**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 вычислительной техники”, 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**

1. Перевод сомножителей из десятичной системы счисления в четверичную.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| \_ 28 | 4 |  |  |  |  |  | 0,69 |
| 28 | \_ 7 | 4 |  |  |  | \* | 4 |
| 0 | 4 | 1 |  |  |  |  | 2,76 |
|  | 3 |  |  |  |  | \* | 4 |
|  |  |  |  |  |  |  | 3,04 |
|  |  |  |  |  |  | \* | 4 |
|  |  |  |  |  |  |  | 0,16 |

Мн4 = 130,230.

В соответствии с заданной кодировкой множимого:

Мн2/4 = 100100,110100

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| \_ 21 | 4 |  |  |  |  |  | 0,59 |
| 20 | \_ 5 | 4 |  |  |  | \* | 4 |
| 1 | 4 | 1 |  |  |  |  | 2,36 |
|  | 1 |  |  |  |  | \* | 4 |
|  |  |  |  |  |  |  | 1,44 |
|  |  |  |  |  |  | \* | 4 |
|  |  |  |  |  |  |  | 1,76 |

Мт4 = 111,211.

В соответствии с заданной кодировкой множителя:

Мт2/4 = 101010,111010

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

Мн = 0,100100110100 РМн = 0.0011 +0310

Мт = 0, 101010111010 РМт = 0.0011 +0310

Порядок произведения:

|  |  |  |  |
| --- | --- | --- | --- |
| РМн | = | 0.0011 | 034 |
| РМт | = | 0.0011 | 034 |
| РМн∙Мт | = | 0.0110 | 124 |
|  |  |  |  |

Знак произведения определяется суммой по модулю два знаков

сомножителей:

зн Мн ⊕ зн Мт = 0 ⊕ 0 = 0.

При умножении чисел в прямых кодах диада 11(34) заменяется на триаду . Преобразованный множитель имеет вид М = .

Перемножение мантисс по алгоритму «Б» представлено в таблице 1.1

Таблица 1.1 — Перемножение мантисс

|  |  |  |
| --- | --- | --- |
| **Четверичная С/С** | | **Комментарии** |
| **1** | | **2** |
| 0. | 000000000000 | ∑0ч |
| 0. | 000000130230 | П1ч = Мн ∙ 1 ∙ 40 |
| 0. | 000000130230 | ∑1ч |
| 0. | 000001302300 | П2ч = Мн ∙ 1 ∙ 41 |
| 0. | 000002033130 | ∑2ч |
| 0. | 000032112000 | П3ч = Мн ∙ 2 ∙ 42 |
| 0. | 000100211130 | ∑3ч |
| 0. | 000130230000 | П4ч = Мн ∙ 1 ∙ 43 |
| 0. | 000231101130 | ∑4ч |
| 0. | 001302300000 | П5ч = Мн ∙ 1 ∙ 44 |
| 0. | 002200001130 | ∑5ч |
| 0. | 013023000000 | П6ч = Мн ∙ 1 ∙ 45 |
| 0. | 021223001130 | ∑6ч |

После окончания умножения необходимо оценить погрешность вычислений. Для этого полученное произведение (Мн4 ∙ Мт4 = 0, 0,21223001130, РМн ∙ Мт = 6) приводится к нулевому порядку, а затем переводится в десятичную систему счисления:

Мн4 ∙ Мт4 = 21223,001130 РМн ∙ Мт = 0;

Мн10 ∙ Мт10 = 619,027344.

Результат прямого перемножения операндов дает следующее:

Мн10 ∙ Мт10 = 28,69 ∙ 21,59 = 619,4171

Абсолютная погрешность:

Δ = 619,027344 – 619,4171 = 0,389756.

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**

Разработка сумматора-умножителя второго типа строится на базе одно-разрядного четверичного сумматора(ОЧС) и одноразрядного четверичного умножителя-сумматора (ОЧУС)

*Если устройство работает как сумматор*, то оба слагаемых последовательно (за два такта) заносятся в регистр множимого, а на управляющий вход формирователя дополнительного кода F2 поступает «1».

Первое слагаемое переписывается в регистр результата под действием управляющих сигналов, поступающих на входы h всех ОЧУС (рисунок 2.1).

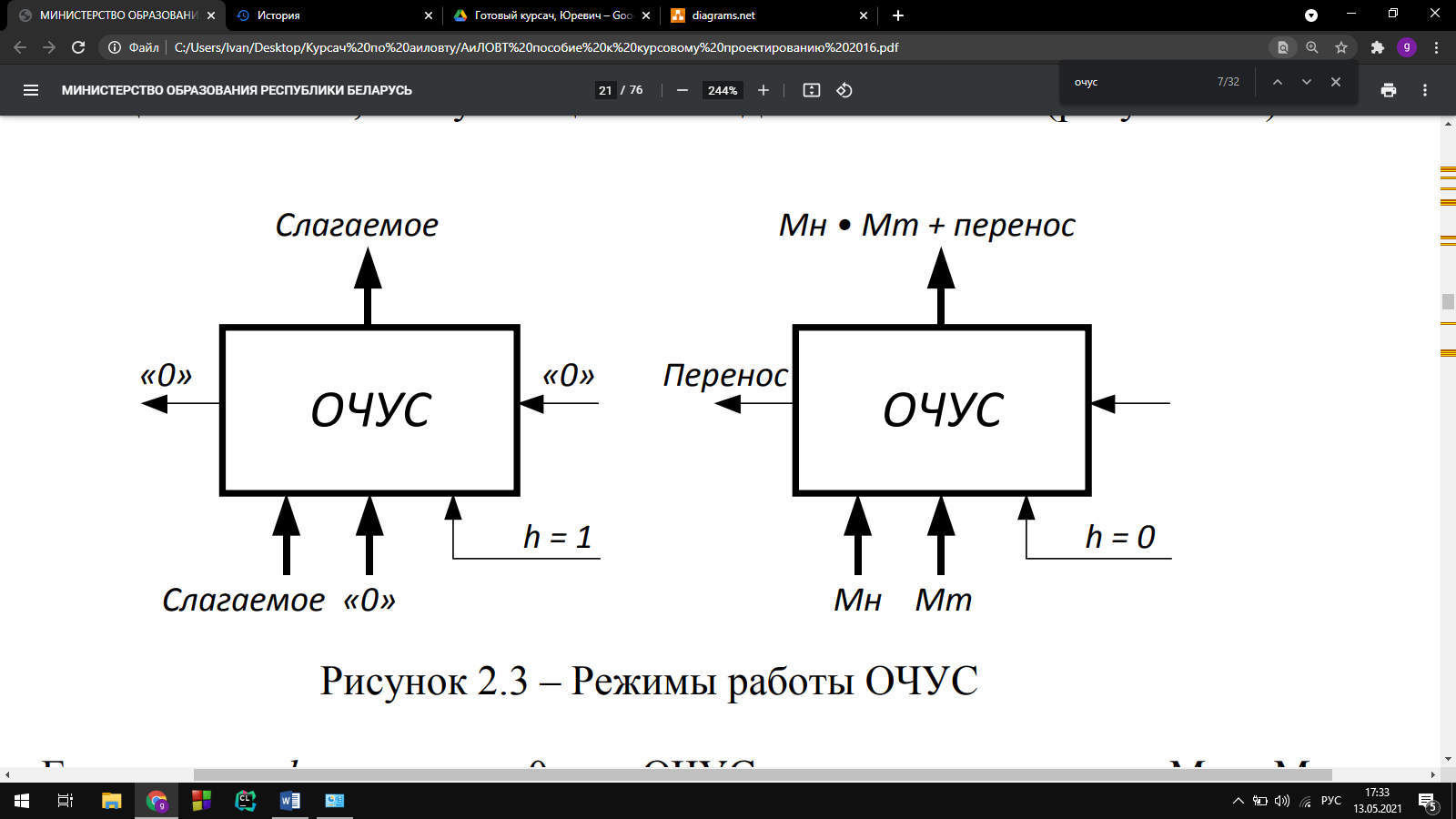


Рисунок 2.1 – Режимы работы ОЧУС

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

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

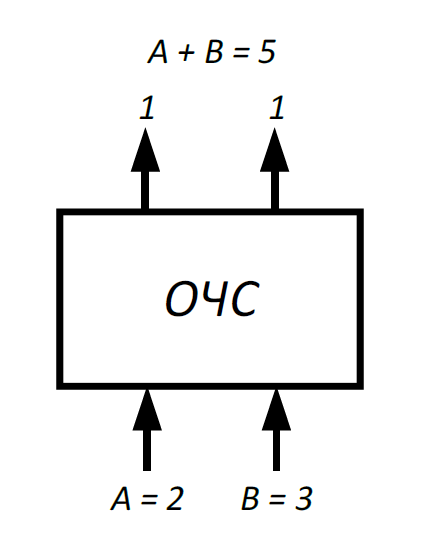


Рисунок 2.2 – Одноразрядный четверичный сумматор

В ОЧС первое слагаемое складывается с нулём, записанным в регистре результата, и переписывается без изменений в регистр результата.

На втором такте второе слагаемое из регистра множимого через цепочку ОЧУС попадает на входы ОЧС и складывается с первым слагаемым, хранящимся в регистре результата.

Сумма хранится в регистре результата. Разрядность регистра результата должна быть на единицу больше, чем разрядность исходных слагаемых, чтобы предусмотреть возможность возникновения при суммировании переноса.

*Если устройство работает как умножитель*, то множимое и множитель помещаются в соответствующие регистры, а на управляющий вход ФДК F2 поступает «0».

Диада множителя поступает на входы преобразователя множителя. Единица переноса в следующую диаду, если она возникает, должна быть добавлена к следующей диаде множителя (выход 1 ПМ) в следующем такте, т. е. должна храниться на триггере до следующего такта.

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

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

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

Таблица 2.1 – Режимы работы формирователя дополнительного кода

|  |  |  |
| --- | --- | --- |
| **Сигналы на входах ФДК** | | **Результат на выходах ФДК** |
| *F1* | *F2* |
| 0 | 0 | Дополнительный код множимого |
| 0 | 1 | Дополнительный код слагаемого |
| 1 | 0 | Меняется знак Мн |
| 1 | 1 | Меняется знак слагаемого |

На выходах 3 и 4 ПМ формируются диады преобразованного множителя, которые поступают на входы ОЧУС вместе с диадами множимого. На трёх выходах ОЧУС формируется результат умножения диад Мн·Мт плюс перенос из предыдущего ОЧУС. Максимальной цифрой в диаде преобразованного множителя является двойка, поэтому перенос, формируемый ОЧУС, может быть только двоичным («0» или «1»):

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 3 | ∙ | 2 | = | 1 2 | (+1 в случае переноса из предыдущего ОЧУС) | |
| max |  | max |  | max | |  | |
| Мн |  | Мт |  | Перенос | |  | |

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

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

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

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

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

**3.1 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 3.1.1)

Кодировка слагаемых обоих разрядов: 0 – 00, 1 – 10, 2 – 11, 3 – 01,

Таблица 3.1.1 — Таблица истинности ОЧС

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

Продолжение таблицы 3.1.1

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

**Минимизация функции П:**

Минимизацию функции П проведем с помощью карт Вейча. Для функции Р заполненная карта приведена на рисунке 3.1.1. В рисунках 3.1.1 – 3.1.1.

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

Рисунок 3.1.1 — Минимизация функции П картой Вейча

Следовательно:

Запишем результат в базисе: .

**Минимизация функции S1:**

Минимизацию функции **S1** проведем с помощью карт Вейча. Для функции Р заполненная карта приведена на рисунке 3.1.2. В рисунках 3.1.2 – 3.1.2.

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

Рисунок 3.1.2 — Минимизация функции **S1** картой Вейча

Следовательно

Запишем результат в базисе:

.

**Минимизация функции S2:**

Минимизацию функции **S2** проведем с помощью карт Вейча. Для функции Р заполненная карта приведена на рисунке 3.1.3. В рисунках 3.1.3 – 3.1.3.

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

Рисунок 3.1.3 — Минимизация функции **S2** картой Вейча

Следовательно

Запишем результат в базисе:

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

Функциональная схема ОЧС приведена в приложении на чертеже ГУИР.400201.306 Э2.1.

**3.2 Логический синтез одноразрядного четверичного сумматора-умножителя**

ОЧУС – это комбинационное устройство, имеющее 5 двоичных входов (2 разряда из регистра Мн, 2 разряда из регистра Мт и управляющий вход *h*) и 3 двоичных выхода.

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

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

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

Управляющий вход *h* определяет тип операции:

«0» – вывод результата умножения закодированных цифр с добавлением переноса из предыдущего ОЧУС, перенос в следующий ОЧУС.

«1» – вывод без изменения значения разрядов, поступивших из регистра множимого, перенос *из* и *в* ОЧУС равны нулю.

В таблице 3.2.1 выделены безразличные наборы, т.к. на входы ОЧУС из разрядов множителя не может поступить код «11».

Таблица 3.2.1 — Таблица истинности ОЧУС

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Пер.** | **Мн.** | | **Мт.** | | **Упр.** | **Перенос** | **Результат** | | **Результат операции**  **в четверичной с/с** |
| ***P1*** | ***x1*** | ***x2*** | ***y1*** | ***y2*** | ***h*** | ***P*** | ***Q1*** | ***Q2*** |
| **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0\*0+0=00 |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | Выход - код "00" |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0\*1+0=00 |
| 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | Выход - код "00" |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0\*2+0=00 |
| 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | Выход - код "00" |
| 0 | 0 | 0 | 1 | 1 | 0 | x | x | x | 0\*3+0=00 |
| 0 | 0 | 0 | 1 | 1 | 1 | x | x | x | Выход - код "00" |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 3\*0+0=00 |
| 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | Выход - код "03" |
| 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 3\*1+0=03 |
| 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | Выход - код "03" |
| 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 3\*2+0=12 |
| 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | Выход - код "03" |
| 0 | 0 | 1 | 1 | 1 | 0 | x | x | x | 3\*3+0=21 |
| 0 | 0 | 1 | 1 | 1 | 1 | x | x | x | Выход - код "03" |

*Продолжение таблицы 3.2.1*

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

*Продолжение таблицы 3.2.1*

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
| 1 | 1 | 1 | 0 | 0 | 0 | x | x | x | 1\*0+1=01 |
| 1 | 1 | 1 | 0 | 0 | 1 | x | x | x | Выход - код "02" |
| 1 | 1 | 1 | 0 | 1 | 0 | x | x | x | 2\*1+1=03 |
| 1 | 1 | 1 | 0 | 1 | 1 | x | x | x | Выход - код "02" |
| 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 2\*2+1=11 |
| 1 | 1 | 1 | 1 | 0 | 1 | x | x | x | Выход - код "02" |
| 1 | 1 | 1 | 1 | 1 | 0 | x | x | x | 2\*3+1=13 |
| 1 | 1 | 1 | 1 | 1 | 1 | x | x | x | Выход - код "02" |

**Минимизация функции P:**

Минимизацию функции P проведем с помощью карт Карно. Для функции Р заполненная карта приведена на рисунке 3.2.1. В рисунках 3.2.1 – 3.2.1 символом «х» отмечены наборы, на которых функция может принимать произвольное значение (безразличные наборы).

y1y2h

P1x1x2

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 000 | 001 | 011 | 010 | 110 | 111 | 101 | 100 |
| 000 |  |  |  |  | х | х |  |  |
| 001 |  |  |  |  | х | х |  | 1 |
| 011 |  |  |  |  | х | х |  | 1 |
| 010 |  |  |  |  | х | х |  |  |
| 110 | х | х | х | х | х | х | х |  |
| 111 | х | х | х | х | х | х | х | 1 |
| 101 | х | х | х | х | х | х | х | 1 |
| 100 | х | х | х | х | х | х | х |  |

Рисунок 3.2.1 — Минимизация функции Р картой Карно

Следовательно

*P1 =*

Запишем результат в базисе:

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

**Минимизация функции Q1:**

y1y2h

px1x2

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 000 | 001 | 011 | 010 | 110 | 111 | 101 | 100 |
| 000 |  |  |  |  | х | х |  |  |
| 001 |  |  |  |  | х | х |  | 1 |
| 011 |  | 1 | 1 | 1 | х | х | 1 |  |
| 010 | 1 | 1 | 1 | 1 | х | х | 1 | 1 |
| 110 | х | х | х | х | х | х | х |  |
| 111 | х | х | х | х | х | х | х | 1 |
| 101 | х | х | х | х | х | х | х |  |
| 100 | х | х | х | х | х | х | х | 1 |

Рисунок 3.2.2 — Минимизация функции Q1 картой Карно

Следовательно

Запишем результат в базисе:

Эффективность минимизации:

­

**Минимизация функции Q1:**

Определим множество единичных кубов:

.

Определим множество безразличных кубов:

Минимизацию безразличных кубов проведём с помощью карты Карно. Для безразличных кубов заполненная карта приведена на рисунке 3.1, где символом «x» отмечены наборы, на которых функция не определена.

y1y2h

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| px1x2 | 000 | 001 | 011 | 010 | 110 | 111 | 101 | 100 |
| 000 |  |  |  |  | х | х |  |  |
| 001 |  |  |  |  | х | х |  |  |
| 011 |  |  |  |  | х | х |  |  |
| 010 |  |  |  |  | х | х |  |  |
| 110 | х | х | х | х | х | х | х |  |
| 111 | х | х | х | х | х | х | х |  |
| 101 | х | х | х | х | х | х | х |  |
| 100 | х | х | х | х | х | х | х |  |

Рисунок 3.2.3 — Минимизация безразличный кубов картой Карно

Множество безразличных наборов после минимизации:

.

Формируем :

.

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 3.2.2

Таблица 3.2.2– Поиск простых импликант (С0\*С0)

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| C0\*C0 | 001001 | 001011 | 001010 | 001101 | 001100 | 011001 | 011011 | 011010 | 011101 |
| 001001 | - |  |  |  |  |  |  |  |  |
| 001011 | 0010y1 | - |  |  |  |  |  |  |  |
| 001010 |  | 00101y | - |  |  |  |  |  |  |
| 001101 | 001y01 |  |  | - |  |  |  |  |  |
| 001100 |  |  |  | 00110y | - |  |  |  |  |

*Продолжение таблицы 3.2.2*

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| C0\*C0 | 001001 | 001011 | 001010 | 001101 | 001100 | 011001 | 011011 | 011010 | 011101 |
| 011001 | 0y1001 |  |  |  |  | - |  |  |  |
| 011011 |  | 0y1011 |  |  |  | 0110y1 | - |  |  |
| 011010 |  |  | 0y1010 |  |  |  | 01101y | - |  |
| 011101 |  |  |  | 0y1101 |  | 011y01 |  |  | - |
| 010100 |  |  |  |  |  |  |  |  |  |
| 111100 |  |  |  |  |  |  |  |  |  |
| 101100 |  |  |  |  | y01100 |  |  |  |  |
| xxx11x |  | 001y11 | 001y10 | 0011y1 | 0011y0 |  | 011y11 | 011y10 | 0111y1 |
| 1xx0xx | y01001 | y01011 | y01010 |  |  | y11001 | y11011 | y11010 |  |
| 1xxxx1 | y01001 | y01011 |  | y01101 |  | y11001 | y11011 |  | y11101 |

*Продолжение таблицы 3.2.2*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| C0\*C0 | 010100 | 111100 | 101100 | xxx11x | 1xx0xx | 1xxxx1 |
| 001001 |  |  |  |  |  |  |
| 001011 |  |  |  |  |  |  |
| 001010 |  |  |  |  |  |  |
| 001101 |  |  |  |  |  |  |
| 001100 |  |  |  |  |  |  |
| 011001 |  |  |  |  |  |  |
| 011011 |  |  |  |  |  |  |
| 011010 |  |  |  |  |  |  |
| 011101 |  |  |  |  |  |  |
| 010100 | - |  |  |  |  |  |
| 111100 |  | - |  |  |  |  |
| 101100 |  | 1y1100 | - |  |  |  |
| xxx11x | 0101y0 | 1111y0 | 1011y0 | - |  |  |
| 1xx0xx |  | 111y00 | 101y00 | 1xxy1x | - |  |
| 1xxxx1 |  | 11110y | 10110y |  |  | - |

В результате этой операции сформируется новое множество кубов:

C1 = {0010x1, 001x01, 0x1001, x01001, 00101x, 0x1011, 001x11, x01011, 0x1010, 001x10, x01010, 00110x, 0x1101, 0011x1, x01101, x01100, 0011x0, 0110x1, 011x01, x11001, 01101x, 011x11, x11011, 011x10, x11010, 0111x1, x11101, 0101x0, 1x1100, 1111x0, 111x00, 11110x, 1011x0, 101x00, 10110x, 1xxx1x, xxx11x, 1xx0xx, 1xxxx1}.

Множество Z0 кубов, не участвовавших в образовании новых кубов, пустое.

Во вложении в таблице 3.2.3 приведён следующий шаг поиска простых импликант с помощью операции С1\*С1.

В результате образовалось множество С2 кубов второй размерности:

C2 = {001xx1, 0x10x1, x010x1, 0x1x01, x01x01, xx1001, 0x101x, x0101x, 001x1x, xx1011, 0x1x11, x01x11, xx1010, 0x1x10, x01x10, x0110x, 0011xx, 0x11x1, xx1101, x011x1, x011x0, 011xx1, x110x1, x11x01, x1101x, 011x1x, x11x11, x11x10, x111x1, 1x11x0, 1x1x00, 1x110x, 111xx0, 1111xx, 111x0x, 101xx0, 1011xx, 101x0x, 1xxx1x, xxx11x, 1xx0xx, 1xxxx1}

Множество Z1 кубов, не участвовавших в образовании новых кубов, имеет вид: Z1 = {0101x0}

Во вложении в таблице 3.2.4 приведён следующий шаг поиска простых импликант с помощью операции С2\*С2.

В результате образовалось множество С3 кубов третьей размерности:

C3 = {0x1xx1, x01xx1, xx10x1, xx1x01, xx101x, 0x1x1x, x01x1x, xx1x11, xx1x10, x011xx, xx11x1, x11xx1, x11x1x, 1x1xx0, 1x11xx, 1x1x0x, 111xxx, 101xxx, 1xxx1x, xxx11x, 1xx0xx, 1xxxx1}.

Множество Z2 кубов, не участвовавших в образовании новых кубов имеет вид: Z2 = {0101x0}.

Во вложении в таблице 3.2.5 приведён следующий шаг поиска простых импликант с помощью операции С3\*С3.

В результате образовалось множество С4 кубов четвёртой размерности:

C4 = {xx1xx1, xx1x1x, 1x1xxx, 1xxx1x, xxx11x, 1xx0xx, 1xxxx1}.

Множество Z3 кубов, не участвовавших в образовании новых кубов имеет вид:

Z3 = {0101x0, x011xx}.

В таблице 3.2.6 приведён следующий шаг поиска простых импликант с помощью операции С4\*С4.

Таблица 3.2.6– Поиск простых импликант С4\*С4

|  |  |  |  |
| --- | --- | --- | --- |
| C4\*C4 | xx1xx1 | xx1x1x | 1x1xxx |
| xx1xx1 | - |  |  |
| xx1x1x |  | - |  |
| 1x1xxx |  |  | - |
| 1xxx1x |  |  |  |

Продолжение таблицы 3.2.6

|  |  |  |  |
| --- | --- | --- | --- |
| C4\*C4 | xx1xx1 | xx1x1x | 1x1xxx |
| xxx11x |  |  |  |
| 1xx0xx |  |  |  |
| 1xxxx1 |  |  |  |

Новых кубов (пятой размерности) не образовалось.

Получено множество

Z = {xx1xx1, xx1x1x, 1x1xxx}.

На этом заканчивается этап поиска простых импликант, т.к. |C5|≤1. Множество простых импликант:

Z =Z1 ⋃ Z2 ⋃ Z3 = {0101x0, x011xx, 1xxx1x, xxx11x, 1xx0xx, xx1xx1, xx1x1x, 1x1xxx, 1xxxx1}

Следующий этап – поиск L-экстремалей на множестве простых импликант (таблица 3.2.7). Для этого используется операция # (вычитание).

Таблица 3.2.7– Поиск *L*-экстремалей

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| z#(Z-z) | 0101x0 | x011xx | 1xxx1x | xxx11x | 1xx0xx | xx1xx1 | xx1x1x | 1x1xxx | 1xxxx1 |
| 0101x0 | - | x011xx | 1xxx1x | 1xx11x  x0x11x  xx111x  xxx111 | 1xx0xx | xx1xx1 | xx1x1x | 1x1xxx | 1xxxx1 |
| x011xx | 0101x0 | - | 11xx1x  1x0x1x  1xx01x | 11x11x  1x011x  x0011x  x1111x  x1x111  xx0111 | 1xx0xx | x11xx1  xx10x1 | x11x1x  xx101x | 111xxx  1x10xx | 11xxx1  1x0xx1  1xx0x1 |
| 1xxx1x | 0101x0 | 0011xx  x0110x | - | 00011x  01111x  01x111  0x0111 | 1xx00x | 011xx1  x11x01  0x10x1  xx1001 | 011x1x  0x101x | 111x0x  1x100x | 11xx01  1x0x01  1xx001 |
| xxx11x | 010100 | 00110x  x0110x | 11x01x  1x001x  1xx01x | - | 1xx00x | 0110x1  011x01  x11x01  0x10x1  xx1001 | 01101x  0x101x | 111x0x  1x100x | 11xx01  1x0x01  1xx001 |

*Продолжение таблицы 3.2.7*

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| z#(Z-z) | 010100 | 00110x  x0110x | 11x01x  1x001x  1xx01x | - | 1xx00x | 0110x1  011x01  x11x01  0x10x1  xx1001 | 01101x  0x101x | 111x0x  1x100x | 11xx01  1x0x01  1xx001 |
| 1xx0xx | 010100 | 00110x  x0110x | Ø | 00011x  01111x  01x111  0x0111 | - | 0110x1  011x01  011x01  x11101  0x10x1  0x1001 | 01101x  0x101x | 11110x | 11x101  1x0101 |
| xx1xx1 | 010100 | 1100  x01100 | Ø | 00011x  011110  010111  0x0111 | 1x000x  1xx000 | - | 011010  0x1010 | 111100 | 110101  1x0101 |
| xx1x1x | 010100 | 1100  x01100 | Ø | 00011x  010111  0x0111 | 1x000x  1xx000 | 011001  011x01  011x01  x11101  0x1001  0x1001 | - | 111100 | 110101  1x0101 |

В таблице 3.2.7 из каждой простой импликанты поочерёдно вычитаются все остальные простые импликанты *Z#(Z-z)*.

Получили кубы, “подозрительные” на *L*-экстремальность. Проверяем в таблице 3.2.8.

Таблица 3.2.8 – Проверка на *L-*экстремальность

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| z#(Z-z) n L | 001001 | 001011 | 001010 | 001101 | 001100 | 011001 | 011011 | 011010 | 011101 |
| 010100 | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø |
| 001100 | Ø | Ø | Ø | Ø | 001100 | Ø | Ø | Ø | Ø |
| 001100 | Ø | Ø | Ø | Ø | 001100 | Ø | Ø | Ø | Ø |
| 00011x | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø |
| 010111 | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø |
| 0x0111 | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø |
| 1x0000 | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø |
| 1x0000 | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø |
| 011001 | Ø | Ø | Ø | Ø | Ø | 011001 | Ø | Ø | Ø |
| 011x01 | Ø | Ø | Ø | Ø | Ø | 011001 | Ø | Ø | 011101 |
| 011x01 | Ø | Ø | Ø | Ø | Ø | 011001 | Ø | Ø | 011101 |

*Продолжение таблицы 3.2.8*

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| z#(Z-z) n L | 001001 | 001011 | 001010 | 001101 | 001100 | 011001 | 011011 | 011010 | 011101 |
| 011101 | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | 011101 |
| 0x1001 | 001001 | Ø | Ø | Ø | Ø | 011001 | Ø | Ø | Ø |
| 0x1001 | 001001 | Ø | Ø | Ø | Ø | 011001 | Ø | Ø | Ø |
| 011010 | Ø | Ø | Ø | Ø | Ø | Ø | Ø | 011010 | Ø |
| 0x1010 | Ø | Ø | 001010 | Ø | Ø | Ø | Ø | 011010 | Ø |
| 111100 | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø |
| 110101 | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø |
| 1x0101 | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø |

*Продолжение таблицы 3.2.8*

|  |  |  |  |
| --- | --- | --- | --- |
| z#(Z-z) n L | 010100 | 111100 | 101100 |
| 010100 | 010100 | Ø | Ø |
| 001100 | Ø | Ø | Ø |
| 001100 | Ø | Ø | Ø |
| 00011x | Ø | Ø | Ø |
| 010111 | Ø | Ø | Ø |
| 0x0111 | Ø | Ø | Ø |
| 1x0000 | Ø | Ø | Ø |
| 1x0000 | Ø | Ø | Ø |
| 011001 | Ø | Ø | Ø |
| 011x01 | Ø | Ø | Ø |
| 011x01 | Ø | Ø | Ø |
| 011101 | Ø | Ø | Ø |
| 0x1001 | Ø | Ø | Ø |
| 011010 | Ø | Ø | Ø |
| 0x1010 | Ø | Ø | Ø |
| 111100 | Ø | 111100 | Ø |
| 110101 | Ø | Ø | Ø |
| 1x0101 | Ø | Ø | Ø |

E = {0101x0, x011xx, xx1xx1, xx1x1x, 1x1xxx}.

Далее необходимо проанализировать, какие из исходных единичных кубов (множество L) не покрыты найденной L-экстремалью. Этот анализ осуществляется с помощью таблицы 3.2.9.

Таблица 3.2.9 – Поиск непокрытых наборов

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| L#E | 001001 | 001011 | 001010 | 001101 | 001100 | 011001 | 011011 | 011010 | 011101 |
| 0101x0 | 001001 | 001011 | 001010 | 001101 | 001100 | 011001 | 011011 | 011010 | 011101 |
| x011xx | 001001 | 001011 | 001010 | Ø | Ø | 011001 | 011011 | 011010 | 011101 |

*Продолжение таблицы 3.2.9*

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| L#E | 001001 | 001011 | 001010 | 001101 | 001100 | 011001 | 011011 | 011010 | 011101 |
| xx1xx1 | Ø | Ø | 001010 | Ø | Ø | Ø | Ø | 011010 | Ø |
| xx1x1x | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø |
| 1x1xxx | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø |

*Продолжение таблицы 3.2.9*

|  |  |  |  |
| --- | --- | --- | --- |
| L#E | 010100 | 111100 | 101100 |
| 0101x0 | Ø | 111100 | 101100 |
| x011xx | Ø | 111100 | Ø |
| xx1xx1 | Ø | 111100 | Ø |
| xx1x1x | Ø | 111100 | Ø |
| 1x1xxx | Ø | Ø | Ø |

Из таблицы 3.2.9 видим, что найденная *L-*экстремаль покрывает все исходные наборы. Поиск минимального покрытия завершён.

*.*

:

Запишем результат в базисе:

Функциональная схема ОЧУС приведена в приложении ГУИР.400201.306 Э2.2.

**3.3 Логический синтез преобразователя множителя**

Преобразователь множителя служит для исключения из множителя диады 11, заменяя её на триаду соответственно.

Принцип работы ПМ представлен с помощью таблицы истинности (таблицы 3.3.1).

Таблица 3.3.1 – Таблица истинности для преобразователя множителя

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Входная диада** | | **Младший бит** | **Перенос** | **Выходная диада** | | **Знак**  **S1S2** |
| Pn | Pn-1 | Pn-2 | Q | S1 | S2 | M |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |

*Продолжение таблицы 3.3.1*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Pn | Pn-1 | Pn-2 | Q | S1 | S2 | M |
| 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 | 0 | 0 | 0 |

Минимизируем функции .

Минимизацию функций проведем с помощью карт Карно. Для функции Р заполненная карта приведена на рисунке 3.3.1. В рисунках 3.3.1 – 3.3.1.

**Минимизация Q:**

-1, -2

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

Рисунок 3.3.1

Следовательно, .

**Минимизация :**

-1, -2

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

*Рисунок 3.3.2*

Следовательно, .

**Минимизация :**

-1, -2

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

*Рисунок 3.3.3*

Следовательно, .

**Минимизация M:**

-1, -2

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

*Рисунок 3.3.3*

Следовательно, .

Функциональная схема ПМ в приведена в приложении на чертеже ГУИР.400201.306 Э2.4.

**4 СИНТЕЗ КОМБИНАЦИОННЫХ СХЕМ НА ОСНОВЕ МУЛЬТИПЛЕКСОРА**

Мультиплексор – это логическая схема, имеющая n информационных входов, m управляющих входов и один выход. При этом должно выполняться условие n = 2m.

ОЧС от пяти переменных можно реализовать на мультиплексоре «один из восьми». Управляющее поле такого мультиплексора определяется тремя переменными, поэтому число групп с одинаковыми значениями этих переменных будет равно восьми.

Синтез ОЧC на базе мультиплексоров приведён в таблицах 4.1

Таблица 4.1 – синтез ОЧС на базе мультиплексоров

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***a1*** | ***a2*** | ***b1*** | ***b2*** | ***p*** | **П** | Функция | **S1** | Функция | **S2** | Функция |
| **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** |
| 0 | 0 | 0 | 0 | 0 | 0 | b2p | 0 | p | 0 | b2 |
| 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |  | 0 | b2+p |
| 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 |
| 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 0 | 0 | b2+p | 0 | b2 | 1 | b2+ |
| 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | b2+p | 0 | b2p |
| 0 | 1 | 1 | 0 | 1 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 0 | 0 | b2 | 1 | +p | 0 | p |
| 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 |
| 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 | 0 | b2p | 1 |  | 1 |  |
| 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 |

*Продолжение таблицы 4.1*

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** |
| 1 | 1 | 0 | 0 | 0 | 0 | b2 | 1 | b2+ | 1 | +p |
| 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 0 | 0 | 0 | b2+p | 0 | b2p | 1 |  |
| 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |

Функциональная схема ОЧС на базе мультиплексоров приведена в приложении на чертеже ГУИР.400201.529 Э2.5.

**ОЦЕНКА РЕЗУЛЬТАТОВ РАЗРАБОТКИ**

**Временные затраты на умножение**

Формула расчёта временных затрат умножения на один разряд:

Tу = TПМ + TФДК + TОЧУС ∙ (2n+1) + TОЧС ∙ 2 + Tсдвига, где

TОЧУС – время умножения на ОЧУС,

TОЧС – время формирования единицы переноса в ОЧС,

TПМ – время преобразования множителя,

TФДК – время формирования дополнительного кода множимого,

Tсдвига – время сдвига.

Тогда формула расчёта временных затрат на все разряды:

Tу = n ∙ (TПМ + TФДК + TОЧУС ∙ (2n+1) + TОЧС ∙ 2 + Tсдвига)

В конечном итоге минимизация функций позволила нам в несколько раз уменьшить затрату времени на выполнение умножения и удешевить схему сумматора-умножителя, за счёт уменьшения количества элементов.

**ЗАКЛЮЧЕНИЕ**

Цель курсового проекта по дисциплине «Арифметические и логические основы вычислительной техники» была достигнута, знания, полученные в результате выполнения, были успешно применены в теории и на практике, в ходе построения сумматора-умножителя.

В ходе разработки алгоритма умножения выявились достоинства и недостатки эти алгоритмов.

Достигнута оценка минимизации булевых функций путём карт Карно (Вейче) и алгоритмом извлечения (Рота). Главные достоинства карт Карно (Вейче) в простоте визуализации и понимания человеком. Главный недостаток – минимизация функций многих переменных. Алгоритмом извлечения в свою очередь можно минимизировать функции от любого числа переменных. Алгоритм является машинно-ориентированным, следуемо, легко программируемым. Для человека же он займёт достаточное количество времени.

Построение функциональных схем в нескольких базисах позволило закрепить полученные знания, такие как правило склеивания (де Моргана).

Осуществление построения схемы ОЧС на базе мультиплексоров помогло облегчить минимизацию этих функций и упростить саму функциональную схему.

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

Луцик, Ю. А. Учебное пособие по курсу «Арифметические и логические основы вычислительной техники» / Ю. А. Луцик, И. В. Лукьянова. - Минск : БГУиР, 2014. – 76с.

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

Единая система конструкторской документации (ЕСКД) : справ. пособие / С. С. Борушек [и др.]. – М. : Изд-во стандартов, 1989. – 352 с.

**ПРИЛОЖЕНИЕ А**

*(обязательное)*

Сумматор-умножитель первого типа. Схема электрическая структурная

**ПРИЛОЖЕНИЕ Б**

*(обязательное)*

Одноразрядный четверичный сумматор-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)*

Ведомость документов

**ПРИЛОЖЕНИЕ Е**

Таблица 3.2.3 – Поиск простых импликант С1\*С1

**ПРИЛОЖЕНИЕ Ж**

Таблица 3.2.4 – Поиск простых импликант С2\*С2

**ПРИЛОЖЕНИЕ З**

Таблица 3.2.5 – Поиск простых импликант С3\*С3