Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет

информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра вычислительных машин, систем и сетей

Дисциплина: Арифметические и логические основы

цифровых устройств

К ЗАЩИТЕ ДОПУСТИТЬ

\_\_\_\_\_\_\_\_\_\_ Ю. А. Луцик

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе

на тему

ПРОЕКТИРОВАНИЕ И ЛОГИЧЕСКИЙ СИНТЕЗ СУММАТОРА-

УМНОЖИТЕЛЯ ДВОИЧНО-ЧЕТВЕРИЧНЫХ ЧИСЕЛ

БГУИР КР 1-40 02 01 213 ПЗ

Студент А. Ю. Зорин

Руководитель Ю. А. Луцик

МИНСК 2023

Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет

информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра электронных вычислительных машин

Дисциплина: Арифметические и логические основы цифровых устройств

УТВЕРЖДАЮ

Заведующий кафедрой ЭВМ

Б. В. Никульшин

« » 20 г.

ЗАДАНИЕ

по курсовой работе студента

Зорина Андрея Юрьевича

1. Тема работы: «Проектирование и логический синтез сумматора- умножителя двоично-четверичных чисел»
2. Срок сдачи студентом законченной работы: до 20 мая 2023 г.
3. Исходные данные к работе:
   1. исходные сомножители: Мн = 64,57; Мт = 14,13;
   2. алгоритм умножения: Г;
   3. метод умножения: умножение закодированного двоично-четверичного множимого на два разряда двоичного множителя одновременно в дополнительных кодах;
   4. коды четверичных цифр множимого для перехода к двоично-четверичной системе кодирования: 04 – 10, 14 – 01, 24 – 00, 34 – 11;
   5. тип синтезируемого умножителя: 1;
   6. тип синтезируемого умножителя: 1;
   7. логический базис для реализации ОЧС: ИЛИ, исключающее ИЛИ, «1»; метод минимизации – карты Карно – Вейча ­;
   8. логический базис для реализации ОЧУ: ИЛИ–НЕ; метод минимизации – алгоритм Рота.
4. Содержание пояснительной записки (перечень подлежащих разработке вопросов):

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

1. Перечень графического материала:
   1. Сумматор-умножитель первого типа. Схема электрическая структурная.
   2. Одноразрядный четверичный сумматор. Схема электрическая функциональная.
   3. Одноразрядный четверичный умножитель. Схема электрическая функциональная.
   4. Преобразователь множителя. Схема электрическая функциональная
   5. Одноразрядный четверичный сумматор. Реализация на мультиплексорах. Схема электрическая функциональная.

КАЛЕНДАРНЫЙ ПЛАН

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование этапов  курсовой работы | Объем этапа,  % | Срок выполнения этапа | Примечания |
| Разработка алгоритма умножения | 10 | 10.02–20.02 |  |
| Разработка структурной схемы  сумматора-умножителя | 10 | 21.02–09.03 | С выполнением  чертежа |
| Разработка функциональных схем  основных узлов сумматора- умножителя | 50 | 10.03–30.04 | С выполнением чертежей |
| Синтез комбинационных схем устройств на основе мультиплексоров | 10 | 01.05–15.05 | С выполнением чертежа |
| Завершение оформления пояснительной записки | 20 | 15.05–20.05 |  |

Дата выдачи задания: 10 февраля 2023 г.

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_ / Луцик Ю.А. /

ЗАДАНИЕ ПРИНЯЛ К ИСПОЛНЕНИЮ \_\_\_\_\_\_\_\_\_\_\_\_\_\_/ Зорин А.Ю. /

**СОДЕРЖАНИЕ**

ВВЕДЕНИЕ ..............................................................................................................

1. РАЗРАБОТКА АЛГОРИТМА УМНОЖЕНИЯ ................................................

2. РАЗРАБОТКА СТРУКТУРНОЙ СХЕМЫ СУММАТОРА-УМНОЖИТЕЛЯ ....................................................................................................................................

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

3.1. Логический синтез одноразрядного четверичного умножителя..................

3.2. Логический синтез одноразрядного четверичного сумматора.....................

3.3. Логический синтез преобразователя множителя...........................................

4. СИНТЕЗ КОМБИНАЦИОННЫХ СХЕМ УСТРОЙСТВ НА ОСНОВЕ

МУЛЬТИПЛЕКСОРОВ......................................................................................

5. ОЦЕНКА РЕЗУЛЬТАТОВ РАЗРАБОТКИ....................................................... ЗАКЛЮЧЕНИЕ........................................................................................................СПИСОК ЛИТЕРАТУРЫ ......................................................................................

ПРИЛОЖЕНИЕ А Сумматор-умножитель первого типа. Схема

электрическая структурная ....................................................................................

ПРИЛОЖЕНИЕ Б Поиск простых импликант C1 \* C1.........................................

ПРИЛОЖЕНИЕ В Одноразрядный четверичный умножитель. Схема электрическая функциональная.............................................................................

ПРИЛОЖЕНИЕ Г Одноразрядный четверичный сумматор. Схема электрическая функциональная.............................................................................

ПРИЛОЖЕНИЕ Д Преобразователь множителя. Схема электрическая функциональная ………………………….............…………………………........

ПРИЛОЖЕНИЕ Е Одноразрядный четверичный сумматор. Реализация на мультиплексорах. Схема электрическая функциональная.................................

ПРИЛОЖЕНИЕ Ж Ведомость документов……………………………………

**ВВЕДЕНИЕ**

Курсовое проектирование является обязательным элементом подготовки специалиста с высшим образованием и одной из форм текущей аттестации студента по учебной дисциплине. Для студентов это первая работа такого рода и объёма. Она содержит результаты теоретических и экспериментальных исследований по дисциплине “Арифметические и логические основы цифровых устройств”, включает совокупность аналитических, расчётных, экспериментальных заданий и предполагает выполнение конструкторских работ и разработку графической документации.

Целью данной курсовой работы является проектирование такого цифрового устройства, как двоично-четверичный сумматор-умножитель (СУ). Сумматор является одним из центральных узлов арифметико-логического устройства (АЛУ) вычислительной машины, поэтому глубокое понимание принципов его работы критически важно для современного инженера. Для того чтобы спроектировать данное устройство, необходимо пройти несколько последовательных этапов разработки:

* Разработка алгоритма умножения чисел, по которому работает СУ
* Разработка структурной схемы СУ
* Разработка функциональной схемы основных узлов структурной схемы СУ
* Оценка результатов проделанной работы
* Оформление документации по проделанной работе

В ходе выполнения курсовой работы автором были пройдены все эти этапы. В настоящей пояснительной записке изложено краткое описание процесса проектирования и приведена разработанная автором графическая документация по структурной схеме и функциональным схемам основных её узлов.

1. **Разработка алгоритма умножения**
2. Перевод сомножителей из десятичной системы счисления в четверичную.

**Множимое**

|  |
| --- |
| 0,57 |
| 4 |
| 2,28 |
| 4 |
| 1,12 |
| 4 |
| 0,48 |
| 4 |
| 1,92 |

|  |  |  |  |
| --- | --- | --- | --- |
| 64 | 4 |  |  |
| 4 | 16 | 4 |  |
| 24 | 16 | 4 | 4 |
| 24 | 0 | 4 | 1 |
| 0 |  | 0 |  |

Мн4 = 1000,21.

В соответствии с заданной кодировкой множимого:

Мн2/4 = 01101010,0001.

**Множитель**

|  |
| --- |
| 0,13 |
| 4 |
| 0,52 |
| 4 |
| 2,08 |
| 4 |
| 0,32 |
| 4 |
| 1,28 |

|  |  |
| --- | --- |
| 14 | 4 |
| 12 | 3 |
| 2 |  |

Мт4 = 32,0201.

В соответствии с обычной весомозначной кодировкой множителя:

Мт2/4 = 1110,00100001.

2. Запишем сомножители в форме с плавающей запятой в дополнительном коде:

Мн = 0,011010100001 РМн = 0.0110 +0410 – закодировано по заданию,

Мт = 0,111000100001 РМт = 0.0010 +0210 – закодировано традиционно.

3.Порядок произведения будет следующим:

РМн = 0.0110 104

РМт = 0.0010 024

РМн\*Мт = 0.0110 124

Знак произведения определяется суммой по модулю два знаков сомножи-телей, т. е.:

зн Мн ⊕ зн Мт = 0 ⊕ 0 = 0.

При умножении чисел в дополнительных кодах диады 11(34) и 10(24) заменяются на триады и . Преобразованный множитель имеет вид М =.

Перемножение мантисс по алгоритму «Г» представлено в таблице 1.1

Таблица 1.1 — Перемножение мантисс

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Четверичная С/С** | | **Двоично-четверичная С/C** | | **Комментарии** |
| **1** | | **2** | | **3** |
| 0. | 000000 0000000 | 10. | 101010101010 10101010101010 | ∑0ч |
| 0. | 010002 1000000 | 10. | 100110101000 01101010101010 | П1ч = [Мн]д\* 4-1 |
| 0. | 010002 1000000 | 10. | 100110101000 01101010101010 | ∑1ч=∑2ч |
| 3. | 333133 2320000 | 11. | 111111011111 00110010101010 | П3ч =[-2Мн]д \* 4-3 |
| 0. | 003201 3320000 | 10. | 101011001001 11110010101010 | ∑3ч |
| 0. | 000010 0021000 | 10. | 101010100110 10100001101010 | П4ч= [Мн]д \* 4-4 |
| 0. | 003212 0001000 | 10. | 101011000100 10101001101010 | ∑4ч |
| 3. | 333331 3323200 | 11. | 111111111101 11110011001010 | П5ч= [-2Мн]д \* 4-5 |
| 0. | 003203 3330200 | 10. | 101011001011 11111110001010 | ∑5ч=∑6ч |
| 0. | 000000 0100021 | 10. | 101010101010 10011010100001 | П7ч = [Мн]д \* 4-7 |
| 0. | 003210 0030221 | 10. | 101011000110 10101110000001 | ∑7ч |

После окончания умножения необходимо оценить погрешность вычислений. Для этого полученное произведение (Мн4 ∙ Мт4 =0,032100030221, РМн ∙ Мт = 6) приводится к нулевому порядку, а затем переводится в десятичную систему счисления:

Мн4 ∙ Мт4 = 32100,030221 РМн ∙ Мт = 0;

Мн10 ∙ Мт10 = 912,1975.

Результат прямого перемножения операндов дает следующее:

Мн10 ∙ Мт10 = 64,57\* 14,13=912,3741.

Абсолютная погрешность:

Δ =912,3741– 912,1975= 0,1766.

Относительная погрешность:

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

**2. Разработка структурной схемы сумматора-умножителя**

*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 зависеть от используемого алгоритма умножения. Этим будет определяться порядок подачи слагаемых на операцию и то, где будет сдвигаться мантисса (в регистре множимого или в регистре результата).

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).

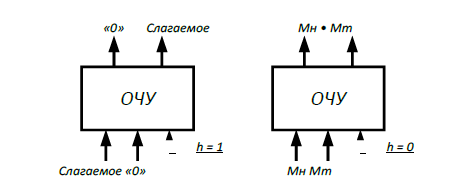


Рисунок 2.1 – Режимы работы ОЧУ

При необходимости выравнивания порядков в регистре-аккумуляторе может выполняться сдвиг мантиссы первого слагаемого. Если на вход *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).

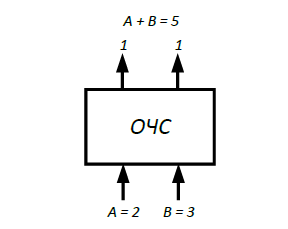


Рисунок 2.2 – Одноразрядный четверичный сумматор

В ОЧС первое слагаемое складывается с нулем, 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 складывается с первым слагаемым. Таким образом, аккумулятор (накапливающий сумматор) складывает операнды и хранит результат.

Разрядность аккумулятора должна быть на единицу больше, чем разрядность исходных слагаемых, чтобы предусмотреть возможность возникновения переноса при суммировании.

*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.

В регистре множителя в конце каждого такта умножения содержимое сдвигается на два двоичных разряда.

Выход 2 ПМ переходит в единичное состояние, если текущая диада содержит отрицание (. В этом случае инициализируется управляющий вход *F*1 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 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 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.

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.

Частичные суммы хранятся в аккумуляторе. Количество тактов умножения определяется разрядностью Мт.

Структурная схема сумматора-умножителя приведена в приложении А.

1. **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.

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

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

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

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

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 | 2·0=00 |
| 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | Выход – код «02» |
| 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 2·1=02 |
| 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | Выход – код «02» |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 2·2=10 |
| 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | Выход – код «02» |
| 0 | 0 | 1 | 1 | 0 | х | х | х | х | 2·3=12 |
| 0 | 0 | 1 | 1 | 1 | х | х | х | х | Выход – код «02» |
| 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1·0=00 |
| 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | Выход – код «01» |
| 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1·1=01 |
| 0 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | Выход – код «01» |
| 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1·2=02 |
| **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
| 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | Выход – код «01» |
| 0 | 1 | 1 | 1 | 0 | х | х | х | х | 1·3=03 |
| 0 | 1 | 1 | 1 | 1 | х | х | х | х | Выход – код «01» |
| 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0·0=00 |
| 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 | х | х | х | х | 0·3=00 |
| 1 | 0 | 1 | 1 | 1 | х | х | х | х | Выход – код «00» |
| 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 3·0=00 |
| 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | Выход – код «03» |
| 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 3·1=03 |
| 1 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | Выход – код «03» |
| 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 3·2=12 |
| 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | Выход – код «03» |
| 1 | 1 | 1 | 1 | 0 | х | х | х | х | 3·3=21 |
| 1 | 1 | 1 | 1 | 1 | х | х | х | х | Выход – код «03» |

Минимизацию переключательных функций проведём с помощью карт Вейча/ Карно. На рисунках 3.1.1 – 3.1.3 символом «х» отмечены наборы, на которых функция может принимать произвольное значение (безразличные наборы)

**Минимизация функции P1:**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | y1y2h |  |  |  |  |  |  |  |  |
| x1x2 |  | 000 | 001 | 011 | 010 | 110 | 111 | 101 | 100 |
|  | 00 |  |  |  |  | x | x |  | 0 |
|  | 01 |  |  |  |  | x | x |  |  |
|  | 11 |  |  |  |  | x | x |  | 0 |
|  | 10 |  |  |  |  | x | x |  |  |

Рисунок 3.1.1 — Минимизация функции Р1 картой Карно

Следовательно:

P1 = (x1 + x2 + + h) \* ( + + +h).

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

*K = =* 1,15.

Запишем функцию P1 в базисе ИЛИ-НЕ:

P1 = (((x1 ↓ x2) ↓ (x1 ↓ x2)) ↓ (((y1 ↓ y1) ↓ h) ↓ ((y1 ↓ y1) ↓ h))) ↓ ((((x1 ↓ x1) ↓ (x2 ↓ x2)) ↓ ((x1 ↓ x1) ↓ (x2 ↓ x2))) ↓ (((y1 ↓ y1) ↓ h) ↓ ((y1 ↓ y1) ↓ h))).

**Минимизация функции P2:**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  | h | |  |  | h | |  |
|  |  |  |  |  | x2 | | | |  |  |
|  |  |  | x1 | | | |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
| y1 |  |  |  |  |  | 1 |  |  |  | 1 |
| y2 |  | x | x | x | x | x | x | x | x |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |

Рисунок 3.1.2 — Минимизация функции P2 картой Вейча

Следовательно:

P2 = x1\*x2\*y1\* + \* \*y1 \* = y1 \* \*(x1\*x2 + \* ).

Эффективность минимизации:

K = *=* 1,33.

Запишем функцию P2 в базисе ИЛИ-НЕ:

P2 = (((y1 ↓ y1) ↓ h) ↓ ((y1 ↓ y1) ↓ h)) ↓ ((x1 ↓ (x2 ↓ x2)) ↓ (x2 ↓ (x1 ↓ x1))) ↓ ((x1 ↓ (x2 ↓ x2)) ↓ (x2 ↓ (x1 ↓ x1))).

**Минимизация функции P3:**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | y1y2h |  |  |  |  |  |  |  |  |
| x1x2 |  | 000 | 001 | 011 | 010 | 110 | 111 | 101 | 100 |
|  | 00 |  | 0 | 0 | 0 | x | x | 0 |  |
|  | 01 |  | 0 | 0 | 0 | x | x | 0 | 0 |
|  | 11 |  |  |  |  | x | x |  | 0 |
|  | 10 |  |  |  |  | x | x |  |  |

Рисунок 3.1.3 — Минимизация функции Р4 картой Карно

Следовательно:

P3 = (x1 + )\*(x1 + )\*(x2 + y1 +).

Эффективность минимизации:

K = *=* 5,42.

Запишем функцию P3 в базисе ИЛИ-НЕ:

P3 = (((x1 ↓(y2 ↓y2)) ↓ (x1 ↓ (h ↓ h))) ↓ ((x1 ↓ (y2 ↓y2)) ↓ (x1 ↓ (h ↓ h)))) ↓ (((x2 ↓ y1) ↓ (x2 ↓ y1)) ↓ (h ↓ h)).

**Минимизация функции P4:**

Определим множество единичных кубов:

L = {01001, 01010, 01011, 01101, 11001, 11010, 11011, 11101}

и множество безразличных кубов:

N = {00110, 00111, 01110, 01111, 10110, 10111, 11110, 11111}.

Первым этапом алгоритма Рота является нахождение множества простых импликант.

Для реализации этого этапа будем использовать операцию умножения (\*) над множествами *С*0, *С*1 и т. д., пока в результате операции будут образовы- ваться новые кубы большей размерности.

Первый шаг умножения (*С*0*\*С*0) приведён в таблице 3.1.2.

Таблица 3.1.2 – Поиск простых импликант (С0\*С0)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *С*0*\*С*0 | 01001 | 01010 | 01011 | 01101 | 11001 | 11010 | 11011 | 11101 | 00110 | 00111 | 01110 | 01111 | 10110 | 10111 | 11110 | 11111 |
| 01001 | ----- |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 01010 |  | ----- |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 01011 | 010x1 | 0101x | ----- |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 01101 | 01x01 |  |  | ----- |  |  |  |  |  |  |  |  |  |  |  |  |
| 11001 | x1001 |  |  |  | ----- |  |  |  |  |  |  |  |  |  |  |  |
| 11010 |  | x1010 |  |  |  | ----- |  |  |  |  |  |  |  |  |  |  |
| 11011 |  |  | x1011 |  | 110x1 | 1101x | ----- |  |  |  |  |  |  |  |  |  |
| 11101 |  |  |  | x1101 | 11x01 |  |  | ----- |  |  |  |  |  |  |  |  |
| 00110 |  |  |  |  |  |  |  |  | ----- |  |  |  |  |  |  |  |
| 00111 |  |  |  |  |  |  |  |  | 0011x | ----- |  |  |  |  |  |  |
| 01110 |  | 01x10 |  |  |  |  |  |  | 0x110 |  | ----- |  |  |  |  |  |
| 01111 |  |  | 01x11 | 011x1 |  |  |  |  |  | 0x111 | 0111x | ----- |  |  |  |  |
| 10110 |  |  |  |  |  |  |  |  | x0110 |  |  |  | ----- |  |  |  |
| 10111 |  |  |  |  |  |  |  |  |  | x0111 |  |  | 1011x | ----- |  |  |
| 11110 |  |  |  |  |  | 11x10 |  |  |  |  | x1110 |  | 1x110 |  | ----- |  |
| 11111 |  |  |  |  |  |  | 11x11 | 111x1 |  |  |  | x1111 |  | 1x111 | 1111x | ----- |

В результате этой операции сформируется новое множество кубов:

*С*1 = {010x1, 01x01, x1001, 0101x, x1010, 01x10, x1011, 01x11, x1101, 011x1, 110x1, 11x01, 1101x, 11x10, 11x11, 111x1, 0011x, 0x110, x0110, 0x111, x0111, 0111x, x1110, x1111, 1011x, 1x110, 1x111, 1111x}.

Множество *Z*0 кубов, не участвовавших в образовании новых кубов, пустое.

В приложении Б приведён следующий шаг поиска простых импликант с помощью операции *С*1*\*С*1.

В результате образовалось множество *С*2 кубов второй размерности:

*С*2 = {01xx1, x10x1, x1x01, x101x, 01x1x, x1x10, x1x11, x11x1, 11xx1, 11x1x, 0x11x, x011x, xx110, xx111, x111x, 1x11x}.

Множество *Z*1 кубов, не участвовавших в образовании новых кубов, пустое.

В таблице 3.1.4 приведен следующий шаг поиска простых импликант – операция *С*2*\*С*2.

В результате образовалось множество *С*3 кубов третьей размерности:

C3 = {x1xx1, x1x1x, xx11x}.

Множество *Z*2 кубов, не участвовавших в образовании новых кубов, пустое.

Таблица 3.1.4 – Поиск простых импликант (*С*2*\*С*2)



Результат C3\* C3 приведён в таблице 3.1.5.

Таблица 3.1.5 – Поиск простых импликант (*С3\*С3*)

|  |  |  |  |
| --- | --- | --- | --- |
| C3\* C3 | x1xx1 | x1x1x | xx11x |
| x1xx1 | ----- |  |  |
| x1x1x |  | ----- |  |
| xx11x |  |  | ----- |

Новых кубов (четвёртой размерности) не образовалось.

Получено множество: пустое.

На этом заканчивается этап поиска простых импликант, т.к. |*С*4|≤1. Мно- жество простых импликант: Z = {x1xx1, x1x1x, xx11x}.

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

Таблица 3.1.6 – Поиск *L*-экстремалей

|  |  |  |  |
| --- | --- | --- | --- |
| *Z*#(*Z*\*z*) | x1xx1 | x1x1x | xx11x |
| x1xx1 | ----- | zzzz0  x1x10 | z0zz0  x011x xx110 |
| x1x1x | zzz0z  x1x01 | ----- | zyzzz z0zzz  x011x x0110 |
| xx11x | zz0yz  x1x01 | zz0zz  x1010 | ----- |

В таблице 3.1.6 из каждой простой импликанты поочередно вычитаются все остальные простые импликанты *Z#*(*Z\z*), результат операции (последняя строка таблицы) указывает на то, что *L*-экстремалями стали следующие про- стые импликанты: x1xx1, x1x1x, xx11x.

Необходимо проверить, нет ли среди полученных *L*-экстремалей таких, которые стали *L*-экстремалями за счёт безразличных кубов. Для этого в таблице 3.1.7 из кубов множества *L* вычитаются остатки простых импликант, полученные в таблице 3.1.6 (результат выполнения операции *Z#*(*Z\z*)).

По результатам таблицы 3.1.7 *L*-экстремалями, не связанными с безразличными наборами, стали кубы x1xx1 и x1x1x (остаток от вычитания из них всех остальных простых импликант – x1x01 и x1010 соответственно – относятся к множеству единичных наборов *L* исходного задания функции). Эти кубы обязательно должны войти в минимальное покрытие.

Таблица 3.1.7 – Проверка *L*-экстремалей

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *L* ∩ *Ê* | 01001 | 01010 | 01011 | 01101 | 11001 | 11010 | 11011 | 11101 |
| x1x01 | 01001 |  |  | 01101 | 11001 |  |  | 11101 |
| x1010 |  | 01010 |  |  |  |  |  |  |
| x011x |  |  |  |  |  |  |  |  |
| x0110 |  |  |  |  |  |  |  |  |

Далее необходимо проанализировать, какие из исходных единичных кубов (множество *L*) не покрыты найденными *L*-экстремалями. Этот анализ осуществляется с помощью таблицы 3.1.8.

Таблица 3.1.8 – Поиск непокрытых исходных наборов

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *L* # *E* | 01001 | 01010 | 01011 | 01101 | 11001 | 11010 | 11011 | 11101 |
| x1xx1 | zzzzz | zzzzy  01010 | zzzzz | zzzzz | zzzzz | zzzzy  11010 | zzzzz | zzzzz |
| x1x1x |  | zzzzz |  |  |  | zzzzz |  |  |

Из таблицы 3.1.8 видно, что L-экстремалями покрыты все единичные кубы.

Следовательно, существует единственная тупиковая (минимальная) форма:

C = {x1xx1, x1x1x}.

P4 = x2\*h + x2\*y2 = x2\*(y2 + h).

Эффективность минимизации:

K = *=* 12.

Запишем функцию P4 в базисе ИЛИ-НЕ:

P4 = (x2 ↓ x2) ↓ (y2 ↓ h).

Функциональная схема ОЧУ приведена в приложении В.

**3.2 Логический синтез одноразрядного четверичного сумматора**

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

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

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

Т.к ОЧС синтезируется для схемы первого типа, то 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 | 2+2+0=10 |
| 0 | 0 | 0 | 0 | 1 | x | x | x | 2+2+1=11 |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 2+1+0=03 |
| 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 2+1+1=10 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 2+0+0=02 |
| 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 2+0+1=03 |
| 0 | 0 | 1 | 1 | 0 | x | x | x | 2+3+0=11 |
| 0 | 0 | 1 | 1 | 1 | x | x | x | 2+3+1=12 |
| 0 | 1 | 0 | 0 | 0 | x | x | x | 1+2+0=03 |
| 0 | 1 | 0 | 0 | 1 | x | x | x | 1+2+1=10 |
| 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1+1+0=02 |
| 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1+1+1=03 |
| 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1+0+0=01 |
| 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1+0+1=02 |
| 0 | 1 | 1 | 1 | 0 | x | x | x | 1+3+0=10 |
| 0 | 1 | 1 | 1 | 1 | x | x | x | 1+3+1=11 |
| 1 | 0 | 0 | 0 | 0 | x | x | x | 0+2+0=02 |
| 1 | 0 | 0 | 0 | 1 | x | x | x | 0+2+1=03 |
| 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0+1+0=01 |
| 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0+1+1=02 |
| 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0+0+0=00 |
| 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0+0+1=01 |

*Продолжение таблицы 3.2.1*

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** |
| 1 | 0 | 1 | 1 | 0 | x | x | x | 0+3+0=03 |
| 1 | 0 | 1 | 1 | 1 | x | x | x | 0+3+1=10 |
| 1 | 1 | 0 | 0 | 0 | x | x | x | 3+2+0=11 |
| 1 | 1 | 0 | 0 | 1 | x | x | x | 3+2+1=12 |
| 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 3+1+0=10 |
| 1 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 3+1+1=11 |
| 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 | x | x | x | 3+3+0=12 |
| 1 | 1 | 1 | 1 | 1 | x | x | x | 3+3+1=13 |

**Минимизация функции П:**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | b1b2p |  |  |  |  |  |  |  |  |
| a1a2 |  | 000 | 001 | 011 | 010 | 110 | 111 | 101 | 100 |
|  | 00 | x | x | 1 |  | x | x |  |  |
|  | 01 | x | x |  |  | x | x |  |  |
|  | 11 | x | x | 1 | 1 | x | x | 1 |  |
|  | 10 | x | x |  |  | x | x |  |  |

Рисунок 3.2.1 — Минимизация функции П картой Карно

Следовательно:

П = \*\*\*p + a1\*a2\*p + a1\*a2\*

Эффективность минимизации:

K = = 1,81.

Запишем функцию П в базисе ИЛИ-Константная единица-Исключающее ИЛИ:

П = (a1 + a2 + b1 + (p1))1 + ((a11) + (a21) + (p1))1) + ((a11) + (a21) + b1)1

**Минимизация функции S1:**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  | p | |  |  | p | |  |
|  |  |  |  |  | b2 | | | |  |  |
|  |  |  |  | | | |  |  | b1 |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  | x | x | 1 | 1 | x | x | 1 |  |
| a2 |  | x | x | 1 |  | x | x |  |  |
|  |  | x | x |  | 1 | x | x | 1 | 1 |
| a1 |  |  | x | x |  |  | x | x |  | 1 |

Рисунок 3.2.2 — Минимизация функции S2 картой Вейча

Следовательно:

S1 = \*\* + a1\*b1\* + a1\*a2\*b1 + a1\*a2\* + \*\*p + \*\*p + \*\*b2 = = \*\*( + p) + a1\*a2\*( +b1) + \*\*p + a1\*b1\*

Эффективность минимизации:

K = = 2,2.

Запишем функцию S1 в базисе ИЛИ-Константная единица-Исключающее ИЛИ:

S1 = (a1 + a2 + (b11 + p)1)1 + (a11 + a21 + (p1+b1)1)1 + (a1 + b1 + p1)1+(a11 + b11 +p)1

**Минимизация функции S2:**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | b1b2p |  |  |  |  |  |  |  |  |
| a1a2 |  | 000 | 001 | 011 | 010 | 110 | 111 | 101 | 100 |
|  | 00 | x | x |  | 1 | x | x | 1 |  |
|  | 01 | x | x | 1 |  | x | x |  | 1 |
|  | 11 | x | x | 1 |  | x | x |  | 1 |
|  | 10 | x | x |  | 1 | x | x | 1 |  |

Рисунок 3.2.3 — Минимизация функции S2 картой Карно

Следовательно:

S2 = \*\* + a2\*b1\* + a2\*\*p + \*b1\*p = \*(\* + b1\*p) + a2\*(b1\* + \*p)

Эффективность минимизации:

K = = 2,52.

Запишем функцию S2 в базисе ИЛИ-Константная единица-Исключающее ИЛИ:

S2 = (a2 + b1p)1 + (a21 + (b1p)1)1.

Функциональная схема ОЧC приведена в приложении Г.

**3.3. Логический синтез преобразователя множителя**

Преобразователь множителя (ПМ) – это устройство, которое преобразовывает диады множителя в соответствии с методом умножения.

При умножении в дополнительных кодах ПМ 11(34) и 10(24) заменяются на триады и . Принцип работы ПМ представлен с помощью таблицы истинности (таблица 3.3.1).

Минимизацию переключательных функций произведём с помощью карт Карно/Вейча и реализуем их в базисе И, ИЛИ, НЕ.

Таблица 3.3.1 – Таблица истинности ПМ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Входная диада(x1x2)**  **и 1й разряд следующей диады(x3)** | | | **Знак** | **Выходная диада** | |
| **x1** | **x2** | **x3** | **S** | **y1** | **y2** |
| **1** | **2** | **3** | **4** | **5** | **6** |
| 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 | 1 | 0 |
| 1 | 0 | 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 1 | 0 | 1 |
| 1 | 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 | 0 | 0 |

**Минимизация функции S:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | x1x2 |  |  |  |  |
| x3 |  | 00 | 01 | 11 | 10 |
|  | 0 |  |  | 1 | 1 |
|  | 1 |  |  |  | 1 |

Рисунок 3.3.1 - Минимизация функции S картой Карно

Следовательно:

S = x1\* + x1\* = x1\*( + )

**Минимизация функции y1:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  | x2 | |  |
|  |  | x1 | |  |  |
|  |  |  |  |  |  |
| x3 |  |  |  | 1 |  |
|  |  | 1 |  |  |  |

Рисунок 3.3.2 - Минимизация функции y1 картой Вейча

Следовательно:

y1 = \*x2\*x3 + x1\*\*

**Минимизация функции y2:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | x1x2 |  |  |  |  |
| x3 |  | 00 | 01 | 11 | 10 |
|  | 0 |  | 1 | 1 |  |
|  | 1 | 1 |  |  | 1 |

Рисунок 3.3.3 - Минимизация функции y2 картой Карно

Следовательно:

y2 = x2\* + \*x3

Функциональная схема ПМ приведена в приложении Д.

**4 Синтез комбинационных схем на основе мультиплексоров**

*Мультиплексор* – это логическая схема, которая имеет *n* информационных входов, *m* управляющих входов и один выход. При этом должно выполнятся условие .

На выход мультиплексора может быть пропущен без изменений один любой 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, подаваемым на управляющие входы.

Так как переменная b2 никак не влияет на работу ОЧС, то наиболее рационально будет синтезировать схему на основе мультиплексоров «один из четырех. В качестве переменных, которые будут подаваться на адресные входы, выберем a1 и a2. Каждая ПФ реализуется отдельным мультиплексором.

Таблица истинности для синтеза ОЧС на мультиплексорах приведена в таблице 4.1.

Таблица 4.1 – Таблица истинности ОЧС на мультиплексорах

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **a1** | ***a2*** | ***b1*** | **p** | **П** | **Функция** | **S1** | **Функция** | **S2** | **Функция** |
| **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
| 0 | 0 | 0 | 0 | 0 | \*p | 1 |  | 1 | \* + b1\*p |
| 0 | 0 | 0 | 1 | 1 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 | «0» | 0 | \*p | 0 | \*p + b1\* |
| 0 | 1 | 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | «0» | 0 | b1\* | 1 | \* + b1\*p |
| 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 |  | 1 | + b1 | 0 | \*p + b1\* |
| 1 | 1 | 0 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 0 |

Функциональная схема ОЧС на основе мультиплексоров представлена в приложении Е.

1. **ОЦЕНКА РЕЗУЛЬТАТОВ РАЗРАБОТКИ**

Время, затраченное на умножение на 1 разряд, может быть рассчитано как сумма промежутков времени, необходимых каждому узлу схемы на выполнение своих функций:

Т1 = 𝑇ПМ + 𝑇ФДК + m∗𝑇ОЧУ + 𝑇ОЧС + 𝑇сдвига + Tаккум, где

𝑇ПМ – время преобразования множителя;

𝑇ФДК – время формирования дополнительного кода множимого;

𝑇ОЧС – время сложения и формирование переноса в следующий разряд;

𝑇ОЧУ – время умножения и формирование переноса в ОЧС;

𝑇сдвига – время сдвига в регистрах;

Таккум – время суммирования промежуточного результата с очередным частичным произведением;

m – количество разрядов множимого;

Тогда для n разрядов:

Тn = n \* T1, где

n – количество разрядов множителя

Время также было уменьшено за счёт минимизации переключательных функций основных узлов сумматора-умножителя.

**ЗАКЛЮЧЕНИЕ**

Результатом курсовой работы стал спроектированный сумматор-умножитель двоично-четверичных чисел. Для данного устройства была разработана структурная схема первого типа, а также функциональных схемы основных узлов. Синтез каждого узла сопровождался минимизацией переключательных функций с целью уменьшения стоимости схемы и улучшения работоспособности.

Основными методами минимизации были карты Карно-Вейча и алгоритм Рота. Карты Карно-Вейча дают возможность ускорить процесс минимизации, однако они не очень подходят для функций от большого числа переменных. Алгоритм Рота же позволяет полностью формализовать данный процесс для любого количества переменных, и может быть реализован программно.

Для построения схем узлов сумматора-умножителя использовались различные логические базисы, что потребовало знание основных законов булевой алгебры и умение применять их.

Реализация ОЧС на мультиплексорах также способствовала уменьшению схемы и ускорению её работы.

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

Луцик, Ю.А. Л86 Арифметические и логические основы вычислительной техники: Учеб. пособие / Ю. А. Луцик, И. В. Лукьянова. – Минск : БГУИР, 2014. − 165с..

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

Савельев, А. Я. Прикладная теория цифровых автоматов / А. Я. Савельев. – М. : Высш. шк., 1987. – 272 с.

**ПРИЛОЖЕНИЕ А**

**(**обязательное**)**

Сумматор-умножитель первого типа. Схема электрическая структурная

Рисунок A.1 – Схема структурная первого типа (алгоритм «Г»)

**ПРИЛОЖЕНИЕ Б**

***(****обязательное****)***

Поиск простых импликант C1 \* C1

Таблица Б.1 – Поиск простых импликант C1 \* C1

**ПРИЛОЖЕНИЕ В**

**(**обязательное**)**

Одноразрядный четверичный умножитель. Схема электрическая функциональная

Рисунок В.1 – Одноразрядный четверичный умножитель. Схема электрическая функциональная

**ПРИЛОЖЕНИЕ Г**

**(**обязательное**)**

Одноразрядный четверичный сумматор. Схема электрическая функциональная

Рисунок Г.1 – Одноразрядный четверичный умножитель. Схема электрическая функциональная

**ПРИЛОЖЕНИЕ Д**

**(**обязательное**)**

Преобразователь множителя. Схема электрическая функциональная

Рисунок Д.1 – Преобразователь множителя. Схема электрическая функциональная

**ПРИЛОЖЕНИЕ E**

**(**обязательное**)**

Однозарядный четверичный сумматор. Схема электрическая функциональная на основе мультиплексоров

Рисунок E.1 – Однозарядный четверичный сумматор. Схема электрическая функциональная на основе мультиплексоров

**ПРИЛОЖЕНИЕ Ж**

**(**обязательное**)**

Ведомость документов