НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

# "КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені Ігоря Сікорського"

# ФАКУЛЬТЕТ ІНФОРМАТИКИ ТА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

# Кафедра обчислювальної техніки

## РОЗРАХУНКОВА ГРАФІЧНА РОБОТА

з дисципліни „Комп'ютерна логіка 2. Комп'ютерна арифметика”

Виконала

Бровченко Анастасія Вікторівна

Група ІО-64

Факультет ІОТ

Залікова книжка № 6403

Керівник: Жабін В. І.

Київ – 2014 р.

**Завдання**

1. Числа  і  в прямому коді записати у формі з плаваючою комою у класичному варіанті (з незміщеним порядком і повною мантисою). На порядок відвести 4 розряди, на мантису 7 розрядів (з урахуванням знакових розрядів). Записати числа і  також за стандартом ANSI/IEEE 754-2008 в короткому 32-розрядному форматі).

2. Виконати 8 операцій з числами, що подані з плаваючою комою в класичному варіанті (чотири способи множення, два способи ділення, додавання та обчислення кореня додатного числа). Номери операцій (для п.3) відповідають порядку переліку, починаючи з нуля (наприклад, 0 – множення першим способом; 5 – ділення другим способом). Операндами для першого способу множення є задані числа та . Для кожної наступної операції першим операндом є результат попередньої операції, а другим операндом завжди є число . (Наприклад, для ділення першим способом першим операндом є результат множення за четвертим способом, для операції обчислення кореня операндом є результат додавання зі знаком плюс).

Для обробки мантис кожної операції, подати:

2.1 теоретичне обґрунтування способу;

2.2 операційну схему;

2.3 змістовний (функціональний) мікроалгоритм;

2.4 таблицю станів регістрів (лічильника), довжина яких забезпечує одержання 6 основних розрядів мантиси результату;

2.5 обробку порядків (показати у довільній формі);

2.6 форму запису нормалізованого результату з плаваючою комою в пам’ять комп’ютера в прямому коді.

Вказані пункти для операції 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  додатково виконати:

3.1 побудувати функціональну схему з відображенням управляючих сигналів, входів для запису операндів при ініціалізації пристрою і схем формування внутрішніх логічних умов;

3.2 розробити закодований (структурний) мікроалгоритм (мікрооперації замінюються управляючими сигналами виду W,SL,SR тощо);

3.3 для операції з парним двійковим номером  додатково подати граф управляючого автомата Мура з кодами вершин, а для непарного номера – автомата Мілі;

3.4 побудувати управляючий автомат на тригерах та елементах булевого базису. Вибрати -тригери для автомата Мура та -тригери для автомата Мілі.

**Визначення та обґрунтування варіанту:**

Перевести номер залікової книжки в двійкову систему. Записати два 10-розрядних двійкових числа:

 і ,

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

640310=11001000000112;

 = -001000,0111

 = +110000,0011

**Основна частина:**

**Завдання №1**

= 1.001000,0111; = 0.110000,0011;

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

X2:

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

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

Y2:

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

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

Представлення чисел за стандартом ANSI/IEEE 754-2008 в короткому 32-розрядному форматі

Ex = 127 + 4 = 13110 = 100000112

X2

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

Порядок

Мантиса

Ey = 127 + 6 = 13310 = 100001012

Y2

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

Порядок

Мантиса

**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.

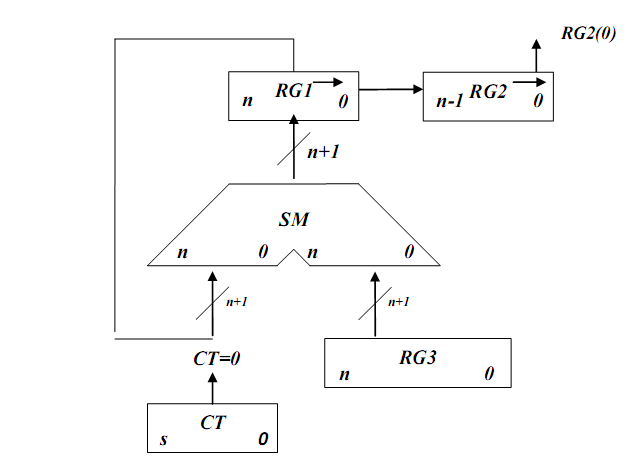
Множення мантис першим способом здійснюється з молодших розрядів множника, сума часткових добутків зсувається вправо, а множене залишається нерухомим. Тоді добуток двох чисел представляється у вигляді:

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:=6;

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*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** | **RG1** | **RG2** | **RG3** | **CT** |
| **пс** | 0 | 100001 | 0110000 | 110 |
| **1** | 0011000 | 010000 |  | 101 |
| **2** | 0001100 | 001000 |  | 100 |
| **3** | 0000110 | 000100 |  | 011 |
| **4** | 0000011 | 000010 |  | 010 |
| **5** | 0000001 | 100001 |  | 001 |
| **6** | +  0110000  =  0110001  **0011000** | **110000** |  | 000 |

**2.1.5 Обробка порядків:**

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

=4; =6; =1010=10112

**2.1.6 Нормалізація результату:**

Отримали результат: 0011000110000

Знак мантиси: 1 0 = 1.

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

Порядок зменшуємо на 2:

11000110000; =82=10002;

Запишемо нормалізований результат:

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

**2.2 Другий спосіб множення.**

**2.2.1 Теоретичне обґрунтування другого способу множення:**

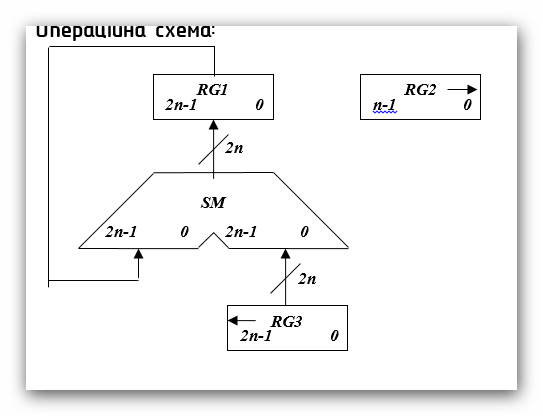
Числа множаться у прямих кодах, знакові та основні розряди обробляються окремо. Визначення знака добутку здійснюють підсумування по модулю 2 цифр, що розміщуються в знакових розрядах співмножників.

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

Z=Y+ Y…+ Y; Z=((0+ Y)+ Y)…+ Y;

Z=;

**2.2.2 Операційна схема:**

****

*Рисунок 2.2.1 - Операційна схема*

**2.2.3 Змістовний мікроалгоритм:**

RG1:=0;

RG2:=X;

RG3:=Y;

RG2[0]

RG1:=RG1+RG3;

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

RX=0

Початок

Кінець

0

1

0

1

*Рисунок 2.2.2 - Змістовний мікроалгоритм.*

**2.2.4 Таблиця станів регістрів:**

*Таблиця 2.2.1 -Таблиця станів регістрів*

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **RG1** | **RG3 ←** | **RG2 →** |
| **ПС** | 0 | 000000110000 | 110001 |
| **1** | 000000110000 | 000001100000 | 011000 |
| **2** | 000000110000 | 000011000000 | 001100 |
| **3** | 000000110000 | 000110000000 | 000110 |
| **4** | 000000110000 | 001100000000 | 000011 |
| **5** | +  001100000000  =  001100110000 | 011000000000 | 000001 |
| **6** | +  011000000000  =  **100100110000** | 110000000000 | 000000 |

**2.2.5 Обробка порядків:**

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

=8; =6; =1410=11102

**2.2.6 Нормалізація результату:**

Отримали результат: 100100110000

Знак мантиси: 1 0 = 1.

Нормалізація мантиси не потрібна.

Запишемо нормалізований результат:

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

**2.3 Третій спосіб множення.**

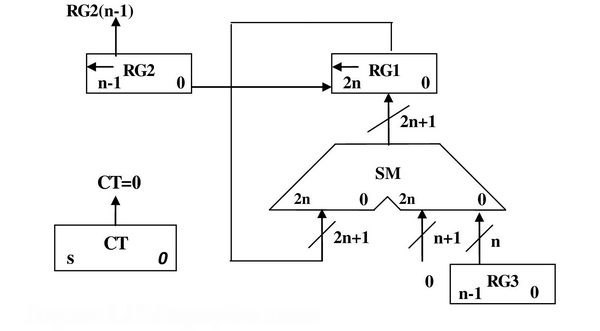
**2.3.1Теоретичне обгрунтування третього способу множення:**

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:**

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** | 0 | 100100 | 110000 | 110 |
| **1** | 000001100000 | 001000 |  | 101 |
| **2** | 000011000000 | 010000 |  | 100 |
| **3** | 000110000000 | 100000 |  | 011 |
| **4** | +  000000110000  =  000110110000  001101100000 | 000000 |  | 010 |
| **5** | 011011000000 | 000000 |  | 001 |
| **6** | **110110000000** | 000000 |  | 000 |

**2.3.5 Обробка порядків:**

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

=14; =6; =2010=101002

**2.3.6 Нормалізація результату:**

Отримали результат: 110110000000

Знак мантиси: 1 0 = 1.

Нормалізація мантиси не потрібна.

Запишемо нормалізований результат:

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

**2.4 Четвертий спосіб множення.**

**2.4.1 Теоритичне 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.

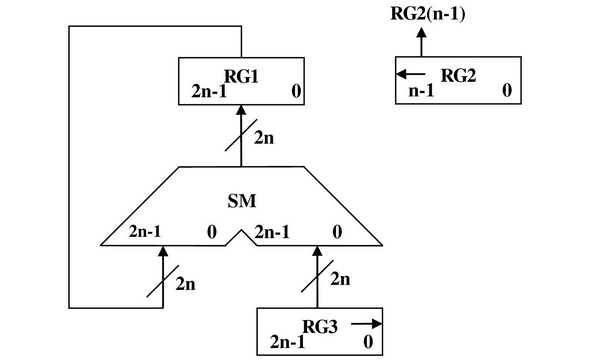
Множення здійснюється зі старших розрядів множника, сума часткових добутків залишається нерухомою, множене зсувається праворуч, множник ліворуч.

.

*.*

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 | 0 | 011000000000 | 110110 |
| **1** | 011000000000 | 001100000000 | 101100 |
| **2** | +  001100000000  =  100100000000 | 000110000000 | 011000 |
| **3** | 100100000000 | 000011000000 | 110000 |
| **4** | +  000011000000  =  100111000000 | 000001100000 | 100000 |
| **5** | +  000001100000  =  101000100000 | 000000110000 | 000000 |
| **6** | **101000100000** | 000000011000 | 000000 |

**2.4.5 Обробка порядків:**

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

=20; =6; =2610=110102

**2.4.6 Нормалізація результату:**

Отримали результат: 101000100000

Знак мантиси: 1 0 = 1.

Нормалізація мантиси не потрібна.

Запишемо нормалізований результат:

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

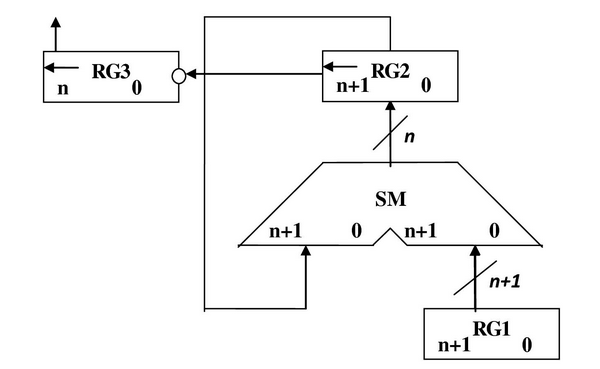
**2.5. Першиий спосіб ділення.**

**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 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:**

|  |  |  |  |
| --- | --- | --- | --- |
| *Таблиця 2.5.1- Таблиця станів регістрів* | | | |
| **№** | **RG3(Z)** | **RG2(X)** | **RG1(Y)** |
| **пс** | 000000 | 00101000 | 110000 |
| **1** | 0000001 | 01010000  +  11010000  =  00100000 |  |
| **2** | 0000011 | 01000000  +  11010000  =  00010000 |  |
| **3** | 0000111 | 00100000  +  11010000  =  11110000 |  |
| **4** | 0001110 | 11100000  +  00110000  =  00010000 |  |
| **5** | 0011101 | 00100000  +  11010000  =  11110000 |  |
| **6** | 0111010 | 11100000  +  00110000  =  00010000 |  |
| **7** | 1110101 | 00100000  +  11010000  =  **11110000** |  |

**2.5.5 Обробка порядків:**

Порядок частки буде дорівнювати:

В моєму випадку =26; =6; =2010=101002;

**2.5.6 Нормалізація результату:**

Отримали результат: 11110000

Знак мантиси: 1 0 = 1.

Нормалізація мантиси не потрібна.

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

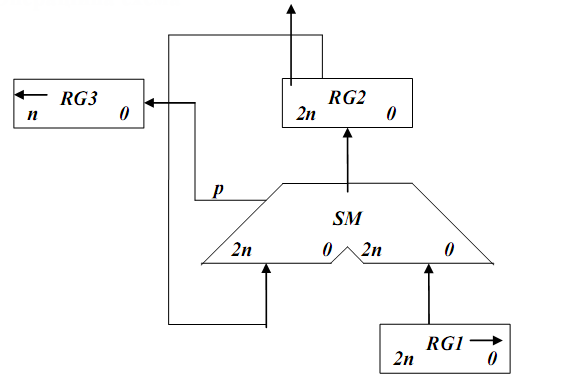
**2.6. Другий спосіб ділення.**

**2.6.1 Теоритичне 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** | 000000 | 001111000000 | 011000000000 |
| **1** | 000001 | 000111100000 | 011000000000  +  110001000000  =  001001000000 |
| **2** | 000011 | 000011110000 | 001001000000  +  111000100000  =  000001100000 |
| **3** | 000110 | 000001111000 | 000001100000  +  111100010000  =  111101110000 |
| **4** | 001100 | 000000111100 | 111101110000  +  000001111000  =  111111101000 |
| **5** | 011001 | 000000011110 | 111111101000  +  000000111100  =  000000100100 |
| **6** | **110011** | 000000001111 | 000000100100  +  111111100010  =  000000000110 |

**2.6.5 Обробка порядків:**

Порядок частки буде дорівнювати:

В моєму випадку =20; =6; =14;

**2.6.6 Нормалізація результату:**

Отримали результат: 110011

Знак мантиси: 1 0 = 1.

Нормалізація мантиси не потрібна.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 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=+1410=+11102

Py=+610=+01102

1410-610=810=10002

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 |
| 0,110000 | 1000 | Початковий стан |
| 0,011000 | 111 | My= 0.r(My); ∆:=∆-1 |
| 0,001100 | 110 | My= 0.r(My); ∆:=∆-1 |
| 0,000110 | 101 | My= 0.r(My); ∆:=∆-1 |
| 0,000011 | 100 | My= 0.r(My); ∆:=∆-1 |
| 0,000001 | 11 | My= 0.r(My); ∆:=∆-1 |
| 0,000000 | 10 | My= 0.r(My); ∆:=∆-1 |
| 0,000000 | 01 | My= 0.r(My); ∆:=∆-1 |
| 0,000000 | 00 | My= 0.r(My); ∆:=∆-1 |

3. Додавання мантис у модифікованому ДК.

X мдк = 11.001101

Yмдк = 00.000000

*Таблиця 2.7.2-Додавання мантис(для додавання)*

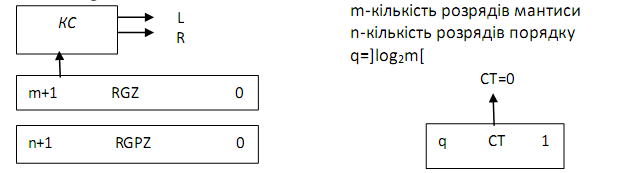
|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| MX | 1 | 1. | 0 | 0 | 1 | 1 | 0 | 1 |
| MY | 0 | 0. | 0 | 0 | 0 | 0 | 0 | 0 |
| MZ | 1 | 1. | 0 | 0 | 1 | 1 | 0 | 1 |

Zпк = 1.110011

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 Таблиця станів регістрів**

**Додавання**

*Таблиця 2.7.5- Таблиця станів регістрів*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№**  **такту** | **RGPZ** | **RGZ** | **ЛПН(L)** | **ППН(R)** | **СT** | **Мікрооперація** |
| **ПС** | 001001 | 11,001101  +  00,000000  =  11,001101 | 1 | 0 | 110 |  |

**2.7.4 Обробка порядків**

PX+Y= 1410 =11102

**2.7.5 Форма запису результату з плаваючою комою**

Результат додавання Z=X+Y.

Zпк = 1.110011

Pz = 1410 =11102 Mz = 1100112

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

**2.7.6 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*

**2.8.Операція добування кореня**

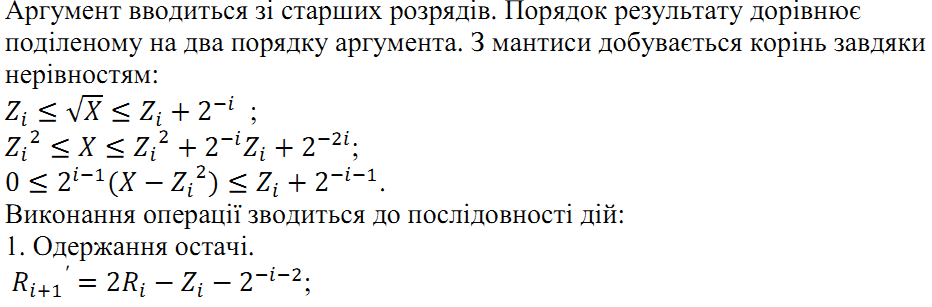
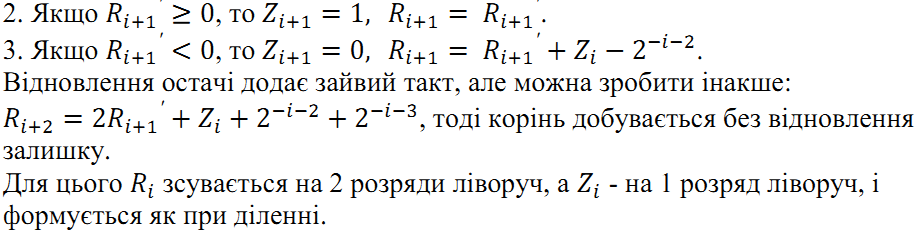
**2.8.1Теоритичне обґрунтування операції обчислення квадратного кореня**

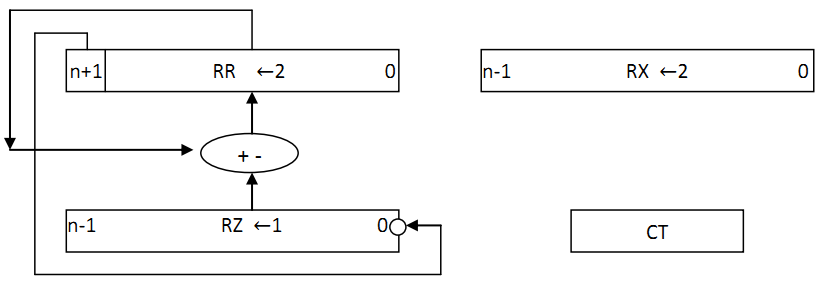
Аргумент вводиться зі старших розрядів. Порядок результату дорівнює поділеному на 2 порядку аргумента. З мантиси добувається корінь завдяки нерівностям:

; ;

.

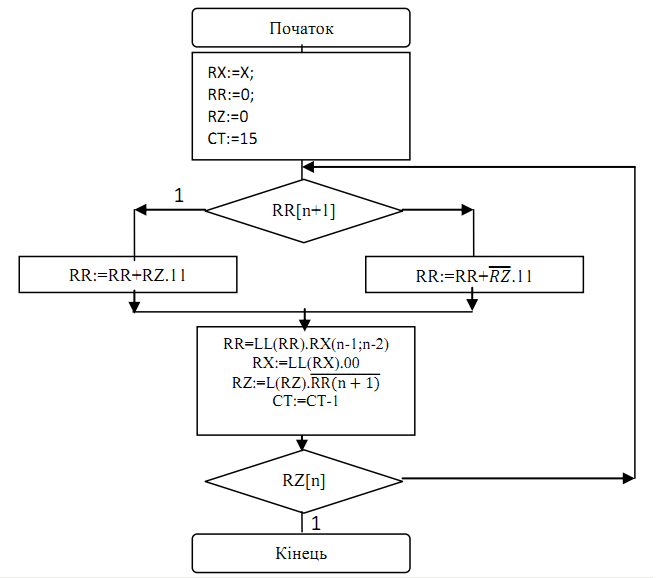
Виконання операції зводиться до послідовності дій:



**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** | 000000 | 00000000  00000011 | 110011 | 110 |
| **ПЗ** |
| **1** | 000001 | 00000011  +  11111111  =  00000010  00001000 | 011001 | 101 |
| **2** | 000011 | 00001000  +  11111011  =  00000011  00001111 | 001100 | 100 |
| **3** | 000111 | 00001111  +  11110011  =  00000010  00001000 | 000110 | 011 |
| **4** | 001110 | 00001000  +  11100011  =  11101011  10101100 | 000011 | 010 |
| **5** | 011100 | 10101100  +  00111011  =  11100111  10011100 | 000001 | 001 |
| **6** | **111001** | 10011100  +  01110011  =  00001111  00111100 | 000000 | 000 |

**2.8.8 Обробка порядків**

В моєму випадку =7;

**2.8.9 Запис результату**

Отримали результат Z = 111001;

Результат нормалізований, готовий до запису у мантису:

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

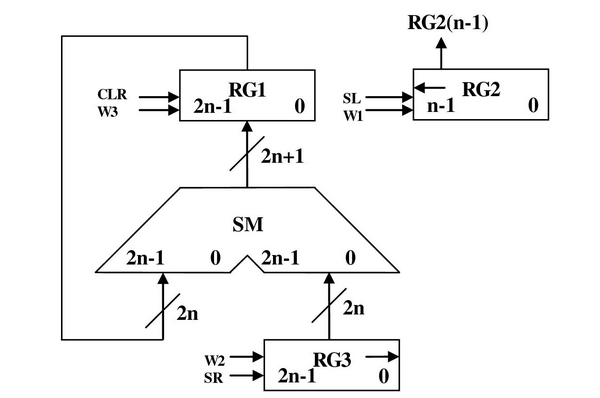
**Завдання 3**

x3 x2 x1  = 0112 = 310 – непарне.

Синтез управляючого автомату Мілі на RS-тригерах для операції множення 4 способом.

**3.1 Таблиця кодування сигналів**

**3.1.1 Функціональна схема:**



*Рисунок 3.1.1* **-** *Функціональна схема.*

**3.1.2 Закодований мікроалгоритм**

*Таблиця 3.1.1 -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

R, W2, W3, ShR

X1

ShR,ShL

X2

Кінець

1

0

1

0

**Z1**

**Z2**

W1

**Z3**

**Z4**

**Z5**

*Рисунок 3.1.3 -Закодований мікроалгоритм.*

Початок

Y1

X1

X2

Кінець

1

0

1

0

**Z1**

**Z2**

Y2

**Z3**

Y3

**Z4**

**Z5**

*Рисунок 3.1.3 -Закодований мікроалгоритм для автомата.*

**3.1.3 Граф управляючого автомата Мілі з кодами вершин:**

**001**

-/Y1

**000**

/-

/-

X1/Y2

**011**

**110**

-/Y3

X2/-

**010**

*Рисунок 3.2 – Граф автомата*

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

*Таблиця 3.2 – Таблиця переходів*

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Пер. | | Ст. ст. | Нов. стан | Вх. сигн. | Вих. сигн. | Функції тригерів | | | |
| Q3Q2Q1 | Q3Q2Q1 | X2X1 | Y1Y2Y3 | R1S1 | R2S2 | | R3S3 |
| Z1→Z2 | 000 | | 001 | -- | 100 | -0 | | -0 | 01 |
| Z2→Z3 | 001 | | 011 | 0- | 000 | -0 | | 01 | 0- |
| Z2→Z3 | 001 | | 011 | 1- | 010 | -0 | | 01 | 0- |
| Z3→Z4 | 011 | | 010 | -- | 001 | -0 | | 0- | 10 |
| Z4→Z2 | 010 | | 001 | -0 | 000 | -0 | | 10 | 01 |
| Z4→Z5 | 010 | | 110 | -1 | 000 | 01 | | 0- | -0 |

**3.5 Мінімізація функцій тригерів**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  | R1 |
|  |  | Q1 | |  |  |  |
| Q2 |  | - | - | - | - |  |
| - | - | - | - | Q3 |
| X2 | - | - | - | - |
| - | - | 0 | 0 |  |
|  | - | - | - | - |  |
|  | - | - | - | - | Q3 |
|  |  | - | - | - | - |
|  |  | - | - | - | - |  |
|  |  |  | X1 | |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  | S1 |
|  |  | Q1 | |  |  |  |
| Q2 |  | - | - | 0 | 0 |  |
| - | - | 0 | 0 | Q3 |
| X2 | - | - | 0 | 0 |
| - | - | 1 | 1 |  |
|  | - | - | 0 | 0 |  |
|  | - | - | 0 | 0 | Q3 |
|  |  | - | - | 0 | 0 |
|  |  | - | - | 0 | 0 |  |
|  |  |  | X1 | |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  | R2 |
|  |  | Q1 | |  |  |  |
| Q2 |  | - | - | 1 | 1 |  |
| - | - | 0 | 0 | Q3 |
| X2 | - | - | 0 | 0 |
| - | - | 0 | 0 |  |
|  | - | - | - | - |  |
|  | - | - | 0 | 0 | Q3 |
|  |  | - | - | 0 | 0 |
|  |  | - | - | - | - |  |
|  |  |  | X1 | |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  | S2 |
|  |  | Q1 | |  |  |  |
| Q2 |  | - | - | 0 | 0 |  |
| - | - | - | - | Q3 |
| X2 | - | - | - | - |
| - | - | - | - |  |
|  | - | - | 0 | 0 |  |
|  | - | - | 1 | 1 | Q3 |
|  |  | - | - | 1 | 1 |
|  |  | - | - | 0 | 0 |  |
|  |  |  | X1 | |  |  |

*Рисунок 3.3 – Мінімізація функцій тригерів*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  | R3 |
|  |  | Q1 | |  |  |  |
| Q2 |  | - | - | 0 | 0 |  |
| - | - | 1 | 1 | Q3 |
| X2 | - | - | 1 | 1 |
| - | - | - | - |  |
|  | - | - | 0 | 0 |  |
|  | - | - | 0 | 0 | Q3 |
|  |  | - | - | 0 | 0 |
|  |  | - | - | 0 | 0 |  |
|  |  |  | X1 | |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  | S3 |
|  |  | Q1 | |  |  |  |
| Q2 |  | - | - | 1 | 1 |  |
| - | - | 0 | 0 | Q3 |
| X2 | - | - | 0 | 0 |
| - | - | 0 | 0 |  |
|  | - | - | 1 | 1 |  |
|  | - | - | - | - | Q3 |
|  |  | - | - | - | - |
|  |  | - | - | 1 | 1 |  |
|  |  |  | X1 | |  |  |

*Рисунок 3.3 – Мінімізація функцій тригерів (продовження)*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  | Y1 |
|  |  | Q1 | |  |  |  |
| Q2 |  | - | - | 0 | 0 |  |
| - | - | 0 | 0 | Q3 |
| X2 | - | - | 0 | 0 |
| - | - | 0 | 0 |  |
|  | - | - | 1 | 1 |  |
|  | - | - | 0 | 0 | Q3 |
|  |  | - | - | 0 | 0 |
|  |  | - | - | 1 | 1 |  |
|  |  |  | X1 | |  |  |

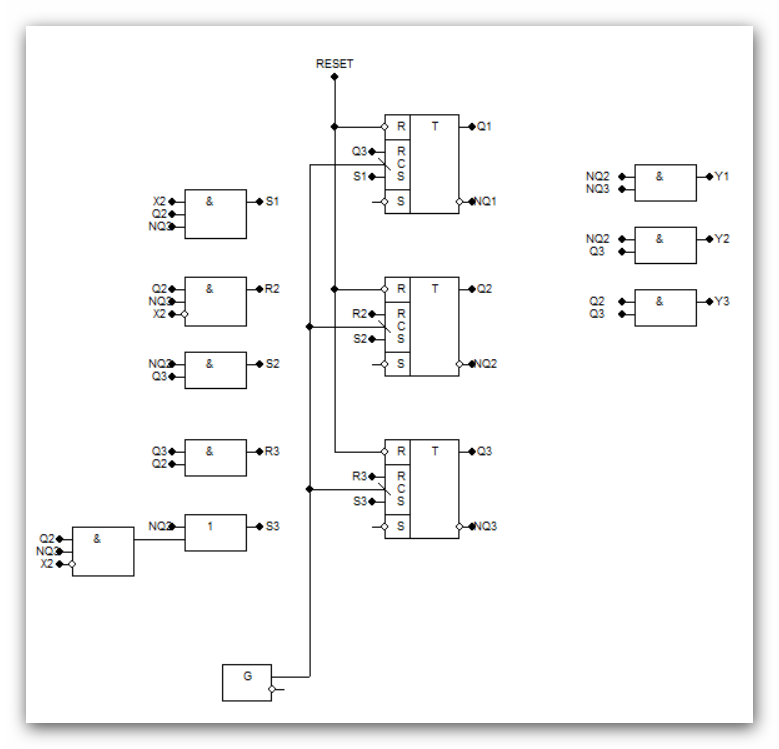
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  | Y2 |
|  |  | Q1 | |  |  |  |
| Q2 |  | - | - | 0 | 0 |  |
| - | - | 0 | 0 | Q3 |
| X2 | - | - | 0 | 0 |
| - | - | 0 | 0 |  |
|  | - | - | 0 | 0 |  |
|  | - | - | 1 | 1 | Q3 |
|  |  | - | - | 1 | 1 |
|  |  | - | - | 0 | 0 |  |
|  |  |  | X1 | |  |  |

*Рисунок 3.4 – Діаграми Вейча для вихідних сигналів*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  | Y3 |
|  |  | Q1 | |  |  |  |
| Q2 |  | - | - | 0 | 0 |  |
| - | - | 1 | 1 | Q3 |
| X2 | - | - | 1 | 1 |
| - | - | 0 | 0 |  |
|  | - | - | 0 | 0 |  |
|  | - | - | 0 | 0 | Q3 |
|  |  | - | - | 0 | 0 |
|  |  | - | - | 0 | 0 |  |
|  |  |  | X1 | |  |  |

*Рисунок 3.4 – Діаграми Вейча для вихідних сигналів (продовження)*

**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. |

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).