителей, необходима комбинация цепочек ОЧУ и ОЧС.

Частичные суммы формируются в аккумуляторе. На первом этапе он обнулён и первая частичная сумма получается за счёт сложения первого частичного произведения (сформированного в ОЧС) и нулевой частичной суммы (хранящейся в аккумуляторе).

В аккумуляторе происходит сложение *i*-й с (*i*+1)-м частичным произведением, результат сложения сохраняется.

На четырёх входах ОЧУ формируется результат умножения диад Мн·Мт.

Максимальной цифрой в диаде преобразованного множителя является двойка, поэтому в старшем разряде произведения максимальной цифрой может оказаться только “1”:

$$\underset{\text{Мн}}{3_{max}} * \underset{\text{Мт}}{2_{max}} = \underline{1}2$$

Это означает, что на младшие входы ОЧС никогда не поступят диады цифр, соответствующие кодам «2» и «3», следовательно, в таблице истинности работы ОЧС будут содержаться 16 безразличных входных наборов.

Частичные суммы хранятся только в аккумуляторе, т.к. алгоритм умножения «Б» не предполагает возможность сдвига сумм в регистр множителя.

Количество тактов умножения определяется разрядностью Мт.

3. РАЗРАБОТКА ФУНКЦИОНАЛЬНЫХ СХЕМ ОСНОВНЫХ УЗЛОВ СУММАТОРА-УМНОЖИТЕЛЯ

3.1 Логический синтез одноразрядного четверичного умножителя

Одноразрядный четверичный умножитель – это комбинационное устройство, имеющее 5 двоичных входов (2 разряда из регистра Мн, 2 разряда из регистра Мт и управляющий вход *h*) и 4 двоичных выхода.

Принцип работы ОЧУ 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 – 01; 3 – 00.

Управляющий вход *h* определяет тип операции:

- «0» – умножение закодированных цифр, поступивших на информационные входы;
- «1» – вывод на выходы без изменения значения разрядов, поступивших из регистра множимого.

В таблице 3.1.1 выделено восемь безразличных наборов, т. к. на входы ОЧУ из разрядов множителя не может поступить код «11».

Таблица 3.1.1 – Таблица истинности ОЧУ

Мн		**Мт**		**Упр.**	**Старшие разряды**		**Младшие разряды**		**Пример операции в четверичной с/с**
x_1	x_2	y_1	y_2	h	P_1	P_2	P_3	P_4	
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
0	0	0	0	0	1	0	1	0	3·0=00
0	0	0	0	1	1	0	0	0	Выход – код «03»
0	0	0	1	0	1	0	0	0	3·1=03
0	0	0	1	1	1	0	0	0	Выход – код «03»
0	0	1	0	0	1	1	0	1	3·2=12
0	0	1	0	1	1	0	0	0	Выход – код «03»
0	0	1	1	0	x	x	x	x	3·3=21
0	0	1	1	1	x	x	x	x	Выход – код «03»
0	1	0	0	0	1	0	1	0	2·0=00
0	1	0	0	1	1	0	0	1	Выход – код «02»
0	1	0	1	0	1	0	0	1	2·1=02
0	1	0	1	1	1	0	0	1	Выход – код «02»
0	1	1	0	0	1	1	1	0	2·2=10
0	1	1	0	1	1	0	0	1	Выход – код «02»
0	1	1	1	0	x	x	x	x	2·3=12
0	1	1	1	1	x	x	x	x	Выход – код «02»
1	0	0	0	0	1	0	1	0	0·0=00

Продолжение таблицы 3.1.1

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

Минимизацию переключательный функций проведём с помощью карт Вейча.

Для функции P_1 заполненная карта приведена на рисунке 3.1.1, где символом «х» отмечены наборы, на которых функция может принимать произвольное значение.

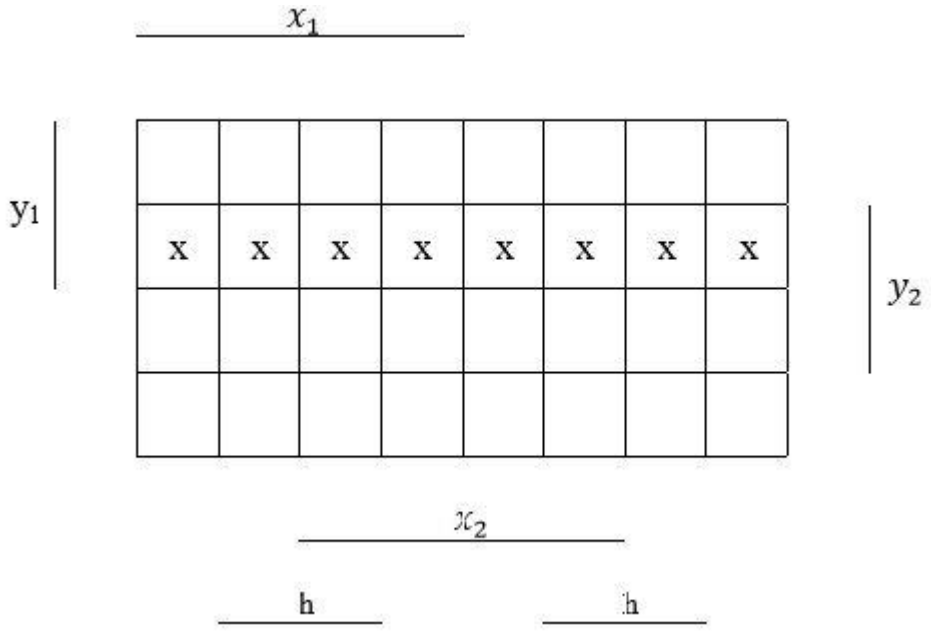

Рисунок 3.1.1 Минимизация функции P_1 при помощи карты Вейча

$P_{1МКНФ} = 0$

Для функции P_2 заполненная карта приведена на рисунке 3.1.2

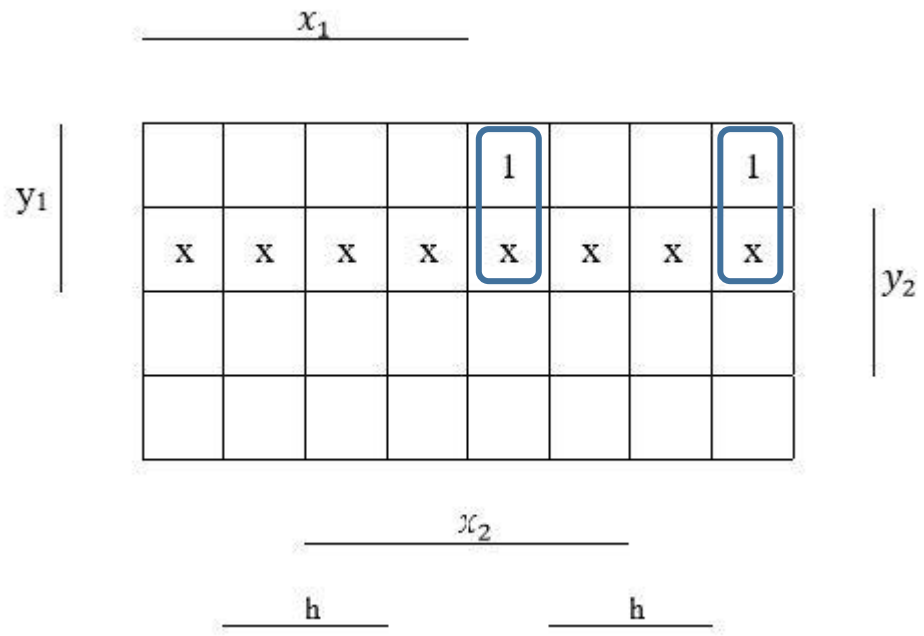

Рисунок 3.1.2 Минимизация функции P_2 при помощи карты Вейча

$P_{2МДНФ} = \overline{x_1} y_1 \bar{h}$

Для функции P_3 заполненная карта приведена на рисунке 3.1.3

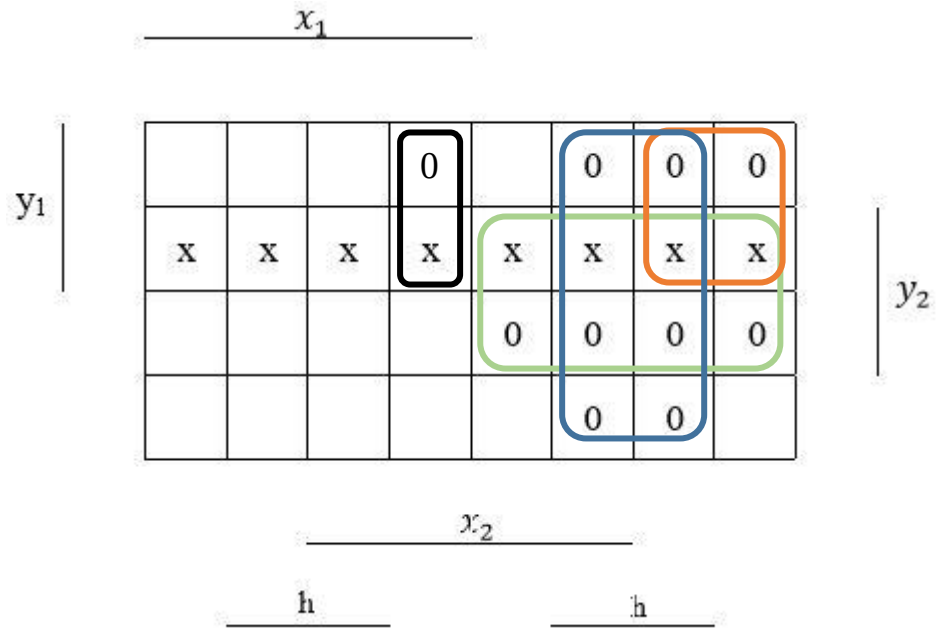

Рисунок 3.1.3 Минимизация функции P_3 при помощи карты Вейча

$P_{3МКНФ} = (\overline{x_1} \vee y_2)(\overline{x_1} \vee h)(\overline{x_1} \vee \overline{x_2} \vee y_1)(x_1 \vee x_2 \vee y_1 \vee \bar{h})$

Для функции P_4 заполненная карта приведена на рисунке 3.1.4

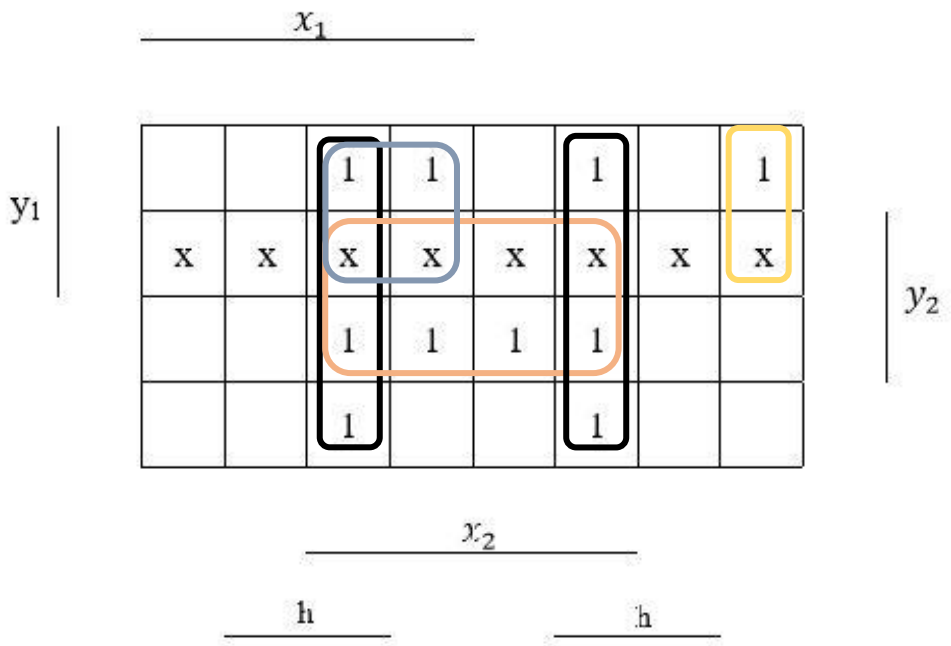

Рисунок 3.1.4 Минимизация функции P_4 при помощи карты Вейча

$P_{4МДНФ} = x_2y_2 \vee x_2h \vee x_1x_2y_1 \vee \overline{x_1x_2}y_1\bar{h}$.

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

$$K_{P1} = 0$$

$$K_{P2} = \frac{2 * 5 + 2 + 2}{3 + 1 + 2} = 2{,}33$$

$$K_{P3} = \frac{10 * 5 + 10 + 5}{11 + 4 + 1 + 3} = 3{,}42$$

$$K_{P4} = \frac{10 * 5 + 10 + 5}{11 + 4 + 3} = 3{,}61$$

Функциональная схема для ОЧУ приведена в приложении В. Функции для реализации в заданном базисе (А1) будут иметь следующий вид:

$$P_{1МКНФ} = 0$$

$$P_{2МДНФ} = \overline{x_1}y_1\bar{h}$$

$$P_{3МКНФ} = (\overline{x_1} \vee y_2)(\overline{x_1} \vee h)(\overline{x_1} \vee \overline{x_2} \vee y_1)(x_1 \vee x_2 \vee y_1 \vee \bar{h})$$

$$P_{4МДНФ} = x_2y_2 \vee x_2h \vee x_1x_2y_1 \vee \overline{x_1x_2}y_1\bar{h}$$

3.2 Логический синтез одноразрядного четверичного сумматора

Одноразрядный четверичный сумматор – это комбинационное устройство, имеющее 5 входов (2 разряда одного слагаемого, 2 разряда второго слагаемого и вход переноса) и 3 выхода (старший и младший разряды и перенос). Принцип работы ОЧС представлен с помощью таблицы истинности (таблица 3.2.1).

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

В таблице 3.2.1 ОЧС синтезируется для схемы первого типа. В таблице истинности необходимо выделить 16 безразличных наборов, т. к. со старших выходов ОЧУ не могут прийти коды «2» и «3».

Таблица 3.2.1 – 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	x	x	x	3+3+0=12
0	0	0	0	1	x	x	x	3+3+1=13
0	0	0	1	0	x	x	x	3+2+0=11
0	0	0	1	1	x	x	x	3+2+1=12
0	0	1	0	0	0	0	0	3+0+0=03
0	0	1	0	1	1	1	0	3+0+1=10
0	0	1	1	0	1	1	0	3+1+0=10
0	0	1	1	1	1	1	1	3+1+1=11
0	1	0	0	0	x	x	x	2+3+0=11
0	1	0	0	1	x	x	x	2+3+1=12
0	1	0	1	0	x	x	x	2+2+0=10
0	1	0	1	1	x	x	x	2+2+1=11
0	1	1	0	0	0	0	1	2+0+0=02
0	1	1	0	1	0	0	0	2+0+1=03
0	1	1	1	0	0	0	0	2+1+0=03
0	1	1	1	1	1	1	0	2+1+1=10
1	0	0	0	0	x	x	x	0+3+0=03
1	0	0	0	1	x	x	x	0+3+1=10
1	0	0	1	0	x	x	x	0+2+0=02
1	0	0	1	1	x	x	x	0+2+1=03
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

Продолжение таблицы 3.2.1

1	2	3	4	5	6	7	8	9
1	0	1	1	1	0	0	1	0+1+1=02
1	1	0	0	0	x	x	x	1+3+0=10
1	1	0	0	1	x	x	x	1+3+1=11
1	1	0	1	0	x	x	x	1+2+0=03
1	1	0	1	1	x	x	x	1+2+1=10
1	1	1	0	0	0	1	1	1+0+0=01
1	1	1	0	1	0	0	1	1+0+1=02
1	1	1	1	0	0	0	1	1+1+0=02
1	1	1	1	1	0	0	0	1+1+1=03

Минимизацию переключательных функций проведём с помощью карт Карно.

Для функции S_1 заполненная карта Карно приведена на рисунке 3.2.1, где символом «x» отмечены наборы, на которых функция может принимать произвольное значение.

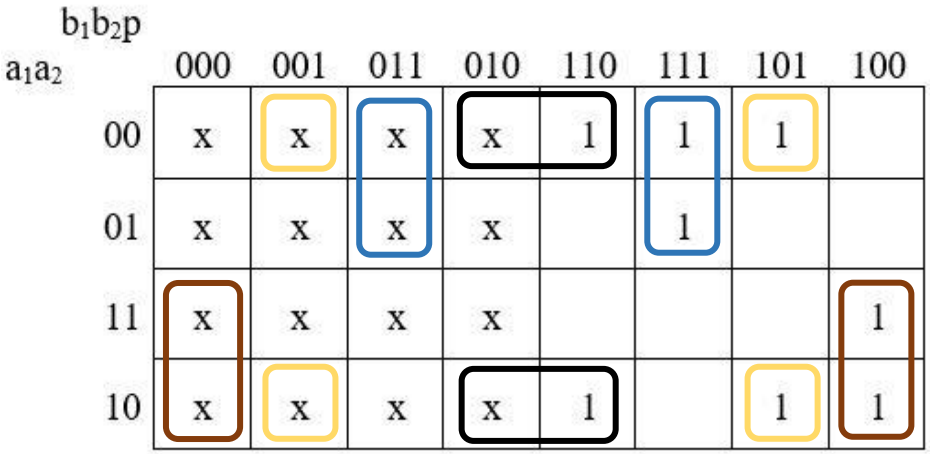

Рисунок 3.2.1 – Минимизация функции S_1 при помощи карты Карно

Результат минимизации примет вид:

$$S_{1\text{МДНФ}} = \overline{a_2} b_2 \bar{p} \vee \overline{a_2}\overline{b_2} p \vee \overline{a_1} b_2 p \vee a_1 \overline{b_2} \bar{p}.$$

Для минимизации функции S_1 также дополнительно воспользуемся алгоритмом Рота.

Определим множество единичных кубов:

$$L = \begin{Bmatrix} 00101, \\ 00110, \\ 00111, \\ 01111, \\ 10100, \\ 10101, \\ 10110, \\ 11100 \end{Bmatrix}$$

Далее определим множество безразличных кубов:

$$N = \begin{Bmatrix} 00000, 00001, 00010, 00011, \\ 01000, 01001, 01010, 01011, \\ 10000, 10001, 10010, 10011, \\ 11000, 11001, 11010, 11011 \end{Bmatrix}$$

В целях упрощения минимизации произведём минимизацию безразличных кубов. Минимизацию безразличных кубов приведём с помощью карты Карно. Для безразличных кубов заполненная карта приведена на рисунке 3.2.2.

x_1x_2 \ y_1y_2h	000	001	011	010	110	111	101	100
00	x	x	x	x				
01	x	x	x	x				
11	x	x	x	x				
10	x	x	x	x				

Рисунок 3.2.2 – Минимизация безразличных кубов функции S_2 при помощи карты Карно.

Множество безразличных кубов после минимизации:

$$N = \{xx0xx\}$$

Сформируем множество $C_0 = L \cup N$:

$$C_0 = \begin{Bmatrix} 00101, 00110, 00111, \\ 01111, 10100, 10101, \\ 10110, 11100, xx0xx \end{Bmatrix}$$

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)* приведен в таблице 3.2.2.

Таблица 3.2.2 – Первый шаг умножения (C_0 * C_0)

*C_0*C_0*	00101	00110	00111	01111	10100	10101	10110	11100	xx0xx
00101	-								
00110		-							
00111	001y1	0011y	-						
01111			0y111	-					
10100					-				
10101	y0101				1010y	-			
10110		y0110			101y0		-		
11100					1y100			-	
xx0xx	00y01	00y10	00y11	01y11	10y00	10y01	10y10	11y00	-
A_1	001x1 x0101 00x01	0011x x0110 00x10	0x111 00x11	01x11	1010x 101x0 1x100 10x00	10x01	10x10	11x00	Ø

Из таблицы 3.2.2 – Поиск простых импликант (C_0 * C_0) следует:

$$A_1 = \begin{Bmatrix} 001x1, x0101, 00x01, \\ 0011x, x0110, 00x10, \\ 0x111, 00x11, 01x11, \\ 1010x, 101x0, 1x100, \\ 10x00, 10x01, 10x10, \\ 11x00 \end{Bmatrix}$$

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

$$Z_0 = \emptyset$$

Сформируем множество B_1:

$$B_1 = C_0 \backslash Z_0 = C_0 \backslash \emptyset = C_0$$

После этой операции сформируется новое множество кубов $C_1 = A_1 \cup B_1$.

$$C_1 = \begin{Bmatrix} 001x1, x0101, 00x01, 0011x, x0110, \\ 00x10, 0x111, 00x11, 01x11, 1010x, \\ 101x0, 1x100, 10x00, 10x01, 10x10, \\ 11x00, xx0xx \end{Bmatrix}$$

В таблице 3.2.3 приведён следующий шаг поиска простых импликант с помощью операции C_1* C_1. Таблица 3.2.3 приведена в приложении Б.

Из таблицы 3.2.2 – Поиск простых импликант ($C_1 * C_1$) следует:

$$A_2 = \begin{Bmatrix} 00xx1, x0x01, 00x1x, x0x10, \\ 0xx11, 10x0x, 10xx0, 1xx00 \end{Bmatrix}$$

Множество Z_1 кубов, не участвовавших в образовании новых кубов, пустое.

$$Z_1 = \emptyset$$

Сформируем множество B_2:

$$B_2 = C_1 \backslash Z_1 = C_1 \backslash \emptyset = C_1$$

$$B_2 = \begin{Bmatrix} 001x1, x0101, 00x01, 0011x, x0110, \\ 00x10, 0x111, 00x11, 01x11, 1010x, \\ 1010x, 101x0, 1x100, 10x00, 10x01, \\ 10x10, 11x00, xx0xx \end{Bmatrix}$$

После этой операции сформируется новое множество кубов $C_2 = A_2 \cup B_2$.

$$C_2 = \begin{Bmatrix} 00xx1, x0x01, 00x1x, x0x10, \\ 0xx11, 10x0x, 10xx0, 1xx00, \\ xx0xx \end{Bmatrix}$$

В таблице 3.2.4 приведён следующий шаг поиска простых импликант с помощью операции C_2* C_2.

Из таблицы 3.2.4 – Поиск простых импликант ($C_2 * C_2$) следует:

$$A_3 = \emptyset$$

$$Z_2 = C_2 = \begin{Bmatrix} 00xx1, x0x01, 00x1x, x0x10, \\ 0xx11, 10x0x, 10xx0, 1xx00 \end{Bmatrix}$$

$$B_3 = C_2 \backslash Z_2 = \emptyset$$

$$C_3 = A_3 \cup B_3 = \emptyset$$

Таблица 3.2.4 – Третий шаг умножения (C_2* C_2)

C_2*C_2	00xx1	x0x01	00x1x	x0x10	0xx11	10x0x	10xx0	1xx00
00xx1	-							
x0x01		-						
00x1x			-					
x0x10				-				
0xx11					-			
10x0x						-		
10xx0							-	
1xx00								-
xx0xx								
A_3	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø

На этом этап поиска простых импликант заканчивается т.к. $C_3 = \{\emptyset\}$. Конечное множество простых импликант:

$$Z = Z_0 \cup Z_1 \cup Z_2 = \begin{Bmatrix} 00xx1, x0x01, 00x1x, x0x10, \\ 0xx11, 10x0x, 10xx0, 1xx00 \end{Bmatrix}$$

Следующий этап – поиск *L*-экстремалей на множестве простых импликант приведен в таблице 3.2.5. Для этого используется операция # (решетчатое вычитание).

Таблица 3.2.5 - Поиск *L*-экстремалей

Z#(Z\z)	00xx1	x0x01	00x1x	x0x10	0xx11	10x0x	10xx0	1xx00	xx0xx
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
00xx1	-	10x01	00x10	x0x10	01x11	10x0x	10xx0	1xx00	1x0xx, x10xx, xx0x0
x0x01	00x11	-	00x10	x0x10	01x11	10x00	10xx0	1xx00	110xx, 1x01x, 1x0x0, x10xx, xx0x0
00x1x	Ø	10x01	-	10x10	01x11	10x00	10xx0	1xx00	110xx, 1x01x, 1x0x0, x10xx, 1x0x0, x10x0, xx000
x0x10	Ø	10x01	Ø	-	01x11	10x00	10x00	1xx00	110xx, 1101x, 1x011, 110x0, 1x000, x10xx, 110x0, 1x000, x10x0, xx000

Продолжение таблицы 3.2.5

1	2	3	4	5	6	7	8	9	10
0xx11	Ø	10x01	Ø	10x10	-	10x00	10x00	1xx00	110xx, 1101x, 1x011, 110x0, 1x000, 110xx, x100x, x10x0, 110x0, 1x000, x10x0, xx000
10x0x	Ø	Ø	Ø	10x10	01x11	-	Ø	11x00	110xx, 1101x, 1x011, 110x0, 11000, 110xx, x100x, x10x0, 110x0, 11000, x10x0, 0x000, x1000
10xx0	Ø	Ø	Ø	Ø	01x11	Ø	-	11x00	110xx, 1101x, 1x011, 110x0, 11000, 110xx, x100x, x10x0, 110x0, 11000, x10x0, 0x000, x1000
1xx00	Ø	Ø	Ø	Ø	01x11	Ø	Ø	-	1101x, 110x1, 1101x, 1x011, 11010, 1101x, 110x1, 0100x, x1001, 010x0, x1010, 11010, 010x0, x1010, 0x000, 01000
xx0xx	Ø	Ø	Ø	Ø	01111	Ø	Ø	11100	-
Остаток	Ø	Ø	Ø	Ø	01111	Ø	Ø	11100	1101x, 110x1, 1101x, 1x011, 11010, 1101x, 110x1, 0100x, x1001, 010x0, x1010, 11010, 010x0, x1010, 0x000, 01000

В таблице 3.2.5 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:

$$E = \begin{Bmatrix} 01111, 11100, 1101x, 110x1, 1101x, 1x011 \\ 11010, 1101x, 110x1, 0100x, x1001, 010x0, \\ x1010, 11010, 010x0, x1010, 0x000, 01000 \end{Bmatrix}$$

Необходимо проверить, нет ли среди полученных L-экстремалей таких, которые стали L-экстремалями за счет безразличных кубов. Для этого в таблице 3.2.6 из кубов множества L вычитаются остатки простых импликант, полученные в таблице 3.2.5 (результат выполнения операции Z#(Z\z)).

Таблица 3.2.6 – Проверка L-экстремалей

Z#(Z\z) n L	00101	00110	00111	01111	10100	10101	10110	11100
01111	Ø	Ø	Ø	01111	Ø	Ø	Ø	Ø
11100	Ø	Ø	Ø	Ø	Ø	Ø	Ø	11100
1101x	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø
110x1	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø
1101x	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø
1x011	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø
11010	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø
1101x	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø
110x1	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø
0100x	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø
x1001	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø
010x0	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø
x1010	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø
11010	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø
010x0	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø
x1010	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø
0x000	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø
01000	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø

Множество L-экстремалей E:

$$E = \{0xx11, 1xx00\}$$

Далее необходимо проанализировать, какие из исходных единичных кубов (множество L) не покрыты найденными L-экстремалями. Этот анализ осуществляется с помощью таблицы 3.2.7.

Таблица 3.2.7 – Поиск непокрытых исходных наборов

L#E	00101	00110	00111	01111	10100	10101	10110	11100
0xx11	00101	00110	Ø	Ø	10100	10101	10110	11100
1xx00	00101	00110	Ø	Ø	Ø	10101	10110	Ø
Остаток	00101	00110	Ø	Ø	Ø	10101	10110	Ø

Из таблицы 3.2.7 видно, что L-экстремалями не покрыто четыре куба из множества L - {00101, 00110, 10101, 10110}. Чтобы их покрыть, воспользуемся множеством $\hat{Z}$ простых импликант, не являющихся L-экстремалями (таблица 3.2.8).

$$\hat{Z} = Z \backslash E = \begin{Bmatrix} 00xx1, x0x01, 00x1x, x0x10, \\ 10x0x, 10xx0, xx0xx \end{Bmatrix}$$

Таблица 3.2.8 – Покрытие оставшихся кубов

$\hat{Z} \cap L$	00101	00110	10101	10110
00xx1	00101	Ø	Ø	Ø
x0x01	00101	Ø	10101	Ø
00x1x	Ø	00110	Ø	Ø
x0x10	Ø	00110	Ø	10110
10x0x	Ø	Ø	10101	Ø
10xx0	Ø	Ø	Ø	10110
xx0xx	Ø	Ø	Ø	Ø

Из таблицы 3.2.8 видно, что непокрытый единичный куб может быть покрыт одним минимальным способом.

Следовательно, существует одна тупиковая (минимальная) форма:

$$S_{1\text{МДНФ}} = \overline{a_2} b_2 \bar{p} \; v \; \overline{a_2}\overline{b_2} p \; v \; \overline{a_1} b_2 p \; v \; a_1 \overline{b_2} \bar{p}.$$

Для функции S_2 проведём минимизацию картой Карно. Заполненная для функции S_2 карта Карно приведена на рисунке 3.2.3.

Результат минимизации примет вид:

$$S_{2\text{МКНФ}} = (\overline{a_1} \; v \; \overline{a_2} \; v \; p)(\overline{a_1} \; v \; b_2 \; v \; \bar{p})(\overline{a_1} \; v \; a_2 v \; p)(a_1 \; v \; \overline{a_2} \; v \; b_2 v \; p) * \\ * \left(a_1 v \; a_2 \; v \; \overline{b_2} \; v \; \bar{p}\right)$$

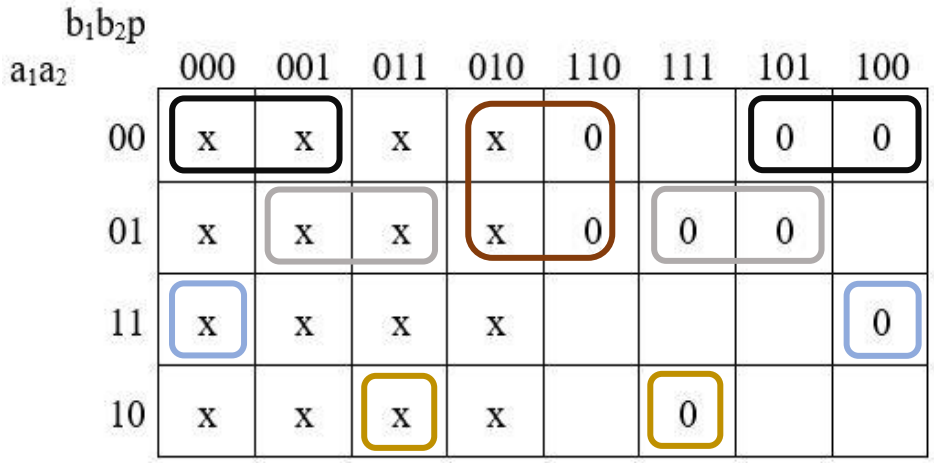

Рисунок 3.2.3 – Минимизация функции S_2 при помощи карты Карно

Для функции П проведём минимизацию картой Карно. Заполненная карта Карно для функции П приведена на рисунке 3.2.4.

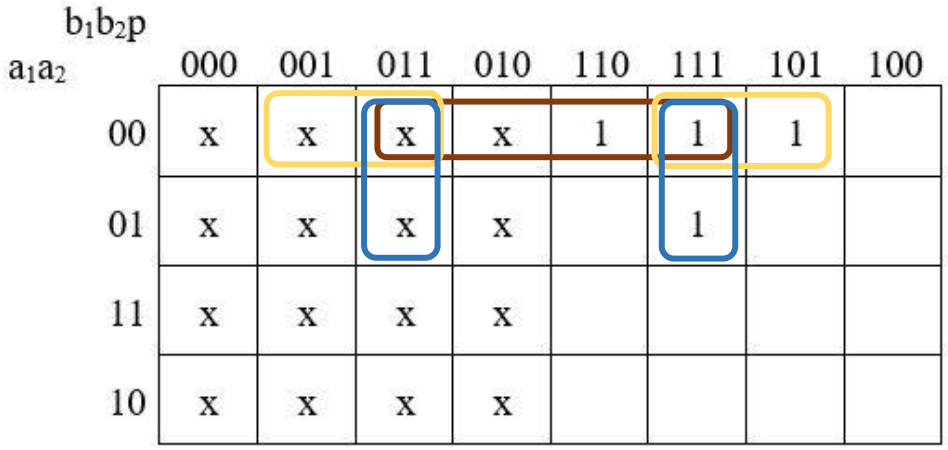

Рисунок 3.2.4 – Минимизация функции П при помощи карты Карно

Результат минимизации примет вид:

$$П = \overline{a_1 a_2} p \vee \overline{a_1 a_2} b_2 \vee \overline{a_1} b_2 p$$

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

$$K_{S1} = \frac{8 * 5 + 8 + 5}{12 + 4 + 1 + 4} = 2{,}79$$

$$K_{S2} = \frac{8 * 5 + 8 + 5}{17 + 5 + 1 + 4} = 1{,}96$$

$$K_{П} = \frac{4 * 5 + 4 + 4}{9 + 3 + 1 + 2} = 1{,}86$$

Функциональная схема для ОЧС приведена в приложении Г. Функции для реализации в заданном базисе (А5) будут иметь следующий вид:

$$S_{1\text{МДНФ}} = \overline{\overline{a_2}b_2\bar{p}}\ v\ \overline{\overline{a_2}b_2}p\ v\ \overline{\overline{a_1}b_2p}\ v\ \overline{a_1\overline{b_2}\bar{p}} = \overline{\overline{a_2}\ v\ \overline{b_2}v\ p}\ v\ \overline{a_2\ v\ b_2v\ \bar{p}}\ v\ v\ \overline{a_1v\ \overline{b_2}\ v\ \bar{p}}\ v\ \overline{\overline{a_1}\ v\ b_2v\ p}$$

$$S_{2\text{МКНФ}} = \overline{\overline{(\overline{a_1}\ v\ \overline{a_2}\ v\ p)(\overline{a_1}\ v\ b_2\ v\ \bar{p})(\overline{a_1}\ v\ a_2v\ p)(a_1v\ \overline{a_2}\ v\ b_2v\ p)} * \overline{* \left(a_1v\ a_2\ v\ \overline{b_2}\ v\ \bar{p}\right)} = \overline{\overline{(\overline{a_1}\ v\ \overline{a_2}\ v\ p)}\ v\ \overline{(\overline{a_1}\ v\ b_2\ v\ \bar{p})}\ v\ \overline{(\overline{a_1}\ v\ a_2v\ p)}\ v\ \overline{(a_1v\ \overline{a_2}\ v\ b_2v\ p)}\ v\ \overline{\left(a_1v\ a_2\ v\ \overline{b_2}\ v\ \bar{p}\right)}}}$$

$$\Pi = \overline{a_1a_2}p\ v\ \overline{a_1a_2}b_2\ v\ \overline{a_1}b_2p = \overline{\overline{a_1a_2}\bar{p}}\ v\ \overline{\overline{a_1a_2}b_2}\ v\ \overline{\overline{a_1}b_2p} = \overline{a_1v\ a_2v\ \bar{p}}\ v\ v\ \overline{a_1v\ a_2\ v\ \overline{b_2}}\ v\ \overline{a_1v\ \overline{b_2}\ v\ \bar{p}}$$

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 должно выполняться условие $n = 2^m$.

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

$a_1a_2b_1$	b_2	p	П	П	S_1	S_1	S_2	S_2
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
000	0	0	*	*	*	*	*	*
000	0	1	*		*		*	
000	1	0	*		*		*	
000	1	1	*		*		*	
001	0	0	0	$b_2 v\ p$	0	$b_2 v\ p$	0	$b_2 p$
001	0	1	1		1		0	
001	1	0	1		1		0	
001	1	1	1		1		1	
010	0	0	*	*	*	*	*	*
010	0	1	*		*		*	
010	1	0	*		*		*	
010	1	1	*		*		*	
011	0	0	0	$b_2 p$	0	$b_2 p$	1	$\overline{b_2 p}$
011	0	1	0		0		0	
011	1	0	0		0		0	
011	1	1	1		1		0	
100	0	0	*	*	*	*	*	*
100	0	1	*		*		*	
100	1	0	*		*		*	

Продолжение таблицы 4.1

1	2	3	4	5	6	7	8	9
100	1	1	*		*		*	
101	0	0	0	“0”	1	$\overline{b_2}\ v\ \overline{p}$	0	$b_2 v\ p$
101	0	1	0		1		1	
101	1	0	0		1		1	
101	1	1	0		0		1	
110	0	0	*	*	*	*	*	*
110	0	1	*		*		*	
110	1	0	*		*		*	
110	1	1	*		*		*	
111	0	0	0	“0”	1	$\overline{b_2 p}$	1	$\overline{b_2}\ v\ \overline{p}$
111	0	1	0		0		1	
111	1	0	0		0		1	
111	1	1	0		0		0	

Функциональная схема реализации ОЧС на мультиплексорах приведена в приложении Д.

5. ЛОГИЧЕСКИЙ СИНТЕЗ ПРЕОБРАЗОВАТЕЛЯ МНОЖИТЕЛЯ

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**	
$\mathbf{Q_n}$	$\mathbf{Q_{n-1}}$	$\mathbf{Q_{n-2}}$	**P**	$\mathbf{S_1}$	$\mathbf{S_2}$
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	0	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_1 и S_2 картами Карно. Заполненные карты для переменных P, S_1 и S_2 соответственно, в табл. 5.2, табл. 5.3, табл. 5.4.

Таблица 5.2 – Минимизация функции P

Q_n \ $Q_{n-1}Q_{n-2}$	00	01	11	10
0				
1		1	1	1

$$P_{\text{МДНФ}} = Q_n Q_{n-1} \; v \; Q_n Q_{n-2}$$

Таблица 5.3 – Минимизация функции S_1

Q_n \ $Q_{n-1}Q_{n-2}$	00	01	11	10
0			1	
1	1			

$$S_{1\,\text{МДНФ}} = Q_n \overline{Q}_{n-1} \overline{Q}_{n-2} \; v \; \overline{Q}_n Q_{n-1} Q_{n-2}$$

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

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

Функциональная схема ПМ приведена в приложении Е.

6. ВРЕМЕННЫЕ ЗАТРАТЫ НА УМНОЖЕНИЕ

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

ЗАКЛЮЧЕНИЕ

В процессе выполнения курсовой работы были разработаны алгоритмы выполнения операций умножения и сложения, структурная схема сумматора-умножителя первого типа, а также функциональные схемы основных узлов данного устройства (ОЧС и ОЧУ). 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 выполнения и минимальные затраты времени. Однако применение данного способа для функций с большим количеством переменных будет затруднительно, также он не является формализованным.

Для минимизации подобных функций удобно использовать алгоритм Рота. Его преимуществом является полная формализация алгоритмов минимизации, что даёт возможность проводить минимизацию в автоматическом режиме.

Функциональные схемы были построены в различных логических базисах. Это позволило закрепить теоретические знания основных законов булевой алгебры, например, правило де Моргана. Реализация переключательных функций на основе мультиплексоров позволила облегчить процесс минимизации этих функций.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

[1] Искра, Н. А. Арифметические и логические основы вычислительной техники: пособие / Н. А. Искра, И. В. Лукьянова, Ю. А. Луцик. – Минск: БГУИР, 2016. – 75 с.

[2] Луцик, Ю. А., Лукьянова И. В. – Учебное пособие по курсу "Арифметические и логические основы вычислительной техники". – Минск: БГУИР, 2014 г.

[3] Лысиков, Б. Г. Арифметические и логические основы цифровых автоматов / Б. Г. Лысиков. – Минск : Выш. шк., 1980. – 342 с.

[4] Лысиков, Б. Г. Цифровая вычислительная техника / Б. Г. Лысиков. – Минск : Выш. шк., 2003. – 242 с.

[5] Савельев, А. Я. Прикладная теория цифровых автоматов / А. Я. Са-. – М. : Высш. шк., 1987. – 272 с.

ПРИЛОЖЕНИЕ А

(обязательное)

Сумматор умножитель первого типа. Схема электрическая структурная

ПРИЛОЖЕНИЕ Б

(обязательное)

Таблица 3.2.3 - Поиск простых импликант ($C_1 * C_1$)

ПРИЛОЖЕНИЕ В

(обязательное)

Одноразрядный четверичный умножитель. Схема электрическая функциональная

ПРИЛОЖЕНИЕ Г

(обязательное)

Одноразрядный четверичный сумматор. Схема электрическая функциональная

ПРИЛОЖЕНИЕ Д

(обязательное)

Одноразрядный четверичный сумматор.
Реализация на мультиплексорах.
Схема электрическая функциональная

ПРИЛОЖЕНИЕ Е

(обязательное)

Преобразователь множителя. Схема электрическая функциональная

ПРИЛОЖЕНИЕ Ж

(обязательное)

Ведомость документов