**СОДЕРЖАНИЕ**

Введение …………………………………………………………………………..5

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

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

3. Разработка функциональных схем основных узлов сумматора-

умножителя ……………………………………………………………………...10

3.1. Логический синтез одноразрядного четвертичного умножителя-сумматора ………………………………………………………………………..10

3.2. Логический синтез одноразрядного четверичного сумматора …………...19

3.3. Логический синтез преобразователя множителя ………………………….22

4. Логический синтез одноразрядного четвертичного сумматора на

основе мультиплексора …………………………………………………............25

5. Временные затраты на умножение …………………………………………..27

Заключение ………………………………………………………………………28

Список использованных источников ...……………………….………………..29

Приложение А сумматор-умножитель первого типа. Схема электрическая структурная ….…………………………………………………………………..30

Приложение Б одноразрядный четверичный умножитель. Схема электрическая функциональная ………………...………………………………31

Приложение В одноразрядный четверичный сумматор. Схема электрическая функциональная …………………………………………………………………32

Приложение Г преобразователь множителя. Схема электрическая функциональная …………………………………………………………………33

Приложение Д Однозарядный четверичный сумматор. Схема электрическая функциональная на основе мультиплексоров …………………………………34

Приложение Е таблица 3.2 – поиск простых импликант (C0\*C0) …………….35

Приложение Ж таблица 3.3 – поиск простых импликант (C1\*C1) ……………36

Приложение И таблица 3.4 – поиск простых импликант (C2\*C2) ……………37

Приложение К ведомость документов …………………………………………38

**ВВЕДЕНИЕ**

Курсовое проектирование является обязательным элементом подготовки специалиста с высшим образованием и одной из форм текущей аттестации студента по учебной дисциплине. Данная работа содержит результаты теоретических и экспериментальных исследований по дисциплине «Арифметические и логические основы цифровых устройств», включает совокупность аналитических, расчетных и экспериментальных заданий, 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:

1) разработка алгоритма умножения чисел, по которому работает СУ;

2) разработка структурной схемы СУ;

3) разработка функциональной схемы основных узлов структурной схемы СУ;

4) оценка результатов проделанной работы;

5) оформление документации по проделанной работе.

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

**1 РАЗРАБОТКА АЛГОРИТМА УМНОЖЕНИЯ**

1. Перевод сомножителей из десятичной системы счисления в четверичную

**Множимое**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| - | 68 |  | 4 |  |  |  |  | \* | 0,41 |  |  |
| 4 | - | 17 |  | 4 |  |  | 4 |  |  |
| - | 28 | 16 | - | 4 | 4 |  | \* | 1,64 |  |  |
| 28 |  | 1 | 4 | 1 |  | 4 |  |  |
|  | 0 |  |  |  | 0 |  |  |  | 2,56 |  |  |

Мн4 = 1010,12.

В соответствии с заданной кодировкой множимого:

Мн2/4 = 1000100,01100.

**Множитель**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| - | 49 | 4 | |  |  |  | \* | 0,57 |  |  |
| 4 | - | 12 | 4 |  |  | 4 |  |  |
| - | 9 | 12 | 3 |  |  | \* | 2,28 |  |  |
| 8 |  | 0 |  |  |  | 4 |  |  |
|  | 1 |  |  |  |  |  | \* | 1,12 |  |  |
|  |  |  |  |  |  |  | 4 |  |  |
|  |  |  |  |  |  |  |  | 0,48 |  |  |

Мт4 = 301,210.

В соответствии с обычной весомозначной кодировкой множителя:

Мт2/4 = 110001,100100.

2. Запишем сомножители в форме с плавающей запятой:

Мн = 0,100010001100 Pмн = 0.0100 + 0410 – закодировано по заданию,

Мт = 0,110001100100 Рмт = 0.0011 + 0310 – закодировано традиционно.

3. Умножение двух чисел с плавающей запятой на два разряда множителя одновременно в прямых кодах.

Порядок произведения будет следующим:

|  |  |  |
| --- | --- | --- |
| Pмн = | 0.0100 | 104 |
| Рмт =  Рмн\*мт= | 0.0011 | 034 |
| 0.0111 | 134 |

Результат закодирован в соответствии с заданной кодировкой 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 . Преобразованный множитель имеет вид: или Перемножение мантисс по алгоритму «Б» представлено в таблице 1.1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Четверичная с/с** | | **Двоично-четверичная с/с** | | **Комментарии** |
| **1** | | **2** | | **3** |
| 0. | 000000000000 | 0. | 000000000000000000000000 |  |
| 0. | 000000000000 | 0. | 000000000000000000000000 | П1 = Мн \* 40 |
| 0. | 000000000000 | 0. | 000000000000000000000000 |  |
| 0. | 000001010120 | 0. | 000000000000100010000100 | П2 = Мн \* 41 |
| 0. | 000001010120 | 0. | 000000000000100010000100 |  |
| 0. | 000020203000 | 0. | 000000000100010000100000 | П3= 2Мн \* 42 |
| 0. | 000021213120 | 0. | 000000000100110010100100 |  |
| 0. | 000101012000 | 0. | 000000001000100001000000 | П4 = Мн \* 43 |
| 0. | 000122231120 | 0. | 000000001101010011100100 |  |
| 0. | 000000000000 | 0. | 000000000000000000000000 | П5 = Мн \* 44 |
| 0. | 000122231120 | 0. | 000000001101010011100100 |  |
| 3. | 323232200000 | 1. | 111101110111110000000000 | П6 = -Мн \* 45 |
| 3. | 330021031120 | 1. | 111110000101000011100100 |  |
| 0. | 101012000000 | 0. | 010001000010000000000000 | П7 = Мн \* 46 |
| 0. | 031033031120 | 0. | 001101001100000111001000 |  |

Таблица 1.1 – Перемножение мантисс

После окончания умножения необходимо оценить погрешность вычислений. Для этого полученное произведение (Мн\*Мт4 = 0,031033031120, РМн·Мт = 7) приводится к нулевому порядку, а затем переводится в десятичную систему счисления:

Мн\*Мт4 = 310330,31120 РМн\*Мт = 0;

Мн\*Мт10 = 3388,8359.

Результат прямого перемножения дает следующее значение:

Мн10\*Мт10 = 68,41 \* 49,57 = 3391,0837.

Абсолютная погрешность:

  3391,0837 – 3388,8359 = 2,2478.

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 сумматора-умножителя первого типа строится на базе заданных узлов ОЧУ, ОЧС и аккумулятора (сумматора).

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) поступает «1».

Если устройство работает как умножитель (на входе Mul/sum – «0»), то множимое и множитель помещаются в соответствующие регистры, а на управляющий вход ФДК поступает «0».

Структурная схема сумматора-умножителя первого типа, работающая на основе алгоритма Б, представлена в приложении А.

**3 РАЗРАБОТКА ФУНКЦИОНАЛЬНЫХ УЗЛОВ СУММАТОРА-УМНОЖИТЕЛЯ**

**3.1 Логический синтез одноразрядного четверичного умножителя**

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

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

Разряды множителя закодированы: 04 – 00, 14 – 01, 24 – 10, 34 – 11.

Разряды множимого закодированы: 04 – 10, 14 – 00, 24 – 01, 34 – 11.

Управляющий вход h определяет тип операции:

- «0» – умножение закодированных цифр, поступивших на информационные входы, и добавление переноса;

- «1» – вывод на выходы без изменения значений разрядов, поступивших из регистра множимого.

В таблице 3.1 определено 8 безразличных наборов, потому что на входы ОЧУ из разрядов множителя не может поступить код «11».

Таблица 3.1 – 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 | 1·0=00 |
| 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | Выход – код «01» |
| 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 1·1=01 |
| 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | Выход – код «01» |
| 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1·2=02 |
| 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | Выход – код «01» |
| 0 | 0 | 1 | 1 | 0 | х | х | x | х | 1·3=03 |
| 0 | 0 | 1 | 1 | 1 | х | х | x | х | Выход – код «01» |
| 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 2·0=00 |
| 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | Выход – код «02» |
| 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 2·1=02 |
| 0 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | Выход – код «02» |
| 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 2·2=10 |
| 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | Выход – код «02» |
| 0 | 1 | 1 | 1 | 0 | х | х | x | х | 2·3=12 |
| 0 | 1 | 1 | 1 | 1 | х | х | x | х | Выход – код «02» |
| 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0·0=00 |

*Продолжение таблицы 3.1*

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 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 | x | х | 0·3=00 |
| 1 | 0 | 1 | 1 | 1 | х | х | x | 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 | 0 | 0 | 1 | 3·2=12 |
| 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | Выход – код «03» |
| 1 | 1 | 1 | 1 | 0 | х | х | x | х | 3·3=21 |
| 1 | 1 | 1 | 1 | 1 | х | х | x | х | Выход – код «03» |

Минимизация переключательных функций проведем с помощью карт Вейча.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | x1 | | | |  |  |  |  |  | |  | |
|  |  |  | | | |  |  |  |  |  | |  | |
| y1 |  | 1 | 1 | 1 |  |  | 1 | 1 | 1 |  | |  |
| \* | \* | \* | \* | \* | \* | \* | \* |  | y2 |
|  |  | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
|  |  | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |  | |  |
|  |  |  |  |  | | | |  |  |  | |  |
|  |  |  |  | | x2 | |  | |  |  | |  |
|  |  |  | h | |  |  | h | |  |  | |  |

Рисунок 3.1 – Минимизация функции Р1

Следовательно:

Запись в базисе ИЛИ-НЕ:

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

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | x1 | | | |  |  |  |  |  | |  | |
|  |  |  | | | |  |  |  |  |  | |  | |
| y1 |  | 1 | 1 | 1 |  | 1 |  |  |  |  | |  |
| \* | \* | \* | \* | \* | \* | \* | \* |  | y2 |
|  |  | 1 | 1 | 1 | 1 |  |  |  |  |
|  |  | 1 | 1 | 1 | 1 | 1 |  |  | 1 |  | |  |
|  |  |  |  |  | | | |  |  |  | |  |
|  |  |  |  | | x2 | |  | |  |  | |  |
|  |  |  | h | |  |  | h | |  |  | |  |

Рисунок 3.2 – Минимизация функции Р3

Следовательно:

Запись в базисе ИЛИ-НЕ:

Эффективность минимизации:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | x1 | | | |  |  |  |  |  | |  | |
|  |  |  | | | |  |  |  |  |  | |  | |
| y1 |  |  |  | 1 | 1 |  | 1 |  | 1 |  | |  |
| \* | \* | \* | \* | \* | \* | \* | \* |  | y2 |
|  |  |  |  | 1 | 1 | 1 | 1 |  |  |
|  |  |  |  | 1 |  |  | 1 |  |  |  | |  |
|  |  |  |  |  | | | |  |  |  | |  |
|  |  |  |  | | x2 | |  | |  |  | |  |
|  |  |  | h | |  |  | h | |  |  | |  |

Рисунок 3.3 – Минимизация функции Р4

Следовательно:

Запись в базисе ИЛИ-НЕ:

Эффективность минимизации:

Минимизацию функции P3 проведем с помощью алгоритма рота.

Определим множество единичных и безразличных кубов:

В множестве N произведем склеивание исходных кубов и получим следующее: .

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) приведен в таблице 3.2 в приложении Е.

Из полученных новых кубов образуется множество А1:

Множество Z0 кубов, не участвовавших в образовании новых кубов:

Z0 = {Ø}.

Сформируем множество кубов :

Формируем новое множество кубов :

Умножение С1\*С1 приведено в таблице 3.3 в приложении Ж

Из полученных новых кубов образуется множество А2:

Множество Z1 кубов, не участвовавших в образовании новых кубов:

Z1 = {01x00; 011x0}.

Сформируем множество кубов :

Формируем новое множество кубов :

Умножение С2\*С2 приведено в таблице 3.4 в приложении И.

Из полученных новых кубов образуется множество А3:

.

Множество Z2 кубов, не участвовавших в образовании новых кубов:

Z2 = {01x00; 011x0; xx000}.

Сформируем множество кубов :

.

Формируем новое множество кубов :

.

Таблица 3.5 – Поиск простых импликант (С3\*С3)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| C3\*C3 | 10xxx | 1x0xx | 1xxx1 | 1xx1x |
| 10xxx | - |  |  |  |
| 1x0xx |  | - |  |  |
| 1xxx1 |  |  | - |  |
| 1xx1x |  |  |  | - |
| xx11x |  |  |  |  |
| A4 | Ø | Ø | Ø | Ø |

Из полученных новых кубов образуется множество А4:

.

Множество Z3 кубов, не участвовавших в образовании новых кубов:

На данном этапе заканчивается поиск простых ипликант.

Множество простых импликант :

Следующий этап – поиск L–экстремалей на множестве простых импликант. Для этого используется операция вычитания (#).

Таблица 3.6 – Поиск L-экстремалей

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| z#(Z-z) | 01x00 | 011x0 | xx000 | 10xxx | 1x0xx | 1xxx1 | 1xx1x | xx11x |
| 01x00 | - | 01110 | 1x000 x0000 | 10xxx | 1x0xx | 1xxx1 | 1xx1x | xx11x |
| 011x0 | 01000 | - | 1x000 x0000 | 10xxx | 1x0xx | 1xxx1 | 1xx1x | 1x11x x011x xx111 |
| xx000 | Ø | 01110 | - | 101xx 10x1x 10xx1 | 1x01x 1x0x1 | 1xxx1 | 1xx1x | 1x11x x011x xx111 |
| 10xxx | Ø | 01110 | 11000 00000 | - | 1101x 110x1 | 11xx1 | 11x1x | 1111x 0011x 0x111 x1111 |
| 1x0xx | Ø | 01110 | 00000 | 101xx 1011x 101x1 | - | 111x1 | 1111x | 1111x 0011x 0x111 x1111 |
| 1xxx1 | Ø | 01110 | 00000 | 101x0 10110 | 11010 | - | 11110 | 11110 0011x 0x111 01111 |

*Продолжение таблицы 3.6*

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1xx1x | Ø | 01110 | 00000 | 10100 | Ø | 11101 | - | 0011x 0x111 01111 |
| xx11x | Ø | Ø | 00000 | 10100 | Ø | 11101 | Ø | - |
| Остаток | Ø | Ø | 00000 | 10100 | Ø | 11101 | Ø | 0011x 0x111 01111 |

В таблице 3.6 из каждой простой импликанты поочередно вычитаются все остальные простые импликанты Z#(Z-z). Результат операции указывает на то, что L-экстремалями стали следующие простые импликанты:

*.*

После выявления L-экстремалей следует выяснить, не являются ли некоторые из них простыми импликантами, остатки которых покрывают только некоторое подмножество кубов комплекса N, которое нет необходимости покрывать. Для этого необходимо выполнить операцию пересечения остатков . Во множестве E необходимо оставить только те кубы, остатки от которых пересекаются с кубами из комплекса L.

Таблица 3.7 – Проверка L-экстремалей

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| z#(Z-z)  ⋂ L | 00000 | 01000 | 01100 | 10000 | 10001 | 10010 | 10011 | 10100 | 10101 | 11000 | 11001 | 11010 | 11011 | 11101 |
| 00000 | 00000 | 0y000 Ø | 0yy00 Ø | y0000 Ø | y000y Ø | y00y0 Ø | y00yy Ø | y0y00 Ø | y0y0y Ø | yy000 Ø | yy00y Ø | yy0y0 Ø | yy0yy Ø | yyy0y Ø |
| 10100 | y0y00 Ø | yyy00 Ø | yy100 Ø | 10y00 Ø | 10y0y Ø | 10yy0 Ø | 10yyy Ø | 10100 | 1010y Ø | 1yy00 Ø | 1yy0y Ø | 1yyy0 Ø | 1yyyy Ø | 1y10y Ø |
| 11101 | yyy0y Ø | y1y0y Ø | y110y Ø | 1yy0y Ø | 1yy01 Ø | 1yyyy Ø | 1yyy1 Ø | 1y10y Ø | 1y101 Ø | 11y0y Ø | 11y01 Ø | 11yyy Ø | 11yy1 Ø | 11101 |
| 0011x | 00yy0 Ø | 0yyy0 Ø | 0y1y0 Ø | y0yy0 Ø | y0yy1 Ø | y0y10 Ø | y0y11 Ø | y01y0 Ø | y01y1 Ø | yyyy0 Ø | yyyy1 Ø | yyy10 Ø | yyy11 Ø | yy1y1 Ø |
| 0x111 | 00yyy Ø | 01yyy Ø | 011yy Ø | y0yyy Ø | y0yy1 Ø | y0y1y Ø | y0y11 Ø | y01yy Ø | y01y1 Ø | y1yyy Ø | y1yy1 Ø | y1y1y Ø | y1y11 Ø | y11y1 Ø |
| 01111 | 0yyyy Ø | 01yyy Ø | 011yy Ø | yyyyy Ø | yyyy1 Ø | yyy1y Ø | yyy11 Ø | yy1yy Ø | yy1y1 Ø | y1yyy Ø | y1yy1 Ø | y1y1y Ø | y1y11 Ø | y11y1 Ø |

По результатам таблицы 3.7 L-экстремали xx000, 10xxx, 1xxx1 оказались не связанными с безразличными наборами, а значит они обязательно должны войти в минимальное покрытие.

*.*

На следующем этапе выясним, какие из вершин комплекса L не покрываются L-экстремалями. Для этого из каждого куба комплекса L вычтем элементы множества Е. В результате вычитания получим .

Таблица 3.8 – Поиск непокрытых исходных наборов

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 00000 | 01000 | 01100 | 10000 | 10001 | 10010 | 10011 | 10100 | 10101 | 11000 | 11001 | 11010 | 11011 | 11101 |
| xx000 | Ø | Ø | 01100 | Ø | 10001 | 10010 | 10011 | 10100 | 10101 | Ø | 11001 | 11010 | 11011 | 11101 |
| 10xxx | Ø | Ø | 01100 | Ø | Ø | Ø | Ø | Ø | Ø | Ø | 11001 | 11010 | 11011 | 11101 |
| 1xxx1 | Ø | Ø | 01100 | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | 11010 | Ø | Ø |
| Остаток | Ø | Ø | 01100 | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | 11010 | Ø | Ø |

Из таблицы 3.8 видно, что L-экстремалью не покрыты два единичных куба *.* Однако непокрытые кубы должны быть покрыты другими импликантами из множества:

Таблица 3.9 – Покрытие оставшихся кубов

|  |  |  |
| --- | --- | --- |
|  | 01100 | 11010 |
| 01x00 | 01100 | y10y0 Ø |
| 011x0 | 01100 | y1y10 Ø |
| 1x0xx | y1y00 Ø | 11010 |
| 1xx1x | y11y0 Ø | 11010 |
| xx11x | 011y0 Ø | 11y10 Ø |

Из таблицы 3.9 видно, что каждый из непокрытых единичных кубов может быть покрыт двумя равнозначными способами.

Следовательно, существуют четыре тупиковые формы:

;

Схема электрическая функциональная ОЧУ представлена в приложении Б.

**3.2 Логический синтез одноразрядного четверичного сумматора**

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

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

Разряды обоих слагаемых закодированы: 04 – 10, 14 – 00, 24 – 01, 34 – 11.

Если ОЧС синтезируется для схемы первого типа, то в таблице истинности необходимо выделить 16 безразличных наборов, так как со старших выходов ОЧУ не могут прийти коды «2» и «3».

Таблица 3.10 – 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 | 0 | 0 | 1 | 1+1+0=02 |
| 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1+1+1=03 |
| 0 | 0 | 0 | 1 | 0 | х | х | х | 1+2+0=03 |
| 0 | 0 | 0 | 1 | 1 | х | х | х | 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 | х | х | х | 1+3+0=10 |
| 0 | 0 | 1 | 1 | 1 | х | х | х | 1+3+1=11 |
| 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 2+1+0=03 |
| 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 2+1+1=10 |
| 0 | 1 | 0 | 1 | 0 | х | х | х | 2+2+0=10 |
| 0 | 1 | 0 | 1 | 1 | х | х | х | 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 | x | x | x | 2+3+0=11 |
| 0 | 1 | 1 | 1 | 1 | x | x | x | 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 | x | x | x | 0+2+0=02 |
| 1 | 0 | 0 | 1 | 1 | x | x | x | 0+2+1=03 |

*Продолжение таблицы 3.10*

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 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 | x | x | x | 0+3+0=03 |
| 1 | 0 | 1 | 1 | 1 | x | x | x | 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 | x | x | x | 3+2+0=11 |
| 1 | 1 | 0 | 1 | 1 | x | x | x | 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 | x | x | x | 3+3+0=12 |
| 1 | 1 | 1 | 1 | 1 | x | x | x | 3+3+1=13 |

Проведём минимизацию переключательных функций с помощью карт Карно.

Минимизация функции П:

b1b2p

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

Рисунок 3.4 – Минимизация функции П

Следовательно:

.

Запись в базисе И, сложение по модулю 2, константная единица:

*.*

Эффективность минимизации:

Минимизация функции S1:

b1b2p

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

Рисунок 3.5 – Минимизация функции S1

Следовательно:

.

Запись в базисе И, сложение по модулю 2, константная единица:

Эффективность минимизации:

Минимизация функции S2:

b1b2p

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

Рисунок 3.6 – Минимизация функции S2

Следовательно:

Запись в базисе И, сложение по модулю 2, константная единица:

Эффективность минимизации:

Схема электрическая функциональная ОЧУ представлена в приложении В.

**3.3. Логический синтез преобразователя множителя**

Преобразователь множителя (ПМ) – это устройство, которое преобразовывает диады множителя в соответствии с методом умножения. Служит для исключения из множителя диад 11 (34), заменяя их на триады .

Таблица 3.11 - Таблица истинности ПМ*.*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Вх. диада** | | **Перенос** | **Зн.** | **Триггер** | **Вых. диада** | |
| Q1 | Q2 | P | Z | П | S1 | S2 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 0 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 0 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 | 1 | 0 | 0 |

Минимизацию переключательных функций произведем с помощью карт Карно-Вейча.

Минимизация функции П:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  | P |  |  |
|  |  | Q2 | |  |  |
|  |  |  | |  |  |
| Q1 |  | 1 | 1 | 1 |  |
|  |  |  |  |  |

Рисунок 3.7 - Минимизация функции П картой Вейча

Следовательно:

Эффективность минимизации:

Минимизация функции Z:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  | P |  |  |
|  |  | Q2 | |  |  |
|  |  |  | |  |  |
| Q1 |  | 1 |  | 1 |  |
|  |  |  |  |  |  |

Рисунок 3.8 - Минимизация функции Z картой Вейча

Следовательно:

.

Эффективность минимизации:

Минимизация функции S1:

Q2p

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Q1 | 00 | 01 | 11 | 10 |
| 0 |  |  | 1 |  |
| 1 | 1 |  |  |  |

Рисунок 3.9 – Минимизация функции S1 картой Карно

Следовательно:

.

Эффективность минимизации:

Минимизация функции S2:

Q2p

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Q1 | 00 | 01 | 11 | 10 |
| 0 |  | 1 |  | 1 |
| 1 |  | 1 |  | 1 |

Рисунок 3.10 – Минимизация функции S2 картой Карно

Следовательно:

.

Эффективность минимизации:

Схема электрическая функциональная ПМ представлена в приложении Г.

**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:

1) на выход мультиплексора может быть пропущен без изменений любой (один) логический сигнал, поступающий на один из информационных входов;

2) 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.

Таблица 4.1 – таблица истинности ОЧС

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***a*1** | ***a*2** | ***b*1** | ***b*2** | ***p*** | **П** | **Функция** | ***S*1** | **Функция** | ***S*2** | **Функция** |
| **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | 11 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | p | 1 | «1» |
| 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 |
| 0 | 0 | 0 | 1 | 0 | х | х | х |
| 0 | 0 | 0 | 1 | 1 | х | х | х |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | p |
| 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 | 0 | х | х | х |
| 0 | 0 | 1 | 1 | 1 | х | х | х |
| 0 | 1 | 0 | 0 | 0 | 0 | p | 1 | «1» | 1 |  |
| 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 | х | х | х |
| 0 | 1 | 0 | 1 | 1 | х | х | х |
| 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | p | 1 | «1» |
| 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 1 | 0 | x | x | x |
| 0 | 1 | 1 | 1 | 1 | x | x | x |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | p |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 | x | x | x |
| 1 | 0 | 0 | 1 | 1 | x | x | x |

*Продолжение таблицы 4.1*

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |  | 0 | 0 |
| 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 | 0 | x | x | x |
| 1 | 0 | 1 | 1 | 1 | x | x | x |
| 1 | 1 | 0 | 0 | 0 | 1 | «1» | 1 |  | 0 | 0 |
| 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 | x | x | x |
| 1 | 1 | 0 | 1 | 1 | x | x | x |
| 1 | 1 | 1 | 0 | 0 | 0 | p | 1 | «1» | 1 |  |
| 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 |
| 1 | 1 | 1 | 1 | 0 | x | x | x |
| 1 | 1 | 1 | 1 | 1 | x | x | X |

Схема электрическая функциональная ОЧС на основе мультиплексоров представлена в приложении Д.

**5 Временные затраты на умножение**

Формула расчета временных затрат на умножение:

.

– время сдвига в регистрах;

– время преобразования множителя;

– время формирования дополнительного кода множимого.

– время умножения на ОЧУ;

– время формирования единицы переноса в ОЧС;

n – количество разрядов множителя;

m – количество разрядов множимого.

Время работы определяется самой длинной цепочкой:

Tэл – среднее время срабатывания одного элемента.

Таким образом, формула расчета временных затрат на умножение примет вид:

.

**ЗАКЛЮЧЕНИЕ**

В ходе выполнения курсовой работы была разработана структурная схема для сумматора-умножителя первого типа, а также функциональные схемы для его основных компонентов.

Для уменьшения цены схем были использованы различные методы минимизации переключательных функций. Это позволило выявить преимущества и недостатки каждого метода. Метод Карно-Вейча можно выделить за простоту и быстроту, но он может быть затруднительным при работе с функциями многих переменных. Для таких функций был использован алгоритм Рота, который полностью формализует процесс минимизации и делает его доступным для выполнения компьютерной программой.

Функциональные схемы были созданы в различных логических базисах, что помогло закрепить теоретические знания булевой алгебры, включая правило де Моргана. Использование мультиплексоров для реализации переключательных функций упростило процесс минимизации и упростило функциональную схему одноразрядного четверичного сумматора.

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

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

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

Преобразователь множителя. Схема электрическая функциональная

**ПРИЛОЖЕНИЕ Д**

(обязательное)

Однозарядный четверичный сумматор. Схема электрическая функциональная на основе мультиплексоров

**ПРИЛОЖЕНИЕ Е**

(обязательное)

Таблица 3.2 – Поиск простых импликант (С0\*С0)

**ПРИЛОЖЕНИЕ Ж**

(обязательное)

Таблица 3.3 – Поиск простых импликант (С1\*С1)

**ПРИЛОЖЕНИЕ И**

(обязательное)

Таблица 3.4 – Поиск простых импликант (С2\*С2)

**ПРИЛОЖЕНИЕ К**

(обязательное)

Ведомость документов