Session: FC2-2

THE LAB WORKSHOP MODELS ON MICROCHIP's PIC MICROCONTROLLERS IN EET PROGRAM

Muhammad M. Baig, Rafiqul Islam
Dept. of Engineering Technology
Northwestern State University
Natchitoches, LA 71497
email: baigm@nsula.edu
email: islamr@nsula.edu

Abstract

The microcontroller is a versatile, simple and low-cost solution for many electronic-based interface and control situations and problem solutions. It is usable in place of a combination of many active and passive electronic components. Lesser components and lesser connections enhance reliability and life of the designed systems besides many other obvious advantages. A microcontroller is intelligent, so programmable, that adds to its awesomeness. We will call an electronic design based on (/ using) a microcontroller as a microcontroller-based project / application. Designing of a microcontroller-based application is a challenge, since it involves simultaneous completion of both its hardware and software designs. Besides, it requires to use and know how to use each of the hardware and software development tools.

This lab workshop models on Microchip's PIC microcontrollers is intended for students undergoing an undergraduate-level of studies in Electronic Engineering Technology (EET) at Northwestern State University of Louisiana, Natchitoches and those who have a credit in a two-semester Digital Electronics course and a credit in a single-semester high-level programming language course. This lab workshop is for the purpose of exploring / explaining various aspects of building the PIC microcontroller-based applications

This paper will be a complete description of hardware & software requirements and procedures on how to experimentally develop, build, test and troubleshoot a PIC microcontroller-based application. This includes writing a program code, translate it to an object code loadable onto a microcontroller. The authors will demonstrate how to use all the required hardware as well as software development tools in three laboratory projects. This hands-on experience will strengthen students' knowledge base and make them more marketable in this field. It will also be valuable to the students of other colleges and universities with similar concentration.

1. Introduction

A microcontroller is a computer implemented on a single very large scale integrated (VLSI) circuit, containing some of the peripheral components, namely, memory, timers (including event counting, input capture, output compare, real-time interrupt and watchdog timer), pulse-width modulation, analog-to-digital converter, digital-to-analog converter, parallel input / output interface, asynchronous serial communication interface, synchronous serial communication interfaces, direct memory access controller, memory component interface circuitry and software debug support hardware.

In the Electronic Engineering Technology (EET) program at Northwestern State University of Louisiana, Natchitoches, we teach the Microchip's 8-bit microcontrollers. Our choice of these microcontrollers is based on the fact that they are one of the new market leaders in the today's competitive 8-bit microcontroller market [1, 2 & 3]. Our program consists of a single semester three credit hours class course associated with a single semester one credit hour laboratory course.

The laboratory course includes labs on the Microchip's PIC16 mid-range core devices and PIC18 high-end core devices, both being the good representatives of Microchip's 8-bit microcontrollers. The former -type features a 14-bit wide code memory, an improved 8 level deep call stack and an increased op-code width allowing 128 registers and 2048 words of code to be directly addressed and are more often programmed in assembly. The latter type features a 16-bit wide op-codes (allowing many new instructions), and a 16 level deep call stack, and more importantly are programmable in C.

The lab workshop will represent a sample of model labs which we are teaching in the microcontrollers' laboratory course. Specifically these will include two labs based on a PIC18 high-end core microcontroller one of them programmed in C and the other in assembly and third lab based on the PIC16 mid-range core microcontroller programmed in PICBasic/PICBasic PRO.

But before we perform the lab workshops, we need to discuss some of the concepts, factors and phases of the lab performance task. These include how to develop a microcontroller-based application, tools needed for and how to use them. Next few paragraphs will briefly describe them.

2. Development of a Microcontroller-Based Application

Development of a microcontroller-based application (project) is a two-portion development process, consisting of: (a) a hardware development process and (b) a software development process. The two development processes are distinct from each other but are closely related to each other and therefore these must progress side-by-side [3].

The Hardware Development: Like development of any electronic engineering project, the hardware development of a microcontroller-based application is a multistep procedure. It, almost sequentially, involves: (a) determining a schematic circuit for a problem according to the

problem's requirements & specifications, (b) selecting and collecting a suitable microcontroller and all other electronic components (hardware) for the circuit, (c) building the circuit on **a** solder-less breadboard, (d) checking the connected circuit for its correctness of connections and (d) using the programmed microcontroller and finally power-on testing of the circuit.

The Software Development: The software development is basically a three-step process, namely: (a) To write a program (/ a source code) in assembly or C or PICBasic / PICBasic PRO; (b) To assemble / to compile (translate) a source code into an object code and; (c)To program (/ load the object code) onto a microcontroller.

(a) To write a program / code: This is the first step in which a program / code is written. At this stage we call the program / code as a source code. The source code is simply a sequence of instructions that is designed, for a selected microcontroller device, to do a job. Each type of programming langauge has its own set of such instructions which are readily available from various sources. In fact, instructions in assembly is freely loadable from Microchip's web site www.microchip.com and those in C and PICBasic / PICBasic PRO are available from the market on a nominal price. We will, for this lab workshop, write our source codes in each of assembly, C and PICBasic / PICBasicPRO.

(b) To assemble / to compile a source code: At this step a source program / code is converted (translated) into a code which is executable by a microcontroller to do an assigned job. A program / code after the successful completion of this step, is usually known as an object code. It is worth-mentioning here that: (a) if a source code / program is written in assembly, the source code needs an assembler software to assemble (convert) it to an executable object code and (b) if a source code is written in a high-level language, the source code needs a compiler software to compile (convert) the source code into an executable object code.

(c) To program the microcontroller: This is the final step in which the object program / code is loaded onto a microcontroller.

Software Development Tools: There are a number of software development tools available. To translate a source code into an object code, we will use Microchip's MPLAB IDE and its associated assembler software and commercially-available compiler software. So in this lab workshop, we will use both an assembler software and two compiler softwares. Microchip's MPLAB IDE is a complete development tool. Microchip's MPLAB IDE is, free of cost, loadable on line from Microchip's web site, namely, www.microchip.com. The MPLAB IDE provides an Integrated Development Environment and it also inherits an assembler software needed to convert a source code in assembly into an object code.

We will also use the PICBasic PRO Compiler software to compile source codes in PICBasic / PICBasic PRO and the Microchip's C18 C Compiler software to compile source codes in C. Each of the PICBasic PRO Compiler software and the Microchip's C18 C Compiler software is available, at nominal cost, in the market.

`Hardware Development Tools:` An object code, which is in an executable format, is, then, loaded onto a microcontroller. This step is performed using a programming carrier board and its associated software. We select Micro-Engineering Lab's Programming Carrier Board along with its associated software for this purpose.

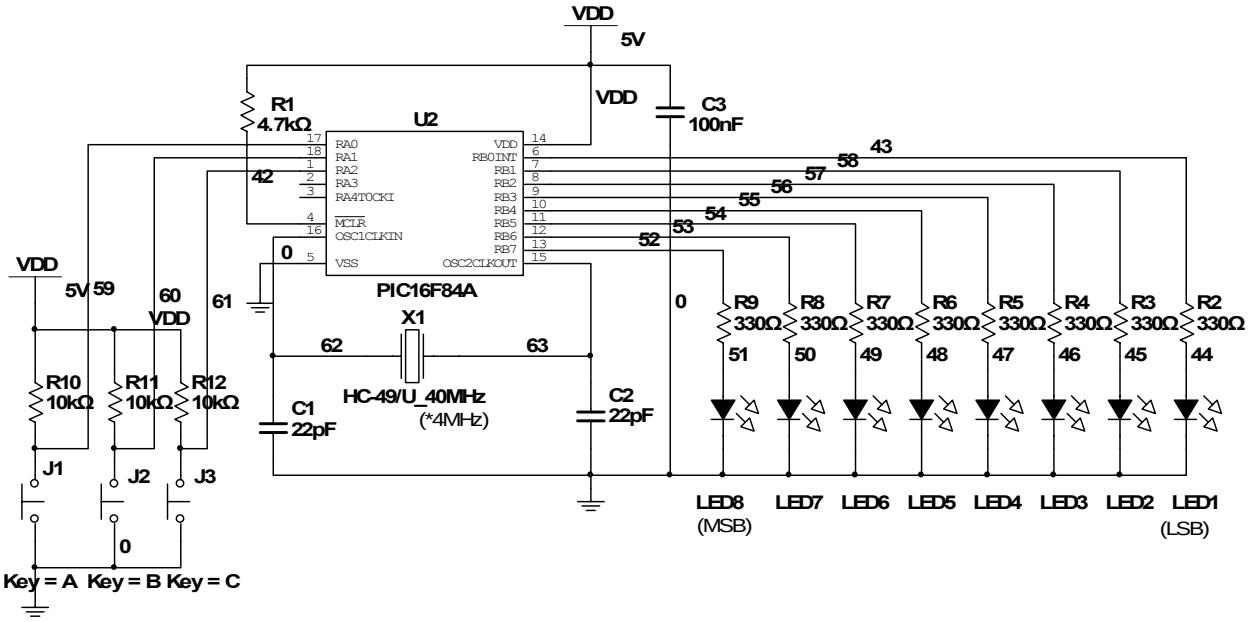
Laboratory Workshop One:

The Microchip's PIC16F84A Microcontroller-Based Project Programmed in PICBasic PRO.

Objectives:

Implement a binary counting (from binary 0000 0000 to binary 1111 1111) program that will light eight LEDs connected to PORTB's eight output lines.

Schematic Circuit



Procedure:

```
Step 1:  Writing of a Program.
B0  VAR  BYTE
delay  VAR  BYTE
delay = 500
TRISA = 15
TRISB = 0
loop:
For B0 = 0 TO 255
```

```
IF PORTA.0 = 0 Then hold
IF PORTA.1 = 0 Then fast
IF PORTA.2 = 0 Then slow
PORTB = B0
Pause delay
Next B0
GoTo loop
hold:
IF PORTA.0 = 0 Then hold
GoTo loop
fast:
IF PORTA.1 = 0 Then
delay = delay + 10
EndIF
IF delay > 1000 Then
delay = 1000
EndIF
GoTo loop
slow:
IF PORTA.2 = 0 Then
delay = delay - 10
EndIF
IF delay < 20 Then
delay = 20
EndIF
GoTo loop
End
```

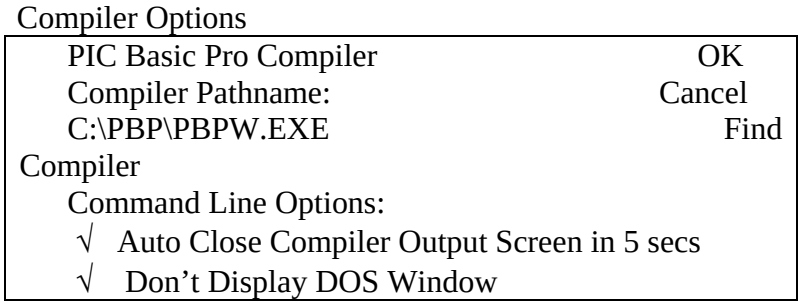
Step 2: Compiling of the Program

1) Software Installation: Install the PICBasicPRO Compiler. Execute SETUP.EXE on the PICBasicPRO Compiler disk and follow the instructions presented. Install all of the necessary files to a subdirectory named C:\PBP on the hard drive.
2) Open Codedesigner Lite (/Click on icon 'cdlite').
3) Open File > New. Start the IDE's editor. Select the microcontroller 16F84A from the IDE's drop-down list. Create the BASIC source file for the program (/Copy the program code from Step 1 and paste).
4) Save the code using File > Save As in C:\CDLite\Pro directory under the name seven_seg_1.pbp.
5) Go to Compile > Compiler Options. This opens a 'Compiler Options' window. Select / Set Compiler Options as shown in Figure 6.1.

Go to Compile > Compile (or press Function F5 key).

6) The successful compilation of the source code is shown on the Compiler Output window. The PICBasic PRO Compiler generates HEX (.HEX) files. The object code will be saved in C:\CDLite\Pro directory.

Figure 7.1



Step 3: Programming of the Microcontroller

We use micro-Engineering Labs Inc. s' the melabs U2 Programmer.

1) Plug the melabs U2 Prgrammer to the PC USB port using a USB cable.
2) Insert the PIC 16F84A microcontroller device on to the Programmer carrier board.
3) Launch the programmer software, i.e., click on the melabs Programmer icon to open melabs Programmer window.
4) Select the 16F84A microcontroller. Go to File > Open. Select the seven_seg_1.pbp program from the dialog box.

5) On the dialog box:
   a) Go to View > Configuration. This opens 'Open' window for the melab Prog-Configutation. On this window, Select/Set:
      i) Oscillator to XT.
      ii) Watchdog Timer to Enabled (Set ON)
      iii) Power-up Timer to Disabled
      iv) Code Protect to Off
   b) Go to Program > Erase to erase the chip.
   c) Go to Program > Program to load the object code on to the chip.
   d) Go to Program > Verify to verify the object code is loaded on to the chip. (Optional)

Step 4: Testing of the microcontroller-based project

Construct the circuit shown in the schematic on a solderless breadboard.

Insert the programmed PIC16F84A on to the circuit.

Power on and test the circuit.

Laboratory Workshop Two:

The Microchip's PIC18F452 Microcontroller-Based Project Programmed in C.

Objectives:
Implement a program in C to control and drive a PIC18F452 microcontroller-based two-way traffic light signal.

Schematic Circuit:

Same as shown in the Laboratory Workshop with PIC 18F452 instead of PIC16F84A.

Procedure:

Step 1: Writing of a Program

```
/*A program on C18 C compiler to run a TLC*/
#include <p18F452.h>
#pragma config WDT = OFF

void delay (void)
{
   unsigned int i;

for (i = 0; i < 50000; i++)
   ;
}


void main (void)
{

   TRISD = 0;
   PORTD = 0;

   while (1){

         PORTD = 0x41;
         delay ( );
         delay ( );
         PORTD = 0x21;
         delay ( );
         delay ( );
         PORTD = 0x11;
         delay ( );
         PORTD = 0x14;
```

```
            delay ( );
            delay ( );
            PORTD = 0x12;
            delay ( );
            delay ( );
            PORTD = 0x11;
            delay ( );

      }
   }
```

Step 2: Compiling of the Program

The compiling of the program is accomplished by using Microchip's MPLAB IDE and C18 C Compiler, and by sequentially following the steps as described below:-

1) Install MPLAB IDE software
   Using its CD for C18 Compiler Software**, i**nstall the C18 C Compiler software on your PC.

2) Start MPLAB IDE
   Select Start > Program > Microchip > MPLAB IDE from your monitor screen or
   a) Double click on the MPLAB IDE icon.

3) Configure the Project
   a) Go to the Configure > Settings menu and choose the Projects tab
   b) Ensure the setup to be as shown in Figure 3.1.
   c) Then, Click on OK.

Figure 3.1

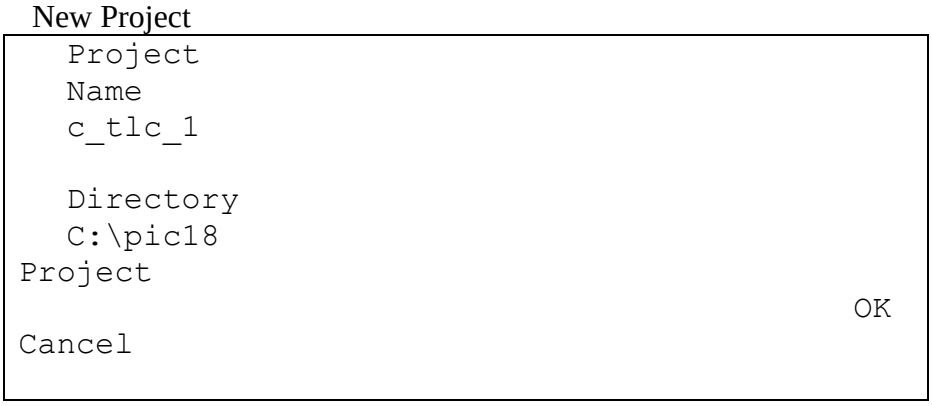
Setting

```
   √ Close open source files on project close
   √ Close output window before build
   √ Save project before build
   Save files before build

   •  Yes                  No
Prompt

   √ Halt build on first failure
   √ Use one-to-one project-work space model

                              Ok
Cancel          Apply
```

4) Select the **Target Device**
   a) Go to the Configure > Select Device
   b) Choose the target device as 18F452.
   c) Click on OK.

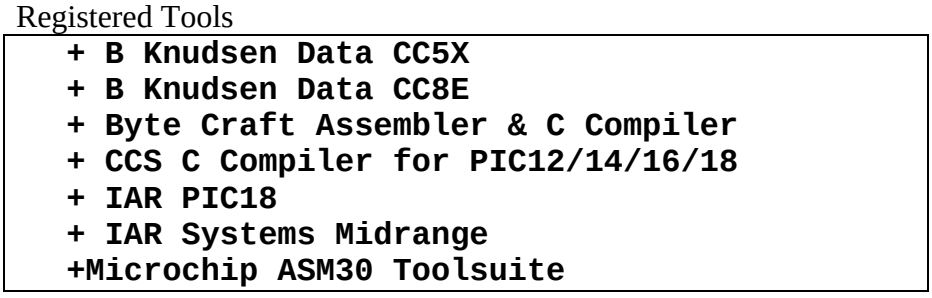
5) Create a New Project & Save it in the directory c:\pic18**.**
   a) Select the Project > New .
   b) The 'New Project' dialog, as shown in Figure 5.1, will open.
   c) Type the Project name as: c_tlc_1.
   d) Type the Project directory as 'C:\ pic18'. OR
   e) Click 'Browse' button.
      i) The 'Browse For Folder' dialog will open.
      ii) Scroll on to pic18 and click on pic18 and then click OK.
   f) Then Click on OK in the 'New Project' dialog.

Figure 5.1



6) Set Language Tool Location
   a) Select Project > Set Language Tool Locations.
   b) This will open 'Set Language Tool Locations' dialogue as shown in Figure 6.1.
   c) Click on 'Microchip C18 Toolsuite'
   d) Click on 'Executables'
   e**)** Click / Highlight on MPLAB C18C Compiler (mcc18.exe)
   f) Click OK

Figure 6.1



```
   +Microchip C17 Toolsuite
   + Microchip C18 Toolsuite
       - Executables
              MPASM Assembler (mpasmwin.exe)
              MPLAB C18 C Compiler (mcc18.exe)
              MPLIB Librarian (mplib.exe)
              MPLINK Object Linker (mplink.exe)
   Location
   C:\MCC18|bin\mcc18.exe
Browse

   Help
OK                Cancel                Apply
```

7) Set Language Toolsuite
   Select Project > Set Language Toolsuite.
   a) This will open 'Select Language Toolsuite' dialog as shown in Figure 7.1.
   b) Highlight MPLAB C18 C Compiler (mcc18.exe),
   c) Thus, highlighting C:\MCC18\bin\mcc18.exe in 'Location'.
   d) Click on OK

Figure 7.1

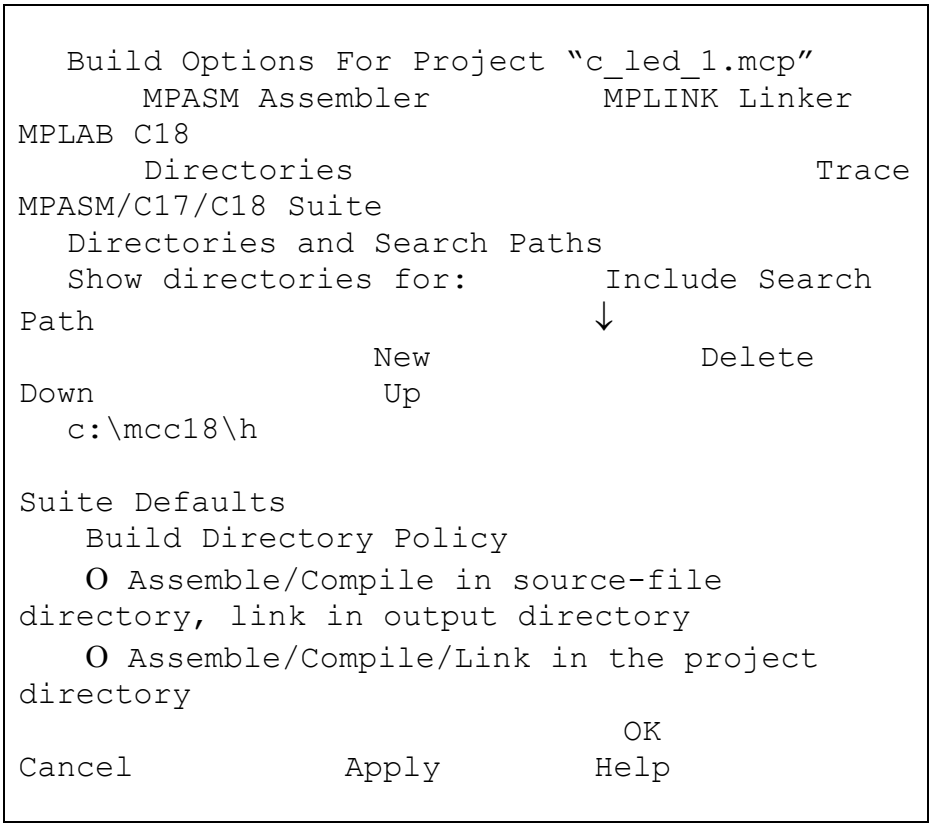
Select Language Toolsuite

```
    Active Toolsuite
Microchip C18 Toolsuite
   Toolsuite Contents
OK
   MPASM Assembler(mpasmwin.exe)
Cancel
   MPLINK Object Linker(mplink.exe)
Apply
    MPLABC18C Compiler (mcc18.exe)
   MPLIB Librarian (mplib.exe)

   Location
    C:\MCC18\bin\mcc18.exe
Browse

          Help
OK                  Cancel
```

8) Enter Source Code
   a) Select File > New
   b) This opens a blank Edit window 'Untitled'.
   c) Enter the program.
   d) After entering the source code, select File > Save
   e) Save the file in the project directory as 'c_tlc_1.c'

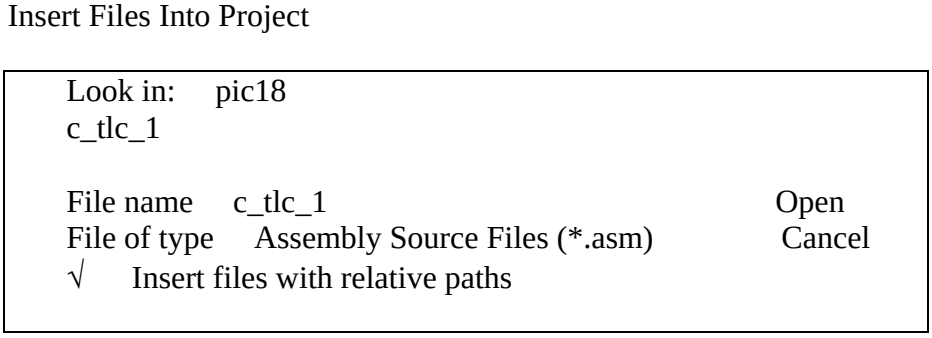
9) Setting Build Options
   a) Select Project>Build Options: this opens Build Options For Project "c_led_1.mcp' dialog window as in Figure 9.1.
   b) Click on Directories Go to Output Directory and click on Include Search Path. Click on 'New'. Type 'c:\mcc18\h' and click on Apply.
   c) Again go to Output Directory and click on Library Search Path. Click 'New'. Type 'c:\mcc18\lib' and click on Apply.
   d) Again go to Output Directory and click on Linker-Script Search Path. Click on 'New'. Type 'c:\mcc18\lkr' and click on Apply.
   e) Click OK.

Figure 9.1



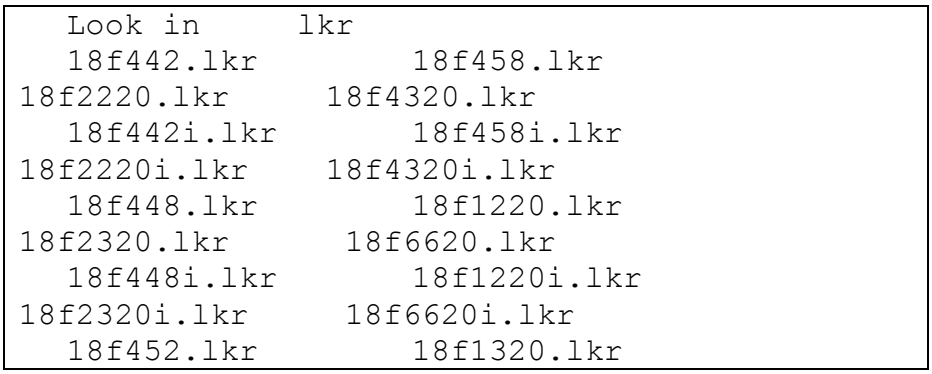
10) Add Source File to the Project
   a) Select Project > Add files to Project
   b) A dialog window 'Insert Files Into Project' will open as in Figure 10.1.
   c) Select Look in pic 18.
   d) Highlight c_tlc_1.c
   e) Click Open.
   f) The newly inserted file will appear in the 'project pane'.
   g) Save the project c_led_1.c by selecting Project > Save.

Figure 10.1



11) Add the Linker File to the Project
   a) Select Project > Add files to Project
   b) A dialog window 'Insert Files Into Project' will open as in Figure 11.1.
   c) Browse the MCC18C compiler installation directory (under c:\mcc18\lkr) and add the appropriate linker file.
   d) Select Look in: lkr.
   e) Select 18f452.lkr (for the device into this project)
   f) Click on Open.

Figure 11.1



```
18f4220.lkr        18f6720.lkr
   18f452i.lkr        18f1320i.lkr
18f4220i.lkr       18f6720i.lkr

   ←
→

   File name :    18f452
Open
   File of type:     Linker Scripts (*.lkr)
Cancel
   □√      Insert files with relative paths
```

12) Build the Project

This step will compile the source code and translate it onto an executable code (object code).

a) Select Project > Build All or Project > Make (or Press Function key F10).
b) On successful build, the debug files (with the file name extension .cod or .cof) will be generated and loaded.
c) On the OUTPUT window, BUILD SUCCEEDED message will appear.

Step 3: Programming The PIC18F452 Microcontroller

We use micro-Engineering Labs Inc. s' the melabs U2 Programmer.

6) Plug the melabs U2 Prgrammer to the PC USB port using a USB cable.
7) Insert the PIC 18F452 microcontroller device on to the Programmer carrier board.
8) Launch the programmer software, i.e., click on the melabs Programmer icon to open melabs Programmer window.
9) Select the 18F452 microcontroller. Go to File > Open C:\pic18 directory. Select the c_ltc_1.c program from the dialog box.

10) On the dialog box:

a) Go to View > Configuration. This opens 'Open' window for the melab Prog-Configutation. On this window, Select/Set:
   i) Oscillator to XT.
   ii) Watchdog Timer to Enabled (Set ON)
   iii) Power-up Timer to Disabled
   iv) Code Protect to Off
b) Go to Program > Erase to erase the chip.
c) Go to Program > Program to load the object code on to the chip.
d) Go to Program > Verify to verify the object code is loaded on to the chip.

Step 4: Testing the Circuit

Construct the circuit shown in the schematic on a solderless breadboard.

Insert the programmed PIC18F452 on to the circuit.

Power on and test the circuit.

Insert the programmed PIC18F452 microcontroller on to the circuit and test it.

References

[1] Muhammad M Baig & Rafiqul Islam: Problems And Intended Solutions In Teaching PIC Microcontroller In EET Program; Publisher: ASEE Mid-Atlantic Conference Fall 2009.

[2] Han-Way Huang, *PIC Microcontroller: An Introduction to Software & Hardware Interfacing*, Publisher: Thomson Delmar Learning, (2005).

[3] Myke Predko, *Programming And Customizing The Pic Microcontroller*, publisher: McGraw-Hill, (2002).

[4] John Iovine, *PIC Microcontroller Project Book* , publisher: McGraw-Hill/TAB Electronics, 2005.

Microchip's web site: *www:microchip.com*

[5] Myke Predko, *Handbook of Microcontrollers*, publisher: McGraw-Hill, (2004).

[6] Microchip, *MPLALB C18 C Compiler User's Guide, 2005.*

[7] MicroEng, *PICBasic PRO Compiler User's Book*