M. G. TECHSOLUTION SALES AND SERVICES

19, Sarasnagar,
Market Yard,
Vidyasagar Complex,
Ahmednagar–414 001 (M.S.) INDIA

Mob.: +91 9960277201, +91 9834975744;
+91 9822590931
Email - mgtechsolution2020@gmail.com
GST Number: 27DPQPS9267M1ZJ

PIC Target Board User Manual

Student Name: ______________________________

Class: ______________________________ ***Batch:*** ________

Subject: ______________________________

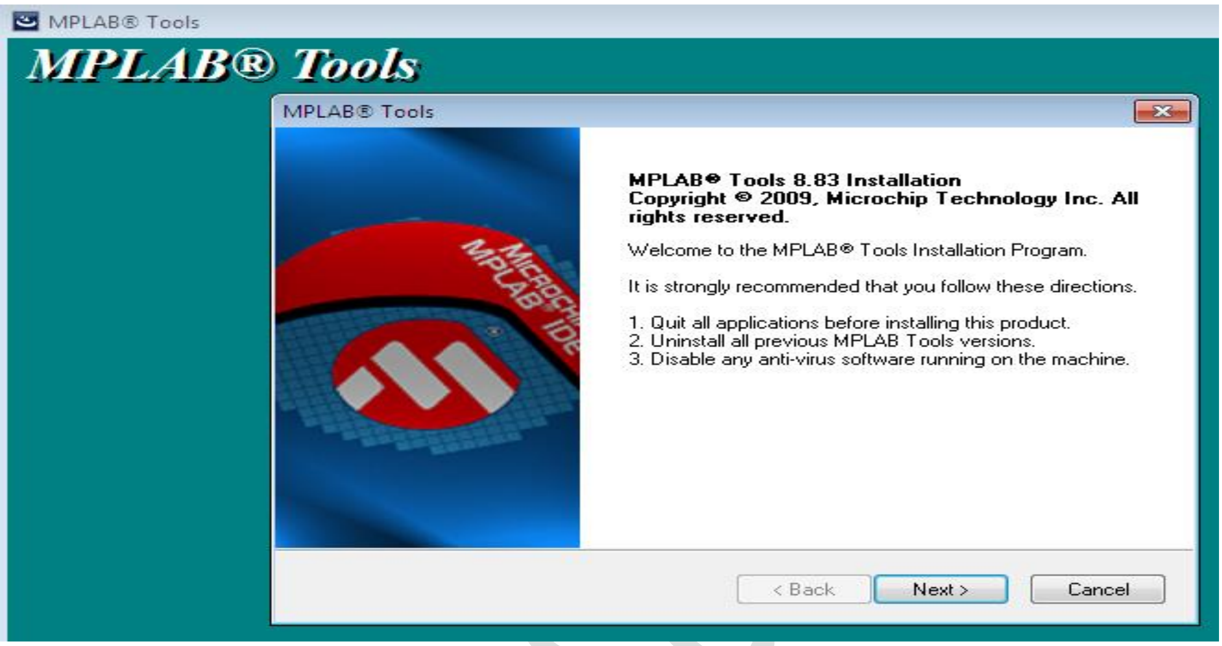
Mr. Girish Avhale (SET, NET)
Email ID: avhaleg@gmail.com
Mobile No.- 9822590931, 8237348429

Dr. M. A. Shaikh (SET, Ph.D.)
Email ID: mudassarshaikh333@gmail.com
Mobile No.- 9960277201, 9834975744

User Guide for PIC Software

Steps for Installation of MPLAB IDE software

1. Double click on setup.exe file of MPLAB IDE.
2. Click Next>.



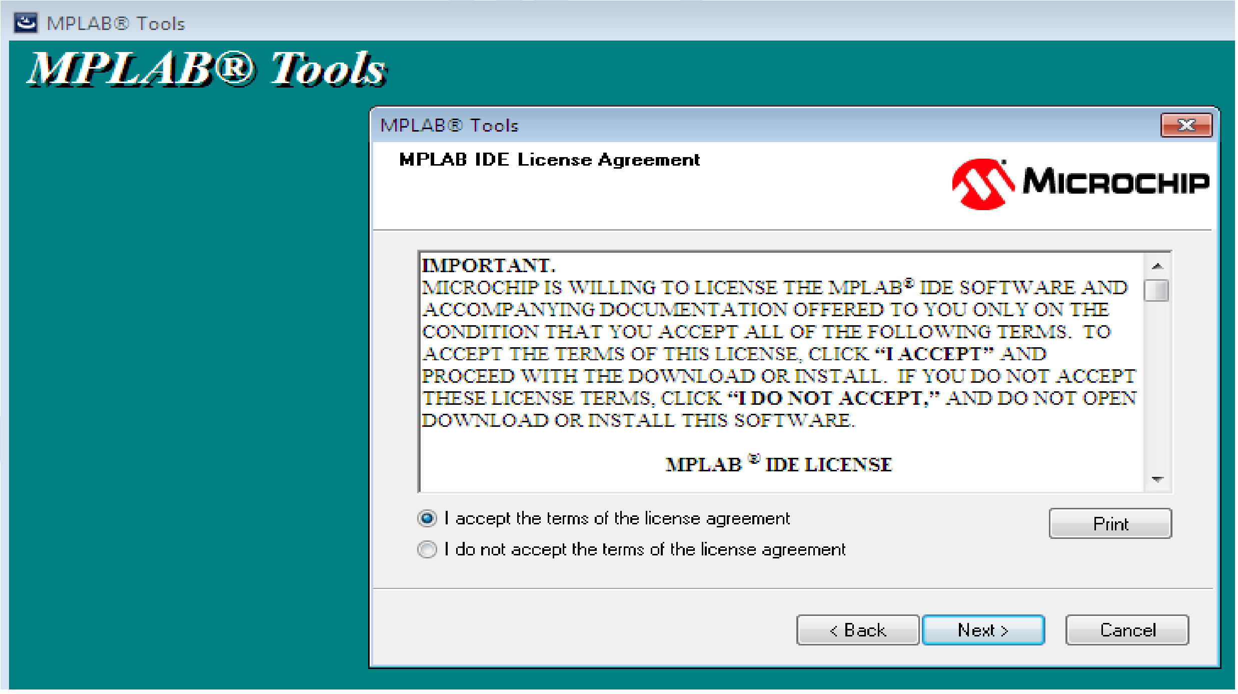
MPLAB® Tools 8.83 Installation
Copyright © 2009, Microchip Technology Inc. All rights reserved.

Welcome to the MPLAB® Tools Installation Program.

It is strongly recommended that you follow these directions.

1. Quit all applications before installing this product.
2. Uninstall all previous MPLAB Tools versions.
3. Disable any anti-virus software running on the machine.

3. Accept the license agreement and click on Next>.



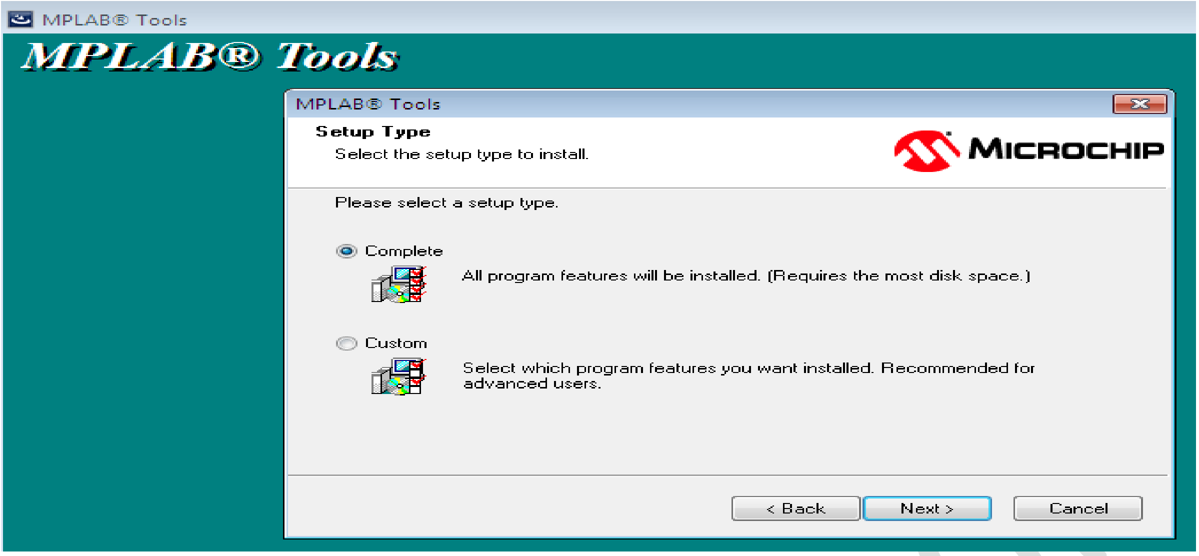
MPLAB IDE License Agreement

IMPORTANT.
MICROCHIP IS WILLING TO LICENSE THE MPLAB® IDE SOFTWARE AND ACCOMPANYING DOCUMENTATION OFFERED TO YOU ONLY ON THE CONDITION THAT YOU ACCEPT ALL OF THE FOLLOWING TERMS. TO ACCEPT THE TERMS OF THIS LICENSE, CLICK **"I ACCEPT"** AND PROCEED WITH THE DOWNLOAD OR INSTALL. IF YOU DO NOT ACCEPT THESE LICENSE TERMS, CLICK **"I DO NOT ACCEPT,"** AND DO NOT OPEN DOWNLOAD OR INSTALL THIS SOFTWARE.

MPLAB ® IDE LICENSE

● I accept the terms of the license agreement
○ I do not accept the terms of the license agreement

4. Select the setup type to install as a complete and click on Next>.



Setup Type

Select the setup type to install.

Please select a setup type.

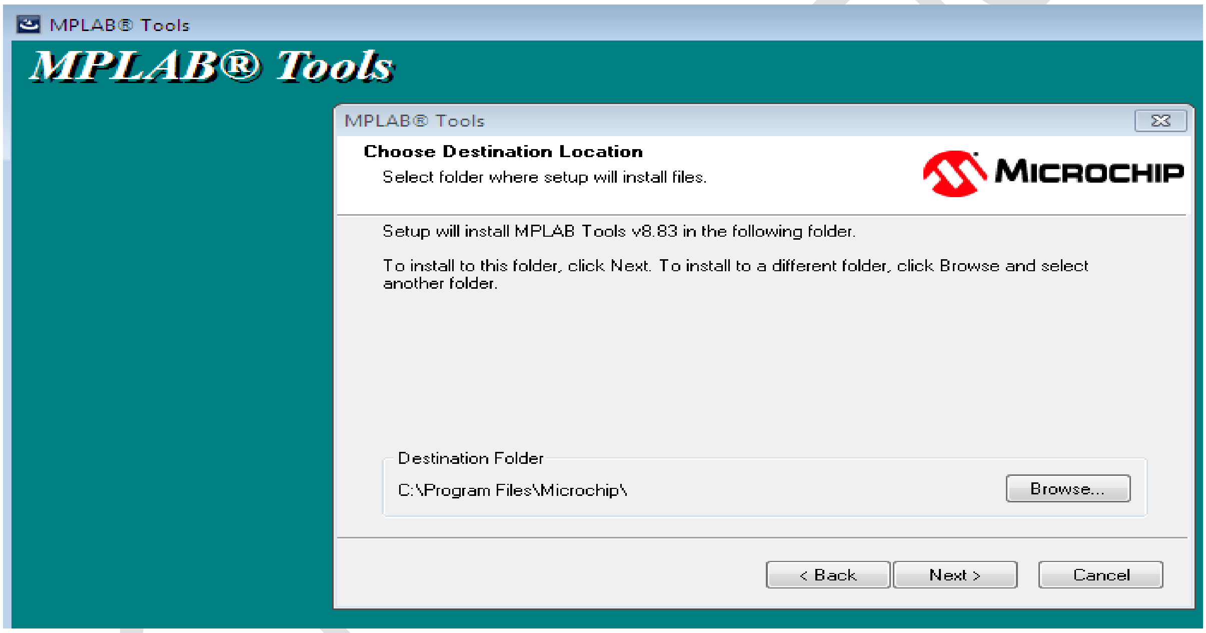
● Complete

All program features will be installed. (Requires the most disk space.)

○ Custom

Select which program features you want installed. Recommended for advanced users.

5. Select the folder for installation and click on Next>.



Choose Destination Location

Select folder where setup will install files.

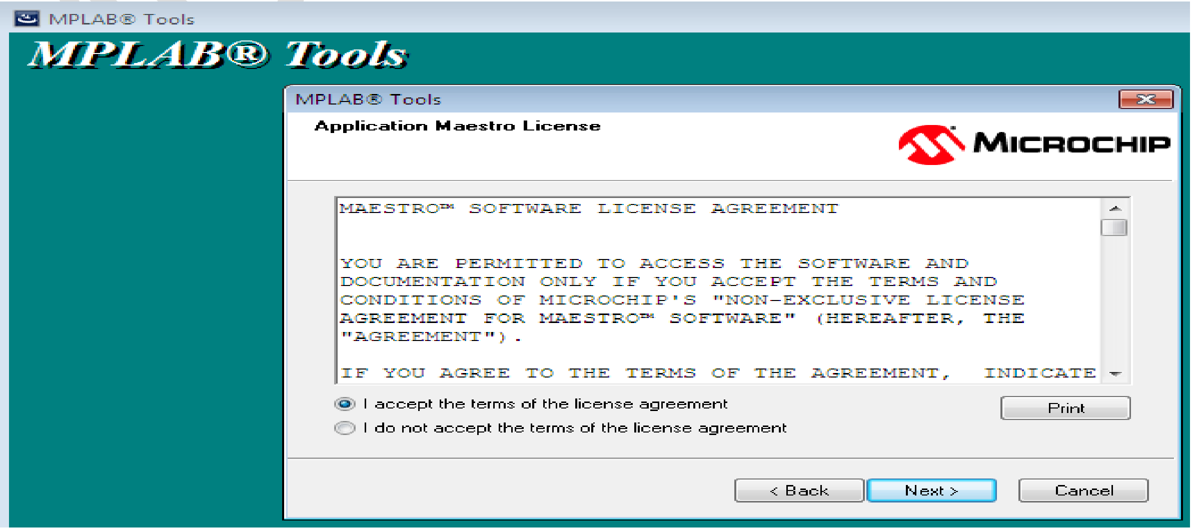
Setup will install MPLAB Tools v8.83 in the following folder.

To install to this folder, click Next. To install to a different folder, click Browse and select another folder.

Destination Folder

C:\Program Files\Microchip\

6. Accept the license agreement and click on Next>.



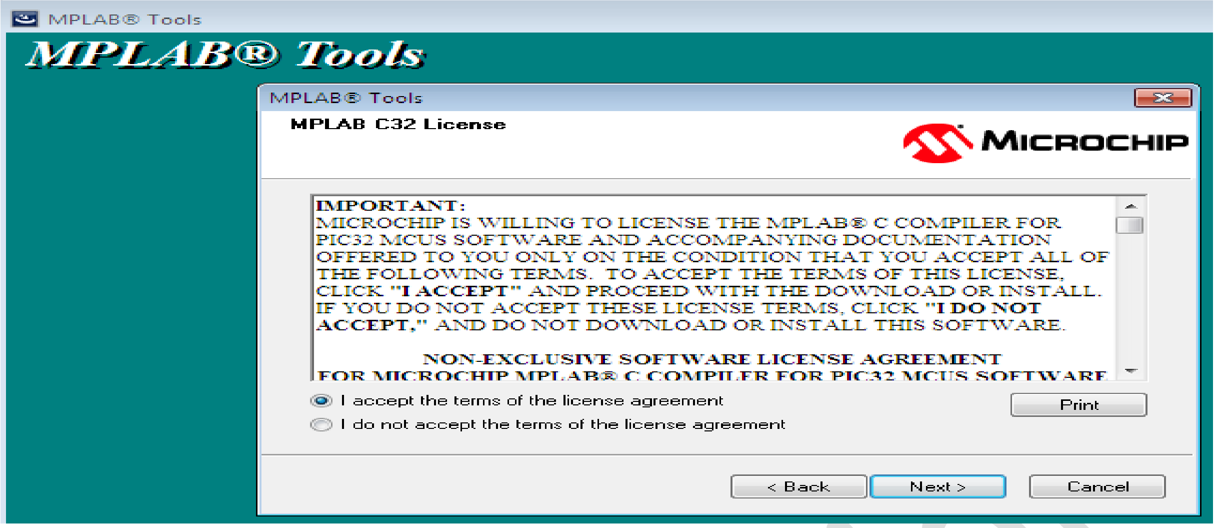
Application Maestro License

```
MAESTRO™ SOFTWARE LICENSE AGREEMENT


YOU ARE PERMITTED TO ACCESS THE SOFTWARE AND
DOCUMENTATION ONLY IF YOU ACCEPT THE TERMS AND
CONDITIONS OF MICROCHIP'S "NON-EXCLUSIVE LICENSE
AGREEMENT FOR MAESTRO™ SOFTWARE" (HEREAFTER, THE
"AGREEMENT").

IF YOU AGREE TO THE TERMS OF THE AGREEMENT,  INDICATE
```

● I accept the terms of the license agreement

○ I do not accept the terms of the license agreement

7. Accept the license agreement for MPLAB C32 and click on Next>.



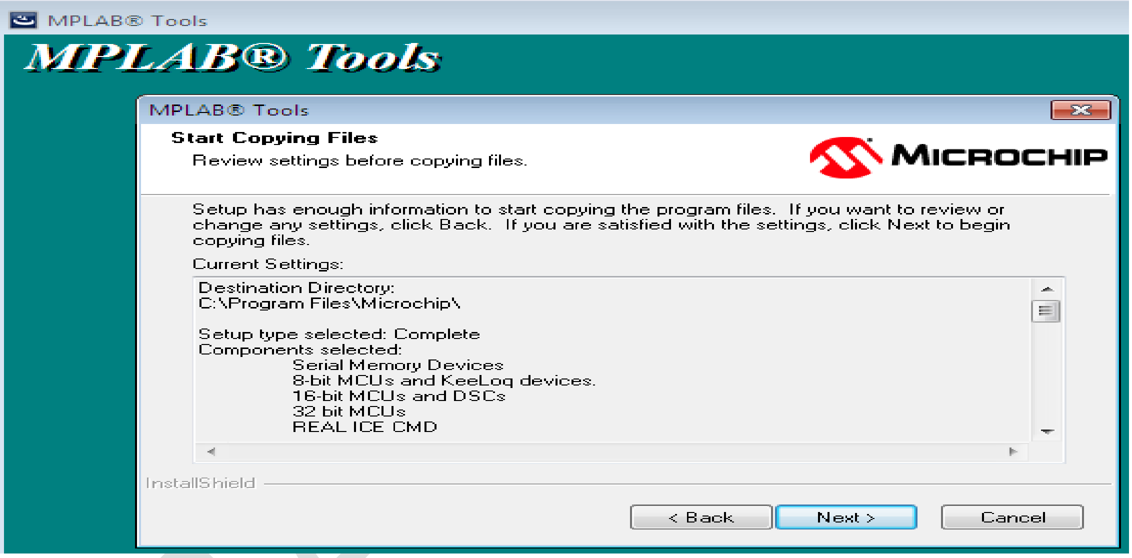
MPLAB C32 License

IMPORTANT:
MICROCHIP IS WILLING TO LICENSE THE MPLAB® C COMPILER FOR PIC32 MCUS SOFTWARE AND ACCOMPANYING DOCUMENTATION OFFERED TO YOU ONLY ON THE CONDITION THAT YOU ACCEPT ALL OF THE FOLLOWING TERMS. TO ACCEPT THE TERMS OF THIS LICENSE, CLICK "I ACCEPT" AND PROCEED WITH THE DOWNLOAD OR INSTALL. IF YOU DO NOT ACCEPT THESE LICENSE TERMS, CLICK "I DO NOT ACCEPT," AND DO NOT DOWNLOAD OR INSTALL THIS SOFTWARE.

NON-EXCLUSIVE SOFTWARE LICENSE AGREEMENT
FOR MICROCHIP MPLAB® C COMPILER FOR PIC32 MCUS SOFTWARE

● I accept the terms of the license agreement
○ I do not accept the terms of the license agreement

8. Review settings before copping file and click on Next>.



Start Copying Files
Review settings before copying files.

Setup has enough information to start copying the program files. If you want to review or change any settings, click Back. If you are satisfied with the settings, click Next to begin copying files.

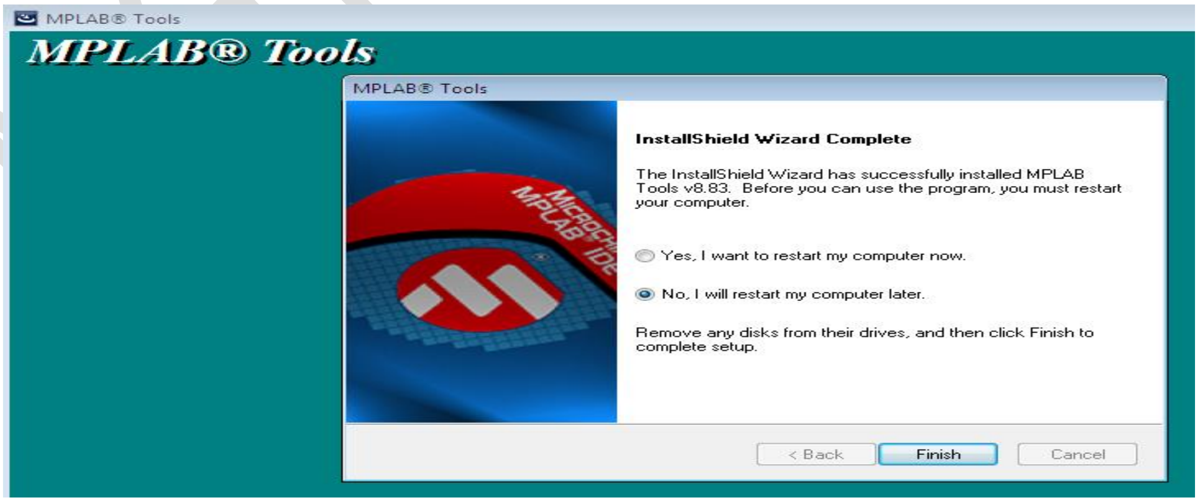
Current Settings:

Destination Directory:
C:\Program Files\Microchip\

Setup type selected: Complete
Components selected:
Serial Memory Devices
8-bit MCUs and KeeLoq devices.
16-bit MCUs and DSCs
32 bit MCUs
REAL ICE CMD

9. Installation begins.

10. After installation is complete, click on Finish.



InstallShield Wizard Complete

The InstallShield Wizard has successfully installed MPLAB Tools v8.83. Before you can use the program, you must restart your computer.

○ Yes, I want to restart my computer now.

● No, I will restart my computer later.

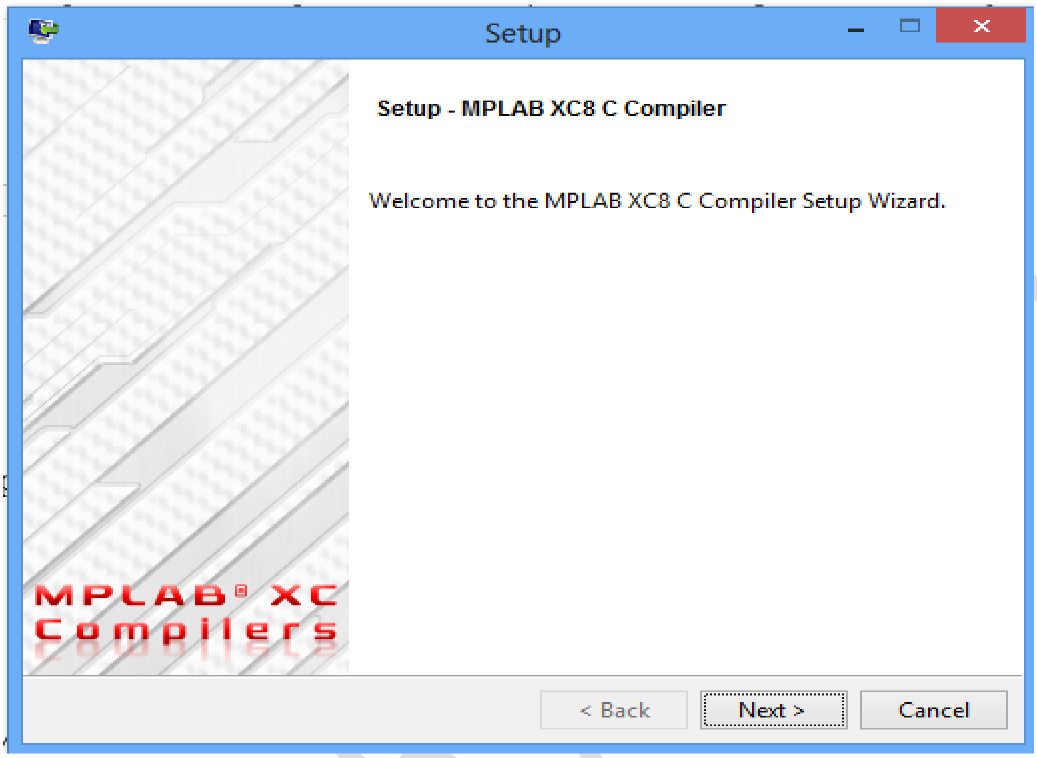
Remove any disks from their drives, and then click Finish to complete setup.

11. The icon for MPLAB IDE is appeared on the desktop.

---********--

Steps for installation of C-compiler

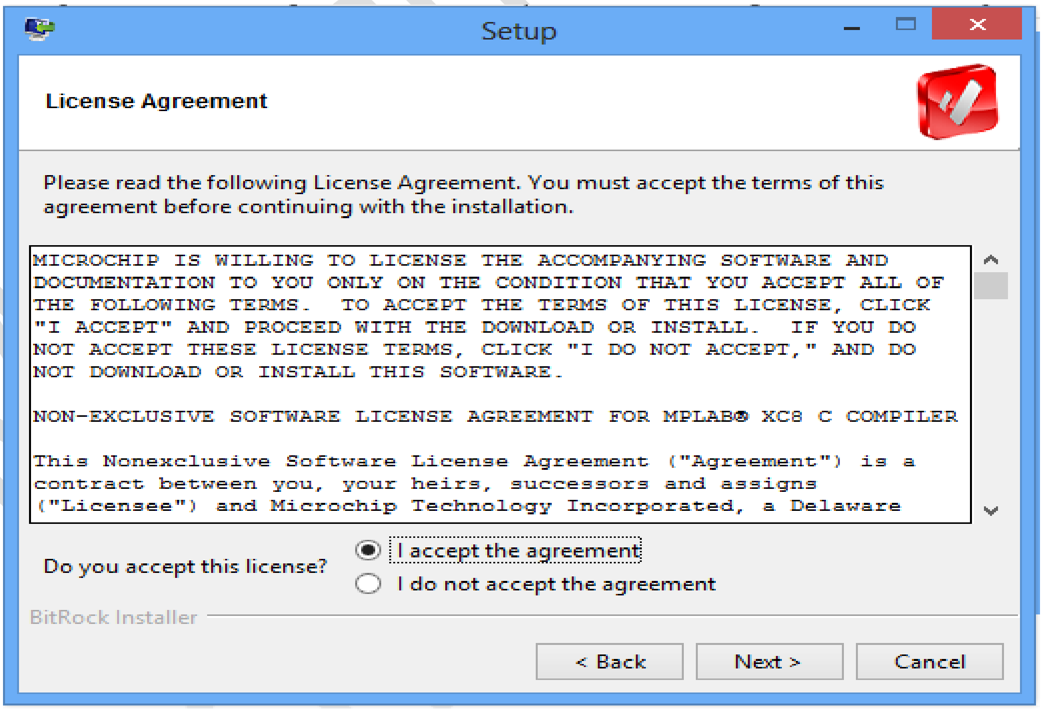
1. Double click on application file of XC8 compiler.
2. Click Next>.



Setup - MPLAB XC8 C Compiler

Welcome to the MPLAB XC8 C Compiler Setup Wizard.

3. Select “I accept the agreement” and Click on Next.



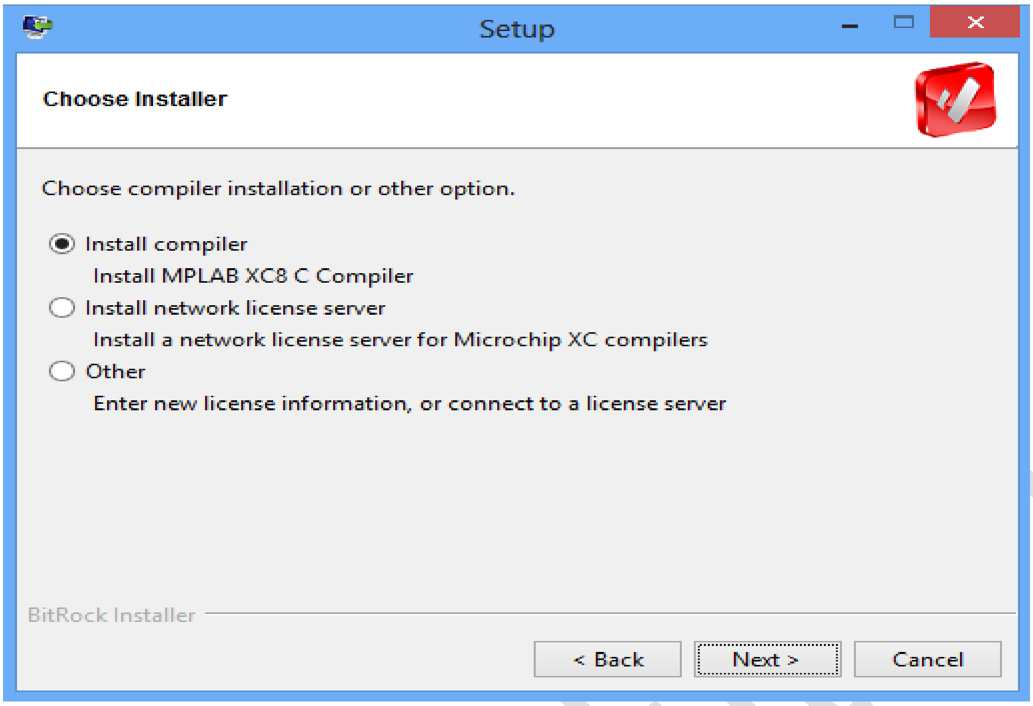
License Agreement

Please read the following License Agreement. You must accept the terms of this agreement before continuing with the installation.

```
MICROCHIP IS WILLING TO LICENSE THE ACCOMPANYING SOFTWARE AND
DOCUMENTATION TO YOU ONLY ON THE CONDITION THAT YOU ACCEPT ALL OF
THE FOLLOWING TERMS.  TO ACCEPT THE TERMS OF THIS LICENSE, CLICK
"I ACCEPT" AND PROCEED WITH THE DOWNLOAD OR INSTALL.  IF YOU DO
NOT ACCEPT THESE LICENSE TERMS, CLICK "I DO NOT ACCEPT," AND DO
NOT DOWNLOAD OR INSTALL THIS SOFTWARE.

NON-EXCLUSIVE SOFTWARE LICENSE AGREEMENT FOR MPLAB® XC8 C COMPILER

This Nonexclusive Software License Agreement ("Agreement") is a
contract between you, your heirs, successors and assigns
("Licensee") and Microchip Technology Incorporated, a Delaware
```

Do you accept this license? ● I accept the agreement ○ I do not accept the agreement

4. Select “Install Compiler” option and click on next.


Setup

Choose Installer

Choose compiler installation or other option.

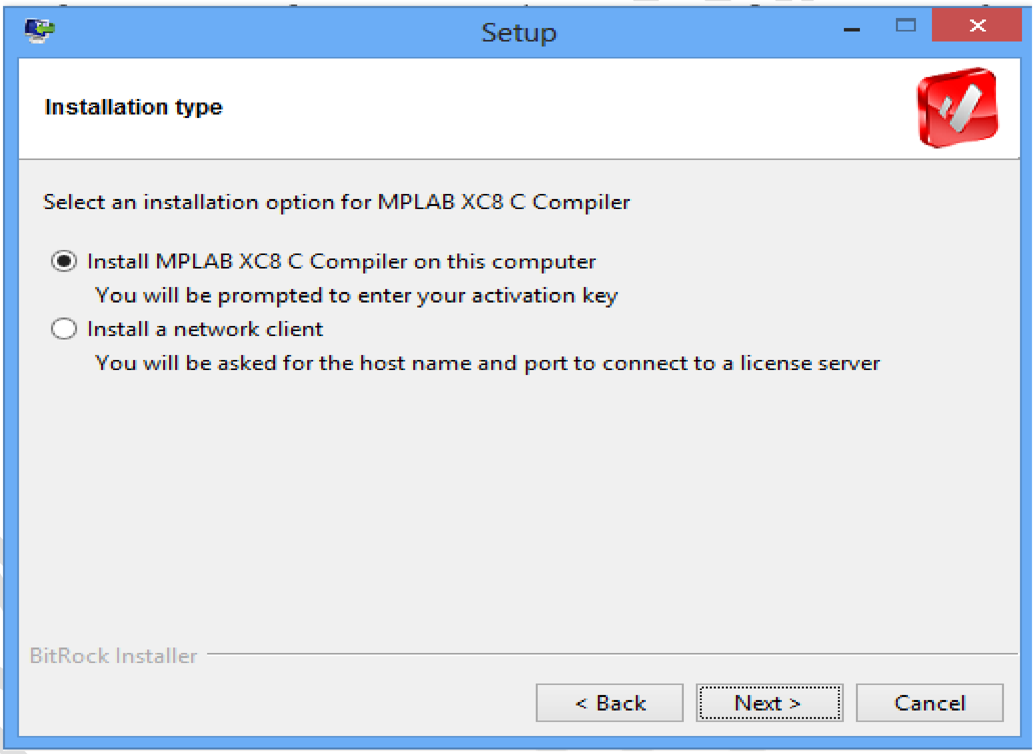
● Install compiler
Install MPLAB XC8 C Compiler

○ Install network license server
Install a network license server for Microchip XC compilers

○ Other
Enter new license information, or connect to a license server

BitRock Installer

5. Select first option and click on Next.


Setup

Installation type

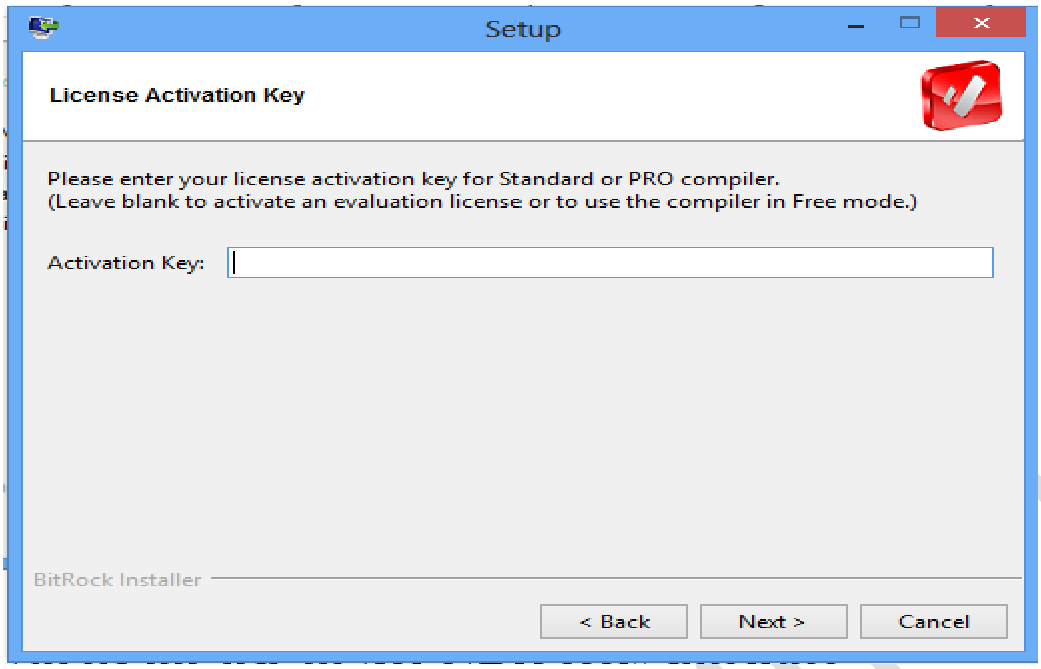
Select an installation option for MPLAB XC8 C Compiler

● Install MPLAB XC8 C Compiler on this computer
You will be prompted to enter your activation key

○ Install a network client
You will be asked for the host name and port to connect to a license server

BitRock Installer

6. Click on Next without any Activation key.



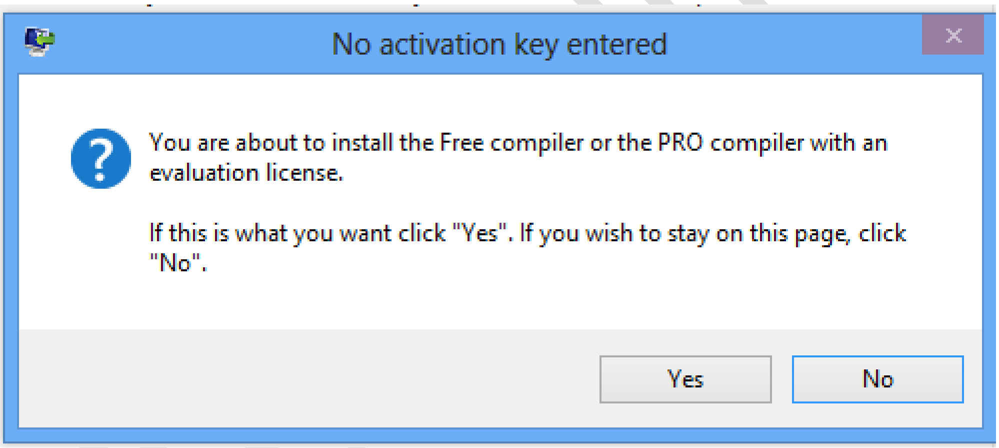
Setup

License Activation Key

Please enter your license activation key for Standard or PRO compiler.
(Leave blank to activate an evaluation license or to use the compiler in Free mode.)

Activation Key:

7. Select “Yes” to install evaluation license.

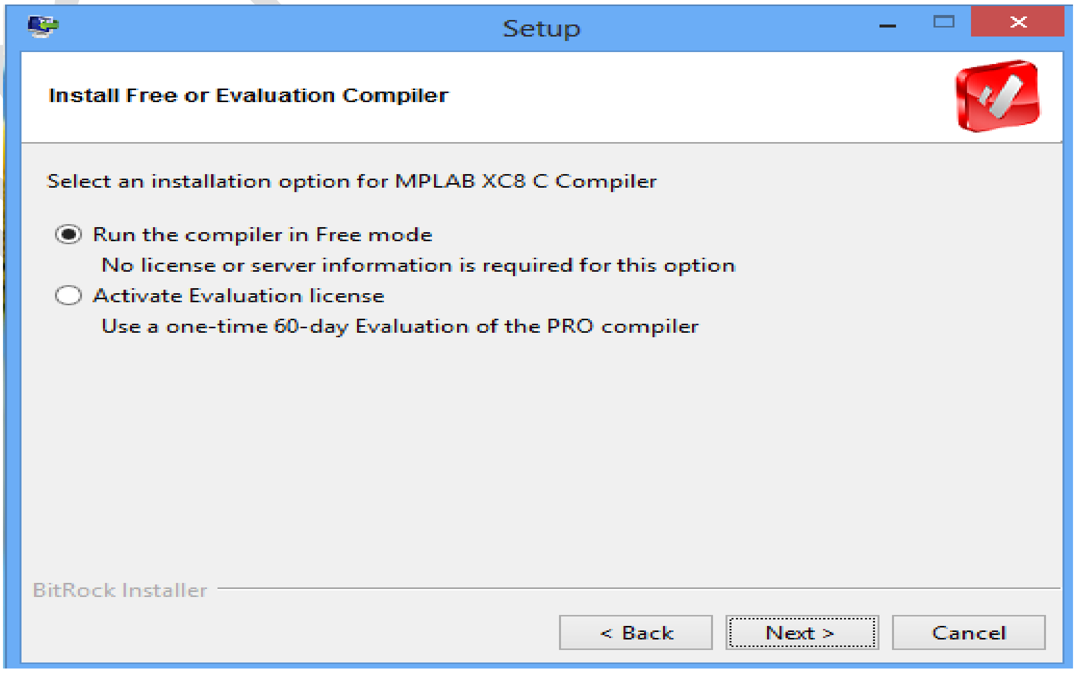


No activation key entered

You are about to install the Free compiler or the PRO compiler with an evaluation license.

If this is what you want click "Yes". If you wish to stay on this page, click "No".

8. Select first option and click on Next.



Setup

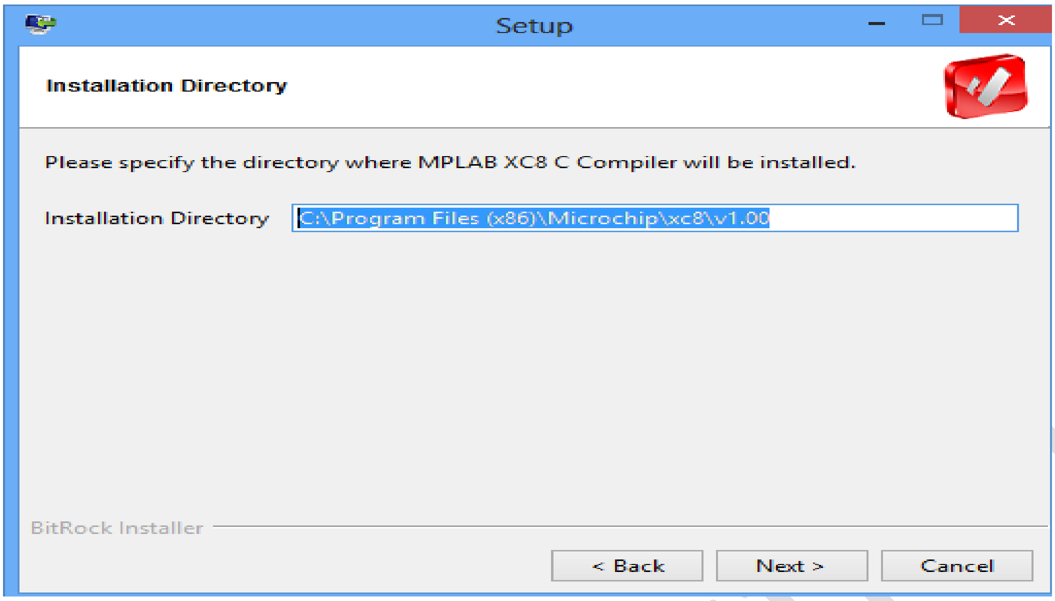
Install Free or Evaluation Compiler

Select an installation option for MPLAB XC8 C Compiler

● Run the compiler in Free mode
No license or server information is required for this option

○ Activate Evaluation license
Use a one-time 60-day Evaluation of the PRO compiler

9. Select installation directory and click on Next.

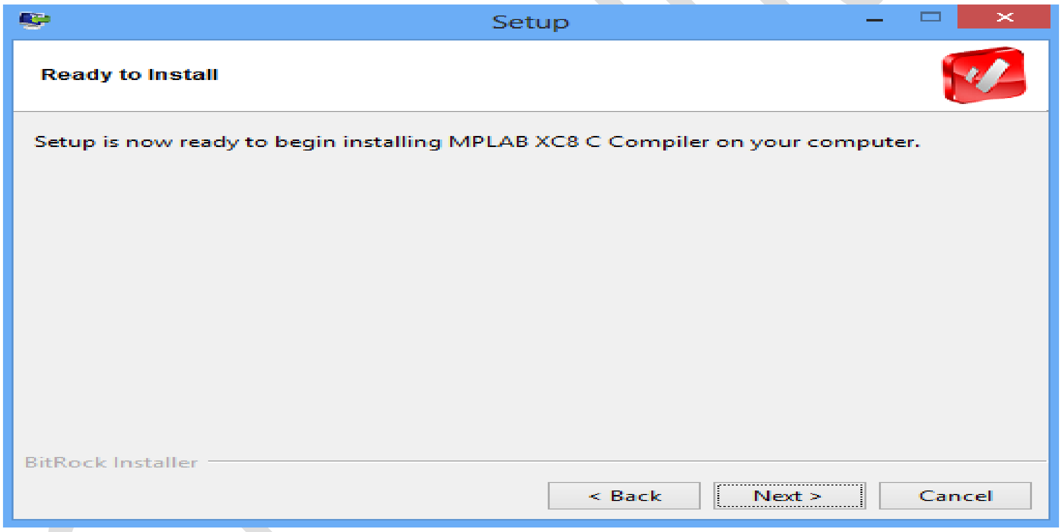


Installation Directory

Please specify the directory where MPLAB XC8 C Compiler will be installed.

Installation Directory: C:\Program Files (x86)\Microchip\xc8\v1.00

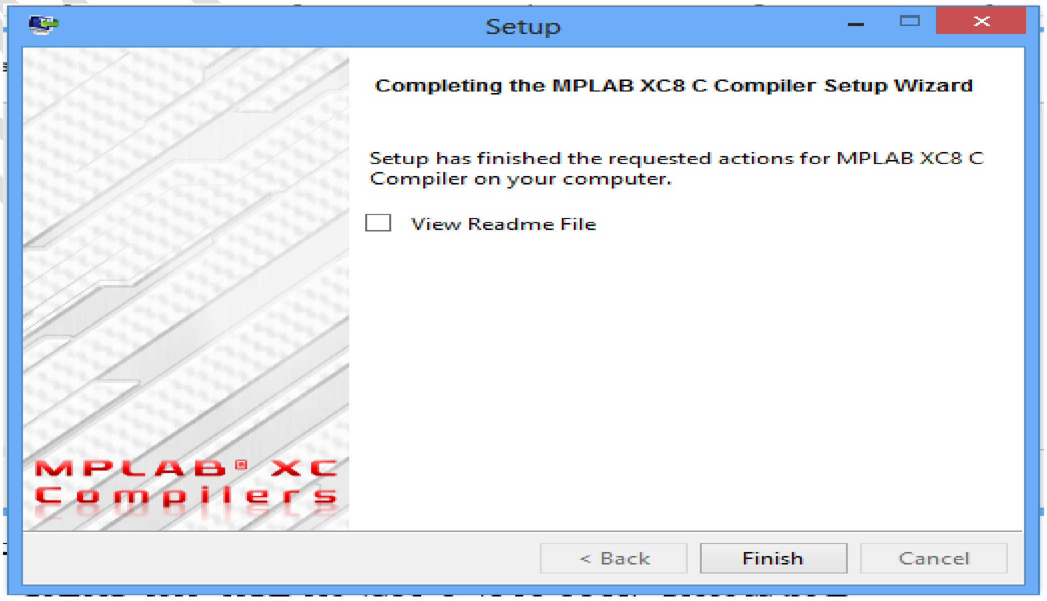
10. Click on Next.



Ready to Install

Setup is now ready to begin installing MPLAB XC8 C Compiler on your computer.

11. After installation completed, click on Finish.



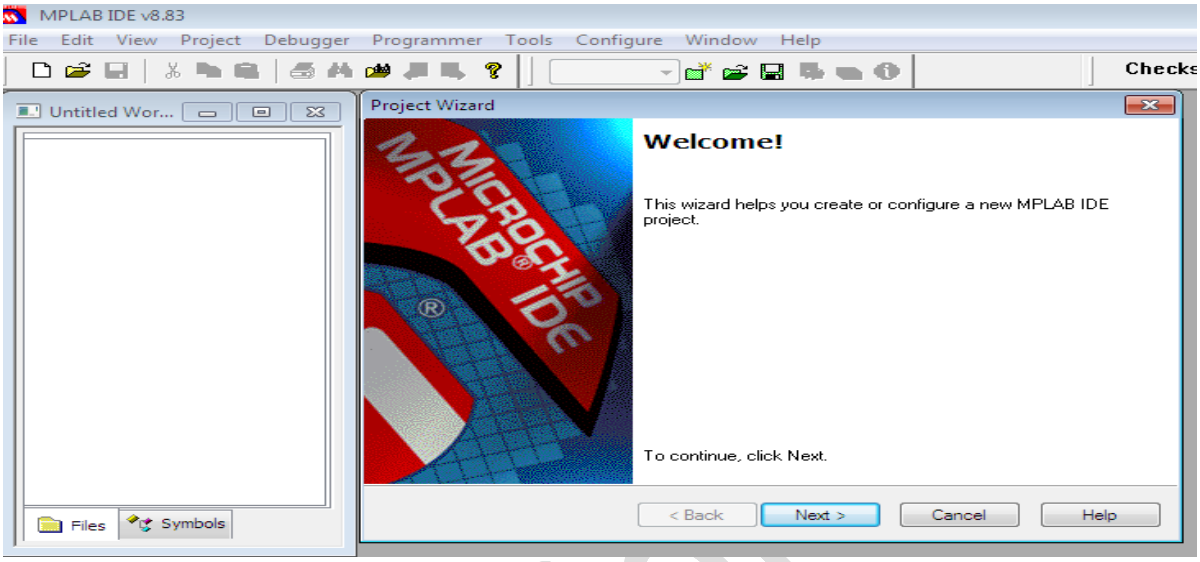
Completing the MPLAB XC8 C Compiler Setup Wizard

Setup has finished the requested actions for MPLAB XC8 C Compiler on your computer.

□ View Readme File

Steps for use of MPLAB IDE software

1. Double click on MPLAB IDE v8.83 icon on the desktop.
2. Close the project if any open. Go to Project -> Close Project.
3. To create new project, go to Project -> Project Wizard. Project Wizard window will open.

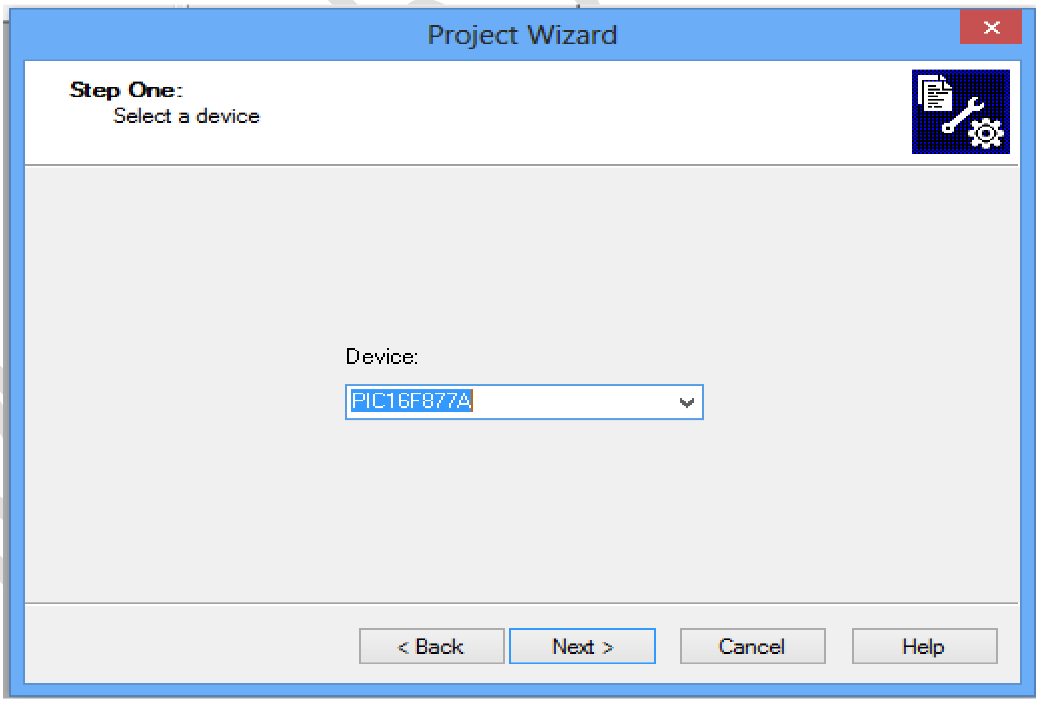


Welcome!

This wizard helps you create or configure a new MPLAB IDE project.

To continue, click Next.

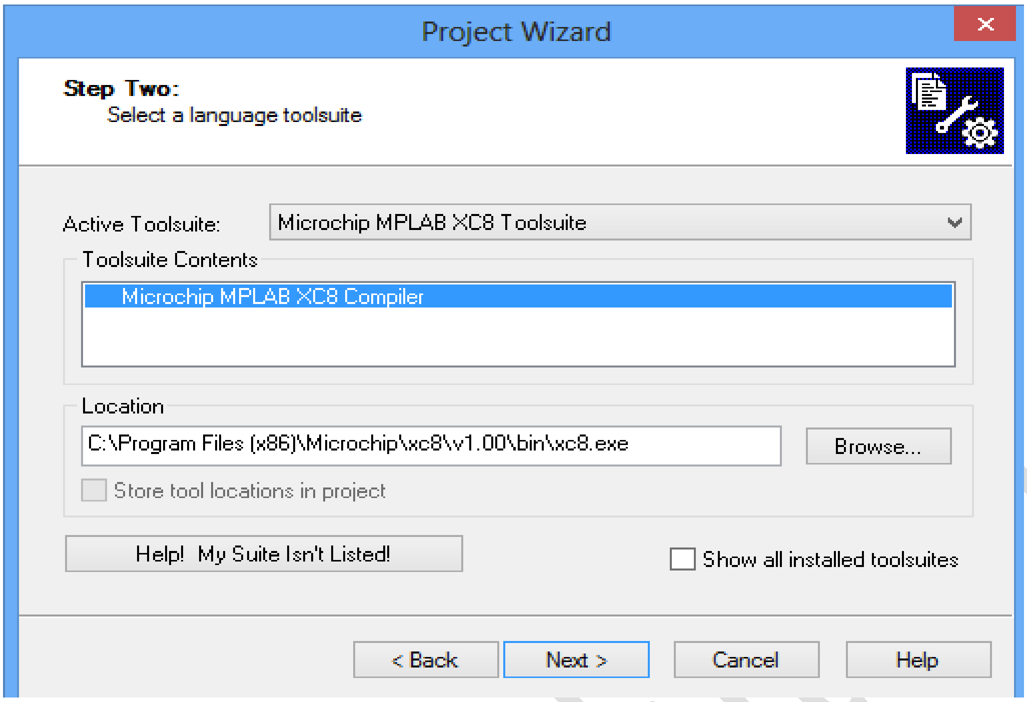
4. To continue, click on Next>.
5. Select device -> PIC16F877A. Click on Next>.



Step One:
Select a device

Device: PIC16F877A

6. Select a Language Tool suite. For C language select Microchip MPLAB XC8 Tool suite. Click on Next>.



Project Wizard

Step Two:
Select a language toolsuite

Active Toolsuite: Microchip MPLAB XC8 Toolsuite

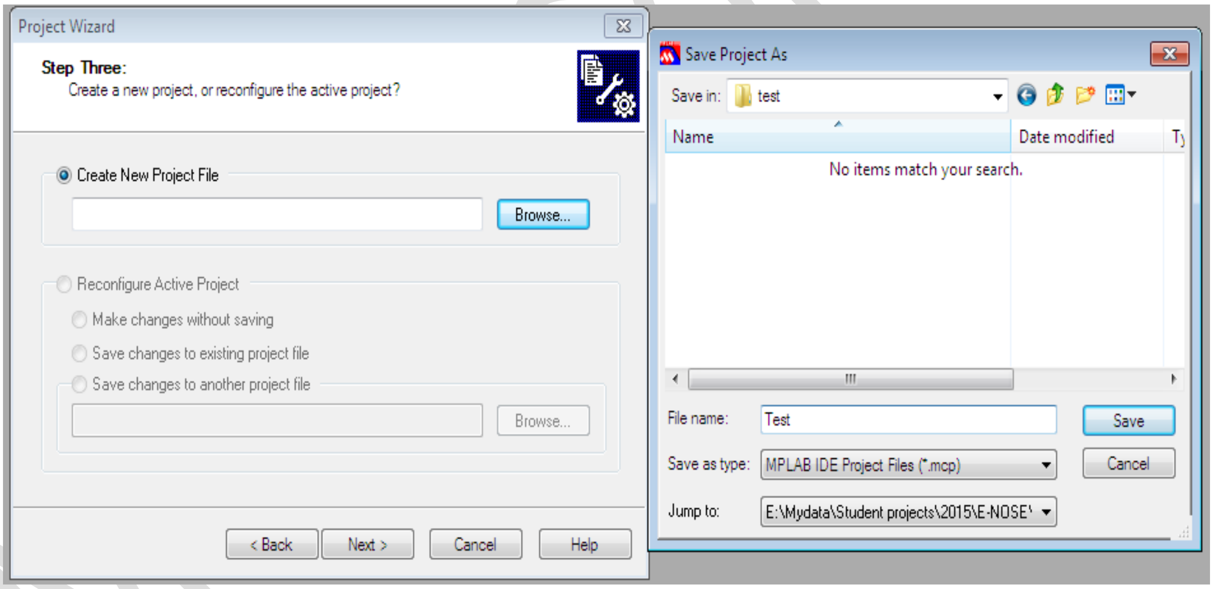
Toolsuite Contents: Microchip MPLAB XC8 Compiler

Location: C:\Program Files (x86)\Microchip\xc8\v1.00\bin\xc8.exe

□ Store tool locations in project

□ Show all installed toolsuites

7. Click on Browse select the destination folder for new project, give name and save. Then click on Next>.



Project Wizard

Step Three:
Create a new project, or reconfigure the active project?

● Create New Project File

○ Reconfigure Active Project

○ Make changes without saving

○ Save changes to existing project file

○ Save changes to another project file

Save Project As

Save in: test

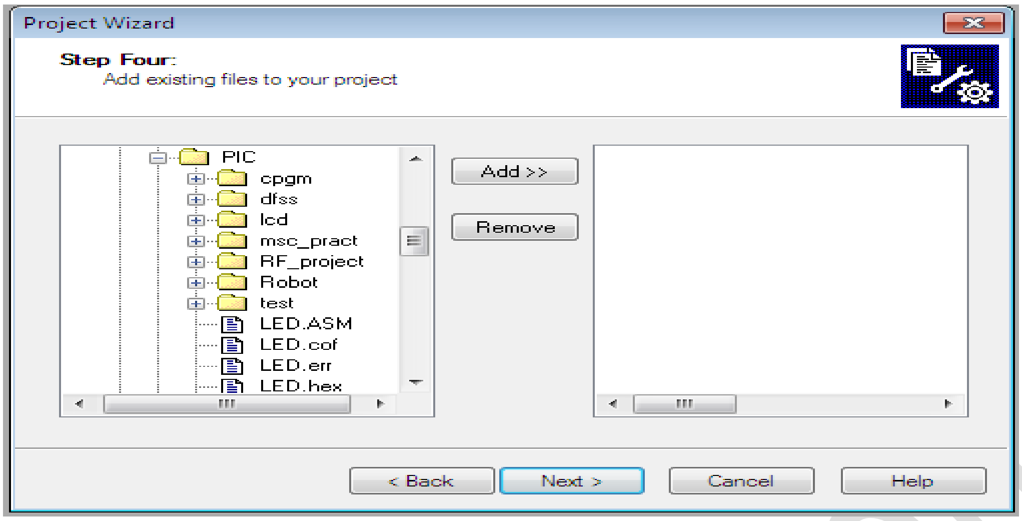
Name	Date modified	Ty
No items match your search.		

File name: Test

Save as type: MPLAB IDE Project Files (*.mcp)

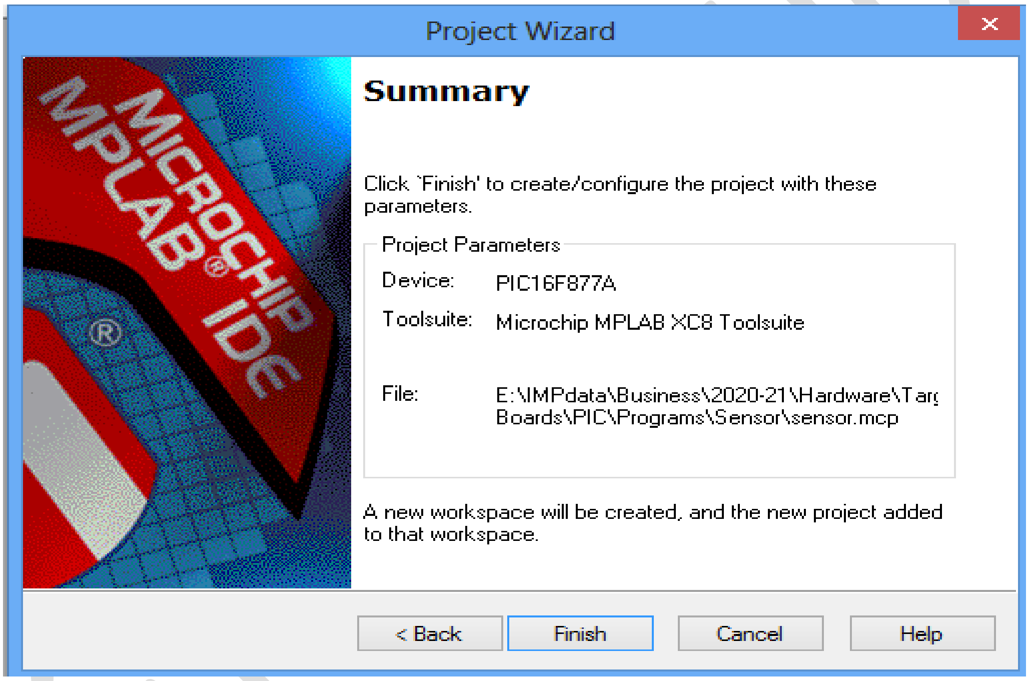
Jump to: E:\Mydata\Student projects\2015\E-NOSE\

8. Add existing files to your project (if any). Click on Next>.



Step Four:
Add existing files to your project

9. Check summary and click on Finish.



Summary

Click `Finish' to create/configure the project with these parameters.

Project Parameters

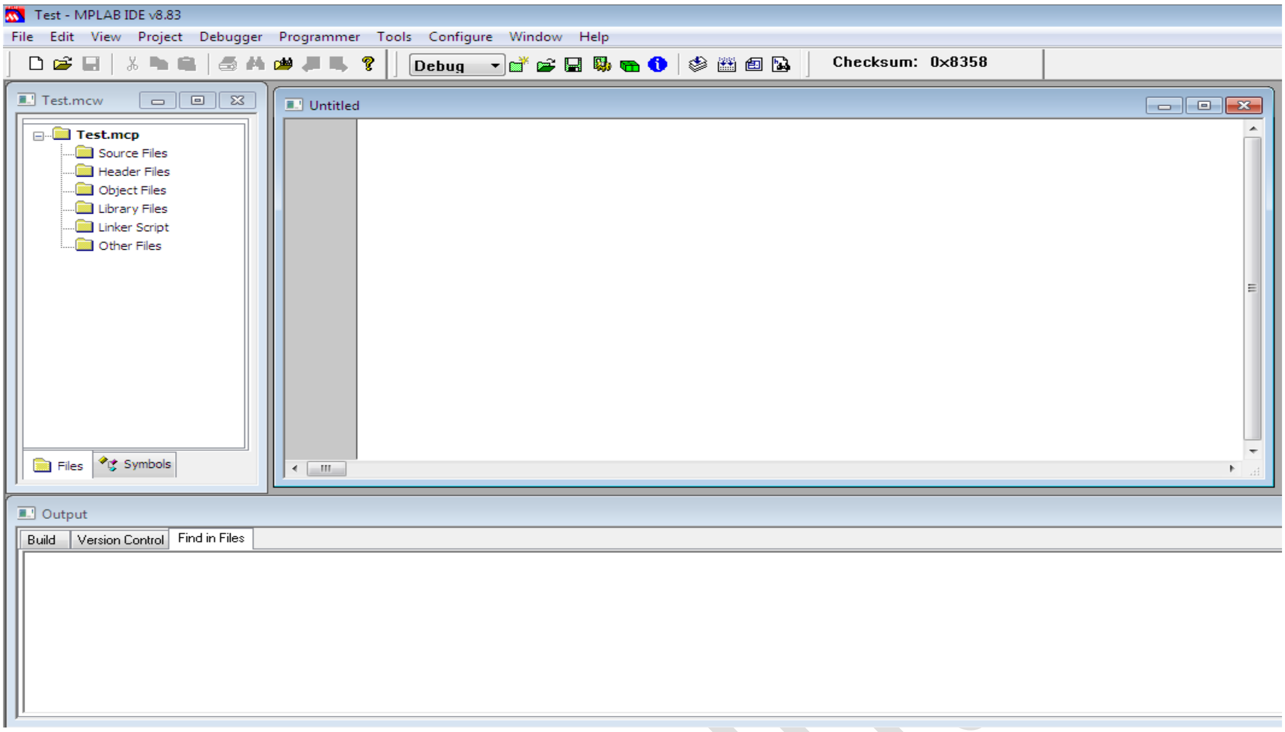
Device: PIC16F877A

Toolsuite: Microchip MPLAB XC8 Toolsuite

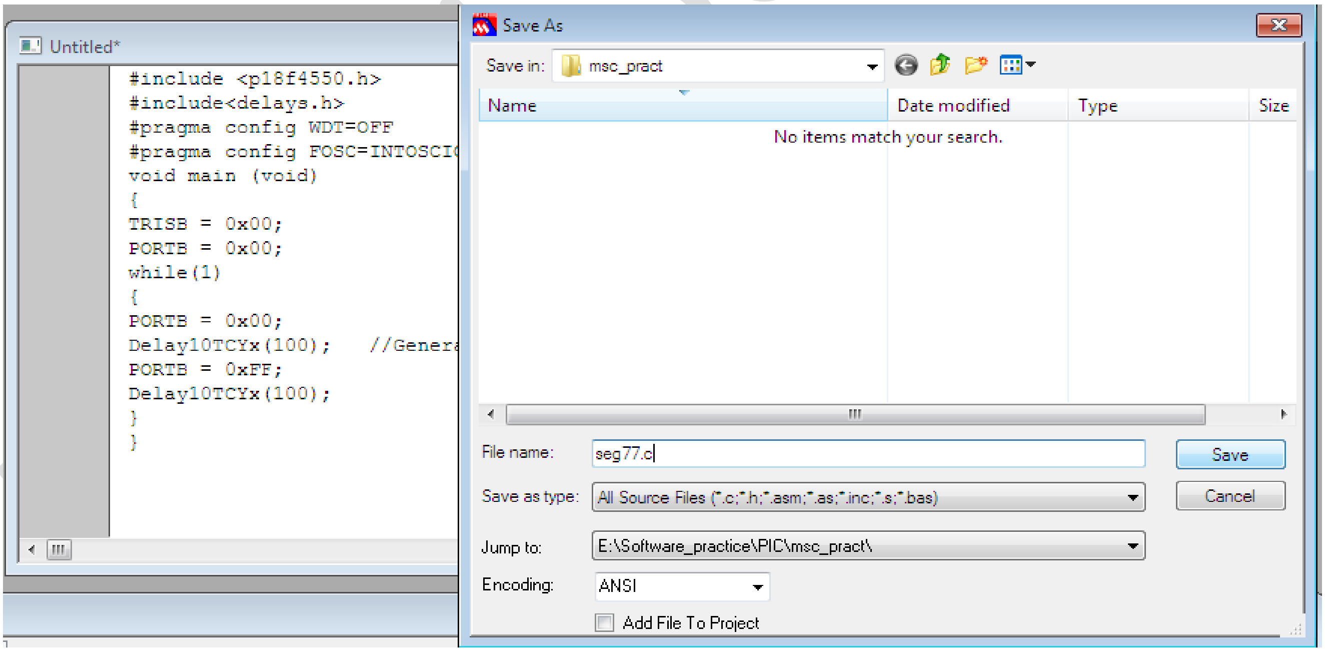
File: E:\IMPdata\Business\2020-21\Hardware\Tar(Boards\PIC\Programs\Sensor\sensor.mcp

A new workspace will be created, and the new project added to that workspace.

10. New project is created. Now there are two different windows are open: Project window & output window.
11. To create new program file go to File-> New. The three windows shown below:



12. Write a program in program file. Then to save the file go to File-> Save. Select the same folder where project is save and give the file name with **.c extension** if program is written in 'C' language and with **.asm extension** if program written in assembly language.



```
#include <p18f4550.h>
#include<delays.h>
#pragma config WDT=OFF
#pragma config FOSC=INTOSCI
void main (void)
{
TRISB = 0x00;
PORTB = 0x00;
while(1)
{
PORTB = 0x00;
Delay10TCYx(100);    //Genera
PORTB = 0xFF;
Delay10TCYx(100);
}
}
```

File name: seg77.c

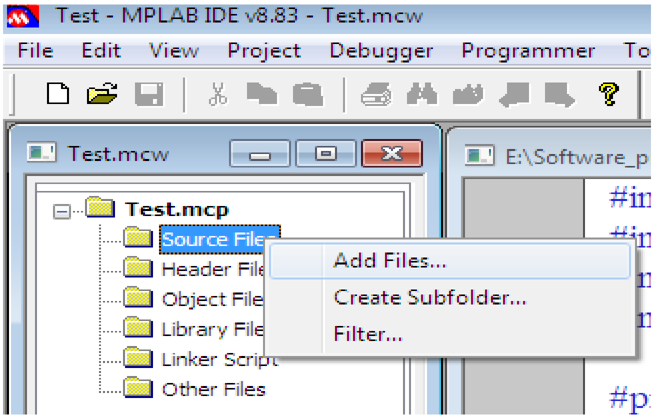
Save as type: All Source Files (*.c;*.h;*.asm;*.as;*.inc;*.s;*.bas)

Jump to: E:\Software_practice\PIC\msc_pract\

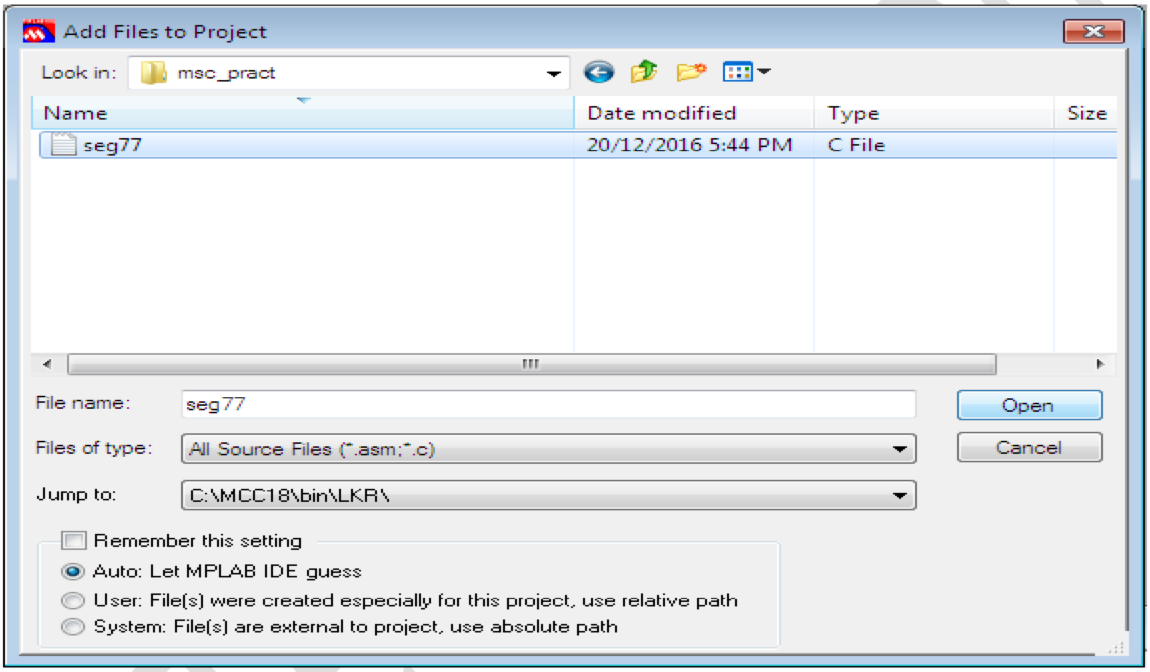
Encoding: ANSI

13. After program is save successfully it becomes colorful.

14. Then go to project window and right click on **Source Files**, select Add Files….



15. Then select the **.asm or .c** file saved in previous steps from the same folder. Click on Open.



Look in: msc_pract

Name	Date modified	Type	Size
seg77	20/12/2016 5:44 PM	C File	

File name: seg77

Files of type: All Source Files (*.asm;*.c)

Jump to: C:\MCC18\bin\LKR\

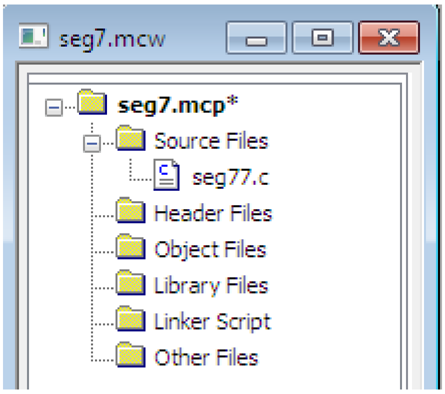
□ Remember this setting

● Auto: Let MPLAB IDE guess

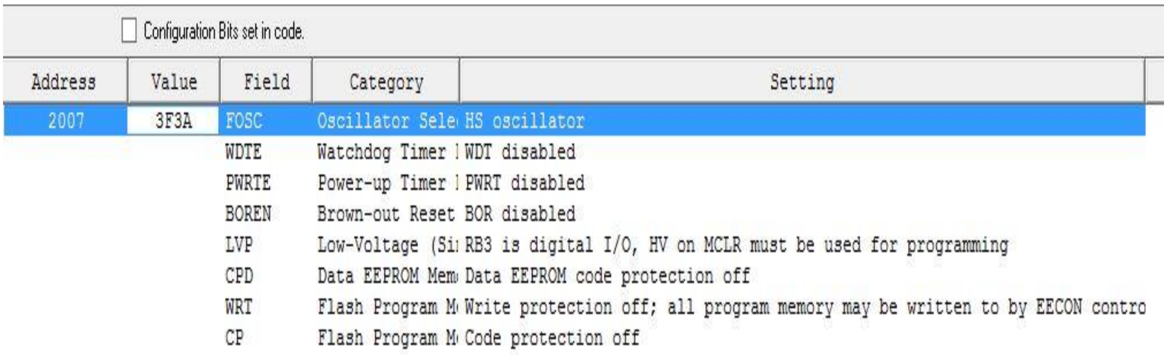
○ User: File(s) were created especially for this project, use relative path

○ System: File(s) are external to project, use absolute path

16. After program file is added, Project window looks like below.



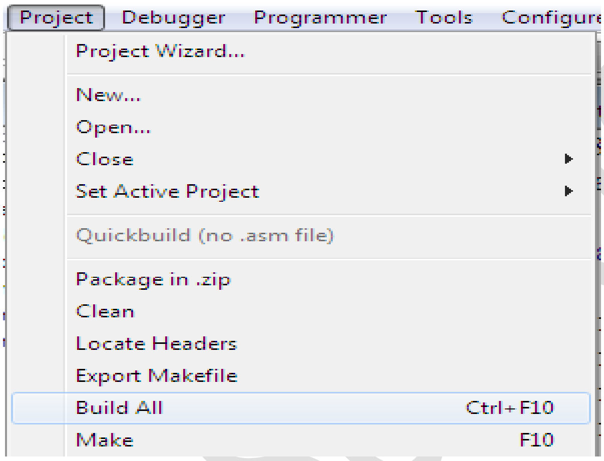
17. After all this perform the configuration settings. Go to Configure -> Configuration bits and perform following settings.



□ Configuration Bits set in code.

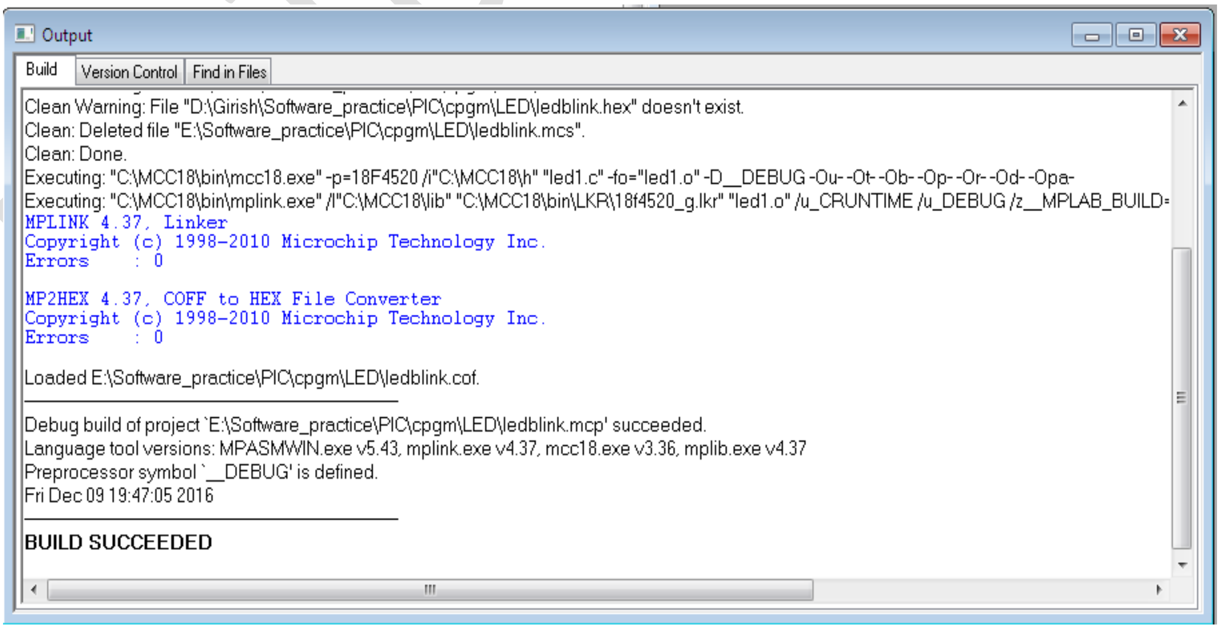
Address	Value	Field	Category	Setting
2007	3F3A	FOSC	Oscillator Sele	HS oscillator
		WDTE	Watchdog Timer !	WDT disabled
		PWRTE	Power-up Timer !	PWRT disabled
		BOREN	Brown-out Reset	BOR disabled
		LVP	Low-Voltage (Si	RB3 is digital I/O, HV on MCLR must be used for programming
		CPD	Data EEPROM Mem	Data EEPROM code protection off
		WRT	Flash Program M	Write protection off; all program memory may be written to by EECON contro
		CP	Flash Program M	Code protection off

18. Go to Project-> Build All.



19. After project is build, it check for any errors by the compiler and if there are no errors it is converted into machine language and save in a file by .hex extension in the same folder where project is save.

20. The output window shows the errors or warnings if any; if not it shows the message **BUILD SUCCEEDED**.

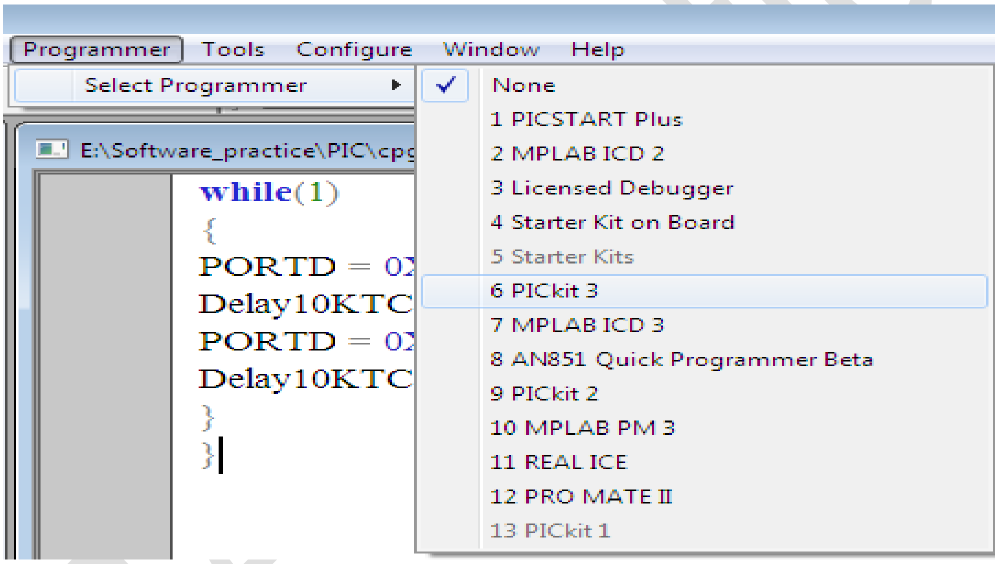


```
Clean Warning: File "D:\Girish\Software_practice\PIC\cpgm\LED\ledblink.hex" doesn't exist.
Clean: Deleted file "E:\Software_practice\PIC\cpgm\LED\ledblink.mcs".
Clean: Done.
Executing: "C:\MCC18\bin\mcc18.exe" -p=18F4520 /i"C:\MCC18\h" "led1.c" -fo="led1.o" -D__DEBUG -Ou- -Ot- -Ob- -Op- -Or- -Od- -Opa-
Executing: "C:\MCC18\bin\mplink.exe" /l"C:\MCC18\lib" "C:\MCC18\bin\LKR\18f4520_g.lkr" "led1.o" /u_CRUNTIME /u_DEBUG /z__MPLAB_BUILD=
MPLINK 4.37, Linker
Copyright (c) 1998-2010 Microchip Technology Inc.
Errors    : 0

MP2HEX 4.37, COFF to HEX File Converter
Copyright (c) 1998-2010 Microchip Technology Inc.
Errors    : 0

Loaded E:\Software_practice\PIC\cpgm\LED\ledblink.cof.
----------------------------------------------------------------------
Debug build of project `E:\Software_practice\PIC\cpgm\LED\ledblink.mcp' succeeded.
Language tool versions: MPASMWIN.exe v5.43, mplink.exe v4.37, mcc18.exe v3.36, mplib.exe v4.37
Preprocessor symbol `__DEBUG' is defined.
Fri Dec 09 19:47:05 2016
----------------------------------------------------------------------
BUILD SUCCEEDED
```

21. After program is successfully build and .hex file is created, its time load this .hex file into the program memory of microcontroller using a programmer.

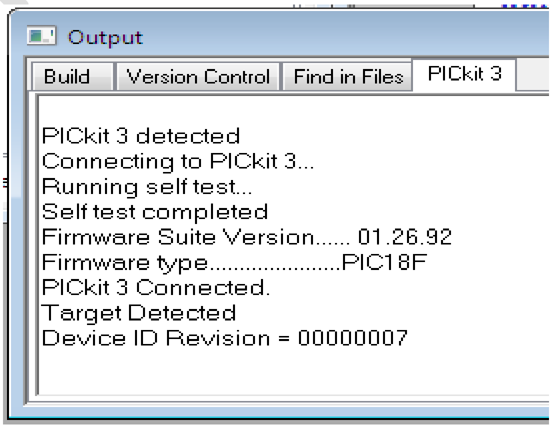
22. Connect the PICkit3 programmer to the target board. Make power supply ON.

23. To select a programmer go to Programmer -> Select Programmer -> **PICkit3**.



```
while(1)
{
PORTD = 0x
Delay10KTC
PORTD = 0x
Delay10KTC
}
}
```

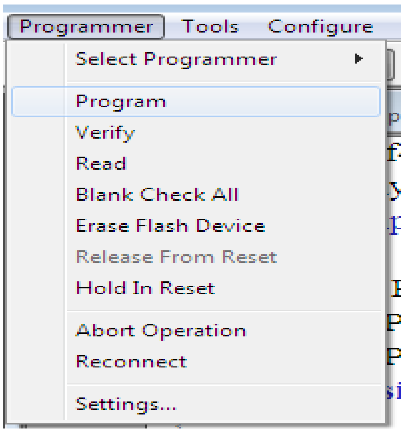
24. If programmer is connected to the device successfully and target device is found, the output window shows the following message.



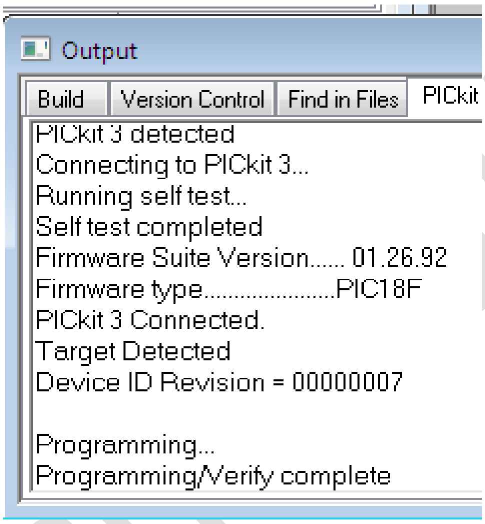
```
PICkit 3 detected
Connecting to PICkit 3...
Running self test...
Self test completed
Firmware Suite Version...... 01.26.92
Firmware type.....................PIC18F
PICkit 3 Connected.
Target Detected
Device ID Revision = 00000007
```

25. If there is any error while connecting programmer it shows in output window.

26. After connecting the device successfully you can program the device. To program go to Programmer and select **Program**.

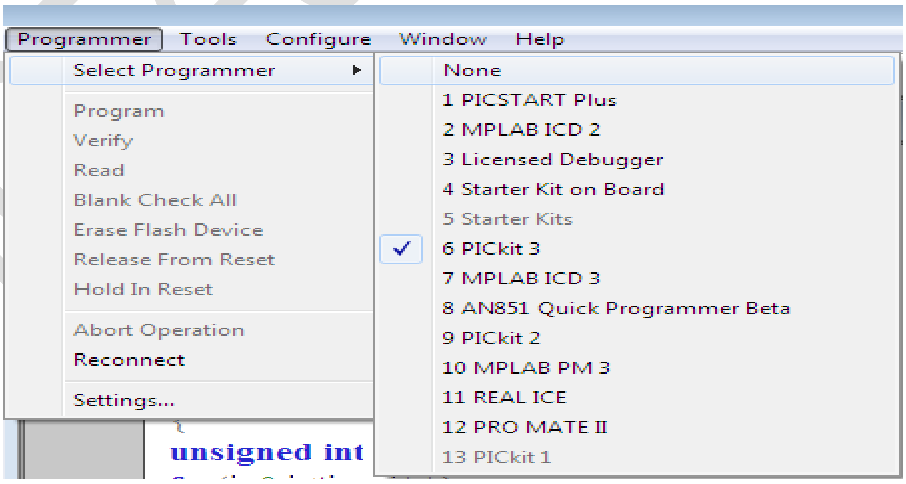


27. The project window shows following message.



```
PICkit 3 detected
Connecting to PICkit 3...
Running self test...
Self test completed
Firmware Suite Version...... 01.26.92
Firmware type......................PIC18F
PICkit 3 Connected.
Target Detected
Device ID Revision = 00000007

Programming...
Programming/Verify complete
```

28. After completing the programming successfully, go to Programmer -> Select Programmer -> None. In order to remove the programmer.



29. Now you are ready to run the program on target board.

--*************--

Experiment – 1

Title: Interfacing LED / Switch to PIC.

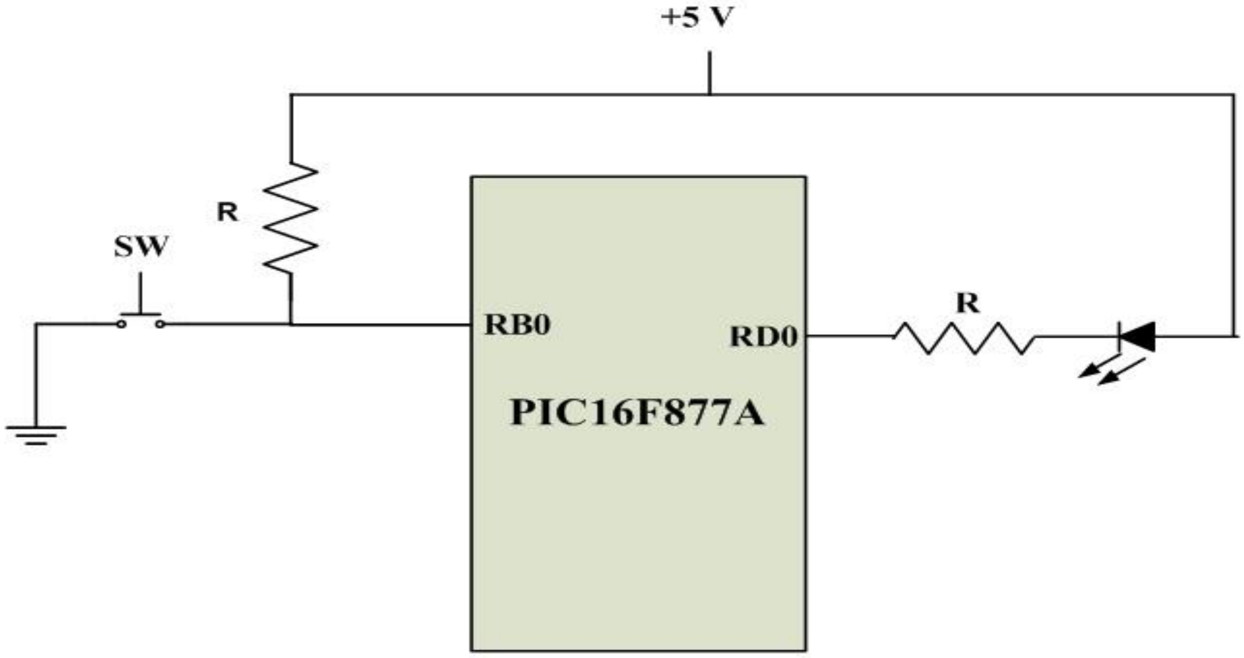
Aim: To interface LED / Switch to PIC and turn on the LED when switch is pressed.

Objectives:

- To study concept of LED / Switch interfacing.
- To study MPLAB IDE software.
- To study programming of LED / Switch interfacing.

Software Used: MPLAB IDE

Block Diagram:



Procedure:

- Make necessary connections to connect the LED / Switch to PIC target board.
- Switch on the power.
- Start MPLAB IDE software PC and write a program to read status of switch and send to the LED.
- Perform the configuration settings and build it.
- Connect the PICKit3 programmer to the Target board.
- Program the .hex file into the PIC.
- Reset the microcontroller and observe the output.

Program:

```
#define _XTAL_FREQ 16000000
#include <xc.h>
int main()
{
  TRISD0 = 0;         //RD0 as Output PIN
  TRISB0 = 1;        //RB0 as Input PIN
  while(1)
  {
   if(RB0 == 0)
   {
    RD0 = 0;          // LED ON
   }
   else
   {
    RD0 = 1;          // LED OFF
   }
  }
  return 0;
}
```

Applications: (Write applications of LED / Switch here)

Result: Interfacing of LED / Switch with PIC microcontroller is studied successfully and observed the output.

Teacher's Sign

--**********--

Experiment – 2

Title: LCD interfacing to PIC Microcontroller.

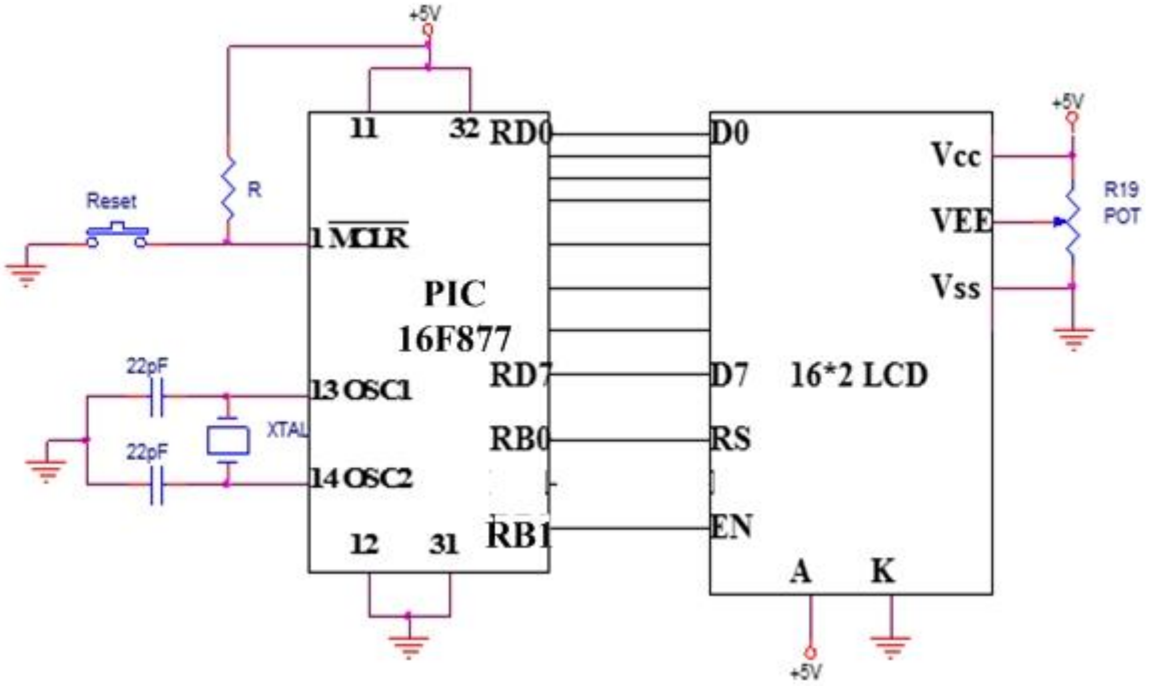
Aim: To interface 16*2 LCD display to PIC and display message on LCD.

Objectives:

- To study concept of LCD.
- To study MPLAB IDE software.
- To study LCD interfacing to PIC, flowcharts & programs.

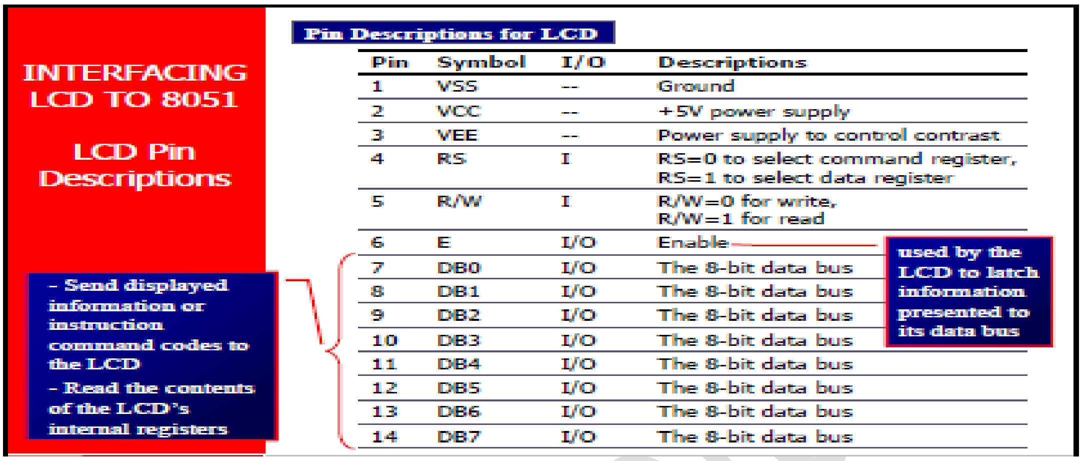
Software Used: MPLAB IDE

Block Diagram:



Theory:

LCD: A liquid crystal display (LCD) is a thin, flat electronic visual display that uses the light modulating properties of liquid crystals (LCs). They are used in a wide range of applications including: computer monitors, television, instrument panels, aircraft, cockpit display, signage, etc. It is an electronically modulated device made up of any number of pixels filled with liquid crystals. LCD has become very popular option for displaying in Embedded Applications. Since they are very cheap and easy to interface with microcontrollers, they are widely found in devices like telephones, vending machines, washing machines, toys etc. LCD comes in several varieties i.e. 16*2, 20*2, 20*4 etc. These different LCD varieties can display different number of characters i.e. 16*2 can display 32 characters at a time. The 16*2 model has 2 lines and 16 columns of display blocks. Each block can be used to display 1 character. So there are total 32 such blocks. One block has 8*5 pixels. Depending on which pixel is ON and which is OFF we can display several Alpha-Numeric characters. LCD also has a backlight, which helps us to see the display even in dark. In reality this module consists of a controller chip, a segment driver chip, LCD display and some passive components. There are total 16 pins in the LCD module. While using LCD, we can think a simple analogy for its operation. Each of the 32 blocks is a memory, as soon as we write an ASCII number into one of these 32 memory locations the corresponding character is displayed on that block. The function of displaying the character after decoding the data is done by an onboard controller chip. The following table shows the LCD pin diagram, LCD commands.



INTERFACING LCD TO 8051

LCD Pin Descriptions

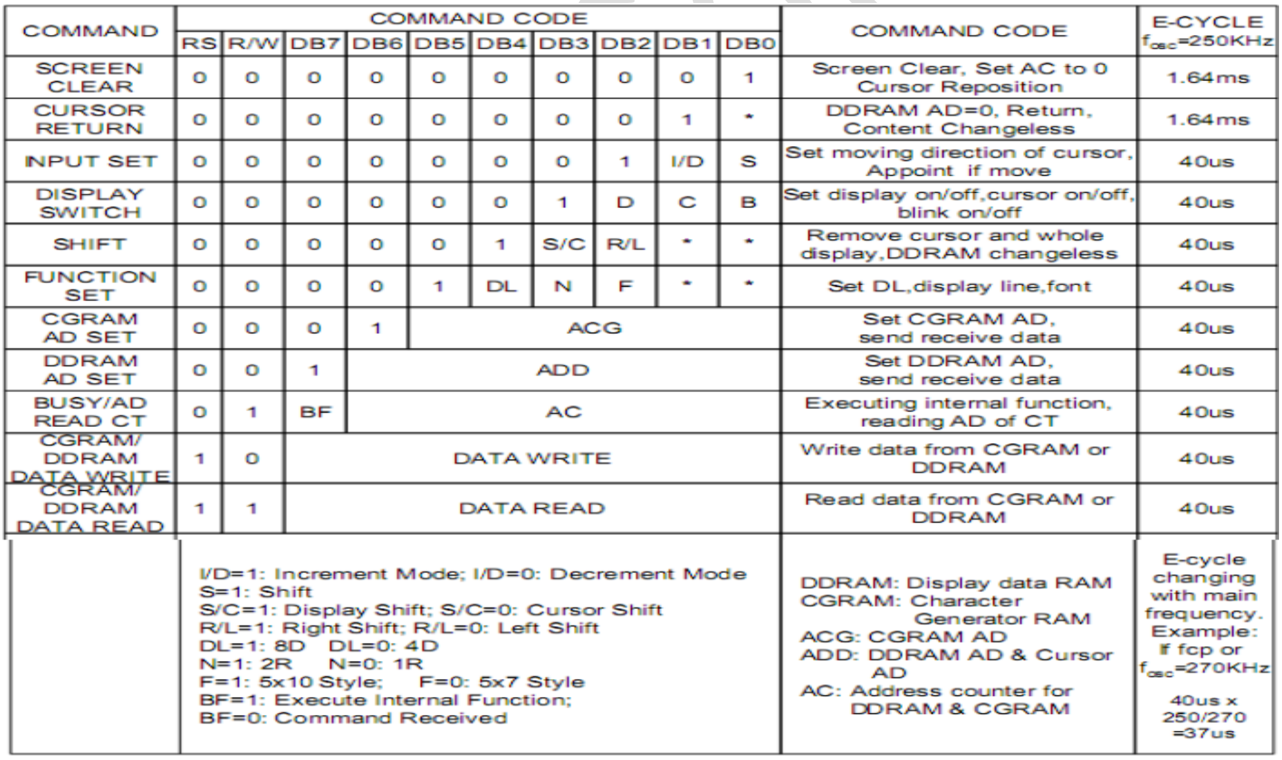
- Send displayed information or instruction command codes to the LCD
- Read the contents of the LCD's internal registers

Pin Descriptions for LCD

Pin	Symbol	I/O	Descriptions
1	VSS	--	Ground
2	VCC	--	+5V power supply
3	VEE	--	Power supply to control contrast
4	RS	I	RS=0 to select command register, RS=1 to select data register
5	R/W	I	R/W=0 for write, R/W=1 for read
6	E	I/O	Enable
7	DB0	I/O	The 8-bit data bus
8	DB1	I/O	The 8-bit data bus
9	DB2	I/O	The 8-bit data bus
10	DB3	I/O	The 8-bit data bus
11	DB4	I/O	The 8-bit data bus
12	DB5	I/O	The 8-bit data bus
13	DB6	I/O	The 8-bit data bus
14	DB7	I/O	The 8-bit data bus

used by the LCD to latch information presented to its data bus

Figure: LCD pin diagram



COMMAND	COMMAND CODE										COMMAND CODE	E-CYCLE f_{osc}=250KHz
	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0		
SCREEN CLEAR	0	0	0	0	0	0	0	0	0	1	Screen Clear, Set AC to 0 Cursor Reposition	1.64ms
CURSOR RETURN	0	0	0	0	0	0	0	0	1	*	DDRAM AD=0, Return, Content Changeless	1.64ms
INPUT SET	0	0	0	0	0	0	0	1	I/D	S	Set moving direction of cursor, Appoint if move	40us
DISPLAY SWITCH	0	0	0	0	0	0	1	D	C	B	Set display on/off,cursor on/off, blink on/off	40us
SHIFT	0	0	0	0	0	1	S/C	R/L	*	*	Remove cursor and whole display,DDRAM changeless	40us
FUNCTION SET	0	0	0	0	1	DL	N	F	*	*	Set DL,display line,font	40us
CGRAM AD SET	0	0	0	1	ACG						Set CGRAM AD, send receive data	40us
DDRAM AD SET	0	0	1	ADD							Set DDRAM AD, send receive data	40us
BUSY/AD READ CT	0	1	BF	AC							Executing internal function, reading AD of CT	40us
CGRAM/ DDRAM DATA WRITE	1	0	DATA WRITE								Write data from CGRAM or DDRAM	40us
CGRAM/ DDRAM DATA READ	1	1	DATA READ								Read data from CGRAM or DDRAM	40us
	I/D=1: Increment Mode; I/D=0: Decrement Mode S=1: Shift S/C=1: Display Shift; S/C=0: Cursor Shift R/L=1: Right Shift; R/L=0: Left Shift DL=1: 8D DL=0: 4D N=1: 2R N=0: 1R F=1: 5x10 Style; F=0: 5x7 Style BF=1: Execute Internal Function; BF=0: Command Received										DDRAM: Display data RAM CGRAM: Character Generator RAM ACG: CGRAM AD ADD: DDRAM AD & Cursor AD AC: Address counter for DDRAM & CGRAM	E-cycle changing with main frequency. Example: If fcp or f_{osc}=270KHz 40us x 250/270 =37us

Figure: LCD commands

Procedure:

- Make necessary connections to connect the LCD to PIC target board. Connect PORTD to the data pins of LCD and PORTB to the control pins.
- Switch on the power.
- Start MPLAB IDE software PC and write a program to display message on LCD.
- Perform the configuration settings and build it.

- Connect the PICKit3 programmer to the Target board.
- Program the .hex file into the PIC.
- Reset the microcontroller and observe the output on LCD.

Program:

```
#define _XTAL_FREQ 16000000
#include <xc.h>
#define RS PORTBbits.RB0
#define EN PORTBbits.RB1
void LCD_init(void);
void lcd_cmd(unsigned char value);
void lcd_data(unsigned char value1);

void main(void)
{
LCD_init();
lcd_data('H');
lcd_data('e');
lcd_data('l');
lcd_data('l');
lcd_data('o');
while(1);
}
void LCD_init(void)
{
TRISD=0X00;                                              //make PORTD o/p
TRISB=0X00;                                              //make PORTD o/p
__delay_ms(100);                                      //delay
EN=1;
__delay_us(2);
EN=0;
__delay_ms(3);                                           //2ms delay
lcd_cmd(0x38);
lcd_cmd(0x0E);
lcd_cmd(0x01);
__delay_ms(2);                                                //2ms delay
lcd_cmd(0x06);
lcd_cmd(0x80);
}

void lcd_cmd(unsigned char value)
{
PORTD=value;
RS=0;
EN=1;
__delay_us(2);
EN=0;
__delay_ms(3);                                          //2ms delay
}

void lcd_data(unsigned char value1)
{
PORTD=value1;
RS=1;                                                   //Select data reg
EN=1;
__delay_us(2);
EN=0;
__delay_ms(3);                                          //2ms delay
}
```

Applications: (Write applications of LCD here)

Result: Interfacing of 16*2 LCD with PIC microcontroller is studied successfully and observed the message displayed on LCD.

Teacher's Sign

---**********---

Experiment – 3

Title: Interfacing sensors to PIC.

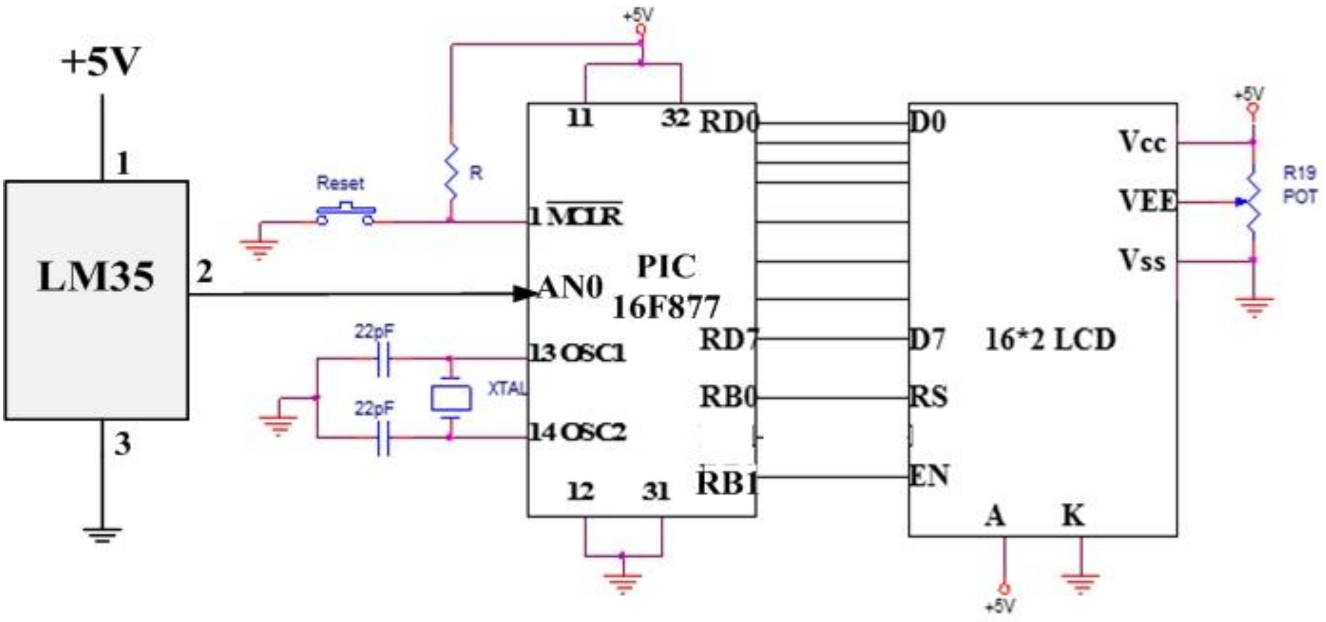
Aim: To interface sensor (LM35) to PIC and display result on LCD.

Objectives:

- To study concept of interfacing sensor (LM35).
- To study MPLAB IDE software.
- To study use of ADC to interface sensors.

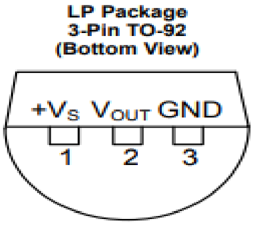
Software Used: MPLAB IDE

Block Diagram:



Theory:

The LM35 series are precision integrated-circuit temperature devices with an output voltage linearly proportional to the Centigrade temperature. The LM35 device does not require any external calibration or trimming to provide typical accuracies of ±¼°C at room temperature and ±¾°C over a full −55°C to 150°C temperature range. Lower cost is assured by trimming and calibration at the wafer level. The low-output impedance, linear output, and precise inherent calibration of the LM35 device makes interfacing to readout or control circuitry especially easy. This sensor has sensitivity of 10mv/^{o}C, operates from 4 V to 30 V.



Pin Diagram of LM 35 Sensor

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0
ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/$\overline{\text{DONE}}$	—	ADON
bit 7							bit 0

bit 7-6 **ADCS1:ADCS0:** A/D Conversion Clock Select bits
`00` = FOSC/2
`01` = FOSC/8
`10` = FOSC/32
`11` = FRC (clock derived from the internal A/D module RC oscillator)

bit 5-3 **CHS2:CHS0**: Analog Channel Select bits
`000` = channel 0, (RA0/AN0)
`001` = channel 1, (RA1/AN1)
`010` = channel 2, (RA2/AN2)
`011` = channel 3, (RA3/AN3)
`100` = channel 4, (RA5/AN4)
`101` = channel 5, (RE0/AN5)(1)
`110` = channel 6, (RE1/AN6)(1)
`111` = channel 7, (RE2/AN7)(1)

bit 2 **GO/$\overline{\text{DONE}}$:** A/D Conversion Status bit
<u>If ADON = 1:</u>
`1` = A/D conversion in progress (setting this bit starts the A/D conversion)
`0` = A/D conversion not in progress (this bit is automatically cleared by hardware when the A/D conversion is complete)

bit 1 **Unimplemented**: Read as '0'

bit 0 **ADON**: A/D On bit
`1` = A/D converter module is operating
`0` = A/D converter module is shut-off and consumes no operating current

Figure: ADCON0 REGISTER

U-0	U-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
ADFM	—	—	—	PCFG3	PCFG2	PCFG1	PCFG0
bit 7							bit 0

bit 7 **ADFM:** A/D Result Format Select bit
`1` = Right justified. 6 Most Significant bits of ADRESH are read as ‘0’.
`0` = Left justified. 6 Least Significant bits of ADRESL are read as ‘0’.

bit 6-4 **Unimplemented:** Read as '0'

bit 3-0 **PCFG3:PCFG0**: A/D Port Configuration Control bits:

PCFG3: PCFG0	AN7(1) RE2	AN6(1) RE1	AN5(1) RE0	AN4 RA5	AN3 RA3	AN2 RA2	AN1 RA1	AN0 RA0	VREF+	VREF-	CHAN/ Refs(2)
0000	A	A	A	A	A	A	A	A	VDD	VSS	8/0
0001	A	A	A	A	VREF+	A	A	A	RA3	VSS	7/1
0010	D	D	D	A	A	A	A	A	VDD	VSS	5/0
0011	D	D	D	A	VREF+	A	A	A	RA3	VSS	4/1
0100	D	D	D	D	A	D	A	A	VDD	VSS	3/0
0101	D	D	D	D	VREF+	D	A	A	RA3	VSS	2/1
011x	D	D	D	D	D	D	D	D	VDD	VSS	0/0
1000	A	A	A	A	VREF+	VREF-	A	A	RA3	RA2	6/2
1001	D	D	A	A	A	A	A	A	VDD	VSS	6/0
1010	D	D	A	A	VREF+	A	A	A	RA3	VSS	5/1
1011	D	D	A	A	VREF+	VREF-	A	A	RA3	RA2	4/2
1100	D	D	D	A	VREF+	VREF-	A	A	RA3	RA2	3/2
1101	D	D	D	D	VREF+	VREF-	A	A	RA3	RA2	2/2
1110	D	D	D	D	D	D	D	A	VDD	VSS	1/0
1111	D	D	D	D	VREF+	VREF-	D	A	RA3	RA2	1/2

A = Analog input D = Digital I/O

Figure: ADCON1 REGISTER

Procedure:

- Make necessary connections to connect the LCD to PIC target board. Connect PORTD to the data pins of LCD and PORTB to the control pins. Also connect o/p of LM35 sensor to AN0 / RA0 pin.
- Switch on the power.
- Start MPLAB IDE software PC and write a program for sensor interfacing.
- Perform the configuration settings and build it.
- Connect the PICKit3 programmer to the Target board.
- Program the .hex file into the PIC.
- Reset the microcontroller and observe the output on LCD.

Program:

```
#define _XTAL_FREQ 16000000
#include <xc.h>
#define RS RB0
#define EN RB1
void ADC_init(void);
unsigned int ADC_READ(void);
void LCD_init(void);
void lcd_cmd(unsigned char value);
void lcd_data(unsigned char value1);
void H2D(unsigned char hexdata);
unsigned char L,M,H;
void main(void)
{
unsigned int ADC_result;
ADC_init();
LCD_init();
while(1)
{
ADC_result=ADC_READ();
ADC_result=ADC_result*5;            //resolution of ADC is 4.88mv
ADC_result=ADC_result/10;             //LM35 sensitivity 10mV/C
H2D(ADC_result);
lcd_cmd(0x01);
lcd_data('T');
lcd_data('=');
lcd_data(H | 0x30);
lcd_data(M | 0x30);
lcd_data(L | 0x30);
__delay_ms(500);                //500ms delay
}
}

void ADC_init(void)
{
TRISAbits.TRISA0=1;     // AN0 set as analog i/p
TRISAbits.TRISA1=1;     // AN1 set as analog i/p
ADRESH=0;
ADRESL=0;
PIR1bits.ADIF=0;        //ADC flag clear
ADCON1=0X8E;            //MAKE AN0 as analog port
ADCON0=0x40;            //Select Chanel 0, conversion clock = FOSC / 8
}

unsigned int ADC_READ()
{
ADCON0bits.ADON=1;          //ADC ON
__delay_ms(2);                  //2ms delay
ADCON0bits.GO=1;            //start a/d conversion
__delay_us(1);
while(PIR1bits.ADIF==0);
__delay_us(1);
ADCON0bits.ADON=0;          //ADC Off
PIR1bits.ADIF=0;
return (((unsigned int)ADRESH)<<8)|(ADRESL);
}

void LCD_init(void)
{
TRISD=0X00;                                                     //make PORTD o/p
```

```
TRISB=0X00;                                    //make PORTD o/p
__delay_ms(100);                               //delay
EN=1;
__delay_us(2);
EN=0;
__delay_ms(3);                                 //2ms delay
lcd_cmd(0x38);
lcd_cmd(0x0E);
lcd_cmd(0x01);
__delay_ms(2);                                 //2ms delay
lcd_cmd(0x06);
lcd_cmd(0x80);
}

void lcd_cmd(unsigned char value)
{
PORTD=value;
RS=0;
EN=1;
__delay_us(2);
EN=0;
__delay_ms(3);                                 //2ms delay
}

void lcd_data(unsigned char value1)
{
PORTD=value1;
RS=1;                                          //Select data reg
EN=1;
__delay_us(2);
EN=0;
__delay_ms(3);                                 //2ms delay
}

void H2D(unsigned char hexdata)
```

```
{
unsigned char x;
x=hexdata/10;
L=hexdata%10;
M=x%10;
H=x/10;
}
```

Applications: (Write applications of Sensor interfacing here)

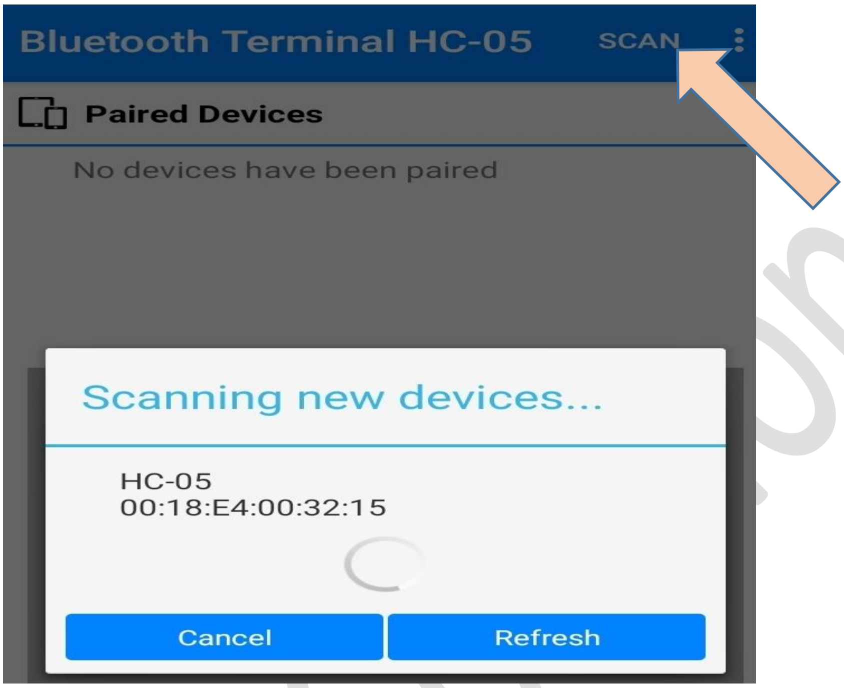
Result: Interfacing of sensor with PIC microcontroller is studied successfully and observed the result on LCD.

Teacher's Sign

---*********---

Steps to use Bluetooth Terminal HC05 Mobile app.

1. Go to Play Store of your smart phone and install the “Bluetooth Terminal HC05” app.
2. Open “Bluetooth Terminal HC05” app. in your smart phone. Click on SCAN.



Bluetooth Terminal HC-05 SCAN

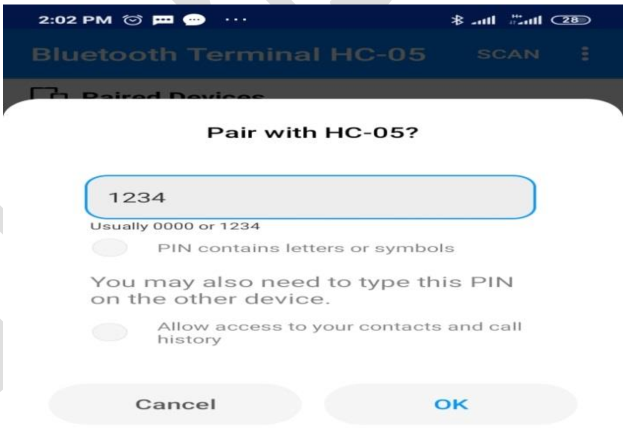
Paired Devices

No devices have been paired

Scanning new devices...

HC-05
00:18:E4:00:32:15

3. App scan new devices and when found, ask for password in order to pair with that new device. Password for all HC05 devices is same i.e. “1234”. Enter the password and press “OK”.



2:02 PM

Bluetooth Terminal HC-05 SCAN

Paired Devices

Pair with HC-05?

1234

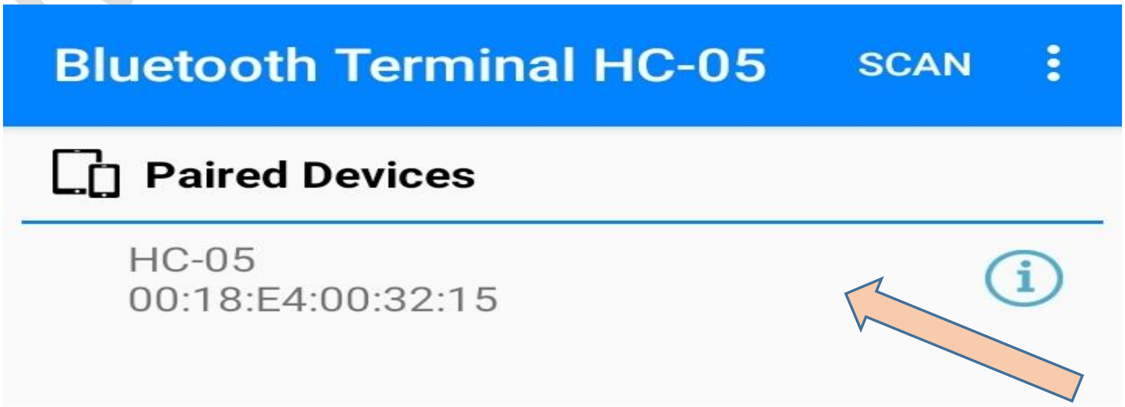
Usually 0000 or 1234

PIN contains letters or symbols

You may also need to type this PIN on the other device.

Allow access to your contacts and call history

4. Then the HC05 device showed under the heading of Paired Devices.

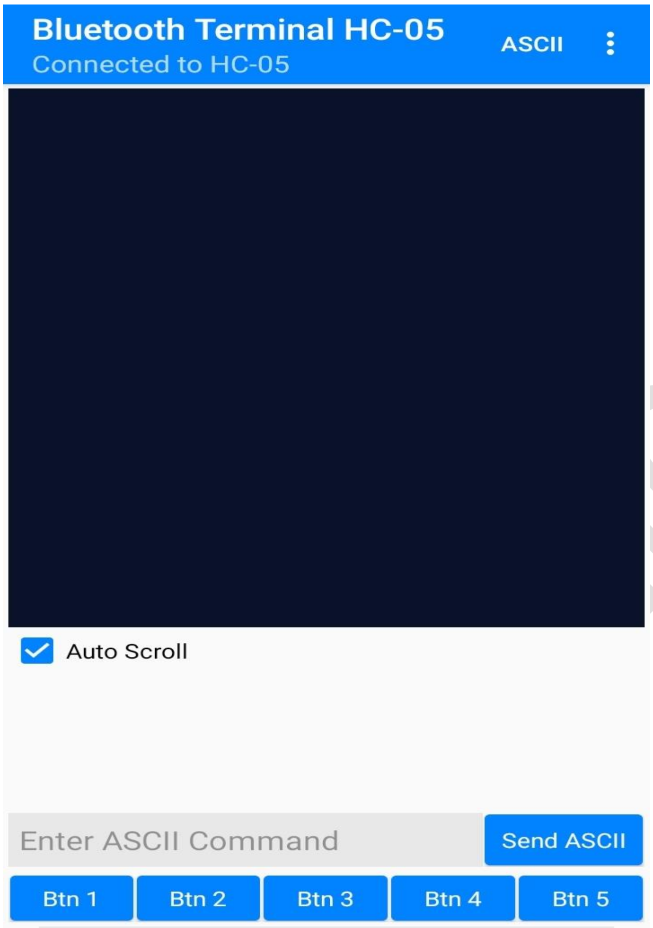


Bluetooth Terminal HC-05 SCAN

Paired Devices

HC-05
00:18:E4:00:32:15

5. Click on the device to open this device for further communication. In the black window you will observe the data send by the microcontroller.


Bluetooth Terminal HC-05

Connected to HC-05

---*************--

Experiment – 4

Title: Interfacing Bluetooth to PIC.

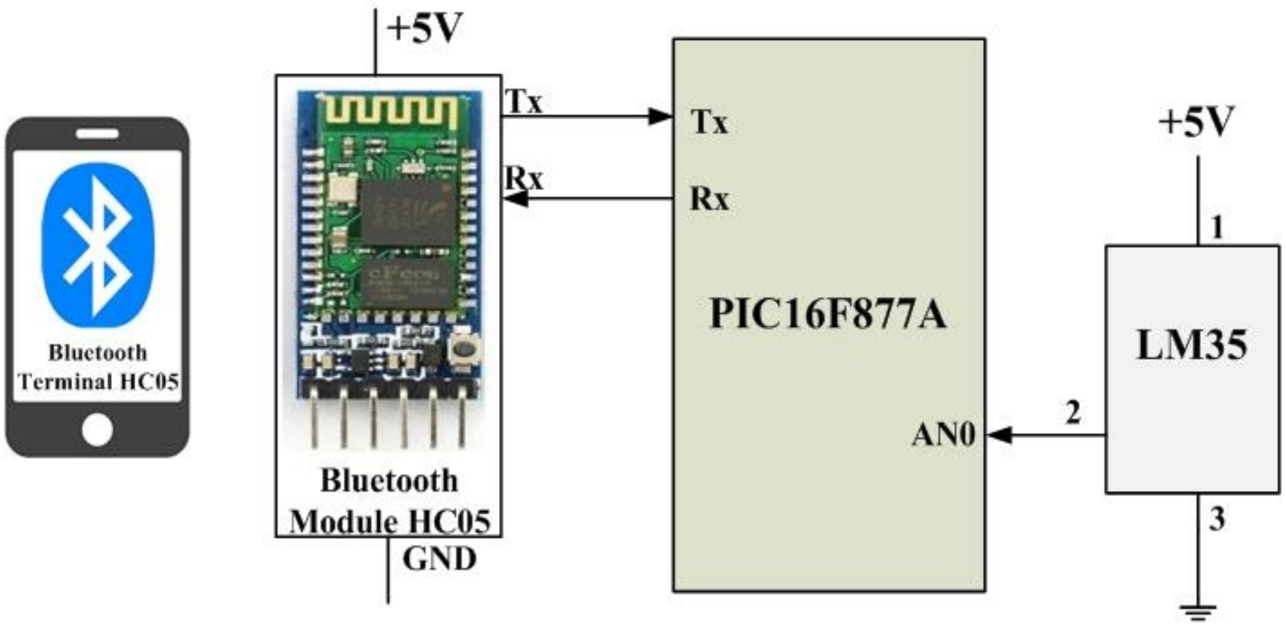
Aim: To interface Bluetooth module (HC05).

Objectives:

- To study concept of interfacing Bluetooth module (HC05).
- To study MPLAB IDE software.
- To study programming serial port for interfacing Bluetooth.

Software Used: MPLAB IDE

Block Diagram:

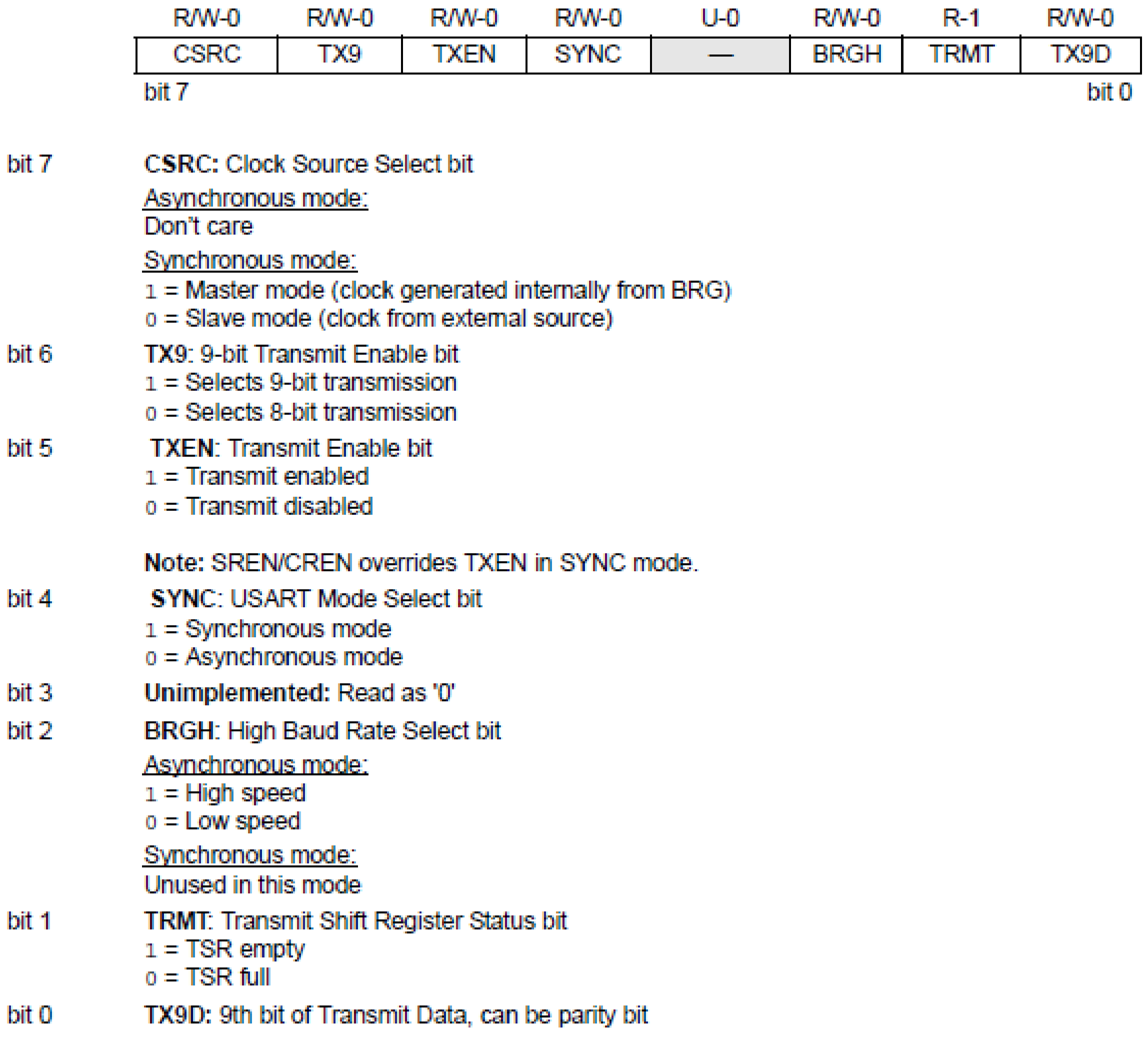


Theory:

Bluetooth is a wireless technology standard used for exchanging data between fixed and mobile devices over short distances using short-wavelength UHF radio waves in the industrial, scientific and medical radio bands, from 2.402 GHz to 2.480 GHz, and building personal area networks (PANs). It was originally conceived as a wireless alternative to RS-232 data cables. The IEEE standardized Bluetooth as IEEE 802.15.1. Bluetooth is a one of the great example for wireless connectivity. It is used in many fields. Bluetooth consumes very small amount of energy. Here we are going to interface a Bluetooth Module (HC-05) with Arduino. HC-05 is a Bluetooth module which can communicate in two ways. Which means, It is full-duplex. We can use it with most micro controllers. Because it operates on Serial Port Protocol (SSP). The module communicates with the help of USART (Universal Synchronous/Asynchronous Receiver/Transmitter) at the baud rate of 9600, and it also supports other baud rate. So we can interface this module with any microcontroller which supports USART. The HC-05 can operate in two modes. One is Data mode and other is AT command mode. When the enable pin is "LOW" the HC-05 is in Data Mode. If that pin set as "HIGH" the module is in AT command mode. Here we operate this module in Data Mode.

Technical Specifications -

- Operating Voltage: 4V to 6V (Typically +5V)
- Operating Current: 30mA
- Range: <100m
- Works with Serial communication (USART) and TTL compatible.
- Can be easily interfaced with Laptop or Mobile phones with Bluetooth.



R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R-1	R/W-0
CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D
bit 7							bit 0

bit 7 **CSRC:** Clock Source Select bit
Asynchronous mode:
Don't care
Synchronous mode:
1 = Master mode (clock generated internally from BRG)
0 = Slave mode (clock from external source)

bit 6 **TX9**: 9-bit Transmit Enable bit
1 = Selects 9-bit transmission
0 = Selects 8-bit transmission

bit 5 **TXEN**: Transmit Enable bit
1 = Transmit enabled
0 = Transmit disabled

Note: SREN/CREN overrides TXEN in SYNC mode.

bit 4 **SYNC**: USART Mode Select bit
1 = Synchronous mode
0 = Asynchronous mode

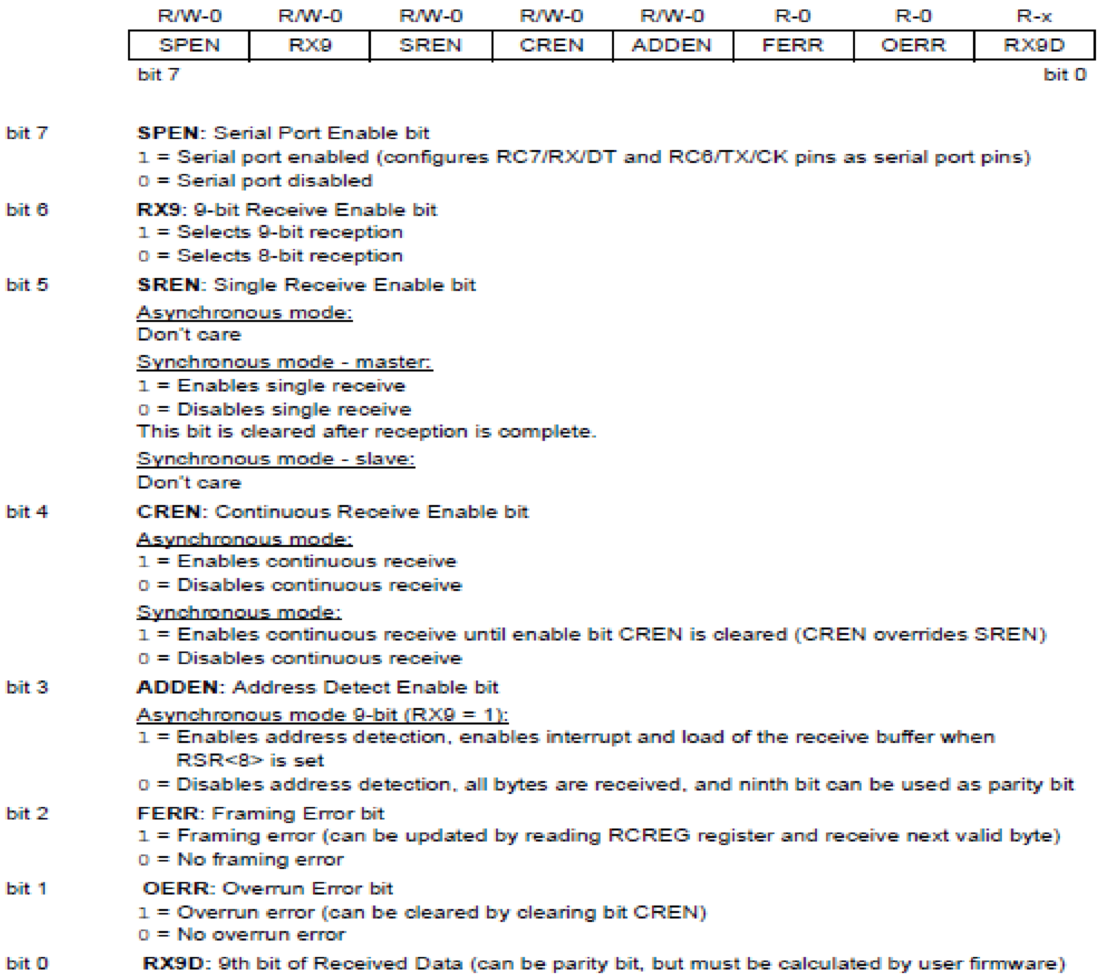
bit 3 **Unimplemented:** Read as '0'

bit 2 **BRGH**: High Baud Rate Select bit
Asynchronous mode:
1 = High speed
0 = Low speed
Synchronous mode:
Unused in this mode

bit 1 **TRMT**: Transmit Shift Register Status bit
1 = TSR empty
0 = TSR full

bit 0 **TX9D:** 9th bit of Transmit Data, can be parity bit

Figure: TXSTA REGISTER



R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	R-0	R-x
SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
bit 7							bit 0

bit 7 **SPEN**: Serial Port Enable bit
1 = Serial port enabled (configures RC7/RX/DT and RC6/TX/CK pins as serial port pins)
0 = Serial port disabled

bit 6 **RX9**: 9-bit Receive Enable bit
1 = Selects 9-bit reception
0 = Selects 8-bit reception

bit 5 **SREN**: Single Receive Enable bit
Asynchronous mode:
Don't care
Synchronous mode - master:
1 = Enables single receive
0 = Disables single receive
This bit is cleared after reception is complete.
Synchronous mode - slave:
Don't care

bit 4 **CREN**: Continuous Receive Enable bit
Asynchronous mode:
1 = Enables continuous receive
0 = Disables continuous receive
Synchronous mode:
1 = Enables continuous receive until enable bit CREN is cleared (CREN overrides SREN)
0 = Disables continuous receive

bit 3 **ADDEN**: Address Detect Enable bit
Asynchronous mode 9-bit (RX9 = 1):
1 = Enables address detection, enables interrupt and load of the receive buffer when RSR<8> is set
0 = Disables address detection, all bytes are received, and ninth bit can be used as parity bit

bit 2 **FERR**: Framing Error bit
1 = Framing error (can be updated by reading RCREG register and receive next valid byte)
0 = No framing error

bit 1 **OERR**: Overrun Error bit
1 = Overrun error (can be cleared by clearing bit CREN)
0 = No overrun error

bit 0 **RX9D**: 9th bit of Received Data (can be parity bit, but must be calculated by user firmware)

Figure: RCSTA REGISTER

Procedure:

- Make necessary connections to connect the Bluetooth module to PIC target board.
- Switch on the power.
- Start MPLAB IDE software PC and write a program for sensor communication with Bluetooth module.
- Perform the configuration settings and build it.
- Connect the PICKit3 programmer to the Target board.
- Program the .hex file into the PIC.
- Install the Bluetooth Terminal HC05 app. in the smartphone.
- Reset the microcontroller and open the Bluetooth Terminal HC05 Mobile app. and observe the output.

Program:

```
#define _XTAL_FREQ 16000000
#include <xc.h>
void ADC_init(void);
unsigned int ADC_READ(void);
```

```
void serial_init(void);
void serial_Tx(unsigned char letter);
void H2D(unsigned char hexdata);
unsigned char L,M,H;
void main(void)
{
unsigned int ADC_result;
ADC_init();
serial_init();
while(1)
{
ADC_result=ADC_READ();
ADC_result=ADC_result*5;      //resolution of ADC is 4.88mv
ADC_result=ADC_result/10;    //LM35 sensitivity 10mV/C
H2D(ADC_result);
serial_Tx('T');
serial_Tx('=');
serial_Tx(H | 0x30);
serial_Tx(M | 0x30);
serial_Tx(L | 0x30);
serial_Tx(0x0A);
__delay_ms(2000);              //500ms delay
}
}

void serial_init(void)
{
SPBRG=103; //set baudrate equal to 9600 formula: Baud Rate = FOSC/(16(X+1)) [X is value of SPBRG]
TXSTA=0X06;
RCSTA=0X80;                             //serial comm enable
TRISCbits.TRISC6=1;               //set transmit pin as i/p
TRISCbits.TRISC7=1;            //set recieve pin as i/p
RCSTAbits.CREN=1;          //receive enable
}
```

```
void serial_Tx(unsigned char letter)
{
TXSTAbits.TXEN=1;    //transmit enable
TXREG=letter;                    //transmit letter
__delay_us(2);
while(PIR1bits.TXIF==0);   //check for transmit interrupt flag
while(TXSTAbits.TRMT==0);
TXSTAbits.TXEN=0;    //transmit desable
__delay_ms(20);
}

void ADC_init(void)
{
TRISAbits.TRISA0=1;  // AN0 set as analog i/p
TRISAbits.TRISA1=1;  // AN1 set as analog i/p
ADRESH=0;
ADRESL=0;
PIR1bits.ADIF=0;             //ADC flag clear
ADCON1=0X8E;              //MAKE AN0 as analog port
ADCON0=0x40;                 //Select Chanel 0, conversion clock = FOSC / 8
}

unsigned int ADC_READ()
{
ADCON0bits.ADON=1;                 //ADC ON
__delay_ms(2);                           //2ms delay
ADCON0bits.GO=1;                     //start a/d conversion
__delay_us(1);
while(PIR1bits.ADIF==0);
__delay_us(1);
ADCON0bits.ADON=0;                //ADC Off
PIR1bits.ADIF=0;
return (((unsigned int)ADRESH)<<8)|(ADRESL);
}
```

```
void H2D(unsigned char hexdata)
{
unsigned char x;
x=hexdata/10;
L=hexdata%10;
M=x%10;
H=x/10;
}
```

Applications: (Write applications of Bluetooth interfacing here)

Result: Interfacing of Bluetooth module with PIC microcontroller is studied successfully and observed the o/p.

Teacher's Sign

---**********---

Experiment – 5

Title: Design and develop On/Off controller using microcontroller.

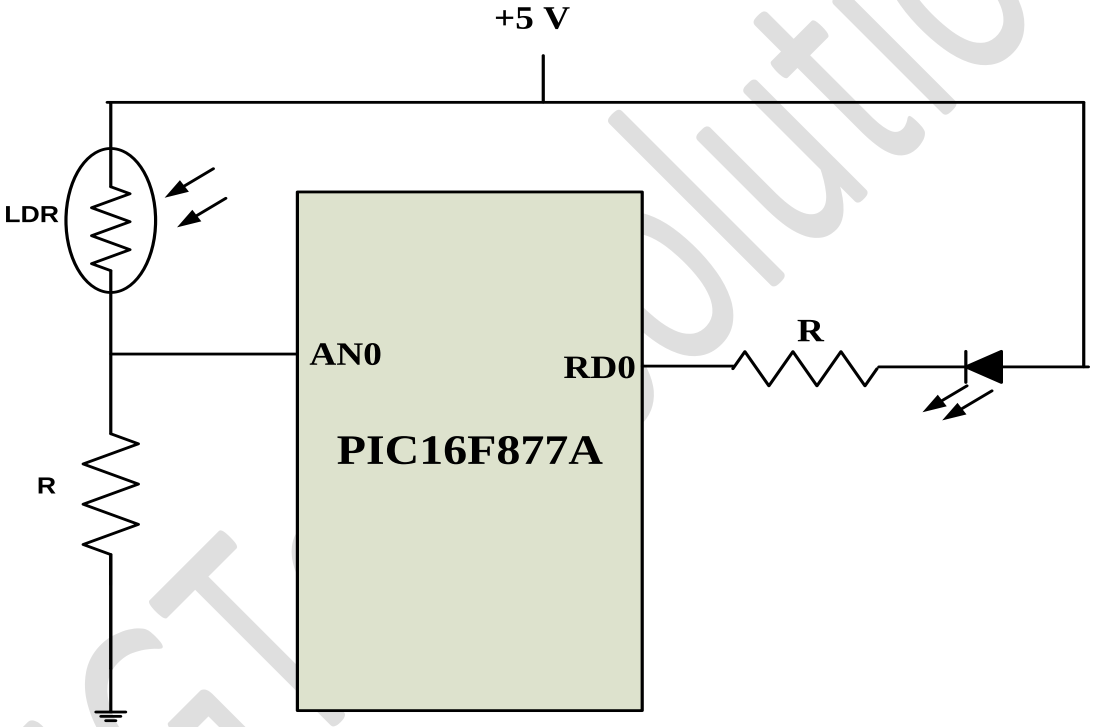
Aim: To Design and develop On/Off controller using PIC.

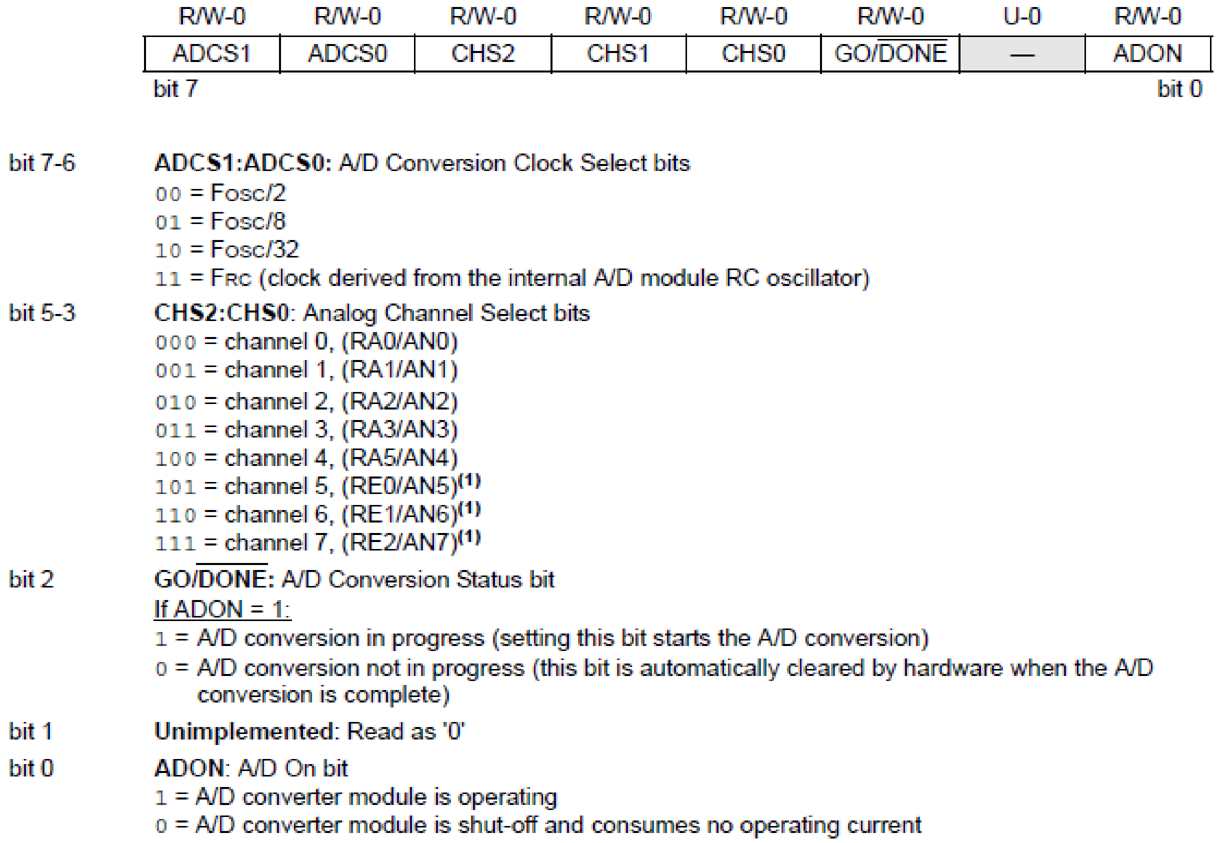
Objectives:

- To study concept of On/Off controller.
- To study MPLAB IDE software.
- To study use of ADC to interface sensors.

Software Used: MPLAB IDE

Block Diagram:





R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0
ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/$\overline{\text{DONE}}$	—	ADON
bit 7							bit 0

bit 7-6 **ADCS1:ADCS0:** A/D Conversion Clock Select bits
00 = Fosc/2
01 = Fosc/8
10 = Fosc/32
11 = FRC (clock derived from the internal A/D module RC oscillator)

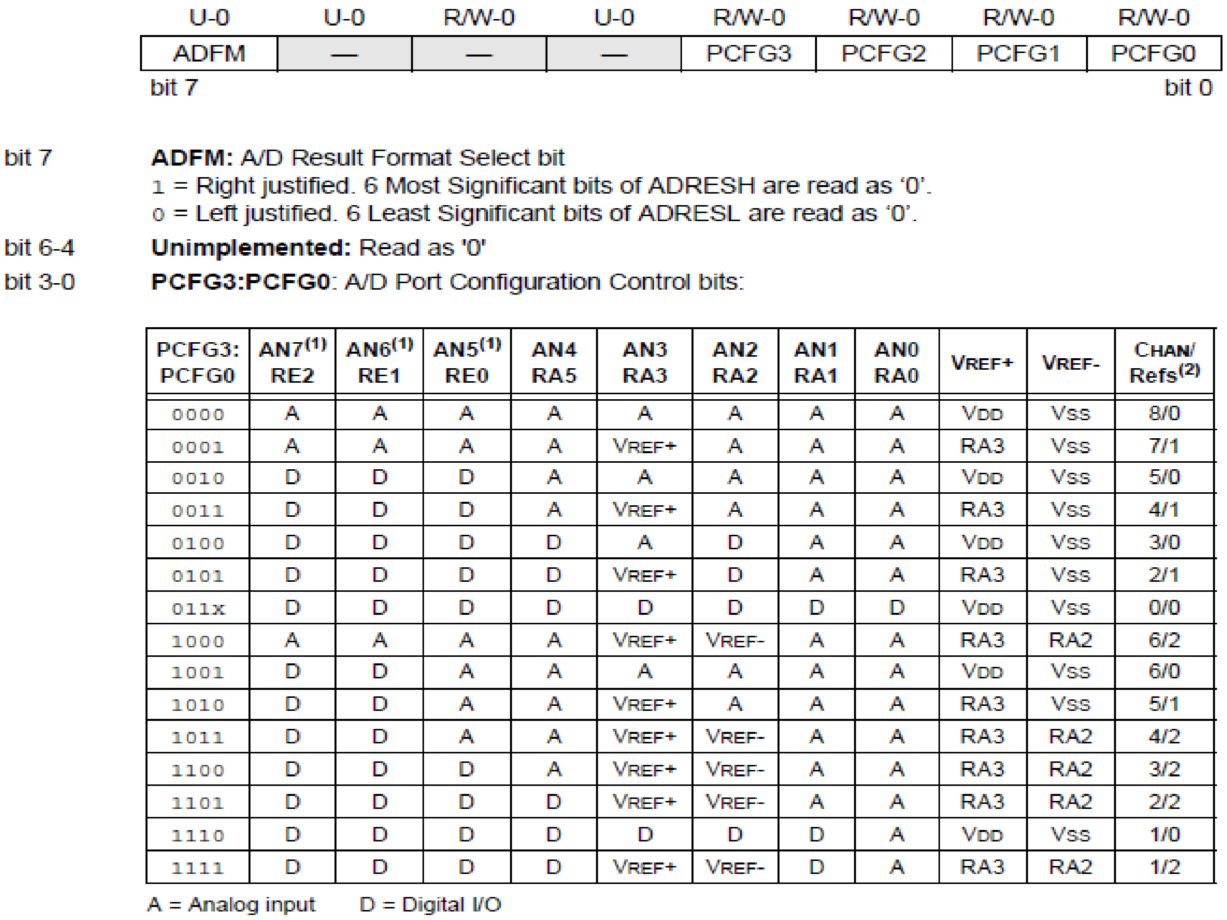
bit 5-3 **CHS2:CHS0**: Analog Channel Select bits
000 = channel 0, (RA0/AN0)
001 = channel 1, (RA1/AN1)
010 = channel 2, (RA2/AN2)
011 = channel 3, (RA3/AN3)
100 = channel 4, (RA5/AN4)
101 = channel 5, (RE0/AN5)[1]
110 = channel 6, (RE1/AN6)[1]
111 = channel 7, (RE2/AN7)[1]

bit 2 **GO/$\overline{\text{DONE}}$:** A/D Conversion Status bit
If ADON = 1:
1 = A/D conversion in progress (setting this bit starts the A/D conversion)
0 = A/D conversion not in progress (this bit is automatically cleared by hardware when the A/D conversion is complete)

bit 1 **Unimplemented**: Read as '0'

bit 0 **ADON**: A/D On bit
1 = A/D converter module is operating
0 = A/D converter module is shut-off and consumes no operating current

Figure: ADCON0 REGISTER



U-0	U-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
ADFM	—	—	—	PCFG3	PCFG2	PCFG1	PCFG0
bit 7							bit 0

bit 7 **ADFM:** A/D Result Format Select bit
1 = Right justified. 6 Most Significant bits of ADRESH are read as '0'.
0 = Left justified. 6 Least Significant bits of ADRESL are read as '0'.

bit 6-4 **Unimplemented:** Read as '0'

bit 3-0 **PCFG3:PCFG0**: A/D Port Configuration Control bits:

PCFG3: PCFG0	AN7[1] RE2	AN6[1] RE1	AN5[1] RE0	AN4 RA5	AN3 RA3	AN2 RA2	AN1 RA1	AN0 RA0	VREF+	VREF-	CHAN/ Refs[2]
0000	A	A	A	A	A	A	A	A	VDD	VSS	8/0
0001	A	A	A	A	VREF+	A	A	A	RA3	VSS	7/1
0010	D	D	D	A	A	A	A	A	VDD	VSS	5/0
0011	D	D	D	A	VREF+	A	A	A	RA3	VSS	4/1
0100	D	D	D	D	A	D	A	A	VDD	VSS	3/0
0101	D	D	D	D	VREF+	D	A	A	RA3	VSS	2/1
011x	D	D	D	D	D	D	D	D	VDD	VSS	0/0
1000	A	A	A	A	VREF+	VREF-	A	A	RA3	RA2	6/2
1001	D	D	A	A	A	A	A	A	VDD	VSS	6/0
1010	D	D	A	A	VREF+	A	A	A	RA3	VSS	5/1
1011	D	D	A	A	VREF+	VREF-	A	A	RA3	RA2	4/2
1100	D	D	D	A	VREF+	VREF-	A	A	RA3	RA2	3/2
1101	D	D	D	D	VREF+	VREF-	A	A	RA3	RA2	2/2
1110	D	D	D	D	D	D	D	A	VDD	VSS	1/0
1111	D	D	D	D	VREF+	VREF-	D	A	RA3	RA2	1/2

A = Analog input D = Digital I/O

Figure: ADCON1 REGISTER

Procedure:

- Make necessary connections to connect LDR Sensor and LED to PIC target board.
- Switch on the power.
- Start MPLAB IDE software PC and write a program for on / off controller.
- Perform the configuration settings and build it.
- Connect the PICKit3 programmer to the Target board.
- Program the .hex file into the PIC.
- Reset the microcontroller and observe the output.

Program:

```
#define _XTAL_FREQ 16000000
#include <xc.h>
void ADC_init(void);
unsigned int ADC_READ(void);
void main(void)
{
unsigned int ADC_result;
ADC_init();
TRISD0 = 0;                //RD0 as Output PIN
while(1)
{
ADC_result=ADC_READ();
if(ADC_result > 500)
{
RD0 = 0;                   // LED ON
}
else
{
RD0 = 1;                   // LED OFF
}
}
}

void ADC_init(void)
{
TRISAbits.TRISA0=1;              // AN0 set as analog i/p
TRISAbits.TRISA1=1;             // AN1 set as analog i/p
ADRESH=0;
ADRESL=0;
PIR1bits.ADIF=0;                //ADC flag clear
ADCON1=0X8E;                    //MAKE AN0 as analog port
ADCON0=0x40;                    //Select Chanel 0, conversion clock = FOSC / 8
}
unsigned int ADC_READ()
{
ADCON0bits.ADON=1;                                              //ADC ON
__delay_ms(2);                                                  //2ms delay
ADCON0bits.GO=1;                                                //start a/d conversion
__delay_us(1);
while(PIR1bits.ADIF==0);
__delay_us(1);
ADCON0bits.ADON=0;                                              //ADC Off
PIR1bits.ADIF=0;
return (((unsigned int)ADRESH)<<8)|(ADRESL);
}
```

Applications: (Write applications of On/Off controller here)

Result: Design and development of On/Off controller using PIC is studied and tested successfully.

Teacher's Sign

---**********---