НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

# "КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ"

#### ФАКУЛЬТЕТ ІНФОРМАТИКИ ТА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

### Кафедра обчислювальної техніки

## РОЗРАХУНКОВА РОБОТА

по курсу „Комп'ютерна логіка-2”

Виконав: Власов Максим Дмитрович

Група ІО-34, Факультет ІОТ,

Залікова книжка № 3405

Номер технічного завдання 1101 0100 1101

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

(підпис керівника)

Київ – 2014 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 добування кореня з ). Номери операцій (для п.3) відповідають порядку переліку (наприклад, 1 – множення першим способом; 6 – ділення другим способом; 8 – добування кореня). 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 нульового результату.

3. Для операції з двійковим номером +1 побудувати управляючий автомат Мура на тригерах 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).

340510=1101010011012;

 = -10110101,0101101;

 = +10110,1010101011;

**Основна частина:**

**Завдання №1**

=1.10110101,0101101;

=0.10110,1010101011;

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

X2:

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

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

Y2:

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

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

Ey = 10000000 + 100 = 10000100

Y2:

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

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

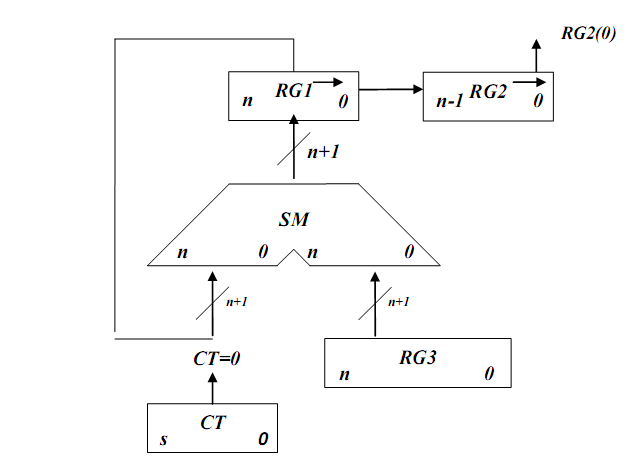
Множення мантис першим способом здійснюється з молодших розрядів множника, сума часткових добутків зсувається вправо, а множене залишається нерухомим. Тоді добуток двох чисел представляється у вигляді:

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

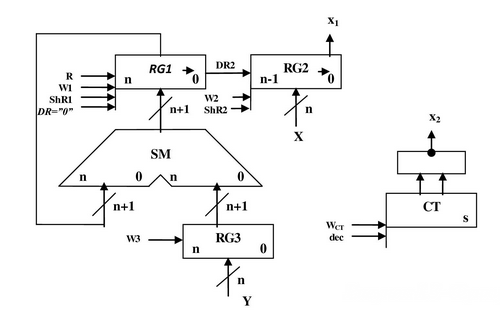
0

*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** | 0 | 101101010101101 | 101101010101011 | 1111 |
| **1** | 0010110101010101 | 110110101010110 |  | 1110 |
| **2** | 0001011010101010 | 111011010101011 |  | 1101 |
| **3** | +0101101010101011  =0111000101010101  0011100010101010 | 111101101010101 |  | 1100 |
| **4** | +0101101010101011  =1001001101010101  0100100110101010 | 111110110101010 |  | 1011 |
| **5** | 0010010011010101 | 011111011010101 |  | 1010 |
| **6** | +0101101010101011  =0111111110000000  0011111111000000 | 001111101101010 |  | 1001 |
| **7** | 0001111111100000 | 000111110110101 |  | 1000 |
| **8** | +0101101010101011  =0111101010001011  0011110101000101 | 100011111011010 |  | 0111 |
| **9** | 0001111010100010 | 110001111101101 |  | 0110 |
| **10** | +0101101010101011  =0111100101001101  0011110010100110 | 111000111110110 |  | 0101 |
| **11** | 0001111001010011 | 011100011111011 |  | 0100 |
| **12** | +0101101010101011  =0111100011111110  0011110001111111 | 001110001111101 |  | 0011 |
| **13** | +0101101010101011  =1001011100101010  0100101110010101 | 000111000111110 |  | 0010 |
| **14** | 0010010111001010 | 100011100011111 |  | 0001 |
| **15** | +0101101010101011  =1000000001110101  **0100000000111010** | **110001110001111** |  | 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: 0100000000111010110001110001111

Знак мантиси: 1 0 = 1.

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

Порядок зменшуємо на 1:

100000000111010110001110001111;=12;

Запишемо нормалізований результат:

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

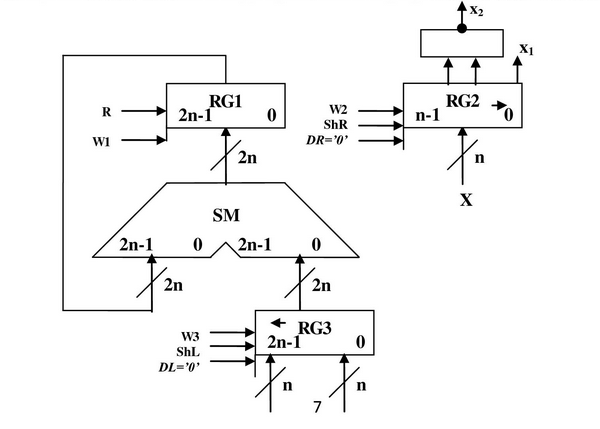
Множення мантис другим способом здійснюється з молодших розрядів, множене зсувається вліво, а сума часткових добутків залишається нерухомою.

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

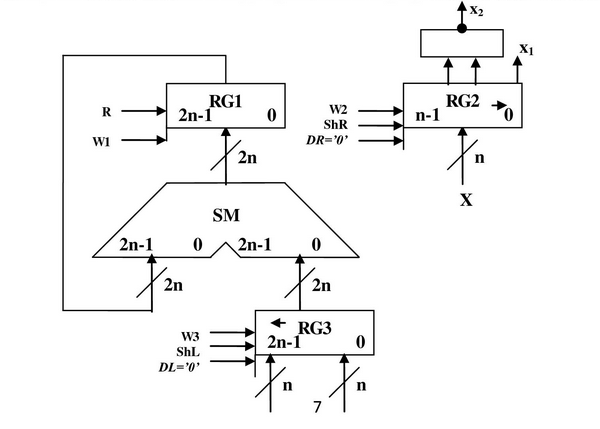
*Рисунок 2.2.2 - Змістовний мікроалгоритм.*

**2.2.4 Таблиця станів регістрів:**

*Таблиця 2.2.1-Таблиця станів регістрів.*

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **RG1** | **RG3 ←** | **RG2 →** |
| **ПС** | 0 | 101101010101011 | 101101010101101 |
| **1** | 0000000000000000101101010101011 | 000000000000001011010101010110 | 010110101010110 |
| **2** | 0000000000000000101101010101011 | 000000000000010110101010101100 | 001011010101011 |
| **3** | +0000000000000010110101010101100  =0000000000000011100010101010111 | 000000000000101101010101011000 | 000101101010101 |
| **4** | +0000000000000101101010101011000  =0000000000001001001101010101111 | 000000000001011010101010110000 | 000010110101010 |
| **5** | 0000000000001001001101010101111 | 000000000010110101010101100000 | 000001011010101 |
| **6** | +0000000000010110101010101100000  =0000000000011111111000000001111 | 000000000101101010101011000000 | 000000101101010 |
| **7** | 0000000000011111111000000001111 | 000000001011010101010110000000 | 000000010110101 |
| **8** | +0000000001011010101010110000000  =0000000001111010100010110001111 | 000000010110101010101100000000 | 000000001011010 |
| **9** | 0000000001111010100010110001111 | 000000101101010101011000000000 | 000000000101101 |
| **10** | +0000000101101010101011000000000  =0000000111100101001101110001111 | 000001011010101010110000000000 | 000000000010110 |
| **11** | 0000000111100101001101110001111 | 000010110101010101100000000000 | 000000000001011 |
| **12** | +0000010110101010101100000000000  =0000011110001111111001110001111 | 000101101010101011000000000000 | 000000000000101 |
| **13** | +0000101101010101011000000000000  =0001001011100101010001110001111 | 001011010101010110000000000000 | 000000000000010 |
| **14** | 0001001011100101010001110001111 | 010110101010101100000000000000 | 000000000000001 |
| **15** | +0010110101010101100000000000000  =**0100000000111010110001110001111** | 101101010101011000000000000000 | 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: 0100000000111010110001110001111

Знак мантиси: 1 0 = 1.

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

Порядок зменшуємо на 1:

100000000111010110001110001111;=12;

Запишемо нормалізований результат:

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

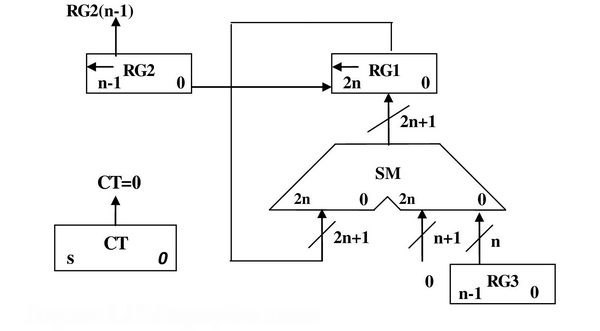
Множення мантис третім способом здійснюється зі старших розрядів множника, сума часткових добутків і множник зсуваються вліво, а множене нерухоме.

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

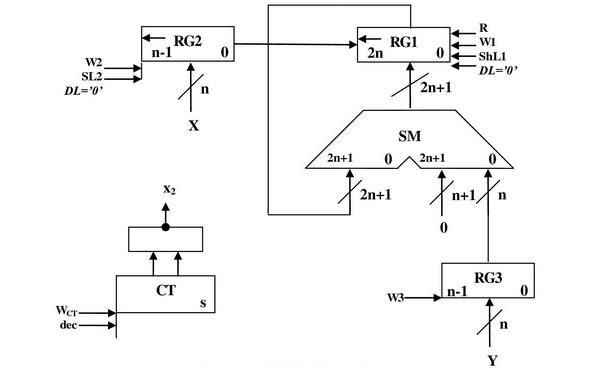
*Рисунок 2.3.2* **-** *Змістовний мікроалгоритм.*

**2.3.4 Таблиця станів регістрів:**

*Таблиця 2.3.1- Таблиця станів регістрів*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** | **RG1 ←** | **RG2 ←** | **RG3** | **CT** |
| **пс** | 0000000000000000000000000000000 | 101101010101101 | 101101010101011 | 1111 |
| **1** | 0000000000000001011010101010110 | 011010101011010 |  | 1110 |
| **2** | 0000000000000010110101010101100 | 110101010110100 |  | 1101 |
| **3** | +000000000000000101101010101011  =000000000000011100010101010111  000000000000111000101010101110 | 101010101101000 |  | 1100 |
| **4** | +000000000000000101101010101011  =000000000000111110010101011001  000000000001111100101010110010 | 010101011010000 |  | 1011 |
| **5** | 000000000011111001010101100100 | 101010110100000 |  | 1010 |
| **6** | +000000000000000101101010101011  =000000000011111111000000001111  000000000111111110000000011110 | 010101101000000 |  | 1001 |
| **7** | 000000001111111100000000111100 | 101011010000000 |  | 1000 |
| **8** | +000000000000000101101010101011  =000000010000000001101011100111  000000100000000011010111001110 | 010110100000000 |  | 0111 |
| **9** | 000001000000000110101110011100 | 101101000000000 |  | 0110 |
| **10** | +000000000000000101101010101011  =000001000000001100011001000111  0000010000000011000110010001110 | 011010000000000 |  | 0101 |
| **11** | 0000100000000110001100100011100 | 110100000000000 |  | 0100 |
| **12** | +0000000000000000101101010101011  =0000100000000110111001111000111  0001000000001101110011110001110 | 101000000000000 |  | 0011 |
| **13** | +0000000000000000101101010101011  =0001000000001110100001000111001  0010000000011101000010001110010 | 010000000000000 |  | 0010 |
| **14** | 0100000000111010000100011100100 | 100000000000000 |  | 0001 |
| **15** | +0000000000000000101101010101011  =**010000000011101011000111000111** | 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: 010000000011101011000111000111

Знак мантиси: 1 0 = 1.

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

порядок зменшуємо на 1:

10000000011101011000111000111;=12;

Запишемо нормалізований результат:

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

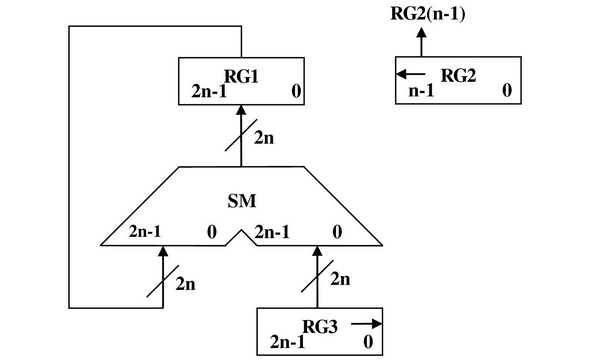
Множення здійснюється зі старших розрядів множника, сума часткових добутків залишається нерухомою, множене зсувається праворуч, множник ліворуч.

.

*.*

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

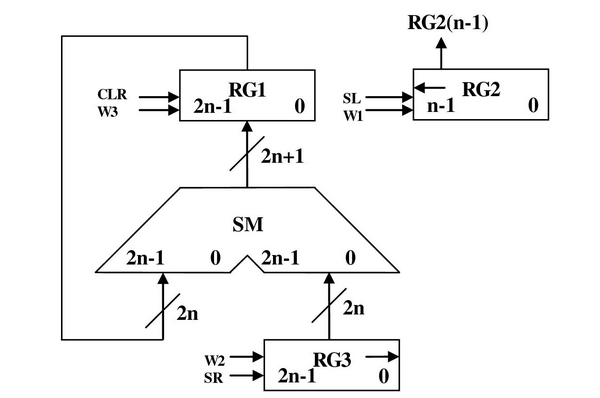
*Рисунок 2.4.2* **-** *Змістовний мікроалгоритм.*

**2.4.4 Таблиця станів регістрів:**

*Таблиця 2.4.1- Таблиця станів регістрів*

|  |  |  |  |
| --- | --- | --- | --- |
| № | RG1 | RG3 → | RG2 ← |
| ПС | 0000000000000000000000000000000 | 010110101010101100000000000000 | 101101010101101 |
| **1** | 0010110101010101100000000000000 | 001011010101010110000000000000 | 011010101011010 |
| **2** | 0010110101010101100000000000000 | 000101101010101011000000000000 | 110101010110100 |
| **3** | +0000101101010101011000000000000  =0011100010101010111000000000000 | 000010110101010101100000000000 | 101010101101000 |
| **4** | +0000010110101010101100000000000  =0011111001010101100100000000000 | 000001011010101010110000000000 | 010101011010000 |
| **5** | 0011111001010101100100000000000 | 000000101101010101011000000000 | 101010110100000 |
| **6** | +0000000101101010101011000000000  =0011111111000000001111000000000 | 000000010110101010101100000000 | 010101101000000 |
| **7** | 0011111111000000001111000000000 | 000000001011010101010110000000 | 101011010000000 |
| **8** | +0000000001011010101010110000000  =0100000000011010111001110000000 | 000000000101101010101011000000 | 010110100000000 |
| **9** | 0100000000011010111001110000000 | 000000000010110101010101100000 | 101101000000000 |
| **10** | +0000000000010110101010101100000  =0100000000110001100100011100000 | 000000000001011010101010110000 | 011010000000000 |
| **11** | 0100000000110001100100011100000 | 000000000000101101010101011000 | 110100000000000 |
| **12** | +0000000000000101101010101011000  =0100000000110111001111000111000 | 000000000000010110101010101100 | 101000000000000 |
| **13** | +0000000000000010110101010101100  =0100000000111010000100011100100 | 000000000000001011010101010110 | 010000000000000 |
| **14** | 0100000000111010000100011100100 | 000000000000000101101010101011 | 100000000000000 |
| **15** | +0000000000000000101101010101011  =**0100000000111010110001110001111** | 000000000000000010110101010101 | 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: 0100000000111010110001110001111

Знак мантиси: 1 0 = 1.

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

Порядок понижаємо на 1:

100000000111010110001110001111;=12;

Запишемо нормалізований результат:

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

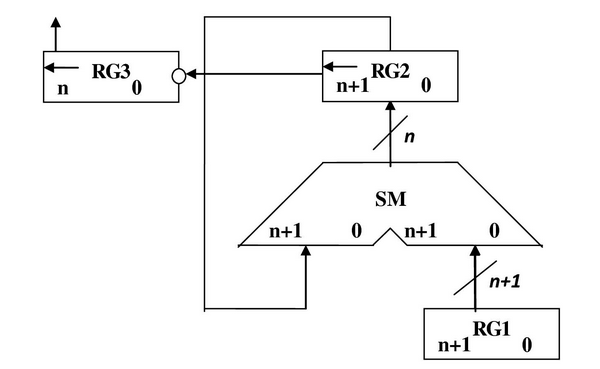
**2.5. Першиий спосіб ділення.**

**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

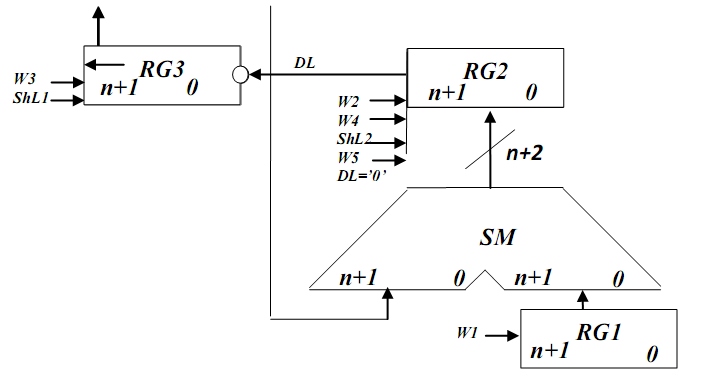
RG2[n+1]

*Рисунок 2.5.2-Змістовний мікроалгоритм*

**2.5.4 Таблиця станів регістрів:**

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **RG3(Z)** | **RG2(X)** | **RG1(Y)** |
| **пс** | 000000000000000 | 00101101010101101 | 101101010101011 |
| **1** | 0000000000000001 | 01011010101011010  +11010010101010101  =00101101010101111 |  |
| **2** | 0000000000000011 | 01011010101011110  +11010010101010101  =00101101010110011 |  |
| **3** | 0000000000000111 | 01011010101100110  +11010010101010101  =00101101010111011 |  |
| **4** | 0000000000001111 | 01011010101110110  +11010010101010101  =00101101011001011 |  |
| **5** | 0000000000011111 | 01011010110010110  +11010010101010101  =00101101011101011 |  |
| **6** | 0000000000111111 | 01011010111010110  +11010010101010101  =00101101100101011 |  |
| **7** | 0000000001111111 | 01011011001010110  +11010010101010101  =00101101110101011 |  |
| **8** | 0000000011111111 | 01011011101010110  +11010010101010101  =00101110010101011 |  |
| **9** | 0000000111111111 | 01011100101010110  +11010010101010101  =00101111010101011 |  |
| **10** | 0000001111111111 | 01011110101010110  +11010010101010101  =00110001010101011 |  |
| **11** | 0000011111111111 | 01100010101010110  +11010010101010101  =00110101010101011 |  |
| **12** | 0000111111111111 | 01101010101010110  +11010010101010101  =00111101010101011 |  |
| **13** | 0001111111111111 | 01111010101010110  +11010010101010101  =01001101010101011 |  |
| **14** | 0011111111111111 | 10011010101010110  +00101101010101011  =11001000000000001 |  |
| **15** | 0111111111111110 | 10010000000000010  +00101101010101011  =10111101010101101 |  |
| **16** | **1111111111111100** | 01111010101011010  +11010010101010101  =01001101010101111 |  |

**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

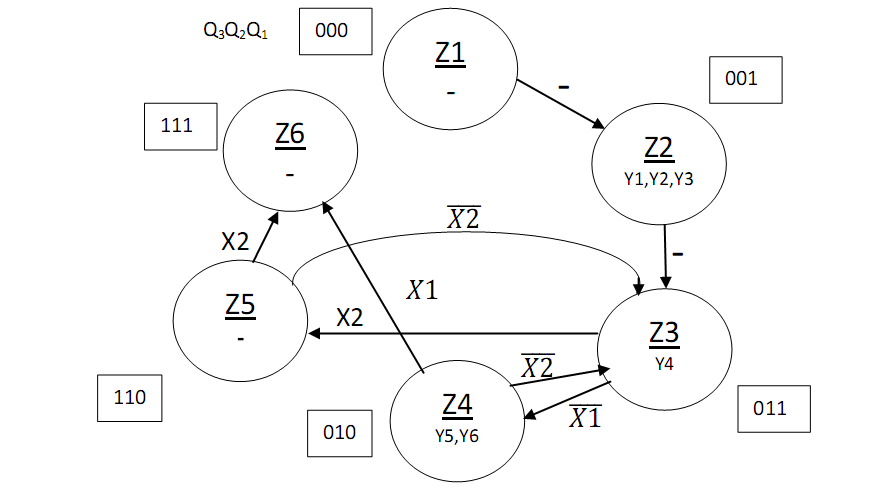
Z6

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: 1111111111111100

Знак мантиси: 1 0 = 1.

Нормалізація мантиси не потрібна.

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

**2.6. Другий спосіб ділення.**

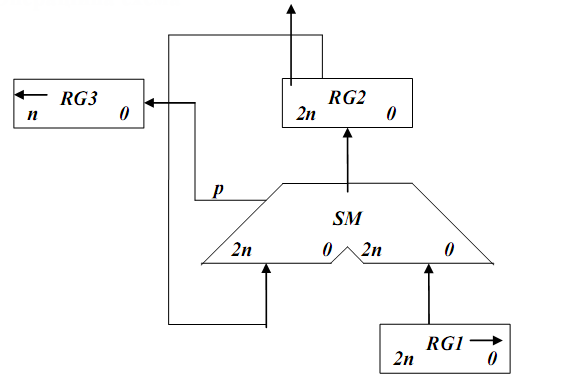
**2.6.1Теоритичне обгрунтування другого способу ділення:**

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

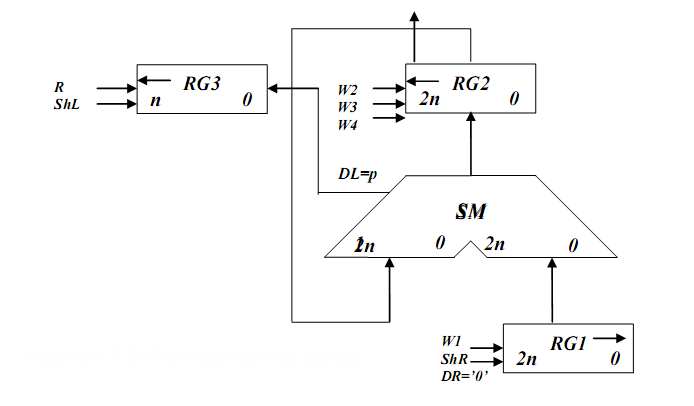
*Рисунок 2.6.2-Змістовний мікроалгоритм*

**2.6.4 Таблиця станів регістрів**

*Таблиця 2.6.1- Таблиця станів регістрів*

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **RG3(Z)** | **RG2(X)** | **RG1(Y)** |
| **ПС** | 0000000000000001 | 010110101010110100000000000000 | 001011010101010110000000000000 |
| **1** | 0000000000000011 | 010110101010110100000000000000  +110100101010101010000000000000  =001011010101011110000000000000 | 000101101010101011000000000000 |
| **2** | 0000000000000111 | 001011010101011110000000000000  +111010010101010101000000000000  =000101101010110011000000000000 | 000010110101010101100000000000 |
| **3** | 0000000000001111 | 000101101010110011000000000000  +111101001010101010100000000000  =000010110101011101100000000000 | 000001011010101010110000000000 |
| **4** | 0000000000011111 | 000010110101011101100000000000  +111110100101010101010000000000  =000001011010110010110000000000 | 000000101101010101011000000000 |
| **5** | 0000000000111111 | 000001011010110010110000000000  +111111010010101010101000000000  =000000101101011101011000000000 | 000000010110101010101100000000 |
| **6** | 0000000001111111 | 000000101101011101011000000000  +111111101001010101010100000000  =000000010110110010101100000000 | 000000001011010101010110000000 |
| **7** | 0000000011111111 | 000000010110110010101100000000  +111111110100101010101010000000  =000000001011011101010110000000 | 000000000101101010101011000000 |
| **8** | 0000000111111111 | 000000001011011101010110000000  +111111111010010101010101000000  =000000000101110010101011000000 | 000000000010110101010101100000 |
| **9** | 0000001111111111 | 000000000101110010101011000000  +111111111101001010101010100000  =000000000010111101010101100000 | 000000000001011010101010110000 |
| **10** | 0000011111111111 | 000000000010111101010101100000  +111111111110100101010101010000  =000000000001100010101010110000 | 000000000000101101010101011000 |
| **11** | 0000111111111111 | 000000000001100010101010110000  +111111111111010010101010101000  =000000000000110101010101011000 | 000000000000010110101010101100 |
| **12** | 0001111111111111 | 000000000000110101010101011000  +111111111111101001010101010100  =000000000000011110101010101100 | 000000000000001011010101010110 |
| **13** | 0011111111111111 | 000000000000011110101010101100  +111111111111110100101010101010  =000000000000010011010101010110 | 000000000000000101101010101011 |
| **14** | 0111111111111111 | 000000000000010011010101010110  +111111111111111010010101010101  =000000000000001101101010101011 | 000000000000000010110101010101 |
| **15** | 1111111111111111 | 000000000000001101101010101011  +111111111111111101001010101011  =000000000000001010110101010110 | 000000000000000001011010101010 |

**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

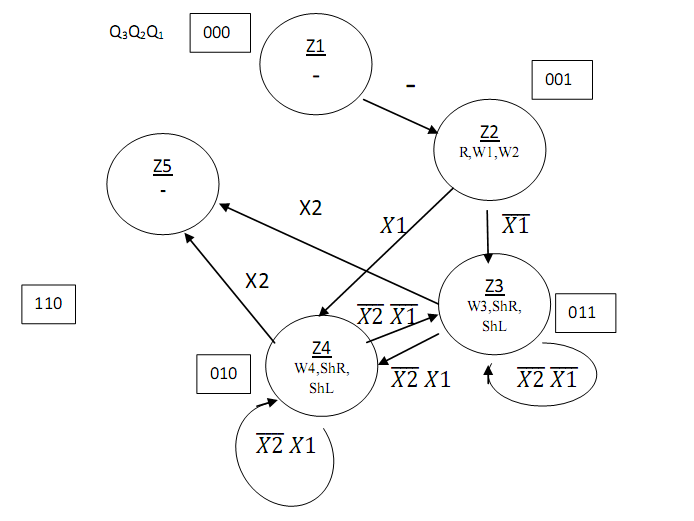
W4, ShR, ShL

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: 1111111111111111

Знак мантиси: 1 0 = 1.

Нормалізація мантиси не потрібна.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 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, 101101010101011 | 11 | Початковий стан |
| 0, 010110101010101 | 10 | My= 0.r(My); ∆:=∆-1 |
| 0, 001011010101010 | 01 | My= 0.r(My); ∆:=∆-1 |
| 0,000101101010101 | 00 | My= 0.r(My); ∆:=∆-1 |

3. Додавання мантис у модифікованому ДК.

X мдк = 11.010010101010011

Yмдк = 00.000101101010101

*Таблиця 2.7.2-Додавання мантис(для додавання)*

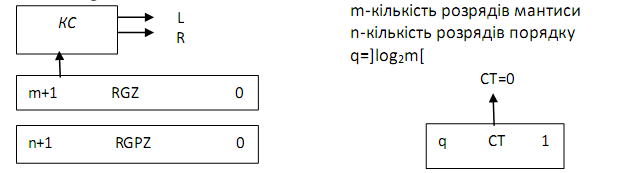
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| MX | 1 | 1. | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 |
| MY | 0 | 0. | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 |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

Zпк = 1. 100111101010111

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*

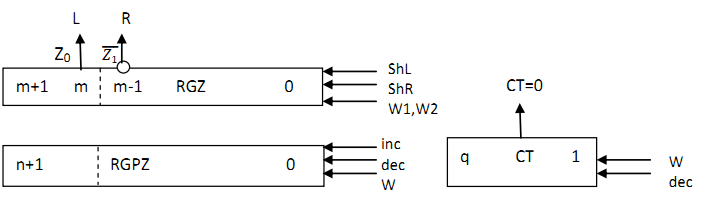
**2.7.4 Таблиця станів регістрів**

**Додавання**

*Таблиця 2.7.5- Таблиця станів регістрів*

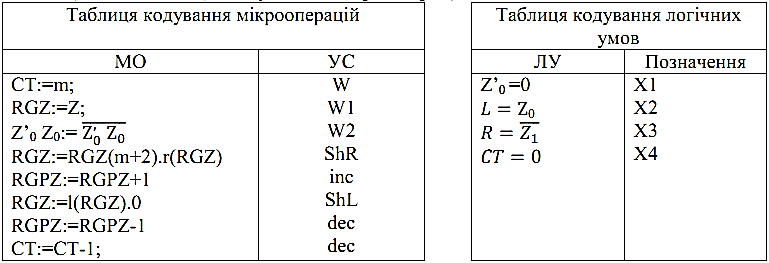
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№**  **такту** | **RGPZ** | **RGZ** | **ЛПН(L)** | **ППН(R)** | **СT** | **Мікрооперація** |
| **ПС** | 001000 | 11. 100111101010111 | 0 | 1 | 100 |  |

**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.

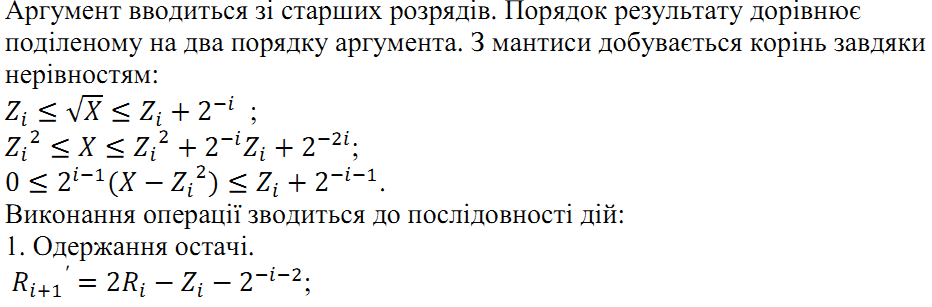
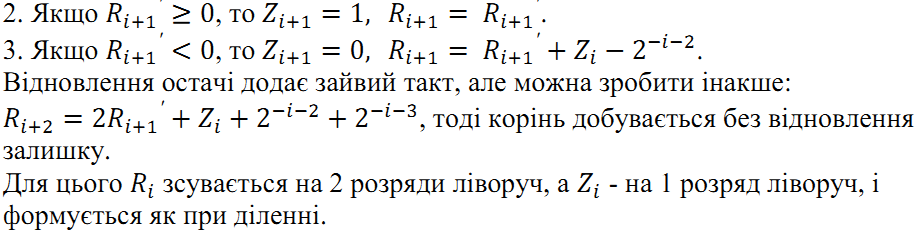
Zпк = 1. 100111101010111

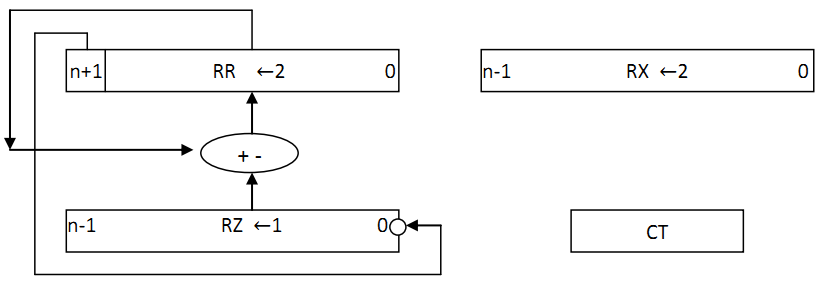
Pz = 810 =10002 Mz = 1001111010101112

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

**2.8.Операція добування кореня**

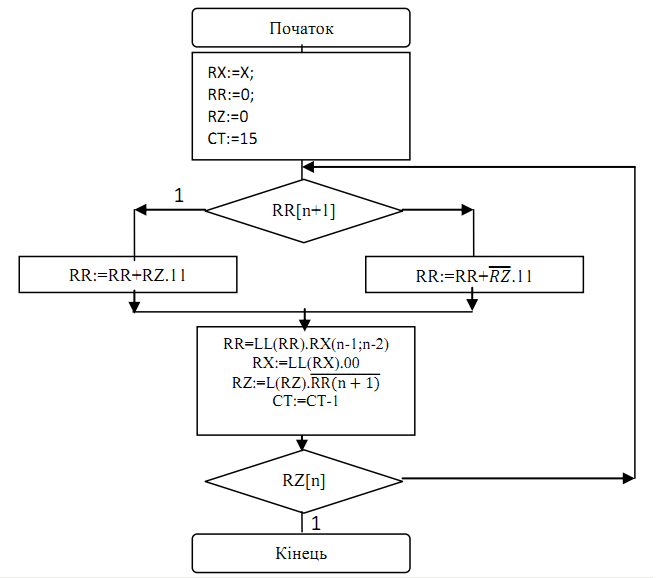
**2.8.1Теоритичне обґрунтування операції обчислення квадратного кореня**



**2.8.2 Операційна схема операції обчислення квадратного кореня**

*Рисунок 2.8.1 –Операційна схема*

**2.8.3 Змістовний мікроалгоритм**



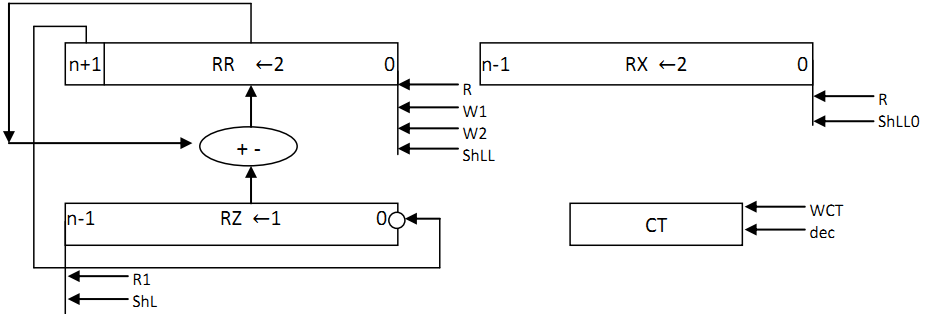
*Рисунок 2.8.2 – Змістовний мікроалгоритм*

**2.8.4 Таблиця станів регістрів**

*Таблиця 2.8.1 – Таблиця станів регістрів*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | RZ | RR | RX | СТ |
| **ПС** | 000000000000000 | 00000000000000000  00000000000000010 | 101101010101101 | 1111 |
| **ПЗ** |
| **1** | 000000000000001 | 00000000000000010  +11111111111111111  =00000000000000001  00000000000000111 | 010110101010110 | 1110 |
| **2** | 000000000000011 | 00000000000000111  +11111111111111011  =00000000000000010  00000000000001001 | 001011010101011 | 1101 |
| **3** | 000000000000110 | 00000000000001001  +11111111111110011  =11111111111111100  11111111111110001 | 000101101010101 | 1100 |
| **4** | 000000000001101 | 11111111111110001  +00000000000011011  =00000000000001100  00000000000110001 | 000010110101010 | 1011 |
| **5** | 000000000011010 | 00000000000110001  +11111111111001011  =11111111111111100  11111111111110001 | 000001011010101 | 1010 |
| **6** | 000000000110101 | 11111111111110001  +00000000001101011  =00000000001011100  00000000101110010 | 000000101101010 | 1001 |
| **7** | 000000001101011 | 00000000101110010  +11111111100101011  =00000000010011101  00000001001110110 | 000000010110101 | 1000 |
| **8** | 000000011010111 | 00000001001110110  +11111111001010011  =00000000011001001  00000001100100100 | 000000001011010 | 0111 |
| **9** | 000000110101110 | 00000001100100100  +11111110010100011  =11111111111000111  11111111100011100 | 000000000101101 | 0110 |
| **10** | 000001101011101 | 11111111100011100  +00000011010111011  =00000010111010111  00001011101011100 | 000000000010110 | 0101 |
| **11** | 000011010111011 | 00001011101011100  +11111001010001011  =00000100111100111  00010011110011100 | 000000000001011 | 0100 |
| **12** | 000110101110111 | 00010011110011100  +11110010100010011  =00000110010101111  00011001010111100 | 000000000000101 | 0011 |
| **13** | 001101011101110 | 00011001010111100  +11100101000100011  =11111110011011111  11111001101111100 | 000000000000010 | 0010 |
| **14** | 011010111011101 | 11111001101111100  +00110101110111011  =00101111100110111  10111110011011100 | 000000000000001 | 0001 |
| **15** | **110101110111011** | 10111110011011100  +01101011101110111  =00101010001010011  10101000101001100 | 000000000000000 | 0000 |

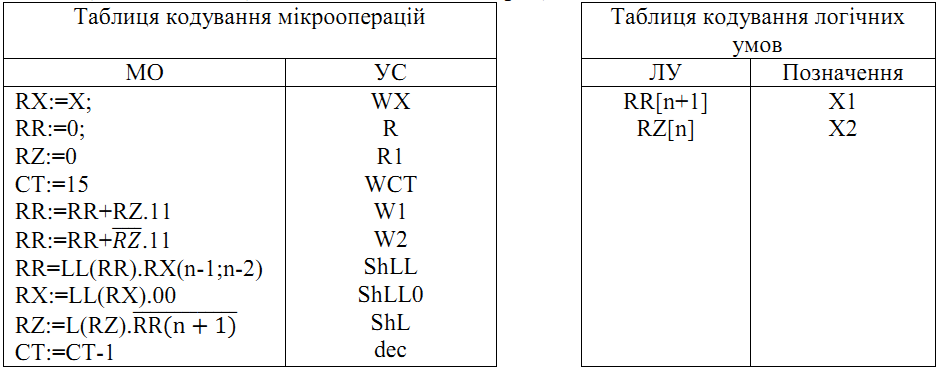
**2.8.5 Функціональна схема операції обчислення квадратного кореня**

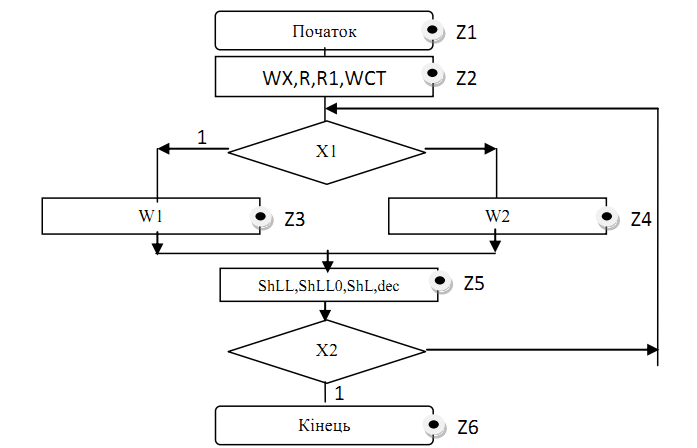


*Рисунок 2.8.3 – Функціональна схема*

**2.8.6 Закодований мікроалгоритм**

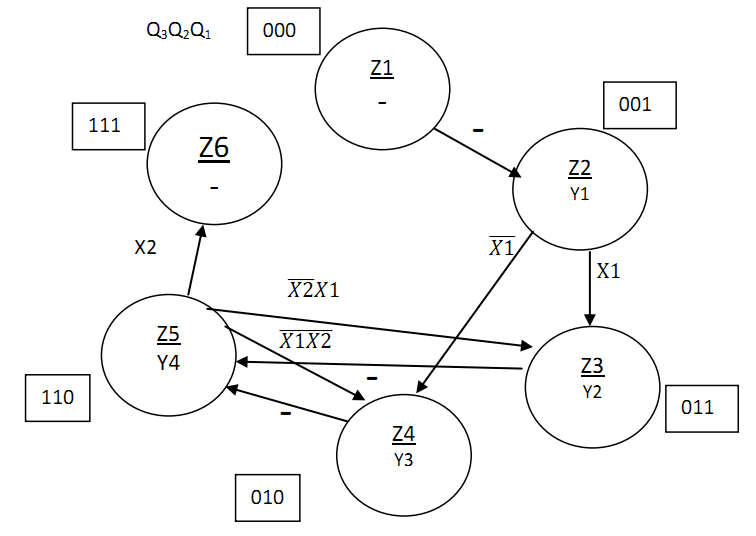
*Таблиця 2.8.2 – Таблиця кодування*





*Рисунок 2.8.4 – Закодований мікроалгоритм*

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



*Рисунок 2.8.5 – Граф управляючого автомата Мура*

**2.8.8 Обробка порядків**

В моєму випадку =4;

**2.8.9 Запис результату**

Отримали результат Z = 110101110111011;

Результат нормалізований, готовий до запису у мантису:

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

**Завдання 3**

x3 x2 x1 +1 = 1102 = 610.

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

**3.1 Таблиця кодування сигналів**

*Таблиця 3.1 – Таблиця кодування сигналів*

|  |  |
| --- | --- |
| R,W1,W2 | Y1 |
| W3, ShR, ShL | Y2 |
| W4, ShR, ShL | Y3 |

**3.2 Мікроалгоритм в термінах управляючого автомата**

Початок

R, W1, W2

X1

Кінець

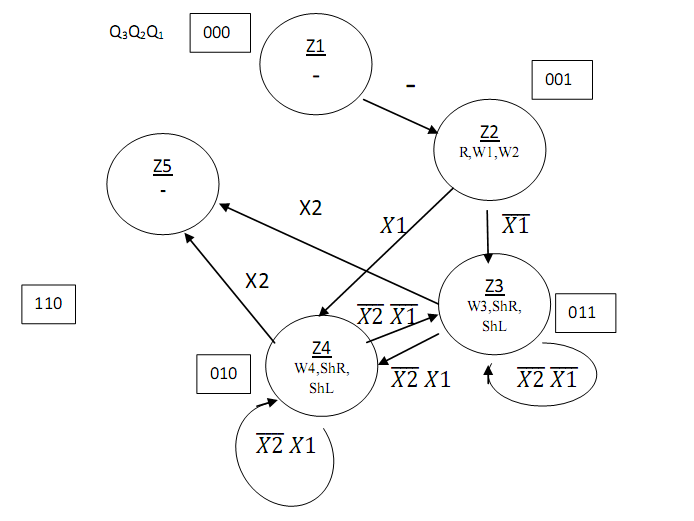
W4, ShR, ShL

W3, ShR, ShL

X2

*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 | -- | 000 | 0 | | 0 | 1 |
| Z2→Z3 | 001 | | 011 | 0- | 100 | 0 | | 1 | 1 |
| Z2→Z4 | 001 | | 010 | 1- | 100 | 0 | | 1 | 0 |
| Z3→Z4 | 011 | | 010 | 10 | 010 | 0 | | 1 | 0 |
| Z4→Z3 | 010 | | 011 | 00 | 001 | 0 | | 1 | 1 |
| Z4→Z5 | 010 | | 110 | -1 | 001 | 1 | | 1 | 0 |
| Z3→Z5 | 011 | | 110 | -1 | 010 | 1 | | 1 | 0 |
| Z3→Z3 | 011 | | 011 | 00 | 010 | 0 | | 1 | 1 |

**3.5 Мінімізація функцій тригерів**

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

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

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

*Рисунок 3.3 – Мінімізація функцій тригерів*

Y2

Y3

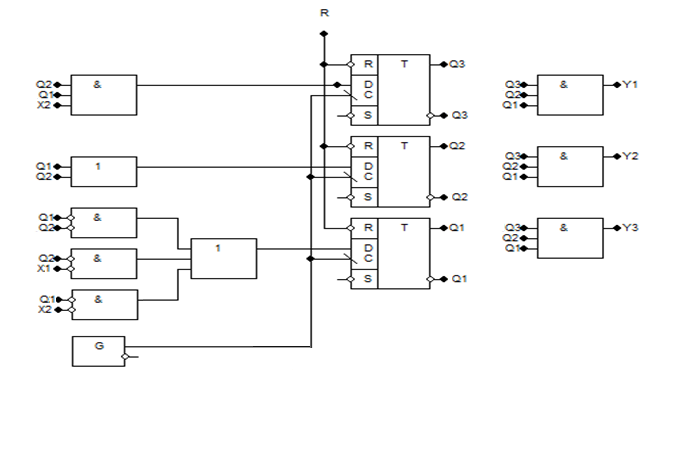
Y1

|  |  |  |  |
| --- | --- | --- | --- |
| *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 та додавання. Для операції ділення другим 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.

|  |  |
| --- | --- |
| ***Використана література:*** | |
| 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» |

Під час виконання даної розрахункової роботи я повторив для себе матеріал курсу «Компютерна логіка - 1», а також закріпив знання з курсу «Компютерна логіка - 2» (Ко мп’ютерна арифметика).