НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

# "КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ"

#### ФАКУЛЬТЕТ ІНФОРМАТИКИ ТА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

### Кафедра обчислювальної техніки

## РОЗРАХУНКОВА РОБОТА

по курсу „Комп'ютерна логіка-2”

Виконав: Заколенко Роман Костянтинович

Група ІО-33 Факультет ІОТ,

Залікова книжка № 3308

Номер технічного завдання 1100 1110 1100

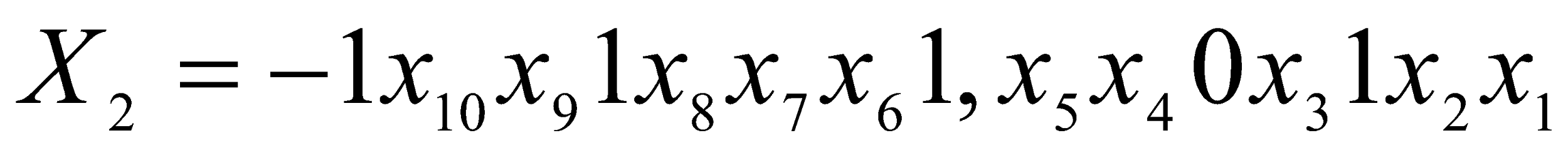
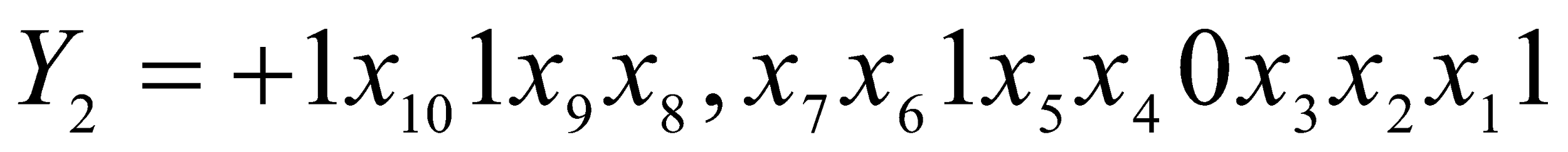
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

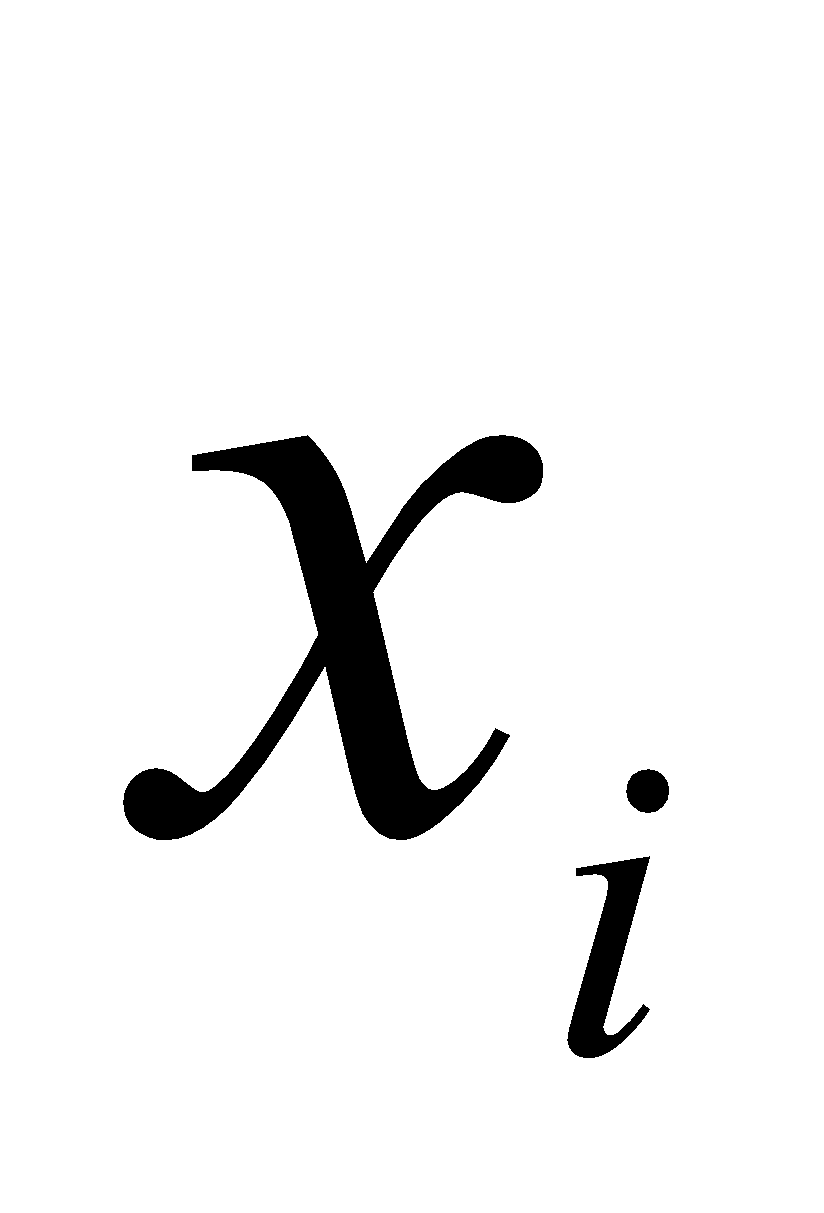
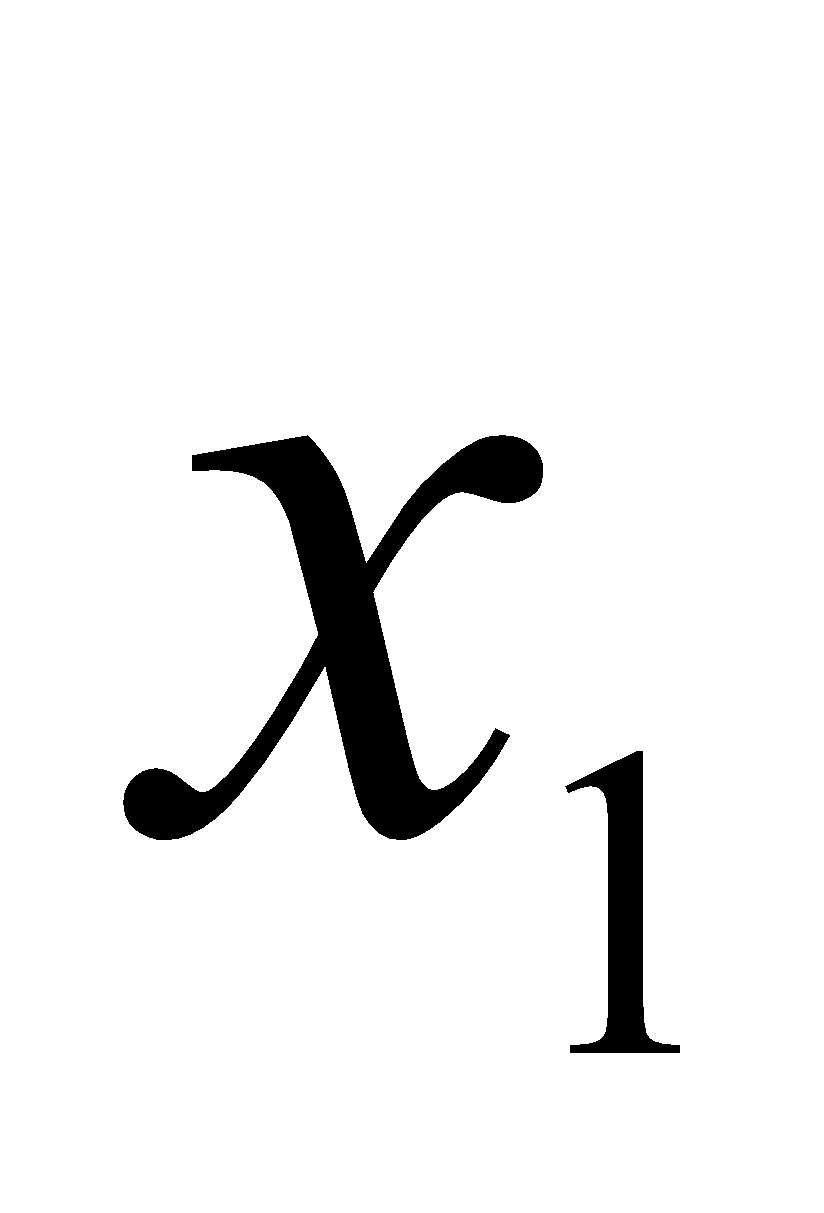
(підпис керівника)

Київ – 2014 р.

**Варіант завдання:**

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

 і ,

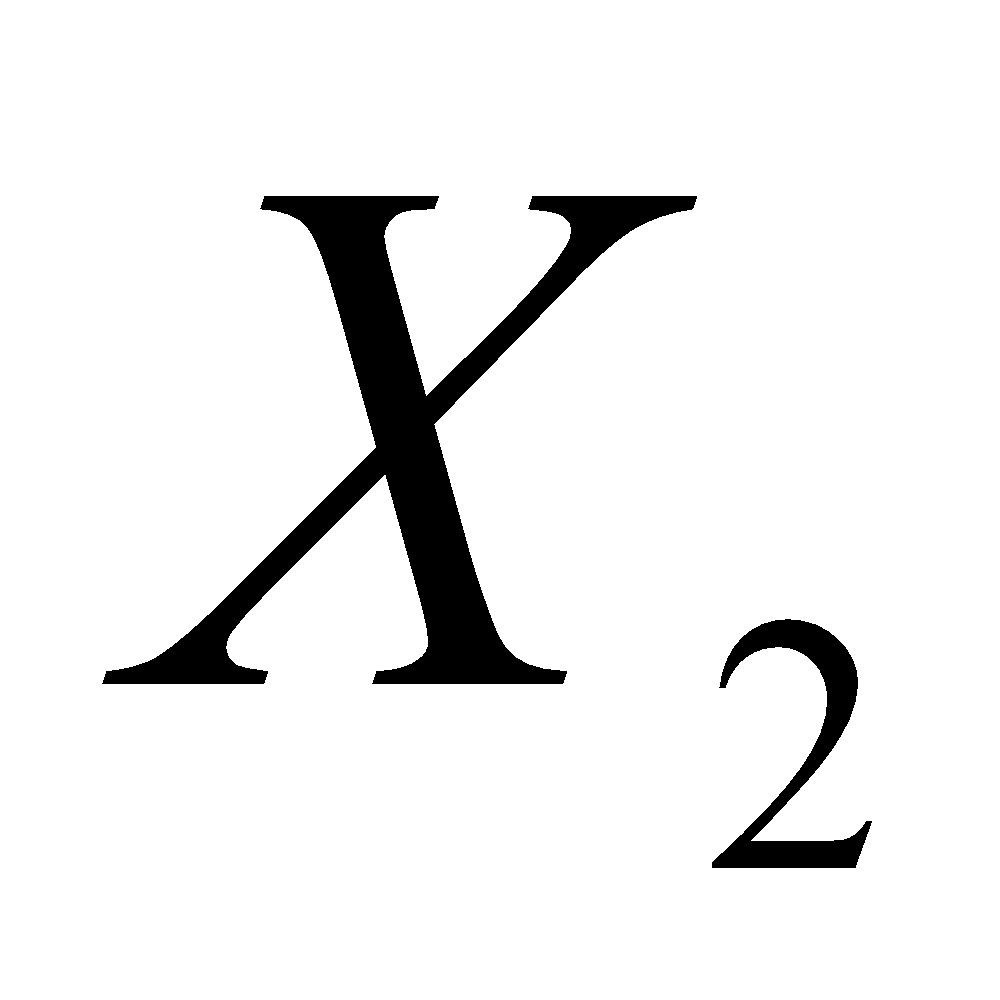
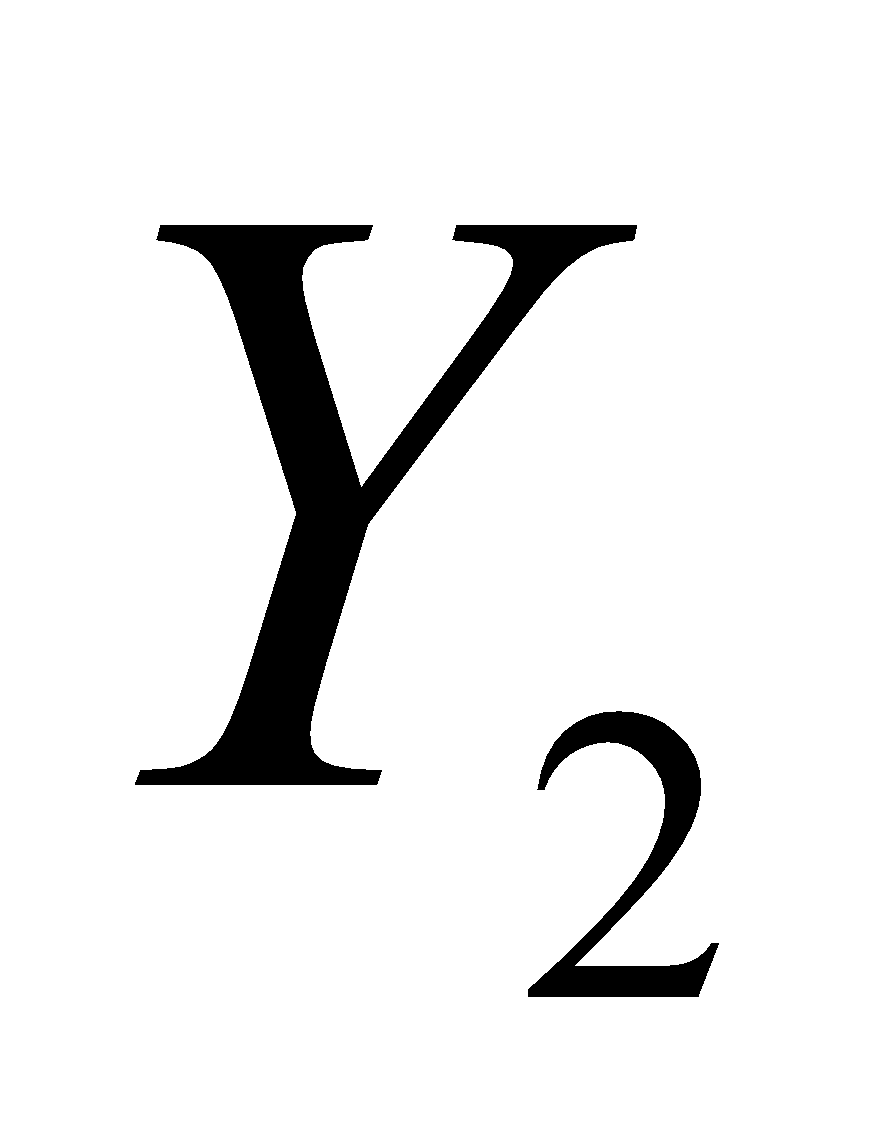
де  - двійкові цифри номера залікової книжки у двійковій системі числення ( - молодший розряд).

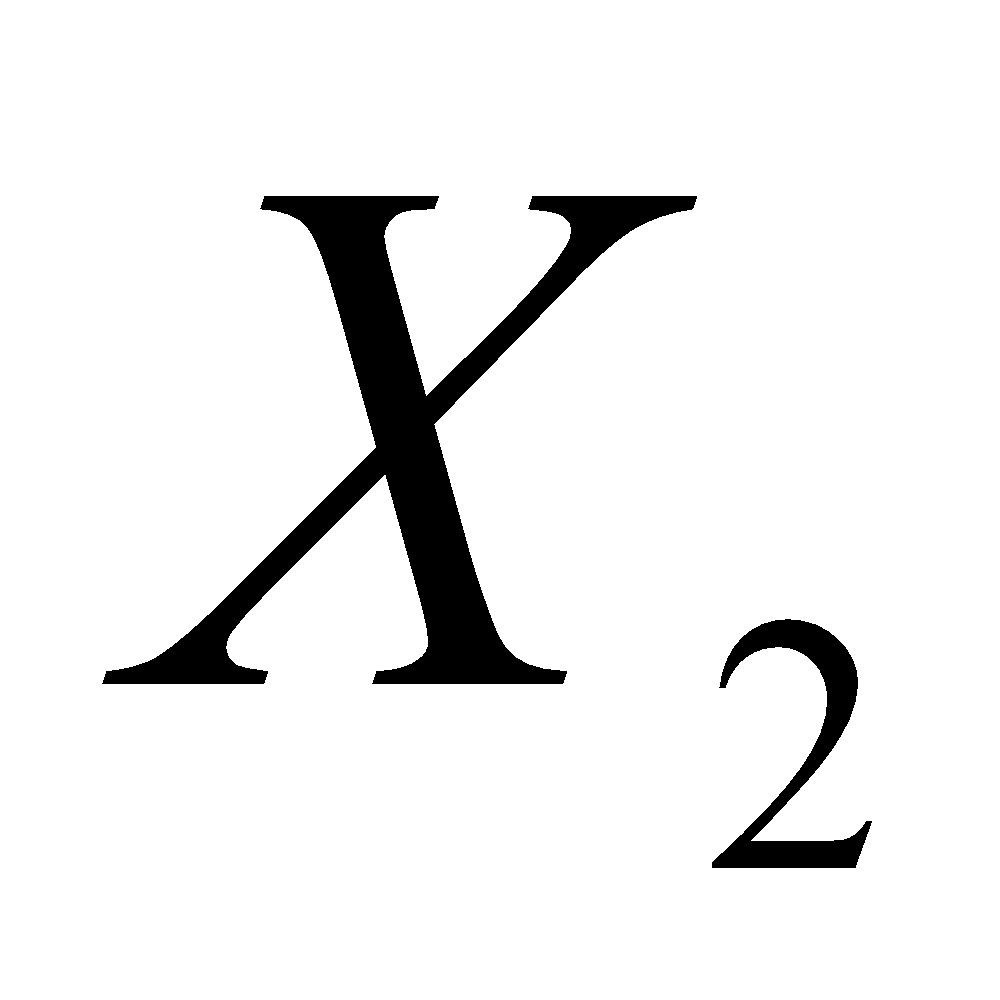
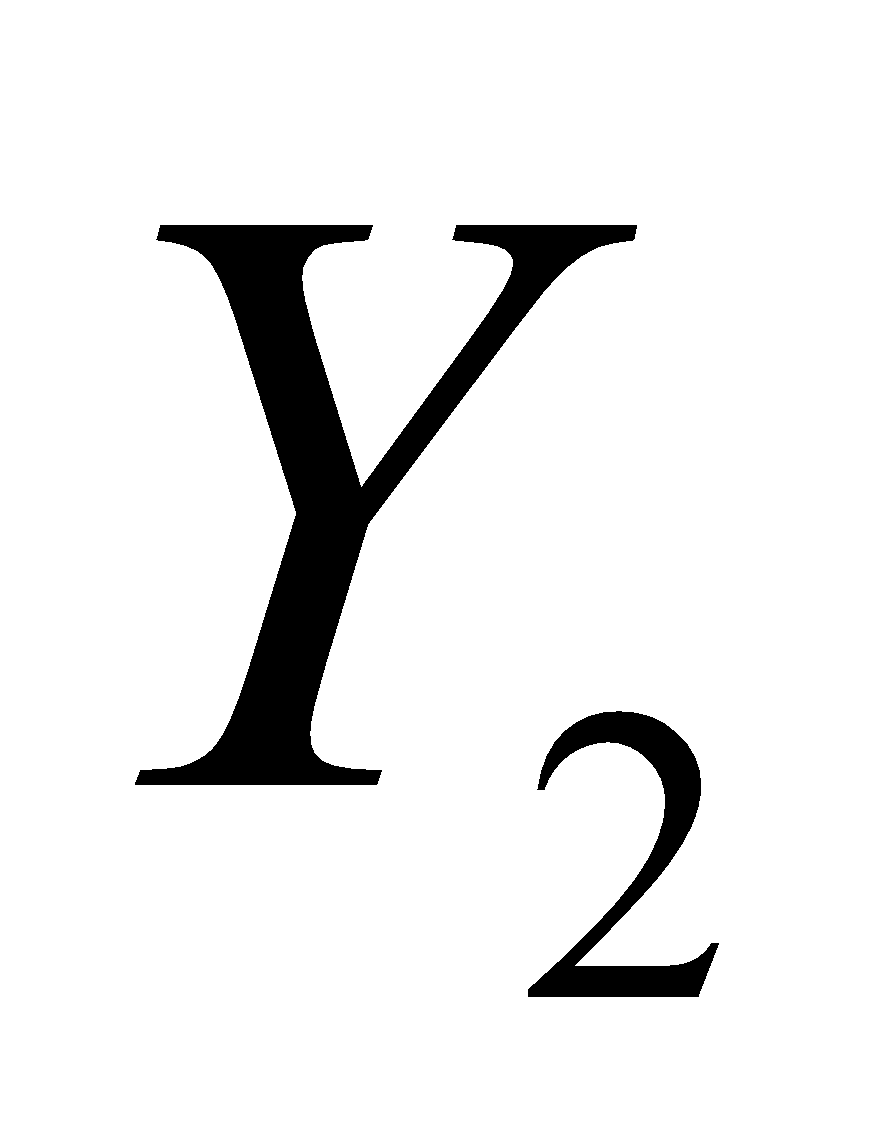
(3308)10 = (1100 1110 1100)2

X2= -10011111,0101100 ;

Y2= +10101,1110101001 ;

**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 обчислення кореня додатнього числа). Номери операцій (для п.3) відповідають порядку переліку (наприклад, 6 – ділення другим способом). Для обробки мантис кожної операції, подати:

2.1 теоретичне обґрунтування способу;

2.1 операційну схему;

2.2 змістовний мікроалгоритм;

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

2.4 функціональну схему з відображенням управляючих сигналів;

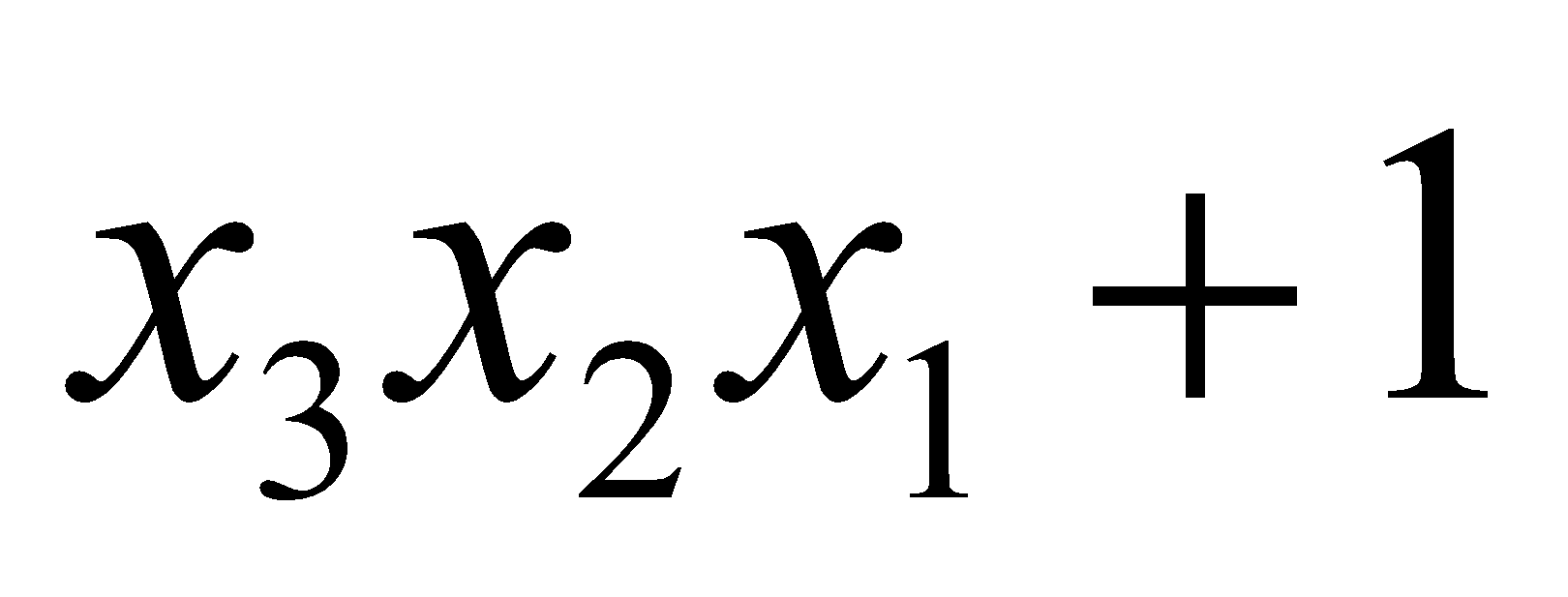
2.5 закодований мікроалгоритм (мікрооперації замінюються управл. сигналами);

2.6 граф управляючого автомата Мура з кодами вершин;

2.7 обробку порядків (показати у довільній формі);

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

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

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

**Завдання №1**

=1. 10011111,0101100;

=0. 10101,1110101001;

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

X2:

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

Y2:

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

Ey = 10000000 + 101 = 10000101

Y2:

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

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

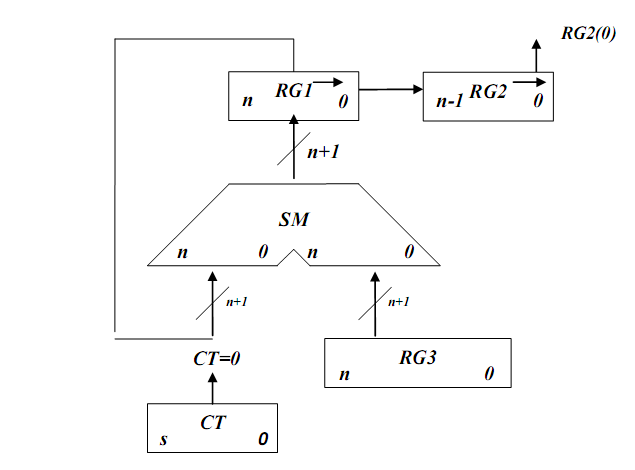
Множення мантис першим способом здійснюється з молодших розрядів множника, сума часткових добутків зсувається вправо, а множене залишається нерухомим. Тоді добуток двох чисел представляється у вигляді:

Z=YХ=+ Y…+ Y=

= ((..((0+Y)+ Y)+…+ Y) +…+ Y);

Z=;

**Операційна схема:**



*Рисунок 2.1.1- Операційна схема.*

**2.1.2 Змістовний мікроалгоритм:**



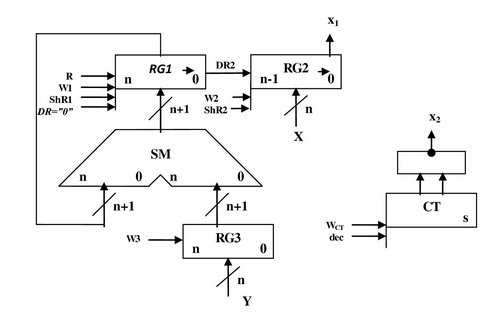
*Рисунок 2.1.2- Змістовний мікроалгоритм виконання операції множення першим способом.*

**2.1.3 Таблиця станів регістрів:**

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

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** | **RG1** | **RG2** | **RG3** | **CT** |
| **пс** | 0 | 100111110101100 | 101011110101001 | 1111 |
| **1** | 0 | 010011111010110 |  | 1110 |
| **2** | 0 | 001001111101011 |  | 1101 |
| **3** | 0010101111010100 | 100100111110101 |  | 1100 |
| **4** | +0101011110101001  =1000001101111101  0100000110111110 | 110010011111010 |  | 1011 |
| **5** | 0010000011011111 | 011001001111101 |  | 1010 |
| **6** | +0101011110101001  =0111100010001000  0011110001000100 | 001100100111110 |  | 1001 |
| **7** | 0001111000100010 | 000110010011111 |  | 1000 |
| **8** | +0101011110101001  =0111010111001011  0011101011100101 | 100011001001111 |  | 0111 |
| **9** | +0101011110101001  =1001001010001110  0100100101000111 | 010001100100111 |  | 0110 |
| **10** | +0101011110101001  =1010000011110000  0101000001111000 | 001000110010011 |  | 0101 |
| **11** | +0101011110101001  =1010100000100001  0101010000010000 | 100100011001001 |  | 0100 |
| **12** | +0101011110101001  =1010101110111001  0101010111011100 | 110010001100100 |  | 0011 |
| **13** | 0010101011101110 | 011001000110010 |  | 0010 |
| **14** | 0001010101110111 | 001100100011001 |  | 0001 |
| **15** | +0101011110101001  =0110110100100000  **0011011010010000** | **000110010001100** |  | 0000 |

**2.1.4 Функціональна схема:**



*Рисунок 2.1.3- Функціональна схема.*

**2.1.5 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 |



*Рисунок 2.1.4-Закодований мікроалгоритм.*

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



*Рисунок 2.1.5-Граф автомата Мура*

**2.1.7 Обробка порядків:**

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

=8; =5; =1310=11012

**2.1.8 Нормалізація результату:**

Отримали результат:011011010010000000110010001100

Знак мантиси: 1 0 = 1.

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

Порядок зменшуємо на 1:

Результат:11011010010000000110010001100;=12;

Запишемо нормалізований результат:

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

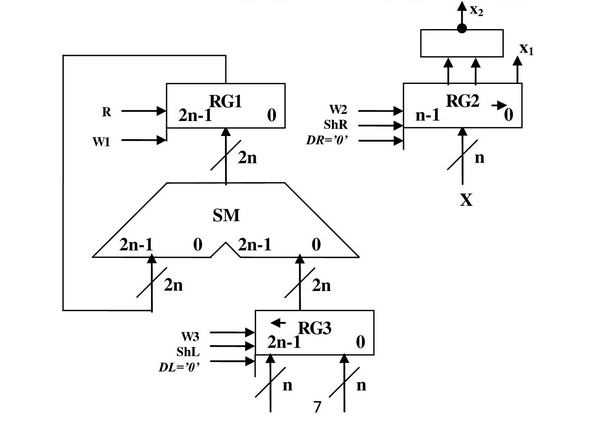
Множення мантис другим способом здійснюється з молодших розрядів, множене зсувається вліво, а сума часткових добутків залишається нерухомою.

Z=Y+ Y…+ Y;

Z=((0+ Y)+ Y)…+ Y;

Z=;

**Операційна схема:**



*Рисунок 2.2.1- Операційна схема*

**2.2.2 Змістовний мікроалгоритм:**



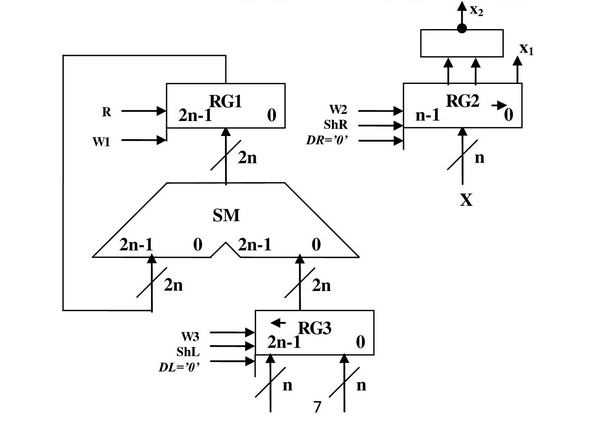
*Рисунок 2.2.2 - Змістовний мікроалгоритм*

**2.2.3 Таблиця станів регістрів:**

*Таблиця 2.2.1-Таблиця станів регістрів.*

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **RG1** | **RG3 ←** | **RG2 →** |
| **пс** | 0 | 000000000000000101011110101001 | 100111110101100 |
| **1** | 0 | 000000000000001010111101010010 | 010011111010110 |
| **2** | 0 | 000000000000010101111010100100 | 001001111101011 |
| **3** | 000000000000010101111010100100 | 000000000000101011110101001000 | 000100111110101 |
| **4** | +000000000000101011110101001000  =000000000001000001101111101100 | 000000000001010111101010010000 | 000010011111010 |
| **5** | 000000000001000001101111101100 | 000000000010101111010100100000 | 000001001111101 |
| **6** | +000000000010101111010100100000  =000 000 0 0 0 0 1 1 1 1 0 0 0 1 0 0 0 1 0 0 0 0 1 1 0 0 | 000000000101011110101001000000 | 000000100111110 |
| **7** | 000000000011110001000100001100 | 000000001010111101010010000000 | 000000010011111 |
| **8** | +000000001010111101010010000000  =000000001110101110010110001100 | 000000010101111010100100000000 | 000000001001111 |
| **9** | +000000010101111010100100000000  =000000100100101000111010001100 | 000000101011110101001000000000 | 000000000100111 |
| **10** | +000000101011110101001000000000  =000001010000011110000010001100 | 000001010111101010010000000000 | 000000000010011 |
| **11** | +000001010111101010010000000000  =000010101000001000010010001100 | 000010101111010100100000000000 | 000000000001001 |
| **12** | +000010101111010100100000000000  =000101010111011100110010001100 | 000101011110101001000000000000 | 000000000000100 |
| **13** | 000101010111011100110010001100 | 001010111101010010000000000000 | 000000000000010 |
| **14** | 000101010111011100110010001100 | 010101111010100100000000000000 | 000000000000001 |
| **15** | +010101111010100100000000000000  =**011011010010000000110010001100** | 101011110101001000000000000000 | 000000000000000 |

**2.2.4 Функціональна схема:**



*Рисунок 2.2.3- Функціональна схема.*

**2.2.5 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

X1

W1Z3



Z4

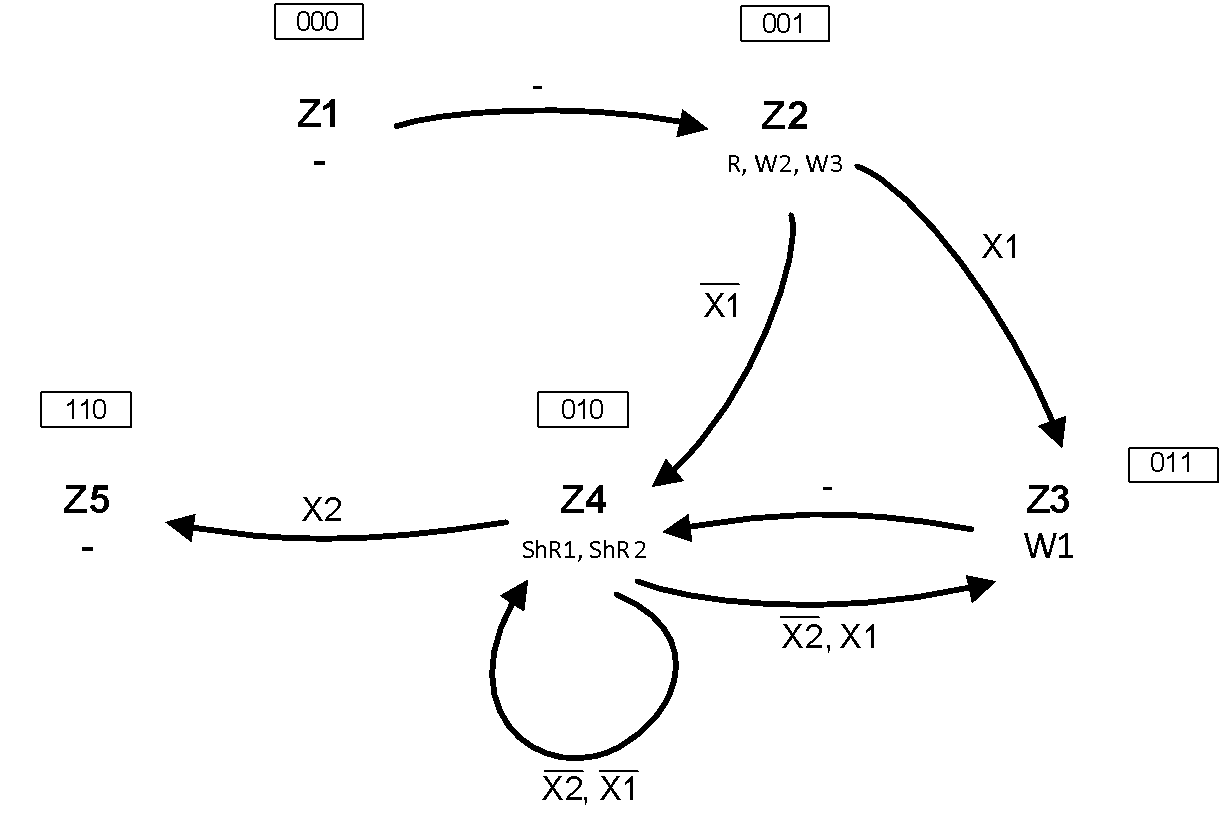
X2

1

Кінець Z5



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



*Рисунок 2.2.5 - Граф автомата Мура*

**2.2.7 Обробка порядків:**

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

=8; =5; =1310=11012

**2.2.8 Нормалізація результату:**

Отримали результат: 011011010010000000110010001100

Знак мантиси: 1 0 = 1.

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

Порядок зменшуємо на 1:

Результат: 11011010010000000110010001100; =12;

Запишемо нормалізований результат:

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

**2.3 Третій спосіб множення.**

**2.3.1 Теоретичне 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.

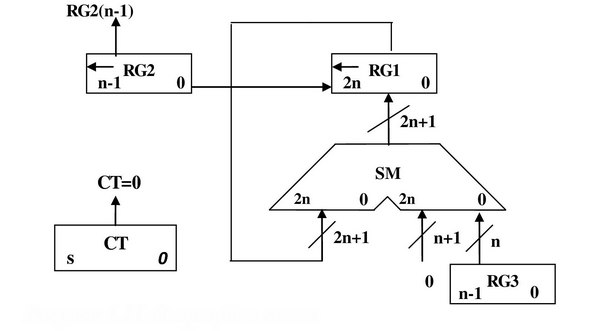
Множення мантис третім способом здійснюється зі старших розрядів множника, сума часткових добутків і множник зсуваються вліво, а множене нерухоме.

Z=Y+ Y…+ Y;

Z= Y+2(Y+2(Y…+2Y));

Z=;

**Операційна схема:**



*Рисунок 2.3.1 - Операційна схема*

**2.3.2 Змістовний мікроалгоритм:**



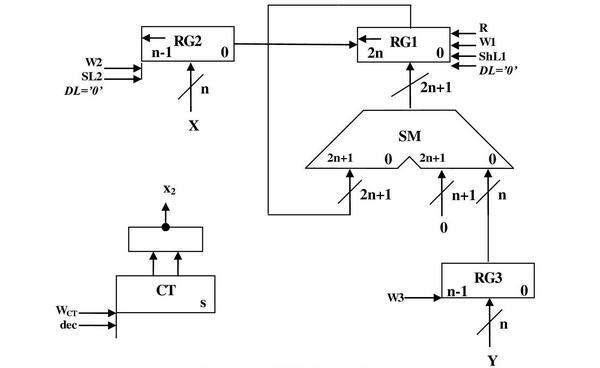


**2.3.3 Таблиця станів регістрів:**

*Таблиця 2.3.1- Таблиця станів регістрів*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** | **RG1 ←** | **RG2 ←** | **RG3** | **CT** |
| **пс** | 000000000000000000000000000000 | 100111110101100 | 101011110101001 | 1111 |
| **1** | 000000000000001010111101010010 | 001111101011000 |  | 1110 |
| **2** | 000000000000010101111010100100 | 011111010110000 |  | 1101 |
| **3** | 000000000000101011110101001000 | 111110101100000 |  | 1100 |
| **4** | +000000000000000101011110101001  =000000000000110001010011110001  000000000001100010100111100010 | 111101011000000 |  | 1011 |
| **5** | +000000000000000101011110101001  =000000000001101000000110001011  000000000011010000001100010110 | 111010110000000 |  | 1010 |
| **6** | +000000000000000101011110101001  =000000000011010101101010111111  000000000110101011010101111110 | 110101100000000 |  | 1001 |
| **7** | +000000000000000101011110101001  =000000000110110000110100100111  000000001101100001101001001110 | 101011000000000 |  | 1000 |
| **8** | +000000000000000101011110101001  =000000001101100111000111110111  000000011011001110001111101110 | 010110000000000 |  | 0111 |
| **9** | 000000110110011100011111011100 | 101100000000000 |  | 0110 |
| **10** | +000000000000000101011110101001  =000000110110100001111110000101  000001101101000011111100001010 | 011000000000000 |  | 0101 |
| **11** | 000011011010000111111000010100 | 110000000000000 |  | 0100 |
| **12** | +000000000000000101011110101001  =000011011010001101010110111101  000110110100011010101101111010 | 100000000000000 |  | 0011 |
| **13** | +000000000000000101011110101001  =000110110100100000001100100011  001101101001000000011001000110 | 000000000000000 |  | 0010 |
| **14** | 011011010010000000110010001100 | 000000000000000 |  | 0001 |
| **15** | 110110100100000001100100011000 | 000000000000000 |  | 0000 |

**2.3.4 Функціональна схема:**



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

**2.3.5 Закодований мікроалгоритм:**

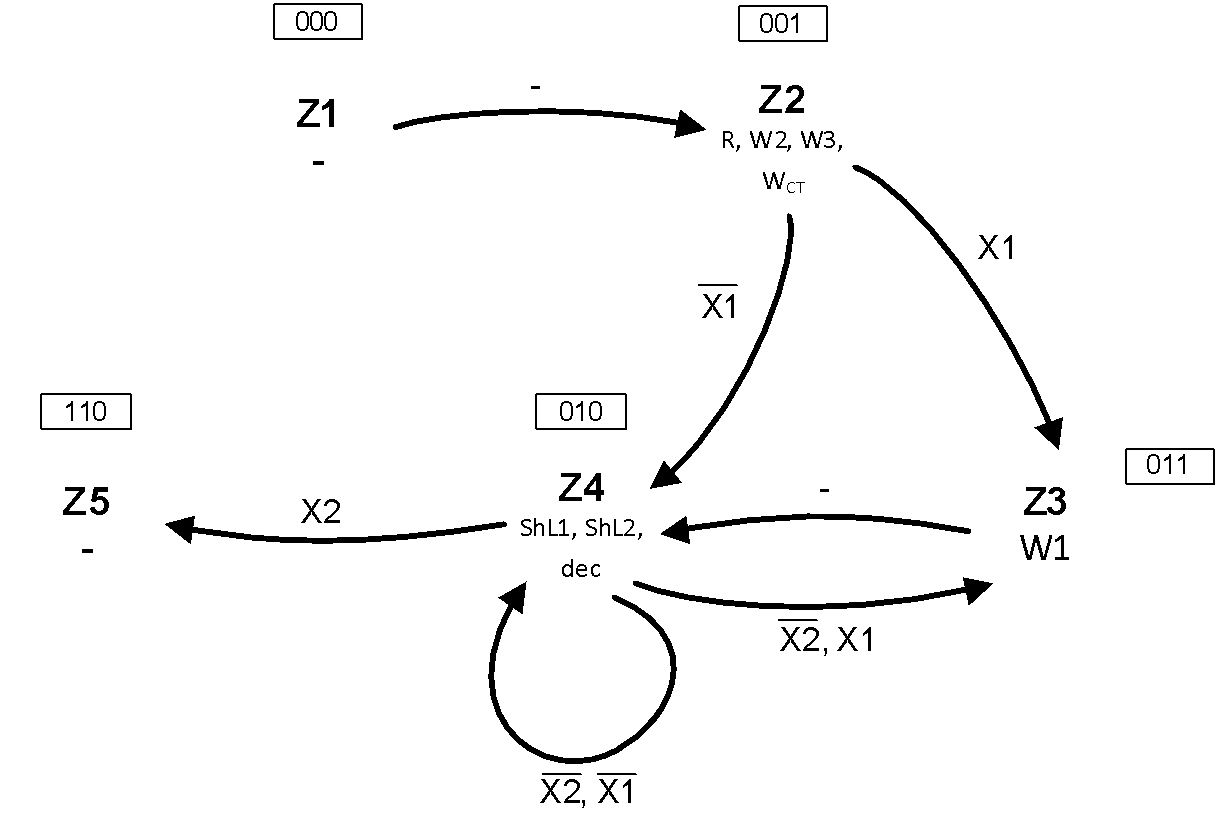
*Таблиця 2.3.2-Таблиця кодування операцій і логічних умов.*

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



*Рисунок 2.3.4-Закодований мікроалгоритм.*

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



*Рисунок 2.3.5 - Граф автомата Мура*

**2.3.7 Обробка порядків:**

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

=8; =5; =1310=11012

**2.3.9 Нормалізація результату:**

Отримали результат: 110110100100000001100100011000

Знак мантиси: 1 0 = 1.

Результат - нормалізований

Запишемо нормалізований результат:

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

**2.4 Четвертий спосіб множення.**

**2.4.1 Теоретичне 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.

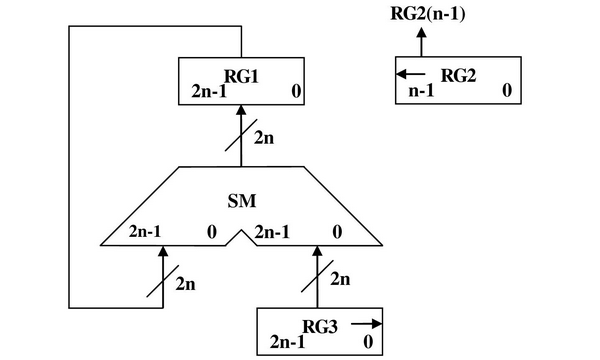
Множення здійснюється зі старших розрядів множника, сума часткових добутків залишається нерухомою,множене зсувається праворуч, множник ліворуч.

.

*.*

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

**Операційна схема:**



*Рисунок 2.4.1-Операційна схема*

**2.4.2 Змістовний мікроалгоритм:**



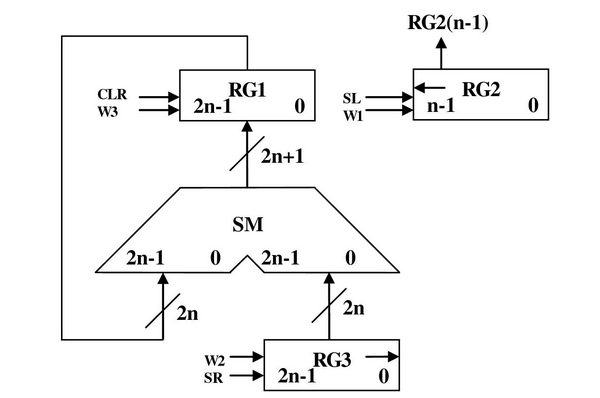


**2.4.3 Таблиця станів регістрів:**

*Таблиця 2.4.1- Таблиця станів регістрів*

|  |  |  |  |
| --- | --- | --- | --- |
| № | RG1 | RG3→ | RG2 ← |
| ПС | 000000000000000000000000000000 | 101011110101001000000000000000 | 100111110101100 |
| **1** | 010101111010100100000000000000 | 001010111101010010000000000000 | 001111101011000 |
| **2** | 010101111010100100000000000000 | 000101011110101001000000000000 | 011111010110000 |
| **3** | 010101111010100100000000000000 | 000010101111010100100000000000 | 111110101100000 |
| **4** | +000010101111010100100000000000  =011000101001111000100000000000 | 000001010111101010010000000000 | 111101011000000 |
| **5** | +000001010111101010010000000000  =011010000001100010110000000000 | 000000101011110101001000000000 | 111010110000000 |
| **6** | +000000101011110101001000000000  =011010101101010111111000000000 | 000000010101111010100100000000 | 110101100000000 |
| **7** | +000000010101111010100100000000  =011011000011010010011100000000 | 000000001010111101010010000000 | 101011000000000 |
| **8** | +000000001010111101010010000000  =011011001110001111101110000000 | 000000000101011110101001000000 | 010110000000000 |
| **9** | 011011001110001111101110000000 | 000000000010101111010100100000 | 101100000000000 |
| **10** | +000000000010101111010100100000  =011011010000111111000010100000 | 000000000001010111101010010000 | 011000000000000 |
| **11** | 011011010000111111000010100000 | 000000000000101011110101001000 | 110000000000000 |
| **12** | +000000000000101011110101001000  =011011010001101010110111101000 | 000000000000010101111010100100 | 100000000000000 |
| **13** | +000000000000010101111010100100  =011011010010000000110010001100 | 000000000000001010111101010010 | 000000000000000 |
| **14** | 011011010010000000110010001100 | 000000000000000101011110101001 | 000000000000000 |
| **15** | 011011010010000000110010001100 | 000000000000000010101111010100 | 000000000000000 |

**2.4.4Функціональна схема:**



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

**2.4.5 Закодований мікроалгоритм**

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

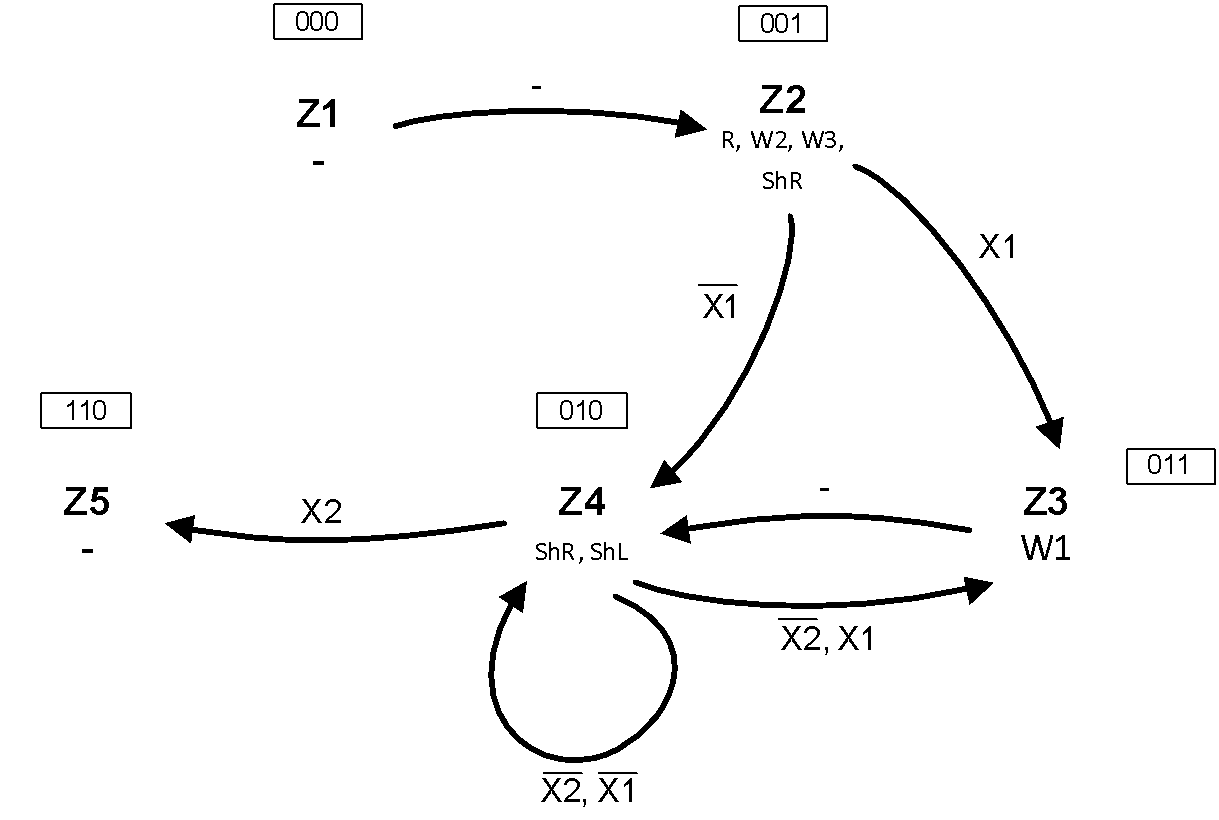
|  |  |  |  |
| --- | --- | --- | --- |
| Кодування мікрооперацій | | Кодування логічних умов | |
| МО | УС | ЛУ | Позначення |
| RG1:=0  RG2:=X  RG3:=Y RG1:=RG1+RG3  RG3:=0.r(RG3) RG2:=l(RG2).0 | R  W2  W3  W1  ShR  ShL | RG2[n-1]  RG2=0 | X1  X2 |





*Рисунок 2.4.4-Закодований мікроалгоритм.*

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



*Рисунок 2.4.5 - Граф автомата Мура*

**2.4.7 Обробка порядків:**

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

=8; =5; =1310=11012

**2.4.8 Нормалізація результату:**

Отримали результат: 011011010010000000110010001100

Знак мантиси: 1 0 = 1.

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

Порядок понижаємо на 1:

11011010010000000110010001100 ; =12;

Запишемо нормалізований результат:

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

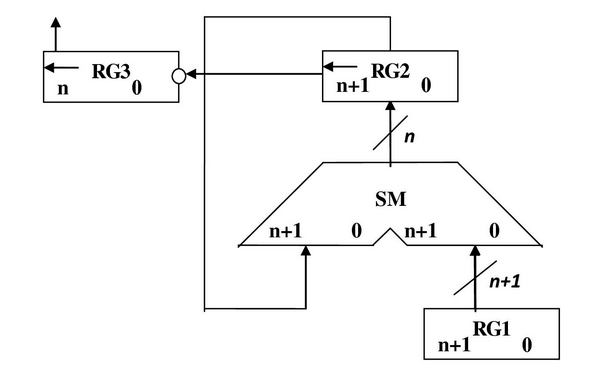
**2.5. Першиий спосіб ділення.**

**2.5.1 Теоритичне обґрунтування першого способу ділення:**

Нехай ділене Х і дільник Y є n-розрядними правильними дробами, поданими в прямому коді. В цьому випадку знаковій 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.

**Операційна схема:**



*Рисунок 2.5.1-Операційна схема*

**2.5.2 Змістовний мікроалгоритм:**

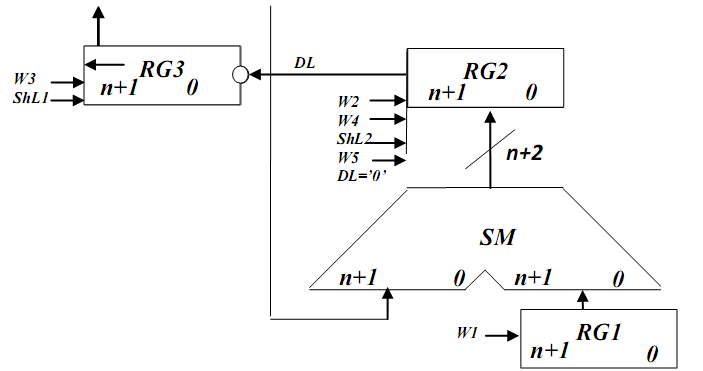


*Рисунок2.5.2-Змістовний мікроалгоритм*

**2.5.3 Таблиця станів регістрів:**

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **RG1(Z)** | **RG2(X)** | **RG3(Y)** |
| **пс** | 000000000000000 | 00100111110101100 | 101011110101001 |
| **1** | 0000000000000001 | 01001111101011000  +11010100001010111  =00100011110101111 |  |
| **2** | 0000000000000011 | 01000111101011110  +11010100001010111  =00011011110110101 |  |
| **3** | 0000000000000111 | 00110111101101010  +11010100001010111  =00001011111000001 |  |
| **4** | 0000000000001111 | 00010111110000010  +11010100001010111  =11101011111011001 |  |
| **5** | 0000000000011110 | 11010111110110010  +00101011110101001  =00000011101011011 |  |
| **6** | 0000000000111101 | 00000111010110110  +11010100001010111  =11011011100001101 |  |
| **7** | 0000000001111010 | 10110111000011010  +00101011110101001  =11100010111000011 |  |
| **8** | 0000000011110100 | 11000101110000110  +00101011110101001  =11110001100101111 |  |
| **9** | 0000000111101000 | 11100011001011110  +00101011110101001  =00001111000000111 |  |
| **10** | 0000001111010001 | 00011110000001110  +11010100001010111  =11110010001100101 |  |
| **11** | 0000011110100010 | 11100100011001010  +00101011110101001  =00010000001110011 |  |
| **12** | 0000111101000101 | 00100000011100110  +11010100001010111  =11110100100111101 |  |
| **13** | 0001111010001010 | 11101001001111010  +00101011110101001  =00010101000100011 |  |
| **14** | 0011110100010101 | 00101010001000110  +11010100001010111  =11111110010011101 |  |
| **15** | 0111101000101010 | 11111100100111010  +00101011110101001  =00101000011100011 |  |
| **16** | **1111010001010101** | 01010000111000110  +11010100001010111  =00100101000011101 |  |

**2.5.4 Функціональна схема:**



*Рисунок 2.5.3 – Функціональна схема*

**2.5.5 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 |

















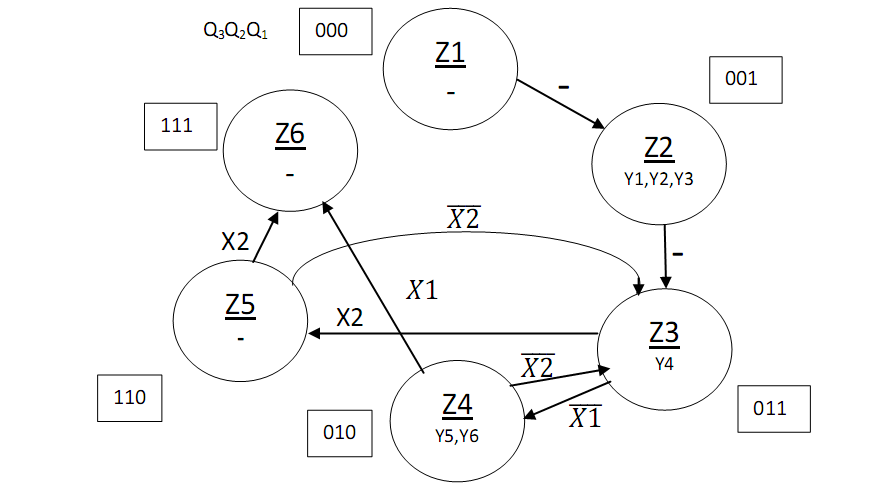






*Рисунок 2.5.4-Закодований мікроалгоритм.*

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



*Рисунок 2.5.5 - Граф управляючого автомата.*

**2.5.8 Обробка порядків:**

Порядок частки буде дорівнювати :

=8; =5; =3;

**2.5.8 Нормалізація результату:**

Отримали результат: 1111010001010101

Знак мантиси: 1 0 = 1.

Нормалізація мантиси не потрібна.

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

**2.6. Другий спосіб ділення.**

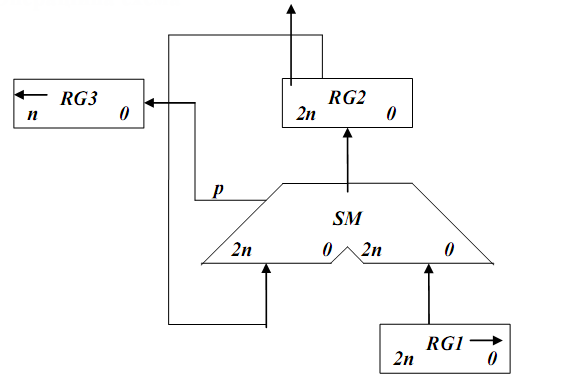
**2.6.1 Теоретичне обґрунтування другого способу ділення:**

Нехай ділене Хі дільник Y є n-розрядними правильними дробами, поданими в прямому коді. В цьому випадку знаковій 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 примноженні з нерухомою сумою часткових добутків можна водночас виконувати підсумування і віднімання, зсув в регістрах Y,Z. Тобто1 цикл може складатися з 1 такту,це дає

прискорення відносно1-госпособу.

**Операційна схема:**



*Рисунок2.6.1-Операційна схема*

**2.6.2 Змістовний мікроалгоритм**























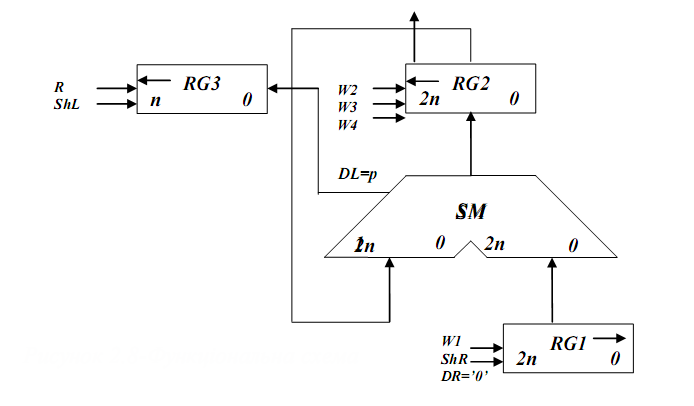
*Рисунок 2.6.2-Змістовний мікроалгоритм*

**2.6.3 Таблиця станів регістрів**

*Таблиця 2.6.1- Таблиця станів регістрів*

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **RG3(Z)** | **RG2(X)** | **RG1(Y)** |
| **пс** | 0000000000000001 | 010011111010110000000000000000 | 001010111101010010000000000000 |
| **1** | 0000000000000011 | 010011111010110000000000000000  +110101000010101110000000000000  =001000111101011110000000000000 | 000101011110101001000000000000 |
| **2** | 0000000000000111 | 001000111101011110000000000000  +111010100001010111000000000000  =000011011110110101000000000000 | 000010101111010100100000000000 |
| **3** | 0000000000001111 | 000011011110110101000000000000  +111101010000101011100000000000  =000000101111100000100000000000 | 000001010111101010010000000000 |
| **4** | 0000000000011110 | 000000101111100000100000000000  +111110101000010101110000000000  =111111010111110110010000000000 | 000000101011110101001000000000 |
| **5** | 0000000000111101 | 111111010111110110010000000000  +000000101011110101001000000000  =000000000011101011011000000000 | 000000010101111010100100000000 |
| **6** | 0000000001111010 | 000000000011101011011000000000  +111111101010000101011100000000  =111111101101110000110100000000 | 000000001010111101010010000000 |
| **7** | 0000000011110100 | 111111101101110000110100000000  +000000001010111101010010000000  =111111111000101110000110000000 | 000000000101011110101001000000 |
| **8** | 0000000111101000 | 111111111000101110000110000000  +000000000101011110101001000000  =111111111110001100101111000000 | 000000000010101111010100100000 |
| **9** | 0000001111010001 | 111111111110001100101111000000  +000000000010101111010100100000  =000000000000111100000011100000 | 000000000001010111101010010000 |
| **10** | 0000011110100010 | 000000000000111100000011100000  +111111111110101000010101110000  =111111111111100100011001010000 | 000000000000101011110101001000 |
| **11** | 0000111101000101 | 111111111111100100011001010000  +000000000000101011110101001000  =000000000000010000001110011000 | 000000000000010101111010100100 |
| **12** | 0001111010001010 | 000000000000010000001110011000  +111111111111101010000101011100  =111111111111111010010011110100 | 000000000000001010111101010010 |
| **13** | 0011110100010101 | 111111111111111010010011110100  +000000000000001010111101010010  =000000000000000101010001000110 | 000000000000000101011110101001 |
| **14** | 0111101000101010 | 000000000000000101010001000110  +111111111111111010100001010111  =111111111111111111110010011101 | 000000000000000010101111010100 |
| **15** | **1111010001010101** | 111111111111111111110010011101  +000000000000000010101111010100  =000000000000000010100001110001 | 000000000000000001010111101010 |

**2.6.4 Функціональна схема з відображенням управляючих сигналів**



*Рисунок2.6.3-Функціональна схем*

**2.6.5 Закодований мікроалгоритм**

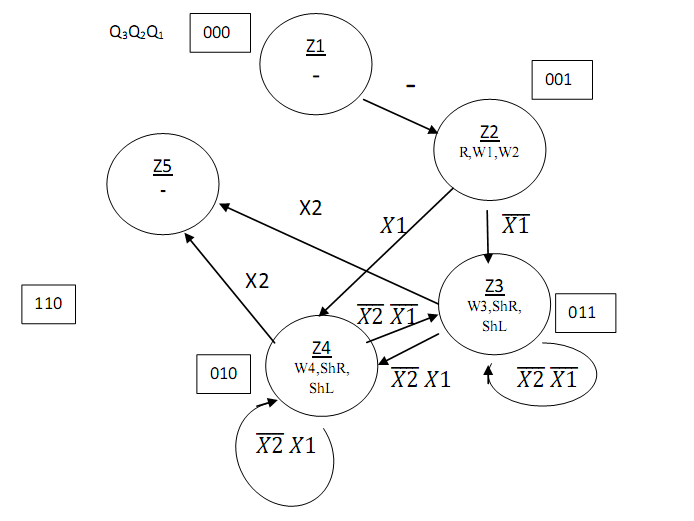
*Таблиця2.6.2- Таблиця кодування мікрооперацій*

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



*Рисунок2.6.4- Закодованиймікроалгоритм*

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



*Рисунок2.6.5- Граф автомата Мура*

**2.6.7 Обробка порядків:**

Порядок частки буде дорівнювати:

В моєму випадку =8; =5; =3;

**2.6.8 Нормалізація результату:**

Отримали результат: 1111010001010101

Знак мантиси: 1 0 = 1.

Нормалізація мантиси не потрібна.

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

**2.7.Операція додавання чисел.**

**2.7.1Теоретичне 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, 101011110101001 | 11 | Початковий стан |
| 0,010101111010100 | 10 | My= 0.r(My); ∆:=∆-1 |
| 0, 001010111101010 | 01 | My= 0.r(My); ∆:=∆-1 |
| 0,000101011110101 | 00 | My= 0.r(My); ∆:=∆-1 |

3. Додавання мантис у модифікованому ДК.

MX мдк = 11. 011000001010100

MY мдк = 00.000101011110101

*Таблиця 2.7.2-Додавання мантис*

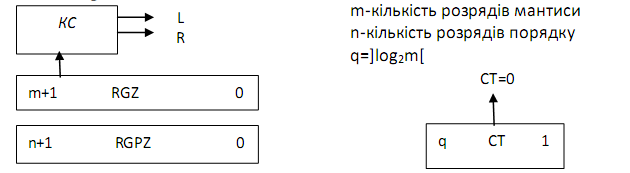
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| MX | 1 | 1, | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 |
| MY | 0 | 0, | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 |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

MZ пк = 1.100010010110111

4.Нормалізація результату

Для даного результату додавання нормалізація не потрібна.

**Операційна схема :**



*Рисунок2.7.1-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.

**2.7.2 Змістовний алгоритм**



*Рисунок2.7.2-Змістовний мікроалгоритм*

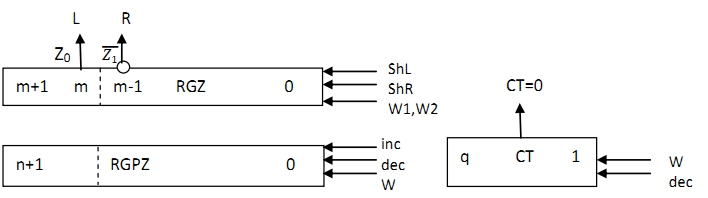
**2.7.3Таблиця станів регістрів**

**1) Додавання**

*Таблиця2.7.5- Таблиця станів регістрів*

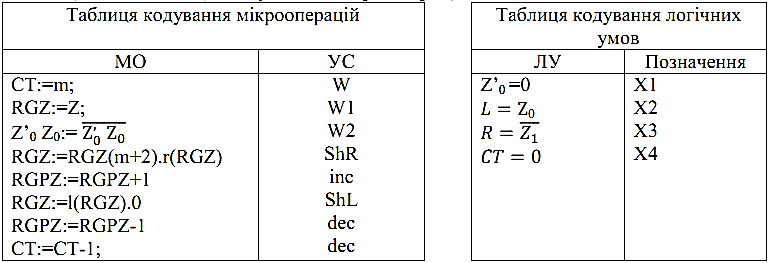
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№**  **такту** | **RGPZ** | **RGZ** | **ЛПН(L)** | **ППН(R)** | **СT** | **Мікрооперація** |
| **ПС** | 001000 | 00, 101011110101001 | 0 | 1 | 100 |  |
| **1** | 000111 | 00, 010111101010010  00, 101111010100100 | 0 | 0 | 011 |  |

**2.7.4 Функціональна схема з відображенням керуючих сигналів**

*****Рисунок 2.7.3 – Функціональна схема*

**2.7.5 Закодований мікроалгоритм**

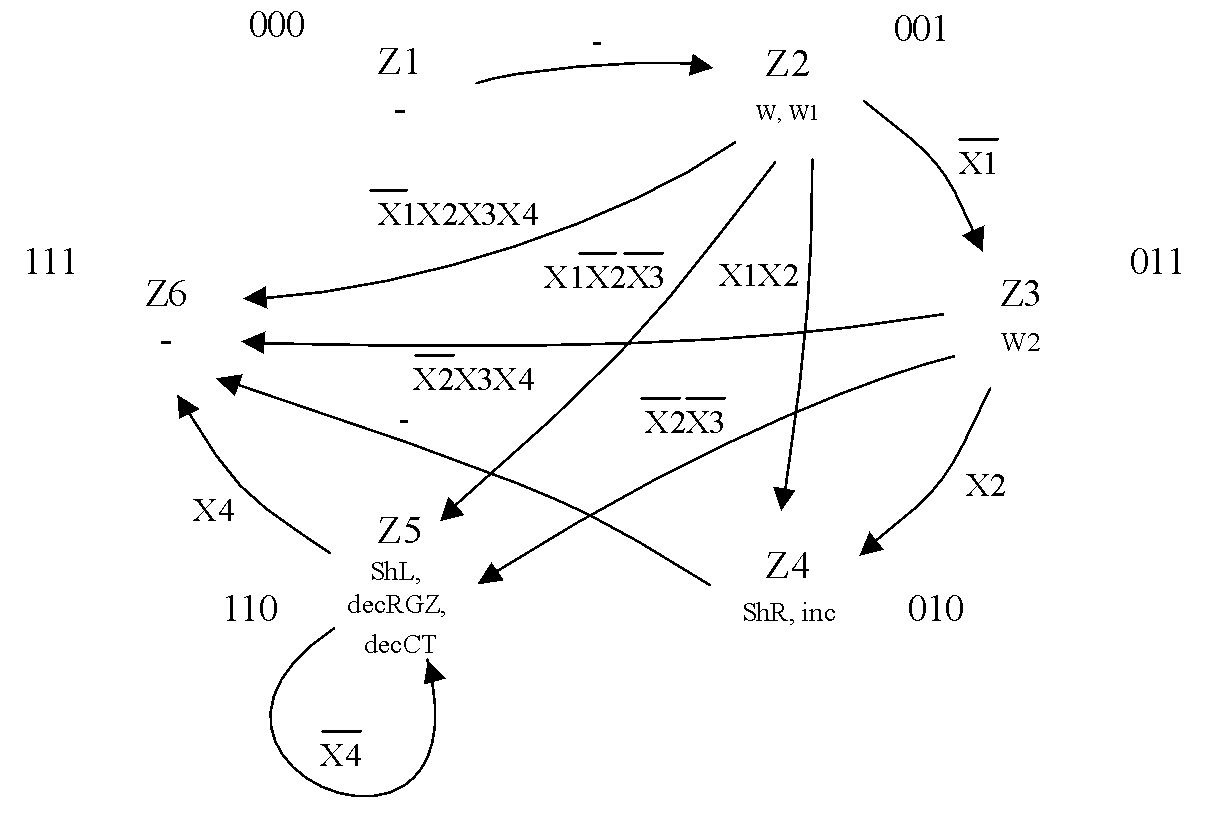
*Таблиця 2.7.7– Таблиця кодування*

****



*Рисунок 2.7.4 – Закодований мікроалгоритм*

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



*Рисунок 2.7.5 – Граф автомата Мура*

**2.7.7 Обробка порядків**

PX+Y=810=10002

**2.7.8 Форма запису результату з плаваючою комою**

Результат додавання

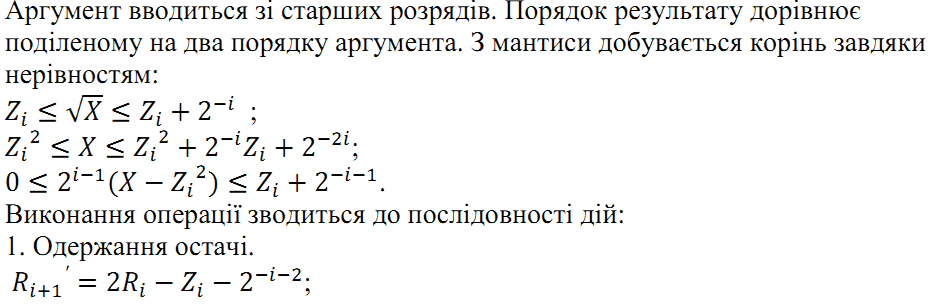
MZ пк = 1.100010010110111

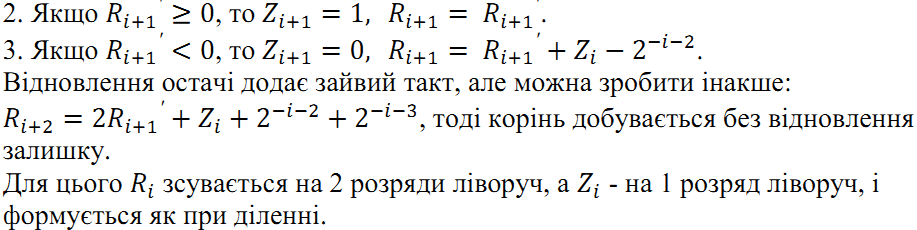
Pz=810=10002

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

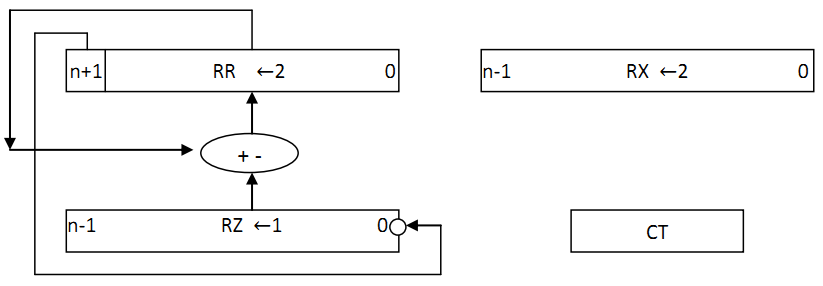
**2.8.Операція добування кореня**

**2.8.1Теоритичне обґрунтування операції обчислення квадратного кореня**



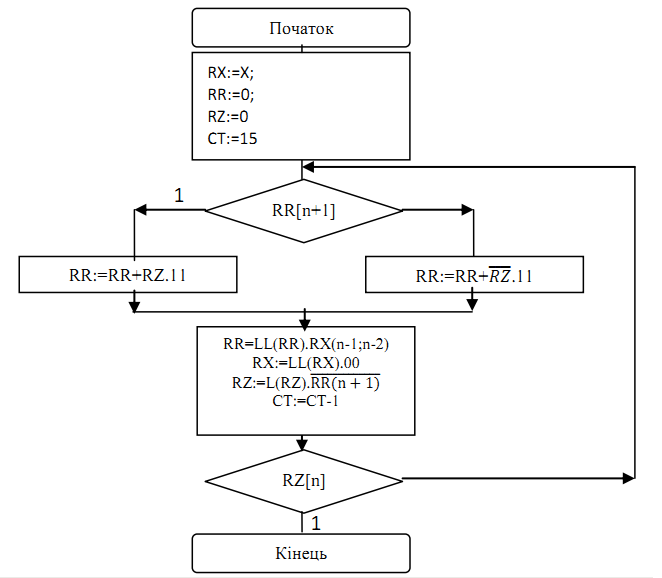


**Операційна схема операції обчислення квадратного кореня**



*Рисунок 2.8.1 –Операційна схема*

**2.8.2 Змістовний мікроалгоритм**



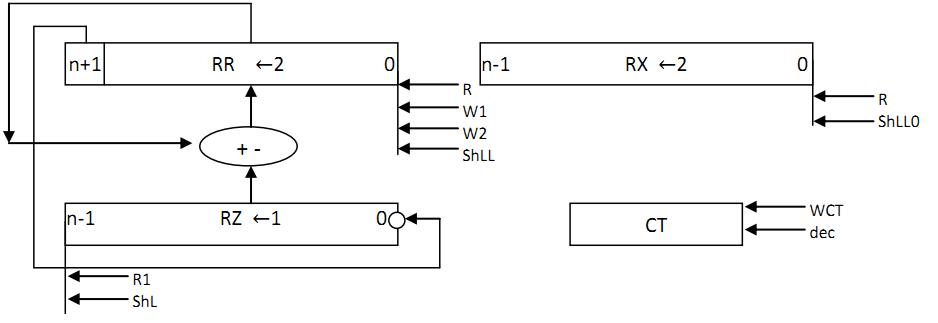
*Рисунок 2.8.2 – Змістовний мікроалгоритм*

**2.8.3 Таблиця станів регістрів**

*Таблиця 2.8.1 – Таблиця станів регістрів*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | RZ | RR | RX | СТ |
| **пс** | 000000000000000 | 00000000000000000  00000000000000010 | 100111010101110 | 1111 |
| **пз** |
| **1** | 000000000000001 | +11111111111111111  =00000000000000001  00000000000000101 | 010011111010110 | 1110 |
| **2** | 000000000000011 | +11111111111111011  =00000000000000000  00000000000000011 | 001001111101011 | 1101 |
| **3** | 000000000000110 | +11111111111110011  =11111111111110110  11111111111011011 | 000100111110101 | 1100 |
| **4** | 000000000001100 | +00000000000011011  =11111111111110110  11111111111011001 | 000010011111010 | 1011 |
| **5** | 000000000011001 | +00000000000110011  =00000000000001100  00000000000110001 | 000001001111101 | 1010 |
| **6** | 000000000110010 | +11111111110011011  =11111111111001100  11111111100110010 | 000000100111110 | 1001 |
| **7** | 000000001100100 | +00000000011001011  =11111111111111101  11111111111110100 | 000000010011111 | 1000 |
| **8** | 000000011001001 | +00000000110010011  =00000000110000111  00000011000011100 | 000000001001111 | 0111 |
| **9** | 000000110010011 | +11111110011011011  =00000001011110111  00000101111011100 | 000000000100111 | 0110 |
| **10** | 000001100100111 | +11111100110110011  =00000010110001111  00001011000111100 | 000000000010011 | 0101 |
| **11** | 000011001001111 | +11111001101100011  =00000100110011111  00010011001111100 | 000000000001001 | 0100 |
| **12** | 000110010011111 | 00010011001111100  +11110011011000011  =00000110100111111  00011010011111100 | 000000000000100 | 0011 |
| **13** | 001100100111111 | +11100110110000011  =00000001001111111  00000100111111100 | 000000000000010 | 0010 |
| **14** | 011001001111110 | +11001101100000011  =11010010011111111  01001001111111100 | 000000000000001 | 0001 |
| **15** | **110010011111100** | +10011011000000111  =11100101000000011  10010100000001100 | 000000000000000 | 0000 |

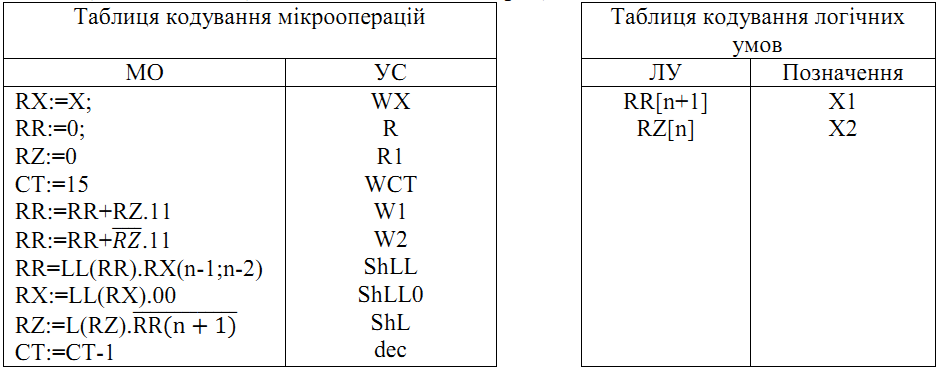
**2.8.4 Функціональна схема операції обчислення квадратного кореня**

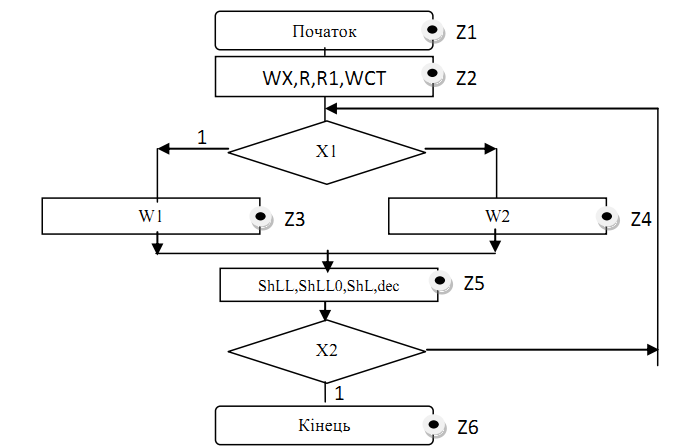


*Рисунок 2.8.3 – Функціональна схема*

**2.8.5 Закодований мікроалгоритм**

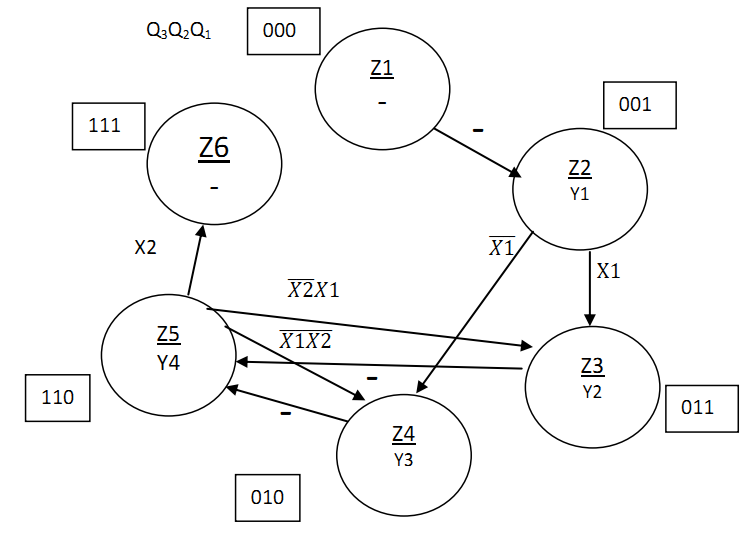
*Таблиця 2.8.2 – Таблиця кодування*





*Рисунок 2.8.4 – Закодований мікроалгоритм*

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



*Рисунок 2.8.5 – Граф управляючого автомата Мура*

**2.8.7 Обробка порядків**

=4;

**2.8.8 Запис результату**

Отримали результат 110010011111100;

Результат нормалізований, готовий до запису у мантису:

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

**Завдання №3**

Варіант завдання: x3x2x1+1 =1012= 510.

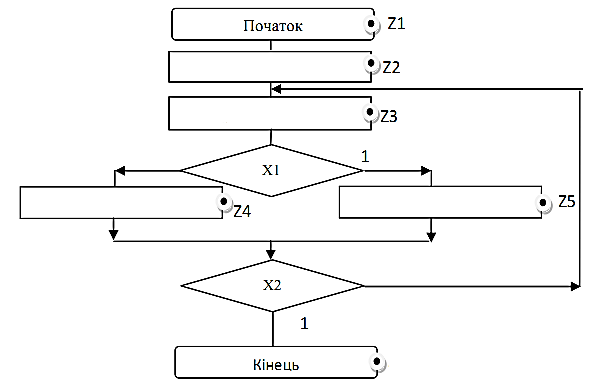
Синтез управляючого автомату Мура на D-тригерах для операції ділення першим способом

**3.1 Таблиця кодування сигналів**

*Таблиця 3.1 – Таблиця кодування сигналів*

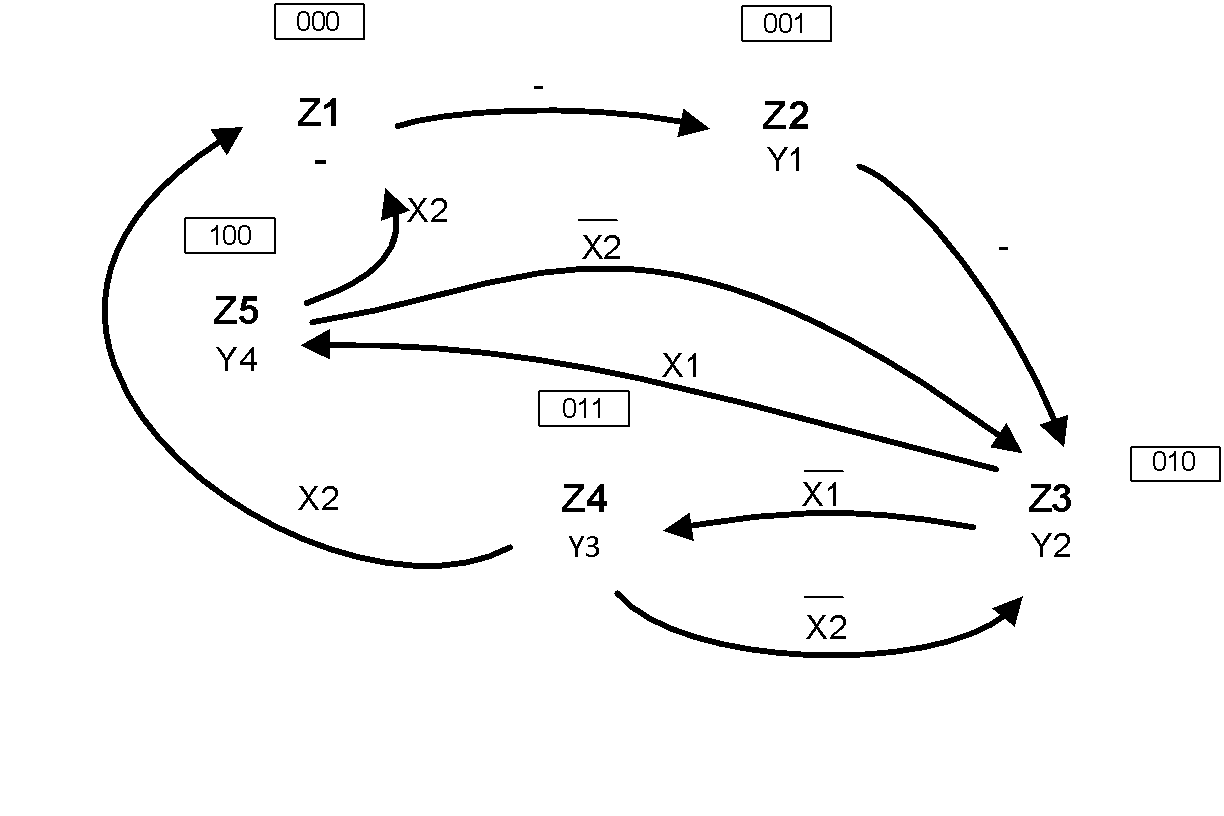
|  |  |
| --- | --- |
| W3, W2, W1 | Y1 |
| ShL1, ShL2 | Y2 |
| W4 | Y3 |
| W5 | Y4 |

**3.2 Мікроалгоритм в термінах управляючого автомата**

****

*Рисунок 3.1 – Закодований мікроалгоритм*

**3.3 Граф автомата**



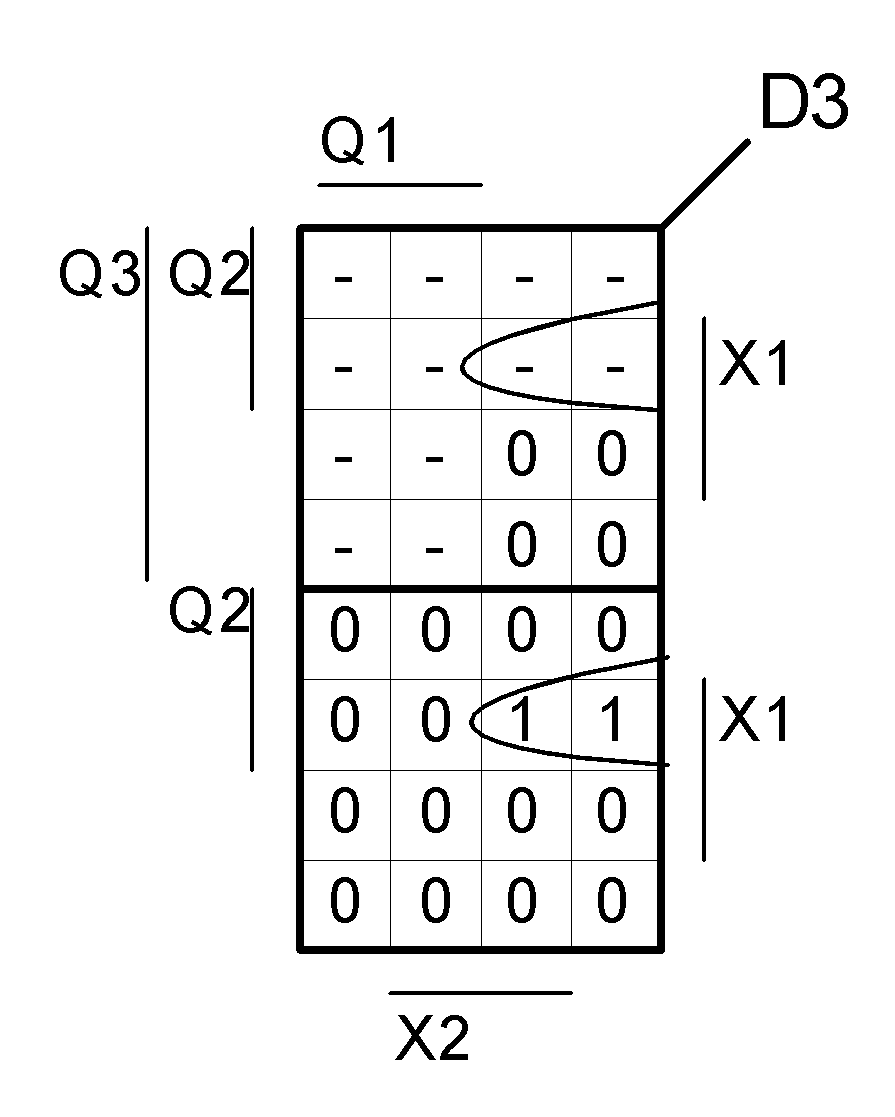
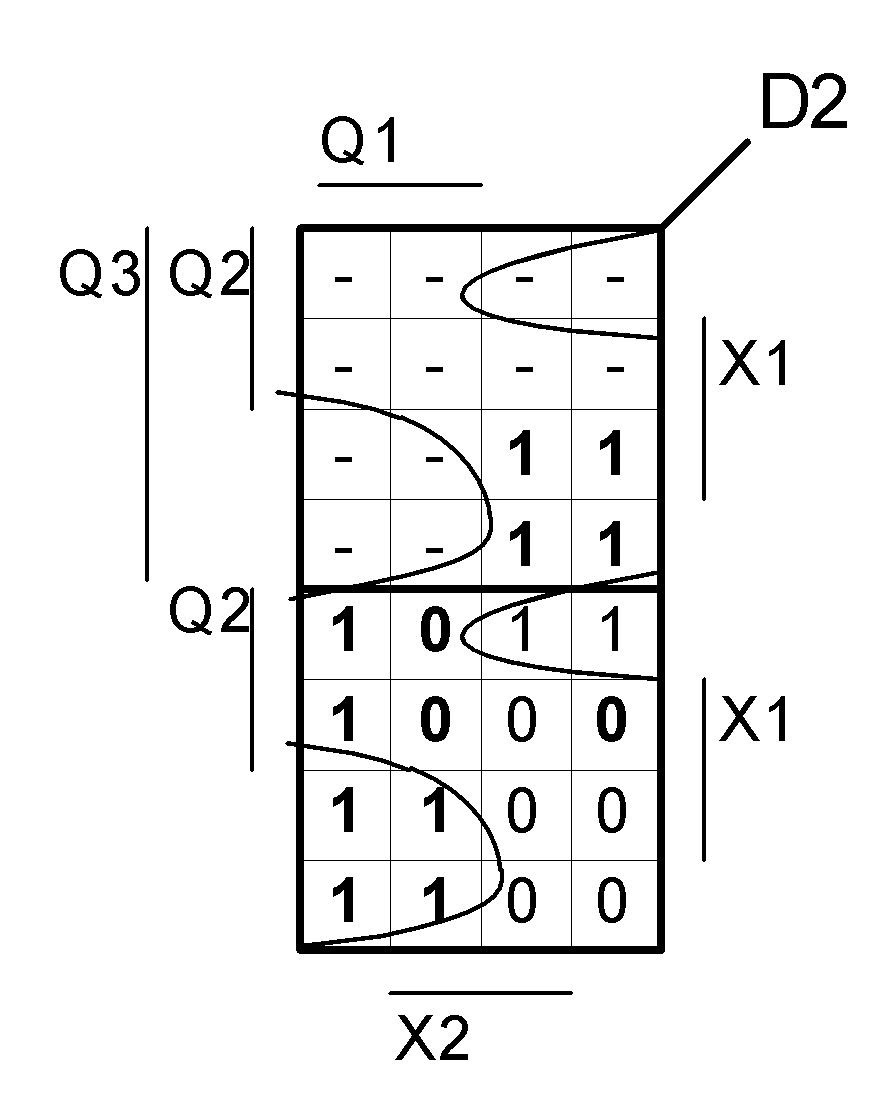
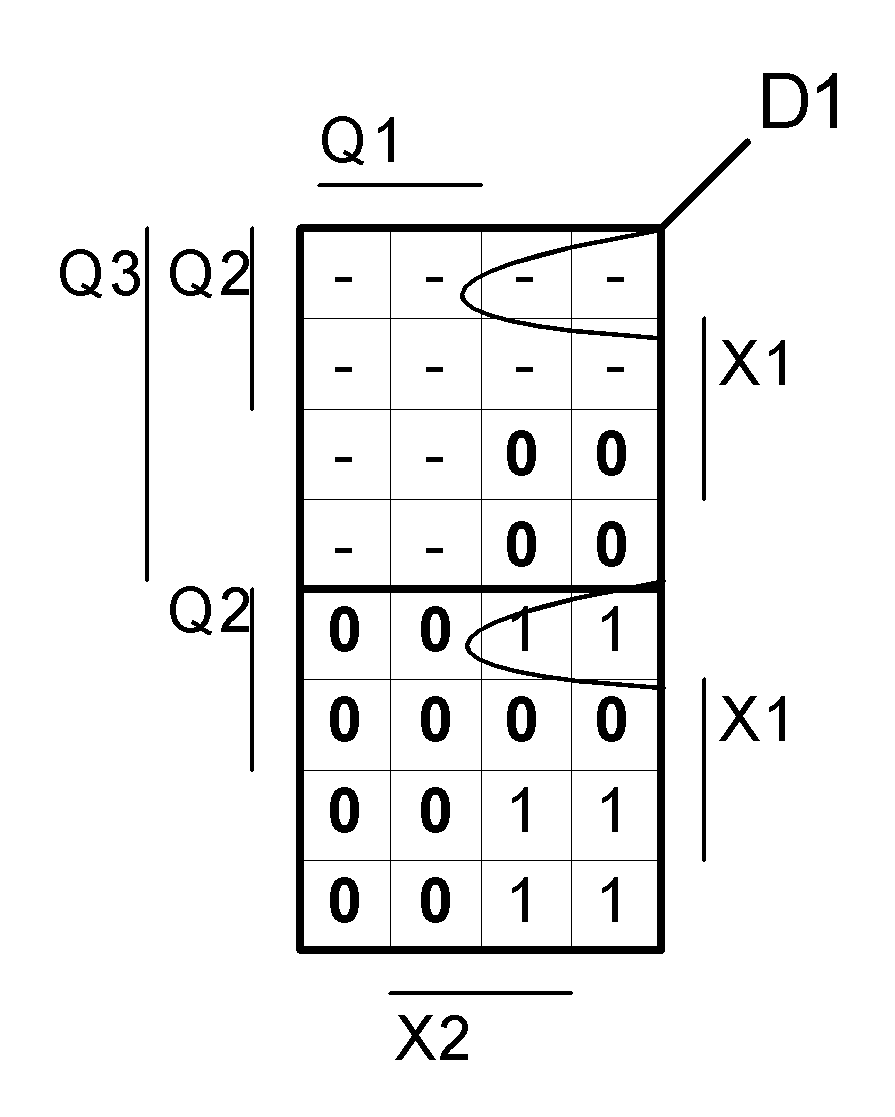
*Рисунок 3.2 – Граф циклічного автомата*

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

*Таблиця 3.2 – Таблиця переходів*

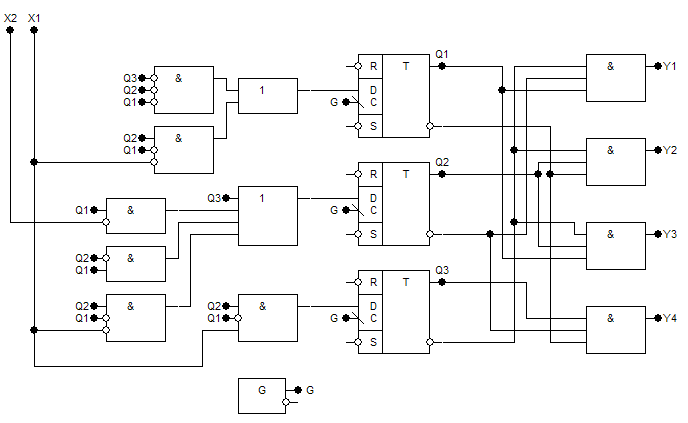
|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Пер. | | Ст. ст. | Нов. стан | Вх. сигн. | Вих. сигн. | Функціїтригерів | | | |
| Q3Q2Q1 | Q3Q2Q1 | X1X2 | Y1Y2Y3Y4 | D3 | D2 | | D1 |
| Z1→Z2 | 000 | | 001 | -- | 0000 | 0 | | 0 | 1 |
| Z2→Z3 | 001 | | 010 | -- | 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*

**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 ділення 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.

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»