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

РАЗРАБОТКА СУММАТОРА-УМНОЖИТЕЛЯ

БГУИР КР 1-40 02 01 306 ПЗ

Студент Гуринович А. В.

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  
 Гуриновича Андрея Викторовича

1. Тема работы: «Разработка сумматора-умножителя»
2. Срок сдачи студентом законченной работы: 20 мая 2020 г.
3. Исходные данные к работе:
   1. Исходные сомножители: Мн = 18,69; Мт = 54,85.
   2. Алгоритм умножения: А.
   3. Метод умножения: умножение закодированного двоично-четверичного множимого на два разряда двоичного множителя одновременно в прямых кодах.
   4. Коды четверичных цифр множимого для перехода к двоично-четверичной системе кодирования; 04 – 10, 14 – 00, 24 – 01, 34 – 11.
   5. Тип синтезируемого умножителя: 2.
   6. Логический базис для реализации ОЧС: А4; метод минимизации – карты Карно-Вейча.
   7. Логический базис для реализации ОЧУС: А6; метод минимизации алгоритм Рота.
4. Содержание пояснительной записки (перечень подлежащих разработке вопросов): Ведение. Разработка алгоритма умножения. Разработка структурной схемы сумматора-умножителя. Разработка функциональных схем основным узлов сумматора-умножителя. Логический синтез одноразрядного четверичного сумматора на основе мультиплексора. Логический синтез преобразователя множителя. Временные затраты на умножение. Заключение. Список использованных источников.
5. Перечень графического материала:
   1. Сумматор-умножитель второго типа. Схема электрическая структурная.
   2. Одноразрядный четверичный умножитель-сумматор. Схема электрическая функциональная.
   3. Одноразрядный четверичный сумматор. Схема электрическая функциональная.
   4. Одноразрядный четверичный сумматор. Реализация на мультиплексорах. Схема электрическая функциональная.
   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. В. Гуринович

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 4](#_Toc41004989)

[1.РАЗРАБОТКА АЛГОРИТМА УМНОЖЕНИЯ 5](#_Toc41004990)

[1.1. Перевод сомножителей из десятичной системы счисления в 5](#_Toc41004991)

[четверичную. 5](#_Toc41004992)

[1.2. Запись сомножителей в форме с плавающей запятой в прямом коде. …………………………………………………………………………….5](#_Toc41004993)

[1.3. Умножение двух чисел с плавающей запятой на два разряда множителя одновременно в прямых кодах. 5](#_Toc41004994)

[2. РАЗРАБОТКА СТРУКТУРНОЙ СХЕМЫ СУММАТОРА-УМНОЖИТЕЛЯ 8](#_Toc41004995)

[3. РАЗРАБОТКА ФУНКЦИОНАЛЬНЫХ СХЕМ ОСНОВНЫХ УЗЛОВ СУММАТОРА-УМНОЖИТЕЛЯ 10](#_Toc41004996)

[3.1. Логический синтез одноразрядного четверичного умножителя-сумматора 10](#_Toc41004997)

[3.2. Логический синтез одноразрядного четверичного сумматора 24](#_Toc41004998)

[4. ЛОГИЧЕСКИЙ СИНТЕЗ ОДНОРАЗРЯДНОГО ЧЕТВЕРИЧНОГО СУММАТОРА НА ОСНОВЕ МУЛЬТИПЛЕКСОРА 29](#_Toc41004999)

[5. ЛОГИЧЕСКИЙ СИНТЕЗ ПРЕОБРАЗОВАТЕЛЯ МНОЖИТЕЛЯ 31](#_Toc41005000)

[6. ВРЕМЕННЫЕ ЗАТРАТЫ НА УМНОЖЕНИЕ 33](#_Toc41005001)

[ЗАКЛЮЧЕНИЕ 34](#_Toc41005002)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 35](#_Toc41005003)

[ПРИЛОЖЕНИЕ А 36](#_Toc41005004)

[ПРИЛОЖЕНИЕ Б 37](#_Toc41005005)

[ПРИЛОЖЕНИЕ В 38](#_Toc41005006)

[ПРИЛОЖЕНИЕ Г 39](#_Toc41005007)

[ПРИЛОЖЕНИЕ Д 40](#_Toc41005008)

[ПРИЛОЖЕНИЕ Е 41](#_Toc41005009)

[ПРИЛОЖЕНИЕ Ж 42](#_Toc41005010)

[ПРИЛОЖЕНИЕ И 43](#_Toc41005011)

# 

# ВВЕДЕНИЕ

Целью данной курсовой работы является проектирование цифрового устройства двоично-четверичного сумматора-умножителя (CУ). 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-сумматора (ОЧУС), 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. **. Перевод сомножителей из десятичной системы счисления в**

**четверичную.**

Множимое

**\_**18 | 4 0.69

16 **\_**4 | 4 \* 4

2 4 1 2.76

0 \* 4

3.04

\* 4

0.16

Мн4 = 102,230.

В соответствии с заданной кодировкой множимого:

Мн2/4 =001001,011110.

Множитель

**\_**54 | 4 0.85

52 **\_**13 | 4 \* 4

2 12 ­­ 3 3.40

1 \* 4

1.60

\* 4

2.40

Мт4 = 312,312.

В соответствии с обычной весомозначной кодировкой множителя:

Мт2/4 = 110110,110110.

**1.2. Запись сомножителей в форме с плавающей запятой в прямом**

**коде.**

Мн = 0,001001011110 РМн = 0.1011 + 0310 – закодировано по заданию, Мт = 0,110110110110 РМт = 0.0011 + 0310 – 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:

РМн = 0.1011 034

РМт = 0.0011 034

РМн∙Мт = 0.0001 124

Результат закодирован в соответствии с заданием на кодировку множимого.

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

зн Мн ⊕ зн Мт = 0 ⊕ 0 = 0.

Для умножения мантисс необходимо предварительно преобразовать множитель. При умножении чисел в прямых кодах диада 11(34) заменяется на триаду 1. Преобразованный множитель имеет вид: Мтп4 = 1212 или Мтп2= 01100110.

Перемножение мантисс по алгоритму “А” приведено в табл. 1.1.

Таблица 1.1 ­­– Перемножение мантисс

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Четверичная с/с** | | | **Двоично-четверичная с/с** | | | **Комментарии** |
| **1** | | | **2** | | | **3** |
| 0. | 000000 |  | 0. | 10 10 10 10 10 10 |  | ∑0 ч |
| 0. | 211120 |  | 0. | 01 00 00 00 01 10 |  | П1Ч= Мн ∙ 2 |
| 0. | 211120 |  | 0. | 01 00 00 00 01 10 |  | ∑1 ч |
| 0. | 021112 | 0 | 0. | 10 01 00 00 00 01 | 10 | ∑1 ч∙ 4-1 |
| 0. | 102230 |  | 0. | 00 10 01 01 11 10 |  | П2Ч =Мн ∙ 1 |
| 0. | 130002 | 0 | 0. | 00 11 10 10 10 01 | 10 | ∑2 ч |
| 0. | 013000 | 20 | 0. | 10 00 11 10 10 10 | 01 10 | ∑2 ч∙ 4-1 |
| 3. | 231110 |  | 1. | 01 11 00 00 00 10 |  | П3Ч = [-Мн]д |
| 3. | 310110 | 20 | 1. | 11 00 10 00 00 10 | 01 10 | ∑3 ч |
| 3. | 331011 | 020 | 1. | 11 11 00 10 00 00 | 10 01 10 | ∑3 ч∙ 4-1 |
| 3. | 231110 |  | 1. | 01 11 00 00 00 10 |  | П4Ч= [-Мн]д |
| 3. | 222121 | 020 | 1. | 01 01 01 00 01 00 | 10 01 10 | ∑4 ч |
| 3. | 322212 | 1020 | 1. | 11 01 01 01 00 01 | 00 10 01 10 | ∑4 ч∙ 4-1 |
| 0. | 211120 |  | 0. | 01 00 00 00 01 10 |  | П5Ч= Мн ∙ 2 |
| 3. | 133332 | 1020 | 1. | 00 11 11 11 11 01 | 00 10 01 10 | ∑5 ч |
| 0. | 013333 | 21020 | 0. | 10 00 11 11 11 11 | 01 00 10 01 10 | ∑5 ч ∙ 4-1 |
| 3. | 231110 |  | 1. | 01 11 00 00 00 10 |  | П6Ч= [-Мн]д |
| 3. | 311103 | 21020 | 1. | 11 00 00 00 10 11 | 01 00 10 01 10 | ∑6 ч |
| 3. | 331110 | 321020 | 1. | 11 11 00 00 00 10 | 11 01 00 10 01 10 | ∑6 ч∙ 4-1 |
| 0. | 102230 |  | 0. | 00 10 01 01 11 10 |  | П7Ч = Мн ∙ 1 |
| 3. | 100000 | 321020 | 1. | 00 10 10 10 10 10 | 11 01 00 10 01 10 | ∑7 ч |
| 0. | 010000 | 0321020 | 0. | 10 00 10 10 10 10 | 10 11 01 00 10 01 10 | ∑7 ч∙ 4-1 |

После окончания умножения необходимо оценить погрешность вычислений. Для этого полученное произведение (Мн ∙ Мт4 = 0,10000321020, РМн ∙ Мт = 6) приводится к нулевому порядку, а затем переводится в десятичную систему счисления:

Мн ∙ Мт4 = 100000, 321020; РМн∙Мт = 0;

Мн ∙ Мт10 = 1024,8925.

Результат прямого перемножения операндов дает следующее значение:

Мн10 · Мт10 = 18,68 · 54,84 = 1024,4112.

Абсолютная погрешность:

Δ = 1024,8925 – 1024,4112 = 0,4813.

Относительная погрешность:

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

# 2. РАЗРАБОТКА СТРУКТУРНОЙ СХЕМЫ СУММАТОРА-УМНОЖИТЕЛЯ

Структура второго типа строится на базе заданных узлов ОЧУС и ОЧС. Управление режимами работы обеих схем осуществляется внешним сигналом *mul/sum*, который определяет вид текущей арифметической операции (умножение или суммирование).

Структурная схема сумматора-умножителя второго типа для алгоритма

умножения «А» приведена в приложении 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». Необходимо обеспечить выполнение алгоритма сложения чисел, представленных в форме с плавающей запятой, базируясь на схеме умножителя, реализующего заданный алгоритм умножения.

Первое слагаемое переписывается в регистр результата под действием управляющих сигналов, поступающих на входы *h* всех ОЧУС (рисунок 1).

Если на вход *h* поступает «0», то ОЧУС перемножает разряды Мн и Мт и добавляет к полученному результату перенос из предыдущего ОЧУС (см. рисунок 2.1).

Мн ∙ Мт + перенос

Слагаемое

“0”

ОЧУС

ОЧУС

Перенос

“0”

h = 1

h = 0

Слагаемое “0”

Мн Мт

Рис. 2.1 – Режимы работы ОЧУС

В ОЧС первое слагаемое складывается с нулём, записанным в регистре

результата, переписывается без изменений в регистр результата.

На втором такте второе слагаемое из регистра множимого через цепочку ОЧУС попадает на входы ОЧС и складывается с первым слагаемым, хранящимся в регистре результата. Сумма хранится в регистре результата. Разрядность регистра результата должна быть на единицу больше, чем разрядность исходных слагаемых, чтобы предусмотреть возможность возникновения при суммировании переноса.

*Если устройство работает как умножитель*,то множимое и множительпомещаются в соответствующие регистры, а на управляющий вход ФДК *F*2 поступает «0».

Диада множителя поступает на входы преобразователя множителя. Единица переноса в следующую диаду, если она возникает, должна быть добавлена к следующей диаде множителя (выход 1 ПМ).

В регистре множителя после каждого такта умножения содержимое сдвигается на два двоичных разряда и в конце умножения регистр обнуляется. Это позволяет использовать регистр множителя для хранения младших разрядов произведения при умножении по алгоритму «А».

Выход 2 ПМ переходит в единичное состояние, если текущая диада содержит отрицание (). В этом случае инициализируется управляющий вход F1 формирователя дополнительного кода и на выходах ФДК формируется дополнительный код множимого с обратным знаком (умножение на «– 1»).

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

Таблица 2.1 – Режимы работы формирователя дополнительного кода -

|  |  |  |
| --- | --- | --- |
| Сигналы на входах ФДК | | Результат на выходах ФДК |
| F1 | F2 |
| 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 ОЧУС вместе с диадами множимого. На трёх выходах ОЧУС формируется результат умножения диад Мн·Мт плюс перенос из предыдущего ОЧУС. Максимальной цифрой в диаде преобразованного множителя является двойка, поэтому перенос, формируемый ОЧУС, может быть только двоичным:

3 ∙ 2 = 1 2

max max max

Мн Мт перенос

Так как на входы ОЧУС из регистра Мт не могут поступить коды “3”, в таблице истинности работы ОЧУС будут содержаться 16 безразличных входных наборов.

Частичные произведения, получаемые на выходах ОЧУС, складываются с накапливаемой частичной суммой из регистра результата с помощью цепочки ОЧС (на первом такте выполняется сложение с нулём).

Частичные суммы хранятся в регистре результата и регистре множителя, т. к. умножения «А» предполагает возможность синхронного сдвига этих регистров. Количество тактов умножения определяется разрядностью Мт.

# 3. РАЗРАБОТКА ФУНКЦИОНАЛЬНЫХ СХЕМ ОСНОВНЫХ УЗЛОВ СУММАТОРА-УМНОЖИТЕЛЯ

## **3.1. Логический синтез одноразрядного четверичного умножителя-сумматора**

Одноразрядный четверичный умножитель-сумматор — это комбинационное устройство, имеющее шесть входов (два разряда из регистра Мн, два разряда из регистра Мт, вход переноса и управляющий вход *h*) и три выхода.

Принцип его работы описывается таблицей истинности (таблица 3.1.1).

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

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

Таблица 3.1.1 – Таблица истинности ОЧУС

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Пер.** | **Мн** | | **Мт** | | **Упр.** | **Перенос** | **Результат** | | **Результат операции в четверичной с/c** |
| ***Р1*** | ***x1*** | ***x2*** | ***y1*** | ***y2*** | ***h*** | ***P*** | ***Q1*** | ***Q2*** |
| **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 ∙ 0 + 0 = 00 |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | Выход 01 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 ∙ 1 + 0 = 01 |
| 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | Выход 01 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 ∙ 2 + 0 = 02 |
| 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | Выход 01 |
| 0 | 0 | 0 | 1 | 1 | 0 | x | x | x | 1 ∙ 3 + 0 = 03 |
| 0 | 0 | 0 | 1 | 1 | 1 | x | x | x | Выход 01 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 2 ∙ 0 + 0 = 00 |
| 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | Выход 02 |
| 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 2 ∙ 1 + 0 = 02 |
| 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | Выход 02 |
| 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 2 ∙ 2 + 0 = 10 |
| 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | Выход 02 |
| 0 | 0 | 1 | 1 | 1 | 0 | x | x | x | 2 ∙ 3 + 0 = 12 |
| 0 | 0 | 1 | 1 | 1 | 1 | x | x | x | Выход 02 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 ∙ 0 + 0 = 00 |
| 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | Выход 00 |
| 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 ∙ 1 + 0 = 00 |
| 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | Выход 00 |
| 0 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 ∙ 2 + 0 = 00 |
| 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | Выход 00 |
| 0 | 1 | 0 | 1 | 1 | 0 | x | x | x | 0 ∙ 3 + 0 = 00 |
| 0 | 1 | 0 | 1 | 1 | 1 | x | x | x | Выход 00 |
| 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 3 ∙ 1 + 0 = 03 |
| 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | Выход 03 |
| 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 3 ∙ 1 + 0 = 03 |
| 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | Выход 03 |

*Продолжение таблицы 3.1.1*

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
| 0 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 3 ∙ 2 + 0 = 12 |
| 0 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | Выход 03 |
| 0 | 1 | 1 | 1 | 1 | 0 | x | x | x | 3 ∙ 3 + 0 = 21 |
| 0 | 1 | 1 | 1 | 1 | 1 | x | x | x | Выход 03 |
| 1 | 0 | 0 | 0 | 0 | 0 | x | x | x | 1 ∙ 0 + 1 = 01 |
| 1 | 0 | 0 | 0 | 0 | 1 | x | x | x | Выход 01 |
| 1 | 0 | 0 | 0 | 1 | 0 | x | x | x | 1 ∙ 1 + 1 = 02 |
| 1 | 0 | 0 | 0 | 1 | 1 | x | x | x | Выход 01 |
| 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 ∙ 2 + 1 = 03 |
| 1 | 0 | 0 | 1 | 0 | 1 | x | x | x | Выход 01 |
| 1 | 0 | 0 | 1 | 1 | 0 | x | x | x | 1 ∙ 3 + 1 = 10 |
| 1 | 0 | 0 | 1 | 1 | 1 | x | x | x | Выход 01 |
| 1 | 0 | 1 | 0 | 0 | 0 | x | x | x | 2 ∙ 0 + 1 = 01 |
| 1 | 0 | 1 | 0 | 0 | 1 | x | x | x | Выход 02 |
| 1 | 0 | 1 | 0 | 1 | 0 | x | x | x | 2 ∙ 1 + 1 = 03 |
| 1 | 0 | 1 | 0 | 1 | 1 | x | x | x | Выход 02 |
| 1 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 2 ∙ 2 + 1 = 11 |
| 1 | 0 | 1 | 1 | 0 | 1 | x | x | x | Выход 02 |
| 1 | 0 | 1 | 1 | 1 | 0 | x | x | x | 2 ∙ 3 + 1 = 13 |
| 1 | 0 | 1 | 1 | 1 | 1 | x | x | x | Выход 02 |
| 1 | 1 | 0 | 0 | 0 | 0 | x | x | x | 0 ∙ 0 + 1 = 01 |
| 1 | 1 | 0 | 0 | 0 | 1 | x | x | x | Выход 00 |
| 1 | 1 | 0 | 0 | 1 | 0 | x | x | x | 0 ∙ 1 + 1 = 01 |
| 1 | 1 | 0 | 0 | 1 | 1 | x | x | x | Выход 00 |
| 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 ∙ 2 + 1 = 01 |
| 1 | 1 | 0 | 1 | 0 | 1 | x | x | x | Выход 00 |
| 1 | 1 | 0 | 1 | 1 | 0 | x | x | x | 0 ∙ 3 + 1 = 01 |
| 1 | 1 | 0 | 1 | 1 | 1 | x | x | x | Выход 00 |
| 1 | 1 | 1 | 0 | 0 | 0 | x | x | x | 3 ∙ 0 + 1 = 01 |
| 1 | 1 | 1 | 0 | 0 | 1 | x | x | x | Выход 03 |
| 1 | 1 | 1 | 0 | 1 | 0 | x | x | x | 3 ∙ 1 + 1 = 10 |
| 1 | 1 | 1 | 0 | 1 | 1 | x | x | x | Выход 03 |
| 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 3 ∙ 2 + 1 = 13 |
| 1 | 1 | 1 | 1 | 0 | 1 | x | x | x | Выход 03 |
| 1 | 1 | 1 | 1 | 1 | 0 | x | x | x | 3 ∙ 3 + 1 = 22 |
| 1 | 1 | 1 | 1 | 1 | 1 | x | x | x | Выход 03 |

Управляющий вход h определяет тип операции: 0 – умножение закодированных цифр, поступивших на информационные входы, и добавление переноса; 1 – вывод на выходы без изменения значения разрядов, поступивших из регистра множимого.

В табл. 3.1.1 выделено 36 безразличных наборов, так как 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 функций *P* и *Q1* проведем при помощи карт Карно, где символом «x» отмечены наборы, на которых функция может принимать произвольное значение (безразличные наборы). Минимизацию функции *Q2* проведем с помощью алгоритма Рота.

Для функции *P* заполненная карта Карно приведена на рисунке 3.1.1:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | y1y2h | 000 | 001 | 011 | 010 | 110 | 111 | 101 | 100 |  |
| P1x1x2 |
|  | 000 | 0 | 0 | 0 | 0 | x | x | 0 | 0 |  |
|  | 001 | 0 | 0 | 0 | 0 | x | x | 0 | 1 |  |
|  | 011 | 0 | 0 | 0 | 0 | x | x | 0 | 1 |  |
|  | 010 | 0 | 0 | 0 | 0 | x | x | 0 | 0 |  |
|  | 110 | x | x | x | x | x | x | x | 0 |  |
|  | 111 | x | x | x | x | x | x | x | 1 |  |
|  | 101 | x | x | x | x | x | x | x | 1 |  |
|  | 100 | x | x | x | x | x | x | x | 0 |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |

Рисунок 3.1.1 – Минимизация функции *P* при помощи карты Карно

Для функции *Q1* заполненная карта Карно приведена на рисунке 3.1.2.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | y1y2h | 000 | 001 | 011 | 010 | 110 | 111 | 101 | 100 |  |
| P1x1x2 |
|  | 000 | 1 | 0 | 0 | 0 | x | x | 0 | 0 |  |
|  | 001 | 1 | 0 | 0 | 0 | x | x | 0 | 1 |  |
|  | 011 | 1 | 1 | 1 | 1 | x | x | 1 | 0 |  |
|  | 010 | 1 | 1 | 1 | 1 | x | x | 1 | 1 |  |
|  | 110 | x | x | x | x | x | x | x | 0 |  |
|  | 111 | x | x | x | x | x | x | x | 1 |  |
|  | 101 | x | x | x | x | x | x | x | 0 |  |
|  | 100 | x | x | x | x | x | x | x | 0 |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |

Рисунок 3.1.2 – Минимизация функции *Q1* при помощи карты Карно

Для минимизации функции *Q2* воспользуемся алгоритмом Рота.

Определим множество единичных кубов:

Далее определим множество безразличных кубов:

Минимизацию безразличных кубов приведём с помощью карты Карно. Для безразличных кубов заполненная карта приведена на рисунке 3.1.3.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | y1y2h | 000 | 001 | 011 | 010 | 110 | 111 | 101 | 100 |  |
| P1x1x2 |
|  | 000 |  |  |  |  | x | x |  |  |  |
|  | 001 |  |  |  |  | x | x |  |  |  |
|  | 011 |  |  |  |  | x | x |  |  |  |
|  | 010 |  |  |  |  | x | x |  |  |  |
|  | 110 | x | x | x | x | x | x | x |  |  |
|  | 111 | x | x | x | x | x | x | x |  |  |
|  | 101 | x | x | x | x | x | x | x |  |  |
|  | 100 | x | x | x | x | x | x | x |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |

Рисунок 3.1.3 – Минимизация безразличных кубов функции *Q2* при помощи карты Карно.

*Множество безразличных кубов после минимизации:*

Сформируем множество:

Первым этапом алгоритма Рота является нахождение множества про-стых импликант. Для реализации этого этапа будем использовать операцию умножения (\*) над множествами *C0*, *C1* и т.д., пока в результате операции будут образовываться новые кубы большей размерности.

Первый шаг умножения (*\** ) приведен в таблице 3.1.4.

Таблица 3.1.4 – Поиск простых импликант ().

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| C0\*C0 | 000100 | 001001 | 001010 | 001011 | 001101 | 011001 | 011010 | 011011 | 011100 | 011101 | 100100 | 101100 | 111100 | 1xx0xx | xxx11x | 1xxxx1 |
| 000100 | - |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 001001 |  | - |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 001010 |  |  | - |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 001011 |  | 0010y1 | 00101y | - |  |  |  |  |  |  |  |  |  |  |  |  |
| 001101 |  | 001y01 |  |  | - |  |  |  |  |  |  |  |  |  |  |  |
| 011001 |  | 0y1001 |  |  |  | - |  |  |  |  |  |  |  |  |  |  |
| 011010 |  |  | 0y1010 |  |  |  | - |  |  |  |  |  |  |  |  |  |
| 011011 |  |  |  | 0y1011 |  | 0110y1 | 01101y | - |  |  |  |  |  |  |  |  |
| 011100 |  |  |  |  |  |  |  |  | - |  |  |  |  |  |  |  |
| 011101 |  |  |  |  | 0y1101 | 011y01 |  |  | 01110y | - |  |  |  |  |  |  |
| 100100 | y00100 |  |  |  |  |  |  |  |  |  | - |  |  |  |  |  |
| 101100 |  |  |  |  |  |  |  |  |  |  | 10y100 | - |  |  |  |  |
| 111100 |  |  |  |  |  |  |  |  | y11100 |  |  | 1y1100 | - |  |  |  |
| 1xx0xx |  | y01001 | y01010 | y01011 |  | y11001 | y11010 | y11011 |  |  | 100y00 | 101y00 | 111y00 | - |  |  |
| xxx11x | 0001y0 |  | 001y10 | 001y11 | 0011y1 |  | 011y10 | 011y11 | 0111y0 | 0111y1 | 1001y0 | 1011y0 | 1111y0 | 1xxy1x | - |  |
| 1xxxx1 |  | y01001 |  | y01011 | y01101 | y11001 |  | y11011 |  | y11101 | 10010y | 10110y | 11110y |  |  | - |
| A1 | x00100 0001x0 | 0010x1 001x01 0x1001 x01001 x01001 | 00101x 0x1010 x01010 001x10 | 0x1011 x01011 001x11 x01011 | 0x1101 0011x1 x01101 | 0110x1 011x01 x11001 x11001 | 01101x x11010 011x10 | x11011 011x11 x11011 | 01110x x11100 0111x0 | 0111x1 x11101 | 10x100 100x00 1001x0 10010x | 1x1100 101x00 1011x0 10110x | 111x00 1111x0 11110x | 1xxx1x | Ø | Ø |

Из таблицы 3.1.5 – Поиск простых импликант () следует:

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

Сформируем множество :

После этой операции сформируется новое множество *.*

В таблице 3.1.5 приведён следующий шаг поиска простых импликант с помощью операции *C1\* C1*. Таблице 3.1.5 приведена в приложении Б.

Из таблицы 3.1.5 – Поиск простых импликант () следует:

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

Сформируем множество :

После этой операции сформируется новое множество кубов .

В таблице 3.1.6 приведён следующий шаг поиска простых импликант с помощью операции *C2\* C2*. Таблица 3.1.6 приведена в приложении В.

Из таблицы 3.1.6 – Поиск простых импликант () следует:

Сформируем множество :

После этой операции сформируется новое множество кубов *.*

В таблице 3.1.7 приведён следующий шаг поиска простых импликант с помощью операции *C3\* C3*.

Таблица 3.1.7 – Поиск простых импликант ().

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| C3\*C3 | 0x1xx1 | x01xx1 | xx10x1 | xx1x01 | xx101x | 0x1x1x | x01x1x | xx1x10 | xx1x11 | xx11x1 | x11xx1 | x11x1x | x111xx | 10xxx0 | 10x1xx | 10xx0x | 100xxx | 1x1xx0 | 1x11xx | 1x1x0x | 101xxx | 111xxx |
| 0x1xx1 | - |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| x01xx1 |  | - |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| xx10x1 |  |  | - |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| xx1x01 |  |  |  | - |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| xx101x |  |  |  |  | - |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 0x1x1x |  |  |  |  |  | - |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| x01x1x |  |  |  |  |  |  | - |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| xx1x10 |  |  |  |  |  |  |  | - |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| xx1x11 |  |  |  | xx1xy1 |  |  |  | xx1x1y | - |  |  |  |  |  |  |  |  |  |  |  |  |  |
| xx11x1 |  |  | xx1yx1 |  |  |  |  |  |  | - |  |  |  |  |  |  |  |  |  |  |  |  |
| x11xx1 |  | xy1xx1 |  |  |  |  |  |  |  |  | - |  |  |  |  |  |  |  |  |  |  |  |
| x11x1x |  |  |  |  |  |  | xy1x1x |  |  |  |  | - |  |  |  |  |  |  |  |  |  |  |
| x111xx |  |  |  |  |  |  |  |  |  |  |  |  | - |  |  |  |  |  |  |  |  |  |
| 10xxx0 |  |  |  |  |  |  |  |  |  |  |  |  |  | - |  |  |  |  |  |  |  |  |
| 10x1xx |  |  |  |  |  |  |  |  |  |  |  |  |  |  | - |  |  |  |  |  |  |  |
| 10xx0x |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | - |  |  |  |  |  |  |
| 100xxx |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | - |  |  |  |  |  |
| 1x1xx0 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | - |  |  |  |  |
| 1x11xx |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | - |  |  |  |
| 1x1x0x |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | - |  |  |
| 101xxx |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 10yxxx |  |  |  | - |  |
| 111xxx |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 1y1xxx | - |
| 1xxx1x |  |  |  |  |  | yx1x1x |  |  |  |  |  |  |  |  |  | 10xxyx |  |  |  | 1x1xyx |  |  |
| 1xx0xx |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 10xyxx |  |  |  |  |  |  |  |
| xxx11x |  |  |  |  | xx1y1x |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 1xxxx1 | yx1xx1 |  |  |  |  |  |  |  |  |  |  |  |  | 10xxxy |  |  |  | 1x1xxy |  |  |  |  |
| A4 | xx1xx1 | xx1xx1 | xx1xx1 | xx1xx1 | xx1x1x | xx1x1x | xx1x1x | xx1x1x | Ø | Ø | Ø | Ø | Ø | 10xxxx | 10xxxx | 10xxxx | 10xxxx | 1x1xxx | 1x1xxx | 1x1xxx | 1x1xxx | Ø |

Из таблицы 3.1.7 – Поиск простых импликант (*)* следует:

Сформируем множество :

После этой операции сформируется новое множество куб*ов* .

В таблице 3.1.8 приведён следующий шаг поиска простых импликант с помощью операции *C4\* C4.*

Таблица 3.1.8 – Поиск простых импликант ().

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| C4\*C4 | xx1xx1 | xx1x1x | 10xxxx | 1x1xxx |
| xx1xx1 | - |  |  |  |
| xx1x1x |  | - |  |  |
| 10xxxx |  |  | - |  |
| 1x1xxx |  |  |  | - |
| 1xxx1x |  |  |  |  |
| 1xx0xx |  |  |  |  |
| xxx11x |  |  |  |  |
| 1xxxx1 |  |  |  |  |
| A5 | Ø | Ø | Ø | Ø |

Из таблицы 3.1.8 – Поиск простых импликант (*)* следует:

На этом этап поиска простых импликант заканчивается т.к. .

Конечное множество простых импликант:

Следующий этап – поиск *L*-экстремалей на множестве простых импликант (таблица 3.1.9). Для этого используется операция # (решетчатое вычитание).

Таблица 3.1.9 – Поиск *L*-экстремалей

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| z#(Z-z) | x001x0 | x111xx | xx1xx1 | xx1x1x | 10xxxx | 1x1xxx | 1xxx1x | 1xx0xx | xxx11x | 1xxxx1 |
| x001x0 | - | x111xx | xx1xx1 | xx1x1x | 101xxx 10x0xx 10xxx1 | 1x1xxx | 11xx1x 1x1x1x 1xx01x 1xxx11 | 1xx0xx | x1x11x xx111x xxx111 | 1xxxx1 |
| x111xx | x001x0 | - | x01xx1 xx10x1 | x01x1x xx101x | 101xxx 10x0xx 10xxx1 | 101xxx 1x10xx | 110x1x 11x01x 101x1x 1x101x 1xx01x 10xx11 1x0x11 1xx011 | 1xx0xx | x1011x x0111x x0x111 xx0111 | 10xxx1 1x0xx1 1xx0x1 |
| xx1xx1 | x001x0 | x111x0 | - | x01x10 xx1010 | 101xx0 1000xx 10x0x0 100xx1 | 101xx0 1x10x0 | 110x1x 11001x 11x010 101x10 1x1010 1x001x 1xx010 100x11 1x0x11 1x0011 | 1x00xx 1xx0x0 | x1011x x01110 x00111 xx0111 | 100xx1 1x0xx1 1x00x1 |
| xx1x1x | x001x0 | x11100 | x01x01 xx1001 | - | 101x00 1000xx 1000x0 10x000 100xx1 | 101x00 1x1000 | 110x1x 11001x 110010 1x001x 1x0010 100x11 1x0x11 1x0011 | 1x00xx 1x00x0 1xx000 | x1011x x00111 xx0111 | 100xx1 1x0xx1 1x00x1 |
| 10xxxx | 0001x0 | x11100 | 001x01 0x1001 x11001 | 001x10 0x1010 x11010 | - | 111000 | 110x1x 11001x 110010 11001x 110010 110x11 110011 | 1100xx 1100x0 11x000 | x1011x 000111 0x0111 x10111 | 110xx1 1100x1 |

*Продолжение таблицы 3.1.9*

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1x1xxx | 0001x0 | 011100 | 001x01 0x1001 011001 | 001x10 0x1010 011010 | 1000xx 1000x0 100000 100xx1 | - | 110x1x 11001x 110010 11001x 110010 110x11 110011 | 1100xx 1100x0 110000 | x1011x 000111 0x0111 x10111 | 110xx1 1100x1 |
| 1xxx1x | 0001x0 | 011100 | 001x01 0x1001 011001 | 001x10 0x1010 011010 | 10000x 100000 100000 100x01 | 111000 | - | 11000x 110000 110000 | 01011x 000111 0x0111 010111 | 110x01 110001 |
| 1xx0xx | 0001x0 | 011100 | 001x01 0x1001 011001 | 001x10 0x1010 011010 | 100101 | Ø | 11011x 110111 | - | 01011x 000111 0x0111 010111 | 110101 |
| xxx11x | 000100 | 011100 | 001x01 0x1001 011001 | 001010 0x1010 011010 | 100101 | Ø | Ø | 11000x 110000 110000 | - | 110101 |
| 1xxxx1 | 000100 | 011100 | 001x01 0x1001 011001 | 001010 0x1010 011010 | Ø | Ø | Ø | 110000 110000 110000 | 01011x 000111 0x0111 010111 | - |
| Остаток | 000100 | 011100 | 001x01 0x1001 011001 | 001010 0x1010 011010 | Ø | Ø | Ø | 110000 110000 110000 | 01011x 000111 0x0111 010111 | 110101 |

В таблице 3.1.9 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:

Необходимо проверить, нет ли среди полученных *L*-экстремалей таких, которые стали *L*-экстремалями за счет безразличных кубов. Для этого в таблице 3.1.10 из кубов множества *L* вычитаются остатки простых импликант, полученные в таблице 3.1.9 (результат выполнения операции Z#(Z\z)).

Таблица 3.1.10 – Проверка *L*-экстремалей

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| z#(Z-z) n L | 000100 | 001001 | 001010 | 001011 | 001101 | 011001 | 011010 | 011011 | 011100 | 011101 | 100100 | 101100 | 111100 |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
| 000100 | 000100 | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø |
| 011100 | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | 011100 | Ø | Ø | Ø | Ø |
| 001x01 | Ø | 001001 | Ø | Ø | 001101 | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø |
| 0x1001 | Ø | 001001 | Ø | Ø | Ø | 011001 | Ø | Ø | Ø | Ø | Ø | Ø | Ø |
| 011001 | Ø | Ø | Ø | Ø | Ø | 011001 | Ø | Ø | Ø | Ø | Ø | Ø | Ø |
| 001010 | Ø | Ø | 001010 | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø |
| 0x1010 | Ø | Ø | 001010 | Ø | Ø | Ø | 011010 | Ø | Ø | Ø | Ø | Ø | Ø |
| 011010 | Ø | Ø | Ø | Ø | Ø | Ø | 011010 | Ø | Ø | Ø | Ø | Ø | Ø |
| 110000 | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø |
| 110000 | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø |

*Продолжение таблицы 3.1.10*

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
| 01011x | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø |
| 000111 | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø |
| 0x0111 | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø |
| 010111 | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø |
| 110101 | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø |

Множество L-экстремалей E:

Далее необходимо проанализировать, какие из исходных единичных кубов (множество *L*) не покрыты найденной *L*-экстремалью. Этот анализ осуществляется с помощью таблицы 3.1.11.

Таблица 3.1.11 – Поиск непокрытых исходных наборов

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| L#E | 000100 | 001001 | 001010 | 001011 | 001101 | 011001 | 011010 | 011011 | 011100 | 011101 | 100100 | 101100 | 111100 |
| x001x0 | Ø | 001001 | 001010 | 001011 | 001101 | 011001 | 011010 | 011011 | 011100 | 011101 | Ø | 101100 | 111100 |
| x111xx | Ø | 001001 | 001010 | 001011 | 001101 | 011001 | 011010 | 011011 | Ø | Ø | Ø | 101100 | Ø |
| xx1xx1 | Ø | Ø | 001010 | Ø | Ø | Ø | 011010 | Ø | Ø | Ø | Ø | 101100 | Ø |
| xx1x1x | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | 101100 | Ø |
| Остаток | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | 101100 | Ø |

Из таблицы 3.1.11 видно, что *L*-экстремалью не покрыт один единичный куб (101100). Чтобы их покрыть, воспользуемся множеством простых импликант, не являющихся *L*-экстремалями (таблица 3.1.12).

Таблица 3.1.12 – Покрытие оставшихся кубов

|  |  |
| --- | --- |
| *Ẑ∩L1* | 101100 |
| 10xxxx | 101100 |
| 1x1xxx | 101100 |
| 1xxx1x | Ø |
| 1xx0xx | Ø |
| xxx11x | Ø |
| 1xxxx1 | Ø |

Из таблицы 3.1.12 видно, что непокрытый единичный куб может быть покрыт двумя равнозначными способами.

Следовательно, существуют две тупиковые (минимальные) формы:

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

Эффективность функции *Q2*приведём для первой минимальной формы.

Функции для реализации в заданном базисе (А6) будут иметь следующий вид:

=

Функциональная схема ОЧУC в базисе А6 показана в приложении Г.

## **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. Принцип работы ОЧС представлен с помощью таблицы истинности (табл. 3.2.1).

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

В таблице 3.2.1 ОЧС синтезируется для схемы 2-го типа, поэтому безразличные наборы в таблице истинности отсутствуют.

Таблица 3.2.1 – Таблица истинности ОЧС

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***a1*** | ***a2*** | ***b1*** | ***b2*** | ***p*** | ***П*** | ***S1*** | ***S2*** | **Пример операции в четверичной с/с** |
| **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1+1+0=02 |
| 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1+1+1=03 |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1+2+0=03 |
| 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 1+2+1=10 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1+0+0=01 |
| 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1+0+1=02 |
| 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1+3+0=10 |
| 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1+3+1=11 |
| 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 2+1+0=03 |
| 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 2+1+1=10 |
| 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 2+2+0=10 |
| 0 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 2+2+1=11 |
| 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 2+0+0=02 |
| 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 2+0+1=03 |
| 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 2+3+0=11 |
| 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 2+3+1=12 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0+1+0=01 |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0+1+1=02 |
| 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0+2+0=02 |
| 1 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0+2+1=03 |
| 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0+0+0=00 |
| 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0+0+1=01 |
| 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0+3+0=03 |
| 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0+3+1=10 |
| 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 3+1+0=10 |
| 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 3+1+1=11 |
| 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 3+2+0=11 |
| 1 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 3+2+1=12 |
| 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 3+0+0=03 |
| 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 3+0+1=10 |
| 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 3+3+0=12 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 3+3+1=13 |

Минимизацию функций *П*, S*1* и S*2* проведём с помощью карт Вейча. Заполненные карты для функций *П*, S*1* и S*2* приведены на рисунках 3.2.1, 3.2.2 и 3.2.3 соответственно.

Для функции *П* заполненная карта приведена на рисунке 3.2.1.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |  |  |  |
|  |  | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 |  |  |  |
|  |  | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |  |  |  |
|  |  | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |

Рисунок 3.2.1 – Минимизация функции *П* при помощи карты Вейча

Для функции *S1* заполненная карта приведена на рисунке 3.2.2.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  | 1 |  |  | 1 |  |  |  |  |  |  |
|  |  | 1 | 1 |  | 1 | 1 | 1 | 1 |  |  |  |  |
|  |  | 1 |  |  |  |  | 1 |  |  |  |  |  |
|  |  | 1 | 1 | 1 |  | 1 | 1 |  | 1 |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |

Рисунок 3.2.2 – Минимизация функции *S1* при помощи карты Вейча

Для функции *S2* заполненная карта приведена на рисунке 3.2.3.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | 1 | 1 | 1 |  |  |  | 1 |  |  |  |  |
|  |  | 1 |  |  |  | 1 |  | 1 | 1 |  |  |  |
|  |  |  |  | 1 |  | 1 | 1 | 1 |  |  |  |  |
|  |  | 1 |  | 1 | 1 | 1 |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |

Рисунок 3.2.3 – Минимизация функции *S2* при помощи карты Вейча

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

Функции для реализации в заданном базисе (А4) будут иметь вид:

*=*

*=*

Функциональная схема ОЧC в базисе А4 показана в приложении Д.

# 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* = .

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 ОЧC приведен в таблице 4.1.

Таблица 4.1. – Таблица истинности для ОЧС на мультиплексорах

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **a1a2b1** | **b2** | **p** | **П** | **П** | **S1** | **S1** | **S2** | **S2** |
| **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** |
| 000 | 0 | 0 | 0 | b2p | 0 | b2 + p | 1 |  |
| 000 | 0 | 1 | 0 | 1 | 1 |
| 000 | 1 | 0 | 0 | 1 | 1 |
| 000 | 1 | 1 | 1 | 1 | 0 |
| 001 | 0 | 0 | 0 | b2 | 0 |  | 0 |  |
| 001 | 0 | 1 | 0 | 0 | 1 |
| 001 | 1 | 0 | 1 | 1 | 0 |
| 001 | 1 | 1 | 1 | 0 | 0 |
| 010 | 0 | 0 | 1 | “1” | 1 |  | 1 |  |
| 010 | 0 | 1 | 1 | 1 | 0 |
| 010 | 1 | 0 | 1 | 1 | 0 |
| 010 | 1 | 1 | 1 | 0 | 0 |
| 011 | 0 | 0 | 0 | b2 | 0 |  | 1 |  |
| 011 | 0 | 1 | 0 | 1 | 1 |
| 011 | 1 | 0 | 1 | 0 | 0 |
| 011 | 1 | 1 | 1 | 0 | 1 |

*Продолжение таблицы 4.1*

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** |
| 100 | 0 | 0 | 0 | “0” | 0 | b2p | 0 | b2 + p |
| 100 | 0 | 1 | 0 | 0 | 1 |
| 100 | 1 | 0 | 0 | 0 | 1 |
| 100 | 1 | 1 | 0 | 1 | 1 |
| 101 | 0 | 0 | 0 | b2p | 1 |  | 0 |  |
| 101 | 0 | 1 | 0 | 0 | 0 |
| 101 | 1 | 0 | 0 | 1 | 1 |
| 101 | 1 | 1 | 1 | 1 | 0 |
| 110 | 0 | 0 | 1 | “1” | 1 |  | 0 | b2p |
| 110 | 0 | 1 | 1 | 0 | 0 |
| 110 | 1 | 0 | 1 | 0 | 0 |
| 110 | 1 | 1 | 1 | 0 | 1 |
| 111 | 0 | 0 | 0 | b2 + p | 1 |  | 1 |  |
| 111 | 0 | 1 | 1 | 1 | 0 |
| 111 | 1 | 0 | 1 | 0 | 1 |
| 111 | 1 | 1 | 1 | 1 | 1 |

Функциональная схема реализации ОЧС на мультиплексорах приведена в приложении Е.

# 5. ЛОГИЧЕСКИЙ СИНТЕЗ ПРЕОБРАЗОВАТЕЛЯ МНОЖИТЕЛЯ

Преобразователь множителя (ПМ) служит для исключения из множителя диад 11, заменяя их на триады 10.

Таблица 5.1 - Таблица истинности ПМ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Вх. диада** | | **Мл. бит** | **Зн.** | **Вых. диада** | |
| **Qn** | **Qn-1** | **Qn-2** | **P** | **S1** | **S2** |
| 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 картами Карно.

Таблица 5.2 – Минимизация функции P

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 00 | 01 | 11 | 10 |
| 0 |  |  |  |  |
| 1 |  | 1 | 1 | 1 |

Таблица 5.3 – Минимизация функции

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 00 | 01 | 11 | 10 |
| 0 |  |  | 1 |  |
| 1 | 1 |  |  |  |

Так как минимизировать функцию не удалось запишем её так:

Таблица 5.4 – Минимизация функции

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 00 | 01 | 11 | 10 |
| 0 |  | 1 |  | 1 |
| 1 |  | 1 |  | 1 |

Функциональная схема ПМ приведена в приложении Ж.

# 6. ВРЕМЕННЫЕ ЗАТРАТЫ НА УМНОЖЕНИЕ

Формула расчёта временных затрат на умножение:

, где

– время преобразования множителя;

– время формирования дополнительного кода множимого;

– время умножения на ОЧУC;

– время формирования единицы переноса в ОЧС;

– время сдвига частичной суммы;

*n* – количество разрядов на множителе.

# ЗАКЛЮЧЕНИЕ

В процессе выполнения курсовой работы были разработаны алгоритмы выполнения операций умножения и сложения, структурная схема сумматора-умножителя второго типа, а также функциональные схемы основных узлов данного устройства (ОЧС и ОЧУС). 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, 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. Савельев. – М. : Высш. шк., 1987. – 272 с.

**ПРИЛОЖЕНИЕ А**

*(обязательное)*

Сумматор-умножитель второго типа. Схема электрическая структкрная

**ПРИЛОЖЕНИЕ Б**

*(обязательное)*

Поиск простых импликант ()

**ПРИЛОЖЕНИЕ В**

*(обязательное)*

Поиск простых импликант ()

**ПРИЛОЖЕНИЕ Г**

*(обязательное)*

Одноразрядный четверичный умножитель-сумматор. Схема электрическая функциональная

**ПРИЛОЖЕНИЕ Д**

*(обязательное)*

Одноразрядный четверичный сумматор. Схема электрическая функциональная

**ПРИЛОЖЕНИЕ Е**

*(обязательное)*

Одноразрядный четверичный сумматор.

Реализация на мультиплексорах.

Схема электрическая функциональная

**ПРИЛОЖЕНИЕ Ж**

*(обязательное)*

Преобразователь множителя. Схема электрическая функциональная

**ПРИЛОЖЕНИЕ И**

*(обязательное)*

Ведомость документов