НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

# "КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ"

#### ФАКУЛЬТЕТ ІНФОРМАТИКИ І ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

### Кафедра обчислювальної техніки

## РОЗРАХУНКОВА РАБОТА

з дисципліни "Комп’ютерна арифметика"

Виконав: Шпурик Владислав

Група ІО-02, Факультет ІОТ,

Залікова книжка № 221

Номер технічного завдання 0011011101

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

(підпис керівника)

Київ – 2011 р.

**I. 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 та добування кореня з ). 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 нульового результату.

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

**II.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).

22110=110111012;

 = - 10011101,1101101;

 = +10101,1011101011;

**III.Ocновна частина:**

**Завдання 1**

=1.10011101,1101101;

=0.10101,1011101011;

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

Y2:

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

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

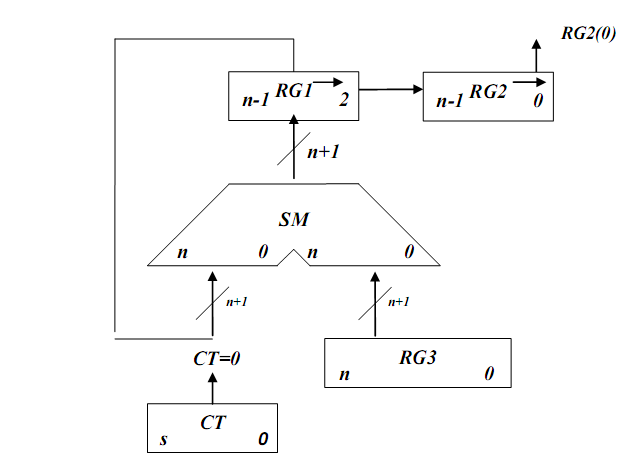
Множення мантис першим способом здійснюється з молодших розрядів множника, сума часткових добутків зсувається вправо, а множене залишається нерухомим. Тоді добуток двох чисел представляється у вигляді:

Z=Y+ Y…+ Y;

Z=(((0+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;

0

RG2[0]

1

RG1:=RG1+RG3;

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

0

CT=0

Кінець

1

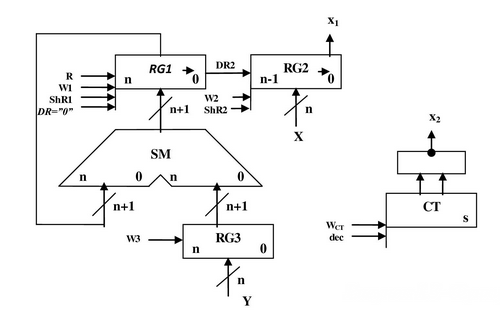
*Рисунок 2.1.2 - Змістовний мікроалгоритм.*

**2.1.4 Таблиця станів регістрів:**

*Таблиця 2.1.1-Таблиця станів регістрів.*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** | **RG1** | **RG2** | **RG3** | **CT** |
| **пс** | 0 | 100111011101101 | 101011011101011 | 1111 |
| **1** | +101011011101011  0101011011101011  0010101101110101 | 110011101110110 |  | 1110 |
| **2** | 0001010110111010 | 111001110111011 |  | 1101 |
| **3** | +101011011101011  0110110010100101  0011011001010010 | 001100101110101 |  | 1100 |
| **4** | +101011011101011  1000110100111101  0100011010011110 | 111110011101110 |  | 1011 |
| **5** | 0010001101001111 | 011111001110111 |  | 1010 |
| **6** | +101011011101011  0111101000111010  0011110100011101 | 001111100111011 |  | 1001 |
| **7** | +101011011101011  1001010000001000  0100101000000100 | 000111110011101 |  | 1000 |
| **8** | +101011011101011  1010000011101111  0101000001110111 | 100011111001110 |  | 0111 |
| **9** | 0010100000111011 | 110001111100111 |  | 0110 |
| **10** | +101011011101011  0111111100100110  0011111110010011 | 011000111110011 |  | 0101 |
| **11** | +101011011101011  1001011001111110  0100101100111111 | 001100011111001 |  | 0100 |
| **12** | +101011011101011  1010001000101010  0101000100010101 | 000110001111100 |  | 0011 |
| **13** | 0010100010001010 | 100011000111110 |  | 0010 |
| **14** | 0001010001000101 | 010001100011111 |  | 0001 |
| **15** | +101011011101011  0110101100110000  **0011010110011000** | **001000110001111** |  | 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 |

Z1

Початок

Z2

R, W2, W3, WCT

0

X1

W1

Z3

1

Z4

1

Кінець

X2

0

ShR1,ShR2,dec

Z5

*Рисунок 2.1.4-Закодований мікроалгоритм.*

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



*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: 011010110011000001000110001111

Знак мантиси: 1 0 = 1.

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

Порядок понижаємо на 1:

11010110011000001000110001111;=12;

Запишемо нормалізований результат:

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

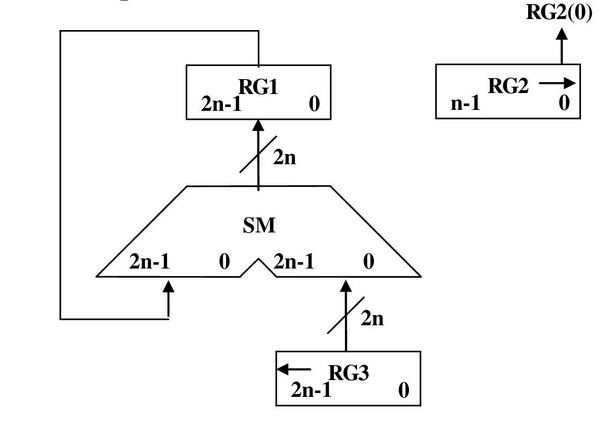
Множення мантис другим способом здійснюється з молодших розрядів, множене зсувається вліво, а сума часткових добутків залишається нерухомою.

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:**

Початок

RG2[0]

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

1

01

RG1:=RG1+RG3;

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

101

RX=0

01

Кінець

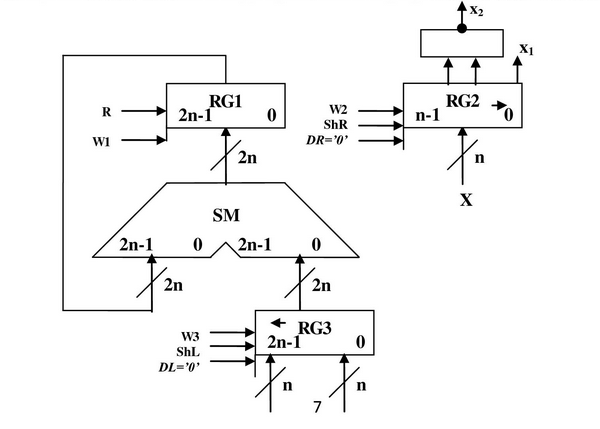
*Рисунок 2.2.2 - Змістовний мікроалгоритм.*

**2.2.4 Таблиця станів регістрів:**

*Таблиця 2.2.1-Таблиця станів регістрів.*

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **RG1** | **RG3** | **RG2** |
| **пс** | 0 | 000000000000000101011011101011 | 100111011101101 |
| **1** | +000000000000000101011011101011  000000000000000101011011101011 | 000000000000001010110111010110 | 010011101110110 |
| **2** | 000000000000000101011011101011 | 000000000000010101101110101100 | 001001110111011 |
| **3** | +000000000000010101101110101100  000000000000011011001010010111 | 000000000000101011011101011000 | 000100111011101 |
| **4** | +000000000000101011011101011000  000000000001000110100111101111 | 000000000001010110111010110000 | 000010011101110 |
| **5** | 000000000001000110100111101111 | 000000000010101101110101100000 | 000001001110111 |
| **6** | +000000000010101101110101100000  000000000011110100011101001111 | 000000000101011011101011000000 | 000000100111011 |
| **7** | +000000000101011011101011000000  000000001001010000001000001111 | 000000001010110111010110000000 | 000000010011101 |
| **8** | +000000001010110111010110000000  000000010100000111011110001111 | 000000010101101110101100000000 | 000000001001110 |
| **9** | 000000010100000111011110001111 | 000000101011011101011000000000 | 000000000100111 |
| **10** | +000000101011011101011000000000  000000111111100100110110001111 | 000001010110111010110000000000 | 000000000010011 |
| **11** | +000001010110111010110000000000  000010010110011111100110001111 | 000010101101110101100000000000 | 000000000001001 |
| **12** | +000010101101110101100000000000  000101000100010101000110001111 | 000101011011101011000000000000 | 000000000000100 |
| **13** | 000101000100010101000110001111 | 001010110111010110000000000000 | 000000000000010 |
| **14** | 000101000100010101000110001111 | 010101101110101100000000000000 | 000000000000001 |
| **15** | +010101101110101100000000000000  **011010110011000001000110001111** | 101011011101011000000000000000 | 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 |

Початок Z1

R,W2,W3 Z2

0

X1

1

W1 Z3

ShR, ShL Z4

0

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: 011010110011000001000110001111

Знак мантиси: 1 0 = 1.

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

Порядок понижаємо на 1:

11010110011000001000110001111;=12;

Запишемо нормалізований результат:

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

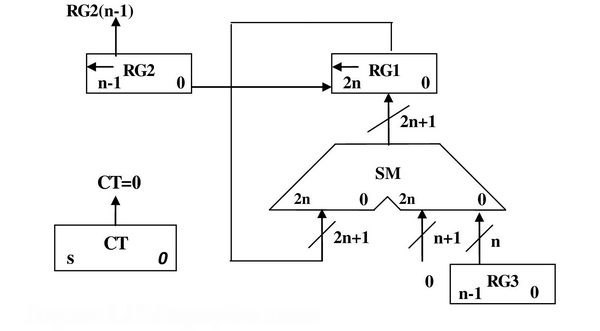
Множення мантис третім способом здійснюється зі старших розрядів множника, сума часткових добутків і множник зсуваються вліво, а множене нерухоме.

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:**

Початок

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

01

RG2[n-1]

1

RG1:=RG1+RG3;

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

01

CT=0

1

Кінець

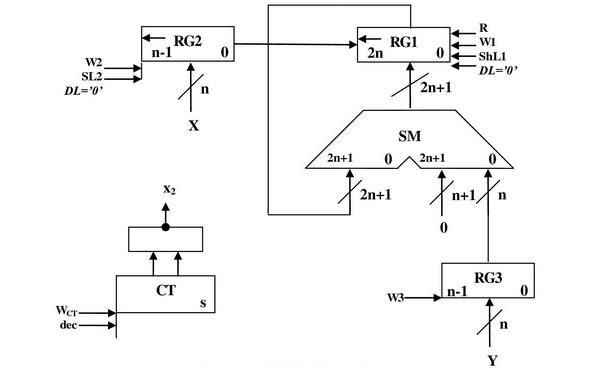
*Рисунок 2.3.2* **-** *Змістовний мікроалгоритм.*

**2.3.4 Таблиця станів регістрів:**

*Таблиця 2.3.1- Таблиця станів регістрів*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** | **RG1** | **RG2** | **RG3** | **CT** |
| **пс** | 0000000000000000000000000000000 | 100111011101101 | 101011011101011 | 1111 |
| **1** | +100111011101101  0000000000000000100111011101101  0000000000000001001110111011010 | 100111011101101 | 010110111010110 | 1110 |
| **2** | 0000000000000010011101110110100 | 100111011101101 | 101101110101100 | 1101 |
| **3** | +100111011101101  0000000000000011000101010100001  0000000000000110001010101000010 | 100111011101101 | 011011101011000 | 1100 |
| **4** | 0000000000001100010101010000100 | 100111011101101 | 110111010110000 | 1011 |
| **5** | +100111011101101  0000000000001100111100101110001  0000000000011001111001011100010 | 100111011101101 | 101110101100000 | 1010 |
| **6** | +100111011101101  0000000000011010100000111001111  0000000000110101000001110011110 | 100111011101101 | 011101011000000 | 1001 |
| **7** | 0000000001101010000011100111100 | 100111011101101 | 111010110000000 | 1000 |
| **8** | +100111011101101  0000000001101010101011000101001  0000000011010101010110001010010 | 100111011101101 | 110101100000000 | 0111 |
| **9** | +100111011101101  0000000011010101111101100111111  0000000110101011111011001111110 | 100111011101101 | 101011000000000 | 0110 |
| **10** | +100111011101101  0000000110101100100010101101011  0000001101011001000101011010110 | 100111011101101 | 010110000000000 | 0101 |
| **11** | 0000011010110010001010110101100 | 100111011101101 | 101100000000000 | 0100 |
| **12** | +100111011101101  0000011010110010110010010011001  0000110101100101100100100110010 | 100111011101101 | 011000000000000 | 0011 |
| **13** | 0001101011001011001001001100100 | 100111011101101 | 110000000000000 | 0010 |
| **14** | +100111011101101  0001101011001011110000101010001  0011010110010111100001010100010 | 100111011101101 | 100000000000000 | 0001 |
| **15** | +100111011101101  0011010110011000001000110001111  **0110101100110000010001100011110** | 100111011101101 | 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 |

R, W2, W3, WCT

X1

W1

ShL1,ShL2,dec

X2

Кінець

1

0

1

0

Z2

Z3

Z4

Z5

Z1

Початок

*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: 0110101100110000010001100011110

Знак мантиси: 1 0 = 1.

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

Порядок понижаємо на 1:

11010110011000001000110001111;=12;

Запишемо нормалізований результат:

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

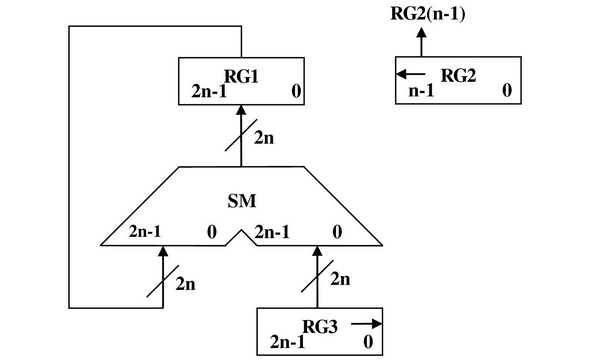
Множення здійснюється зі старших розрядів множника, сума часткових добутків залишається нерухомою, множене зсувається праворуч, множник ліворуч.

.

*.*

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:**

Початок

RG1:=0; RG2:=X; RG3:=Y; RG3:=0.r(RG3)

01

RG2[n-1]

1

RG1:=RG1+RG3;

RG3:=0.r(RG3)

RG2:=l(RG2).0

01

RG2=0

1

Кінець

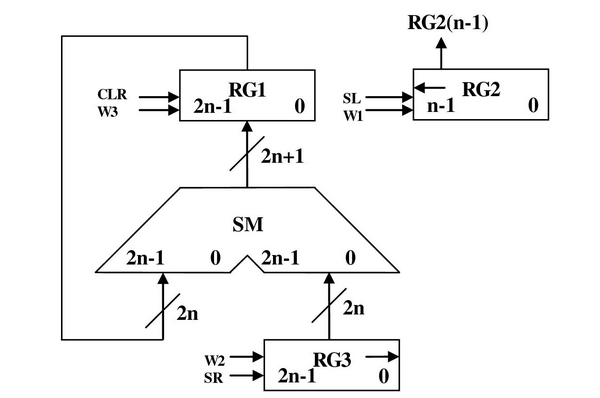
*Рисунок 2.4.2* **-** *Змістовний мікроалгоритм.*

**2.4.4 Таблиця станів регістрів:**

*Таблиця 2.4.1- Таблиця станів регістрів*

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **RG1** | **RG3** | **RG2** |
| **пс** | 000000000000000000000000000000 | 010101101110101100000000000000 | 100111011101101 |
| **1** | +010101101110101100000000000000  010101101110101100000000000000 | 001010110111010110000000000000 | 001110111011010 |
| **2** | 010101101110101100000000000000 | 000101011011101011000000000000 | 011101110110100 |
| **3** | 010101101110101100000000000000 | 000010101101110101100000000000 | 111011101101000 |
| **4** | +000010101101110101100000000000  011000011100100001100000000000 | 000001010110111010110000000000 | 110111011010000 |
| **5** | +000001010110111010110000000000  011001110011011100010000000000 | 000000101011011101011000000000 | 101110110100000 |
| **6** | +000000101011011101011000000000  011010011110111001101000000000 | 000000010101101110101100000000 | 011101101000000 |
| **7** | 011010011110111001101000000000 | 000000001010110111010110000000 | 111011010000000 |
| **8** | +000000001010110111010110000000  011010101001110000111110000000 | 000000000101011011101011000000 | 110110100000000 |
| **9** | +000000000101011011101011000000  011010101111001100101001000000 | 000000000010101101110101100000 | 101101000000000 |
| **10** | +000000000010101101110101100000  011010110001111010011110100000 | 000000000001010110111010110000 | 011010000000000 |
| **11** | 011010110001111010011110100000 | 000000000000101011011101011000 | 110100000000000 |
| **12** | +000000000000101011011101011000  011010110010100101111011111000 | 000000000000010101101110101100 | 101000000000000 |
| **13** | +000000000000010101101110101100  011010110010111011101010100100 | 000000000000001010110111010110 | 010000000000000 |
| **14** | 011010110010111011101010100100 | 000000000000000101011011101011 | 100000000000000 |
| **15** | +000000000000000101011011101011  **011010110011000001000110001111** | 000000000000000010101101110101 | 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 |

R, W2, W3, ShR

X1

W1

ShR,ShL

X2

Кінець

1

0

1

0

Z2

Z3

Z4

Z5

Z1

Початок

*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: 011010110011000001000110001111

Знак мантиси: 1 0 = 1.

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

Порядок понижаємо на 1:

11010110011000001000110001111;=12;

Запишемо нормалізований результат:

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

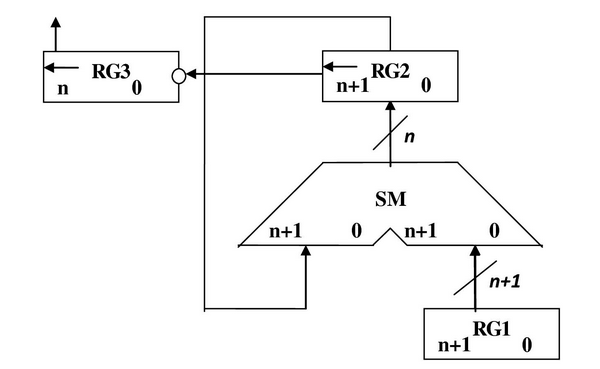
**2.5. Першиий спосіб ділення.**

**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.

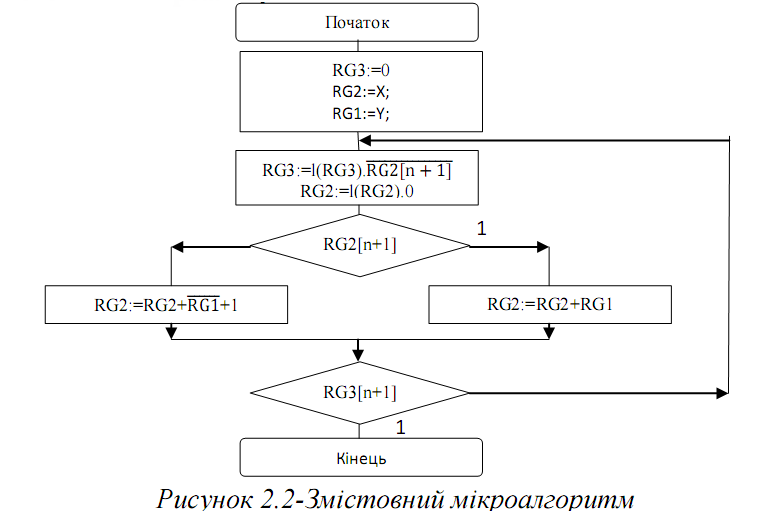
При реалізації ділення за першим методом здійснюється зсув вліво залишку при нерухомому дільнику. Черговий залишок формується в регістрі Р2 (у вихідному стані в цьому регістрі записаний Х). Виходи Р2 підключені до входів СМ безпосередньо, тобто ланцюги видачі коду з Р2 не потрібні. Час для підключення 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:**

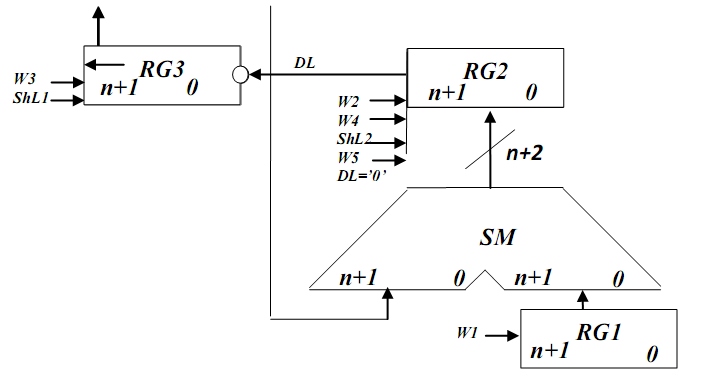


*Рисунок 2.5.2-Змістовний мікроалгоритм*

**2.5.4 Таблиця станів регістрів:**

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **RG3(Z)** | **RG2(X)** | **RG1(Y)** |
| **пс** | 000000000000000 | 00100111011101101 | 00101011011101011пк  11010100100010101дк |
| **1** | 0000000000000001 | 01001110111011010  +11010100100010101  =00100011011101111 |  |
| **2** | 0000000000000011 | 01000110111011110  +11010100100010101  =00011011011110011 |  |
| **3** | 0000000000000111 | 00110110111100110  +11010100100010101  =00001011011111011 |  |
| **4** | 0000000000001111 | 00010110111110110  +11010100100010101  =11101011100001011 |  |
| **5** | 0000000000011110 | 11010111000010110  +00101011011101011  =00000010100000001 |  |
| **6** | 0000000000111101 | 00000101000000010  +11010100100010101  =11011001100010111 |  |
| **7** | 0000000001111010 | 10110011000101110  +00101011011101011  =11011110100011001 |  |
| **8** | 0000000011110100 | 10111101000110010  +00101011011101011  =11101000100011101 |  |
| **9** | 0000000111101000 | 11010001000111010  +00101011011101011  =11111100100100101 |  |
| **10** | 0000001111010000 | 11111001001001010  +00101011011101011  =00100100100110101 |  |
| **11** | 0000011110100001 | 01001001001101010  +11010100100010101  =00011101101111111 |  |
| **12** | 0000111101000011 | 00111011011111110  +11010100100010101  =00010000000010011 |  |
| **13** | 0001111010000111 | 00100000000100110  +11010100100010101  =11110100100111011 |  |
| **14** | 0011110100001110 | 11101001001110110  +00101011011101011  =00010100101100001 |  |
| **15** | 0111101000011101 | 00101001011000010  +11010100100010101  =11111101111010111 |  |
| **16** | **1111010000111010** | 11111011110101110  +00101011011101011  =00100111010011001 |  |

**2.5.5 Функціональна схема:**

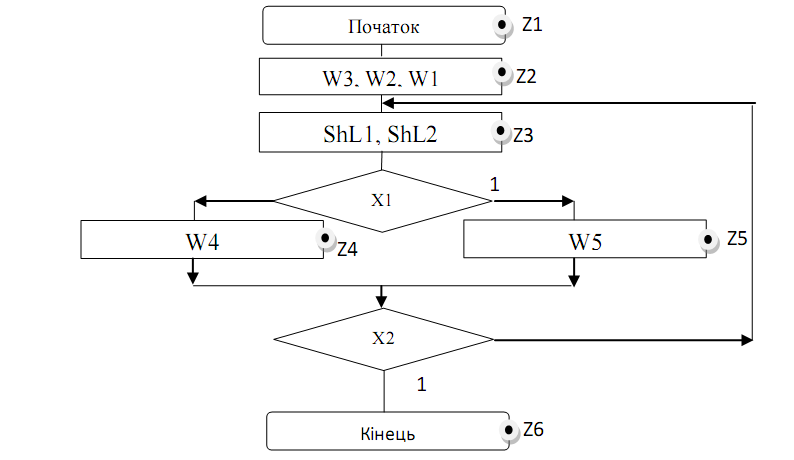


*Рисунок 2.5.3 – Функціональна схема*

**2.5.6 Закодований мікроалгоритм**

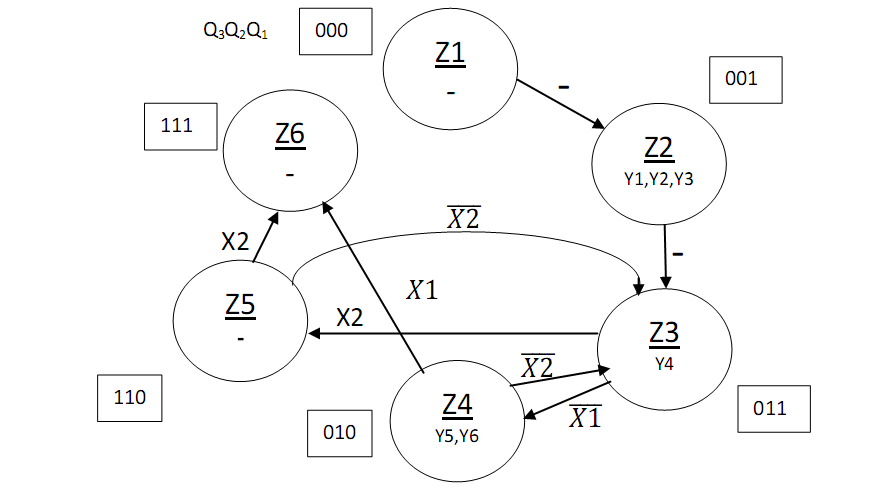
*Таблиця 2.5.2-Таблиця кодування операцій і логічних умов.*

|  |  |  |  |
| --- | --- | --- | --- |
| 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 |



*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: 1111010000111010

Знак мантиси: 1 0 = 1.

Нормалізація мантиси не потрібна.

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

**2.6. Другий спосіб ділення.**

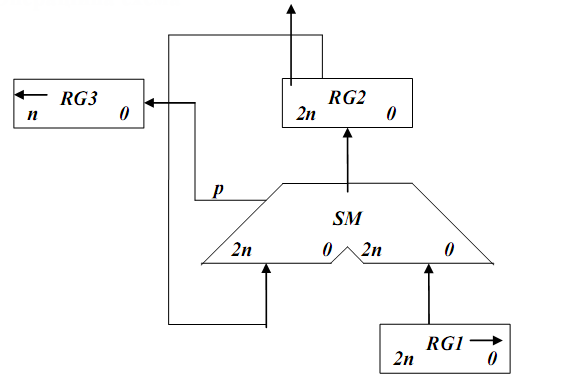
**2.6.1Теоритичне обгрунтування другого способу ділення:**

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

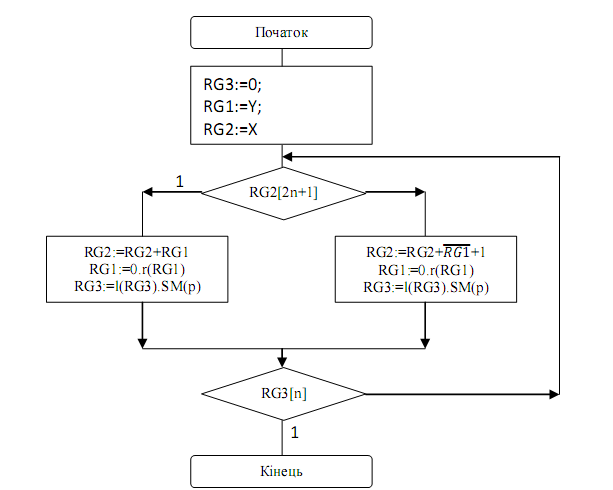
прискорення відносно 1-го способу.

**2.6.2 Операційна схема**



*Рисунок 2.6.1-Операційна схема*

**2.6.3 Змістовний мікроалгоритм**



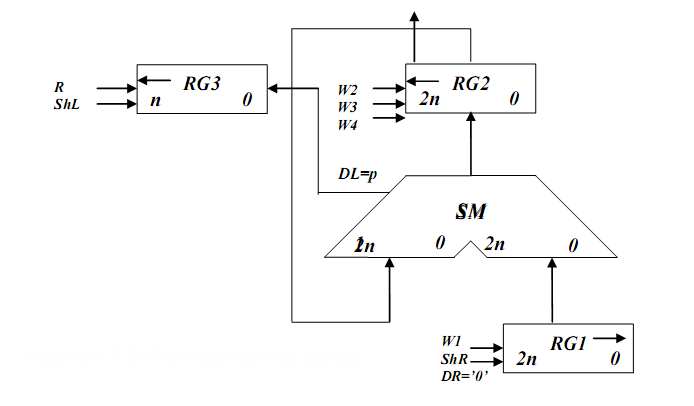
*Рисунок 2.6.2-Змістовний мікроалгоритм*

**2.6.4 Таблиця станів регістрів**

*Таблиця 2.6.1- Таблиця станів регістрів*

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **RG3(Z)** | **RG2(X)** | **RG1(Y)** |
| **пс** | 0000000000000001 | 010011101110110100000000000000 | 001010110111010110000000000000 |
| **1** | 0000000000000011 | +110101001000101010000000000000  =001000110111011110000000000000 | 000101011011101011000000000000 |
| **2** | 0000000000000111 | +111010100100010101000000000000  =000011011011110011000000000000 | 000010101101110101100000000000 |
| **3** | 0000000000001111 | +111101010010001010100000000000  =000000101101111101100000000000 | 000001010110111010110000000000 |
| **4** | 0000000000011110 | +111110101001000101010000000000  =111111010111000010110000000000 | 000000101011011101011000000000 |
| **5** | 0000000000111101 | +000000101011011101011000000000  =000000000010100000001000000000 | 000000010101101110101100000000 |
| **6** | 0000000001111010 | +111111101010010001010100000000  =111111101100110001011100000000 | 000000001010110111010110000000 |
| **7** | 0000000011110100 | +000000001010110111010110000000  =111111110111101000110010000000 | 000000000101011011101011000000 |
| **8** | 0000000111101000 | +000000000101011011101011000000  =111111111101000100011101000000 | 000000000010101101110101100000 |
| **9** | 0000001111010000 | +000000000010101101110101100000  =111111111111110010010010100000 | 000000000001010110111010110000 |
| **10** | 0000011110100001 | +000000000001010110111010110000  =000000000001001001001101010000 | 000000000000101011011101011000 |
| **11** | 0000111101000011 | +111111111111010100100010101000  =000000000000011101101111111000 | 000000000000010101101110101100 |
| **12** | 0001111010000111 | +111111111111101010010001010100  =000000000000001000000001001100 | 000000000000001010110111010110 |
| **13** | 0011110100001110 | +111111111111110101001000101010  =111111111111111101001001110110 | 000000000000000101011011101011 |
| **14** | 0111101000011101 | +000000000000000101011011101011  =000000000000000010100101100001 | 000000000000000010101101110101 |
| **15** | **1111010000111010** | +111111111111111101010010001011  =111111111111111111110111101100 | 000000000000000001010110111010 |

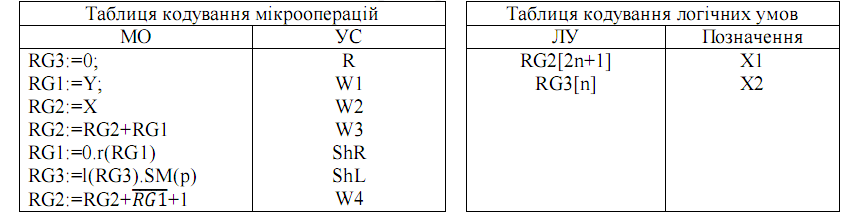
**2.6.5 Функціональна схема з відображенням управляючих сигналів**

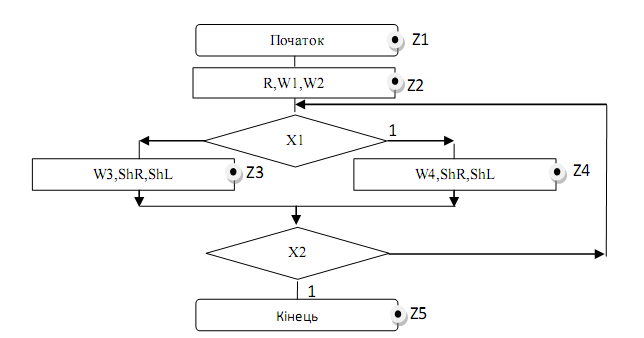


*Рисунок 2.6.3-Функціональна схема*

**2.6.6 Закодований мікроалгоритм**

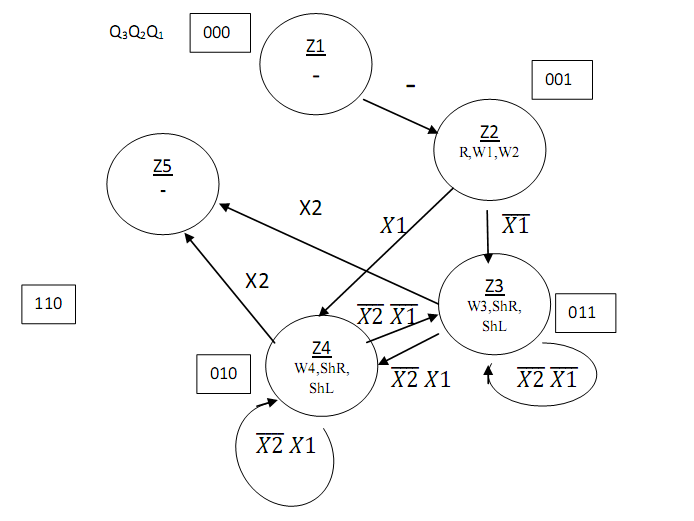
*Таблиця 2.6.2- Таблиця кодування мікрооперацій*





*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: 1111010000111010

Знак мантиси: 1 0 = 1.

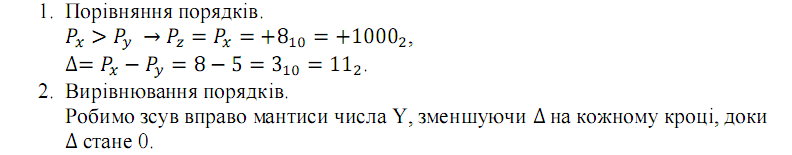
Нормалізація мантиси не потрібна.

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



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

|  |  |  |
| --- | --- | --- |
| MY | ∆ | Мікрооперація |
| 0. 101011011101011 | 11 | П.С. |
| 0.010101101110101 | 10 | ∆:=∆-1 |
| 0.001010110111010 | 01 | ∆:=∆-1 |
| 0.000101011011101 | 00 | ∆:=∆-1 |

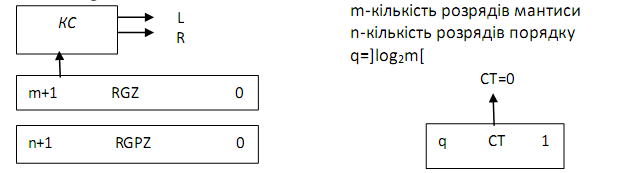
3. Додавання мантис у модифікованому ДК.

*Таблиця 2.7.2-Додавання мантис*

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

4. Нормалізація результату (В ПК).

**2.7.2 Операційна схема**



*Рисунок 2.7.1-Операційна схема*

Виконаємо синтез КС для визначення порушення нормалізації.

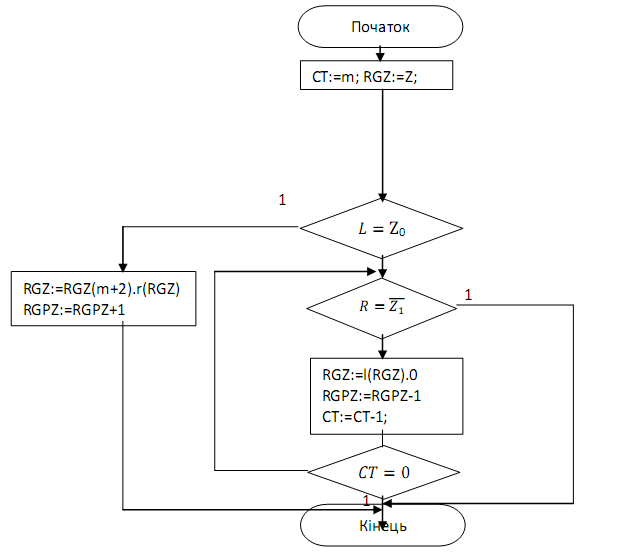
*Таблиця 2.7.3-Визначення порушення нормалізації*

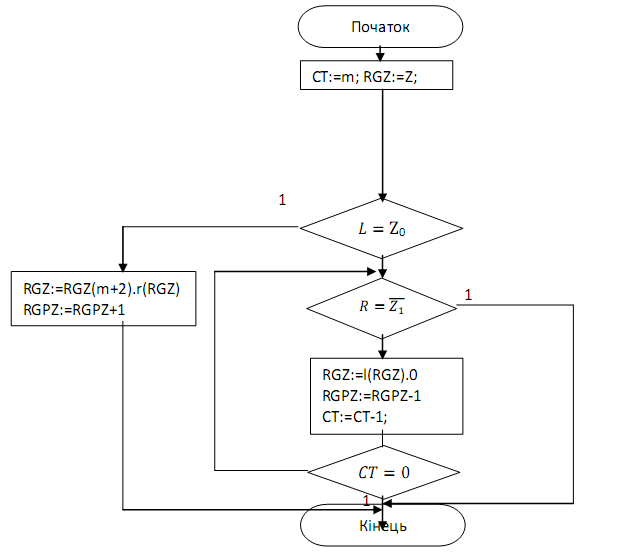
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Розряди регістру  RGZ | | | Значення  функцій | |
| Z’0 | Z0 | Z1 | L | R |
| 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 1 | 0 |



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

**2.7.5 Змістовний алгоритм**





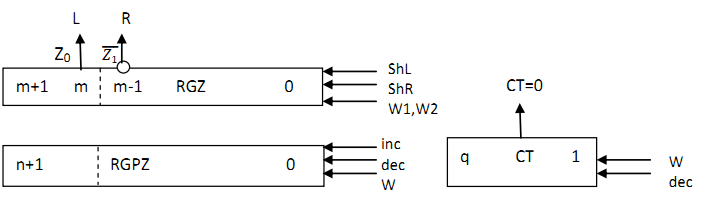
*Рисунок 2.7.2-Змістовний мікроалгоритм*

**2.7.4 Таблиця станів регістрів**

*Таблиця 2.7.4- 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 |  |

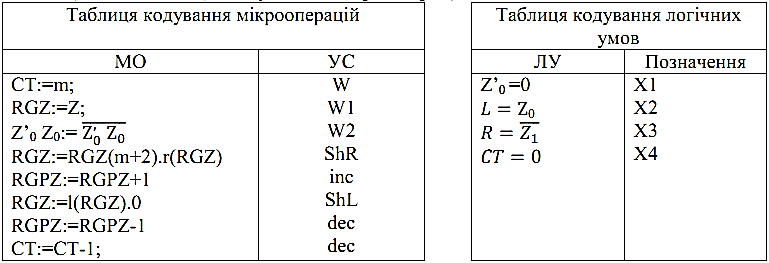
**2.7.5 Функціональна схема з відображенням керуючих сигналів**

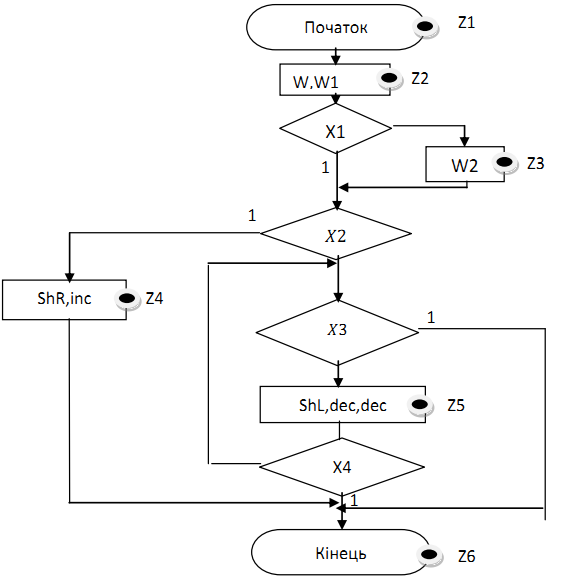
****

*Рисунок 2.7.3 – Функціональна схема*

**2.7.6 Закодований мікроалгоритм**

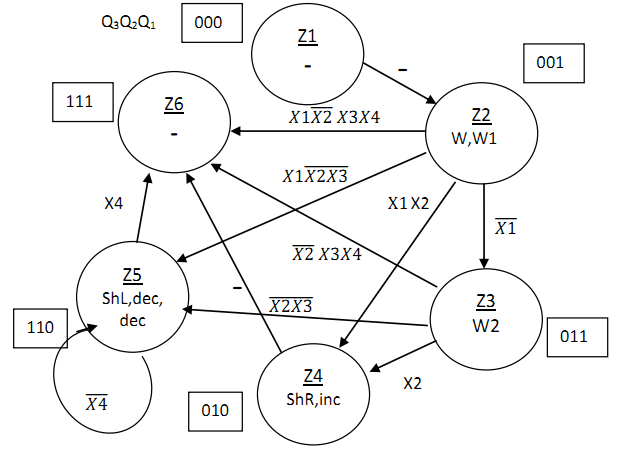
*Таблиця 2.7.5 – Таблиця кодування*

****

****

*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**

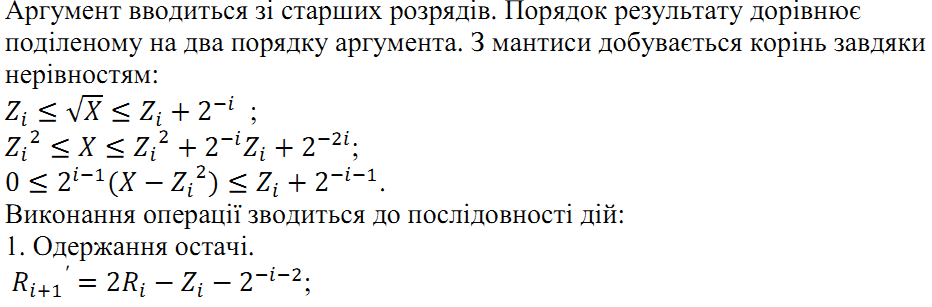
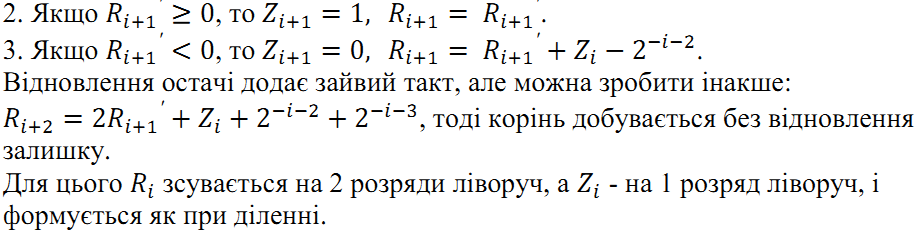
PZ=10002

**2.7.9 Форма запису результату з плаваючою комою**

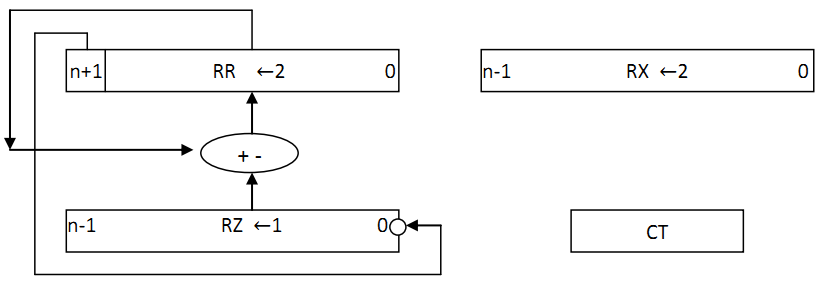
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |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

**2.8. Операція добування кореня**

**2.8.1 Теоритичне обгрунтування операції обчислення квадратного кореня**

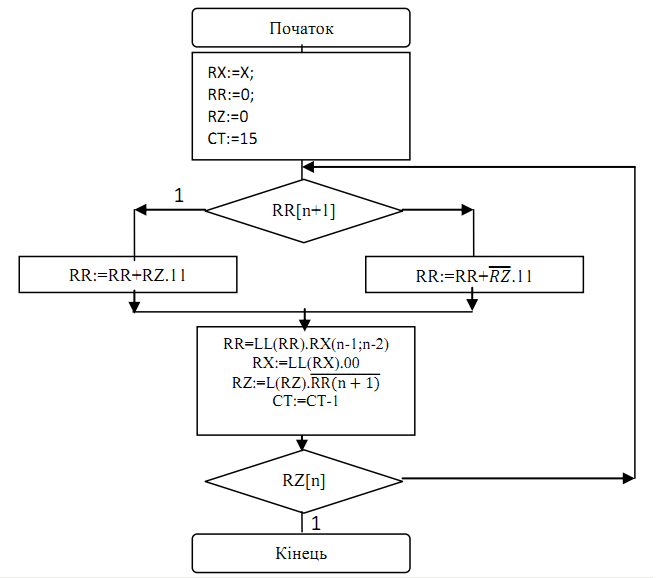


**2.8.2 Операційна схема операції обчислення квадратного кореня**



*Рисунок 2.8.1 – Операційна схема*

**2.8.3 Змістовний мікроалгоритм**



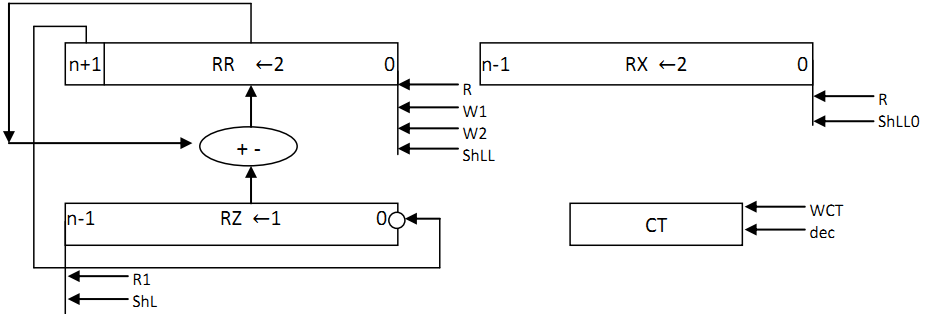
*Рисунок 2.8.2 – Змістовний мікроалгоритм*

**2.8.4 Таблиця станів регістрів**

*Таблиця 2.8.1 – Таблиця станів регістрів*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | RZ | RR | RX | СТ |
| **пс** | 000000000000000 | 00000000000000000  00000000000000010 | 100111011101101 | 1111 |
| **пз** |
| **1** | 000000000000001 | +11111111111111111  =00000000000000001  00000000000000101 | 010011101110110 | 1110 |
| **2** | 000000000000011 | +11111111111111011  =00000000000000000  00000000000000011 | 001001110111011 | 1101 |
| **3** | 000000000000110 | +11111111111110011  =11111111111110110  11111111111011001 | 000100111011101 | 1100 |
| **4** | 000000000001100 | +00000000000011011  =11111111111110100  11111111111010011 | 000010011101110 | 1011 |
| **5** | 000000000011001 | +00000000000110011  =00000000000000110  00000000000011001 | 000001001110111 | 1010 |
| **6** | 000000000110010 | +11111111110011011  =11111111110110100  11111111011010010 | 000000100111011 | 1001 |
| **7** | 000000001100100 | +00000000011001011  =11111111110011101  11111111001110110 | 000000010011101 | 1000 |
| **8** | 000000011001001 | +00000000110010011  =00000000000001001  00000000000100100 | 000000001001110 | 0111 |
| **9** | 000000110010010 | +11111110011011011  =11111110011111111  11111001111111100 | 000000000100111 | 0110 |
| **10** | 000001100100100 | +00000011001001011  =11111101001000111  11110100100011100 | 000000000010011 | 0101 |
| **11** | 000011001001000 | +00000110010010011  =11111010110101111  11101011010111100 | 000000000001001 | 0100 |
| **12** | 000110010010000 | +00001100100100011  =11110111111011111  11011111101111100 | 000000000000100 | 0011 |
| **13** | 001100100100000 | +00011001001000011  =11111000110111111  11100011011111100 | 000000000000010 | 0010 |
| **14** | 011001001000001 | +00110010010000011  =00010101101111111  01010110111111100 | 000000000000001 | 0001 |
| **15** | **110010010000010** | +10011011011111011  =11110010011110111  11001001111011100 | 000000000000000 | 0000 |

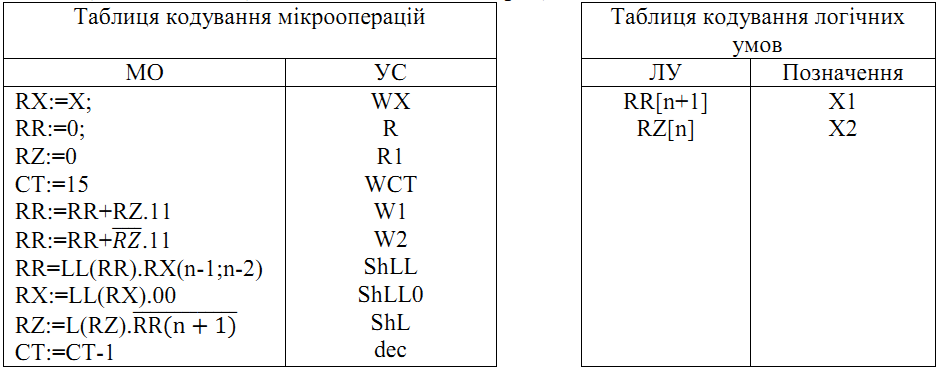
**2.8.5 Функціональна схема операції обчислення квадратного кореня**

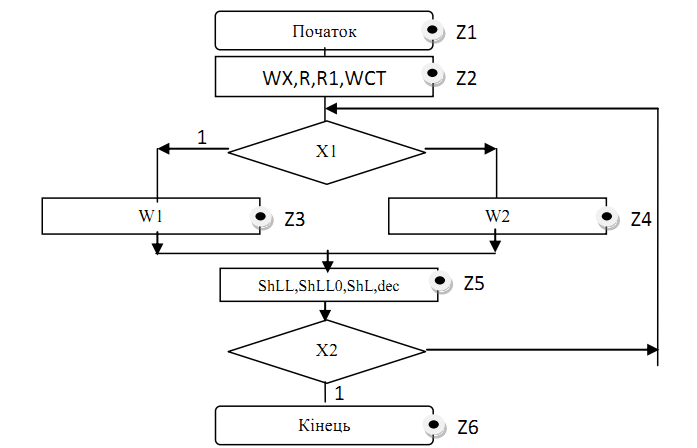


*Рисунок 2.8.3 – Функціональна схема*

**2.8.6 Закодований мікроалгоритм**

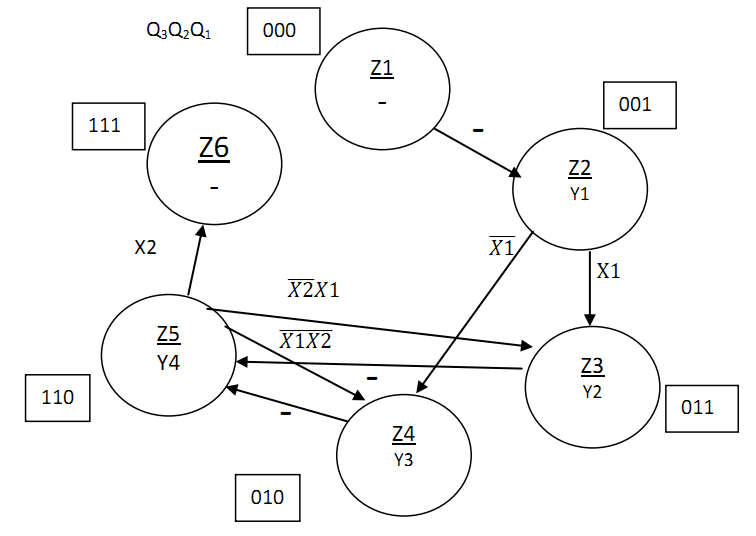
*Таблиця 2.8.2 – Таблиця кодування*





*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=110010010000010;

Результат нормалізований, готовий до запису у мантису:

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

**Завдання 3**

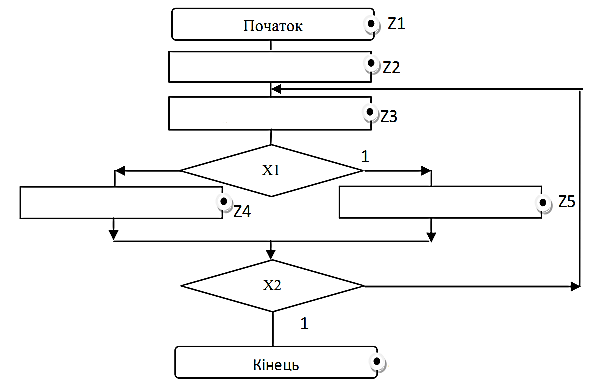
**Синтез управляючого автомату Мура на тригерах для операції ділення першим способом**

**3.1 Таблиця кодування сигналів**

*Таблиця 3.1 – Таблиця кодування сигналів*

|  |  |
| --- | --- |
| W3, W2, W1 | Y1 |
| ShL1, ShL2 | Y2 |
| W4 | Y3 |
| W5 | Y4 |

**3.2 Мікроалгоритм в термінах управляючого автомата**

****

**Z11**

Y4

Y3

Y2

Y1

*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*

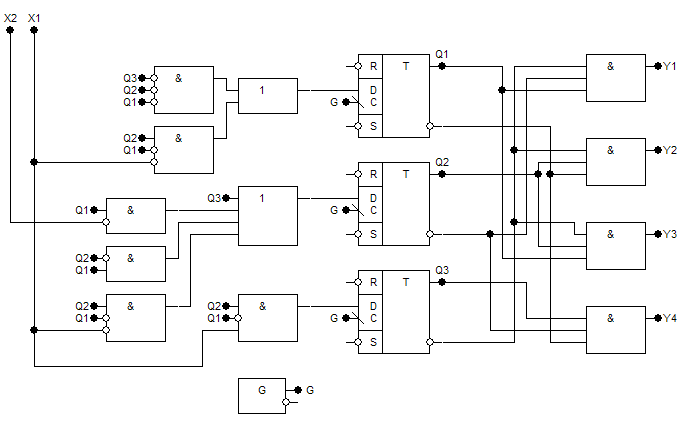
|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Пер. | | Ст. ст. | Нов. стан | Вх. сигн. | Вих. сигн. | Функції тригерів | | | |
| Q3Q2Q1 | Q3Q2Q1 | X1X2 | Y1Y2Y3Y4 | D3 | D2 | | D1 |
| Z1→Z2 | 000 | | 001 | -- | 0000 | 0 | | 0 | 1 |
| Z2→Z3 | 001 | | 010 | -- | 1000 | 0 | | 1 | 0 |
| Z3→Z4 | 010 | | 011 | 0- | 0100 | 0 | | 1 | 1 |
| Z3→Z5 | 010 | | 100 | 1- | 0100 | 1 | | 0 | 0 |
| Z4→Z1 | 011 | | 000 | -1 | 0010 | 0 | | 0 | 0 |
| Z4→Z3 | 011 | | 010 | -0 | 0010 | 0 | | 1 | 0 |
| Z5→Z1 | 100 | | 000 | -1 | 0001 | 0 | | 0 | 0 |
| Z5→Z3 | 100 | | 010 | -0 | 0001 | 0 | | 1 | 0 |

**3.5 Мінімізація функцій тригерів**



*Рисунок 3.3 – Мінімізація функцій тригерів*

**3.6 Функціональна схема автомата**



*Рисунок 3.4 - Функціональна схема*