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

\_\_\_\_\_\_\_\_\_\_ И.В. Лукьянова

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе  
на тему

ПРОЕКТИРОВАНИЕ И ЛОГИЧЕСКИЙ СИНТЕЗ СУММАТОРА-УМНОЖИТЕЛЯ ДВОИЧНО-ЧЕТВЕРИЧНЫХ ЧИСЕЛ

БГУИР КР 1-40 02 01 415 ПЗ

Студент

Руководитель

C.К. Матусевич

 И.В. Лукьянова

МИНСК 2021

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

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_2021 г.

ЗАДАНИЕ

по курсовой работе студента  
 Матусевич Семён Климентьевич

1. Тема работы: Проектирование и логический синтез сумматора-умножителя двоично-четверичных чисел.
2. Срок сдачи студентом законченной работы: 1 июня 2021 г.
3. Исходные данные к работе:
   1. исходные сомножители: Мн = 52,26; Мт = 83,31.
   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 – 01, 14 – 11, 24 – 00, 34 – 10.
   5. тип синтезируемого умножителя: структурные схемы приведены для умножителя 1-ого типа (ОЧУ, ОЧС, аккумулятор).
   6. логический базис для реализации ОЧС: ИЛИ, ИСКЛ. ИЛИ, “1”; метод минимизации – алгоритм Рота.
   7. логический базис для реализации ОЧУ: И-НЕ; метод минимизации – карты Карно-Вейча.
4. Содержание пояснительной записки (перечень подлежащих разработке вопросов):

1. Разработка алгоритм 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.

1. Перечень графического материала:
   1. Сумматор-умножитель первого типа. Схема электрическая структурная.
   2. Одноразрядный четвертичный сумматор. Схема электрическая функциональная.
   3. Регистр-аккумулятор. Схема электрическая функциональная.
   4. Одноразрядный четвертичных сумматор. Реализация на мультиплексорах. Схема электрическая функциональная.

КАЛЕНДАРНЫЙ ПЛАН

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование этапов курсовой работы | Объём этапа, % | Срок выполнения этапа | Примечания |
| Разработка алгоритма умножения | 10 | 22.02-13.03 |  |
| Разработка структурной схемы сумматора-умножителя | 10 | 14.03-27.03 | С выполнением чертежа |
| Разработка функциональных схем основных узлов сумматора-умножителя | 50 | 28.03-08.05 | С выполнением чертежей |
| Синтез комбинационных схем устройств на основе мультиплексоров | 10 | 09.05-22.05 | С выполнением чертежа |
| Завершение оформления пояснительной записки | 20 | 23.05-05.06 |  |

Дата выдачи задания: 11 февраля 2021 г.

Руководитель И.В. Лукьянова

ЗАДАНИЕ ПРИНЯЛ К ИСПОЛНЕНИЮ \_\_\_\_\_\_\_\_\_\_

**СОДЕРЖАНИЕ**

Введение ……………………………………………………………………… 4

1. Разработка алгоритма умножения ………………………………………… 6

2. Разработка структурной схемы сумматора-умножителя …………………. 9

3. Логический синтез одноразрядного четверичного сумматора ………….. 12

4. Логический синтез одноразрядного четверичного умножителя………..…21

5. Логический синтез одноразрядного четвертичного сумматора на

основе мультиплексора …………………..………………………………… 23

6. Логический синтез преобразователя множителя ………………………. 24

7. Временные затраты на умножение .……………………………………... 26

Заключение ………………………………………………………………….. 29

Список использованных источников ...………………………………….… 30

Приложение А ………………………………………………………………. 31

Приложение Б ……………………………………………………………….. 32

Приложение В ………………………………………………………………. 33

Приложение Г ………………………………………………………………. 34

Приложение Д ………………………………………………………………. 35

**ВВЕДЕНИЕ**

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

# 

**1. РАЗРАБОТКА АЛГОРИТМА УМНОЖЕНИЯ**

* 1. **Перевод сомножителей из десятичной системы счисления в четверичную.**

Множимое

52 | 4 0.26 Мн4 = 310,100

52 13| 4 4 в соответствии с заданной

0 12 3 1.04 кодировкой множимого

1 4 Мн2/4 = 101101,110101

0.16

4

0.64

Множитель

83 | 4 0.31 Мт4 = 1103,10

80 20 | 4 4

3 20 5 | 4 1.24 Мт2/4 = 01010011,0100

0 4 1 4 *множитель представляется*

1 0.96 *обычным весомозначным*

*кодом:* 04 - 01, 14 - 11, 24 - 00, 34 - 10

для всех вариантов

Запишем сомножители в форме с плавающей запятой в прямом коде:

Мн = 0,101101110101 РМн = 0.0010 + 034 – закодировано по заданию

Мт = 0,010100110100 РМт = 0.0100 + 044 – закодировано традиционно

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 = 0.0010 03

РМт = 0.0100 04

РМн∙Мт = 0.1110 13

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

Таблица 1.1 - Перемножение мантисс

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Четверичная с/c | | | Двоично-четверичная с/с | | | Комментарии |
| **1** | | | **2** | | | **3** |
| 0. | 000000 |  | 01. | 010101010101 |  | ∑0 ч |
| 0. | 000000 |  | 01. | 010101010101 |  | П1Ч=Мн∙0 |
| 0. | 000000 |  | 01. | 010101010101 |  | ∑1 ч |
| 0. | 000000 |  | 01. | 010101010101 | 01 | ∑1 ч∙ 2-2 |
| 0. | 310100 |  | 01. | 101101110101 |  | П2Ч =Мн∙2 |
| 0. | 310100 | 0 | 01. | 101101110101 | 01 | ∑2 ч |
| 0. | 031010 | 00 | 01. | 011011011101 | 0101 | ∑2 ч∙ 2-2 |
| 3. | 023300 |  | 10. | 010010100101 |  | П3Ч =Мн∙(-1) |
| 3. | 120310 | 00 | 10. | 110001101101 | 0101 | ∑3 ч |
| 3. | 312031 | 000 | 10. | 101100011011 | 010101 | ∑3 ч∙ 2-2 |
| 0. | 310100 |  | 01. | 101101110101 |  | П4Ч= Мн∙1 |
| 0. | 222131 | 000 | 01. | 000000110111 |  | ∑4 ч |
| 0. | 022213 | 1000 | 01. | 010000001110 | 11010101 | ∑4 ч∙ 2-2 |
| 0. | 310100 |  | 01. | 101101110101 |  | П5Ч= Мн∙1 |
| 0. | 332313 | 1000 | 01. | 101000101110 | 11010101 | ∑5 ч |
| 0. | 033231 | 31000 | 01. | 011010001011 | 1011010101 | ∑5 ч∙ 2-2 |
| 0. | 310100 |  | 01. | 101101110101 |  | П6Ч= Мн∙1 |
| 1. | 003331 | 31000 | 11. | 010110101011 | 1011010101 | ∑6 ч |
| 0. | 100333 | 131000 | 01. | 110101101010 | 111011010101 | ∑6 ч∙ 2-2 |

После окончания умножения необходимо оценить погрешность вычислений. Для этого полученное произведение (Мн∙Мт4 = 0, 100333131000, РМн∙Мт = 7) приводится к нулевому порядку, а затем переводится в десятичную систему счисления:

Мн∙Мт4 = 1003331.31000 РМн∙Мт = 0;

Мн∙Мт10 = 4349.8125

Результат прямого перемножения операндов дает следующее значение:

Мн10 · Мт10 = 52,26 · 83,31 = 4353,7806.

Абсолютная погрешность:

Δ = 4353,7806– 4349,8125= 3,9681.

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**

Структурная схема сумматора-умножителя первого типа строится на базе заданных узлов ОЧУ, ОЧС, формирователя дополнительного кода, преобразователя множителя и аккумулятора. Управление режимами работы схемы осуществляется внешним сигналом *mul/sum*, который определяет вид текущей арифметической операции (умножение или суммирование)

*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».

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)

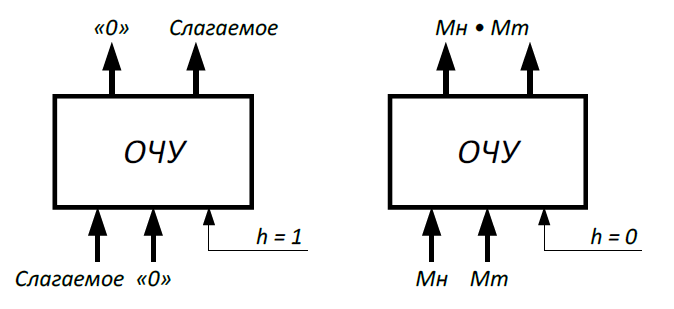
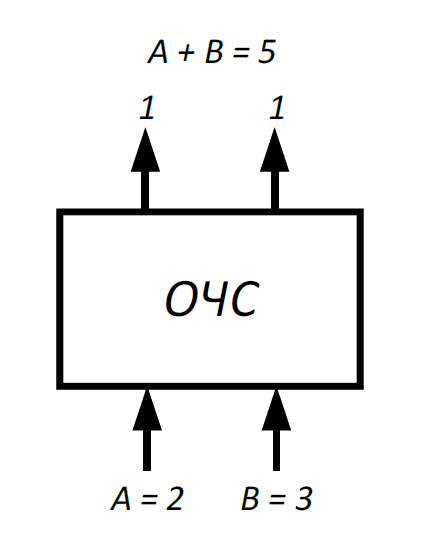


Рисунок 2.1 – Режимы работы ОЧУ

Если на вход *h* 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 в аккумулятор, где складывает с первым слагаемым. Таким образом, аккумулятор (накапливающий сумматор) складывает операнды и хранит результат.

Когда 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. Это позволяет использовать регистр множителя для хранения младших разрядов произведения.

Выход 1 ПМ переходит в единичное состояние, если текущая диада содержит отрицание (). В этом случае инициализируется управляющий вход 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** |
| **F1** | **F2** |
| 0 | 0 | Дополнительный код множимого |
| 0 | 1 | Дополнительный код слагаемого |
| 1 | 0 | Меняется знак множимого |
| 1 | 1 | Дополнительный код слагаемого |

На выходах 2 и 3 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 предназначен 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 на выходах ОЧС) и нулевой частичной суммы (хранящейся в аккумуляторе).

В аккумуляторе происходит сложение *i*-й частичной суммы с (*i*+1)-м частичным произведением, результат сложения сохраняется. Содержимое аккумулятора сдвигается на один четверичный разряд вправо в конце каждого такта умножения по алгоритму «А».

На четырёх выходах 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»:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 3 | · | 2 | = | 1 2 |
| max |  | max |  |  |
| Мн |  | Мт |  |  |

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.

Количество тактов умножения определяется разрядностью Мт.

**3. РАЗРАБОТКА ФУНКЦИОНАЛЬНЫХ СХЕМ ОСНОВНЫХ УЗЛОВ СУММАТОРА-УМНОЖИТЕЛЯ**

**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)

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

В таблице 3.2 выделено 16 безразличных наборов, т.к. со старших выходов ОЧУ не могут прийти коды «2» и «3».

Таблица 3.2 – Таблица истинности ОЧС:

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

Минимизацию функций *П, S1* и *S2*проведем при помощи карт Карно.

Минимизацию функции *S2* проведем с помощью алгоритма Рота

Для функции *П*:



Рисунок 3.2.1 – Минимизация функции *П*при помощи карты Карно.

*=*

Для функции *S1*:



Рисунок 3.2.2 – Минимизация функции *S1* при помощи карты Карно.

*=*

Для функции *S2*:



Рисунок 3.2.3 – Минимизация функции *S2* при помощи карты Карно.

*=*

**Минимизация функции S2 с помощью алгоритма Рота:**

Определим множество единичных кубов

*L* =

и множество безразличных кубов

.

Минимизацию безразличных кубов проведём с помощью карты Карно. Для безразличных кубов заполненная карта приведена на рисунке 3.2.4, где символом «x» отмечены наборы, на которых функция не определена.



Рисунок 3.2.4 – Минимизация безразличных кубов с помощью карты Карно

Множество безразличных наборов после минимизации:

Сформируем множество *С0* = *L* ⋃ *N*:

.

Первым этапом алгоритма Рота является нахождение множества простых импликант.

Для реализации этого этапа будем использовать операцию умножения (\*) над множествами *С0, С1* и т. д., пока в результате операции будут образовываться новые кубы большей размерности.

Первый шаг умножения (*С0\*С0*)приведён в таблице 3.2.5.

Таблица 3.2.5 – Поиск простых импликант (*С0\*С0*)

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *С0\*С0* | 00111 | 01011 | 01010 | 01110 | 11010 | 10011 | 10110 | 10111 |  |
| 00111 | - |  |  |  |  |  |  |  |  |
| 01011 | 0yy11 | - |  |  |  |  |  |  |  |
| 01010 | 0yy1y | 0101y | - |  |  |  |  |  |  |
| 01110 | 0y11y | 01y1y | 01y10 | - |  |  |  |  |  |
| 11010 | yyy1y | y101y | y1010 | y1y10 | - |  |  |  |  |
| 10011 | y0y11 | yy011 | yy01y | yyy1y | 1y01y | - |  |  |  |
| 10110 | y011y | yyy10 | yyy10 | yy110 | 1yy10 | 10y1y | - |  |  |
| 10111 | y0111 | yyy11 | yyy1y | yy11y | 1yy1y | 10y11 | 1011y | - |  |
|  | 001y1 | 010y1 | 010y0 | 011y0 | 110y0 | 100y1 | 101y0 | 101y1 | - |

В результате сформируем новое множество кубов:

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

Для следующего шага получения множества Z формируется множество = ∪ .

В таблице 3.2.6 приведён следующий шаг поиска простых импликант с помощью операции \*.

В результате этой операции образуется множество кубов:

Таблица 3.2.6 – Поиск простых импликант (*С1\*С1*)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| C1\*C1 | x0111 | 001x1 | 0101x | 010x1 | 01x10 | x1010 | 010x0 | 011x0 | 110x0 | 10x11 | 100x1 | 1011x | 101x0 | 101x1 |
| x0111 | - |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 001x1 | 00111 | - |  |  |  |  |  |  |  |  |  |  |  |  |
| 0101x | 0yy11 | 0yy11 | - |  |  |  |  |  |  |  |  |  |  |  |
| 010x1 | 0yy11 | 0yyx1 | 01011 | - |  |  |  |  |  |  |  |  |  |  |
| 01x10 | 0y11y | 0y11y | 01010 | 0101y | - |  |  |  |  |  |  |  |  |  |
| x1010 | xyy1y | 0yy1y | 01010 | 0101y | 01010 | - |  |  |  |  |  |  |  |  |
| 010x0 | 0yy1y | 0yyxy | 01010 | 010xy | 01010 | 01010 | - |  |  |  |  |  |  |  |
| 011x0 | 0y11y | 0y1xy | 01y10 | 01yxy | 01110 | 01y10 | 01yx0 | - |  |  |  |  |  |  |
| 110x0 | 1yy1y | yyyxy | y1010 | y10xy | y1010 | 11010 | y10x0 | y1yx0 | - |  |  |  |  |  |
| 10x11 | 10111 | y0111 | yy011 | yy011 | yyx1y | 1y01y | yy01y | yy11y | 1y01y | - |  |  |  |  |
| 100x1 | 10y11 | y0yx1 | yy011 | yy0x1 | yy01y | 1y01y | yy0xy | yyyxy | 1y0xy | 10011 | - |  |  |  |
| 1011x | 10111 | y0111 | yyy1x | yyy11 | yy110 | 1yy10 | yyy10 | yy110 | 1yy10 | 10111 | 10y11 | - |  |  |
| 101x0 | 1011y | y01xy | yyy10 | yyyxy | yy110 | 1yy10 | yyyx0 | yy1x0 | 1yyx0 | 1011y | 10yxy | 10110 | - |  |
| 101x1 | 10111 | y01x1 | yyy11 | yyyx1 | yy11y | 1yy1y | yyyxy | yy1xy | 1yyxy | 10111 | 10yx1 | 10111 | 101xy | - |
| xxx0x | x01y1 | 00101 | 010yx | 01001 | 01xy0 | x10y0 | 01000 | 01100 | 11000 | 10xy1 | 10001 | 101yx | 10100 | 10101 |
| A2 | x01x1 | x01x1 | 010xx | 010xx | 01xx0 | x10x0 | 01xx0 x10x0 | Ø | Ø | 10xx1 | 10xx1 | 101xx | 101xx | Ø |

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

Для следующего шага получения множества Z формируется множество = ∪ .

В таблице 3.2.7 приведён следующий шаг поиска простых импликант – операция \*.

Таблица 3.2.7 – Поиск простых импликант (*С1\*С1*)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| C2\*C2 | x01x1 | 010xx | 01xx0 | x10x0 | 10xx1 | 101xx |
| x01x1 | - |  |  |  |  |  |
| 010xx | 0yyx1 | - |  |  |  |  |
| 01xx0 | 0y1xy | 010x0 | - |  |  |  |
| x10x0 | xyyxy | 010x0 | 010x0 | - |  |  |
| 10xx1 | 101x1 | yy0x1 | yyxxy | 1y0xy | - |  |
| 101xx | 101x1 | yyyxx | yy1x0 | 1yyx0 | 101x1 | - |
| xxx0x | x0101 | 0100x | 01x00 | x1000 | 10x01 | 1010x |
| A3 | Ø | Ø | Ø | Ø | Ø | Ø |

Новых кубов (третьей размерности) не образовалось.

Получено множество

Поскольку |C3|≤1, поиск простых импликант заканчивается. Множество простых импликант:

*Z* =*Z0* ⋃ *Z1* ⋃ *Z2 =*

Таблица 3.2.8 Поиск L-экстремалей.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| z#(Z-z) | x01x1 | 010xx | 01xx0 | x10x0 | 10xx1 | 101xx | xxx0x |
| x01x1 | - | 010xx | 01xx0 | x10x0 | 100x1 | 101x0 | x1x0x,xx00x,xxx00 |
| 010xx | x01x1 | - | 011x0 | 110x0 | 100x1 | 101x0 | 11x0x,x110x,1x00x,x000x 1xx00,x0x00,xx100 |
| 01xx0 | x01x1 | 010x1 | - | 110x0 | 100x1 | 101x0 | 11x0x,1110x,x1101,1x00x x000x,1xx00,x0x00,1x100 x0100 |
| x10x0 | x01x1 | 010x1 | 011x0 | - | 100x1 | 101x0 | 1110x,11x01,1110x,x1101 1000x,1x001,x000x,10x00 1x100,x0x00,1x100,x0100 |
| 10xx1 | 001x1 | 010x1 | 011x0 | 110x0 | - | 101x0 | 1110x,11x01,1110x,x1101 10000,11001,0000x,x0000 10x00,1x100,x0x00,1x100 x0100 |
| 101xx | 001x1 | 010x1 | 011x0 | 110x0 | 100x1 | - | 1110x,11x01,1110x,x1101 10000,11001,0000x,x0000 10000,11100,00x00,x0000 11100,00100 |
| xxx0x | 00111 | 01011 | 01110 | 11010 | 10011 | 10110 | - |
| Остаток | 00111 | 01011 | 01110 | 11010 | 10011 | 10110 | 1110x,11x01,1110x,x1101 10000,11001,0000x,x0000 10000,11100,00x00,x0000 11100 00100 |

В таблице 3.2.8 из каждой простой импликанты поочерёдно вычитаются все остальные простые импликанты *Z#(Z-z)*.

Получили кубы, “подозрительные” на *L*-экстремальность. Проверяем в таблице 3.2.9.

Таблица 3.2.9 – Проверка на L-экстремальность

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| z#(Z-z) n L | 00111 | 01011 | 01010 | 01110 | 11010 | 10011 | 10110 | 10111 |
| 00111 | 00111 | Ø | Ø | Ø | Ø | Ø | Ø | Ø |
| 01011 | Ø | 01011 | Ø | Ø | Ø | Ø | Ø | Ø |
| 01110 | Ø | Ø | Ø | 01110 | Ø | Ø | Ø | Ø |
| 11010 | Ø | Ø | Ø | Ø | 11010 | Ø | Ø | Ø |
| 10011 | Ø | Ø | Ø | Ø | Ø | 10011 | Ø | Ø |
| 10110 | Ø | Ø | Ø | Ø | Ø | Ø | 10110 | Ø |
| 1110x | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø |
| 11x01 | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø |
| 1110x | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø |
| x1101 | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø |
| 10000 | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø |
| 11001 | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø |
| 0000x | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø |
| x0000 | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø |
| 10000 | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø |
| 11100 | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø |
| 00x00 | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø |
| x0000 | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø |
| 11100 | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø |
| 00100 | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø |

По результатам таблицы 3.2.9, L-экстремалями стали кубы E.

E =

Далее необходимо проанализировать, какие из исходных единичных кубов не покрыты найденной L-экстремалью. Анализ осуществляется с помощью таблицы 3.2.10

Таблица 3.2.10– Поиск непокрытых наборов

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| L#E | 00111 | 01011 | 01010 | 01110 | 11010 | 10011 | 10110 | 10111 |
| x01x1 | Ø | 01011 | 01010 | 01110 | 11010 | 10011 | 10110 | Ø |
| 010xx | Ø | Ø | Ø | 01110 | 11010 | 10011 | 10110 | Ø |
| 01xx0 | Ø | Ø | Ø | Ø | 11010 | 10011 | 10110 | Ø |
| x10x0 | Ø | Ø | Ø | Ø | Ø | 10011 | 10110 | Ø |
| 10xx1 | Ø | Ø | Ø | Ø | Ø | Ø | 10110 | Ø |
| 101xx | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø |

Из таблицы 3.2.10 видно, что все наборы покрыты.

Следовательно, минимальное покрытие будет выглядеть следующим образом

**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 описывается с помощью таблицы истинности (табл.3.1).

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

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

Таблица 3.1- Таблица истинности ОЧУ



Минимизацию функций *P1, P2*, *P3*, *P4*проведем при помощи карт Вейча.

Для функции *P1*:

 Рисунок 3.1.1 – Минимизация функции *P1* при помощи карты Вейча.

Для функции *P2*:

 Рисунок 3.1.2 – Минимизация функции *P2* при помощи карты Вейча.

Для функции *P3*:



Рисунок 3.1.3 – Минимизация функции *P3* при помощи карты Вейча

Для функции *P4*:

 Рисунок 3.1.4 – Минимизация функции *P4* при помощи карты Вейча

**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 управляющими входами, это позволит упростить одну нашу большую функцию от пяти аргументов до восьми функций от одной переменной.

Функциональная схема ОЧС на базе мультиплексоров приведена на чертеже ГУИР.400201.415 Э2.3



Tаблица 4.1 – таблица истинности ОЧС на базе мультиплексора

**5 ЛОГИЧЕСКИЙ СИНТЕЗ ПРЕОБРАЗОВАТЕЛЯ МНОЖИТЕЛЯ (ПМ)**

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 |

Видно, что не минимизируется, поэтому

Проведём минимизацию при помощи карт Карно:

Таблица *5.3 –* Минимизация функции

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

1. **ОЦЕНКА РЕЗУЛЬТАТОВ РАЗРАБОТКИ**

**6.1 ОЦЕНКА ЭФФЕКТИВНОСТИ МИНИМИЗАЦИИ ПЕРЕКЛЮЧАТЕЛЬНЫХ ФУНКЦИЙ**

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

Таблица 6.1 – Эффективность минимизации ОЧУ

|  |  |  |  |
| --- | --- | --- | --- |
| Вых.  схемы | Рассчитанная цена схемы | | Эфф.  мин. k |
| До минимизации | После минимизации |
| P1 | с=1\*5+5+1=11 | с=5 | 2,2 |
| P2 | с=24\*5+5+24=149 | с=1 | 149 |
| P3 | с=8\*5+5+8=53 | с=2\*2+2=6 | 8,8 |
| P4 | с=14\*5+5+14=89 | c=6+2+2+3=13 | 6,8 |

Таблица 6.2 – Эффективность минимизации ОЧС

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Вых.  схемы | Рассчитанная цена схемы | | | Эфф.  мин. k |
| До минимизации | | После минимизации |
|  | c=4\*5+5+8=33 |  | с=3\*3+3+3=15 | 2,2 |
|  | с=8\*5=53 |  | с=3\*4+4+5=21 | 2,5 |
|  | с=8\*5=53 |  | с=6\*3+7+6=31 | 1,7 |

**6.2 Временные затраты на умножение**

Формула расчёта временных затрат на умножение:

, где

– время сдвига частичной суммы;

– время умножения на ОЧУ;

– время формирования единицы переноса в ОЧС;

– время преобразования множителя;

– время формирования дополнительного кода множимого.

**ЗАКЛЮЧЕНИЕ**

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

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

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

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

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

Луцик Ю.А., Лукьянова И.В. – Методические указания к курсовому проекту по курсу “Арифметические и логические основы вычислительной техники”. – Мн.: БГУИР, 2004 г.

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

Лысиков Б.Г. Арифметические и логические основы цифровых автоматов. Мн.: Вышейшая школа, 1980.

Лысиков Б.Г. Цифровая вычислительная техника. Мн.: 2003 г.

**ПРИЛОЖЕНИЕ А**

*(обязательное)*

Сумматор-умножитель первого типа. Схема электрическая структурная

**ПРИЛОЖЕНИЕ Б**

*(обязательное)*

Одноразрядный четверичный умножитель.

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

**ПРИЛОЖЕНИЕ Е**

*(обязательное)*

Ведомость документов