Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Комп'ютерна арифметика

Лабораторна робота №2

«ПРОЕКТУВАННЯ ТА ДОСЛІДЖЕННЯ ПРИСТРОЇВ

ДЛЯ МНОЖЕННЯ ЧИСЕЛ»

Виконав:

студент групи ІО-64

Бандурін В. Ю.

Залікова книжка №6402

Перевірив Верба О.А.

Київ

2017 р.

**Лабораторна робота №2**

|  |  |
| --- | --- |
| *Ціль роботи* – | вивчити методи реалізації операції множення чисел в прямих кодах, одержати навики в дослідженні операційних пристроїв. |

***Теоретичні відомості***

*Третій спосіб множення.*

Представимо (10.1) 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.

Принцип побудови пристроїв, що реалізують різні способи множення, показаний на рис. 2.1, де *RG*3 – регістр множеного, *RG*1 – регістр добутку, *RG*2 – регістр множника. Цифрами зазначені номери розрядів *SM* і регістрів, а стрілками показаний напрямок зсуву кодів у регістрах.



*а б*



*в г*

Рис. 2.1. Операційні схеми пристроїв для множення чисел: *а* – перший спосіб; *б* – другий спосіб; *в* – третій спосіб; *г* – четвертий спосіб

При множенні *третім способом* (рис. 2.1, в) 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.

*t*м = *n*(*t*п + *t*з), де *t*пі *t*з– тривалості тактів підсумовування та зсуву відповідно.

*Етапи побудови операційних пристроїв для множення чисел.*

1. Вивчити алгоритм множення чисел заданим методом.

2. Побудувати операційну схему пристрою.

3. Розробити змістовний (функціональний) мікроалгоритм з використанням операторів присвоєння та зсуву.

4. Виконати логічне моделювання роботи пристрою за допомогою таблиці станів вузлів (регістрів, лічильника) у кожному такті. Перевірити правильність вибору розрядності вузлів на операційній схемі.

5. Побудувати функціональну схему з відображенням управляючих сигналів для всіх вузлів.

6. Розробити структурний мікроалгоритм, в якому змістовні мікрооперації замінюються на сукупність управляючих сигналів, що забезпечують виконання мікрооперацій.

7. Побудувати і відлагодити схему в системі AFDK.

***Виконання роботи***

1. Варіант завдання визначається молодшими двійковими розрядами десяткового номера залікової книжки студента відповідно табл.1.

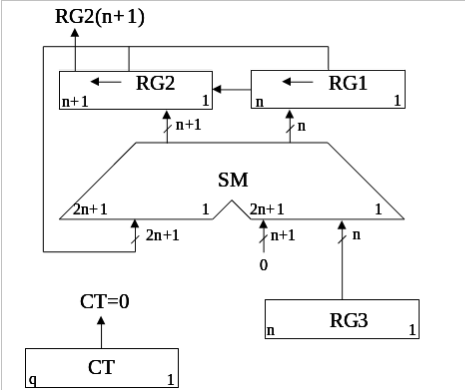
Варіант = 1100001000010, a1 = 0, a2 = 1, a3 = 0, a4 = 0, a5 = 0, a6 = 0.

Отже мій варіант

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ***a3*** | ***a*2** | ***a*1** | **Спосіб множення, розрядність операндів** | **Значення додатних операндів** | | | **Повна**  **операція** |
| ***X*** | ***Y*** | ***G*** |
| 0 | 1 | 0 | 3-й, 7 | ,1000001 | ,0100111 | ,1101011 | *F=XY*+*G* |

*Етапи побудови операційного пристрою для множення чисел.*

1. Операційна схема(рис.3.1).
2. Змістовний (функціональний) мікроалгоритм(рис 3.2).
3. Таблиця станів вузлів(табл. 3.1).
4. Функціональна схема(рис. 3.3).
5. Структурний мікроалгоритм(рис. 3.4).
6. Схема в системі ПРОГМОЛС-2 (AFDK)(рис 3.5).

Рис. 3.1 – операційна схема множення 3-м способом

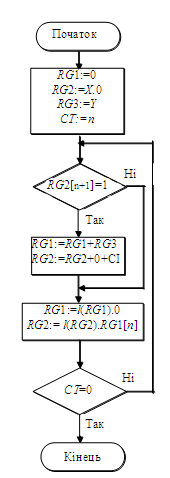
Рис. 3.2 – функціональний мікроалгоритм множення

Табл. 3.1. Стани регістрів і лічильника при множенні

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№**  **циклу** | ***RG*2** | ***RG*1** | ***RG*3** | ***CT*** | **Мікрооперації** |
| 0 | **1**000001.0 | 0000000 | 0100111 | 100 | *RG*1:=0; *RG*2:=*X*.0; *RG*3:=*Y*; *CT:=n* |
| 1 | +00000000  10000010  00000100 | +0100111  0100111  1001110 |  | 011 | *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 | 00001011 | 1010100 |  | 010 | *RG*1:=*l*(*RG*1).0; *RG*2:= *l*(*RG*2).*RG*1[*n*] |
| 3 | 00010111 | 0101000 |  | 010 | *RG*1:=*l*(*RG*1).0; *RG*2:= *l*(*RG*2).*RG*1[*n*] |
| 4 | 00101110 | 1010000 |  | 010 | *RG*1:=*l*(*RG*1).0; *RG*2:= *l*(*RG*2).*RG*1[*n*] |
| 5 | 01011101 | 0100000 |  | 010 | *RG*1:=*l*(*RG*1).0; *RG*2:= *l*(*RG*2).*RG*1[*n*] |
| 6 | 10111010 | 1000000 |  |  |  |
| 7 | 01110101 | +0100111  1100111  1001110 |  |  | *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*] |
| 8 | **01001111** | **00111**00 |  |  |  |

Результат



Рис. 3.3 - функціональна схема множення

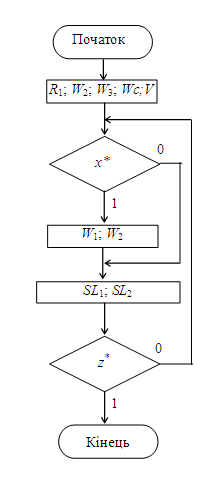


Рис. 3.4 – структурний мікроалгоритм

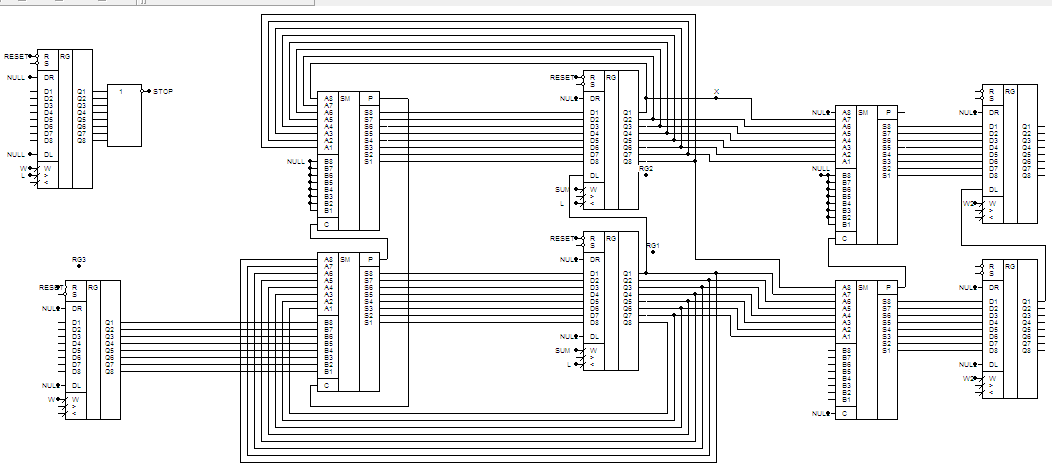


Рис. 3.5 – схема в системі AFDK

Висновок: В ході роботи я оволодів методами множення двійкових чисел. Під час виконання проблем не виникло. Схема працює вірно.