НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

# "КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ"

#### ФАКУЛЬТЕТ ІНФОРМАТИКИ ТА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

### Кафедра обчислювальної техніки

## РОЗРАХУНКОВА РОБОТА

по курсу „Комп'ютерна логіка-2”

Виконав: Бас Андрій Васильович

Група ІО-22, Факультет ІОТ,

Залікова книжка № 2201

Номер технічного завдання 100010011001

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

(підпис керівника)

Київ – 2013 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.

Операцію додавання до етапу нормалізації результату можна проілюструвати у довільній формі. Вказані пункти виконати для етапу нормалізації результату з урахуванням можливого нулевого результату.

3. Для операції з двійковим номером +1 побудувати управляючий автомат Мура на тригерах ( =00 відповідає RS-тригеру; 01 – D-тригеру; 10 – JK-тригеру; 11 – T-тригеру) 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).

220110=1000100110012;

 = - 10011001,1100101;

 = + 10101,0011100011;

**Ocновна частина:**

**Завдання №1**

=1. 10011001,1100101;

=0. 10101,0011100011;

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

Y2:

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

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

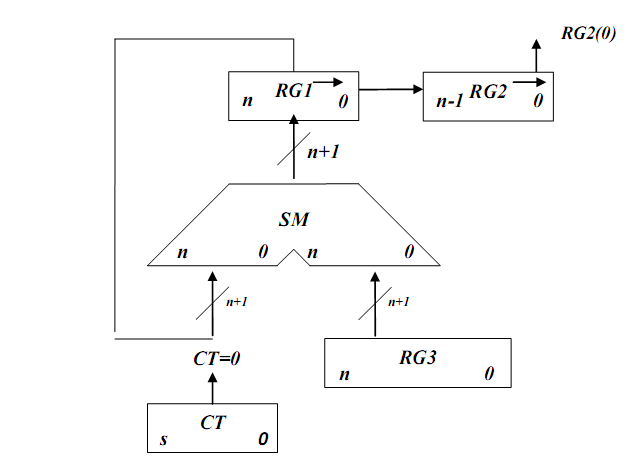
Множення мантис першим способом здійснюється з молодших розрядів множника, сума часткових добутків зсувається вправо, а множене залишається нерухомим. Тоді добуток двох чисел представляється у вигляді:

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

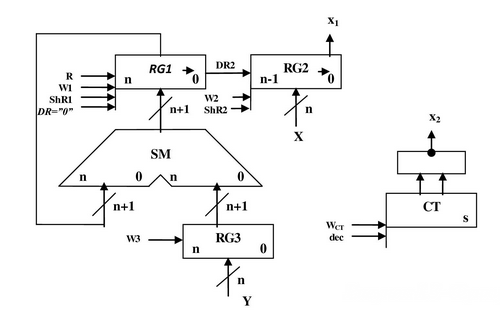
*Рисунок 2.1.2 - Змістовний мікроалгоритм виконання операції множення першим способом.*

**2.1.4 Таблиця станів регістрів:**

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

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** | **RG1** | **RG2** | **RG3** | **CT** |
| **пс** | 0 | 100110011100101 | 0101010011100011 | 1111 |
| **1** | +0101010011100011  =0101010011100011  0010101001110001 | 110011001110010 |  | 1110 |
| **2** | 0001010100111000 | 111001100111001 |  | 1101 |
| **3** | +0101010011100011  =0110101000011011  0011010100001101 | 111100110011100 |  | 1100 |
| **4** | 0001101010000110 | 111110011001110 |  | 1011 |
| **5** | 0000110101000011 | 011111001100111 |  | 1010 |
| **6** | +0101010011100011  =0110001000100110  0011000100010011 | 001111100110011 |  | 1001 |
| **7** | +0101010011100011  =1000010111110110  0100001011111011 | 000111110011001 |  | 1000 |
| **8** | +0101010011100011  =1001011111011110  0100101111101111 | 000011111001100 |  | 0111 |
| **9** | 0010010111110111 | 100001111100110 |  | 0110 |
| **10** | 0001001011111011 | 110000111110011 |  | 0101 |
| **11** | +0101010011100011  =0110011111011110  0011001111101111 | 011000011111001 |  | 0100 |
| **12** | +0101010011100011  =1000100011010010  0100010001101001 | 001100001111100 |  | 0011 |
| **13** | 0010001000110100 | 100110000111110 |  | 0010 |
| **14** | 0001000100011010 | 010011000011111 |  | 0001 |
| **15** | +0101010011100011  =0110010111111101  **0011001011111110** | **101001100001111** |  | 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: **011001011111110101001100001111**

Знак мантиси: 1 0 = 1.

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

Порядок зменшуємо на 1:

**11001011111110101001100001111**;=12;

Запишемо нормалізований результат:

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

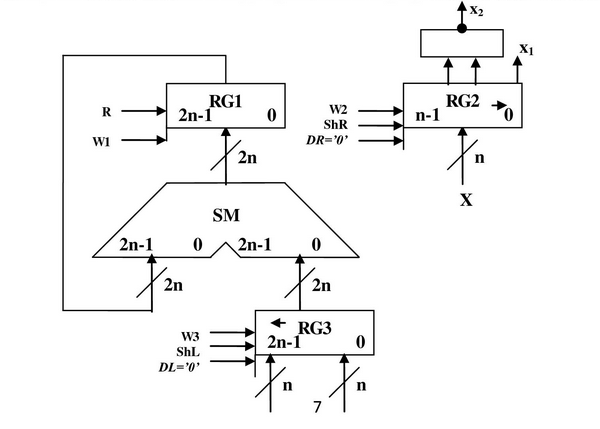
Множення мантис другим способом здійснюється з молодших розрядів, множене зсувається вліво, а сума часткових добутків залишається нерухомою.

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

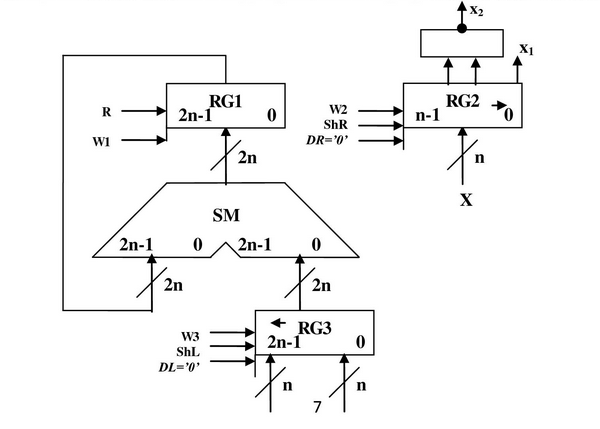
*Рисунок 2.2.2 - Змістовний мікроалгоритм.*

**2.2.4 Таблиця станів регістрів:**

*Таблиця 2.2.1-Таблиця станів регістрів.*

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **RG1** | **RG3 ←** | **RG2 →** |
| **пс** | 0 | 000000000000000101010011100011 | 100110011100101 |
| **1** | +000000000000000101010011100011  =000000000000000101010011100011 | 000000000000001010100111000110 | 010011001110010 |
| **2** | 000000000000000101010011100011 | 000000000000010101001110001100 | 001001100111001 |
| **3** | +000000000000010101001110001100  =000000000000011010100001101111 | 000000000000101010011100011000 | 000100110011100 |
| **4** | 000000000000011010100001101111 | 000000000001010100111000110000 | 000010011001110 |
| **5** | 000000000000011010100001101111 | 000000000010101001110001100000 | 000001001100111 |
| **6** | +000000000010101001110001100000  =000000000011000100010011001111 | 000000000101010011100011000000 | 000000100110011 |
| **7** | +000000000101010011100011000000  =000000001000010111110110001111 | 000000001010100111000110000000 | 000000010011001 |
| **8** | +000000001010100111000110000000  =000000010010111110111100001111 | 000000010101001110001100000000 | 000000001001100 |
| **9** | 000000010010111110111100001111 | 000000101010011100011000000000 | 000000000100110 |
| **10** | 000000010010111110111100001111 | 000001010100111000110000000000 | 000000000010011 |
| **11** | +000001010100111000110000000000  =000001100111110111101100001111 | 000010101001110001100000000000 | 000000000001001 |
| **12** | +000010101001110001100000000000  =000100010001101001001100001111 | 000101010011100011000000000000 | 000000000000100 |
| **13** | 000100010001101001001100001111 | 001010100111000110000000000000 | 000000000000010 |
| **14** | 000100010001101001001100001111 | 010101001110001100000000000000 | 000000000000001 |
| **15** | +010101001110001100000000000000  =**011001011111110101001100001111** | 101010011100011000000000000000 | 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: **011001011111110101001100001111**

Знак мантиси: 1 0 = 1.

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

Порядок зменшуємо на 1:

11001011111110101001100001111;=12;

Запишемо нормалізований результат:

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

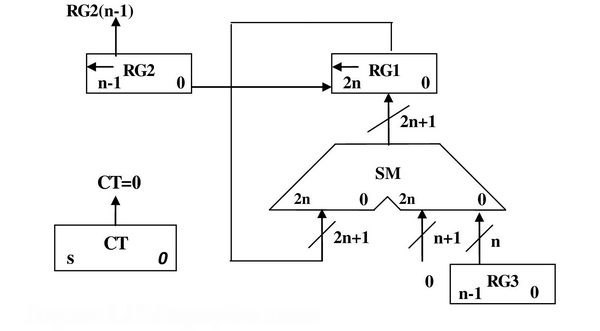
Множення мантис третім способом здійснюється зі старших розрядів множника, сума часткових добутків і множник зсуваються вліво, а множене нерухоме.

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

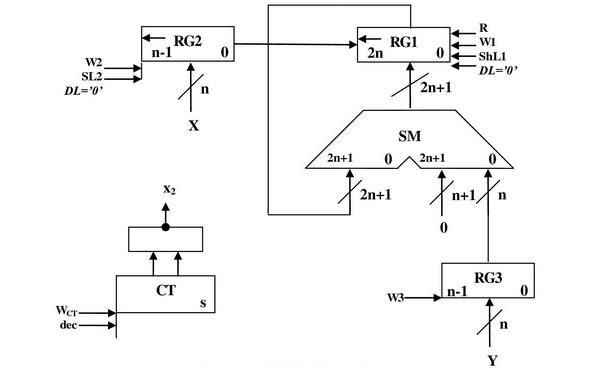
*Рисунок 2.3.2* **-** *Змістовний мікроалгоритм.*

**2.3.4 Таблиця станів регістрів:**

*Таблиця 2.3.1- Таблиця станів регістрів*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** | **RG1 ←** | **RG2 ←** | **RG3** | **CT** |
| **пс** | 000000000000000000000000000000 | 100110011100101 | 101010011100011 | 1111 |
| **1** | +000000000000000101010011100011  =000000000000000101010011100011  000000000000001010100111000110 | 001100111001010 |  | 1110 |
| **2** | 000000000000010101001110001100 | 011001110010100 |  | 1101 |
| **3** | 000000000000101010011100011000 | 110011100101000 |  | 1100 |
| **4** | +000000000000000101010011100011  =000000000000101111101111111011  000000000001011111011111110110 | 100111001010000 |  | 1011 |
| **5** | +000000000000000101010011100011  =000000000001100100110011011001  000000000011001001100110110010 | 001110010100000 |  | 1010 |
| **6** | 000000000110010011001101100100 | 011100101000000 |  | 1001 |
| **7** | 000000001100100110011011001000 | 111001010000000 |  | 1000 |
| **8** | +000000000000000101010011100011  =000000001100101011101110101011  000000011001010111011101010110 | 110010100000000 |  | 0111 |
| **9** | +000000000000000101010011100011  =000000011001011100110000111001  000000110010111001100001110010 | 100101000000000 |  | 0110 |
| **10** | +000000000000000101010011100011  =000000110010111110110101010101  000001100101111101101010101010 | 001010000000000 |  | 0101 |
| **11** | 000011001011111011010101010100 | 010100000000000 |  | 0100 |
| **12** | 000110010111110110101010101000 | 101000000000000 |  | 0011 |
| **13** | +000000000000000101010011100011  =000110010111111011111110001011  001100101111110111111100010110 | 010000000000000 |  | 0010 |
| **14** | 011001011111101111111000101100 | 100000000000000 |  | 0001 |
| **15** | +000000000000000101010011100011  =011001011111110101001100001111 | 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:**



*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: **011001011111110101001100001111**

Знак мантиси: 1 0 = 1.

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

порядок зменшуємо на 1:

11001011111110101001100001111;=12;

Запишемо нормалізований результат:

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

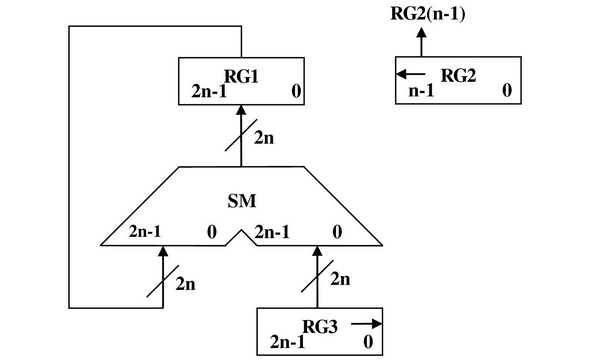
Множення здійснюється зі старших розрядів множника, сума часткових добутків залишається нерухомою, множене зсувається праворуч, множник ліворуч.

.

*.*

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

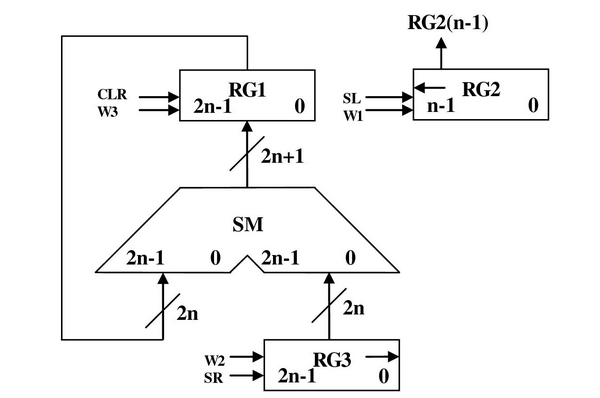
*Рисунок 2.4.2* **-** *Змістовний мікроалгоритм.*

**2.4.4 Таблиця станів регістрів:**

*Таблиця 2.4.1- Таблиця станів регістрів*

|  |  |  |  |
| --- | --- | --- | --- |
| № | RG1 | RG3 → | RG2 ← |
| ПС | 000000000000000000000000000000 | 010101001110001100000000000000 | 100110011100101 |
| **1** | +010101001110001100000000000000  =010101001110001100000000000000 | 001010100111000110000000000000 | 001100111001010 |
| **2** | 010101001110001100000000000000 | 000101010011100011000000000000 | 011001110010100 |
| **3** | 010101001110001100000000000000 | 000010101001110001100000000000 | 110011100101000 |
| **4** | +000010101001110001100000000000  =010111110111111101100000000000 | 000001010100111000110000000000 | 100111001010000 |
| **5** | +000001010100111000110000000000  =011001001100110110010000000000 | 000000101010011100011000000000 | 001110010100000 |
| **6** | 011001001100110110010000000000 | 000000010101001110001100000000 | 011100101000000 |
| **7** | 011001001100110110010000000000 | 000000001010100111000110000000 | 111001010000000 |
| **8** | +000000001010100111000110000000  =011001010111011101010110000000 | 000000000101010011100011000000 | 110010100000000 |
| **9** | +000000000101010011100011000000  =011001011100110000111001000000 | 000000000010101001110001100000 | 100101000000000 |
| **10** | +000000000010101001110001100000  =011001011111011010101010100000 | 000000000001010100111000110000 | 001010000000000 |
| **11** | 011001011111011010101010100000 | 000000000000101010011100011000 | 010100000000000 |
| **12** | 011001011111011010101010100000 | 000000000000010101001110001100 | 101000000000000 |
| **13** | +000000000000010101001110001100  =011001011111101111111000101100 | 000000000000001010100111000110 | 010000000000000 |
| **14** | 011001011111101111111000101100 | 000000000000000101010011100011 | 100000000000000 |
| **15** | +000000000000000101010011100011  =011001011111110101001100001111 | 000000000000000010101001110001 | 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:**



*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: 011001011111110101001100001111

Знак мантиси: 1 0 = 1.

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

Порядок понижаємо на 1:

11001011111110101001100001111;=12;

Запишемо нормалізований результат:

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

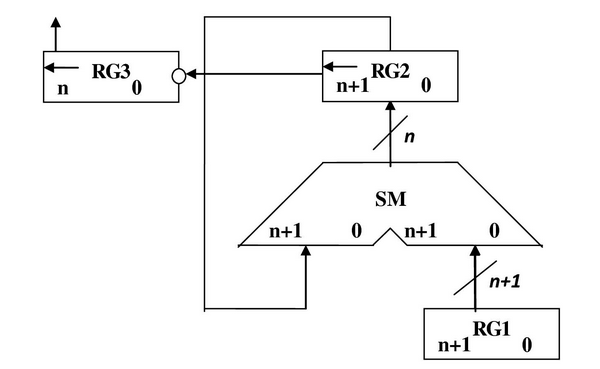
**2.5. Першиий спосіб ділення.**

**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

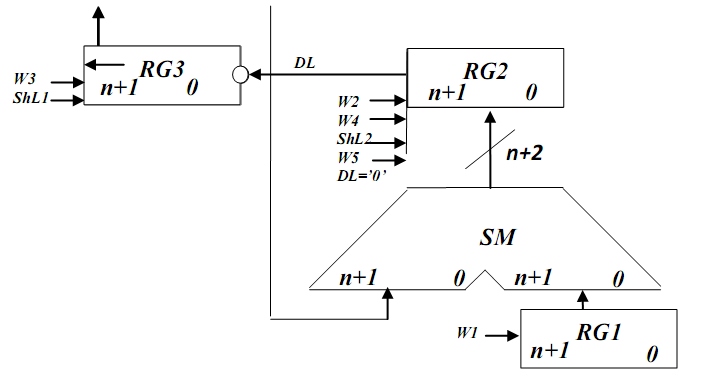
RG2[n+1]

*Рисунок 2.5.2-Змістовний мікроалгоритм*

**2.5.4 Таблиця станів регістрів:**

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **RG3(Z)** | **RG2(X)** | **RG1(Y)** |
| **пс** | 000000000000000 | 00100110011100101 | 101010011100011 |
| **1** | 0000000000000001 | 01001100111001010  +11010101100011101  =00100010011100111 |  |
| **2** | 0000000000000011 | 01000100111001110  +11010101100011101  =00011010011101011 |  |
| **3** | 0000000000000111 | 00110100111010110  +11010101100011101  =00001010011110011 |  |
| **4** | 0000000000001111 | 00010100111100110  +11010101100011101  =11101010100000011 |  |
| **5** | 0000000000011110 | 11010101000000110  +00101010011100011  =11111111011101001 |  |
| **6** | 0000000000111100 | 11111110111010010  +00101010011100011  =00101001010110101 |  |
| **7** | 0000000001111001 | 01010010101101010  +11010101100011101  =00101000010000111 |  |
| **8** | 0000000011110011 | 01010000100001110  +11010101100011101  =00100110000101011 |  |
| **9** | 0000000111100111 | 01001100001010110  +11010101100011101  =00100001101110011 |  |
| **10** | 0000001111001111 | 01000011011100110  +11010101100011101  =00011001000000011 |  |
| **11** | 0000011110011111 | 00110010000000110  +11010101100011101  =00000111100100011 |  |
| **12** | 0000111100111111 | 00001111001000110  +11010101100011101  =11100100101100011 |  |
| **13** | 0001111001111110 | 11001001011000110  +00101010011100011  =11110011110101001 |  |
| **14** | 0011110011111100 | 11100111101010010  +00101010011100011  =00010010000110101 |  |
| **15** | 0111100111111001 | 00100100001101010  +11010101100011101  =11111001110000111 |  |
| **16** | **1111001111110010** | 11110011100001110  +00101010011100011  =00011101111110001 |  |

**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

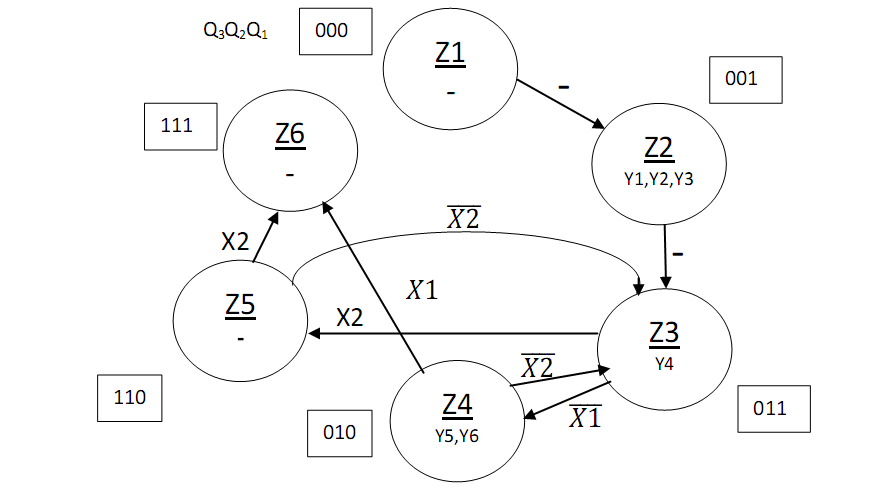
Z6

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:

В моєму випадку =8; =5; =3;

**2.5.8 Нормалізація результату:**

Отримали результат: **1111001111110010**

Знак мантиси: 1 0 = 1.

Нормалізація мантиси не потрібна.

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

**2.6. Другий спосіб ділення.**

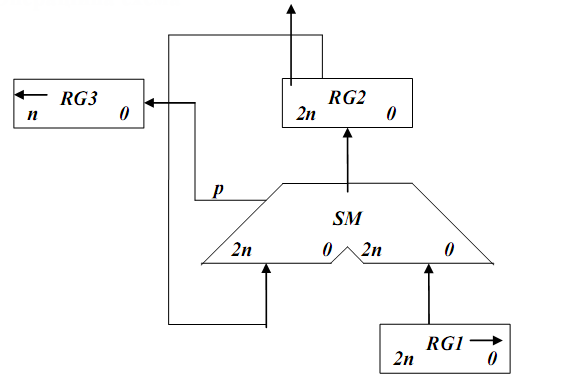
**2.6.1Теоритичне обгрунтування другого способу ділення:**

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

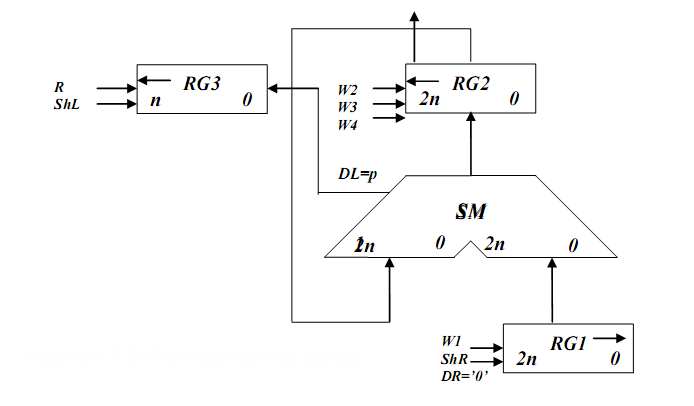
*Рисунок 2.6.2-Змістовний мікроалгоритм*

**2.6.4 Таблиця станів регістрів**

*Таблиця 2.6.1- Таблиця станів регістрів*

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **RG3(Z)** | **RG2(X)** | **RG1(Y)** |
| **пс** | 0000000000000001 | 010011001110010100000000000000 | 001010100111000110000000000000 |
| **1** | 0000000000000011 | 010011001110010100000000000000  +110101011000111010000000000000  =001000100111001110000000000000 | 000101010011100011000000000000 |
| **2** | 0000000000000111 | 001000100111001110000000000000  +111010101100011101000000000000  =000011010011101011000000000000 | 000010101001110001100000000000 |
| **3** | 0000000000001111 | 000011010011101011000000000000  +111101010110001110100000000000  =000000101001111001100000000000 | 000001010100111000110000000000 |
| **4** | 0000000000011110 | 000000101001111001100000000000  +111110101011000111010000000000  =111111010101000000110000000000 | 000000101010011100011000000000 |
| **5** | 0000000000111100 | 111111010101000000110000000000  +000000101010011100011000000000  =111111111111011101001000000000 | 000000010101001110001100000000 |
| **6** | 0000000001111001 | 111111111111011101001000000000  +000000010101001110001100000000  =000000010100101011010100000000 | 000000001010100111000110000000 |
| **7** | 0000000011110011 | 000000010100101011010100000000  +111111110101011000111010000000  =000000001010000100001110000000 | 000000000101010011100011000000 |
| **8** | 0000000111100111 | 000000001010000100001110000000  +111111111010101100011101000000  =000000000100110000101011000000 | 000000000010101001110001100000 |
| **9** | 0000001111001111 | 000000000100110000101011000000  +111111111101010110001110100000  =000000000010000110111001100000 | 000000000001010100111000110000 |
| **10** | 0000011110011111 | 000000000010000110111001100000  +111111111110101011000111010000  =000000000000110010000000110000 | 000000000000101010011100011000 |
| **11** | 0000111100111111 | 000000000000110010000000110000  +111111111111010101100011101000  =000000000000000111100100011000 | 000000000000010101001110001100 |
| **12** | 0001111001111110 | 000000000000000111100100011000  +111111111111101010110001110100  =111111111111110010010110001100 | 000000000000001010100111000110 |
| **13** | 0011110011111100 | 111111111111110010010110001100  +000000000000001010100111000110  =111111111111111100111101010010 | 000000000000000101010011100011 |
| **14** | 0111100111111001 | 111111111111111100111101010010  +000000000000000101010011100011  =000000000000000010010000110101 | 000000000000000010101001110001 |
| **15** | **1111001111110010** | 000000000000000010010000110101  +111111111111111101010110001111  =111111111111111111100111000100 | 000000000000000001010100111000 |

**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

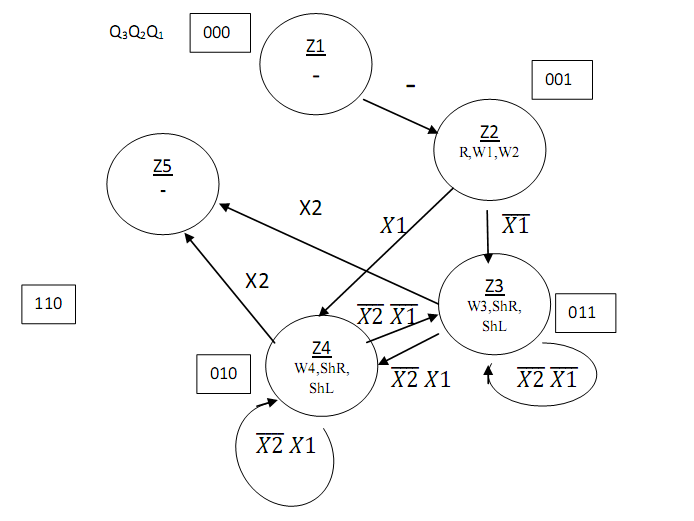
W4, ShR, ShL

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: **1111001111110010**

Знак мантиси: 1 0 = 1.

Нормалізація мантиси не потрібна.

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

**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, 101010011100011 | 11 | Початковий стан |
| 0, 010101001110001 | 10 | My= 0.r(My); ∆:=∆-1 |
| 0, 001010100111000 | 01 | My= 0.r(My); ∆:=∆-1 |
| 0,000101010011100 | 00 | My= 0.r(My); ∆:=∆-1 |

3. Додавання мантис у модифікованому ДК.

X мдк = 11. 011001100011010

Yмдк = 00. 000101010111000

*Таблиця 2.7.2-Додавання мантис(для додавання)*

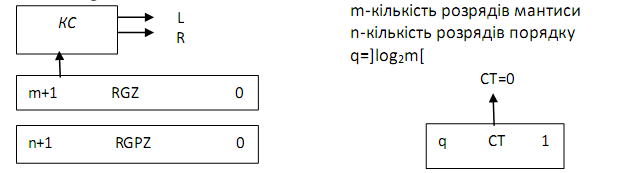
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| MX | 1 | 1, | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 |
| MY | 0 | 0, | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 |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

Zпк = 1. 100001001001001

4. Нормалізація результату (В ПК).

Для даного результату додавання 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*

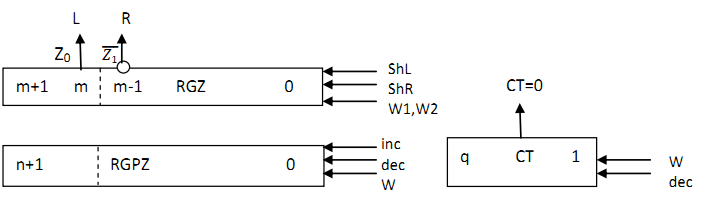
**2.7.4 Таблиця станів регістрів**

**1) Додавання**

*Таблиця 2.7.5- Таблиця станів регістрів*

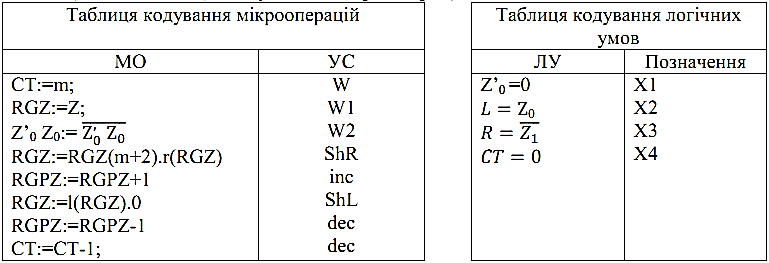
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№**  **такту** | **RGPZ** | **RGZ** | **ЛПН(L)** | **ППН(R)** | **СT** | **Мікрооперація** |
| **ПС** | 001000 | 00.111110011001001 | 0 | 1 | 100 |  |
| **1** | 000111 | 00.111100110010010  00.111001100100100 | 0 | 0 | 011 |  |

**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.

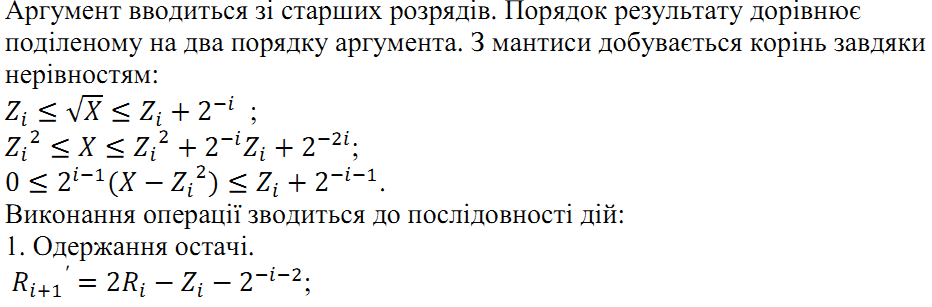
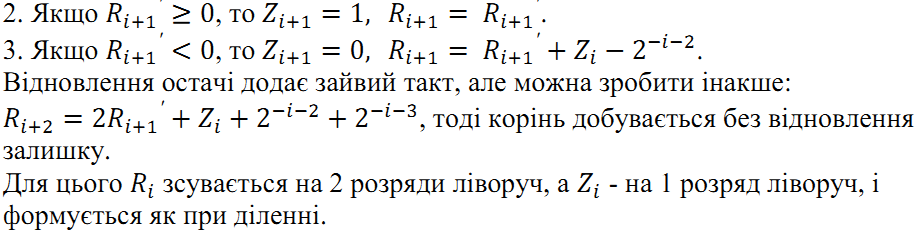
Zпк = 1. 100001001001001

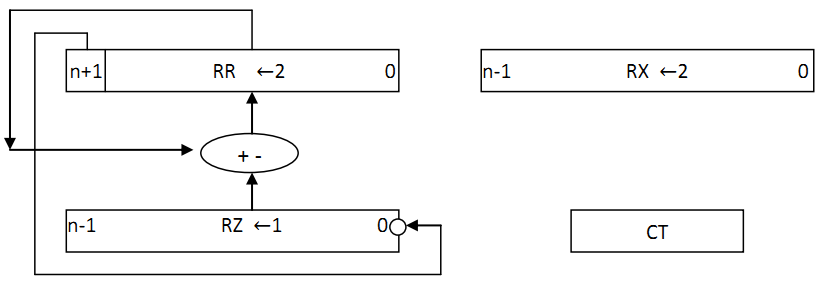
Pz = 810 =10002 Mz = 1000010010010012

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

**2.8.Операціядобування кореня**

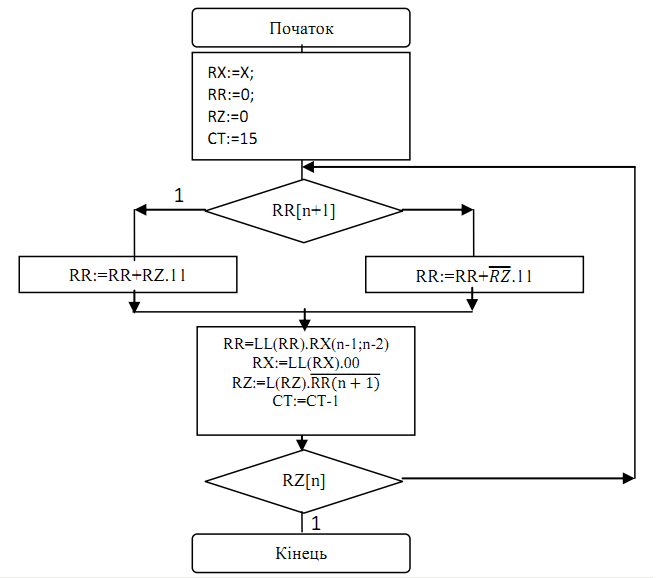
**2.8.1Теоритичнеобгрунтуванняоперації обчислення квадратного кореня**



**2.8.2 Операційна схема операції обчислення квадратного кореня**

*Рисунок 2.8.1 –Операційна схема*

**2.8.3 Змістовний мікроалгоритм**



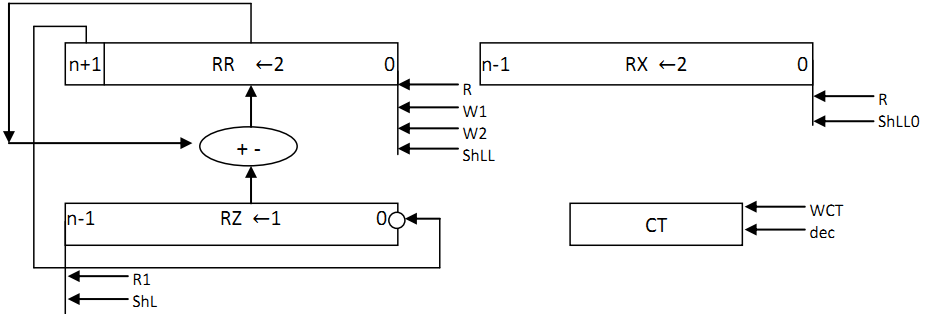
*Рисунок 2.8.2 – Змістовний мікроалгоритм*

**2.8.4 Таблиця станів регістрів**

*Таблиця 2.8.1 – Таблиця станів регістрів*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | RZ | RR | RX | СТ |
| **пс** | 000000000000000 | 00000000000000000  00000000000000010 | 100111011101101 | 1111 |
| **пз** |
| **1** | 000000000000001 | +11111111111111111  =00000000000000001  00000000000000101 | 010011001110010 | 1110 |
| **2** | 000000000000011 | +11111111111111011  =00000000000000000  00000000000000010 | 001001100111001 | 1101 |
| **3** | 000000000000110 | +11111111111110011  =11111111111110101  11111111111010101 | 000100110011100 | 1100 |
| **4** | 000000000001100 | +00000000000011011  =11111111111110000  11111111111000011 | 000010011001110 | 1011 |
| **5** | 000000000011000 | +00000000000110011  =11111111111110110  11111111111011000 | 000001001100111 | 1010 |
| **6** | 000000000110001 | +00000000001100011  =00000000000111011  00000000011101110 | 000000100110011 | 1001 |
| **7** | 000000001100011 | +11111111100111011  =00000000000101001  00000000010100110 | 000000010011001 | 1000 |
| **8** | 000000011000110 | +11111111001110011  =11111111100011001  11111110001100100 | 000000001001100 | 0111 |
| **9** | 000000110001100 | +00000001100011011  =11111111101111111  11111110111111100 | 000000000100110 | 0110 |
| **10** | 000001100011001 | +00000011000110011  =00000010000101111  00001000010111100 | 000000000010011 | 0101 |
| **11** | 000011000110011 | +11111001110011011  =00000010001010111  00001000101011100 | 000000000001001 | 0100 |
| **12** | 000110001100110 | +11110011100110011  =11111100010001111  11110001000111100 | 000000000000100 | 0011 |
| **13** | 001100011001101 | +00011000110011011  =00001001111010111  00100111101011100 | 000000000000010 | 0010 |
| **14** | 011000110011010 | +11001110011001011  =11110110000100111  11011000010011100 | 000000000000001 | 0001 |
| **15** | **110001100110101** | +01100011001101011  =00111011100000111  11101110000011100 | 000000000000000 | 0000 |

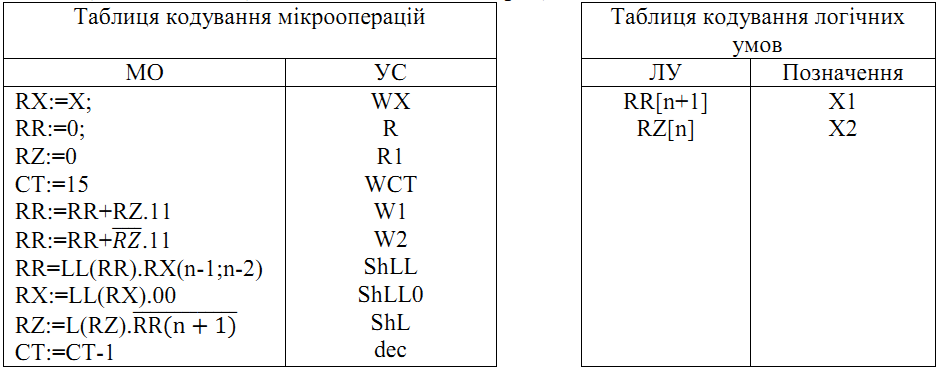
**2.8.5Функціональна схема операції обчислення квадратного кореня**

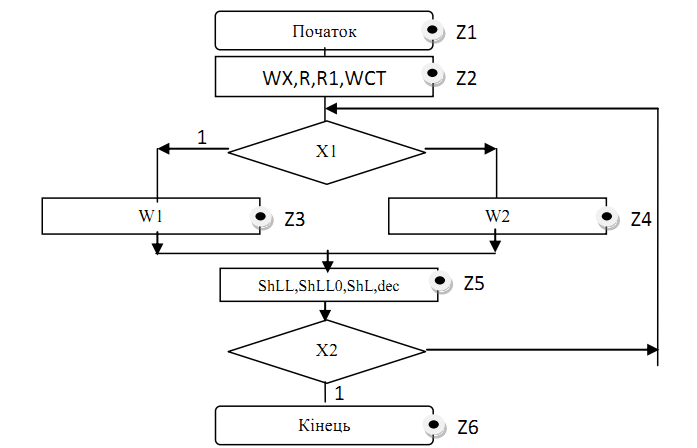


*Рисунок 2.8.3 – Функціональна схема*

**2.8.6 Закодований мікроалгоритм**

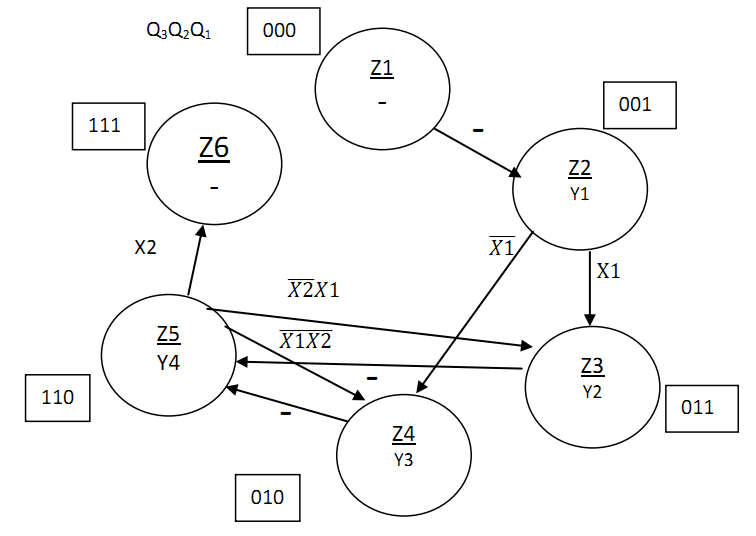
*Таблиця 2.8.2 – Таблиця кодування*





*Рисунок 2.8.4 – Закодований мікроалгоритм*

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



*Рисунок 2.8.5 – Граф управляючого автомата Мура*

**2.8.8 Обробка порядків**

В моєму випадку =4;

**2.8.9Запис результату**

Отримали результат Z = **110001100110101**;

Результат нормалізований, готовий до запису у мантису:

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

**Завдання 3**

x3 x2 x1 +1 = 0102 = 210.

Синтез управляючого автомату Мура на D-тригерах для операції множення другим способом

**3.1 Таблиця кодування сигналів**

*Таблиця 3.1 – Таблиця кодування сигналів*

|  |  |
| --- | --- |
| R,W2,W3 | Y1 |
| W1 | Y2 |
| ShR, ShL | 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**



*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 | | | |
| Q3Q2Q1 | Q3Q2Q1 | X2X1 | Y1Y2Y3 | D3 | D2 | | D1 |
| Z1→Z2 | 000 | | 001 | - - | 0 0 0 | 0 | | 0 | 1 |
| Z2→Z3 | 001 | | 011 | - 1 | 1 0 0 | 0 | | 1 | 1 |
| Z2→Z4 | 001 | | 010 | - 0 | 1 0 0 | 0 | | 1 | 0 |
| Z3→Z4 | 011 | | 010 | - - | 0 1 0 | 0 | | 1 | 0 |
| Z4→Z3 | 010 | | 011 | 0 1 | 0 0 1 | 0 | | 1 | 1 |
| Z4→Z4 | 010 | | 010 | 0 0 | 0 0 1 | 0 | | 1 | 0 |
| Z4→Z5 | 010 | | 110 | 1 - | 0 0 1 | 1 | | 1 | 0 |

**3.5 Мінімізація функцій тригерів**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |
|  |  | Q1 | |  |  |  |
| Q3 | Q2 | - | - | - | - |  |
| - | - | - | - | X1 |
|  | - | - | - | - |
|  | - | - | - | - |  |
|  | Q2 | 1 | 1 | 1 | 1 |  |
|  | 1 | 1 | 1 | 1 | X1 |
|  |  | 1 | 1 | 0 | 0 |
|  |  | 1 | 1 | 0 | 0 |  |
|  |  |  | X2 | |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |
|  |  | Q1 | |  |  |  |
| Q3 | Q2 | - | - | - | - |  |
| - | - | - | - | X1 |
|  | - | - | - | - |
|  | - | - | - | - |  |
|  | Q2 | 0 | 0 | 1 | 0 |  |
|  | 0 | 0 | 1 | 0 | X1 |
|  |  | 0 | 0 | 0 | 0 |
|  |  | 0 | 0 | 0 | 0 |  |
|  |  |  | X2 | |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |
|  |  | Q1 | |  |  |  |
| Q3 | Q2 | - | - | - | - |  |
| - | - | - | - | X1 |
|  | - | - | - | - |
|  | - | - | - | - |  |
|  | Q2 | 0 | 0 | 0 | 0 |  |
|  | 0 | 0 | 0 | 1 | X1 |
|  |  | 1 | 1 | 1 | 1 |
|  |  | 0 | 0 | 1 | 1 |  |
|  |  |  | X2 | |  |  |

*Рисунок 3.3 – Мінімізація функцій тригерів*

Y2

Y3

Y1

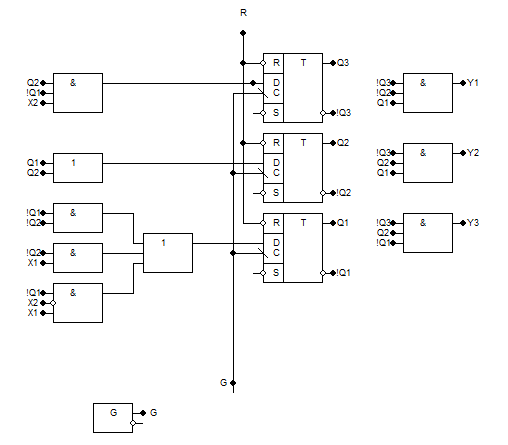
|  |  |  |  |
| --- | --- | --- | --- |
| *0* | *0* | *0* | *0* |
| *0* | *0* | *1* | *0* |

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

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

*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, 2009. *–* 360 с.

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

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