НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

# "КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ"

#### ФАКУЛЬТЕТ ІНФОРМАТИКИ ТА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

### Кафедра обчислювальної техніки

## РОЗРАХУНКОВА РОБОТА

по курсу „Комп'ютерна логіка-2”

Виконав: Бедь Анатолій Михайлович

Група ІО-12, Факультет ІОТ,

Залікова книжка № 1202

Номер технічного завдання 10010110001

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис керівника)

Київ – 2012 р.

**Завдання:**

1. Числа  і  в прямому коді записати у формі з плаваючою комою (з порядком і мантисою, а також з характеристикою та мантисою), як вони зберігаються у пам’яті. На порядок відвести 8 розрядів, на мантису 16 розрядів (з урахуванням знакових розрядів).

2. Виконати 8 операцій з числами  і  з плаваючою комою (чотири способи множення, два способи ділення, додавання додавання та віднімання). Номери операцій (для п.3) відповідають порядку переліку (наприклад, 6 – ділення другим способом). Для обробки мантис кожної операції, подати:

2.1 теоретичне обґрунтування способу;

2.1 операційну схему;

2.2 змістовний мікроалгоритм;

2.3 таблицю станів регістрів (лічильника), довжина яких забезпечує одержання 15 основних розрядів мантиси результату;

2.4 функціональну схему з відображенням управляючих сигналів;

2.5 закодований мікроалгоритм (мікрооперації замінюються управл. сигналами);

2.6 граф управляючого автомата Мура з кодами вершин;

2.7 обробку порядків (показати у довільній формі);

2.8 форму запису нормалізованого результату з плаваючою комою в пам’ять.

Вказані пункти для операцій додавання та віднімання виконати для етапу нормалізації результату з урахуванням можливого нулевого результату. Інші дії до етапу нормалізації результату можна проілюструвати у довільній формі.

3. Для операції з номером  побудувати управляючий автомат Мура на тригерах (тип вибрати самостійно) і елементах булевого базису.

**Визначення та обгрунтування варіанту:**

Перевести номер залікової книжки в двійкову систему. Записати два двійкових числа:

 і ,

де  - двійкові цифри номера залікової книжки у двійковій системі числення (- молодший розряд).

120210=100101100012;

 = - 10011011,1000101;

 = + 10101,0111000011;

**Ocновна частина:**

**Завдання №1**

=1. 10011011,1000101;

=0. 10101,0111000011;

Представлення чисел у формі з плаваючою точкою з порядком і мантисою:

X2:

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

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

Y2:

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

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

Представлення чисел у формі з плаваючою точкою з характеристикою і мантисою:

E = P + 2m ,

m = 7;

27 = 100000002

Ex = 10000000 + 1000 = 10001000

X2:

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

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

Ey = 10000000 + 101 = 10000101

Y2:

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

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

**Завдання №2**

**2.1 Перший спосіб множення.**

**2.1.1 Теоретичне обґрунтування першого способу множення:**

Числа множаться у прямих кодах, знакові та основні розряди обробляються окремо. Для визначення знака добутку здійснюють підсумування по модулю 2 цифр, що розміщуються в знакових розрядах співмножників.

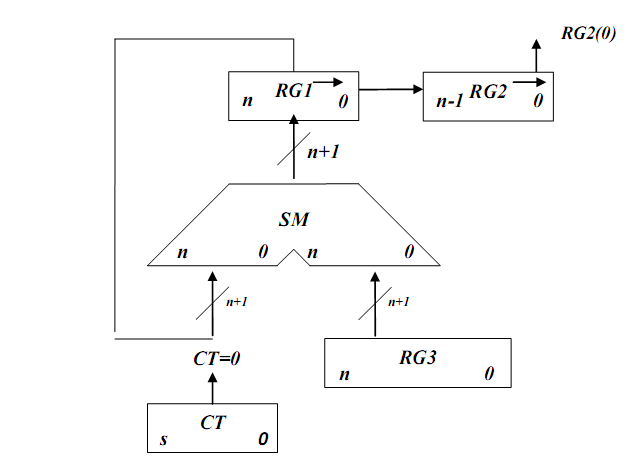
Множення мантис першим способом здійснюється з молодших розрядів множника, сума часткових добутків зсувається вправо, а множене залишається нерухомим. Тоді добуток двох чисел представляється у вигляді:

Z=YХ=+ Y…+ Y =

= ((..((0+Y)+ Y)+…+ Y) +…+ Y);

Z=;

**2.1.2 Операційна схема:**



*Рисунок 2.1.1- Операційна схема.*

**2.1.3 Змістовний мікроалгоритм:**

Початок

RG1:=0; RG2:=X; RG3:=Y; CT:=15;

RG2[0]

RG1:=RG1+RG3;

RG1:=0.r(RG1); RG2:=RG1(n).r(RG2); CT:=CT-1;

CT=0

Кінець

1

0

1

0

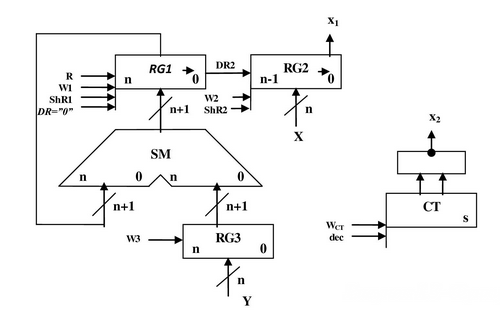
*Рисунок 2.1.2 - Змістовний мікроалгоритм виконання операції множення першим способом.*

**2.1.4 Таблиця станів регістрів:**

*Таблиця 2.1.1-Таблиця станів регістрів для першого способу множення.*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** | **RG1** | **RG2** | **RG3** | **CT** |
| **пс** | 0 | 10011011100010**1** | 0101010111000011 | 1111 |
| **1** | +101010111000011  =0101010111000011  0010101011100001 | 11001101110001**0** |  | 1110 |
| **2** | 001010101110000 | 11100110111000**1** |  | 1101 |
| **3** | +101010111000011  =0110101100110011  0011010110011001 | 11110011011100**0** |  | 1100 |
| **4** | 0001101011001100 | 11111001101110**0** |  | 1011 |
| **5** | 0000110101100110 | 01111100110111**0** |  | 1010 |
| **6** | 0000011010110011 | 00111110011011**1** |  | 1001 |
| **7** | +101010111000011  =0101110001110110  0010111000111011 | 00011111001101**1** |  | 1000 |
| **8** | +101010111000011  =1000001111111110  0100000111111111 | 00001111100110**1** |  | 0111 |
| **9** | +101010111000011  =1001011111000010  0100101111100001 | 00000111110011**0** |  | 0110 |
| **10** | 0010010111110000 | 10000011111001**1** |  | 0101 |
| **11** | +101010111000011  =0111101110110011  0011110111011001 | 11000001111100**1** |  | 0100 |
| **12** | +101010111000011  =1001001110011100  0100100111001110 | 01100000111110**0** |  | 0011 |
| **13** | 0010010011100111 | 00110000011111**0** |  | 0010 |
| **14** | 0001001001110011 | 10011000001111**1** |  | 0001 |
| **15** | +101010111000011  =0110100000110110  **0011010000011011** | **010011000001111** |  | 0000 |

**2.1.5 Функціональна схема:**



*Рисунок 2.1.3- Функціональна схема.*

**2.1.6 Закодований мікроалгоритм**

*Таблиця 2.1.2-Таблиця кодування операцій і логічних умов.*

|  |  |  |  |
| --- | --- | --- | --- |
| Кодування мікрооперацій | | Кодування логічних умов | |
| МО | УС | ЛУ | Позначення |
| G1:=0  RG2:=X  RG3:=Y  CT:=15  RG1:=RG1+RG3  RG1:=0.r(RG1)  RG2:=RG1[0].r(RG2)  CT:=CT-1 | R  W2  W3  WCT  W1  ShR1  ShR2  dec | RG2[0]  CT=0 | X1  X2 |

Початок

R, W2, W3, WCT

X1

W1

ShR1,ShR2,dec

X2

Кінець

1

0

1

0

Z1

Z2

Z3

Z4

Z5

*Рисунок 2.1.4-Закодований мікроалгоритм.*

**2.1.7 Граф управляючого автомата Мура з кодами вершин:**

-

X1

X2

000

100

101

111

110

*Рисунок 2.1.5-Граф автомата Мура*

**2.1.8 Обробка порядків:**

Порядок добутку буде дорівнювати сумі порядків множників з урахуванням знаку порядків:

=8; =5; =1310=11012

**2.1.9 Нормалізація результату:**

Отримали результат: **011010000011011010011000001111**

Знак мантиси: 1 0 = 1.

Робимо зсув результату вліво, доки у першому розряді не буде одиниця,

Порядок зменшуємо на 1:

**11010000011011010011000001111**;=12;

Запишемо нормалізований результат:

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

**2.2 Другий спосіб множення.**

**2.2.1 Теоретичне обґрунтування другого способу множення:**

Числа множаться у прямих кодах, знакові та основні розряди обробляються окремо. Визначення знака добутку здійснюють підсумування по модулю 2 цифр, що розміщуються в знакових розрядах співмножників.

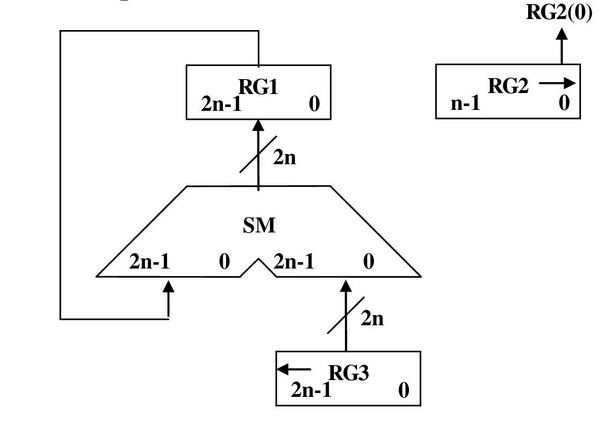
Множення мантис другим способом здійснюється з молодших розрядів, множене зсувається вліво, а сума часткових добутків залишається нерухомою.

Z=Y+ Y…+ Y;

Z=((0+ Y)+ Y)…+ Y;

Z=;

**2.2.2 Операційна схема:**



*Рисунок 2.2.1- Операційна схема.*

**2.2.3 Змістовний мікроалгоритм:**

RG1:=0;

RG2:=X;

RG3:=Y;

RG2[0]

RG1:=RG1+RG3;

RG2:=0.r(RG2); RG3:=l(RG1).0;

RX=0

Початок

Кінець

0

1

0

1

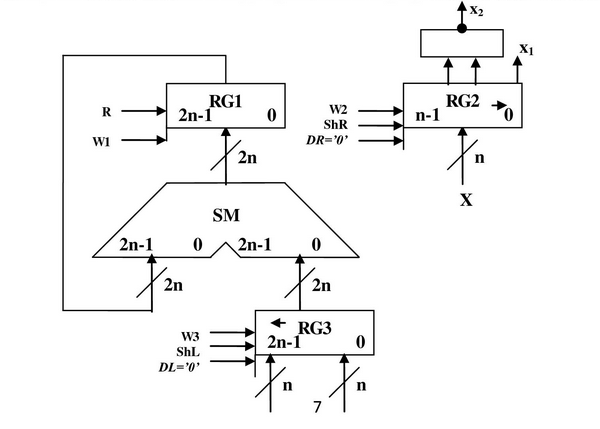
*Рисунок 2.2.2 - Змістовний мікроалгоритм.*

**2.2.4 Таблиця станів регістрів:**

*Таблиця 2.2.1-Таблиця станів регістрів.*

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **RG1** | **RG3 ←** | **RG2 →** |
| **пс** | 0 | 000000000000000101010111000011 | 10011011100010**1** |
| **1** | +000000000000000101010111000011  =000000000000000101010111000011 | 000000000000001010101110000110 | 01001101110001**0** |
| **2** | 000000000000000101010111000011 | 000000000000010101011100001100 | 00100110111000**1** |
| **3** | +000000000000010101011100001100  =000000000000011010110011001111 | 000000000000101010111000011000 | 00010011011100**0** |
| **4** | 000000000000011010110011001111 | 000000000001010101110000110000 | 00001001101110**0** |
| **5** | 000000000000011010110011001111 | 000000000010101011100001100000 | 00000100110111**0** |
| **6** | 000000000000011010110011001111 | 000000000101010111000011000000 | 00000010011011**1** |
| **7** | +000000000101010111000011000000  =000000000101110001110110001111 | 000000001010101110000110000000 | 00000001001101**1** |
| **8** | +000000001010101110000110000000  =000000010000011111111100001111 | 000000010101011100001100000000 | 00000000100110**1** |
| **9** | +000000010101011100001100000000  =000000100101111100001000001111 | 000000101010111000011000000000 | 00000000010011**0** |
| **10** | 000000100101111100001000001111 | 000001010101110000110000000000 | 00000000001001**1** |
| **11** | +000001010101110000110000000000  =000001111011101100111000001111 | 000010101011100001100000000000 | 00000000000100**1** |
| **12** | +000010101011100001100000000000  =000100100111001110011000001111 | 000101010111000011000000000000 | 00000000000010**0** |
| **13** | 000100100111001110011000001111 | 001010101110000110000000000000 | 00000000000001**0** |
| **14** | 000100100111001110011000001111 | 010101011100001100000000000000 | 00000000000000**1** |
| **15** | +010101011100001100000000000000  =**011010000011011010011000001111** | 101010111000011000000000000000 | 000000000000000 |

**2.2.5 Функціональна схема:**



*Рисунок 2.2.3- Функціональна схема.*

**2.2.6 Закодований мікроалгоритм**

*Таблиця 2.2.2-Таблиця кодування операцій і логічних умов.*

|  |  |  |  |
| --- | --- | --- | --- |
| Кодування мікрооперацій | | Кодування логічних умов | |
| МО | УС | ЛУ | Позначення |
| RG1:=0  RG2:=X  RG3:=Y  RG1:=RG1+RG3  RG2:=0.r(PG2)  RG3:=l(RG3).0 | R  W2  W3  W1  ShR  ShL | RG2[0]  RG2=0 | X1  X2 |

0

0

Початок Z1

R,W2,W3 Z2

X1

W1 Z3

ShR, ShL Z4

X2

1

Кінець Z5

*Рисунок 2.2.4-Закодований мікроалгоритм.*

**2.2.7 Граф управляючого автомата Мура з кодами вершин:**

-

X1

X2

000

100

101

111

110

*Рисунок 2.2.5 - Граф автомата Мура*

**2.2.8 Обробка порядків:**

Порядок добутку буде дорівнювати сумі порядків множників з урахуванням знаку порядків:

=8; =5; =1310=11012

**2.2.9 Нормалізація результату:**

Отримали результат: 011010000011011010011000001111

Знак мантиси: 1 0 = 1.

Робимо здвиг результату вліво, доки у першому розряді не буде одиниця,

Порядок зменшуємо на 1:

11010000011011010011000001111;=12;

Запишемо нормалізований результат:

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

**2.3 Третій спосіб множення.**

**2.3.1Теоретичне обгрунтування третього способу множення:**

Числа множаться у прямих кодах, знакові та основні розряди обробляються окремо. Визначення знака добутку здійснюють підсумування по модулю 2 цифр, що розміщуються в знакових розрядах співмножників.

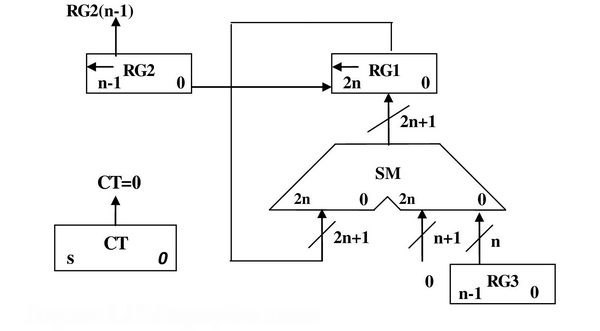
Множення мантис третім способом здійснюється зі старших розрядів множника, сума часткових добутків і множник зсуваються вліво, а множене нерухоме.

Z=Y+ Y…+ Y;

Z= Y+2(Y+2(Y…+2Y));

Z=;

**2.3.2 Операційна схема:**



*Рисунок 2.3.1 - Операційна схема*

**2.3.3 Змістовний мікроалгоритм:**

RG1:=0;

RG2:=X;

RG3:=Y;

CT:=n;

RG2[n-1]

RG1:=RG1+RG3;

RG1:=l(RG1).0;

RG2:=l (RG2).0;

CT:=CT-1;

CT=0

Початок

Кінець

1

1

0

0

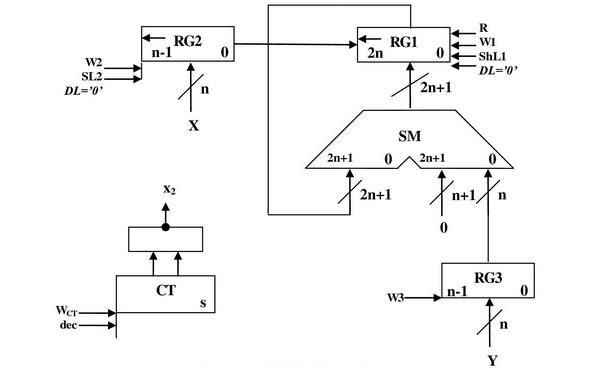
*Рисунок 2.3.2* **-** *Змістовний мікроалгоритм.*

**2.3.4 Таблиця станів регістрів:**

*Таблиця 2.3.1- Таблиця станів регістрів*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** | **RG1 ←** | **RG2 ←** | **RG3** | **CT** |
| **пс** | 000000000000000000000000000000 | **1**00110111000101 | 101010111000011 | 1111 |
| **1** | +000000000000000101010111000011  =000000000000000101010111000011  000000000000001010101110000110 | **0**01101110001010 |  | 1110 |
| **2** | 000000000000010101011100001100 | **0**11011100010100 |  | 1101 |
| **3** | 000000000000101010111000011000 | **1**10111000101000 |  | 1100 |
| **4** | +000000000000000101010111000011  =000000000000110000001111011011  000000000001100000011110110110 | **1**01110001010000 |  | 1011 |
| **5** | +000000000000000101010111000011  =000000000001100101110101111001  000000000011001011101011110010 | **0**11100010100000 |  | 1010 |
| **6** | 000000000110010111010111100100 | **1**11000101000000 |  | 1001 |
| **7** | +000000000000000101010111000011  =000000000110011100101110100111  000000001100111001011101001110 | **1**10001010000000 |  | 1000 |
| **8** | +000000000000000101010111000011  =000000001100111110110100010001  000000011001111101101000100010 | **1**00010100000000 |  | 0111 |
| **9** | +000000000000000101010111000011  =000000011010000010111111100101  000000110100000101111111001010 | **0**00101000000000 |  | 0110 |
| **10** | 000001101000001011111110010100 | **0**01010000000000 |  | 0101 |
| **11** | 000011010000010111111100101000 | **0**10100000000000 |  | 0100 |
| **12** | 000110100000101111111001010000 | **1**01000000000000 |  | 0011 |
| **13** | +000000000000000101010111000011  =000110100000110101010000010011  001101000001101010100000100110 | **0**10000000000000 |  | 0010 |
| **14** | 011010000011010101000001001100 | **10**0000000000000 |  | 0001 |
| **15** | +000000000000000101010111000011  =011010000011011010011000001111 | 000000000000000 |  | 0 |

**2.3.5 Функціональна схема:**



*Рисунок 2.3.3* **-** *Функціональна схема.*

**2.3.6 Закодований мікроалгоритм:**

*Таблиця 2.3.2-Таблиця кодування операцій і логічних умов.*

|  |  |  |  |
| --- | --- | --- | --- |
| Кодування мікрооперацій | | Кодування логічних умов | |
| МО | УС | ЛУ | Позначення |
| RG1:=0  RG2:=X  RG3:=Y  CT:=15  RG1:=RG1+RG3  RG1:=l(RG1).0  RG2:=l(RG2).0  CT:=CT-1 | R  W2  W3  WCT  W1  ShL1  ShL2  dec | RG2[n-1]  CT=0 | X1  X2 |

Початок

R, W2, W3, WCT

X1

W1

ShL1,ShL2,dec

X2

Кінець

1

0

1

0

Z1

Z2

Z3

Z4

Z5

*Рисунок 2.3.4-Закодований мікроалгоритм.*

**2.3.7 Граф управляючого автомата Мура з кодами вершин:**

-

X1

X2

000

100

101

111

110

*Рисунок 2.3.5 - Граф автомата Мура*

**2.3.8 Обробка порядків:**

Порядок добутку буде дорівнювати сумі порядків множників з урахуванням знаку порядків:

=8; =5; =1310=11012

**2.3.9 Нормалізація результату:**

Отримали результат: 011010000011011010011000001111

Знак мантиси: 1 0 = 1.

Робимо здвиг результату вліво, доки у першому розряді не буде одиниця,

порядок зменшуємо на 1:

11010000011011010011000001111;=12;

Запишемо нормалізований результат:

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

**2.4 Четвертий спосіб множення.**

**2.4.1Теоритичне обґрунтування четвертого способу множення:**

Числа множаться у прямих кодах, знакові та основні розряди обробляються окремо. Визначення знака добутку здійснюють підсумування по модулю 2 цифр, що розміщуються в знакових розрядах співмножників.

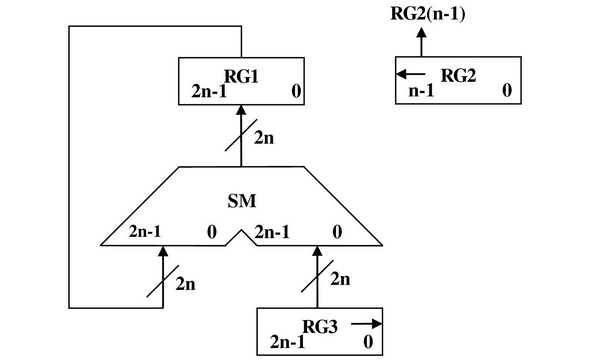
Множення здійснюється зі старших розрядів множника, сума часткових добутків залишається нерухомою, множене зсувається праворуч, множник ліворуч.

.

*.*

з початковими значеннями i=1, Y0=2-1Y, Z0=0.

**2.4.2 Операційна схема:**



*Рисунок 2.4.1-* *Операційна схема*

**2.4.3 Змістовний мікроалгоритм:**

RG1:=0;

RG2:=X;

RG3:=Y;

RG3:=0.r(RG3)

RG2[n-1]

RG1:=RG1+RG3;

RG3:=0.r(RG3)

RG2:=l(RG2).0

RG2=0

Початок

Кінець

1

1

0

0

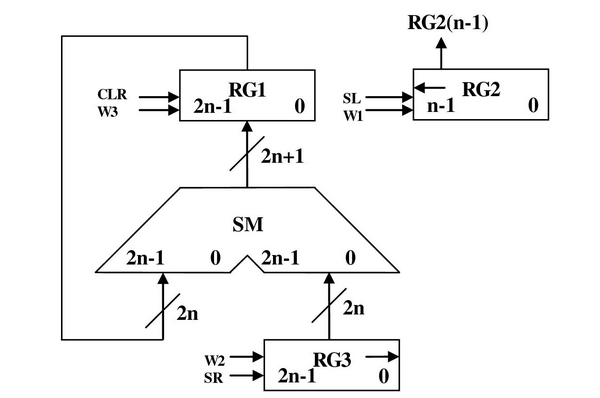
*Рисунок 2.4.2* **-** *Змістовний мікроалгоритм.*

**2.4.4 Таблиця станів регістрів:**

*Таблиця 2.4.1- Таблиця станів регістрів*

|  |  |  |  |
| --- | --- | --- | --- |
| № | RG1 | RG3 → | RG2 ← |
| ПС | 000000000000000000000000000000 | 010101011100001100000000000000 | **1**00110111000101 |
| **1** | +010101011100001100000000000000  =010101011100001100000000000000 | 001010101110000110000000000000 | **0**01101110001010 |
| **2** | 010101011100001100000000000000 | 000101010111000011000000000000 | **0**11011100010100 |
| **3** | 010101011100001100000000000000 | 000010101011100001100000000000 | **1**10111000101000 |
| **4** | +000010101011100001100000000000  =011000000111101101100000000000 | 000001010101110000110000000000 | **1**01110001010000 |
| **5** | +000001010101110000110000000000  =011001011101011110010000000000 | 000000101010111000011000000000 | **0**11100010100000 |
| **6** | 011001011101011110010000000000 | 000000010101011100001100000000 | **1**11000101000000 |
| **7** | +000000010101011100001100000000  =011001110010111010011100000000 | 000000001010101110000110000000 | **1**10001010000000 |
| **8** | +000000001010101110000110000000  =011001111101101000100010000000 | 000000000101010111000011000000 | **1**00010100000000 |
| **9** | +000000000101010111000011000000  =011010000010111111100101000000 | 000000000010101011100001100000 | **0**00101000000000 |
| **10** | 011010000010111111100101000000 | 000000000001010101110000110000 | **0**01010000000000 |
| **11** | 011010000010111111100101000000 | 000000000000101010111000011000 | **0**10100000000000 |
| **12** | 011010000010111111100101000000 | 000000000000010101011100001100 | **1**01000000000000 |
| **13** | +000000000000010101011100001100  =011010000011010101000001001100 | 000000000000001010101110000110 | **0**10000000000000 |
| **14** | 011010000011010101000001001100 | 000000000000000101010111000011 | **1**00000000000000 |
| **15** | +000000000000000101010111000011  =**011010000011011010011000001111** | 000000000000000010101011100001 | 000000000000000 |

**2.4.5Функціональна схема:**



*Рисунок 2.4.3* **-** *Функціональна схема.*

**2.4.6 Закодований мікроалгоритм**

*Таблиця 2.4.2-Таблиця кодування операцій і логічних умов.*

|  |  |  |  |
| --- | --- | --- | --- |
| Кодування мікрооперацій | | Кодування логічних умов | |
| МО | УС | ЛУ | Позначення |
| RG1:=0  RG2:=X  RG3:=Y RG1:=RG1+RG3  RG3:=0.r(RG3) RG2:=l(RG2).0 | R  W2  W3  W1  ShR  ShL | RG2[n-1]  RG2=0 | X1  X2 |

Початок

Z1

R, W2, W3, ShR

X1

ShR,ShL

X2

Кінець

1

0

1

0

Z2

Z3

Z4

Z5

W1

*Рисунок 2.4.4-Закодований мікроалгоритм.*

**2.4.7 Граф управляючого автомата Мура з кодами вершин:**

-

X1

X2

000

100

101

111

110

*Рисунок 2.4.5 - Граф автомата Мура*

**2.4.8 Обробка порядків:**

Порядок добутку буде дорівнювати сумі порядків множників з урахуванням знаку порядків:

=8; =5; =1310=11012

**2.4.9 Нормалізація результату:**

Отримали результат: 011010000011011010011000001111

Знак мантиси: 1 0 = 1.

Робимо здвиг результату вліво, доки у першому розряді не буде одиниця,

Порядок понижаємо на 1:

11010000011011010011000001111;=12;

Запишемо нормалізований результат:

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

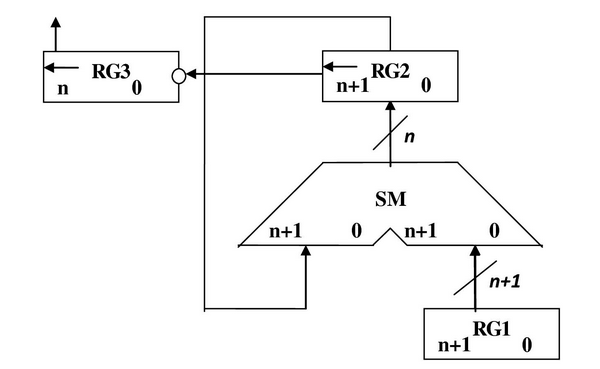
**2.5. Першиий спосіб ділення.**

**2.5.1Теоритичне обґрунтування першого способу ділення:**

Нехай ділене Х і дільник Y є n-розрядними правильними дробами, поданими в прямому коді. В цьому випадку знакові й основні розряди операндів обробляються окремо. Знак результату визначається шляхом підсумовування по модулю 2 цифр, записаних в знакових розрядах.

При реалізації ділення за першим методом здійснюється зсув вліво залишку при нерухомому дільнику. Черговий залишок формується в регістрі RG2 (у вихідному стані в цьому регістрі записаний Х). Виходи RG2 підключені до входів СМ безпосередньо, тобто ланцюги видачі коду з RG2 не потрібні. Час для підключення n+1 цифри частки визначається виразом t=(n+1)(tt+tc), де tt - тривалість виконання мікрооперації додавання-віднімання; tc - тривалість виконання мікрооперації зсуву.

**2.5.2 Операційна схема:**



*Рисунок 2.5.1-* *Операційна схема*

**2.5.3 Змістовний мікроалгоритм:**

Початок

RG3:=0

RG2:=X

RG1:=Y

RG2[n+1]

Кінець

RG3:=l(RG3).

RG2:=l(RG2).0

RG2:=RG2+RG1

RG2:=RG2++1

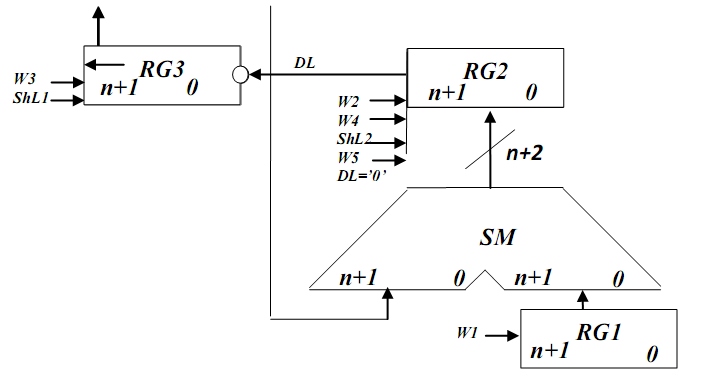
RG2[n+1]

*Рисунок 2.5.2-Змістовний мікроалгоритм*

**2.5.4 Таблиця станів регістрів:**

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **RG3(Z)** | **RG2(X)** | **RG1(Y)** |
| **пс** | 000000000000000 | **0**0100110111000101 | 00101010111000011 |
| **1** | 0000000000000001 | 01001101110001010  +11010101000111101  =**0**0100010111000111 |  |
| **2** | 0000000000000011 | 01000101110001110  +11010101000111101  =**0**0011010111001011 |  |
| **3** | 0000000000000111 | 00110101110010110  +11010101000111101  =**0**0001010111010011 |  |
| **4** | 0000000000001111 | 00010101110100110  +11010101000111101  =**1**1101010111100011 |  |
| **5** | 0000000000011110 | 11010101111000110  +00101010111000011  =**0**0000000110001001 |  |
| **6** | 0000000000111101 | 00000001100010010  +11010101000111101  =**1**1010110101001111 |  |
| **7** | 0000000001111010 | 10101101010011110  +00101010111000011  =**1**1011000001100001 |  |
| **8** | 0000000011110100 | 10110000011000010  +00101010111000011  =**1**1011011010000101 |  |
| **9** | 0000000111101000 | 10110110100001010  +00101010111000011  =**1**1100001011001101 |  |
| **10** | 0000001111010000 | 11000010110011010  +00101010111000011  =**1**1101101101011101 |  |
| **11** | 0000011110100000 | 11011011010111010  +00101010111000011  =**0**0000110001111101 |  |
| **12** | 0000111101000001 | 00001100011111010  +11010101000111101  =**1**1100001100110111 |  |
| **13** | 0001111010000010 | 11000011001101110  +00101010111000011  =**1**1101110000110001 |  |
| **14** | 0011110100000100 | 11011100001100010  +00101010111000011  =**0**0000111000100101 |  |
| **15** | 0111101000001001 | 00001110001001010  +11010101000111101  =**1**1100011010000111 |  |
| **16** | **1111010000010010** | 11000110100001110  +00101010111000011  =11110001011010001 |  |

**2.5.5 Функціональна схема:**



*Рисунок 2.5.3 – Функціональна схема*

**2.5.6 Закодований мікроалгоритм**

*Таблиця 2.5.2-Таблиця кодування операцій і логічних умов.*

|  |  |  |  |
| --- | --- | --- | --- |
| Кодування мікрооперацій | | Кодування логічних умов | |
| МО | УС | ЛУ | Позначення |
| RG3:=0  RG2:=X;  RG1:=Y;  RG3:=l(RG3).RG2[n+1]  RG2:=l(RG2).0  RG2:=RG2+RG1+1  RG2:=RG2+RG1 | W3  W2  W1  ShL1  ShL2  W4  W5 | RG2[n-1]  RG2=0 | X1  X2 |

Z1

Початок

Z2

W3, W2, W1

Z33

ShL1, ShL2

1

0

X1

Z5

Z4

W4

W5

1

X2

Z6

0

Кінець

*Рисунок 2.5.4-Закодований мікроалгоритм.*

**2.5.7 Граф управляючого автомата Мура з кодами вершин:**

011

010

000

100

001

-

**-**

101

*Рисунок 2.5.5 - Граф управляючого автомата.*

**2.5.8 Обробка порядків:**

Порядок частки буде дорівнювати:

В моєму випадку =8; =5; =3;

**2.5.8 Нормалізація результату:**

Отримали результат: 1111010000010010

Знак мантиси: 1 0 = 1.

Нормалізація мантиси не потрібна.

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

**2.6. Другий спосіб ділення.**

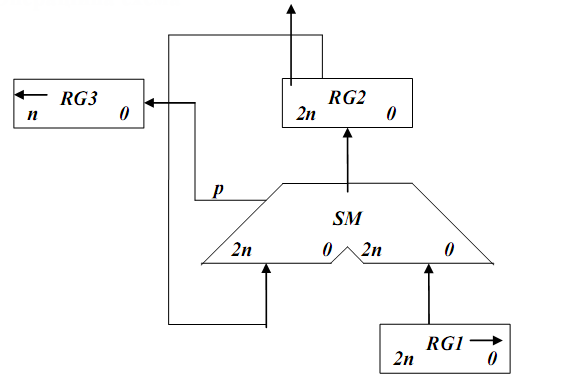
**2.6.1Теоритичне обгрунтування другого способу ділення:**

Нехай ділене Х і дільник Y є n-розрядними правильними дробами, поданими в прямому коді. В цьому випадку знакові й основні розряди операндів обробляються окремо. Знак результату визначається шляхом підсумовування по модулю 2 цифр, записаних в знакових розрядах.

Остача нерухома, дільник зсувається праворуч. Як і при множенні з нерухомою сумою часткових добутків можна водночас виконувати підсумування і віднімання, зсув в регістрах Y,Z. Тобто 1 цикл може складатися з 1 такту, це дає

прискорення відносно 1-го способу.

**2.6.2 Операційна схема**



*Рисунок 2.6.1-Операційна схема*

**2.6.3 Змістовний мікроалгоритм**

Початок

RG3:=0

RG1:=Y

RG2=X

RG2[2n+1]

RG2:=RG2++1

RG1:=0.r(RG1)

RG3:=l.(RG3).SM(p)

RG2:=RG2+RG1

RG1:=0.r(RG1)

RG3:=l(RG3).SM(p)

RG3[n]

Кінець

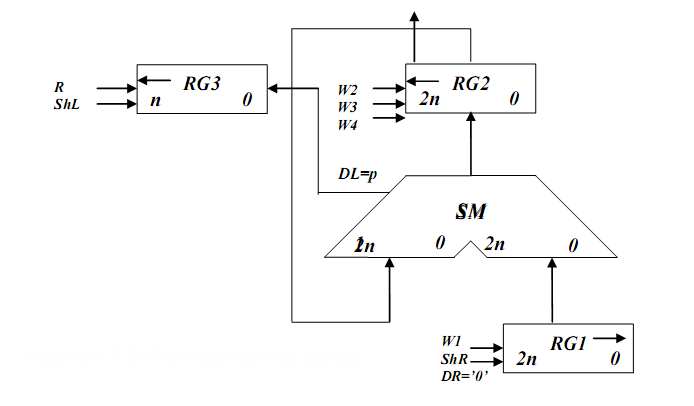
*Рисунок 2.6.2-Змістовний мікроалгоритм*

**2.6.4 Таблиця станів регістрів**

*Таблиця 2.6.1- Таблиця станів регістрів*

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **RG3(Z)** | **RG2(X)** | **RG1(Y)** |
| **пс** | 0000000000000001 | 010011011100010100000000000000 | 001010101110000110000000000000 |
| **1** | 0000000000000011 | 010011011100010100000000000000  +110101010001111010000000000000  =001000101110001110000000000000 | 000101010111000011000000000000 |
| **2** | 0000000000000111 | 001000101110001110000000000000  +111010101000111101000000000000  =000011010111001011000000000000 | 000010101011100001100000000000 |
| **3** | 0000000000001111 | 000011010111001011000000000000  +111101010100011110100000000000  =000000101011101001100000000000 | 000001010101110000110000000000 |
| **4** | 0000000000011110 | 000000101011101001100000000000  +111110101010001111010000000000  =111111010101111000110000000000 | 000000101010111000011000000000 |
| **5** | 0000000000111101 | 111111010101111000110000000000  +000000101010111000011000000000  =000000000000110001001000000000 | 000000010101011100001100000000 |
| **6** | 0000000001111010 | 000000000000110001001000000000  +111111101010100011110100000000  =111111101011010100111100000000 | 000000001010101110000110000000 |
| **7** | 0000000011110100 | 111111101011010100111100000000  +000000001010101110000110000000  =111111110110000011000010000000 | 000000000101010111000011000000 |
| **8** | 0000000111101000 | 111111110110000011000010000000  +000000000101010111000011000000  =111111111011011010000101000000 | 000000000010101011100001100000 |
| **9** | 0000001111010000 | 111111111011011010000101000000  +000000000010101011100001100000  =111111111110000101100110100000 | 000000000001010101110000110000 |
| **10** | 0000011110100000 | 111111111110000101100110100000  +000000000001010101110000110000  =111111111111011011010111010000 | 000000000000101010111000011000 |
| **11** | 0000111101000001 | 111111111111011011010111010000  +000000000000101010111000011000  =000000000000000110001111101000 | 000000000000010101011100001100 |
| **12** | 0001111010000010 | 000000000000000110001111101000  +111111111111101010100011110100  =111111111111110000110011011100 | 000000000000001010101110000110 |
| **13** | 0011110100000100 | 111111111111110000110011011100  +000000000000001010101110000110  =111111111111111011100001100010 | 000000000000000101010111000011 |
| **14** | 0111101000001001 | 111111111111111011100001100010  +000000000000000101010111000011  =000000000000000000111000100101 | 000000000000000010101011100001 |
| **15** | **1111010000010010** | 000000000000000000111000100101  +111111111111111101010100011111  =111111111111111110001101000100 | 000000000000000001010101110000 |

**2.6.5 Функціональна схема з відображенням управляючих сигналів**



*Рисунок 2.6.3-Функціональна схема*

**2.6.6 Закодований мікроалгоритм**

*Таблиця 2.6.2- Таблиця кодування мікрооперацій*

|  |  |  |  |
| --- | --- | --- | --- |
| Таблиця кодування мікрооперацій | |  | Таблиця кодування логічних умов |
| МО | УС | ЛУ | Позначення |
| RG3:=0  RG1:=Y  RG2:=X  RG2:=RG2+RG1  RG1:=0.r(RG1)  RG3:=l(RG3).SM(p)  RG2:==RG2++1 | R  W1  W2  W3  ShR  ShL  W4 | RG2[2n+1]  RG3[n] | X1  X2 |

Початок

R, W1, W2

X1

Кінець

W4, ShR, ShL

W3, ShR, ShL

X2

*Рисунок 2.6.4- Закодований мікроалгоритм*

**2.6.7 Граф управляючого автомата Мура з кодами вершин**

011

010

000

100

-

101

*Рисунок 2.6.5- Граф автомата Мура*

**2.6.8 Обробка порядків:**

Порядок частки буде дорівнювати:

В моєму випадку =8; =5; =3;

**2.6.9 Нормалізація результату:**

Отримали результат: 1111010000010010

Знак мантиси: 1 0 = 1.

Нормалізація мантиси не потрібна.

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

**2.7. Операція додавання та віднімання чисел.**

**2.7.1 Теоретичне обґрунтування способу**

В пам’яті числа зберігаються у ПК. На першому етапі додавання чисел з плаваючою комою виконують вирівнювання порядків до числа із старшим порядком. На другому етапі виконують додавання мантис. Додавання мантис виконується у доповнювальних кодах, при необхідності числа у ДК переводяться в АЛП. Додавання виконується порозрядно на n-розрядному суматорі з переносом. Останній етап – нормалізація результату. Виконується за допомогою зсуву мантиси результату і коригування порядку результату. Порушення нормалізації можливо вліво і вправо, на 1 розряд вліво і на n розрядів вправо. Віднімання будемо виконувати як операцію додавання але змінивши знак від’ємника на протилежний Z = X - Y = X+(-Y)

1. Порівняння порядків.

Px=+810=+10002

Py=+510=+01012

810-510=310=112

2. Вирівнювання порядків.

Робимо зсув вправо мантиси числа Y, зменшуючи на кожному кроці, доки не стане 0.

*Таблиця 2.7.1- Таблиця зсуву мантиси на етапі вирівнювання порядків*

|  |  |  |
| --- | --- | --- |
| MY | ∆ | Мікрооперація |
| 0,101010111000011 | 11 | Початковий стан |
| 0,010101011100001 | 10 | My= 0.r(My); ∆=∆-1 |
| 0,001010101110000 | 01 | My= 0.r(My); ∆=∆-1 |
| 0,000101010111000 | 00 | My= 0.r(My); ∆=∆-1 |

3. Додавання мантис у модифікованому ДК.

X мдк = 11.011001000111011

Yмдк = 00. 000101010111000

*Таблиця 2.7.2-Додавання мантис(для додавання)*

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| MX | 1 | 1, | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 |
| MY | 0 | 0, | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 |
| MZ | 1 | 1, | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 |

Zпк = 1.100001100001101

4. Віднімання мантис у модифікованому ДК.

X мдк = 11.011001000111011

Yмдк : = - Yмдк = 11. 111010101001000

*Таблиця 2.7.3-Додавання мантис(для віднімання)*

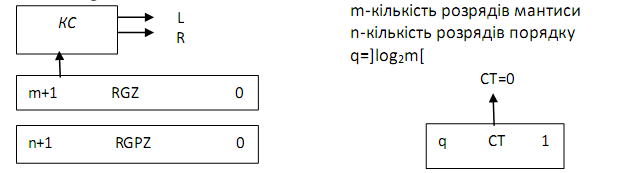
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| MX | 1 | 1, | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 |
| MY | 1 | 1, | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| MZ | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |

Zпк = 1.101100001111100

5. Нормалізація результату (В ПК).

Для даних результатів додавання та віднімання нормалізація не потрібна.

**2.7.2 Операційна схема**



*Рисунок 2.7.1-Операційна схема*

Виконаємо синтез КС для визначення порушення нормалізації.

*Таблиця 2.7.4-Визначення порушення нормалізації*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Розряди регістру  RGZ | | | Значення  функцій | |
| Z’0 | Z0 | Z1 | L | R |
| 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 1 | 0 |



Результат беремо по модулю, знак встановлюємо за Z’0 до нормалізації.

**2.7.3 Змістовний алгоритм**

Початок

CT:=m;

RGZ:=Z;

L=Z0

Кінець

RGZ:=l(RGZ).0

RGPZ:=RGPZ-1

CT:=CT-1

RGZ:=RGZ(m+2).r(RGZ)

RGZP:=RGZP+1

R=

CT=0

1

0

Z’0=0

Z’0 Z0:=

*Рисунок 2.7.2-Змістовний мікроалгоритм*

**2.7.4 Таблиця станів регістрів**

**1) Додавання**

*Таблиця 2.7.5- Таблиця станів регістрів*

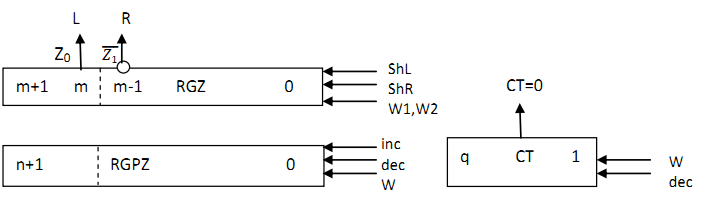
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№**  **такту** | **RGPZ** | **RGZ** | **ЛПН(L)** | **ППН(R)** | **СT** | **Мікрооперація** |
| **ПС** | 001000 | 11.100001100001101 | 0 | 0 | 100 |  |

**2)Віднімання**

*Таблиця 2.7.6- Таблиця станів регістрів*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№**  **такту** | **RGPZ** | **RGZ** | **ЛПН(L)** | **ППН(R)** | **СT** | **Мікрооперація** |
| **ПС** | 001000 | 11.101100001111100 | 0 | 0 | 100 |  |

**2.7.5 Функціональна схема з відображенням керуючих сигналів**

*****Рисунок 2.7.3 – Функціональна схема*

**2.7.6 Закодований мікроалгоритм**

*Таблиця 2.7.7– Таблиця кодування*

|  |  |  |  |
| --- | --- | --- | --- |
| Таблиця кодування мікрооперацій | |  | Таблиця кодування логічних умов |
| МО | УС | ЛУ | Позначення |
| CT:=m  RGZ:=Z  Z’0 Z0:=  RGZ:=RGZ(m+2).r(RGZ)  RGPZ:=RGPZ+1  RGZ:=l(RGZ).0  RGPZ:=RGPZ-1  CT:=CT-1 | W  W1  W2  ShR  Inc  ShL  decRGZ  decCT | Z’0=0  L=Z0  R=  CT=0 | X1  X2  X3  X4 |

1

1

0

0

0

Z6

Z2

Z3

Z4

Z5

Z1

1

Початок

W, W1

X2

Кінець

ShL, decRGZ, decCT

ShR, Inc

X3

X4

1

0

X1

W2

*Рисунок 2.7.4 – Закодований мікроалгоритм*

**2.7.7 Граф управляючого автомата Мура з кодами вершин**



*Рисунок 2.7.5 – Граф автомата Мура*

**2.7.8 Обробка порядків**

1) PX+Y= 810 =10002

2) PX-Y = 810 = 10002

**2.7.9 Форма запису результату з плаваючою комою**

1) Результат додавання Z=X+Y.

Zпк = 1.100001100001101

Pz = 810 =10002 Mz = 1000011001100012

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

2) Результат віднімання Z=X-Y.

Zпк = 1.101100001111100

Pz = 810 =10002 Mz = 1011000011111002

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

**Завдання 3**

x3 x2 x1 = 0012 = 110.

Синтез управляючого автомату Мура на тригерах для операції множення першим способом

**3.1 Таблиця кодування сигналів**

*Таблиця 3.1 – Таблиця кодування сигналів*

|  |  |
| --- | --- |
| R, W2, W3, WCT | Y1 |
| W1 | Y2 |
| ShR1,ShR2,dec | Y3 |

**3.2 Мікроалгоритм в термінах управляючого автомата**

Початок

Y1

X1

Y2

Y3

X2

Кінець

1

0

1

0

Z1

Z2

Z3

Z4

Z1

*Рисунок 3.1 – Закодований мікроалгоритм*

**3.3 Граф автомата**

00

10

11

01

-

X1

X2

*Рисунок 3.2 – Граф циклічного автомата*

**3.4 Таблиця переходів циклічного автомата на D-тригерах**

*Таблиця 3.2 – Таблиця переходів*

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Пер. | | Ст. ст. | Нов. стан | Вх. сигн. | Вих. сигн. | Функції тригерів | | |
| Q2Q1 | Q2Q1 | X2X1 | Y1Y2Y3 | D2 | D1 | |
| Z1→Z2 | 00 | | 10 | - - | 0 0 0 | 1 | | 0 |
| Z2→Z3 | 10 | | 11 | - 1 | 1 0 0 | 1 | | 1 |
| Z2→Z4 | 10 | | 01 | - 0 | 1 0 0 | 0 | | 1 |
| Z3→Z4 | 11 | | 01 | - - | 0 1 0 | 0 | | 1 |
| Z4→Z3 | 01 | | 11 | 0 1 | 0 0 1 | 1 | | 1 |
| Z4→Z4 | 01 | | 01 | 0 0 | 0 0 1 | 0 | | 1 |
| Z4→Z1 | 01 | | 00 | 1 - | 0 0 1 | 0 | | 0 |

**3.5 Мінімізація функцій тригерів**

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

D2

D1

Y2

Y1

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

*Рисунок 3.3 – Мінімізація функцій тригерів*

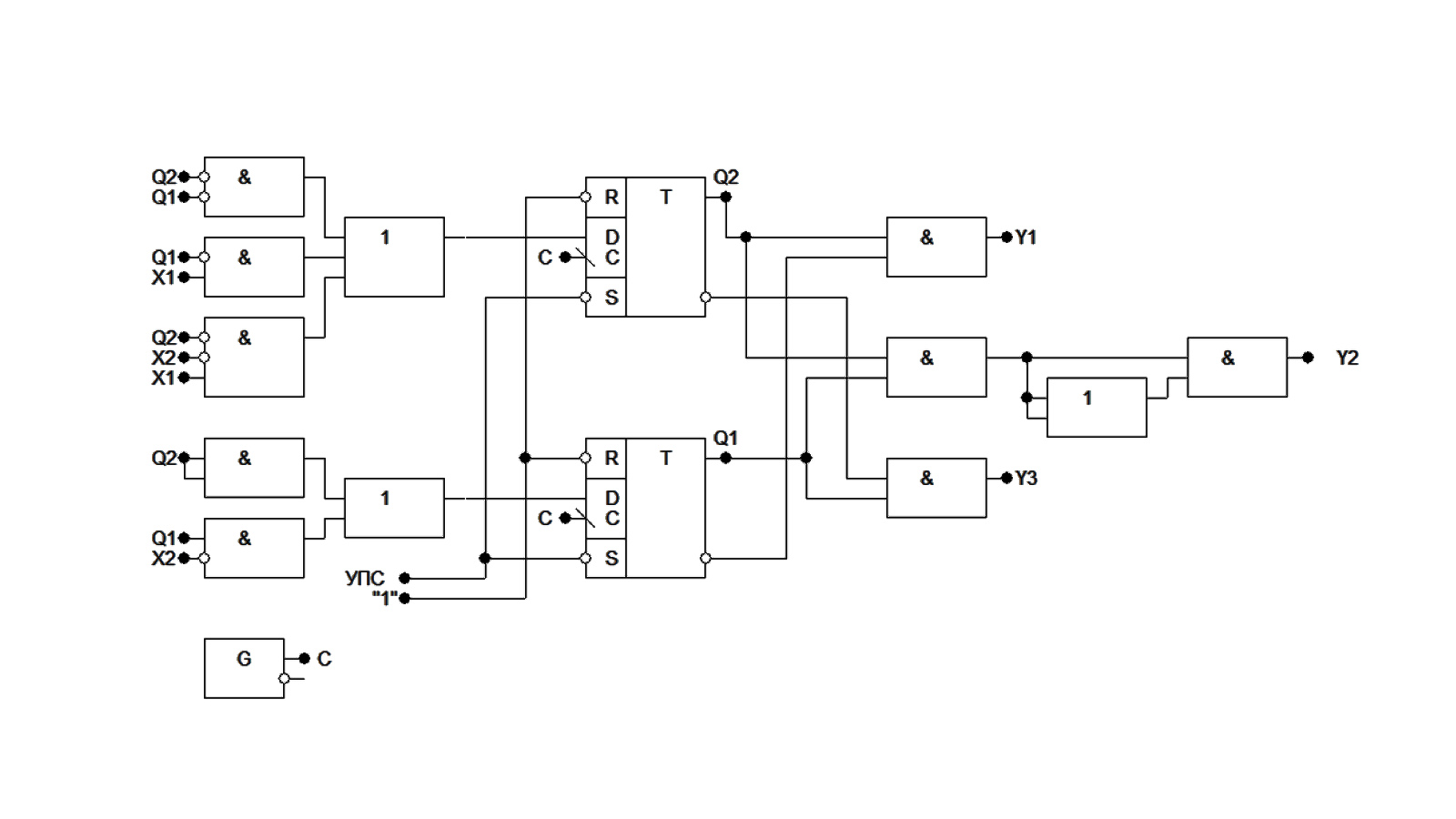
|  |  |
| --- | --- |
| *0* | *1* |
| *0* | *0* |

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

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

Y3

*Рисунок 3.4 – Діаграми Вейча для вихідних сигналів*

**3.6 Функціональна схема автомата**

*Рисунок 3.5 - Функціональна схема*

Висновок.

У даній розрахунковій роботі було виконано операції з числами в двійковому коді з плаваючою комою, а саме: множення чотирма способами, ділення двома способами, додавання та віднімання. Для операції множення першим способом було побудовано управляючий автомат Мура на D-тригерах і елементах булевого базису. Зроблено мінімізацію функцій тригерів і в середовищі AFDK побудована функціональна схема автомата. На одній з функцій використано фільтр для запобігання виникненню просічок.

Під час виконання даної розрахункової роботи я повторив для себе матеріал курсу «Компютерна логіка - 1», а також закріпив знання з курсу «Компютерна логіка - 2».

Було використано наступну літературу:

1) Жабін В.І., Жуков І.А., Клименко І.А.,Ткаченко В.В. Прикладна теорія цифрових автоматів: Навчальний посібник.*–*К.: Книжкове вид-во НАУ, 2007. *–* 360 с.

2) Конспект лекцій з курсу «Комп*’*ютерна логіка - 1»

3) Конспект лекцій з курсу «Комп*’*ютерна логіка - 2»

4) Жабін В.І., Жуков І.А., Клименко І.А. Стіренко С.Г. Арифметичні та управляючі пристрої цифрових ЕОМ: Навчальний посібник. - К.: ВЕК +, 2008. - 176 с.