НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

# "КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ"

#### ФАКУЛЬТЕТ ІНФОРМАТИКИ ТА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

### Кафедра обчислювальної техніки

## РОЗРАХУНКОВА РОБОТА

по курсу „Комп'ютерна логіка-2. Комп'ютерна арифметика”

Виконав: Шудренко Євген Олегович

Група ІО-43, Факультет ІОТ,

Залікова книжка № 4328

Допущений до захисту \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Номер технічного завдання 1000011101000

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

(підпис керівника)

.

Київ – 2015 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 (характеристику) 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 та віднімання). Номери операцій (для п.3) відповідають порядку переліку (починаючи з нулевого номера). Наприклад, 0 – перший спосіб множення, 5 - ділення другим способом і т.і.).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 двійковим номером  побудувати управляючий автомат Мура на тригерах (тип тригера вибрати самостійно) і елементах булевого базису.

**Визначення та обґрунтування 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).

432810=10000111010002;

 = -10011111,0100100;

 = +10101,1110100001;

**Основна частина:**

**Завдання №1**

=1. 10011111,0100100;

= 0. 10101,1110100001;

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 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |

Y2:

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

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

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

E = P + 2m ,

m = 7;

27 = 100000002

Ex = 10000000 + 111 = 10000111

X2:

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

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

Ey = 10000000 + 100 = 10000100

Y2:

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

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

**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.

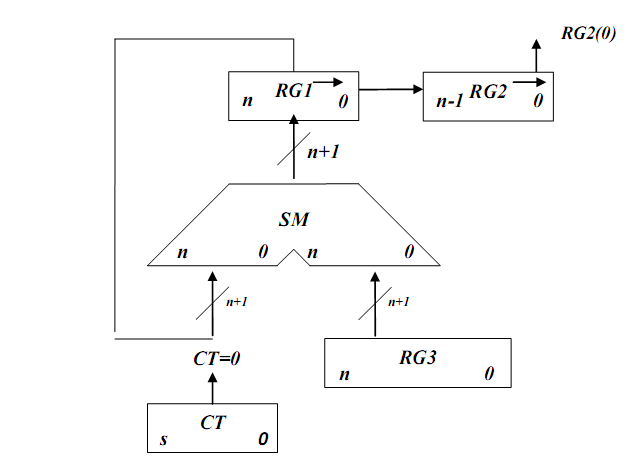
Множення мантис першим способом здійснюється з молодших розрядів множника, сума часткових добутків зсувається вправо, а множене залишається нерухомим. Тоді добуток двох чисел представляється у вигляді:

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

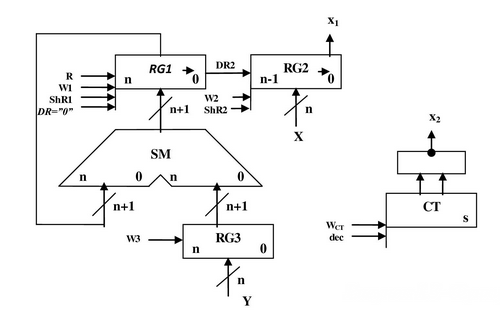
*Рисунок 2.1.2 - Змістовний мікроалгоритм виконання операції множення першим способом.*

**2.1.4 Таблиця станів регістрів:**

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

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** | **RG1** | **RG2** | **RG3** | **CT** |
| **пс** | 0000000000000000 | 100111110100100 | 101011110100001 | 1111 |
| **1** | 0000000000000000 | 001001111101001 |  | 1110 |
| **2** | 0000000000000000 | 100100111110100 |  | 1101 |
| **3** | 0010101111010000 | 010010011111010 |  | 1100 |
| **4** | 0001010111101000 | 001001001111101 |  | 1011 |
| **5** | 0000101011110100 | 011101001001100 |  | 1010 |
| **6** | +0101011110100001  =0110001010010101  0011000101001010 | 100100100111110 |  | 1001 |
| **7** | 0001100010100101 | 010010010011111 |  | 1000 |
| **8** | +0101011110100001  =0111000001000110  0011100000100011 | 001001001001111 |  | 0111 |
| **9** | +0101011110100001  =1000111111000100  0100011111100010 | 000100100100111 |  | 0110 |
| **10** | + 0101011110100001  =1001111110000011  0100111111000001 | 100010010010011 |  | 0101 |
| **11** | +0101011110100001  =1010011101100010  0101001110110001 | 010001001001001 |  | 0100 |
| **12** | +0101011110100001 =1010101101010010  0101010110101001 | 001000100100100 |  | 0011 |
| **13** | 0010101011010100 | 100100010010010 |  | 0010 |
| **14** | 0001010101101010 | 010010001001001 |  | 0001 |
| **15** | +0101011110100001  =0110110100001011  **0011011010000101** | **101001000100100** |  | 0000 |

**2.1.5 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: 011011010000101101001000100100

Знак мантиси: 1 0 = 1.

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

Порядок зменшуємо на 1:

11011010000101101001000100100;=12;

Запишемо нормалізований результат:

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

**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.

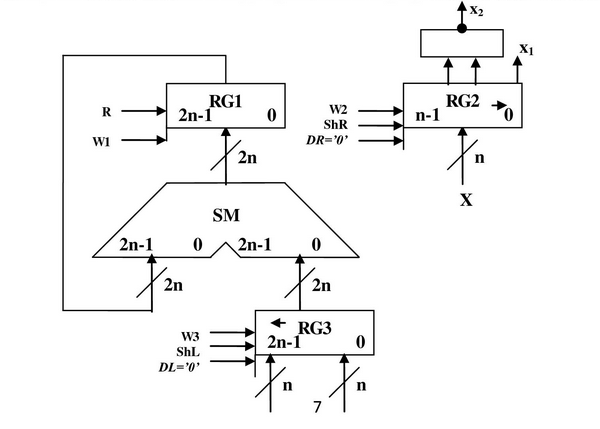
Множення мантис другим способом здійснюється з молодших розрядів, множене зсувається вліво, а сума часткових добутків залишається нерухомою.

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

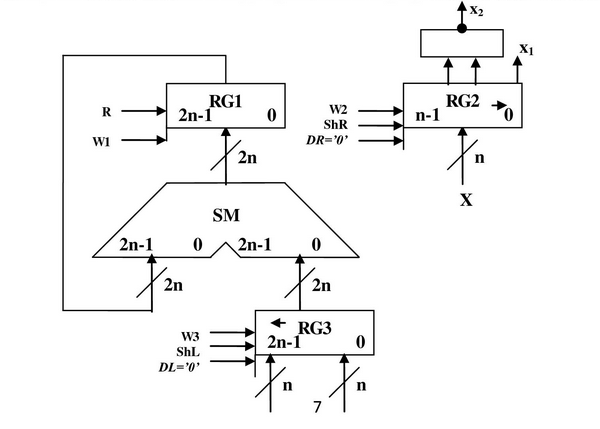
*Рисунок 2.2.2 - Змістовний мікроалгоритм.*

**2.2.4 Таблиця станів регістрів:**

*Таблиця 2.2.1-Таблиця станів регістрів.*

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **RG1** | **RG3 ←** | **RG2 →** |
| **пс** | 000000000000000000000000000000 | 000000000000000101011110100001 | 100111110100100 |
| **1** | 000000000000000000000000000000 | 000000000000001010111101000010 | 010011111010010 |
| **2** | 000000000000000000000000000000 | 000000000000010101111010000100 | 001001111101001 |
| **3** | 000000000000010101111010000100 | 000000000000101011110100001000 | 000100111110100 |
| **4** | 000000000000010101111010000100 | 000000000001010111101000010000 | 000010011111010 |
| **5** | 000000000000010101111010000100 | 000000000010101111010000100000 | 000001001111101 |
| **6** | +000000000010101111010000100000  =000000000011000101001010100100 | 000000000101011110100001000000 | 000000100111110 |
| **7** | 000000000011000101001010100100 | 000000001010111101000010000000 | 000000010011111 |
| **8** | +000000001010111101000010000000  =000000001110000010001100100100 | 000000010101111010000100000000 | 000000001001111 |
| **9** | +000000010101111010000100000000  =000000100011111100010000100100 | 000000101011110100001000000000 | 000000000100111 |
| **10** | +000000101011110100001000000000  =000001001111110000011000100100 | 000001010111101000010000000000 | 000000000010011 |
| **11** | +000001010111101000010000000000  =000010100111011000101000100100 | 000010101111010000100000000000 | 000000000001001 |
| **12** | +000010101111010000100000000000  =000101010110101001001000100100 | 000101011110100001000000000000 | 000000000000100 |
| **13** | 000101010110101001001000100100 | 001010111101000010000000000000 | 000000000000010 |
| **14** | 000101010110101001001000100100 | 010101111010000100000000000000 | 000000000000001 |
| **15** | +010101111010000100000000000000  =**011011010000101101001000100100** | 101011110100001000000000000000 | 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:**



*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: 011011010000101101001000100100

Знак мантиси: 1 0 = 1.

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

Порядок зменшуємо на 1:

11011010000101101001000100100;=12;

Запишемо нормалізований результат:

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

**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.

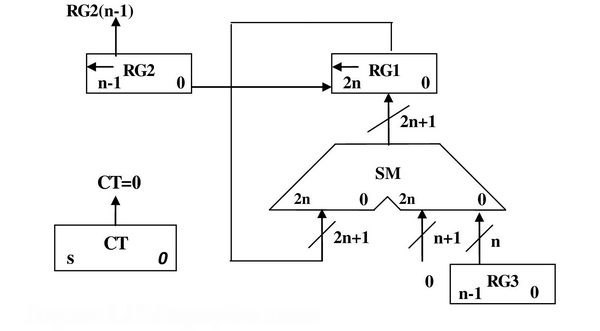
Множення мантис третім способом здійснюється зі старших розрядів множника, сума часткових добутків і множник зсуваються вліво, а множене нерухоме.

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

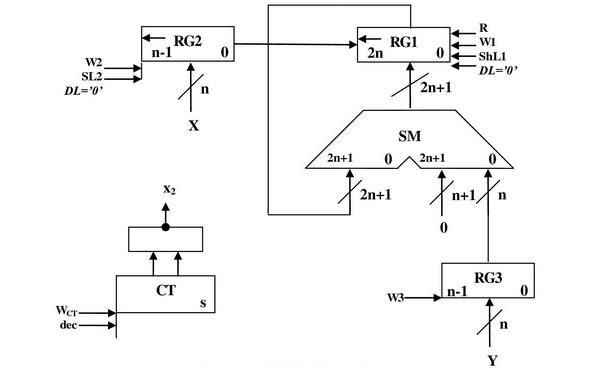
*Рисунок 2.3.2* **-** *Змістовний мікроалгоритм.*

**2.3.4 Таблиця станів регістрів:**

*Таблиця 2.3.1- Таблиця станів регістрів*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** | **RG1 ←** | **RG2 ←** | **RG3** | **CT** |
| **пс** | 000000000000000000000000000000 | 100111110100100 | 101011110100001 | 1111 |
| **1** | 000000000000001010111101000010 | 001111101001000 |  | 1110 |
| **2** | 000000000000010101111010000100 | 011111010010000 |  | 1101 |
| **3** | 000000000000101011110100001000 | 111110100100000 |  | 1100 |
| **4** | +000000000000000101011110100001  =000000000000110001010010101001  000000000001100010100101010010 | 111101001000000 |  | 1011 |
| **5** | +000000000000000101011110100001  =000000000001101000000011110011  000000000011010000000111100110 | 111010010000000 |  | 1010 |
| **6** | +000000000000000101011110100001  =000000000011010101100110000111  000000000110101011001100001110 | 110100100000000 |  | 1001 |
| **7** | +000000000000000101011110100001  =000000000110110000101010101111  000000001101100001010101011110 | 101001000000000 |  | 1000 |
| **8** | +000000000000000101011110100001  =000000001101100110110011111111  000000011011001101100111111110 | 010010000000000 |  | 0111 |
| **9** | 000000110110011011001111111100 | 100100000000000 |  | 0110 |
| **10** | +000000000000000101011110100001  =000000110110100000101110011101  000001101101000001011100111010 | 001000000000000 |  | 0101 |
| **11** | 000011011010000010111001110100 | 010000000000000 |  | 0100 |
| **12** | 000110110100000101110011101000 | 100000000000000 |  | 0011 |
| **13** | +000000000000000101011110100001  =000110110100001011010010001001  001101101000010110100100010010 | 000000000000000 |  | 0010 |
| **14** | 011011010000101101001000100100 | 000000000000000 |  | 0001 |
| **15** | **110110100001011010010001001000** | 000000000000000 |  | 0000 |

**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**

*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: 011011010000101101001000100100

Знак мантиси: 1 0 = 1.

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

Порядок зменшуємо на 1:

11011010000101101001000100100;=12;

Запишемо нормалізований результат:

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

**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.

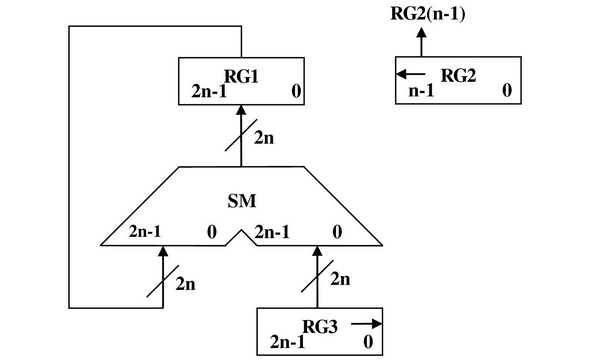
Множення здійснюється зі старших розрядів множника, сума часткових добутків залишається нерухомою, множене зсувається праворуч, множник ліворуч.

.

*.*

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

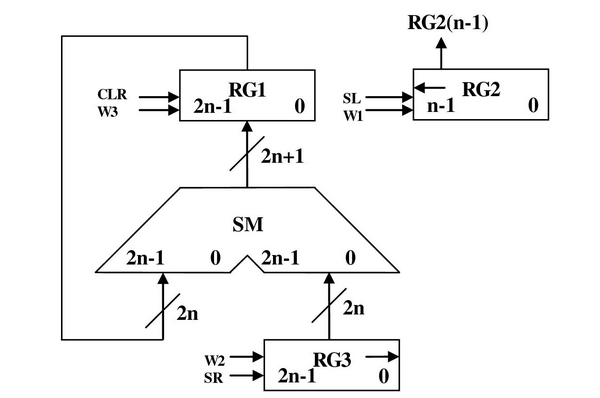
*Рисунок 2.4.2* **-** *Змістовний мікроалгоритм.*

**2.4.4 Таблиця станів регістрів:**

*Таблиця 2.4.1- Таблиця станів регістрів*

|  |  |  |  |
| --- | --- | --- | --- |
| № | RG1 | RG3 → | RG2 ← |
| ПС | 000000000000000000000000000000 | 010101111010000100000000000000 | 100111110100100 |
| **1** | 010101111010000100000000000000 | 001010111101000010000000000000 | 001111101001000 |
| **2** | 010101111010000100000000000000 | 000101011110100001000000000000 | 011111010010000 |
| **3** | 010101111010000100000000000000 | 000010101111010000100000000000 | 111110100100000 |
| **4** | +000010101111010000100000000000  =011000101001010100100000000000 | 000001010111101000010000000000 | 111101001000000 |
| **5** | +000001010111101000010000000000  =011010000000111100110000000000 | 000000101011110100001000000000 | 111010010000000 |
| **6** | +000000101011110100001000000000  =011010101100110000111000000000 | 000000010101111010000100000000 | 110100100000000 |
| **7** | +000000010101111010000100000000  =011011000010101010111100000000 | 000000001010111101000010000000 | 101001000000000 |
| **8** | +000000001010111101000010000000  =011011001101100111111110000000 | 000000000101011110100001000000 | 010010000000000 |
| **9** | 011011001101100111111110000000 | 000000000010101111010000100000 | 100100000000000 |
| **10** | +000000000010101111010000100000  =011011010000010111001110100000 | 000000000001010111101000010000 | 001000000000000 |
| **11** | 011011010000010111001110100000 | 000000000000101011110100001000 | 010000000000000 |
| **12** | 011011010000010111001110100000 | 000000000000010101111010000100 | 100000000000000 |
| **13** | +000000000000010101111010000100  =011011010000101101001000100100 | 000000000000001010111101000010 | 000000000000000 |
| **14** | 011011010000101101001000100100 | 000000000000000101011110100001 | 000000000000000 |
| **15** | **011011010000101101001000100100** | 000000000000000010101111010000 | 000000000000000 |

**2.4.5 Функціональна 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:**



*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: 011011010000101101001000100100

Знак мантиси: 1 0 = 1.

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

Порядок зменшуємо на 1:

11011010000101101001000100100;=12;

Запишемо нормалізований результат:

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

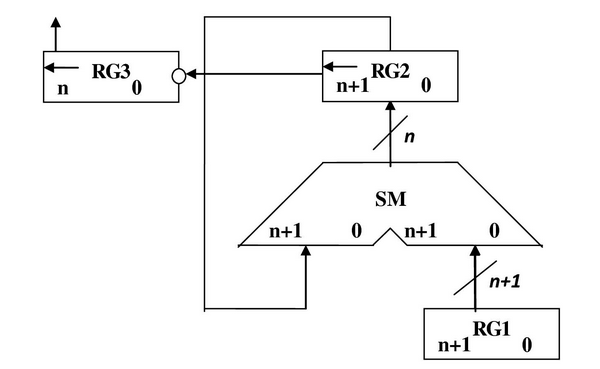
**2.5. Першиий спосіб ділення.**

**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

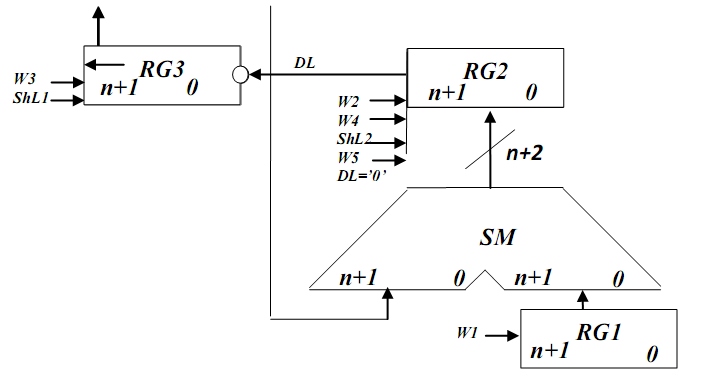
RG2[n+1]

*Рисунок 2.5.2-Змістовний мікроалгоритм*

**2.5.4 Таблиця станів регістрів:**

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **RG3(Z)** | **RG2(X)** | **RG1(Y)** |
| **пс** | 0000000000000000 | 00100111110100100 | 101011110100001 |
| **1** | 0000000000000001 | 01001111101001000  +11010100001011111  =00100011110100111 |  |
| **2** | 0000000000000011 | 01000111101001110  +11010100001011111  =00011011110101101 |  |
| **3** | 0000000000000111 | 00110111101011010  +11010100001011111  =00001011110111001 |  |
| **4** | 0000000000001111 | 00010111101110010  +11010100001011111  =11101011111010001 |  |
| **5** | 0000000000011110 | 11010111110100010  +00101011110100001  =00000011101000011 |  |
| **6** | 0000000000111101 | 00000111010000110  +11010100001011111  =11011011011100101 |  |
| **7** | 0000000001111010 | 10110110111001010  +00101011110100001  =11100010101101011 |  |
| **8** | 0000000011110100 | 11000101011010110  +00101011110100001  =11110001001110111 |  |
| **9** | 0000000111101000 | 11100010011101110  +00101011110100001  =00001110010001111 |  |
| **10** | 0000001111010001 | 00011100100011110  +11010100001011111  =11110000101111101 |  |
| **11** | 0000011110100010 | 11100001011111010  +00101011110100001  =00001101010011011 |  |
| **12** | 0000111101000101 | 00011010100110110  +11010100001011111  =11101110110010101 |  |
| **13** | 0001111010001010 | 11011101100101010  +00101011110100001  =00001001011001011 |  |
| **14** | 0011110100010101 | 00010010110010110  +11010100001011111  =11100110111110101 |  |
| **15** | 0111101000101010 | 11001101111101010  +00101011110100001  =11111001110001011 |  |
| **16** | **1111010001010100** | 11110011100010110  +00101011110100001  =00011111010110111 |  |

**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

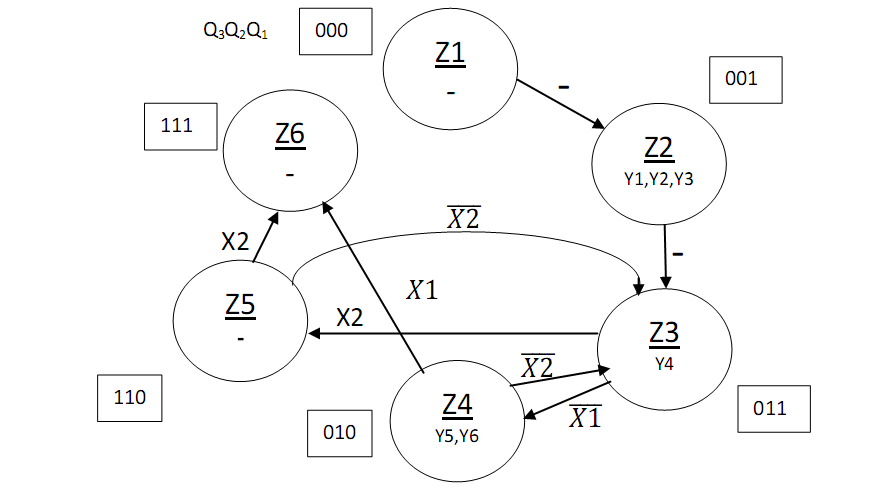
Z6

0

Кінець

*Рисунок 2.5.4-Закодований мікроалгоритм.*

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



*Рисунок 2.5.5 - Граф управляючого автомата.*

**2.5.8 Обробка порядків:**

Порядок частки буде дорівнювати:

=8; =5; =3;

**2.5.8 Нормалізація результату:**

Отримали результат: 1111010001010100

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 | 1 | 0 | 1 | 0 | 1 | 0 |

**2.6. Другий спосіб ділення.**

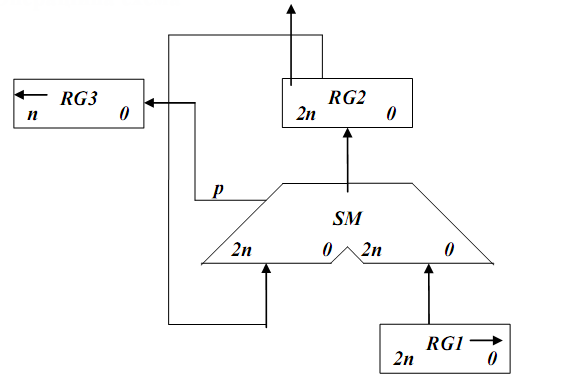
**2.6.1Теоритичне обгрунтування другого способу ділення:**

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

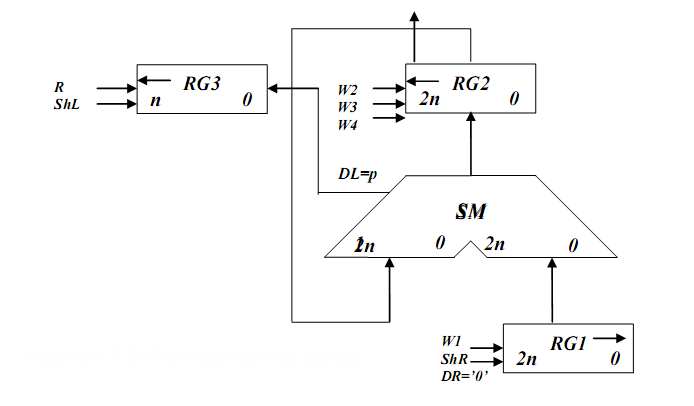
*Рисунок 2.6.2-Змістовний мікроалгоритм*

**2.6.4 Таблиця станів регістрів**

*Таблиця 2.6.1- Таблиця станів регістрів*

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **RG3(Z)** | **RG2(X)** | **RG1(Y)** |
| **пс** | 0000000000000001 | 010011111010010000000000000000 | 001010111101000010000000000000 |
| **1** | 0000000000000011 | 010011111010010000000000000000  +110101000010111110000000000000  =001000111101001110000000000000 | 000101011110100001000000000000 |
| **2** | 0000000000000111 | 001000111101001110000000000000  +111010100001011111000000000000  =000011011110101101000000000000 | 000010101111010000100000000000 |
| **3** | 0000000000001111 | 000011011110101101000000000000  +111101010000101111100000000000  =000000101111011100100000000000 | 000001010111101000010000000000 |
| **4** | 0000000000011110 | 000000101111011100100000000000  +111110101000010111110000000000  =111111010111110100010000000000 | 000000101011110100001000000000 |
| **5** | 0000000000111101 | 111111010111110100010000000000  +000000101011110100001000000000  =000000000011101000011000000000 | 000000010101111010000100000000 |
| **6** | 0000000001111010 | 000000000011101000011000000000  +111111101010000101111100000000  =111111101101101110010100000000 | 000000001010111101000010000000 |
| **7** | 0000000011110100 | 111111101101101110010100000000  +000000001010111101000010000000  =111111111000101011010110000000 | 000000000101011110100001000000 |
| **8** | 0000000111101000 | 111111111000101011010110000000  +000000000101011110100001000000  =111111111110001001110111000000 | 000000000010101111010000100000 |
| **9** | 0000001111010001 | 111111111110001001110111000000  +000000000010101111010000100000  =000000000000111001000111100000 | 000000000001010111101000010000 |
| **10** | 0000011110100010 | 000000000000111001000111100000  +111111111110101000010111110000  =111111111111100001011111010000 | 000000000000101011110100001000 |
| **11** | 0000111101000101 | 111111111111100001011111010000  +000000000000101011110100001000  =000000000000001101010011011000 | 000000000000010101111010000100 |
| **12** | 0001111010001010 | 000000000000001101010011011000  +111111111111101010000101111100  =111111111111110111011001010100 | 000000000000001010111101000010 |
| **13** | 0011110100010101 | 111111111111110111011001010100  +000000000000001010111101000010  =000000000000000010010110010110 | 000000000000000101011110100001 |
| **14** | 0111101000101010 | 000000000000000010010110010110  +111111111111111010100001011111  =111111111111111100110111110101 | 000000000000000010101111010000 |
| **15** | **1111010001010100** | 111111111111111100110111110101  +000000000000000010101111010000  =111111111111111111100111000101 | 000000000000000001010111101000 |

**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

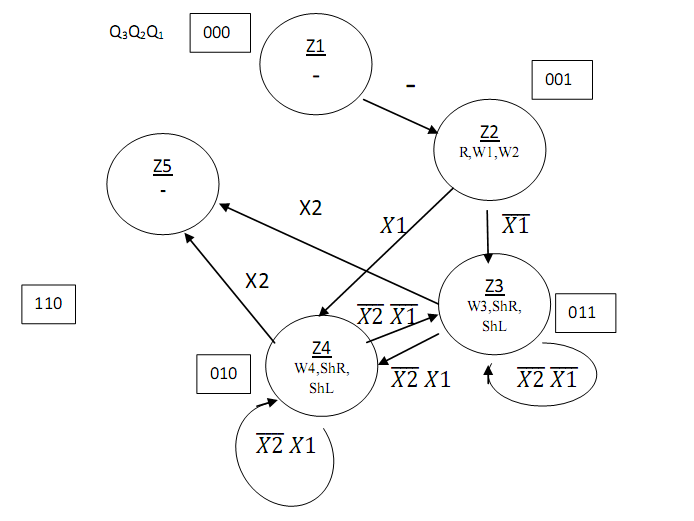
W4, ShR, ShL

W3, ShR, ShL

X2

*Рисунок 2.6.4- Закодований мікроалгоритм*

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



*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: 1111010001010100

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 | 1 | 0 | 1 | 0 | 1 | 0 |

**2.7. Операція додавання 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.

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 |
| 0,101011110100001 | 11 | Початковий стан |
| 0, 010101111010000 | 10 | My= 0.r(My); ∆:=∆-1 |
| 0, 001010111101000 | 01 | My= 0.r(My); ∆:=∆-1 |
| 0, 000101011110100 | 00 | My= 0.r(My); ∆:=∆-1 |

3. Додавання мантис у модифікованому ДК.

X мдк = 11. 011000001011100

Yмдк = 00. 101011110100001

*Таблиця 2.7.2-Додавання мантис(для додавання)*

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

Zдк = 1. 100011110101101

Zпк = 1. 011100001010011

4. Нормалізація результату (В ПК).

Pz = 710 =1112 Mz = - 0111000010100112

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

5. Нормалізація мантиси

Знак мантиси: 1 0 = 1.

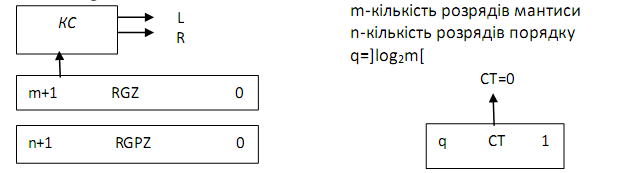
Робимо зсув результату вліво, доки у першому розряді не буде одиниця,

Порядок зменшуємо на 1:

11100001010011;=6; нормалізований результат:

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

**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*

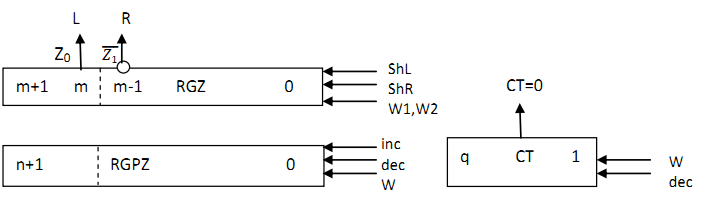
**2.7.4 Таблиця станів регістрів**

**1) Додавання**

*Таблиця 2.7.5- Таблиця станів регістрів*

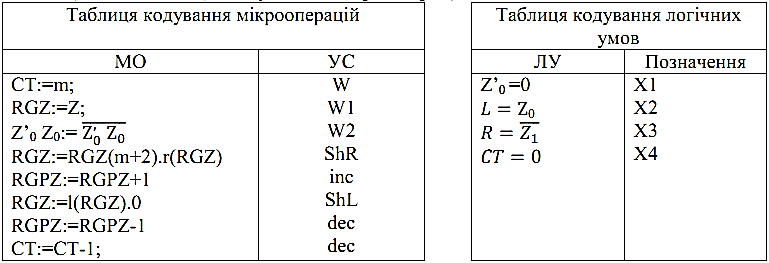
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№**  **такту** | **RGPZ** | **RGZ** | **ЛПН(L)** | **ППН(R)** | **СT** | **Мікрооперація** |
| **ПС** | 001000 | 11. 100000 01010 1011 | 0 | 1 | 100 |  |

**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*

****

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**

PX+Y= 810 =10002

**2.7.9 Форма запису результату з плаваючою комою**

Результат додавання Z=X+Y.

Zпк = 1. 011100001010011

Pz = 610 =1102 Mz = - 111000010100112

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

**2.8. Операція віднімання чисел.**

**2.8.1 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 доповню вальних 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.

Для отримання різниці чисел знак мантиси від’ємника змінюється на протилежний.

1. Порівняння порядків.

Px=+810=+10002

Py=+510=+01012

810-510=310=112

Pz=1011

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 |
| 0,101011110100001 | 11 | Початковий стан |
| 0,010101111010000 | 10 | My= 0.r(My); ∆=∆-1 |
| 0,001010111101000 | 01 | My= 0.r(My); ∆=∆-1 |
| 0,000101011110100 | 00 | My= 0.r(My); ∆=∆-1 |

Віднімання мантис у модифікованому ДК.

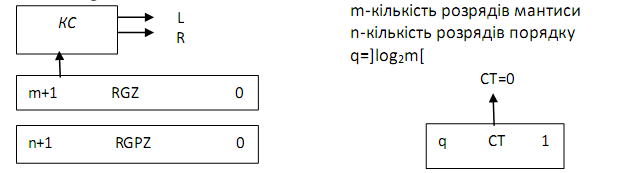
X мдк = 11,+1 = 11. 011000001011100

-Yмдк =11,  +1 = 11. 010100001011110

*Таблиця 2.7.3-Віднімання мантис у модифікованому ДК*

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

**2.8.2 Операційна схема**



*Рисунок 2.7.1-Операційна схема*

Виконаємо синтез КС для визначення порушення нормалізації.

z'0 – старший знаковий розряд мантиси,

z0 – молодший знаковий розряд мантиси,

z1 – старший розряд мантиси.

L та R є функціями порушення нормалізації мантиси вліво та вправо відповідно.

*Таблиця 2.7.3-Визначення порушення нормалізації*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Розряди регістру  RGZ | | | Значення  функцій | |
| z'0 | z0 | z1 | L | R |
| 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 | 0 |



Знак мантиси результату встановлюємо за Z'0 до нормалізації.

**2.8.3 Змістовний алгоритм**

Початок

CT:=m;

RGZ:=Z;

Кінець

RGZ:=l(RGZ).0

RGPZ:=RGPZ-1

CT:=CT-1

RGZ:=0.r(RGZ)

RGZP:=RGZP+1



CT=0

1



0

1



0

0

1

RGZ:=minM

RGPZ:=minP

*Рисунок 2.7.2-Змістовний мікроалгоритм*

**2.8.4 Таблиця станів регістрів**

Нормалізація результату віднімання.

*Таблиця 2.7.6- Таблиця станів регістрів*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№**  **такту** | **RGPZ** | **RGZ** | **ЛПН(L)** | **ППН(R)** | **СT** | **Мікрооперація** |
| **ПС** | 0001000 | 010011001100011 | 0 | 0 | 100 |  |

**2.8.5 Функціональна схема з відображенням керуючих сигналів**



*Рисунок 2.7.3 – Функціональна схема*

**2.8.6 Закодований мікроалгоритм**

*Таблиця 2.7.7 – Таблиця кодування*

|  |  |
| --- | --- |
| Таблиця кодування мікрооперацій | |
| МО | УС |
| CT:=m  RGZ:=Z  RGZ:=RGZ(m+2).r(RGZ)  RGPZ:=RGPZ+1  RGZ:=l(RGZ).0  RGPZ:=RGPZ-1  CT:=CT-1  RGZ:=minM  RGPZ:=minP | W  W1  ShR  Inc  ShL  Dec  Dec  W2  W3 |

|  |  |
| --- | --- |
| Таблиця кодування логічних умов | |
| ЛУ | Позначення |
| CT=0 | X1  X2  X3 |

Z2

Z1

Початок

W, W1

X1

Кінець

ShL, dec

ShR, Inc

X2

X3



1

Z3

0

0

1

Z4

0

1

Z5

W2, W3

Z6

*Рисунок 2.7.4 – Закодований мікроалгоритм*

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



*Рисунок 2.7.5 – Граф автомата Мура*

**2.8.8 Обробка порядкy**

PX-Y = 810 = 10002

**2.8.9 Форма запису результату з плаваючою комою**

Результат віднімання Z=X-Y.

Mz(мпк) = 11.010011101000110

Pz = 810 =10002 Mz = 010011101000110

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

**2.8.9.1 Нормалізація результату**

Mz = 010011101000110

Знак мантиси: 1 0 = 1.

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

Порядок зменшуємо на 1:

1001110100011;=7;

Запишемо нормалізований результат:

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

**Завдання 3**

x3x2x1 =0– перший спосіб множення.

**3.1 Таблиця співвідношення управляючих входів операційного автомата і виходів управляючого автомата**

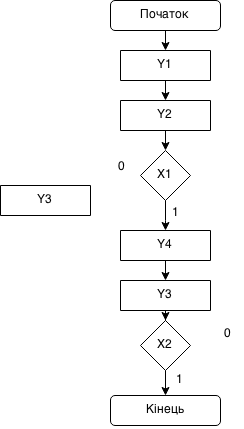
За закодованим мікроалгоритмом складемо таблицю:

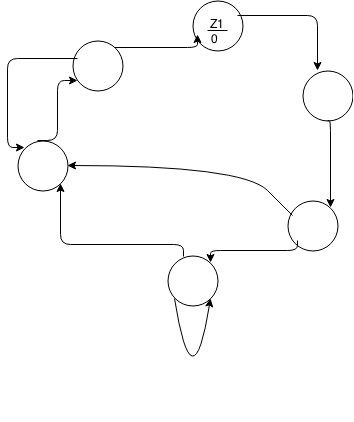
*Таблиця 3.1 Таблиця кодування сигналів*

|  |  |
| --- | --- |
| Входи операційного автомата | Виходи управляючого автомата |
| R | Y1 |
| W | Y2 |
| SH | Y3 |
| SUM | Y4 |

**3.2 Мікроалгоритм в термінах управляючого автомата**

Зробимо автомат Мура циклічним задля зменшення кількості вершин.



Будуємо граф автомата Мура

100

000

X2

Y3

001

Z6

Z2

X1

Y1

Y4

101

Z5

Z3

X1

Y2

011

Z4

Y3

111

*Рисунок 5.2- Граф автомата Мура*

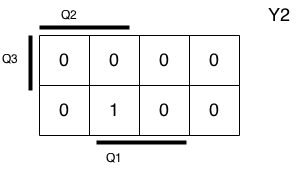
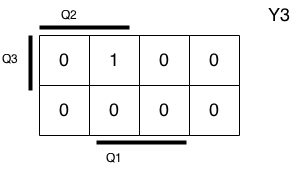
**3.3 Структурна таблиця автомата**

## 3.3.1Структурна таблиця автомата

Таблиця 3.2 - Структурна таблиця автомата

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Q3Q2Q1 | Q3Q2Q1 |  |  |  |  |  |
| 0 0 0 | 0 0 1 | \_ \_ | 0 0 0 0 | 0 | 0 | 1 |
| 0 0 1 | 0 1 1 | \_ \_ | 0 0 0 1 | 0 | 1 | 1 |
| 0 1 1 | 1 1 1 |  | 0 0 1 0 | 1 | 1 | 1 |
| 0 1 1 | 1 0 1 |  | 0 0 1 0 | 1 | 0 | 1 |
| 1 1 1 | 1 1 1 |  | 0 1 0 0 | 1 | 1 | 1 |
| 1 1 1 | 1 0 1 |  | 0 1 0 0 | 1 | 0 | 1 |
| 1 0 1 | 1 0 0 | \_ \_ | 1 0 0 0 | 1 | 0 | 0 |
| 1 0 0 | 1 0 1 |  | 1. 1 0 0 | 1 | 0 | 1 |
| 1 0 0 | 0 0 0 |  | 1. 1 0 0 | 0 | 0 | 0 |

## 3.3.2Синтез функцій виходів і переходів



0

0

0

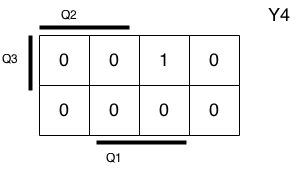


Рисунок 3.2 – Діаграми Вейча для функцій виходу

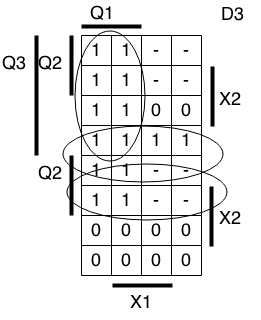
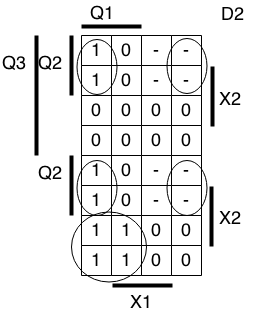
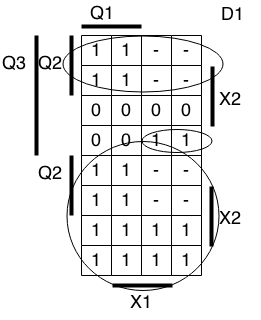
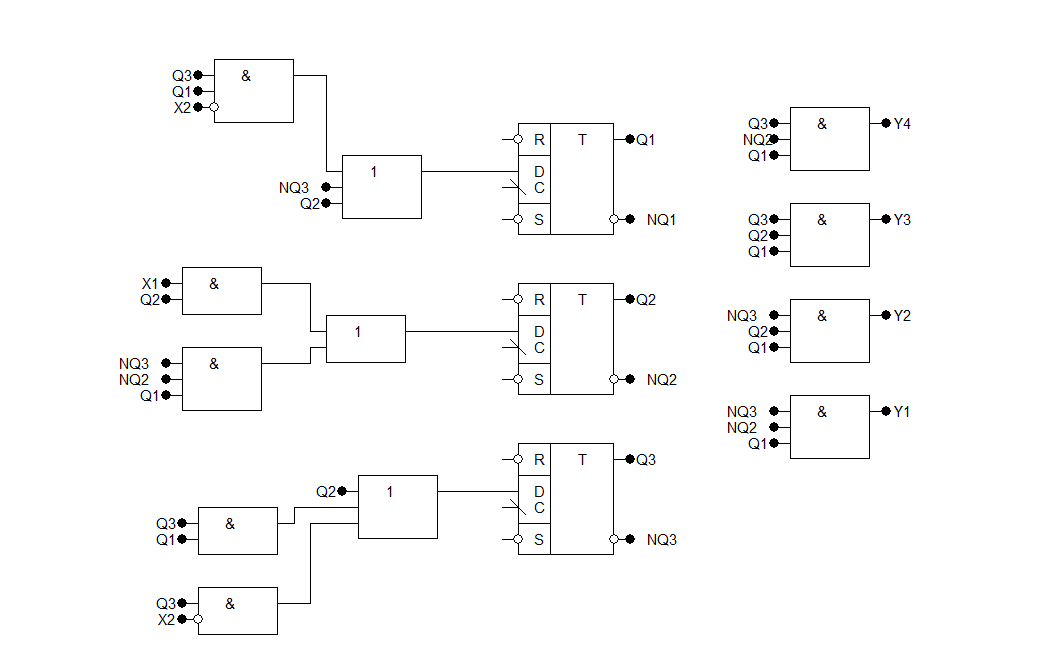


Рисунок 3.3 – Діаграми Вейча для функцій переходу

## 3.3 Функціональна схема пристою (виходи управляючого автомата підключені до входів операційного автомата)



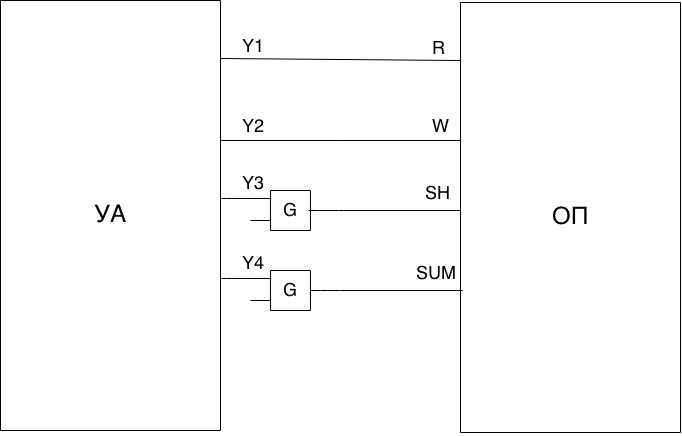


Рисунок 3.4 – Функціональна схема управляючого автомата

**Висновок**

У даній розрахунковій роботі було виконано операції з числами в двійковому коді з плаваючою комою, а саме: множення чотирма способами, ділення двома способами та додавання з відніманням.

Було побудувано управляючий автомат Мура на тригерах D і елементах булевого базиса для операції додавання.

Зроблено мінімізацію функцій тригерів і в середовищі AFDK побудована функціональна схема автомата.

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, 2009. *–* 360 с.

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

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