

## Лабораторна робота №5

**Тема:** організація пам'яті даних (пд) і програм (пп)

**Мета:** Вивчити сторінкову організацію і способи адресації пам'яті програм.

Навчитися здійснювати явний та неявний виклик підпрограм.

**Індивідуальне завдання:**

1. Написати програму мовою асемблера для мікроконтролера PIC16F84:

- виконати ініціалізацію мікроконтролера (зробити необхідні налаштування елементів, вузлів та модулів мікроконтролера, які використовуються для вирішення поставленої задачі);

- розмістити в нульовій сторінці пам'яті програм за адресою  $A_i = N_i + 100$  ( $N_i$  - номер за списком у журналі групи) підпрограму, що виконує запис константи  $M$  у комірку пам'яті даних  $A$  (див. лабораторну роботу № 1). Виклик підпрограми здійснювати явним чином;

- розмістити в першій (якщо вона є) або в нульовій (якщо першої немає) сторінці пам'яті програм за адресою  $A_i = N_i + 200$  підпрограму для організації паралельного інтерфейсу, що виконує читання порту  $B$  і запис його вмісту в пам'ять даних та виведення цих даних у порт  $A$  (див. лабораторну роботу № 2). Виклик підпрограми здійснювати неявним чином;

- розмістити в нульовій сторінці пам'яті програм за адресою  $A_i = N_i + 300$  підпрограму, що виконує за допомогою таймера ділення зовнішньої частоти  $F$  на коефіцієнт  $K1$  і виводить отриману частоту  $F/K1$  на вивід  $RB0$  мікроконтролера (див. лабораторну роботу № 3). Виклик підпрограми здійснювати явним чином.

2. Відлагодити програму у пакеті MPLAB. Індивідуальні варіанти завдань, окрім початкових адрес підпрограм, наведені в лабораторних роботах № 1, 2, 3.

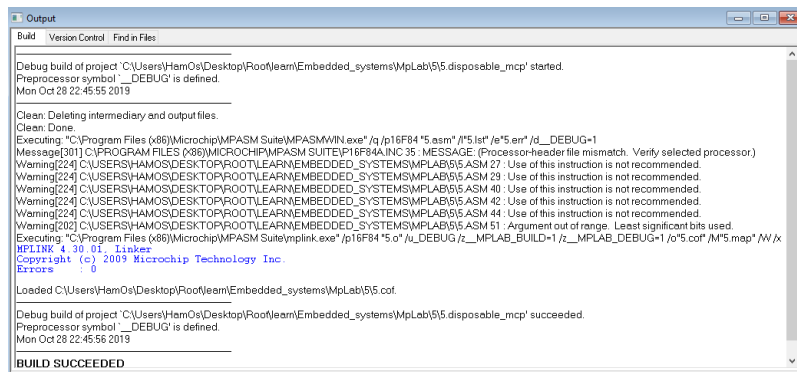


Рисунок 1 - Компіляція програма

```
#include<pic16f84a.inc>
A equ .11 ;.number - dec number
M equ .176
k1 equ .480
k2 equ .0

ORG 0 ;pic16f84 - 0 (точка входа)контр...pic 16056 - (точка входа) 3ff
call proc1
call proc2
call proc3
org 0x112; розмістити в нульовій сторінці пам'яті програм за адресою Ai = Ni + 100 (Ni - номер
proc1
movlw M
movlw M;mov - пересылка,1 - литерал,w - регистр аккумулятора.т.е. пересылка литерала в аккумуля
movwf A;f - пересылка в файл регистр M до 60-80
;nop
;goto $-1 ;метка в виде текущего адреса,переход на пор.
return
proc2
movlw 0x02;запис старших розрядів аدرس
movwf PCLATH;
movlw 0x45
movwf PCL;
org 0x212
movlw 0xFF;
tris PORTB; настроювання порту B на ввід
movlw 0x00;
tris PORTA; настроювання порту A на ввід
movf PORTB,0; читання порту B у акумулятор
movwf A; запис із акумулятора у пам'ять даних
movwf PORTA; вивід у порт A молодшої тетради,
swarf A,0; обмін тетрад у комірці пам'яті A, результат; буде поміщений в акумулятор (W)
movwf PORTA; вивід у порт A старшої тетради
;loop goto loop ; останов програми
return
org 0x312
proc3
movlw b'00101000';передільника з коефіцієнтом розподілу
OPTION
movlw 0x00 ;всі розряді на ввід
TRIS PORTB ;настроювання порта B на вивід
movlw 0xff ;всі розряді на вивід
TRIS PORTA ;настроювання порта A на ввід
start movlw .256-k1 ;формування константи для скидання таймера
movwf TMR0 ;ініціалізації таймера
bcf INTCON,TOIF ;скидання прпорця переповнення таймера
M1 btfs INTCON,TOIF ;очікування переповнення таймера
goto M1; перехід, якщо таймер не переповнений
bcf PORTB,4 ;установка початкового рівня вихідного сигналу
movlw .256-k2
movwf TMR0 ;ініціалізації таймера
bcf INTCON,TOIF
M2 btfs INTCON,TOIF ;очікування переповнення таймера
goto M2 ; перехід, якщо таймер не переповнений
bcf PORTB,4
goto start
end
```

Рисунок 2 - код програми

підпрограма proc1

Address	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	ASCII
00	--	00	15	18	00	00	00	--	00	00	00	B0	00	00	00	00	.....
10	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
20	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
30	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
40	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
50	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	.....
60	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	.....
70	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	.....
80	--	FF	15	18	00	1F	FF	--	00	00	00	B0	00	00	00	00	.....
90	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
A0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
B0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
C0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
D0	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	.....
E0	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	.....
F0	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	.....

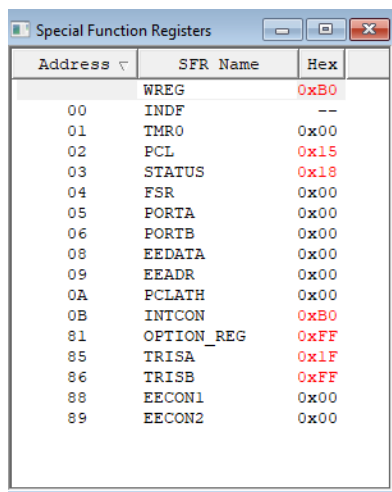
Рисунок 2 – Перегляд пам'яті даних

Address	Hex	Decimal	Symbol Name
00	--	-	INDF
01	0x00	0	TMR0
02	0x01	1	PCL
03	0x18	24	STATUS
04	0x00	0	FSR
05	0x00	0	PORTA
06	0x00	0	PORTB
07	--	-	GIE
08	0x00	0	EEDATA
09	0x00	0	EEADR
0A	0x00	0	PCLATH
0B	0xB0	176	INTCON
0C	0x00	0	
0D	0x00	0	
0E	0x00	0	
0F	0x00	0	_CP_ON

Рисунок 3 – Перегляд пам'яті даних

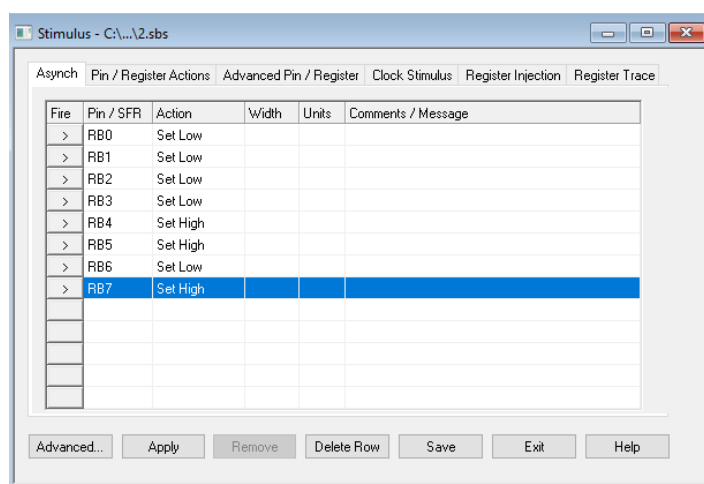
Line	Address	Opcode	Disass
1	000	2112	CALL 0x112
2	001	2116	CALL 0x116
3	002	2312	CALL 0x312
4	003	3FFF	
5	004	3FFF	
6	005	3FFF	
7	006	3FFF	

Рисунок 4 - Перегляд пам'яті програм



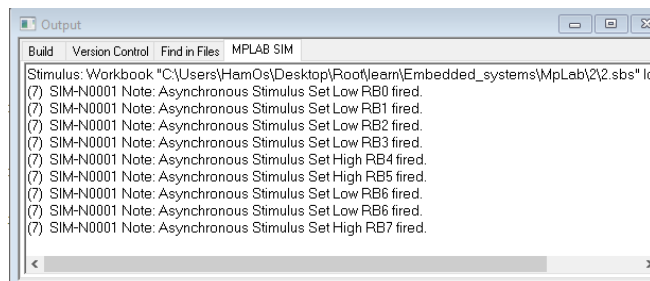
Address	SFR Name	Hex
	WREG	0xB0
00	INDF	--
01	TMR0	0x00
02	PCL	0x15
03	STATUS	0x18
04	FSR	0x00
05	PORTA	0x00
06	PORTB	0x00
08	EEDATA	0x00
09	EEADR	0x00
0A	PCLATH	0x00
0B	INTCON	0xB0
81	OPTION_REG	0xFF
85	TRISA	0x1F
86	TRISB	0xFF
88	EECON1	0x00
89	EECON2	0x00

Рисунок 5 - Перегляд спеціальних регістрів  
 підпрограма proc2



Fire	Pin / SFR	Action	Width	Units	Comments / Message
>	RB0	Set Low			
>	RB1	Set Low			
>	RB2	Set Low			
>	RB3	Set Low			
>	RB4	Set High			
>	RB5	Set High			
>	RB6	Set Low			
>	RB7	Set High			

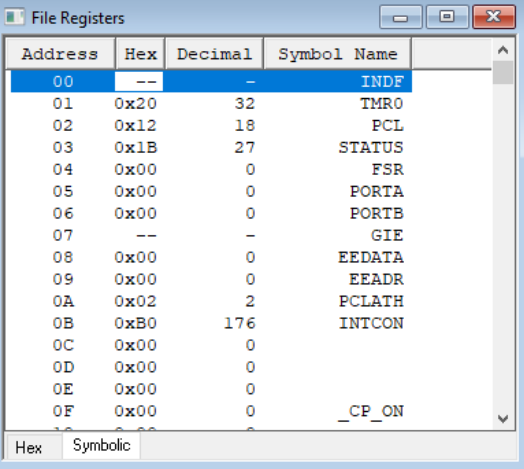
Рисунок 6 - вибираємо висновки порту В



```

Build   Version Control  Find in Files  MPLAB SIM
Stimulus: Workbook "C:\Users\HamOs\Desktop\Root\learn\Embedded_systems\Mplab\2\2.sbs" to
(7) SIM-N0001 Note: Asynchronous Stimulus Set Low RB0 fired.
(7) SIM-N0001 Note: Asynchronous Stimulus Set Low RB1 fired.
(7) SIM-N0001 Note: Asynchronous Stimulus Set Low RB2 fired.
(7) SIM-N0001 Note: Asynchronous Stimulus Set Low RB3 fired.
(7) SIM-N0001 Note: Asynchronous Stimulus Set High RB4 fired.
(7) SIM-N0001 Note: Asynchronous Stimulus Set High RB5 fired.
(7) SIM-N0001 Note: Asynchronous Stimulus Set Low RB6 fired.
(7) SIM-N0001 Note: Asynchronous Stimulus Set Low RB6 fired.
(7) SIM-N0001 Note: Asynchronous Stimulus Set High RB7 fired.
  
```

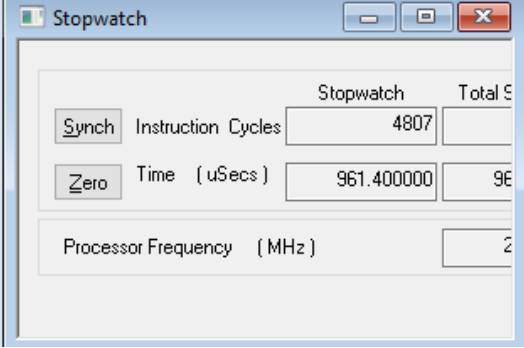
Рисунок 7 – натискаем fire



Address	Hex	Decimal	Symbol Name
00	--	-	INDF
01	0x20	32	TMR0
02	0x12	18	PCL
03	0x1B	27	STATUS
04	0x00	0	FSR
05	0x00	0	PORTA
06	0x00	0	PORTB
07	--	-	GIE
08	0x00	0	EEDATA
09	0x00	0	EEADR
0A	0x02	2	PCLATH
0B	0xB0	176	INTCON
0C	0x00	0	
0D	0x00	0	
0E	0x00	0	
0F	0x00	0	_CP_ON

Рисунок 8 - перегляд пам'яті даних

Proc3



Stopwatch		Total S
<input type="button" value="Synch"/>	Instruction Cycles	4807
<input type="button" value="Zero"/>	Time (uSecs)	961.400000
Processor Frequency (MHz)		2

Рисунок 9 – використання StopWatch

## Висновок

1. Написати програму мовою асемблера для мікроконтролера PIC16F84, що виконує запис константи  $M$  у комірку пам'яті даних  $A$  в нульовій сторінці пам'яті за адресою  $A_i = N_i + 100$  (Виклик підпрограми здійснювати явним чином), виконує читання порту  $B$  і запис його вмісту в пам'ять даних та виведення цих даних у порт  $A$  в першій сторінці пам'яті програм за адресою  $A_i = N_i + 200$  (Виклик підпрограми неявним чином) та програму в нульовій сторінці пам'яті за адресою  $A_i = N_i + 300$ , яка допомогою таймера ділення зовнішньої частоти  $F$  на коефіцієнт  $K1$  і виводить отриману частоту  $F/K1$  на вивід  $RB0$  мікроконтролера.