Національний технічний університет України

«Київський політехнічний інститут»

Факультет інформатики і обчислювальної техніки

Кафедра обчислювальної техніки

Лабораторна робота №2

з дисципліни «Архітектура комп’ютерів»

*Виконав:*

Студент групи ІО-31

Долинний О.В.

*Перевірив:*

Чеснішний І.А.

м. Київ

2015 р.

1. **Варіант завдання:**

№ ЗК: 311010=110 000 100 1102

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *a6* | *a5* | *a4* | Спосіб множення | Розрядність операндів |
| 0 | 0 | 0 | Перший | 8 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *a6* | *a3* | *a2* | Значення операндів | |
| 1 | 1 | 1 | *4ch* | *48h* |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| a4 | a2 | Спосіб  адресації мікрокоманд | Ємність  ПМК, слова | Використати зону β4 для перевірки слова МК |
| 0 | 1 | примусовий | 32 | на парність |

|  |  |  |  |
| --- | --- | --- | --- |
| a6 | a5 | a4 | Тривалість мікрооперації підсумовування, такти |
| 1 | 0 | 0 | 11 |
| Інші мікрооперації виконуються за один такт | | | |

1. **Короткі теоретичні відомості:**

БМУ з примусовою адресацією

За примусової адресації зона *β*1 має наступний формат:

,

де М – поле управління мультиплексором;

q – довжина поля управління мультиплексором;

К – константа, що визначає адресу наступної мікрокоманди;

n – розрядність адреси мікрокоманди.

Довжина поля управління мультиплексором визначається за формулою: *q*= ]log2(*k*+2) де *k* – кількість зовнішніх умов.

Поле константи *К* являє собою (*n*–1) старших розрядів адреси мікрокоманди.

|  |  |
| --- | --- |
| Формат адреси мікрокоманди має наступний вигляд:  де *α*– визначає умову переходу, яка формується на виході мультиплексора в залежності від логічних умов *Хi*. |  |

Спрощена структурна схема БМУ з примусовою адресацією зображена на рис. 1. На цій та подальших схемах БМУ входи для занесення початкової адреси *D* в РАМК умовно не показані.



##### Рис. 1. Структурна схема БМУ з примусовою адресацією

1. **Виконання завдання**

## **3.1 Розробка алгоритму**

Операційна схема, що реалізує перший спосіб множення, подана на рис. 2, де *RG*1 – регістр накопичення суми часткових добутків, *RG*2 – регістр множника, *RG*3 – регістр множеного, *RG*4 (*СТ*) – лічильник циклів, *SM* – комбінаційний суматор. *DR* – вхід заповнення старшого розряду при зсуві вправо. Регістри *RG*1 та *R*G2 реалізують мікрооперації зсуву, лічильник *RG*4 дозволяє формувати ознаку нуля. За нульовим вмістом регістру *RG*4 результат обчислення є сформованим в регістрах *RG*1 та *R*G2.

|  |  |
| --- | --- |
| Рис. 2. Операційна схема множення | Рис. 4. Ф-мікроалгоритм |
| Рис. 3. Функціональна схема операційного пристрою |

|  |  |
| --- | --- |
| Для розробленої операційної схеми побудуємо Ф-мікроалгоритм (рис. 4), де *RG*2*(n) –* значення молодшого розряду регістру *RG*2.  На підставі операційної схеми множення та Ф-мікроалгоритму складемо перелік керівних сигналів для всіх функціональних вузлів ОПр та побудуємо функціональну схему.  Функціональна схема ОПр зображена на рис. 3. За функціональною схемою будуємо структурний мікроалгоритм (С-мікроалгоритм), що зображений на рис 5. Індекс указує, до якої з функціональних частин пристрою належить керівний сигнал. Кодування керівних сигналів та логічних умов наведене в табл. 1-2.  Закодований С-мікроалгоритм зображений на рис.5, де керівні сигнали та сигнали логічних умов відповідають рис 4 та табл. 1-2. Сигнали, що завжди формуються разом, кодуються одним символом. Цьому символу відповідає один вихід пристрою управління.  Порожня вершина 5 введена для запобігання перетинання у часі сигналів і , що подаються на *RG*1. Це необхідно для правильного виконання послідовності мікрооперацій на одному регістрі. Для забезпечення перепаду сигналів управління *SR1, SR2, dec* необхідно ввести порожню додаткову вершину 8. Ця вершина забезпечує також вірну оцінку стану лічильника після декрементації в наступному такті, коли новий стан лічильника вже буде встановлено. | Рис. 5. Закодований структурний мікроалгоритм |

|  |  |  |
| --- | --- | --- |
| *Таблиця 1.* Кодування керівних сигналів | | *Таблиця 2.* Кодування логічних умов |
| Керівні сигнали | Код | |  |  | | --- | --- | | Логічні умови | Код | | Значення молодшого розряду множника | TC | | Нульовий вміст лічильника | z | |
| *RG*1:=0  *RG2*:=*X*(7..0)  *RG3*:=*Y*(7..0)  *RG4*:=8  *TC*:=0 | y1 |
| *RG*1:=*RG*1+*RG3* | y2 |
| *RG*1:=0.*r* [*RG*1]  *RG*2:=*RG*1(0).*r* [*RG*2]  *RG4*:=*RG4*–1  *TC*:=*RG2*(0) | y3 |

Одержаний закодований С-микроалгоритм є вихідним для здійснення синтезу БМУ. Керівні сигнали з виходів БМУ підмикаються до входів відповідних функціональних вузлів операційного пристрою.

## **3.2 Розробка структури БМУ та її обґрунтування**

Визначимо формат зони *β*1:

; ;

*; .*

Визначимо спосіб управління мультиплексором (табл. 3).

*Таблиця 3.* Кодування поля *М*

|  |  |
| --- | --- |
| *m*2 *m*1 | УС |
| 00  01  10  11 | 0  *ТС*  *z*  1 |

Визначимо формат зони *β*2. Використовуємо горизонтальне кодування.

*.*

Розрахуємо довжину зони *β*3:

;

*.*

Для перевірки на парність у зоні *β*4 необхідно виділити один розряд.

## **3.3. Визначення формату мікрокоманди**

Отримаємо наступний формат мікрокоманди ():



Розміщуємо мікрокоманди в пам’яті мікрокоманд (рис. 6).



* + - * 1. Рис. 6. Розміщення мікрокоманд в ПМК

Карта програмування БМУ наведена у табл. 4.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *Таблиця 4.* Карта програмування БМУ | | | | | | | |  |
| № МК | Адреса | β1 | | β2 | β3 | | β4 | |
| k | m | y3 y2 y1 | ЗР |  |
| 1(П)  2  3  4  5  6  7(К)  8 | 00000  00001  00010  00011  00100  00101  00110  00111 | 0000  0010  0001  0001  0011  0011  0010  0010 | 11  00  01  00  00  11  11  10 | 000  001  000  010  000  100  000  000 | 0  0  0  1  0  0  0  0 | 0000  0000  0000  0110  0000  0000  0000  0000 | 0  0  0  1  0  1  1  0 | |

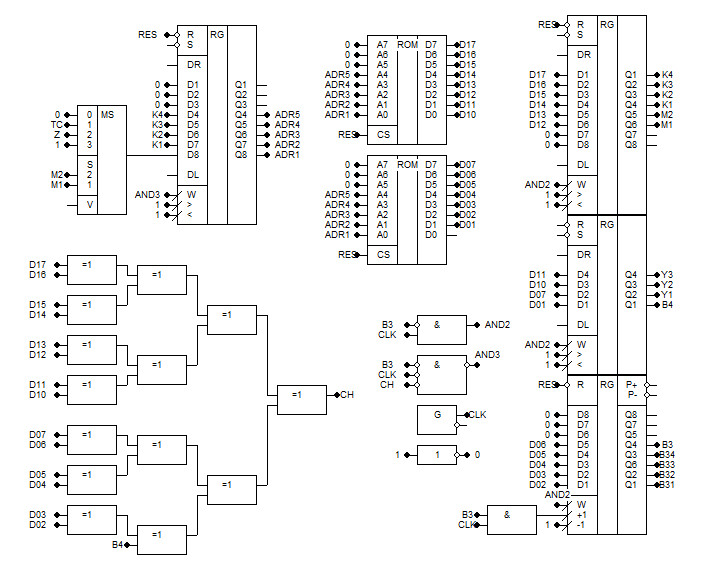
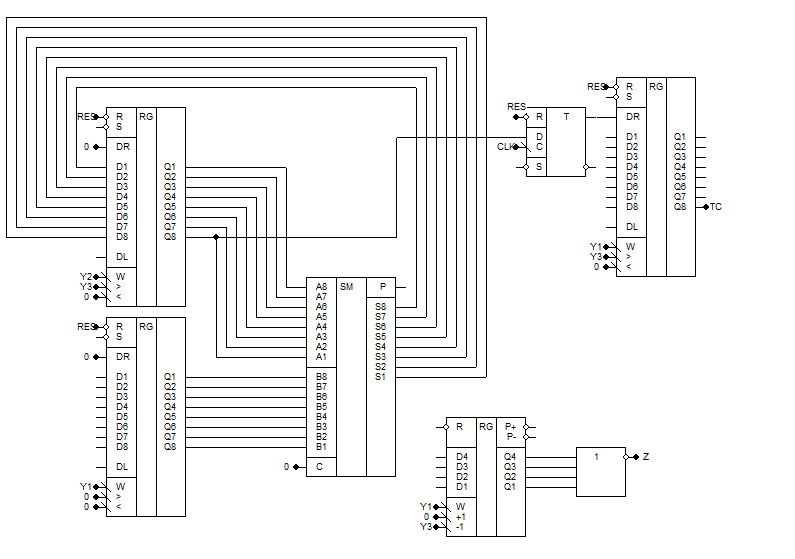
Структурна схема БМУ із лінійною ПМК та примусовим способом адресації мікрокоманд зображена на рис. 7.



* + - * 1. Рис. 7. Схема БМУ з примусовою адресацією

1. **4. Висновки:**
2. *У даній роботі побудована функціональна схема в програмі AFDK 2.0, яка виконує обчислення першого способу множення з 8-х розрядними значеннями в якості операндів. В результаті виконання цієї роботи, я згадав навички по використанню AFDK та пригадав й закріпив теоретичні аспекти цієї теми.*

**5. Функціональна схема:**

****