# *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****"*

***Виконав:*** *Братун Андрій Юрійович*

***Факультет***  *ІОТ*

***Група*** *ІО-63* ***Залікова книжка №*** *6305*

*Керівник: Жабін В. І.*

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*(підпис керівника)*

***Київ - 2017р.***

Зміст

|  |  |  |
| --- | --- | --- |
| 1 | Завдання. | 3 |
| 2 | Обгрунтування індивідуального варіанту завдання. | 4 |
| 3 | Виконання індивідуального завдання. | 4 |
| 3.1. | Завдання 1 | 4 |
| 3.2. | Завдання 2 | 5 |
| 3.2.1. | Множення першим способом. | 5 |
| 3.2.2. | Множення другим способом. | 8 |
| 3.2.3. | Множення третім способом. | 10 |
| 3.2.4. | Множення четвертим способом. | 14 |
| 3.2.5. | Ділення першим способом. | 16 |
| 3.2.6. | Ділення другим способом. | 19 |
| 3.2.7. | Додавання чисел, поданих в формі з плаваючою комою. | 21 |
| 3.2.8. | Обчислення кореня. | 26 |
| 3.3. | Завдання 3 | 29 |
| 3.3.1. | Функціональна схема пристрою | 29 |
| 3.3.2. | Закодований (структурний) мікроалгоритм | 30 |
| 3.3.3. | Граф управляючого автомату Мілі | 30 |
| 3.3.4. | Функціональна схема автомату Мілі | 32 |
| 4. | Заключення. | 32 |
|  | Список використаної літератури | 34 |

1. **Завдання.**

1.1. 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).

1.2. 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 першим операндом є результат додавання).

Для обробки мантис кожної операції, подати:

1.2.1 теоретичне обґрунтування способу;

1.2.2 операційну схему;

1.2.3 змістовний (функціональний) мікроалгоритм;

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

1.2.5 обробку порядків (показати у довільній формі);

1.2.6 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.

1.3 Для операції з номером  додатково виконати:

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

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

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

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

1. **Обгрунтування індивідуального варіанту завдання.**

Номер залікової книжки:

630510 = 11000101000012 , отже всі задані в завдання операції будемо виконувати з числами:



і



Згідно п. 1.3 завдання для операції 001 (множення другим способом) додатково виконаємо п.п. 1.3.1 – 1.3.4. побудуємо автомат Мілі з використанням RS-тригерів.

1. **Виконання індивідуального завдання.**

**3.1.** **Завдання 1**.

Числа *X* і Y у формі з плаваючою комою у класичному варіанті (з незміщеним порядком і повною мантисою, порядок - 4 розряди, мантиса - 7 розрядів (з урахуванням знакових розрядів)) :

X:

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

Y:

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

Числа *X* і Y у формі з плаваючою комою за стандартом ANSI/IEEE 754-2008 в короткому 32-розрядному форматі ( мантиса зсувається вліво на 1 розряд, знак мантиси не змінюється, а порядок зменшується на 1 внаслідок зсуву і заміняється на зміщений порядок E = P+27 = 2­­m-1  + 10000000. Тоді Ex = 100+100000000 = 100000100 – зміщений порядок для *X,* Ey = 100+100000000=100000100 – зміщений порядок для *Y* ):

X:

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

Y:

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

**3.2. Завдання 2.**

**3.2.1. Множення першим способом.**

*Теоретичне обгрунтування.*

Вираз



можна представити у вигляді

 .

Звідси випливає, що отримані суми часткових добутків в *i*-му циклі (*i*= ) зводиться до обчислення



з початковими значеннями *i*=1, *Z*0=0, причому *Zn=Z=YX.*

Множення здійснюється з молодших розрядів множника, сума часткових добутків зсувається вправо, а множене залишається нерухомим.

*Операційна схема.*



*Рис. 3.1 Операційна схема пристрою*

*для множення чисел першим способом*

Під час множення *першим способом* (рис. 3.1) 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, при цьому 0 в n+1 розряді регістру RG1 відкидається.

Максимальний час множення, якщо не застосовуються методи прискорення операції, визначається виразом *t*м = *n*(*t*п + *t*з), де *t*пі *t*з– тривалості тактів підсумовування та зсуву відповідно.

*Змістовний мікроалгоритм.*



*Рис. 3.2 Змістовний мікроалгоритм множення чисел першим способом*

*Таблиця станів регістрів (лічильника).*

В регістр RG1 запишемо 0 – це добуток Z1=X\*Y, в регістр RG2 запишемо значення  , в RG3 – значення  , в СT – 110 (розрядність мантиси, без врахування знаку). Отже:

Px = 0 101

Mx=1 110001

Py = 0 101

My = 0 110100

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *Табл. 3.1 Таблиця станів регістрів (1-й спосіб множення)* | | | | | |
| № циклу | RG1 | RG2 | RG3 | CT | Мікрооперації |
| 0 | 0000000 | 110100 | 110001 | 110 | RG1:=0; RG2:=Y; RG3:=X;CT:=110 |
| 1 | 0000000 | 011010 | 110001 | 101 | RG1:=0.r(RG1); RG2:=RG[1].r(RG2);CT:=CT-1 |
| 2 | 0000000 | 001101 | 110001 | 100 | RG1:=0.r(RG1); RG2:=RG[1].r(RG2);CT:=CT-1 |
| 3 | 0000000  0110001  0110001  0011000 | 100110 | 110001 | 011 | RG1:=RG1+RG3;  RG1:=0.r(RG1); RG2:=RG[1].r(RG2);CT:=CT-1 |
| 4 | 0001100 | 010011 | 110001 | 010 | RG1:=0.r(RG1); RG2:=RG[1].r(RG2);CT:=CT-1 |
|  | 0001100  0110001  0111101  0011110 | 101001 | 110001 | 001 | RG1:=RG1+RG3;  RG1:=0.r(RG1); RG2:=RG[1].r(RG2);CT:=CT-1 |
| 5 | 0011110  0110001  1001111  0100111 | 110100 | 110001 | 000 | RG1:=RG1+RG3;  RG1:=0.r(RG1); RG2:=RG[1].r(RG2);CT:=CT-1 |

Результат множення: 100111110100, тобто можна записати:

Mz1=101000

*Обробка порядків.*

Pz1 = Px+Py = 0 101 + 0 101, тут виникає переповнення, збільшимо розрядність порядку, отримаємо:

Pz1 =0 0101 + 0 0101 = 0 1010

*Форма запису нормалізованого результату з плаваючою комою в пам'ять комп’ютера в прямому коді.*

Знак мантиси добутку визначимо шляхом додавання по модулю 2 знаків мантис множеного і множника:

1⊕0=1

Мантиса добутку нормалізовано, тому можна сформувати запис нормалізованого результату з плаваючою комою в пам'ять комп’ютера в прямому коді:

Z1:

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

**3.2.2. Множення другим способом.**

*Теоретичне обгрунтування.*

Запишемо



у вигляді



Очевидно, що процес множення може бути зведений до n-кратного виконання циклу



з початковими значеннями *i*=1, *Y*0*=Y*2*-n* , *Z0*=0. Множення здійснюється з молодших розрядів, множене зсувається вліво, а сума часткових добутків залишається нерухомою.

*Операційна схема.*



*Рис. 3.3 Операційна схема пристрою*

*для множення чисел другим способом*

Перед початком множення *другим способом* (рис. 3.3) 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.

*Змістовний мікроалгоритм.*



*Рис. 3.4 Змістовний мікроалгоритм множення чисел другимим способом*

*Таблиця станів регістрів (лічильника).*

В регістр RG1 запишемо 0 – це добуток Z2=Z1\*Y, в регістр RG2 запишемо значення  , в RG3 – значення  . Отже:

Pz1 = 0 1010

Mz1=1 101000

Py = 0 0101

My = 0 110100

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Табл. 3.2 Таблиця станів регістрів (2-й спосіб множення)* | | | | |
| № циклу | RG1 | RG2 | RG3 | Мікрооперації |
| 0 | 000000000000 | 101000 | 000000110100 | RG1:=0; RG2:=Z1; RG3:=0.Y |
| 1 | 000000000000 | 010100 | 000001101000 | RG2:=0.r(RG2); RG3:=l(RG2).0 |
| 2 | 000000000000 | 001010 | 000011010000 | RG2:=0.r(RG2); RG3:=l(RG2).0 |
| 3 | 000000000000 | 000101 | 000110100000 | RG2:=0.r(RG2); RG3:=l(RG2).0 |
| 4 | 000000000000  000110100000  000110100000 | 000010 | 001101000000 | RG1:=RG1+RG3;  RG2:=0.r(RG2); RG3:=l(RG2).0 |
| 5 | 000110100000 | 000001 | 011010000000 | RG2:=0.r(RG2); RG3:=l(RG2).0 |
| 6 | 000110100000  011010000000  100000100000 | 000000 | 110100000000 | RG1:=RG1+RG3;  RG2:=0.r(RG2); RG3:=l(RG2).0 |

Результат множення: 1000001000000, тобто можна записати:

Mz2=100001

*Обробка порядків.*

Pz2 = Pz1+Py = 0 1010 + 0 101 = 0 1111

*Форма запису нормалізованого результату з плаваючою комою в пам'ять комп’ютера в прямому коді.*

Знак мантиси добутку визначимо шляхом додавання по модулю 2 знаків мантис множеного і множника:

1⊕0=1

Мантиса добутку нормалізована, тому можна сформувати запис нормалізованого результату з плаваючою комою в пам'ять комп’ютера в прямому коді:

Z2:

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

**3.2.3. Множення третім способом.**

*Теоретичне обгрунтування.*

Представимо



у виді



Отже, суму часткових добутків у i-м циклі (i= ) можна одержати по формулі



Початковими значеннями є i=1, Z0=0. Множення здійснюється зі старших розрядів множника, сума часткових добутків зсувається вліво, а множене нерухоме.

*Операційна схема.*



*Рис. 3.5 Операційна схема пристрою*

*для множення чисел третім способом*

При множенні *третім способом* (рис. 3.5) 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 підсумування 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, при цьому молодший розряд відкидається, а старші – в регістрі *RG*2. Час множення третім способом визначається аналогічно першому способу.

*Змістовний мікроалгоритм.*



*Рис. 3.6 Змістовний мікроалгоритм множення чисел третім способом*

*Таблиця станів регістрів (лічильника).*

В регістр RG1 запишемо 0 – це добуток Z3=Z2\*Y, в старші розряди регістру RG2 запишемо значення  , в RG3 – значення  , а в СT значення 110 (розрядність мантиси без врахування розряду знака). Отже:

Pz2 = 0 1111

Mz2=1 100001

Py = 0 0101

My = 0 110100

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *Табл. 3.3 Таблиця станів регістрів (3-й спосіб множення)* | | | | | |
| № циклу | RG1 | RG2 | RG3 | CT | Мікрооперації |
| 0 | 000000 | 100001.0 | 110100 | 110 | RG1:=0; RG2:=Z2.0; RG3:=Y |
| 1 | 000000  110100  110100  101000 | 1000010  0000000  0  1000010  0000101 |  | 101 | *RG*1:=*RG*1+*RG*3; *RG*2:=*RG*2+0+CI  *RG*1:=*l*(*RG*1).0; *RG*2:= *l*(*RG*2).*RG*1[*n*] |
| 2 | 010000 | 0001011 |  | 100 | *RG*1:=*l*(*RG*1).0; *RG*2:= *l*(*RG*2).*RG*1[*n*] |
| 3 | 100000 | 0010110 |  | 011 | *RG*1:=*l*(*RG*1).0; *RG*2:= *l*(*RG*2).*RG*1[*n*] |
| 4 | 000000 | 0101101 |  | 010 | *RG*1:=*l*(*RG*1).0; *RG*2:= *l*(*RG*2).*RG*1[*n*] |
| 5 | 000000 | 1011010 |  | 001 | *RG*1:=*l*(*RG*1).0; *RG*2:= *l*(*RG*2).*RG*1[*n*] |
| 6 | 000000  110100  110100  10100.0 | 1011010  0000000  0  1011010  0110101 |  | 000 | *RG*1:=*RG*1+*RG*3; *RG*2:=*RG*2+0+CI  *RG*1:=*l*(*RG*1).0; *RG*2:= *l*(*RG*2).*RG*1[*n*] |

Результат множення: 0,011010110100

*Обробка порядків.*

Pz3 = Pz2+Py = 0 1111 + 0 101 = 0 10100

*Форма запису нормалізованого результату з плаваючою комою в пам'ять комп’ютера в прямому коді.*

Знак мантиси добутку визначимо шляхом додавання по модулю 2 знаків мантис множеного і множника:

1⊕0=1

Нормалізуємо мантису добутку, для цього виконаємо зсув вліво на один розряд, при цьому порядок зменшимо на 1:

Mz3=0,11010110100,

і округлимо число до 7 розрядів, враховуючи знаковий розряд:

Mz3=0,110110

Pz3 = 0 10100 – 1 = 0 10011

Сформуємо запис нормалізованого результату з плаваючою комою в пам'ять комп’ютера в прямому коді:

Z3:

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

**3.2.4. Множення четвертим способом.**

*Теоретичне обгрунтування.*



Процес множення може бути зведений до *n*-кратного виконання циклу



з початковими значеннями *i*=1, *Y*0=*Y*2-1, *Z*0=0.

Множення виконується зі старших розрядів множника, сума часткових добутків залишається нерухомою, а множене зсувається вправо.

*Операційна схема.*



*Рис. 3.7 Операційна схема пристрою*

*для множення чисел четвертим способом*

Перед множенням *четвертим способом* (рис. 3.7) 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*+1], 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. Результат множення буде в регістрі RG1, 0 в розряді RG1[1] відкидається. Час виконання множення четвертим способом складає *tм*=*nt*п, визначається аналогічно другому способу.

*Змістовний мікроалгоритм.*



*Рис. 3.8 Змістовний мікроалгоритм множення чисел четвертим способом*

*Таблиця станів регістрів (лічильника).*

В регістр RG1 запишемо 0 – це добуток Z4=Z3\*Y, в регістр RG2 запишемо значення  , в старші розряди регістру RG3 – значення мантиси  . Значення розрядів 2n+1 регістрів RG1 і RG3 встановлюються в 0. Отже:

Pz3 = 0 10011

Mz3=1 110110

Py = 0 0101

My = 0 110100

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Табл. 3.4 Таблиця станів регістрів (4-й спосіб множення)* | | | | |
| № циклу | RG1 | RG2 | RG3 | Мікрооперації |
| 0 | 0000000000000 | 110100 | 0110110000000 | RG1:=0; RG2:= Y; RG3:= Z3.0 |
| 1 | 0000000000000  0110110000000  0110110000000 | 101000 | 0011011000000 | RG1:=RG1+RG3;  RG2:=l(RG2).0; RG3:=0.r(RG3) |
| 2 | 0110110000000  0011011000000  1010001000000 | 010000 | 0001101100000 | RG1:=RG1+RG3;  RG2:=l(RG2).0; RG3:=0.r(RG3) |
| 3 | 1010001000000 | 100000 | 0000110110000 | RG2:=l(RG2).0; RG3:=0.r(RG3) |
| 4 | 1010001000000  0000110110000  1010111110000 | 000000 | 0000011011000 | RG1:=RG1+RG3;  RG2:=l(RG2).0; RG3:=0.r(RG3) |

Результат множення: 0,101011111000

*Обробка порядків.*

Pz4 = Pz3+Py = 0 10011 + 0 101 = 0 11000

*Форма запису нормалізованого результату з плаваючою комою в пам'ять комп’ютера в прямому коді.*

Знак мантиси добутку визначимо шляхом додавання по модулю 2 знаків мантис множеного і множника:

1⊕0=1

Мантиса добутку нормалізована, округлимо її до 7 розрядів, враховуючи розряд знаку і сформуємо запис нормалізованого результату з плаваючою комою в пам'ять комп’ютера в прямому коді:

Z4:

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

**3.2.5. Ділення першим способом.**

*Теоретичне обгрунтування.*

При реалізації ділення першим способом здійснюється зсув вліво залишку при нерухомому дільнику. На рис. 3.9 показана можлива побудова пристрою ділення. Чергова остача формується в регістрі *RG*2 (у вихідному стані в цьому регістрі записаний Z4). 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*

*Операційна схема.*



*Рис. 3.9 Операційна схема пристрою*

*для ділення чисел першим способом*

*Змістовний мікроалгоритм.*

*Початок*

*RG3:=1…11*

*RG1:=Y*

*RG2:=Z4*

*RG2[n+2]*

*1*

*0*

*RG2:=RG2++1*

*RG2:=l(RG2).0*

*RG3:=l.(RG3).*

*RG2:=RG2+RG1*

*RG2:=l(RG2).0*

*RG3:=l.(RG3).*

*1*

*RG3[n+1]*

*0*

*кінець*

*Рис. 3.10 Змістовний мікроалгоритм ділення чисел першим способом*

*Таблиця станів регістрів (лічильника).*

В регістр RG1 запишемо , в регістр RG2 запишемо значення мантиси , в регістрі RG3 буде формуватися результат ділення. Отже:

Pz4 = 0 11000

Mz4=1 101100

Py = 0 0101

My = 0 110100

Мантиси дільника і діленого нормалізовані, тому, для виконання умови:

Мділене < Мдільник

виконаємо зсув вправо мантиси діленого і збільшимо порядок діленого на 1:

Mz4=1 010110

Pz4 = 0 11000 + 1 = 0 11001

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Табл. 3.5 Таблиця станів регістрів (1-й спосіб ділення)* | | | | |
| № циклу | RG3 | RG2 | RG1 | Мікрооперації |
| 0 | 1 111111 | 00 010110 | 00 110100 | RG3:=1…11; RG1:= 00.Y; RG2:= 00.Z4 |
| 1 | 1 111110 | 00 010110  11 001011  1  11 100010  11 000100 |  |  |
| 2 | 1 111100 | 11 000100  00 110100  11 111000  11 110000 |  |  |
| 3 | 1 111001 | 11 110000  00 110100  00 100100  01 001000 |  |  |
| 4 | 1 110011 | 01 001000  11 001011  1  00 010100  00 101000 |  |  |
| 5 | 1 100110 | 00 101000  11 001011  1  11 110100  11 101000 |  |  |
| 6 | 1 001101 | 11 101000  00 110100  00 011100  00 111000 |  |  |
| 7 | 0 011011 | 00 111000  11 001011  1  00 000100  00 001000 |  |  |

Результат ділення: 0,011011

*Обробка порядків.*

Pz5 = Pz4-Py = 0 11001 - 0 101 = 0 10100

*Форма запису нормалізованого результату з плаваючою комою в пам'ять комп’ютера в прямому коді.*

Знак мантиси ділення визначимо шляхом додавання по модулю 2 знаків мантис діленого і дільника:

1⊕0=1

Нормалізуємо мантису частки, для чого зробимо зсув вліво мантису, а порядок зменшимо на 1:

Mz5=1 110110

Pz5= 0 10011

Z5:

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

**3.2.6. Ділення другим способом.**

*Теоретичне обгрунтування.*

При реалізації ділення другим способом (із зсувом дільника) збільшується розрядність регістрів *RG*1, *RG*3 і суматора *SM* (рис. 3.11). 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 .

*Операційна схема.*



*Рис. 3.11 Операційна схема пристрою*

*для ділення чисел другим способом*

*Змістовний мікроалгоритм.*



*Рис. 3.12 Змістовний мікроалгоритм ділення чисел другим способом*

*Таблиця станів регістрів (лічильника).*

В регістр RG1 запишемо , в регістр RG2 запишемо значення мантиси , в регістрі RG3 буде формуватися результат ділення. Отже:

Pz5 = 0 10011

Mz5=1 110110

Py = 0 0101

My = 0 110100

Мантиси дільника і діленого нормалізовані, тому, для виконання умови:

Мділене < Мдільник

виконаємо зсув вправо мантиси діленого і збільшимо порядок діленого на 1:

Mz5=1 011011

Pz5 = 0 10011 + 1 = 0 10100

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Табл. 3.6 Таблиця станів регістрів (2-й спосіб ділення)* | | | | |
| № циклу | RG3 | RG2 | RG1 | Мікрооперації |
| 0 | 1 111111 | 0 011011000000 | 0 110100000000 | *RG3:=1…11; RG1:= 0Y.0; RG2:= 0Z5.0* |
| 1 | 1 111110 | 0 011011000000  1 001100000000  1 100111000000 | 0 011010000000 | *RG1:=0.r(RG1); RG3:=l(RG3).SM(p)* |
| 2 | 1 111101 | 1 100111000000  0 011010000000  (1)0 000001000000 | 0 001101000000 | *RG1:=0.r(RG1); RG3:=l(RG3).SM(p)* |
| 3 | 1 111010 | 0 000001000000  1 110011000000  (0)1 110100000000 | 0 000110100000 | *RG1:=0.r(RG1); RG3:=l(RG3).SM(p)* |
| 4 | 1 110100 | 1 110100000000  0 000110100000  (0)1 111010100000 | 0 000011010000 | *RG1:=0.r(RG1); RG3:=l(RG3).SM(p)* |
| 5 | 1 101000 | 1 111010100000  0 000011010000  (0)1 111101110000 | 0 000001101000 | *RG1:=0.r(RG1); RG3:=l(RG3).SM(p)* |
| 6 | 1 010000 | 1 111101110000  0 000001101000  (0)1 111111011000 | 0 000000110100 | *RG1:=0.r(RG1); RG3:=l(RG3).SM(p)* |
| 7 | 0 100001 | 1 111111011000  0 000000110100  (1)0 000000001100 | 0 000000011010 | *RG1:=0.r(RG1); RG3:=l(RG3).SM(p)* |

Результат ділення: 0,100001

*Обробка порядків.*

Pz6 = Pz5-Py = 0 10100 - 0 101 = 0 01111

*Форма запису нормалізованого результату з плаваючою комою в пам'ять комп’ютера в прямому коді.*

Знак мантиси ділення визначимо шляхом додавання по модулю 2 знаків мантис діленого і дільника:

1⊕0=1

Мантиса добутку нормалізована, сформуємо запис нормалізованого результату з плаваючою комою в пам'ять комп’ютера в прямому коді:

Z6:

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

**3.2.7. Додавання чисел, поданих в формі з плаваючою комою.**

*Теоретичне обгрунтування.*

Суму двох чисел X=2PxMx і Y=2PyMy, поданих у форматі із плаваючою комою, можна записати у вигляді

2PxMx + 2PyMy = 2PzMz

У загальному випадку дійсні ваги однакових розрядів мантиси різних чисел із урахуванням порядків чисел можуть відрізнятись. Тому для додавання чисел із плаваючою комою необхідно привести їх до загального порядку *Рзаг*, яким зручно обрати більший порядок з двох доданків

*Рзаг = max(Px,Py)*

при цьому мантиса числа з меншим порядком зменшується за рахунок зсуву праворуч на кількість розрядів, що становить різницю значень між більшим і меншим порядками. Після цього суму двох чисел можна подати у вигляді

2РзагМх + 2РзагМ’y = 2Pзаг(Mx+M’y), де

за M’y прийнято перетворену мантису числа з меншим порядком.

Можна визначити такі етапи додавання чисел з плаваючою комою:

1. Зрівняння порядків.
2. Додавання мантис.
3. Нормалізація результату.
4. Формування результату.

Важливо зауважити, що після операції додавання мантис може виявитися нульовий результат, і процес нормалізації (зсуву мантиси) в цьому випадку буде нескінченним, тому кількість зсувів обмежується кількістю розрядів мантиси і результат операції подається машинним нулем.

Виконаємо етапи додавання для чисел:

Z7=Z6+Y

Z6=

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

Y=

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

1. Зрівняння порядків (числа подані в прямому коді)

Pріз=Pz6-Py= 0.01111пк-0.00101пк=0.01111пк+(-0.00101пк)=

=0.01111пк+1.11011дк

0.01111

1.11011

0.01010

Pріз=0.01010 – додатне число, це значить, що Pz6 більше за Py на число 1010 – кількість розрядів, на які потрібно виконати зсув вправо мантису числа Y. За Pзаг приймемо значення Pz6=0 01111.

Приведемо порядок і виконаємо перетворення мантиси числа Y:

P’y= Pзаг=0 01111

M’y=0 0000000000110100

Після округлення мантиси до 7 розрядів, враховуючи знаковий розряд отримаємо значення 0, тому спробуємо виконати етапи операції додавання для початкових заданих чисел X i Y. Маємо:

Z=X+Y

X:

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

Y:

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

1. Зрівняння порядків (числа подані в прямому коді)

Pріз=Px-Py= 0.101пк-0.101пк=0.101пк+(-0.101пк)=0.101пк+1.011дк=0.000

Pріз=0.000 – це значить, що Px = Py , тому Pзаг = 0 101, з мантисами обох чисел не потрібно виконувати перетворень.

2. Додавання мантис

Mz=Mx+My

Mx=1.110001пк, число від’ємне, переведемо в доповняльний код:

Mx=1.001111дк

My=0.110100пк

Додавання мантис виконаємо в модифікованих кодах для виявлення можливого переповнення розрядної сітки

|  |  |  |
| --- | --- | --- |
| Mx[дк] = | 11.001111 |  |
| My= + | 00.110100 |  |
| Mz= | 00.000011 |  |

3. Нормалізація результату.

*Теоретичне обгрунтування*

Після додавання мантиса результату може виявитися відразу нормалізованою або подана з *порушенням нормалізації вліво чи вправо*, тоді проводиться нормалізація результату.

Якщо під час додавання виникає переповнення розрядної сітки, то це *порушення нормалізації вліво*. Ознакою *порушення нормалізації вліво* є розбіжність у знакових розрядах модифікованих кодів чисел. Для приведення результату до нормалізованої форми необхідно зсунути мантису результату право на один розряд, а загальний порядок збільшити на одиницю.

Нормалізовані мантиси додатніх чисел у розряді з вагою 2-1 завжди мають одиницю. Нормалізовані мантиси від’ємних чисел, поданих в оберненому або доповняльному кодах, мають у цьому розряді нуль. Збіжність цифр у знаковому і старшому розряді мантиси у додатніх і від’ємних чисел говорить про *порушення нормалізації вправо*. Для приведення до нормалізованого виду мантиса отриманого результату зсувається вліво до появи одиниці або нуля (залежно від використовуваних кодів) у старшому розряді., а із загального порядку результату віднімається число одиниць, що дорівнює кількості зсувів мантиси.

*Операційна схема пристрою нормалізації результату операції додавання*

RG1

n 1

RG2

m 1

RG3

2 1

SM

1

=1

CT2

*Рис. 3.13 Операційна схема пристрою*

*для нормалізації результату додавання*

*Змістовний (функціональний) мікроалгоритм*

В регістр RG1 запишемо значення мантиси результату додавання мантис Mz. Розрядність регістру становить розрядність мантиси без врахування знакових розрядів. В регістр RG2 запишемо значення загального порядку Pзаг. Розрядність регістру – розрядність порядку без врахування знакового розряду. В регістр RG3 запишемо значення знакових розрядів модифікованого коду результату. Регістр складається з двох розрядів. Лічильник СТ1 відповідає кількості розрядів мантиси, лічильник СТ2 потрібен для підрахунку числа, яке необхідно додати до загального порядку після завершення нормалізації мантиси числа.

Початок

RG1:=Mz;

RG2:=Pz;

RG3:=ЗР2.ЗР1;

CT1:=n;

CT2:=0

RG3[2]=RG3[1]

RG1:=RG3[1].r(RG1);

RG2:=RG2+1;

RG3:=RG3[2].r(RG3)

RG3[1]=RG1[n]

RG1:=l(RG1).0;

CT1:=CT-1;

CT2:=CT2+1

RG2:=RG2++1

CT1=0

кінець

так

ні

так

ні

ні

так

*Рис. 3.14 Змістовний мікроалгоритм нормалізації результату додавання*

*Таблиця станів регістрів*

Після етапу додавання мантис отримали

Pзаг = 0 101

Mz=00.000011

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *Табл. 3.7 Таблиця станів регістрів (нормалізація результату додавання)* | | | | | | |
| № циклу | RG3 | RG2 | RG1 | СТ1 | СТ2 | Мікрооперації |
| 0 | 00 | 101 | 000011 | 110 | 000 | *RG1:= Mz; RG2:= Pзаг; RG3:=ЗР2.ЗР1* |
| 1 |  |  | 000110 | 101 | 001 | *RG1:=l(RG1).0; CT1:=CT1-1; CT2:=CT2+1* |
| 2 |  |  | 001100 | 100 | 010 | *RG1:=l(RG1).0; CT1:=CT1-1; CT2:=CT2+1* |
| 3 |  |  | 011000 | 011 | 011 | *RG1:=l(RG1).0; CT1:=CT1-1; CT2:=CT2+1* |
| 4 |  |  | 110000 | 010 | 100 | *RG1:=l(RG1).0; CT1:=CT1-1; CT2:=CT2+1* |
| 5 |  | 101  011  1  001 |  |  |  | *RG2:=RG2-CT2* |

*Етап формування результату додавання.*

Після нормалізації результату додавання мантис можна подати отриманий результат в формі з плаваючою комою.

Z:

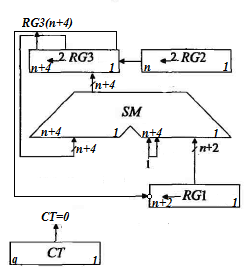
|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 1 |  | 0 | 1 | 1 | 0 | 0 | 0 | 0 |

**3.2.8. Обчислення кореня.**

Аргумент вводиться зі старших порядків. Порядок результату дорівнює поділеному на 2 порядку аргумента. Кількість розрядів мантиси повинна бути парною. Значення порядку числа повинно бути парним, в іншому випадку мантису числа зсувають на 1 розряд праворуч, а до порядку додають одиницю. Обчислення кореня виконується за наступними формулами:

Час виконання операції: T=n(tзсуву+ tsm)

*Операційна схема.*



*Рис. 3.15 Операційна схема пристрою для обчислення квадратного кореня*

*Змістовний мікроалгоритм.*

Початок

RG1:=0;

RG2:=Y;

RG3:=0;

CT:=n;

RG3[n+4]

0

1

RG3:=l2(RG3).RG2[n,n-1]

RG2:=l2(RG2).00

RG3:=RG3+.11

RG1:=l(RG1). 

CT:=CT-1

RG3:=RG3+.11

RG1:=l(RG1). 

CT:=CT-1

CT=0

Кінець

0

1

*Рис. 3.16 Змістовний мікроалгоритм обчислення кореня додатного числа*

*Таблиця станів регістрів (лічильника).*

В регістр RG1 та регістр RG3 запишемо нулі, а в регістр RG2 запишемо значення , поданого в формі з плаваючою комою. В регістрі RG1 отримаємо результат обчислення кореня числа Z8=.

Y:

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

Значення порядку числа непарне, тому значення мантиси зсунемо вправо на 1 розряд, а до значення порядку додамо одиницю, отримаємо:

Y:

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

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *Табл. 3.8 Таблиця станів регістрів та лічильника для обчислення кореня додатного числа* | | | | | |
| № циклу | RG1 | RG3 | RG2 | СТ | Мікрооперації |
| 0 | 00 000000 | 0000 000000 | 011010 | 110 | *RG1:=0; RG3:= 0; RG2:= Y* |
| 1 | 00 000001 | 0000 000001  1111 111111  0000 000000 | 101000 | 101 | *RG3:=2l(RG3).RG2[n,n-1];*  *RG2:=2l(RG2).00; RG3:=RG3+.11*  *RG1:=l(RG1).* |
| 2 | 00 000010 | 0000 000010  1111 111011  1111 111101 | 100000 | 100 | *RG3:=2l(RG3).RG2[n,n-1];*  *RG2:=2l(RG2).00; RG3:=RG3+.11*  *RG1:=l(RG1).* |
| 3 | 00 000101 | 1111 110110  0000 001011  0000 000001 | 000000 | 011 | *RG3:=2l(RG3).RG2[n,n-1];*  *RG2:=2l(RG2).00; RG3:=RG3+RG1.11*  *RG1:=l(RG1).* |
| 4 | 00 001010 | 0000 000100  1111 101011  1111 101111 | 000000 | 010 | *RG3:=2l(RG3).RG2[n,n-1];*  *RG2:=2l(RG2).00; RG3:=RG3+.11*  *RG1:=l(RG1).* |
| 5 | 00 010100 | 1110 111100  0000 101011  1111 100111 | 000000 | 001 | *RG3:=2l(RG3).RG2[n,n-1];*  *RG2:=2l(RG2).00; RG3:=RG3+RG1.11*  *RG1:=l(RG1).* |
| 6 | 00 101000 | 1110 011100  0001 010011  1111 101111 | 000000 | 000 | *RG3:=2l(RG3).RG2[n,n-1];*  *RG2:=2l(RG2).00; RG3:=RG3+RG1.11*  *RG1:=l(RG1).* |

Значення кореня числа Y: 0,101000

*Обробка порядків.*

Порядок результату отримується внаслідок ділення порядку числа на 2 (зсув порядку праворуч на 1 розряд):

Pz8=Py/2=0 110 / 2 = 0 011

*Форма запису нормалізованого результату з плаваючою комою в пам'ять комп’ютера в прямому коді.*

Мантиса значення кореня числа нормалізована, сформуємо запис нормалізованого результату з плаваючою комою в пам'ять комп’ютера в прямому коді:

Z8:

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

**3.3. Завдання 3.**

Варіант завдання, згідно з п.2 – 001, відповідно завдання 3 будемо виконувати для операції множення 2-м способом.

**3.3.1. Функціональна схема пристрою.**

*RG*2

*W*2

*SR*2

*n*

1

*X*

*x1*

*RG*1

*2n*

1

*W*1

*R*

*SM*

*2n*

1

1

*2n*

*RG*3

*W*3

*SR3*

*2n*

1

*0.Y*

*R*

*R*

*ЗРX*

*ЗРY*

*Знак*

*добутку*

*=1*

*1*

*n*

**

*RG2*

*x2*

*Рис. 3.17 Функціональна схема пристрою множення 2-м способом*

**3.3.2. Закодований (структурний) мікроалгоритм.**



*Рис. 3.18 Закодований (структурний) мікроалгоритм множення 2-м способом*

**3.3.3. Граф управляючого автомата Мілі.**

Перед побудовою графа приймемо:

y1 = *R,W1,W2,W3,ЗРX,ЗРY;*

y2 = W1;

y3 =SR3,SL3

і на закодованому мікроалгоритмі (рис. 3.18) позначимо стани автомату Мілі.

Складемо таблицю кодування станів автомату Мілі згідно з рис. 3.18

|  |  |  |
| --- | --- | --- |
| *Таблиця 3.9* | | |
| *Таблиця кодування станів автомата Мілі* | | |
| Стан | Код стану | |
| *Q*1 | *Q*2 |
| *a*1 | 0 | 0 |
| *a*2 | 0 | 1 |
| *a*3 | 1 | 1 |
| *a*4 | 1 | 0 |

-/y1

x1/y2

/-

-/y1

-/y3

x2/0

/-

/-

*Рис. 3.19 Граф автомату Мілі*

Складемо структурну таблицю автомату Мілі. Згідно варіанту завдання використаємо RS-тригери.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *Таблиця 3.10* | | | | | | | | | | | | | | |
| *Структурна таблиця автомата* | | | | | | | | | | | | | | |
| ПС | Код ПС | | НС | Код НС | | Логічні умови | | Керуючі  Сигнали | | | Функції  збудження  тригерів | | | |
| *Q*1*S* | *Q*2*S* |  | *Q*1*S*+1 | *Q*2*S*+1 | *x*1 | *x*2 | *y*1 | *y*2 | *y*3 | *R*1 | *S*1 | *R*2 | *S*2 |
| *a*1 | 0 | 0 | *a*2 | 0 | 1 | \* | \* | 1 | 0 | 0 | \* | 0 | 0 | 1 |
| *a*2 | 0 | 1 | *a*3 | 1 | 1 | 1 | \* | 0 | 1 | 0 | 0 | 1 | 0 | \* |
| *a*2 | 0 | 1 | *a*3 | 1 | 1 | 0 | \* | 0 | 0 | 0 | 0 | 1 | 0 | \* |
| *a*3 | 1 | 1 | *a*4 | 1 | 0 | \* | \* | 0 | 0 | 1 | 0 | \* | 1 | 0 |
| *a*4 | 1 | 0 | *a*3 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | \* | 0 | 1 |
| *a*4 | 1 | 0 | *a*4 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | \* | \* | 0 |
| *a*4 | 1 | 0 | *a*1 | 0 | 0 | \* | 1 | 0 | 0 | 0 | 1 | 0 | \* | 0 |

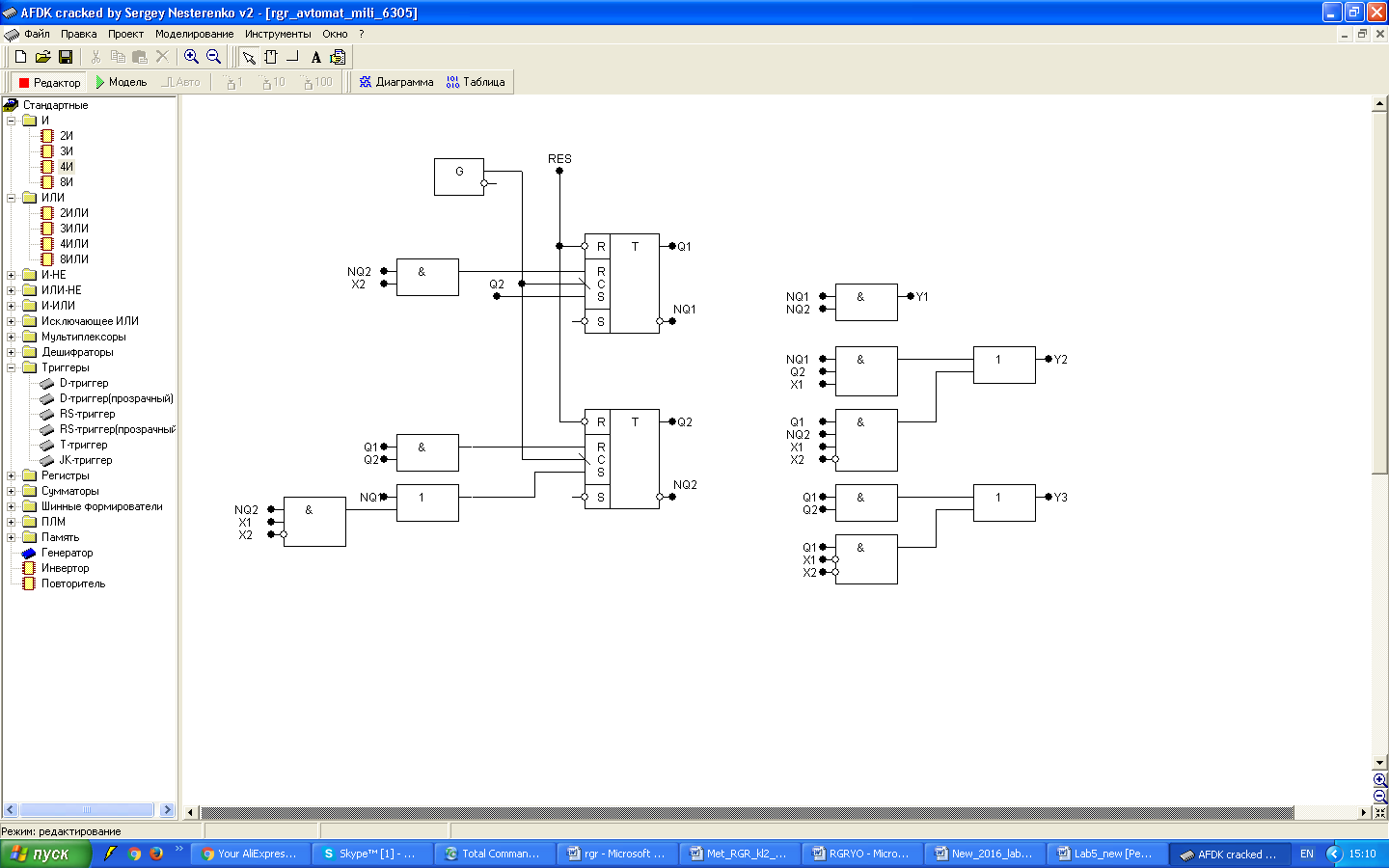
Виконавши процес мінімізації, використавши, наприклад, метод діаграм Вейча, отримаємо:

*y*1 = ; *y*2 = ; *y*3 = ;

*R*1 = ; *S*1 = ; *R*2 = ; *S*2 = 

**3.3.4. Функціональна схема автомата Мілі.**

Всі необхідні етапи для розробки функціональної схеми автомату Мілі були виконані. Отже, використовуючи елементи 2АБО, 2І, 3І та RS-тригери побудуємо схему автомату (рис. 3.20)



*Рис. 3.20 Функціональна схема автомату Мілі на RS-тригерах*

**4. Заключення.**

При виконанні цієї розрахункової графічної роботи вдосконалено та закріплено навички з виконання основних арифметичних операцій з двійковими числами з плаваючою комою, наведені приклади функціональних схем для цих операцій та виконані контрольні завдання з числами. На базі завдання розроблено та побудовано управляючий автомат Мілі на RS-тригерах для виконання арифметичної операції, в заданому варіанті – множення чисел 2-м способом.

Всі способи виконаня операцій, представлених в даній роботі є однаково ефективними, залежно від поставлених цілей: швидкодія чи простота пристрою. Підсумовуючи, можна сказати, що четвертий спосіб множення дозволяє виконувати множення найшвидше, тоді як третій є сумісним з апаратурою для операції добування кореня та ділення. При цьому найшвидшим способом множення є четвертий, а ділення – другий.

Задані алгоритми показали правильність своєї роботи. Проте, через малу розрядність сітки мантиси, деякі алгоритми показали невисоку точність результату. Також було подано детальну обробку порядків та нормалізацію результату, що є невід'ємними частинами виконання операцій з двійковими числами з плаваючою комою.

**Список використаної літератури.**

1. Жабін В.І., Жуков І.А., Клименко І.А., Ткаченко В.В. Прикладна теорія цифрових автоматів: Навч. посібник. – К.: Вид-во НАУ, 2009. – 364 с.

2. Жабін В.І., Жуков І.А., Клименко І.А., Стіренко С.Г. Арифметичні та управляючі пристрої цифрових ЕОМ. – К.: ВЕК+, 2008. – 176 с.

3. Самофалов К.Г. и др. Прикладная теория цифровых автоматов. - Киев: Вища школа, 1987. – 375 с.

4. Карцев М.А. Арифметика цифрових машин. – М.: Наука, 1969. – 578 с.

5. Самофалов К.Г., Корнейчук В.И., Тарасенко В.П. Цифровые ЭВМ. - Киев: Вища школа, 1989. – 424 с.