НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

# "КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ"

#### ФАКУЛЬТЕТ ІНФОРМАТИКИ І ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

### Кафедра обчислювальної техніки

## РОЗРАХУНКОВА ГРАФІЧНА РОБОТА

з дисципліни ” Комп’ютерна логіка 2. Комп’ютерна арифметика ”

Виконав

Лисенко Дмитро Вадимович

Факультет ІОТ,

Група ІО-61,

Залікова книжка № 6116

Керівник \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

(підпис керівника)

Київ - 2017р.

***Вибір 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).



***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.

**Завдання №1**



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

Px=; Mx=;

Py=; My=;



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

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



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

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

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

Ex = Px+ (2m-1-1) =Px+ (27-1) =1102+11111112= 100001012

Ey = Py+(2m-1-1)=Py+(27-1)=1102+11111112= 100001012



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



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

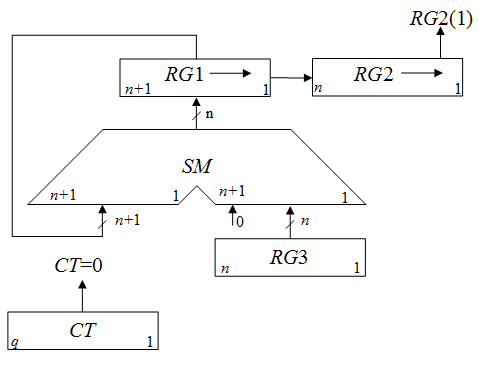
**Завдання №2**

**2.1 Перший спосіб множення.**

2.1.1 Теоретичне обґрунтування першого способу множення:

Під час множення *першим способом* в першому такті *i*-го циклу аналізується значення *RG*2[1] – молодшого (*n*-го) розряду регістру *RG*2, в якому знаходиться чергова цифра множника. Вміст *RG*3 додається до суми часткових добутків, що знаходяться в регістрі *RG*1, якщо *RG*2[1]=1, або не додається, якщо *RG*2[1]=0. В другому такті здійснюється правий зсув у регістрах *RG*1 і *RG*2, що еквівалентно множенню їхнього вмісту на 2–1. При зсуві цифра молодшого розряду регістру *RG*1 записується у вивільнюваний старший розряд регістру *RG*2. Після виконання *n* циклів молодші розряди 2*n*-розрядного добутку будуть записані в регістр *RG*2, а старші – у *RG*1.

2.1.2 Операційна схема:



*Рисунок 2.1.1 Операційна схема пристрою для множення першим способом*

2.1.3 Змістовний мікроалгоритм:

Початок

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

0

RG2[1]

1

RG1:=RG1+RG3

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

0

CT=0

Кінець

1

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

2.1.4 Таблиця станів регістрів:

*Таблиця 2.1.1. Таблиця станів регістрів пристрою множення першим способом*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | RG1🡪 | RG2🡪 | RG3 | CT |
| П.С. | 0000000 | 11100**1** | 111110 | 110 |
| 1 | +  0011111  =  0011111 | 01110**0** | 111110 | 101 |
| 2 | 0011111 | 00111**0** | 111110 | 100 |
| 3 | 0011111 | 00011**1** | 111110 | 011 |
| 4 | +  0111110  =  1000101  0100010 | 00001**1** | 111110 | 010 |
| 5 | +  0111110  =  1100000  0110000 | 10000**1** | 111110 | 001 |
| 6 | +  0111110  =  1101110  0**110111** | **010000** | 111110 | 000  **кінець** |

2.1.5 Обробка порядків і нормалізація:

.

Отримали результат: 0,110111010000

Округлена мантиса: Mz = 0,110111; .

Знак мантиси: .

2.1.6 Форма запису нормалізованого результату з плаваючою комою в пам’ять

Зн.РZ PZ Зн.МZ MZ

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

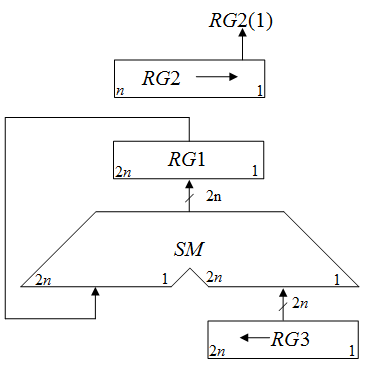
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1. | 1 | 1 | 0 | 1 | 1 | 1 |

**2.2 Другий спосіб множення**

2.2.1 Теоретичне обґрунтування другого способу множення:

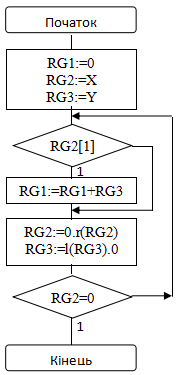
Перед початком множення *другим способом* множник *Х* записують в регістр *RG*2, а множене *Y* – в молодші розряди регістру *RG*3 (тобто в регістрі *RG*3 установлюють *Y*0 = *Y*2*–n*). В кожному *i*-му циклі множення додаванням кодів *RG*3 і *RG*1 управляє цифра *RG*2[1], а в регістрі *RG*3 здійснюється зсув вліво на один розряд, в результаті чого формується величина *Yi* = 2*Yi–*1. Оскільки сума часткових добутків в процесі множення нерухома, зсув в регістрі *RG3* можна виконати суміщення в часі з підсумовуванням (як правило, *t*п ≥ *t*з). В цьому випадку *t*м = *nt*п. Завершення операції множення визначається за нульовим вмістом регістру *RG*2, що також приводить до збільшення швидкодії, якщо множник ненормалізований.

2.2.2 Операційна схема

**

*Рисунок 2.2.1. Операційна схема пристрою множення другим способом*

2.2.3 Змістовний мікроалгоритм



*Рисунок 2.2.2. Змістовний мікроалгоритм пристрою множення другим способом*

2.2.4 Таблиця станів регістрів:

В ЕОМ при роботі із дробовими числами часто потрібно обчислювати не 2*n*, а тільки (*n*+1) цифр добутку й округляти його до n розрядів. В цьому випадку при реалізації другого способу можна зменшити довжину *SM* і *RG*1, а при реалізації четвертого – зменшити довжину *SМ*, *RG*1 і *RG*3. Для того щоб похибка від відкидання молодших розрядів не перевищила половини ваги *n*-го розряду результату, в перерахованих вузлах досить мати тільки по *l* додаткових молодших розрядів, де *l* вибирається з умови

*l* ≥1+ log2(*n* – *l* – 1).

При *n*=6 мінімальне l, яке задовольняє дану умову дорівнює *l=2*. Отже розрядність RG1 та суматора замінюємо на *n+l=*6*+*2*=*8.

*Таблиця 2.2.1. Таблиця станів регістрів пристрою множення другим способом*

|  |  |  |  |
| --- | --- | --- | --- |
| № | RG1 | RG2🡪 | RG3🡨 |
| П.С. | 00000000 | 11011**1** | 000000111110 |
| 1 | +  00000011  =  00000011 | 01101**1** | 000001111100 |
| 2 | +  00000111  =  00001010 | 00110**1** | 000011111000 |
| 3 | +  00001111  =  00011001 | 00011**0** | 000111110000 |
| 4 | 00011001 | 00001**1** | 001111100000 |
| 5 | +  00111110  =  01010111 | 00000**1** | 011111000000 |
| 6 | +  01111100  =  **1101001**1  (*n*+1) | 000000  **кінець** | 111110000000 |

2.2.5 Обробка порядків і нормалізація

.

Отримали результат: 0,1101001

Округлена мантиса: Mz = 0,110101 ; .

Знак мантиси: .

2.2.6 Форма запису нормалізованого результату з плаваючою комою в пам’ять

Зн.РZ PZ Зн.МZ MZ

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

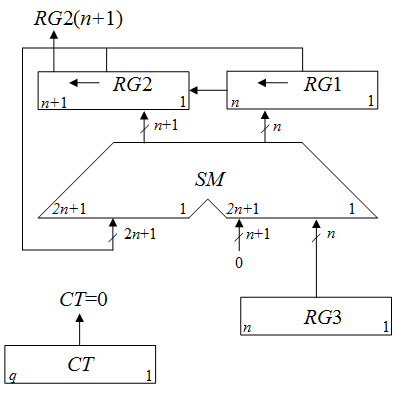
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1. | 1 | 1 | 0 | 1 | 0 | 1 |

**2.3 Третій спосіб множення.**

2.3.1 Теоретичне обґрунтування третього способу множення:

При множенні *третім способом* множник *Х* записується в старші розряди *RG*2, при цьому *RG*2[1]=0. Вага молодшого розряду *RG*3 дорівнює 2–2*n*, тому код в регістрі *RG*3 являє собою значення *Y*2–*n*. В кожному циклі множення підсування виконується при *RG*2[*n+*1]=1. В регістрах *RG*1 і *RG*2 виконується лівий зсув. В результаті підсумовування вмісту *RG*3 і *RG*1 може виникнути перенос в молодший розряд регістру *RG*2, що реалізується на *SM.* Збільшення довжини *RG*2 на один розряд усуває можливість поширення переносу в розряди множника. Після виконання *n* циклів молодші розряди добутку будуть знаходитися в регістрі *RG*1, а старші – в регістрі *RG*2. Час множення третім способом визначається аналогічно першому способу.

2.3.2 Операційна схема



*Рисунок 2.3.1. Операційна схема пристрою множення третім способом*

2.3.3 Змістовний мікроалгоритм



*Рисунок 2.3.2. Змістовний мікроалгоритм пристрою множення третім способом*

2.3.4 Таблиця станів регістрів

*Таблиця 2.3.1. Таблиця станів регістрів пристрою множення третім способом*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № ц. | RG2🡨 | RG1🡨 | RG3 | СТ |
| П.С. | **1**101010 | 000000 | 111110 | 110 |
| 1 | +  0000000  =  1101010  **1**010101 | +  111110  =  111110  111100 | 111110 | 101 |
| 2 | +  0000001  =  1010110  **0**101101 | +  111110  =  111010  110100 | 111110 | 100 |
| 3 | **1**011011 | 101000 | 111110 | 011 |
| 4 | +  0000001  =  1011100  **0**111001 | +  111110  =  100110  001100 | 111110 | 010 |
| 5 | **1**110010 | 011000 | 111110 | 001 |
| 6 | +  0000001  =  1110011  **1100110** | +  111110  =  010110  **10110**0 | 111110 | 000  **кінець** |

2.3.5 Обробка порядків і нормалізація

.

Отримали результат: 0,110011010110

Округлена мантиса: Mz = 0,110011; .

Знак мантиси: .

2.3.6 Форма запису нормалізованого результату з плаваючою комою в пам’ять

Зн.РZ PZ Зн.МZ MZ

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

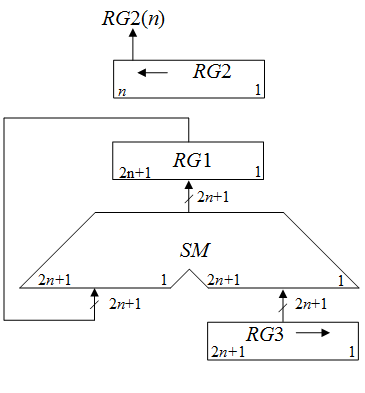
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1. | 1 | 1 | 0 | 0 | 1 | 1 |

**2.4 Четвертий спосіб множення.**

2.4.1 Теоретичне обґрунтування четвертого способу множення:

Перед множенням *четвертим способом* множник записують в регістр *RG*2, а множене – в старші розряди регістру *RG*3 (тобто в *RG*3 установлюють *Y*0=*Y*2–1). В кожнім циклі цифра *RG*2[*n*], що знаходиться в старшому розряді регістру *RG*2, управляє підсумовуванням, а в *RG*3 здійснюється правий зсув на один розряд, що еквівалентно множенню вмісту цього регістра на 2–1. Час виконання множення четвертим способом складає *tм*=*nt*п, визначається аналогічно другому способу.

2.4.2 Операційна схема



*Рисунок 2.4.1. Операційна схема пристрою множення четвертим способом*

2.4.3 Змістовний мікроалгоритм

Початок

Кінець

RG1:=0

RG2:=X

RG3:=0.Y.000000

RG1:=RG1+RG3

RG3:=0.r(RG3)

RG2:=l(RG2).0

RG2[n]

RG2=0

1

1

*Рисунок 2.4.2. Змістовний мікроалгоритм пристрою множення четвертим способом*

2.4.4 Таблиця станів регістрів

Замінюємо розрядність RG1, RG3 та SM на *n*+*l*+1= 6+2+1=9

*Таблиця 2.4.1. Таблиця станів регістрів пристрою множення четвертим способом*

|  |  |  |  |
| --- | --- | --- | --- |
| № ц. | RG1 | RG2🡨 | RG3🡪 |
| П.С. | 000000000 | **1**10011 | 011111000 |
| 1 | +  011111000  =  011111000 | **1**00110 | 001111100 |
| 2 | +  001111100  =  101110100 | **0**01100 | 000111110 |
| 3 | 101110100 | **0**11000 | 000011111 |
| 4 | 101110100 | **1**10000 | 000001111 |
| 5 | +  000001111  =  110000011 | **1**00000 | 000000111 |
| 6 | +  000000111  =  **1100010**10  (*n*+1) | 000000  **кінець** | 000000011 |

2.4.5 Обробка порядків і нормалізація

.

Отримали результат: 0,1100010

Округлена мантиса: Mz = 0,110001; .

Знак мантиси: .

2.4.6 Форма запису нормалізованого результату з плаваючою комою в пам’ять

Зн.РZ PZ Зн.МZ MZ

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

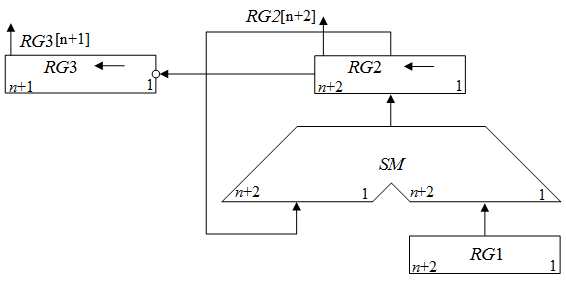
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1. | 1 | 1 | 0 | 0 | 0 | 1 |

**2.5 Перший спосіб ділення**

2.5.1 Теоретичне обґрунтування способу

При реалізації ділення за *першим* варіантом здійснюється зсув вліво залишку при нерухомому дільнику. Чергова остача формується в регістрі *RG*2 (у вихідному стані в цьому регістрі записаний *Х*). Виходи *RG*2 підключені до входів суматора *SM* безпосередньо, тобто ланцюги видачі коду з *RG*2 не потрібні. Дільник *Y* знаходиться в регістрі *RG*1. Результат формується в регістрі *RG*3 за (*n*+ 1) циклів. Знак остачі визначається розрядом *RG*2[*n*+2]. Розряд *RG*3[*n*+1] використовується для визначення кінця операції, ознакою цього є маркерний нуль на виході розряду. Максимальний час одержання цифри результату визначається виразом *tЦ*= *t*Д+ *t*З, де *t*Д – тривалість виконання мікрооперації додавання/віднімання; *t*З – тривалість виконання мікрооперації зсуву. Час для одержання *n*+1 цифри частки визначається виразом *t*=(*n*+1) *tЦ.*

2.5.2 Операційна схема



*Рисунок 2.5.1. Операційна схема пристрою ділення першим способом*

2.5.3 Змістовний мікроалгоритм

Початок

Кінець

RG3:=l(RG3).

RG2:=l(RG2).0

RG2:=RG2++1

RG2:=RG2+RG1

RG2[n+2]

RG3[n+1]

1

1

RG3:=0

RG2:=X;

RG1:=Y;

*Рисунок 2.5.2. Змістовний мікроалгоритм пристрою ділення першим способом*

2.5.4 Таблиця станів регістрів

*Таблиця 2.5.1. Таблиця станів регістрів пристрою ділення першим способом*

|  |  |  |  |
| --- | --- | --- | --- |
| № циклу | RG3🡨 | RG2🡨 | RG1 |
| П.С. | 0000000 | 00110001 | 00111110 |
| 1 | 0000001\* | **0**1100010  +  11000010  =  00100100 | 00111110 |
| 2 | 000001\*1 | **0**1001000  +  11000010  =  00001010 | 00111110 |
| 3 | 00001\*11 | **0**0010100  +  11000010  =  11010110 | 00111110 |
| 4 | 0001\*110 | **1**0101100  +  00111110  =  11101010 | 00111110 |
| 5 | 001\*1100 | **1**1010100  +  00111110  =  00010010 | 00111110 |
| 6 | 01\*11001 | **0**0100100  +  11000010  =  11100110 | 00111110 |
| 7 | 1\***110010**  **кінець** | **1**1001100  +  00111110  =  00001010 | 00111110 |

2.5.5 Обробка порядків і нормалізація

.

Мантиса: Mz = 0,110010; .

Знак мантиси: .

2.5.6 Форма запису нормалізованого результату з плаваючою комою в пам’ять

Зн.РZ PZ Зн.МZ MZ

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

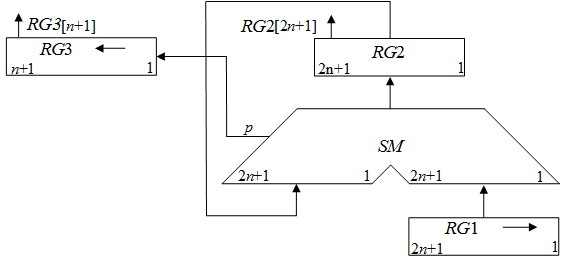
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1. | 1 | 1 | 0 | 0 | 1 | 0 |

**2.6. Другий спосіб ділення.**

2.6.1 Теоретичне обґрунтування другого способу ділення:

При реалізації ділення другим способом (із зсувом дільника) збільшується розрядність регістрів *RG*2, *RG*3 і суматора *SM* (рис. 3.2). В даному випадку процеси додавання/віднімання і зсуву можуть бути суміщені у часі. Отже, для ділення за другим способом час одержання цифри результату дорівнює *tЦ*= *t*Д. Цифра результату формується на виході переносу суматора *SM*(*p*)*.* Загальний час ділення визначається як .

2.6.2 Операційна схема



*Рисунок 2.6.1. Операційна схема пристрою ділення другим способом*

2.6.3 Змістовний мікроалгоритм

Початок

Кінець

RG2:=RG2+RG1

RG1:=0.r(RG1)

RG3:=l(RG3).SM(p)

RG2:=RG2++1

RG1:=0.r(RG1)

RG3:=l(RG3).SM(p)

RG2[2n+1]

RG3[n+1]

1

1

RG3:=1…11;

RG1:=Y;

RG2:= X

*Рисунок 2.6.2. Змістовний мікроалгоритм пристрою ділення другим способом*

2.6.4 Таблиця станів регістрів

*Таблиця 2.6.1. Таблиця станів регістрів пристрою ділення другим способом*

|  |  |  |  |
| --- | --- | --- | --- |
| № ц. | RG3🡨 | RG2 | RG1🡪 |
| П.С. | 1111111 | **0**110010000000 | 0111110000000 |
| 1 | 1111110\* | +  1000010000000  =  **1**110100000000 | 0011111000000 |
| 2 | 111110\*1 | +  0011111000000  =  **0**010011000000 | 0001111100000 |
| 3 | 11110\*11 | +  1110000100000  =  **0**000011100000 | 0000111110000 |
| 4 | 1110\*110 | +  1111000010000  =  **1**111011110000 | 0000011111000 |
| 5 | 110\*1100 | +  0000011111000  =  **1**111111101000 | 0000001111100 |
| 6 | 10\*11001 | +  0000001111100  =  **0**000001100100 | 0000000111110 |
| 7 | 0\***110011**  **кінець** | +  1111111000010  =  0000000100110 | 0000000011111 |

2.6.5 Обробка порядків і нормалізація

.

Мантиса: Mz = 0,110011; .

Знак мантиси: .

2.6.6 Форма запису нормалізованого результату з плаваючою комою в пам’ять

Зн.РZ PZ Зн.МZ MZ

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

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

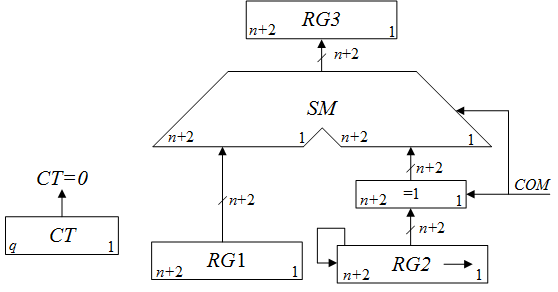
**2.7 Операція додавання чисел**

2.7.1 Теоретичне обґрунтування способу

На першому етапі додавання чисел з плаваючою комою виконують вирівнювання порядків до числа із старшим порядком.

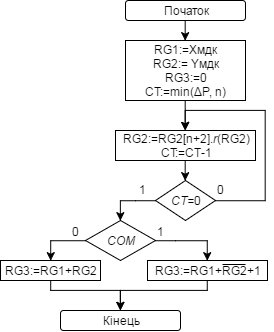
На другому етапі виконують додавання мантис. Додавання мантис виконується у *доповняльних кодах*.

2.7.2 Операційна схема



*Рисунок 2.7.1. Операційна схема пристрою додавання/віднімання у МДК*

2.7.3 Змістовний мікроалгоритм

**

*Рисунок 2.7.2. Змістовний мікроалгоритм пристрою додавання/віднімання у МДК*

2.7.4Таблиця станів регістрів

Вирівнювання порядків:

,

.

ΔP = 1210; n = 610; min(ΔP, n) = 610 = 1102

Числа у МДК:

****

*Таблиця 2.7.1. Таблиця станів регістрів пристрою додавання/віднімання у МДК двох чисел*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № ц. | RG1 | RG2🡪 | RG3 | CT |
| П.С. | 11001101 | 00111110 | 00000000 | 110 |
| 1 | 11001101 | 00011111 | 00000000 | 101 |
| 2 | 11001101 | 00001111 | 00000000 | 100 |
| 3 | 11001101 | 00000111 | 00000000 | 011 |
| 4 | 11001101 | 00000011 | 00000000 | 010 |
| 5 | 11001101 | 00000001 | 00000000 | 001 |
| 6 | 11001101 | 00000000 | 00000000 | 000 |
| - | 11001101 | 00000000 | 11001101  +  00000000  =  **11.001101** | 000 |

2.7.5 Обробка порядків

Мантиса: Mz = 0,110011; .

Знак мантиси: .

2.7.6 Форма запису нормалізованого результату з плаваючою комою в пам’ять

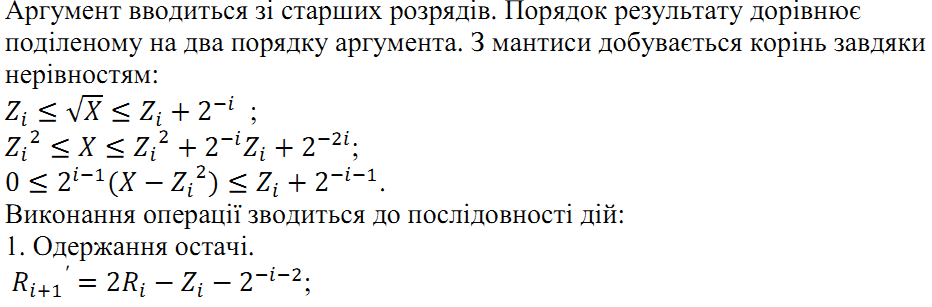
Зн.РZ PZ Зн.МZ MZ

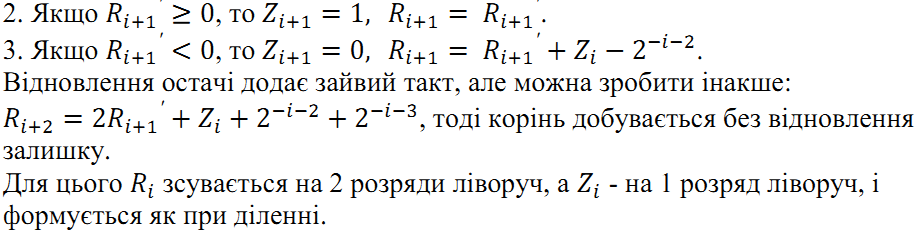
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0. | 1 | 0 | 0 | 1 | 0 |

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

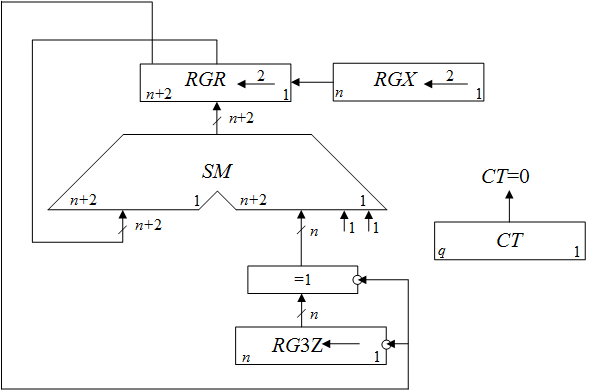
**2.8 Операція добування кореня додатного числа**

2.8.1 Теоретичне обґрунтування способу





2.8.2 Операційна схема

****

*Рисунок 2.8.1. Операційна схема пристрою знаходження кореня*

2.8.3 Змістовний мікроалгоритм

RGR := *l*2(RGR).RGX[n,n-1];

RGX:=*l*2(RGX).00

так

ні

1

Початок

RGR := 0; RGX := X;

RGZ := 0; CT := 110;

RGR[n+2]

RGR:= RGZ.11+ RGR;

Rr := *l*2(Rr).Rx[1, 2];

Rz := *l*(Rz).Rr[n+2];

CT := CT – 1;

Rx := 00.*r*2(Rx);

CT = 0

Кінець

*Рисунок 2.8.2 Змістовний мікроалгоритм пристрою знаходження кореня*

RGR:= .11+ RGR;

2.8.4Таблиця станів регістрів

*Таблиця 2.8.1. Таблиця станів регістрів пристрою добування кореня*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № ц. | RGZ🡨 | RGR🡨 | RGX🡨 | CT |
| ПС | 000000 | 00000000  **0**0000011 | 110011  001100 | 110 |
| 1 | 000001 | +  11111111  =  00000010  **0**0001000 | 110000 | 101 |
| 2 | 000011 | +  11111011  =  00000011  **0**0001111 | 000000 | 100 |
| 3 | 000111 | +  11110011  =  00000010  **0**0001000 | 000000 | 011 |
| 4 | 001110 | +  11100011  =  11101011  **1**0101100 | 000000 | 010 |
| 5 | 011100 | +  00111011  =  11100111  **1**0011100 | 000000 | 001 |
| 6 | **111001** | +  01110011  =  00001111  00111100 | 000000 | 000  **кінець** |

2.8.5 Обробка порядків

Мантиса: Mz = 0,111001; .

Шукали корінь з модуля, тоді знак мантиси: .

2.8.6 Форма запису нормалізованого результату з плаваючою комою в пам’ять

Зн.РZ PZ Зн.МZ MZ

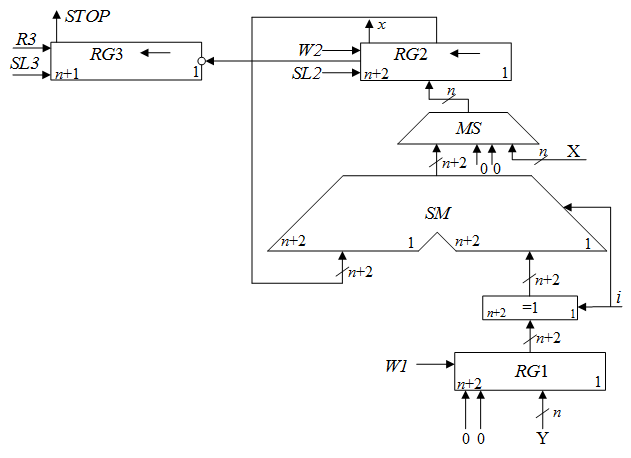
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 0. | 1 | 0 | 0 | 1 |

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

**3. Управляючий автомат Мура на тригерах**

x3x2x1 =100 – операція ділення першим способом.

3.1 Функціональна схема з відображенням управляючих сигналів

**

*Рисунок 3.1. Функціональна схема пристрою ділення другим способом*

3.2 Закодований мікроалгоритм

Початок

Кінець

*SL3; SL2*

*i, W2*

*W2*

*x*

*STOP*

1

1

*R3; W2; W1*

*Рисунок 3.2. Змістовний мікроалгоритм пристрою ділення першим способом*

За закодованим мікроалгоритмом складемо таблицю:

*Таблиця 3.1 Таблиця кодування сигналів*

|  |  |
| --- | --- |
| Сигнали операційного автомата | Сигнали управляючого автомата |
| *R3, W1* | *Y1* |
| *SL3, SL2* | *Y2* |
| *W2* | *Y3* |
| *i* | *Y4* |
| *x* | *X1* |
| *STOP* | *X2* |

Закодований мікроалгоритм з управляючими сигналами автомата

Z1

Початок

Кінець

*Y2*

*Y3; Y4*

*Y3*

*X1*

*X2*

1

1

*Y1; Y3*

Z2

Z3

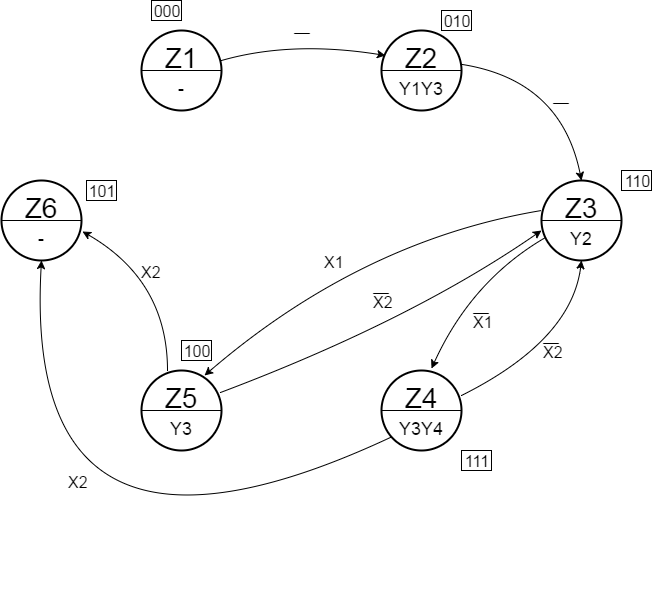
Z5

Z4

Z6

*Рисунок 3.3. Змістовний мікроалгоритм пристрою ділення першим способом з управляючими сигналами автомата*

3.3 Граф управляючого автомата Мура

**

*Рисунок 3.4 Граф автомата Мура*

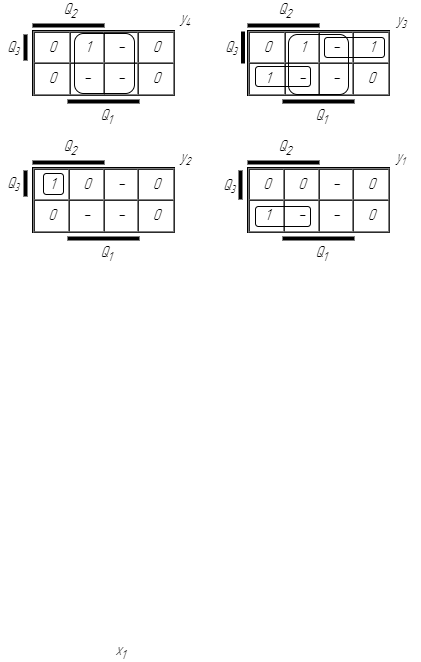
3.4 Побудова автомата

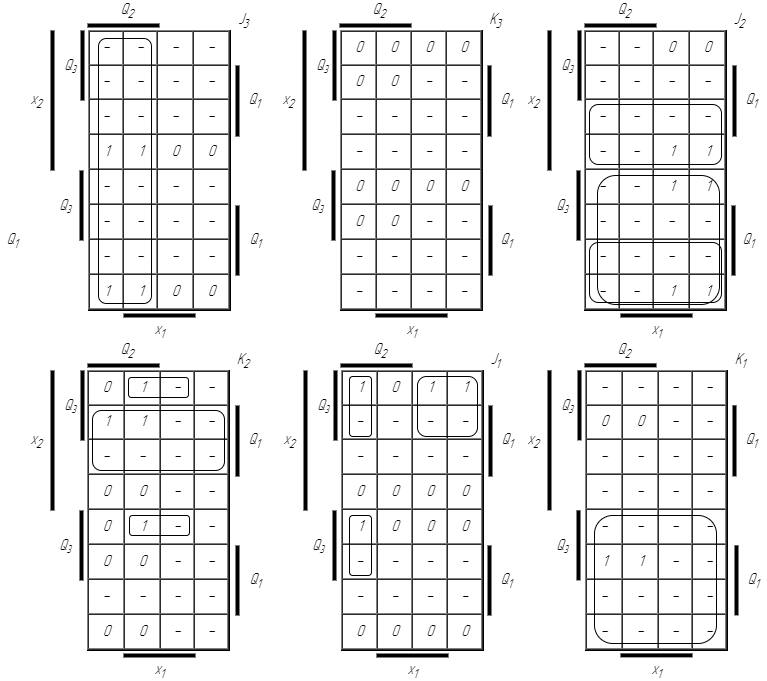
*Таблиця 3.2. Структурна таблиця автомата*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Перехід | Q3Q2Q1 | Q3Q2Q1 | x1x2 | y4y3y2y1 | J3K3 | J2K2 | J1K1 |
| *z1z2* | 000 | 010 | -- | 0000 | 0- | 1- | 0- |
| *z2 z3* | 010 | 110 | -- | 0101 | 1- | -0 | 0- |
| *z3 z4* | 110 | 111 | 0- | 0010 | -0 | -0 | 1- |
| *z3 z5* | 110 | 100 | 1- | 0010 | -0 | -1 | 0- |
| *z4 z3* | 111 | 110 | -0 | 1100 | -0 | -0 | -1 |
| *z4 z6* | 111 | 101 | -1 | 1100 | -0 | -1 | -0 |
| *z5 z3* | 100 | 110 | -0 | 0100 | -0 | 1- | 0- |
| *z5 z6* | 100 | 101 | -1 | 0100 | -0 | 0- | 1- |

JK:

Мінімізація за допомогою діаграм Вейча:





*Рисунок 3.5 Діаграми Вейча*

Y4 = Q1

Y3 = Q1∨Q3Q2∨Q3Q2

Y2 = Q3Q2Q1

Y1 = Q3Q2

J3 = Q2

K3 = 0

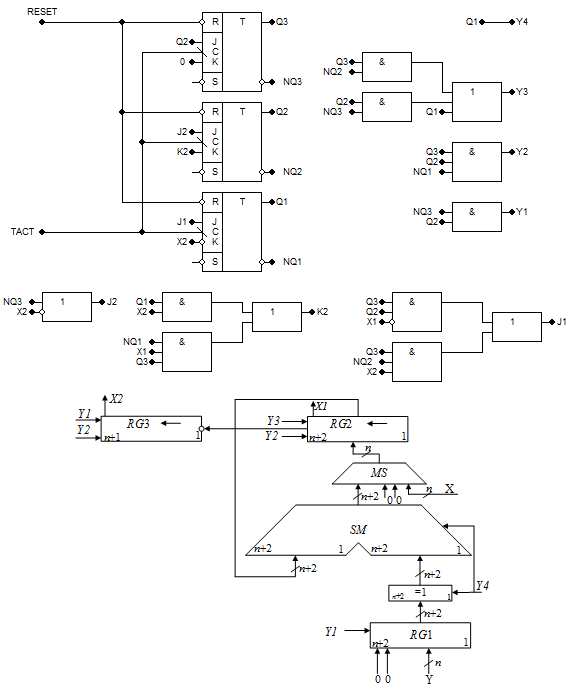
J2=Q3∨X2

K2 = Q1X2∨Q3Q1X1

J1 = Q3Q2X1∨Q3Q2X2

K1 = X2

Управляючий автомат:



*Рисунок 3.6 Управляючий автомат та пристрій для ділення другим способом*

**Висновок:** Таким чином, виконуючи дану розрахункову роботу, я повторив такі операції над числами в двійковому коді, як множення, ділення, додавання та знаходження кореня. Для кожної операції була подана операційна схема та змістовний мікроалгоритм. Був синтезований управляючий автомат для операційного пристрою ділення першим способом, згідно з варіантом, даний операційний пристрій був побудований на JK-тригерах.