Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
Кафедра безпеки інформації та телекомунікацій

РОБОТА З ПОРТАМИ ВВОДУ-ВИВОДУ ІНФОРМАЦІЇ ТА СТВОРЕННЯ ПІДПРОГРАМ ЗАТРИМКИ НА МОВІ АСЕМБЛЕР

Методичні рекомендації
до виконання лабораторної роботи ОТМП-1
з дисципліни «Обчислювальна техніка та мікропроцесори»
для студентів спеціальності
172 Телекомунікації та радіотехніка

Дніпро
НТУ «ДП»
2020

Робота з портами вводу-виводу інформації та створення підпрограм затримки на мові асемблер. Методичні рекомендації до виконання лабораторної роботи ОТМП-1 з дисципліни «Обчислювальна техніка та мікропроцесори» для студентів спеціальності 172 Телекомунікації та радіотехніка / О.О. Плєц, І.Г. Олішевський, О.В. Кручінін, Ю.П. Рибальченко ; М-во освіти і науки України, , Нац. техн. ун-т «Дніпровська політехніка». – Дніпро : НТУ «ДП», 2020. – 29 с.

Автори:
О.О. Плєц, асист.,
І.Г. Олішевський, асист.,
О.В. Кручінін, ст. викл.,
Ю.П. Рибальченко, асист.

Затверджено методичною комісією за спеціальністю 172 Телекомунікації та радіотехніка (протокол № 2 від 05.11.2020) за поданням кафедри безпеки інформації та телекомунікацій (протокол № 4 від 05.11.2020).

Методичні рекомендації призначено для виконання лабораторних робіт з дисципліни «Обчислювальна техніка та мікропроцесори» студентами спеціальності 172 Телекомунікації та радіотехніка.

Орієнтовано на активізацію навчальної діяльності бакалаврів та закріплення практичних знань з даної дисципліни.

Відповідальний за випуск завідувач кафедри безпеки інформації та телекомунікацій В.І. Корнієнко, д-р техн. наук, проф.

Зміст

Мета та програма роботи

Мета − ознайомитись із середовищем розробки Microchip MPLAB IDE та Proteus. Здобути навички написання програм мовою асемблера для мікроконтролера середнього сімейства PIC16F84(A).

Програма роботи:

1. За допомогою програмного забезпечення Microchip MPLAB IDE написати на мові асемблера підпрограму, яка почергово встановлює на виводі порта B високий та низький рівні сигналу. Тривалість високого та низького рівнів сигналу повинна бути однакова та виконана за допомогою підпрограми затримки.

2. У ПЗ Proteus перевірити роботи написаної програми.

3. Використовуючи програматор/відлагоджувач ICD 2 та демонстраційну плату PICDEM 2 PLUS, записати hex-файл у пам'ять мікроконтролера та пересвідчитись у його працездатності.

1 Теоретичні відомості

Результат будь-якої операції, що виконується центральним процесором, зберігається у робочому регістрі, однак у ньому не завжди присутні всі необхідні дані стосовно щойно виконаної операції. Наприклад, якщо в результаті виконання операції складання буде перевищено 8-розрядний діапазон, то робочий регістр не розпізнає подібну ситуацію, та збереже некоректний результат. Враховуючи це, у будь-який центральний процесор комп’ютера вбудовано набір логічних розрядів, які ще називають прапорами умов. Вони призначені для передавання додаткової інформації про результат виконання останньої команди. Наприклад, він є нульовим, від’ємним або додатним. У мікроконтролері PIC16F84A ці прапори зберігаються у регістрі стану STATUS (рис. 1). Нумерація бітів починається з 0 та йде з права наліво.

R/W-0	R/W-0	R/W-0	R-1	R-1	R/W-x	R/W-x	R/W-x
IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C

Рис. 1. Регістр стану STATUS

По суті, до категорії прапорів умов можна віднести тільки три розряди цього регістру: від 0 до 2 біта.

Розряди 7–6: зарезервовані, підтримуються як «0».

Розряд 5: RP0 – біт вибору банку пам’яті (використовується як пряма адресація). При значенні «0» – Банк 0 (адреса регістрів 00h – 7Fh). При значенні «1» – Банк 1 (адреса регістрів 80h – FFh).

Розряд 2: Z – біт нульового результату (Zero). «1» – якщо результат логічної або арифметичної операції є нульовим. «0» – якщо результат логічної або арифметичної операції не є нульовим.

Розряд 1: DC – біт знака переносу (Digit Carry). «1» – є вихідний сигнал знака переносу з 4-го молодшого розряду для отриманого результату. «0» – немає вихідного сигналу знаку переносу з 4-го молодшого розряду для отриманого результату.

Розряд 0: C – біт переносу (Carry). «1» – вихідний сигнал переносу найстаршого розряду для отриманого результату. «0» – немає вихідного сигналу переносу найстаршого розряду для отриманого результату.

Проблема, притаманна будь-якій області пам’яті, полягає в тому, що чим більше пам’яті, тим ширше повинна бути її шина адреси. Одним з методів зменшення адресної шини є розбиття пам’яті на декілька менших блоків однакового розміру, які називають банками (рис. 2). Це дозволяє використовувати більш вузьку шину адреси, за допомогою якої можна однаковим способом отримати доступ до всіх банків. При цьому, у певний момент часу як цільовий може бути ідентифікований тільки один банк.

У мікроконтролерах PIC структура з банками застосовується для пам’яті RAM. Так, мікроконтролер 16F84A має тільки два таких банки. Адреса кожного з них – семирозрядна адреса RAM. Активний банк обирається за допомогою розряду 5 регістру стану STATUS (рис. 1). Перед отриманням доступу до пам’яті програміст повинен переконатися, що в регістрі стану розряд банку встановлено правильно.

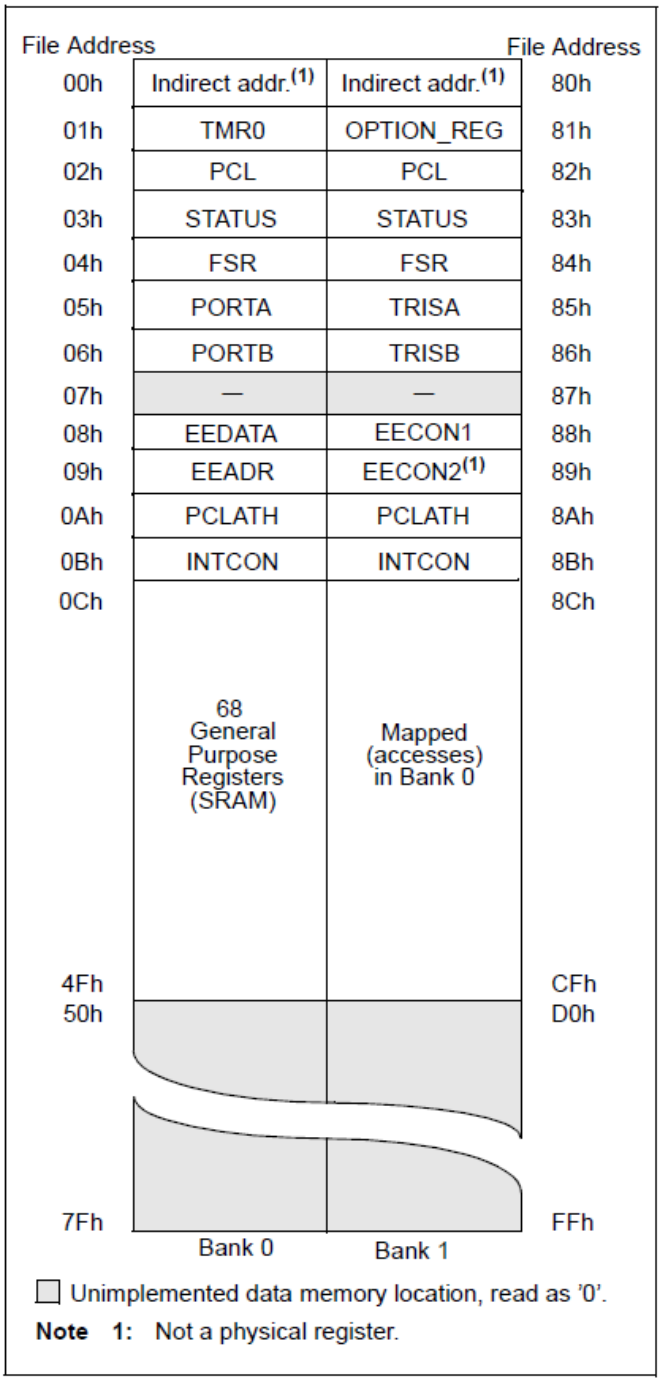

Рис. 2. Пам’ять даних та схема розміщення регістрів спеціального призначення мікроконтролера 16F84A

Регістри PORTB і TRISB

PORTB – це восьмирозрядний, двонаправлений порт вводу–виводу. Біти регістру TRISB задають напрямок інформації за відповідними бітами (пінами) регістру PORTB. Встановлення деякого біту TRISB в «1» спрямовує відповідний пін регістру PORTB на вивід інформації. Встановлення деякого

біту TRISB в «0» спрямовує відповідний пін регістру PORTB на ввід інформації.

При написанні програми необхідно дотримуватися таких правил:

1) програмний код повинен закінчуватися директивою **End;**
2) після директиви End необхідно **додати два пусті рядки;**
3) після директиви **org 0x00** повинен розміщуватися програмний код.

Приклад: визначення констант асемблера

```
org 0x00
        перший рядок коду
        другий рядок коду
        ……
        ……

        ……..
        останній рядок коду
End
```

^ Два пустих рядки.

Таблиця 1

Формати відображення чисел у MPASM

Формат	Синтаксис	Приклад
Десятичний	**D**„число“ .число	D„100“ .100
Шістнадцятиричний	**H**„число“ **0x**число	H„G7“ 0xDF
Восьмеричний	**O**„число“	O„112“
Двійковий	**B**’число’	B’11110000’

Основні директиви MPASM

_ _CONFIG

Встановлення бітів конфігурації мікроконтролера

Синтаксис

```
__config <expr> OR _ _config <addr>, <expr>
```

Приклад

```
__config 0xFFFF
```

#DEFINE

Ця директива замінить у тексті рядок <name> на рядок [<string>]. У випадку використання без рядка [<string>] директива взаємодіє з блоком умовної компіляції IFDEF.

Синтаксис

```
#define <name> [<string>]
#define <name> [<arg>,...,<arg>] <string>
```

Приклад

```
#define SIZE 64
#define timer 0x89 ,15
```

END

Зазначає кінець програмного коду

Синтаксис

```
End
```

Приклад

```
    decfsz MARS, f
    goto star
    return
end
```

EQU

Визначення константи асемблера

Синтаксис

```
    <label> equ <expr>
```

Приклад

```
    STATUS equ 03
```

```
status equ 03
TRISB  equ 06
TRISB  equ 86
```

INCLUDE

Підключити додатковий файл з програмним кодом

Синтаксис

```
include <file>
include "file"
```

Приклад

```
include "my_lib"  ;
include <regs.h> ;
```

ORG

Встановити адрес програми. Команда асемблера, яка зазначена після директиви, буде розміщена в пам'яті програм за наведеною адресою. Якщо перед директивою встановлена мітка, то вона буде мати адресу, яку наведено в директиві.

Синтаксис

<label> org <expr>

Приклад

start org 0x00
……..
……..
pluton org start+0x04
……
……

2 Хід виконання роботи

1. Відкрити MPLAB IDE. Повинно відкритися вікно аналогічне зображеному на рис. 3.

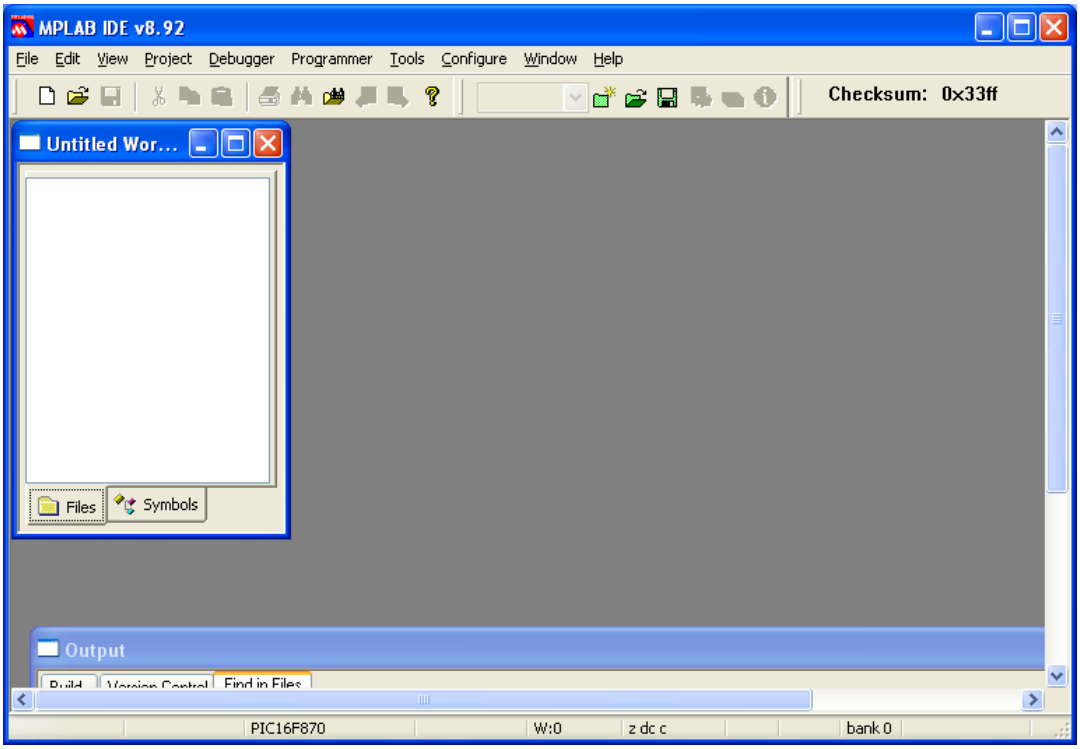

Рис. 3. Середовище MPLAB IDE

2. Вибраємо меню Project >> Project Wizard…

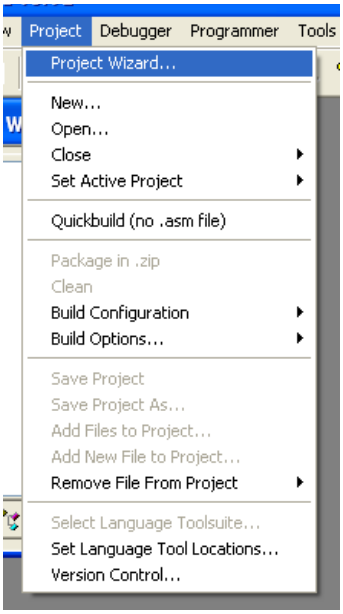

Рис. 4. Меню Project

3. Повинно з’явитися вікно Майстра Проектів, у якому натискаємо кнопку «Далі».

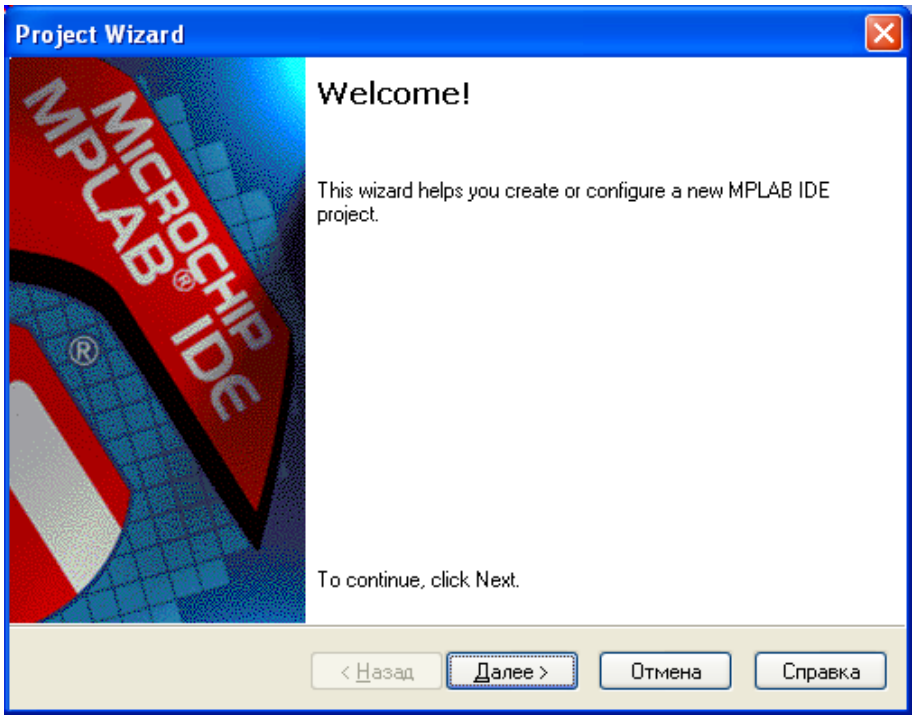

Рис. 5. Вікно Майстра Проектів

4. Наступним кроком потрібно вибрати необхідний мікроконтролер.

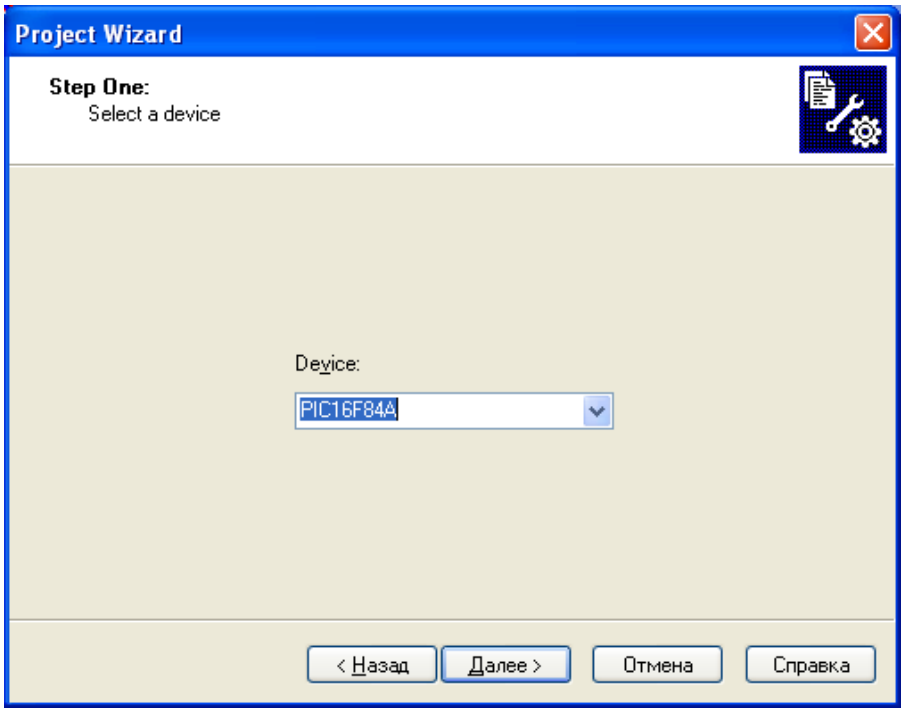

Рис. 6. Вибір мікроконтролера у Майстрі Проектів

5. Вибираємо набір інструментів асемблера Microchip MPASM Toolsuite у розділі Active Toolsuite.

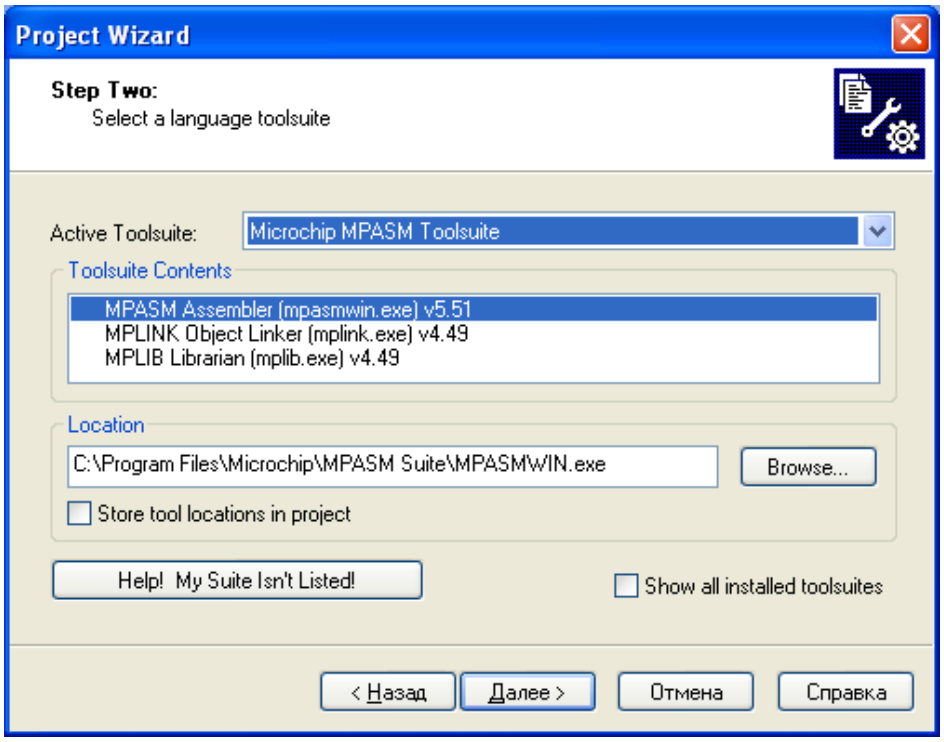

Рис. 7. Вибір набору інструментів асемблера Microchip MPASM Toolsuite

6. Вибираємо місце для зберігання проекту.

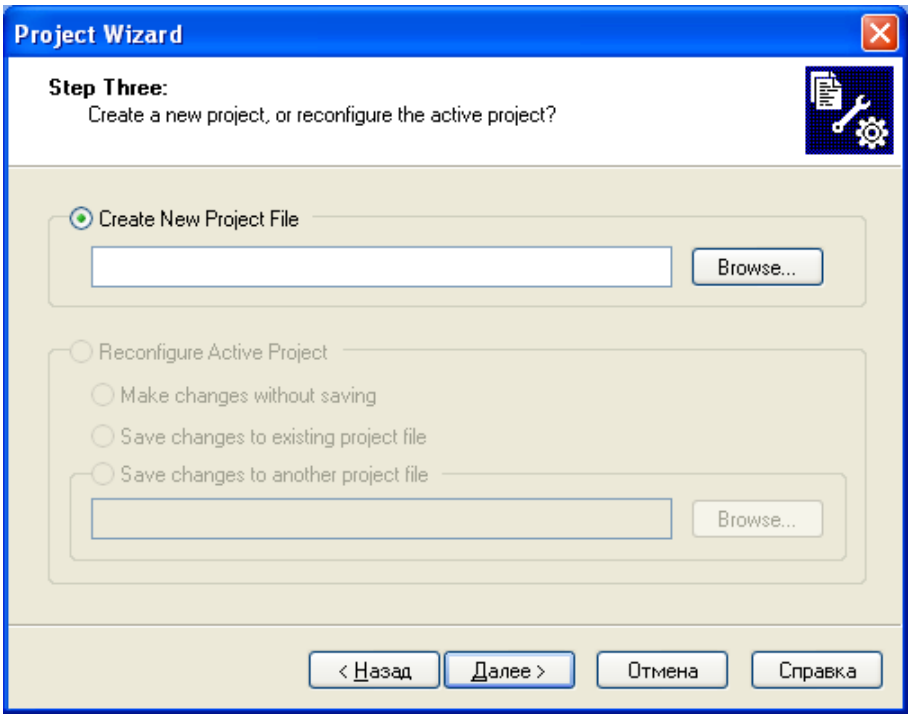

Рис. 8. Вибір місця для зберігання проекту

7. Якщо потрібно додати файли до проекту, то в даному меню є можливість додати. Оскільки це перший проект, тому ми не додаємо жодного файлу.

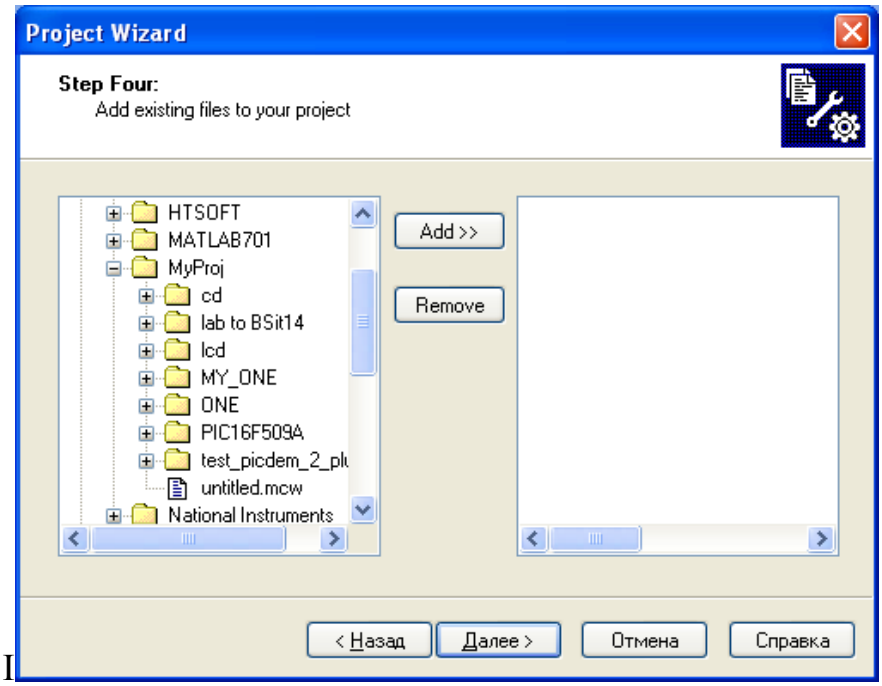

Рис. 9. Добавлення існуючих файлів до проекту

8. Перевіряємо, що задані параметри ми зазначили правильно. Натискаємо кнопку «Готово».

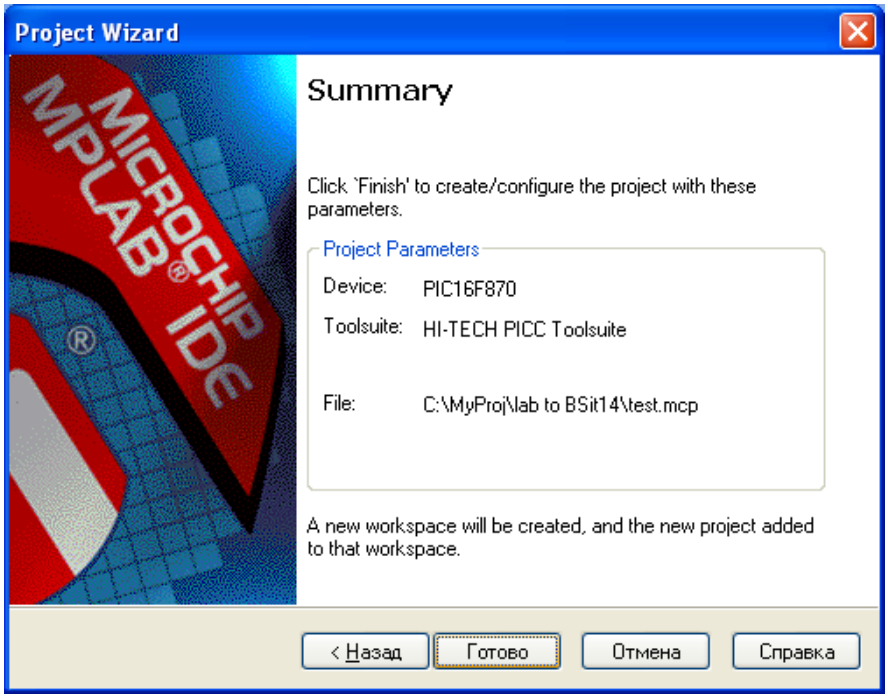

Рис. 10. Підсумкове вікно створення проекту

9. Потрібна бути картина, що аналогічна рис. 11.

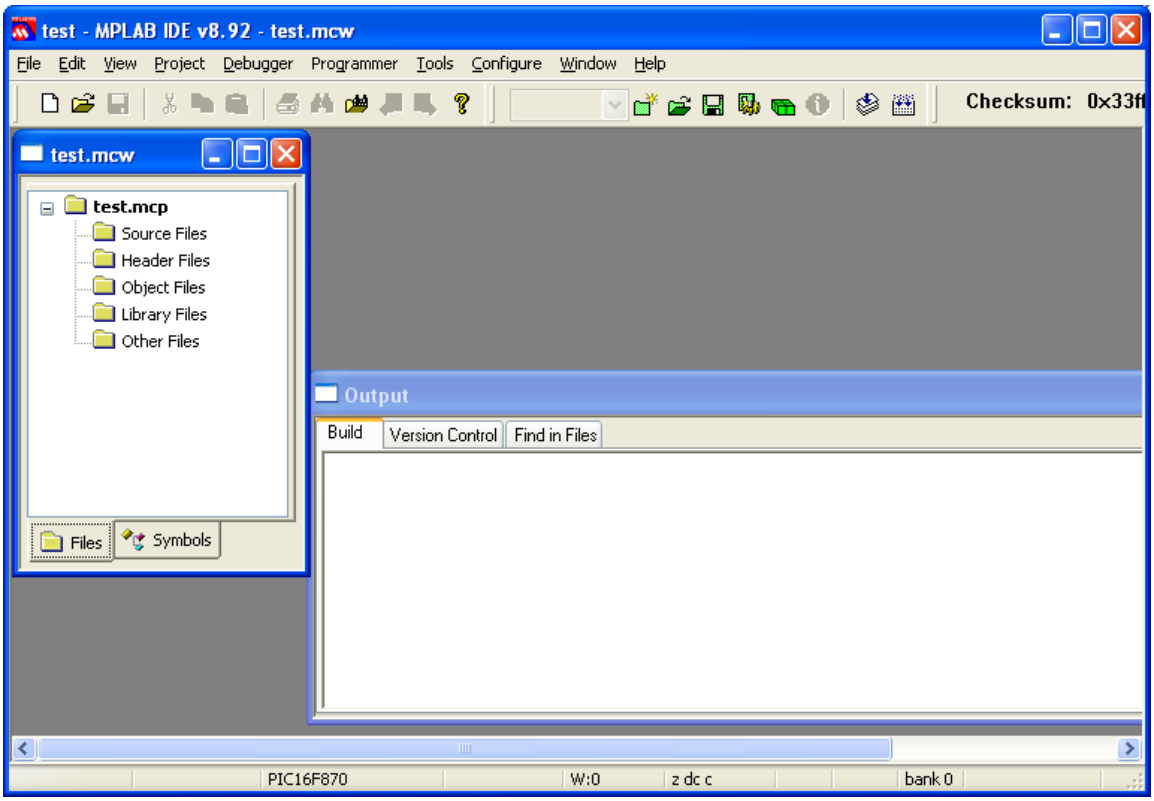

Рис. 11. Середовище MPLAB IDE

10. Необхідно створити файл. Це робиться за допомогою меню File>>New.

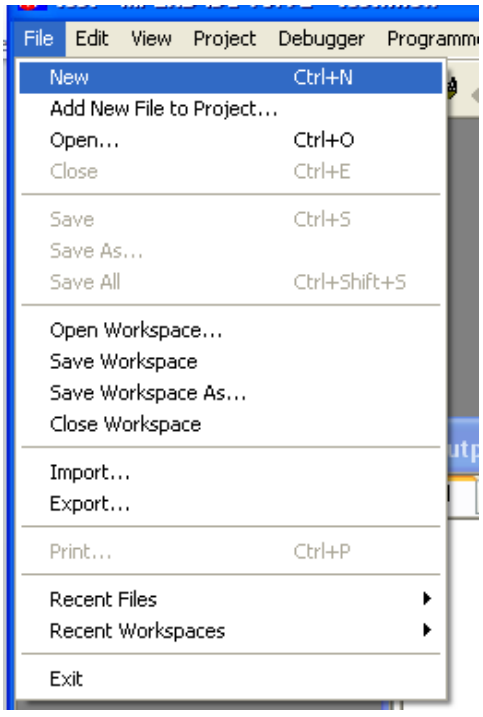

Рис. 12. Меню File

11. Має з’явитися текстовий редактор.

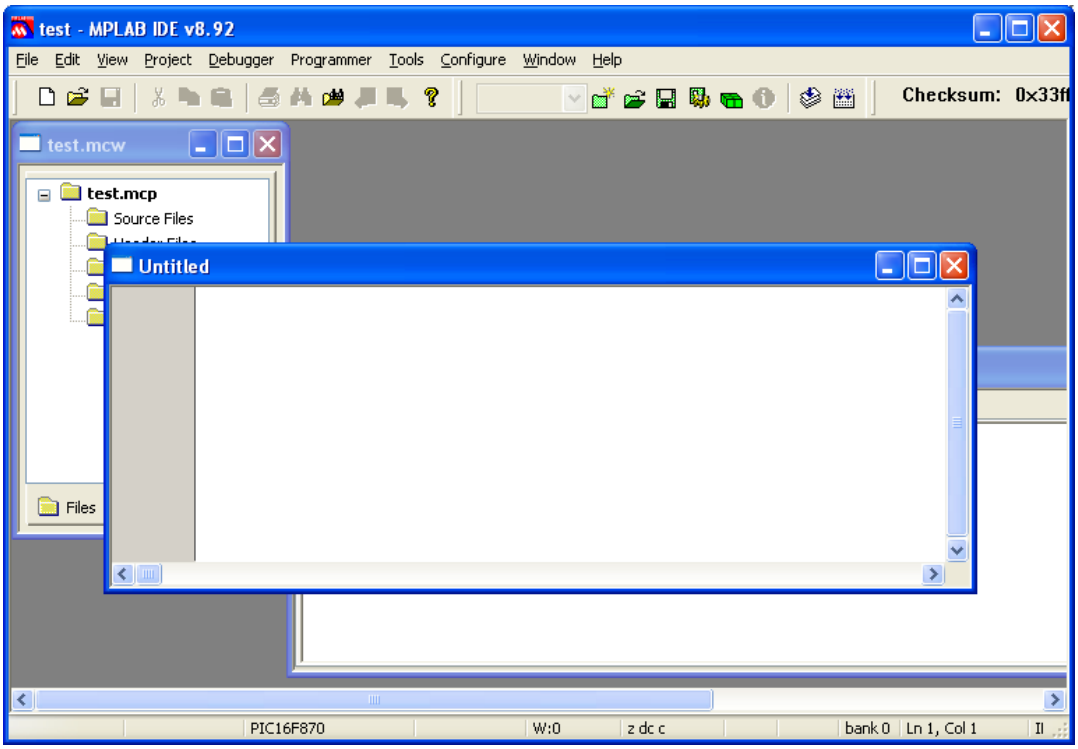

Рис. 13. Текстовий редактор

12. Цей файл потрібно зберегти. Виконуємо це за допомогою команди File>>Save As.

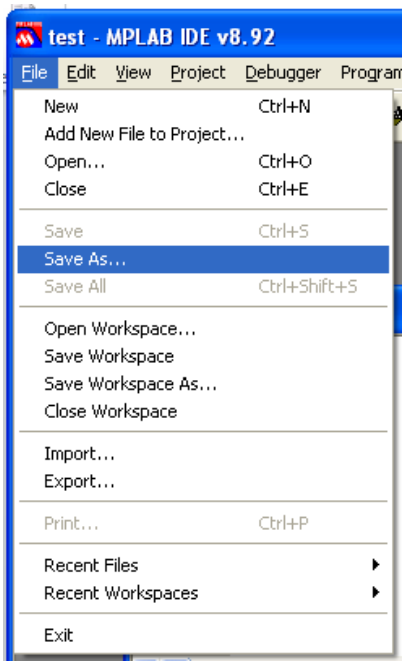

Рис. 14. Збереження файлу

13. Зазначаємо ім’я файлу та розширення .asm і натискаємо кнопку «Сохранить».

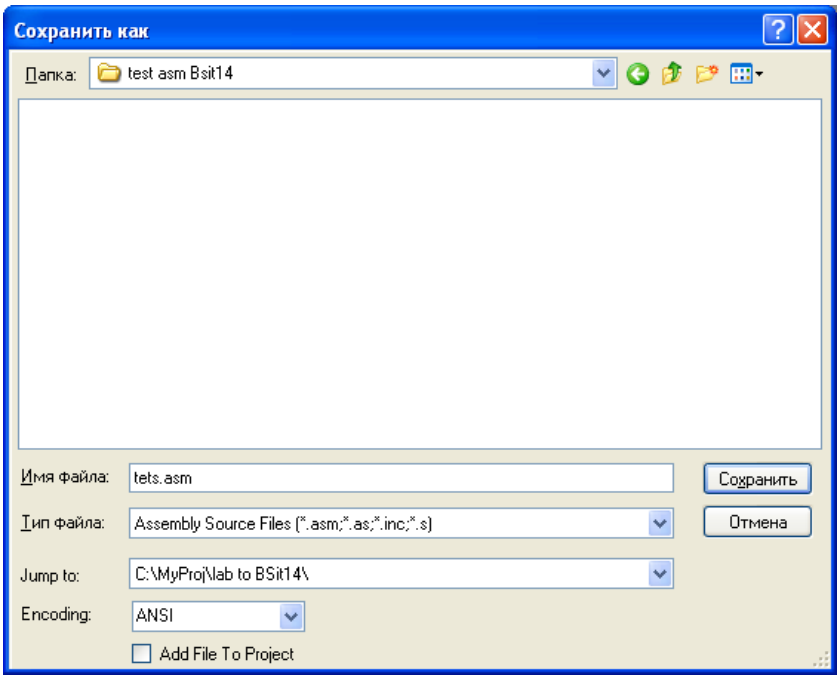

Рис. 15. Збереження файлу

14. Додаємо збережений файл до проекту.

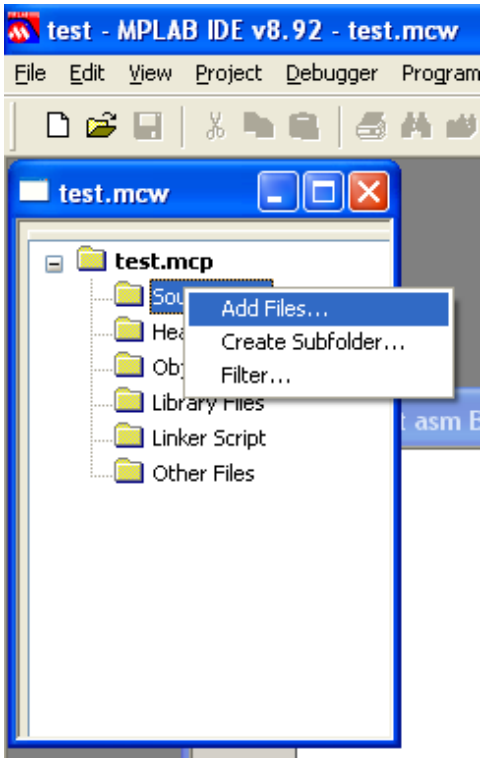

Рис. 16. Додавання файлу до проекту

15. Знаходимо наш файл та відкриваємо його.

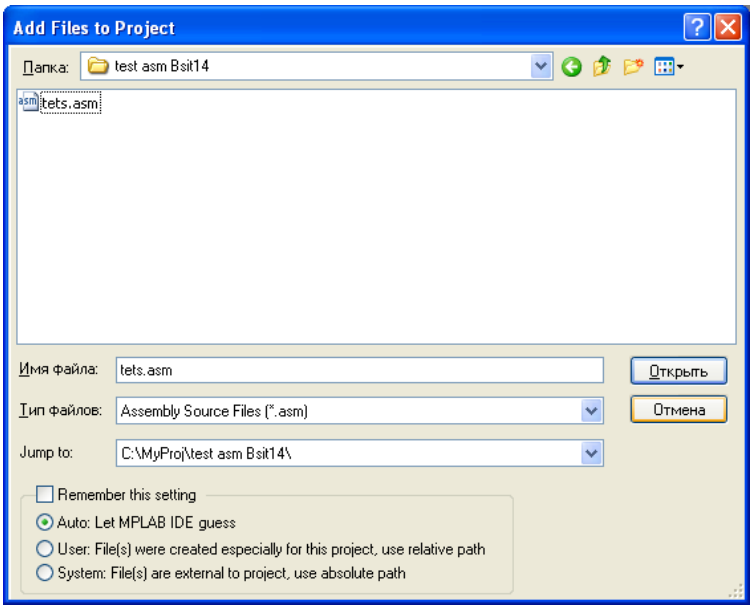

Рис. 17. Діалогове вікно додавання файлу до проекту

16. У вікні менеджера файлів проекту повинен з’явитися наш файл.

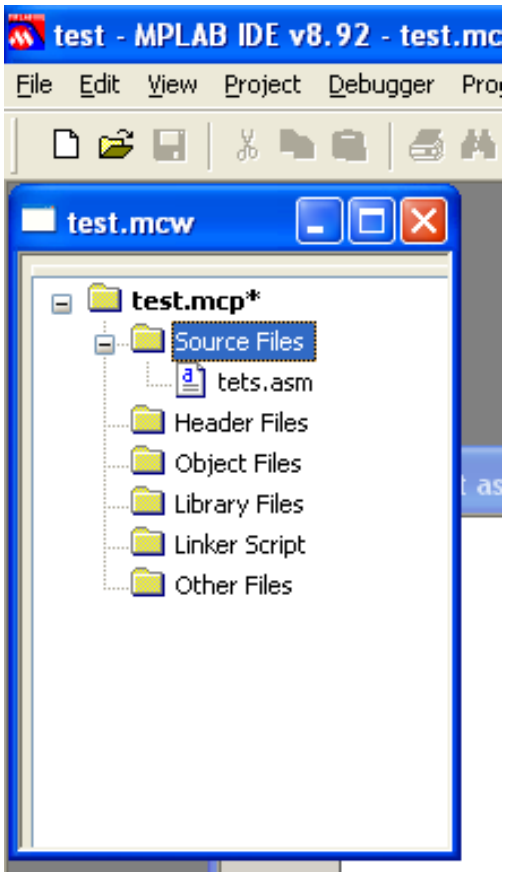

Рис. 18. Діалогове вікно додавання файлу до проекту

17. Після введення коду програми до файлу потрібно натиснути кнопку Збудувати весь проект: Project >> Build All.

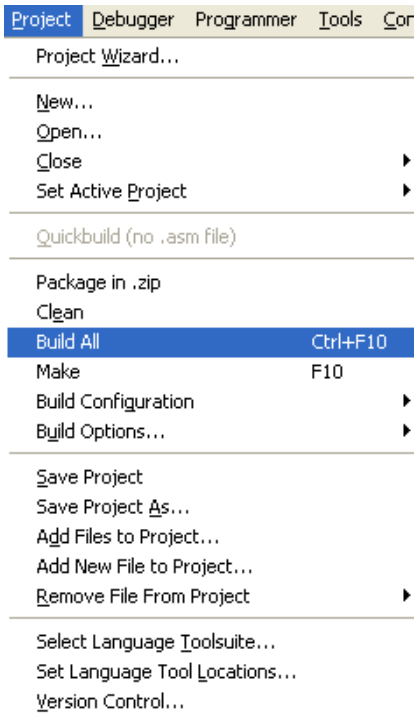

Рис. 19. Побудова проекту

18. Якщо помилок не знайдено, то у вікні Output з'явиться надпис «BUILD SUCCEEDED».

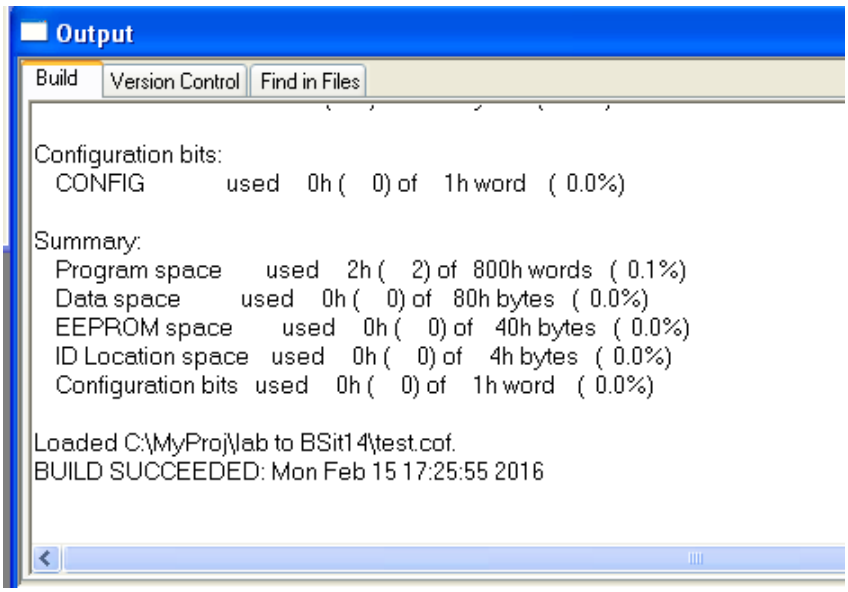

Рис. 20. Повідомлення про результат побудови проекту

19. Результатом роботи буде створений hex-файл.

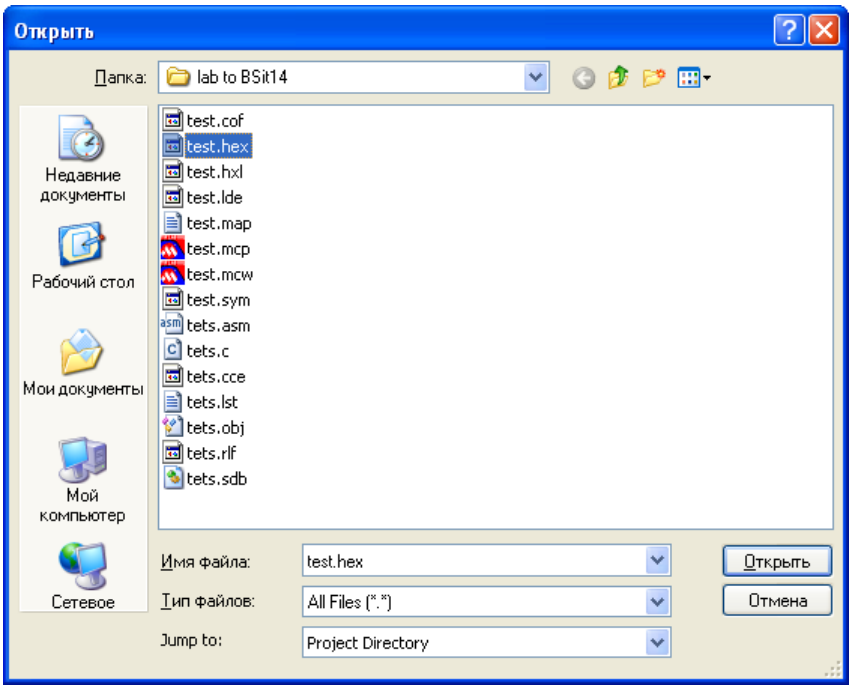

Рис. 21. Діалогове вікно

20. За допомогою ПЗ Proteus ISIS зібрати схему, зображену на рис. 22.

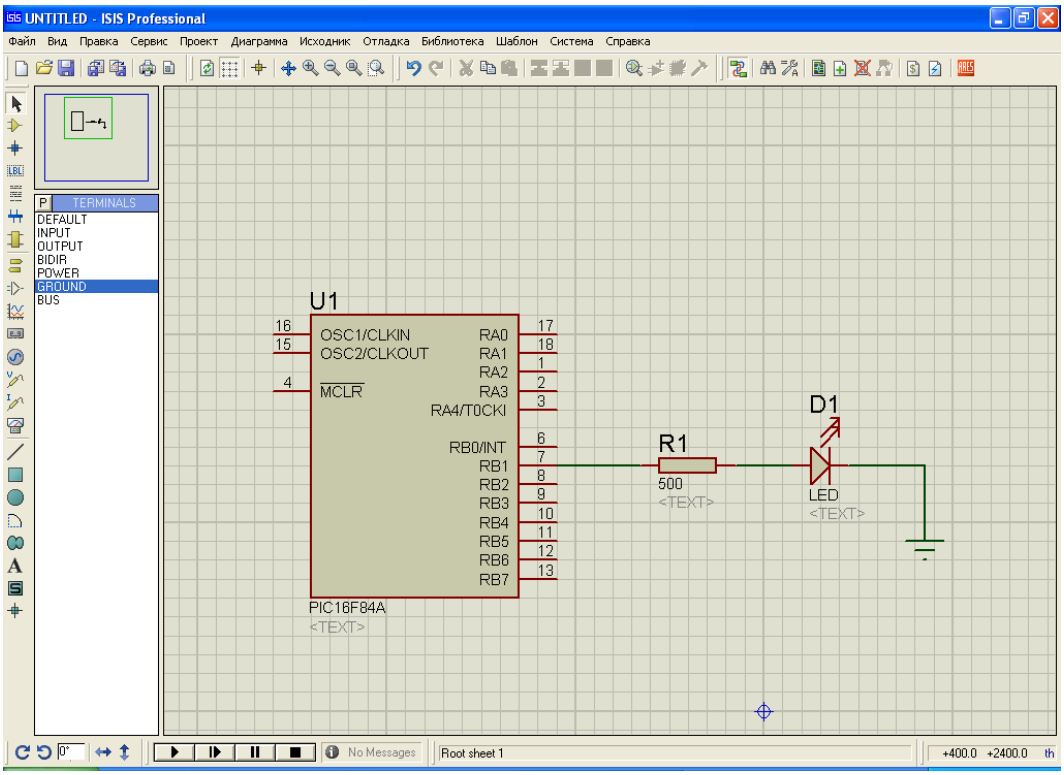

Рис. 22. Робоче середовище Proteus ISIS

21. Натиснути ПКМ на рисунку мікроконтролера та вибрати розділ «Правка свойств» (рис. 23–25)
22. У рядку Program File вибрати hex-файл, який було створено за допомогою ПЗ MPLAB IDE. Встановити необхідну частоту роботи мікроконтролера в рядку Processor Clock Freaquency.

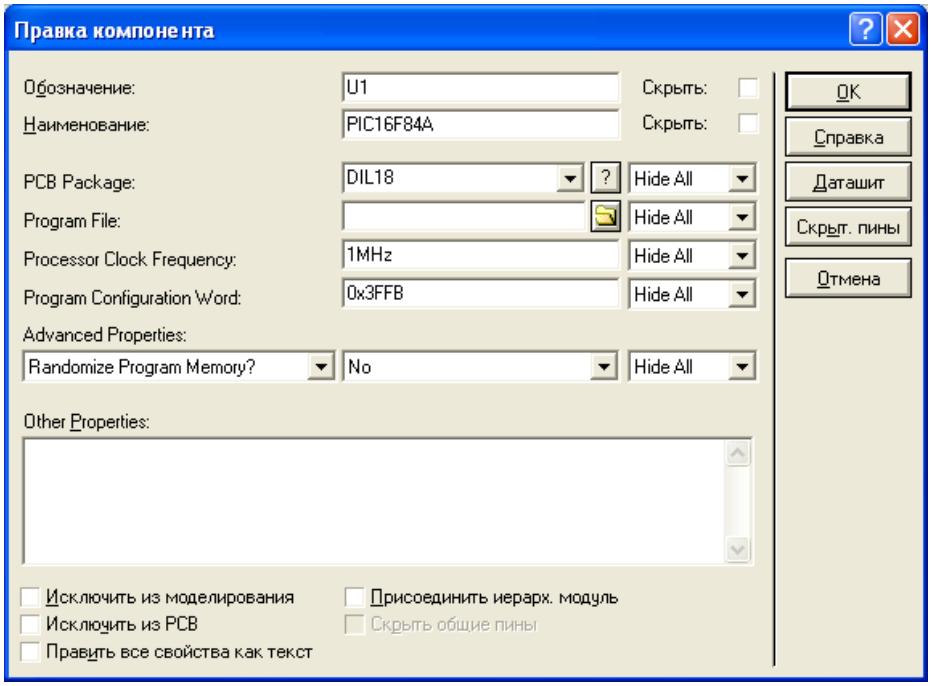

Рис. 23. Вікно налаштування компонента PIC16F84A

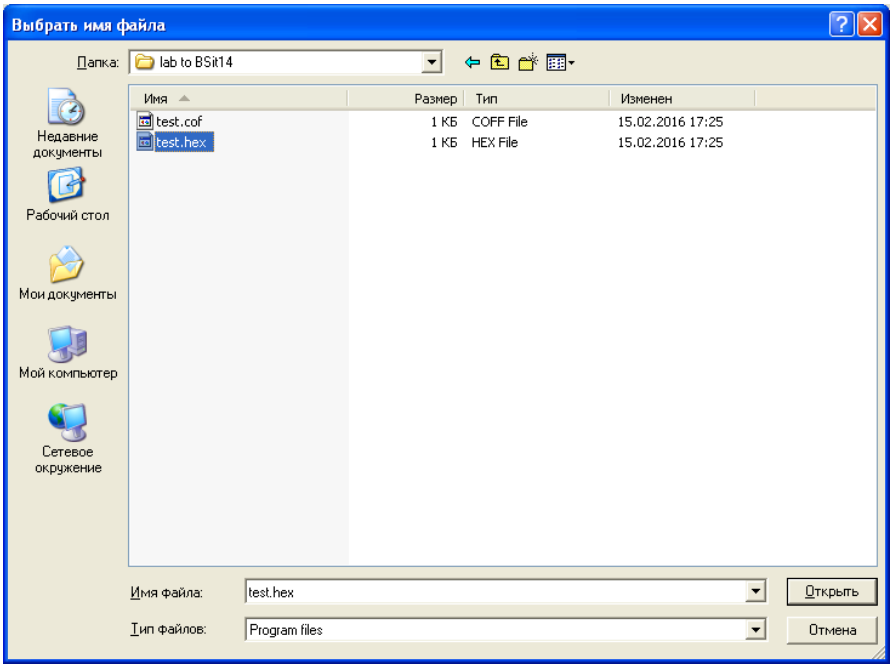

Рис. 24. Діалогове вікно підключення .hex файлу до PIC16F84A

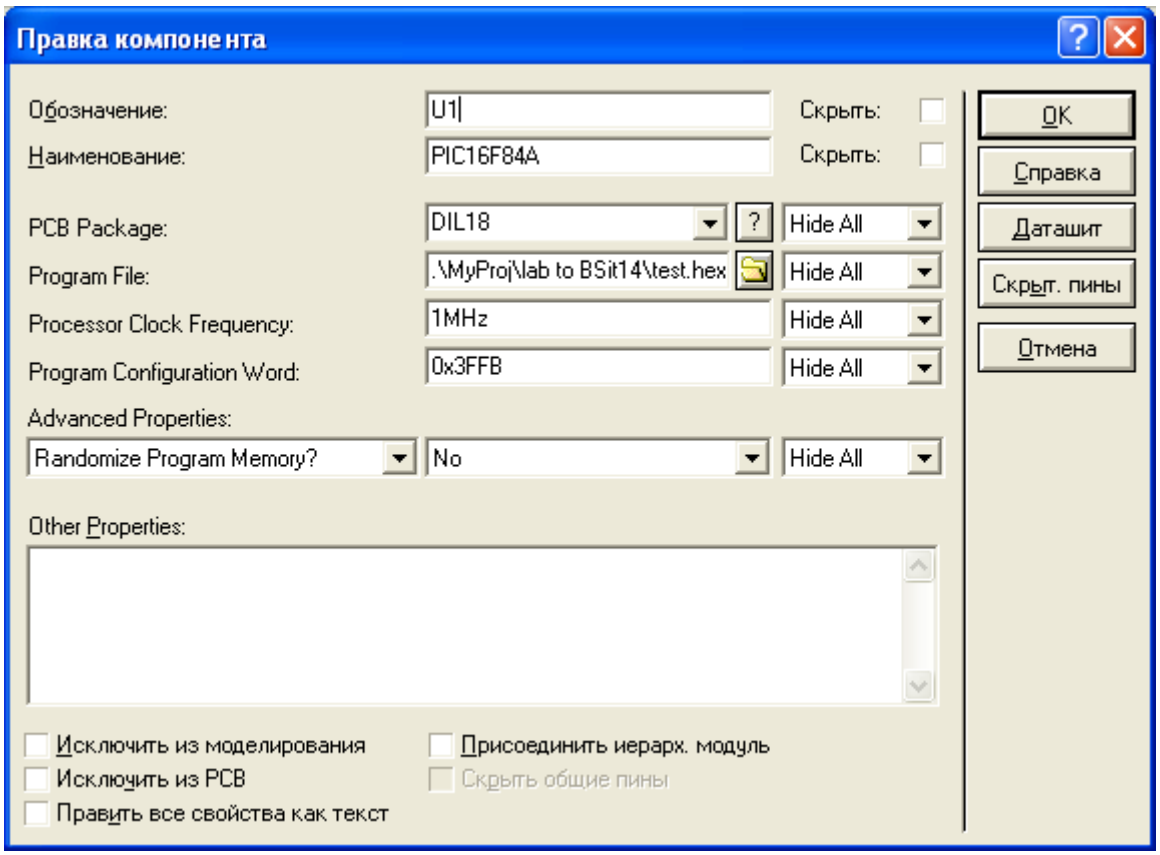

Рис. 25. Результат підключення .hex файлу до PIC16F84A

23. Запустити симуляцію роботи схеми.

3 Вихідні дані

Таблиця 2

Варіанти завдань

№ варіанта	Тривалість затримки, мс	Номер контакту порта B	№ варіанта	Тривалість затримки, мс	Номер контакту порта B
1	*10*	*RB0*	*17*	*170*	*RB3*
2	*20*	*RB0*	*18*	*180*	*RB3*
3	*30*	*RB0*	*19*	*190*	*RB3*
4	*40*	*RB0*	*20*	*200*	*RB3*
5	*50*	*RB0*	*21*	*210*	*RB0*
6	*60*	*RB1*	*22*	*220*	*RB1*
7	*70*	*RB1*	*23*	*230*	*RB2*
8	*80*	*RB1*	*24*	*240*	*RB3*
9	*90*	*RB1*	*25*	*250*	*RB0*
10	*100*	*RB1*	*26*	*260*	*RB1*
11	*110*	*RB2*	*27*	*270*	*RB2*
12	*120*	*RB2*	*28*	*280*	*RB3*
13	*130*	*RB2*	*29*	*290*	*RB0*
14	*140*	*RB2*	*30*	*300*	*RB1*
15	*150*	*RB2*	*31*	*310*	*RB2*
16	*160*	*RB3*	*32*	*320*	*RB3*

4 Вимоги до оформлення звіту

Звіт має бути оформлений у текстовому редакторі MS Word. Текст потрібно набирати шрифтом Times New Roman, 14 pt, вирівнювати по ширині, формат сторінки А4, книжка, абзацний відступ 10 мм, поля 20 мм з кожного боку. Формули додавати за допомогою Microsoft Equation або редактора формул. Усі таблиці та рисунки мають бути підписані. Дозволяється вихідний код розміщувати в декілька стовпців, а пусті рядки (незначні) пропускати.

Звіт обов'язково повинен мати такі складові:

1. Титульний аркуш (див. додаток).
2. Номер, назву, мету і задачу лабораторної роботи.
3. Теоретичну складову.
4. Код програми мовою асемблера з коментарями.
5. Зміст hex-файлу.
6. Зображення значення регістрів.
7. Розмір зайнятої пам'яті мікропроцесора (у кілобайтах та відсотках).
8. Результати роботи програматора ICD2.
9. Скріншоти роботи в симуляторі MPSIM та ПЗ Proteus (схему та осцилограму сигналу).
10. Порівняння файлу *.asm та файла *.lst.
11. Умотивовані висновки.

5 Зразок виконання завдання

Таблиця 3

Вихідні дані зразкового варіанта

№ варіанта	Тривалість затримки, мс	Номер контакту порта B
0	350	RB6

Код мовою асемблера.

```
STATUS    EQU 0x03          ; Визначення констант STATUS, TRISB, PORTB, які розміщені за адресами
TRISB     EQU 0x86          0x03, 0x86, 0x06 пам’яті даних.
PORTB     EQU 0x06


Reg_1     EQU 0x0c          ; Визначення констант Reg_1, Reg_2, Reg_3, які розміщені за адресами
Reg_2     EQU 0x0d          0x0C, 0x0D, 0x0E пам’яті даних.
Reg_3     EQU 0x0e


     org 0x00               ; Зазначаємо, що наступна команда (BSF STATUS,5) повинна розміщюватися за
                            адресою 0x00 пам’яті програм.
          BSF STATUS,5      ;встановлюємо одиницю в 5 біт регістра STATUS (перехід до банку 1,
                            оскільки в банку 1 знаходиться регістр TRISB)
          BCF TRISB, 6      ; встановлюємо нуль у 6 біт регістру TRISB (налаштовуємо контакт RB5
                            регістру PORTB на вихід )
          BCF STATUS,5      ;встановлюємо нуль у 5 біт регістру STATUS (перехід до банку 0, оскільки
                            в банку 1 знаходиться регістр PORTB)
```

```
k1        BSF PORTB,6       ; встановлюємо одиницю в 6 біт регістру PORTB (встановлюється високий
                            рівень (+5 Вольт) на контакті RB6 )
          CALL delay        ; виклик підпрограми delay
          BCF PORTB, 5      ; встановлюємо нуль у 6 біт регістру PORTB (встановлюється низький рівень
                            (0 Вольт) на контакті RB6 )
          CALL delay        ; виклик підпрограми delay
          GOTO k1           ; перейти на мітку k1
delay     movlw  .136       ; підпрограма delay
          movwf  Reg_1
          movlw  .199
          movwf  Reg_2
          movlw  .2
          movwf  Reg_3
m1        decfsz  Reg_1, F
          goto  m1
          decfsz  Reg_2, F
          goto  m1
          decfsz  Reg_3, F
          goto  m1
          nop
          return            ; вихід з підпрограми
end                         ; кінець програми
```

Контрольні питання

1. Чим відрізняються регістри загального користування від регістрів спеціального призначення?
2. У яких випадках встановлюється прапор C регістру STATUS?
3. Для чого використовується регістр PORTB?
4. Для чого використовується регістр TRISB?
5. У яких випадках встановлюється прапор DC регістру STATUS?
6. У чому відмінність регістрів PORTA та PORTB?
7. У яких випадках встановлюється прапор Z регістру STATUS?
8. Яким чином відбувається виклик підпрограми? Які дії виконує при цьому мікроконтролер?
9. Яким чином відбувається вихід з підпрограми? Які дії виконує при цьому мікроконтролер?
10. За допомогою якої команди відбувається безумовний перехід?

Список літератури

Уилмсхерст Т. Разработка встроенных систем с помощью микроконтроллеров PIC .– Киев : МК-Пресс, 2008. – 543 с.

Додаткова документація

1. PIC16F84A Data Sheet.
2. Справочник по среднему семейству микроконтроллеров PICmicro.
3. MPASM. Руководство пользователя.

Додаток А

Приклади підпрограм затримки

Затримка на 5 мкс

```
        movlw   .1
        movwf   Reg_1
        decfsz  Reg_1,F
        goto    $-1
        nop
```

Затримка на 10 мкс

```
        movlw   .3
        movwf   Reg_1
        decfsz  Reg_1,F
        goto    $-1
```

Затримка на 20 мкс

```
        movlw   .6
        movwf   Reg_1
        decfsz  Reg_1,F
        goto    $-1
        nop
```

Затримка на 50 мкс

```
        movlw   .16
        movwf   Reg_1
        decfsz  Reg_1,F
        goto    $-1
        nop
```

Затримка на 100 мкс

```
        movlw   .33
        movwf   Reg_1
        decfsz  Reg_1,F
        goto    $-1
```

Затримка на 200 мкс

```
        movlw   .66
        movwf   Reg_1
        decfsz  Reg_1,F
        goto    $-1
        nop
```

Затримка на 500 мс

```
        movlw   .85
        movwf   Reg_1
        movlw   .138
        movwf   Reg_2
        movlw   .3
        movwf   Reg_3
        decfsz  Reg_1,F
        goto    $-1
        decfsz  Reg_2,F
        goto    $-3
        decfsz  Reg_3,F
        goto    $-5
        nop
        nop
```

Затримка на 1 с

```
        movlw   .173
        movwf   Reg_1
        movlw   .19
        movwf   Reg_2
        movlw   .6
        movwf   Reg_3
        decfsz  Reg_1,F
        goto    $-1
        decfsz  Reg_2,F
        goto    $-3
        decfsz  Reg_3,F
        goto    $-5
        nop
        nop
```

Додаток Б

Зразок титульного аркуша для лабораторних робіт

Міністерство освіти і науки України
Національний технічний університет «Дніпровська політехніка»

Факультет інформаційних технологій
Кафедра безпеки інформації та телекомунікацій

Лабораторна робота ОТМП-1
«Робота з портами вводу–виводу інформації та створення підпрограм затримки на мові асемблер»
Варіант № 1

Виконав: ст. гр. 172-20-1
Петров Іван Петрович
Перевірив: асистент Плєц О.О.

Дніпро
НТУ «ДП»
2021

Плєц Олексій Олександрович
Олішевський Ілля Геннадійович
Кручінін Олександр Володимирович
Рибальченко Юрій Петрович

РОБОТА З ПОРТАМИ ВВОДУ–ВИВОДУ ІНФОРМАЦІЇ ТА СТВОРЕННЯ ПІДПРОГРАМ ЗАТРИМКИ

МЕТОДИЧНІ РЕКОМЕНДАЦІЇ ДО ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ ОТМП-1 З ДИСЦИПЛІНИ «ОБЧИСЛЮВАЛЬНА ТЕХНІКА ТА МІКРОПРОЦЕСОРИ» ДЛЯ СТУДЕНТІВ СПЕЦІАЛЬНОСТІ 172 ТЕЛЕКОМУНІКАЦІЇ ТА РАДІОТЕХНІКА

Редактор Ю.В. Рачковська

Підписано до друку 11.12.2020. Формат 30х42/4.
Папір офсетний. Ризографія. Ум. друк. арк. 1,6.
Обл.-вид. арк. 1,6. Тираж 8 пр. Зам. №

НТУ «Дніпровська політехніка»
49005, м. Дніпро, просп. Д. Яворницького, 19.