Курсовий проект на тему:

**«Розробка операційного та керуючого автомату, що виконує операцію прискореного множення»**

**Вступ**

# У наш час, з розвитком науково-технічного прогресу розвивається і обчислювальна техніка, що сприяє більшому удосконаленню. Обчислювальна техніка займає не останнє місце і грає визначальну роль в науковому технічному прогресі, сприяє підвищенню ефективності виробництва, покращанню якості продукції, росту продуктивності праці.

ЕОМ відкрили нову сторінку в історії людських знань і можливостей, звільнили тисячі обчислювачів, значно полегшили працю вчених, дали можливість вивчати найскладніші процеси.

В основу проектування операційних пристроїв різноманітного призначення покладено принцип функціонування мікропрограмування. Пристрої проектуються, як композиція операційного і керуючого автоматів. Мікропрограмування – це спосіб опису функцій операційних пристроїв незалежно від технічних засобів, які використовуються для їх реалізації. Таке тлумачення мікропрограми дозволяє синтезувати структуру будь-яких операційних пристроїв незалежно від способу керування роботою пристрою.

Необхідно відмітити, що принципи побудови і методи проектування операційних і керуючих автоматів є тією основою, на якій базується теорія і практика проектування більшої частини пристроїв ЕОМ.

**1. Розробка операційного автомату**

## **1.1 Метод виконання операції прискореного множення**

В даній курсовій роботі розглядається операція прискореного множення в доповняльному коді, зі старших розрядів.

Стосовно двійкової системи числення найбільш відомі наступні основні способи виконання операції множення: множення, починаючи з молодших розрядів множника; множення, починаючи з старших розрядів множника.

В обох випадках операція множення складається з ряду послідовних операцій додавання часткових добутків. Операціями додавання керують розряди множника: якщо в деякому розряді множника знаходиться одиниця, то до суми часткових добутків додається множене з відповідним зсувом, якщо в розряді множника – нуль, то множене не додається.

Якщо розрядність чисел значна то процес множення може потребувати значного часу, тому існують різні методи прискорення операції множення. Зокрема існують такі методи:

1) Прискорення за рахунок заміни комбінацій цифр. Наприклад: комбінації 00, 01 і 10 не перетворюються, а комбінація 11 замінюється на 01 і 1 йде в наступний розряд. При виконані множення, якщо в деякому розряді множника знаходиться 1, то до суми часткових добутків додається множене з відповідним зсувом, а якщо 1, то до суми часткових добутків додається доповняльний код множеного з відповідним зсувом.

2) Множення з запам’ятовуванням переносів: додаються три часткові добутки без розповсюдження переносу і формується проміжна сума і проміжний перенос; додаються без розповсюдження переносу проміжну суму, проміжний перенос та наступний частковий добуток, формується чергова проміжна сума і проміжний перенос; аналогічні дії виконуються до тих пір, поки не виконається додавання останнього часткового добутку; останній крок додавання з розповсюдженням переносу проміжної суми та проміжного переносу.

3) Прискорення множення при поділенні множника на дві частини: якщо множник має парну кількість розрядів, його ділять на дві рівні частини, відбувається множення множеного на число в два разу меншої розрядності ніж у В, отриманні добутки додаються з врахуванням зсуву.

В даній курсовій роботі будемо використовувати метод множення з запам’ятовуванням переносів, розглянемо його більш детально:

Множимо числа з фіксованою комою в доповняльному коді. Множення виконується з врахуванням знакового розряду, якщо В>0 то виконуємо множення, якщо В<0 то робимо корегуючі дії: додаємо доповнений код числа А. Додаємо також ще два часткові добутки (з врахуванням зсуву) без розповсюдження переносу і формуємо проміжну суму і проміжний перенос.

Потім додаємо без розповсюдження переносу проміжну суму, проміжний перенос та наступний частковий добуток (з врахуванням зсуву) і формуємо чергову проміжну суму і проміжний перенос. Ці дії виконуємо до тих пір, поки не додамо останній частковий добуток.

Останній крок: додавання з розповсюдженням переносу проміжної суми та проміжного переносу.

Отриманий результат теж буде в доповняльному коді, тому дивимося на знак результату: якщо результат додатній, то його прямий код співпадає з оберненим і не потрібно переводити, якщо ж результат від’ємний, то доповняльний код необхідно перевести у прямий, для цього інвертуємо результат і додаємо до нього одиницю, отримаємо прямий код результату.

## **1.2 Розробка алгоритму**

### Початок

СМ:= 0;

Рг1:=0;

Рг2:=0;

Рг3:=0;

##### РгВ:=Вдоп;

##### РгА[0:8]:=Aдоп;

##### n:=8;

РгВ[0]=1

#### Рг1:=Адоп

0

РгА:=R1[РгA]

РгВ[1]=1

#### Рг2:=Адоп

РгА:=R1[РгA]

РгВ[2]=1

#### Рг3:=Адоп

0

РгА:=R1[РгA]

**1.**

1

1

1

Згідно обраного методу виконання множення побудуємо блок схему алгоритму:

0

### Кінець

0

**1.**

CM:=Рг1+Рг2+Рг3

Формуемо проміжну суму і проміжний перенос Р

РгВ:=L2[РгВ]

n:=n-2

k = n

1

РгВ[1]=1

CM:=CМ+Р+РгА

Формуемо проміжну суму і проміжний перенос Р

0

CM:=СМ+Р

Формуемо проміжну суму і проміжний перенос Р

РгА:=R1[РгA]

РгВ:=L1[РгВ]

n:=n-1

1

СМ:=СМ+Р

1

РгВ[1]=1

0

СМ:=СМ+1

**Описання блок-схеми алгоритму**

Спочатку виконуємо ініціалізацію:

* в суматор заносимо 0;
* в регістр Рг1 заносимо 0;
* в регістр Рг2 заносимо 0;
* в регістр Рг3 заносимо 0;
* в лічильник n заносимо 8;
* у перші 8 розрядів регістра РгА заносимо доповняльний код числа А;
* в регістр РгВ заносимо доповняльний код числа В;

Перевіряємо знак числа В (РгВ[0]), якщо він рівний 1 (від’ємне) то в

регістр Рг1 заносимо доповняльний код числа А. Зсуваємо регістр РгА вправо на один розряд. Дивимося на старший розряд числа В (РгВ[1]), якщо там 1 то заносимо в регістр Рг2 значення регістра РгА (першій частковий добуток). Зсуваємо регістр РгА вправо на один розряд. Дивимося на другий розряд числа В (РгВ[2]), якщо там 1 то заносимо в регістр Рг3 значення регістра РгА (другий частковий добуток). Зсуваємо регістр РгА вправо на один розряд. Додаємо в суматорі значення з регіс-трів Рг1, Рг2, Рг3 та формуємо проміжну суму та проміжний перенос Р. Зсуваємо регістр РгВ вліво на два розряди. Зменшуємо значення лічильника на 2.

Далі починаємо цикл:

Поки n не дорівнює 0 дивимося на старший розряд числа В (РгВ[1]), якщо там одиниця додаємо до суматора (попередня проміжної суми) вміст регістра РгА (черговий частковий добуток) та попередній проміжний перенос Р, якщо ж 0, то додаємо лише попередній проміжний перенос Р; формуємо чергову проміжну суму та проміжний перенос. Зсуваємо регістр РгА вправо на один розряд. Зсуваємо регістр РгВ вліво на один розряд. Зменшуємо значення лічильника на 1. Повторюємо цикл.

Після закінчення циклу додаємо до суматора (останньої проміжної суми), з розповсюдженням переносу, останній проміжний перенос Р. Дивимося на знак результату (СМ[0]), якщо там 1 (результат від’ємний), перетворюємо доповняльний код в прямий: інвертуємо суматор і додаємо 1. Операцію множення виконано результат знаходиться в суматорі.

**1.3 Приклад множення**

Візьмемо для прикладу помноження два числа:

A=0,69140625 та B= -0,80078125**.**

## Переведемо ці числа в двійкову систему числення:

А2=0.10110001; Адоп=0.10110001;

В2=1.11001101; Вдоп=1.00110011.

Розглянемо приклад помноження цих чисел, за допомогою алгоритму:

|  |  |  |
| --- | --- | --- |
| СМ | РгB | ПРИМІТКИ |
| 0.0000000000000000  1.01001111  +0.000000000  0.0000000000 | 1.00110011  1.110011\_ \_  1.10011\_ \_ \_  1.0011\_ \_ \_ \_  1.011\_ \_ \_ \_ \_  1.11\_ \_ \_ \_ \_ \_  1.1\_ \_ \_ \_ \_ \_ \_ | СM:=0;  Рг1:=Адоп;  СМ:=Рг1+Рг2+Рг3; |
| 1.0100111100000000 +0.0000000000000000  0.00010110001 | форм. пром. суму і пер. Р  РгВ←2; РгА→1;  СМ:=СМ+РгА+Р; |
| 1.0101100100100000  +0.0000110000000000  0.000010110001 | форм. пром. суму і пер. Р  РгВ←1; РгА→1;  СМ:=СМ+РгА+Р; |
| 1.0101111000110000  +0.0001001000000000  0.0000000000000 | форм. пром. суму і пер. Р  РгВ←1; РгА→1;  СМ:=СМ+Р; |
| 1.0100110000110000  +0.0010010000000000  0.00000000000000 | форм. пром. суму і пер. Р  РгВ←1; РгА→1;  СМ:=СМ+Р; |
| 1.0110100000110000  +0.0000100000000000  0.000000010110001 | форм. пром. суму і пер. Р  РгВ←1; РгА→1;  СМ:=СМ+РгА+Р; |
| 1.0110000101010010  +0.0001000001000000  0.0000000010110001 | форм. пром. суму і пер. Р  РгВ←1; РгА→1;  СМ:=СМ+РгА+Р; |
| 1.0111000110100011  +0.0000000010100000 | СМ:=СМ+Р; |
| 1.0111001001000011  1.1000110110111101 | СМ:=СМ+1; |

В результаті отримали відповідь:

А\*В=1.10001101101111012= – 0,553665161133;

**2. Синтез керуючого автомату**

**2.1 Теоретичні відомості**

Як такого конкретного визначення автомату не має, цей термін використовується в двох аспектах: автомат – як пристрій, виконуючий деякі функції, без участі людини; з другого боку, автомат як математичне поняття – математична модель реальних технічних автоматів.

Автомат називається скінченим, якщо множина його внутрішнього стану та множина значень вхідних сигналів – скінчена множина. В практиці часто використовується поняття цифрового автомату, під яким сприймають пристрій, призначений для перетворення цифрової інформації.

Автомат задається трьома алфавітами і двома функціями (функція переходів та функція виходів), одним початковим станом. Поняття стану автомату використовується для описання систем, виходи яких залежать не тільки від вхідних сигналів в даний момент часу, але і від деяких сигналів, які поступили на входи системи раніше. Функція переходів – це залежність нового стану від попереднього та вхідних сигналів. Функція виходів – залежність вихідного сигналу від вхідного та попереднього стану.

Закон функціонування управляючого автомату можна описати у вигляді списку переходів автомата. Так, закон функціонування автомата можна представити у вигляді таблиці з такими розділами: номер переходу, вихідний стан, його код, наступний стан, його код, вхідний набір, вихідний набір, сигнали збудження. Цей список переходів дозволяє компактно і наочно зобразити закон функціонування автоматів. Перемикання автомата з одного стана в інший виконується шляхом зміни стану запам’ятовуючих елементів, які переключають сигнали збудження.

Керуючі автомати з жорсткою логікою будуються на базі логічних й запам’ятовуючих елементів, які об’єднуються в логічну схему, яка функціонує в відповідності з заданим законом.

Розрізняється два типи автоматів: Мура і Мілі. Мура простіший в розумінні, Мілі – складніший, при реалізації – навпаки. У Мура Вихідні сигнали пов’язані тільки зі станами, для Мілі вихідні сигнали залежать як від станів, так і від вхідних сигналів.

Коли графік програми позначають станами, то для Мура станами позначають операційні вершини, для Мілі – зв’язки між операційними вершинами так, щоб витратити якомога менше станів, причому, щоб кожна операційна вершина знаходилась між двома станами, і між двома операційними вершинами був стан.

**2.2 Розробка алгоритму роботи автомату**

### Початок

Y1

a1

X1

#### Y2

0

#### Y3

#### X2

#### Y4

#### Y3

#### X3

#### Y5

0

#### Y3

**1.**

1

1

1

0

a3

a2

a6

a7

a5

a4

Перейдемо від блок-схеми алгоритму до блок-схеми автомата позначивши станами зв’язки між операційними вершинами так, щоб витратити якомога менше станів, причому, щоб кожна операційна вершина знаходилась між двома станами, і між двома операційними вершинами був стан.

### Кінець

0

**1.**

#### Y6

#### Y7

#### Y8

#### X4

1

#### X2

#### Y9

0

#### Y10

#### Y3

#### Y11

#### Y12

1

#### Y13

1

#### X5

0

#### Y14

#### Y15

a10

a1

a9

a8

Позначаємо через X умовні вершини, якщо зустрічаються однакові умовні вершини, то і їх позначаємо однаково. Отримали всього 5 різних умовних вершин.

Через Y позначимо операції, якщо зустрічаються однакові операції, то вони теж позначаються однаково. Якщо між операціями немає умовної вершини то їх об’єднуємо в одну операційну вершину і між ними не ставиться стан. Кількість розставлених станів дорівнює 10, для їх кодування використаємо 4 двійкових розрядів.

**2.3 Побудова графа автомата**

Наступним кроком у синтезі керуючого автомату є перехід до графу автомата. Якщо станам поставити в відповідність вершини графа, а шляхам переходу від одного стану до іншого через умовні та операційні вершини – дуги (з цієї вершини в наступну), які відмічені набором значень вхідних та вихідних сигналів, то отримаємо граф який буде визначати закон функціонування автомата Мілі. Відмітимо, що дуги на графі автомата відмічаються тільки тими вхідними сигналами, які визначають можливість переходу між станами, і тими вихідними сигналами, які в даній ситуації приймають відповідне значення.

a10

a1

a3

a2

a4

a5

a6

a7

a9

a8

Y3

Y1

Х1Y2

Х1Y3

Y3

Х2Y4

Х2Y3

Х3Y15

Х3Y14

Х4Y13

Х3Y5

Х4 Х2Y9

Х3 Y3Y6Y7Y8

Y3Y6Y7Y8

Y3Y11Y7

Х4 Х2Y9

**2.4 Побудова таблиці функціонування**

Таблиця функціонування – це таблиця переходів. В ній кожна стрічка це перехід між станами автомата, де вказується попередній стан, його код, наступний стан і його код, вхідні сигнали, які викликають цей перехід, вихідні сигнали, які пов’язані з цим переходом, сигнали збудження входів запам’ятовуючих елементів (тригерів).

Автомат вказаний на графі має 10 станів, для їх кодування використовуємо 4 двійкових розряди, тому потрібно 4 тригера. Закодуємо стани автомата таким чином: а1 = 0001, а2 = 0010, а3 = 0011, а4 =0100, а5 = 0101, а6 = 0110, а7= 0111, а8 = 1000, а9 = 1001, а10 = 1001.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Попередній  стан | Код  **Т4Т3Т2Т1** | Наступний  стан | Код  **Т4Т3Т2Т1** | Вхідні  сигнали | Вихідні  сигнали |
| a10 | 1010 | a1 | 0001 | X5  X5 | Y14  Y15 |
| a1 | 0001 | a2 | 0010 |  | Y1 |
| a2 | 0010 | a3 | 0011 | X1 | Y2 |
| a2  a3 | 0010  0011 | a4 | 0100 | X1 | Y3  Y3 |
| a4 | 0100 | a5 | 0101 | X2 | Y4 |
| a4  a5 | 0100  0101 | a6 | 0110 | X2 | Y3  Y3 |
| a6 | 0110 | a7 | 0111 | X3 | Y5 |
| a6  a7  a9 | 0110  0111  1001 | a8 | 1000 | X3 | Y3 Y6 Y7 Y8  Y3 Y6 Y7 Y8  Y3 Y11 Y12 |
| a8 | 1000 | a9 | 1001 | X4 X2  X4 X2 | Y9  Y10 |
| a8 | 1000 | a10 | 1010 | X4 | Y13 |

**2.5 Синтез схеми керуючого автомату**

На основі таблиці функціонування складаємо рівняння збудження тригерів:

Для вхідних сигналів:

DT1 = a10 + a2 x1 + a4 x2 + a6x3 + a8x4;

DT2 = a1 + a2 x1 + a4 x2+ a5 + a6x3 + a8 x4;

DT3 = a2 x1 + a3 + a4 +a5 + a6x3;

DT4 = a7 + a6x3 + a9+ a8;

Для вихідних сигналів:

y1 = a1;

y2 = a2x1;

y3 = a2x1+a3+a4x2+a5+a6x3+a7+a9;

y4 = a4x2;

y5 = a6x3;

y6 =a6x3+a7;

y7 = a6x3+a7;

y8 = a6x3+a7;

y9 = a8x4x2;

y10= a8x4x2;

y11= a9;

y12= a9;

y13= a8x4;

y14= a10x5;

y15= a10x5.

На основі цих рівнянь синтезуємо схему керуючого автомату. Для зручності побудови використаємо шину. Оскільки автомат має 10 станів, для кодування яких потрібно 4 двійкових розряди то для побудови схеми використаємо 4 D-тригера з керуючими тактовими входами.

**2.6 Побудова структурної схеми автомату**

Побудуємо також структурну схему автомату:

**0 1 РГ А** 1 **16**

**0 1 РГ B** 21 **8**

**0 1 СМ 16**

**0 1 РГ 3 16**

**0 1 РГ 2 16**

**0 1 РГ1 16**

#### 1 n 8

**Y1**

**Y1**

**Х1**

**Адоп**

**Вдоп**

**Y1**

**Y2**

**Y4**

**Y5**

**Y3**

**Y6**

**Y1**

**Y8**

**Y12**

**Y9**

**Y13**

**Y10**

**Y11**

**Y7**

**Y14**

**Х2**

**Х3**

**Х5**

**Висновок**

# В курсовій роботі було розроблено алгоритм та синтезовано автомат Мілі для виконання операція прискореного множення чисел в доповняльному коді, з фіксованою комою, зі старших розрядів.

У першому розділ курсової роботи було розроблено алгоритм виконання вказаної операції над числами.

Друга частина – присвячена розробці автомата Мілі: побудова граф схеми автомата, побудова таблиці переходів, синтез керуючого автомата і побудова структурної схеми автомата.

**Використана література**

1. А.Я. Савльев «Прикладна теория цифрових автоматів» – Москва В. Шк. – 1987.

2. Методичні вказівки до вивчення курсу «Прикладна теорія цифрових автоматів» – Вінниця ВДТУ-1997.

3. Каган Б.М. «Электронные вычислительные машины и системы». Москва Энергоатомиздат 1991 г. с. 592

4. Методические указания к практическим занятиям по курсу «Теория и проектирование ЦВМ» – Винница ВПИ - 1982.

5. К.Г. Самофалов «Цифровые ЭВМ» – ВШ – 1989.